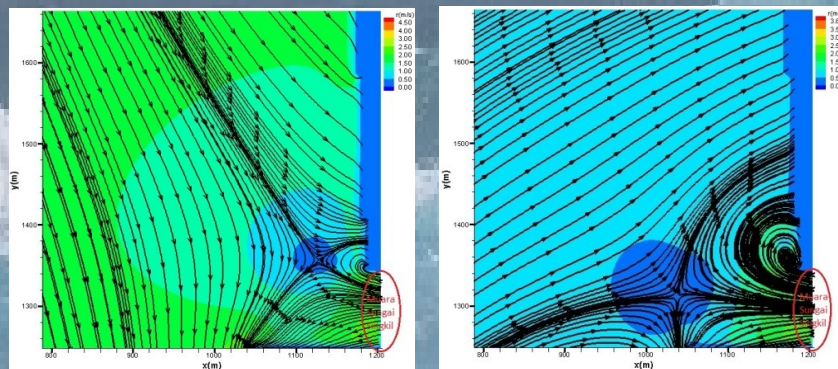
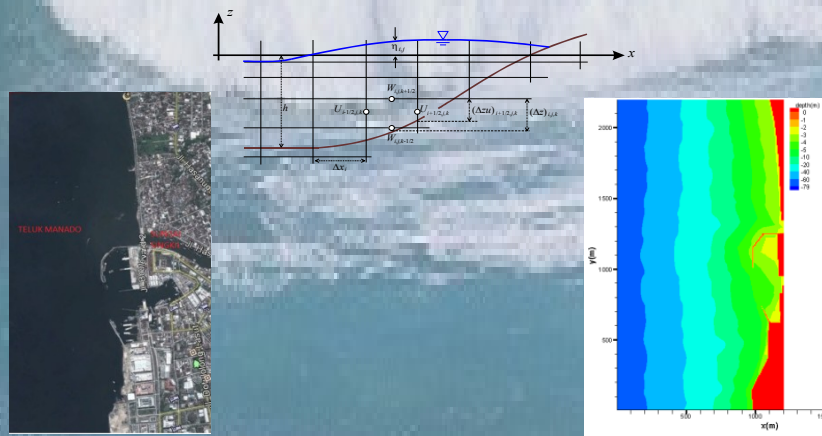


SEBUAH MODEL NUMERIK ARUS LAUT DAN SUNGAI

*DI TELUK MANADO,
PROPINSI SULAWESI UTARA, INDONESIA*



Parabelem T.D. Rompas



PENERBIT UNIMA PRESS

**SEBUAH MODEL NUMERIK ARUS LAUT DAN SUNGAI
DI TELUK MANADO, PROPINSI SULAWESI UTARA, INDONESIA**

Penulis :

Parabelem T. D. Rompas

ISBN : 978-602-1376-17-1

Editor :

Verry Ronny Palilingan

Penyunting :

Ichdar Domu

Desain sampul dan Tata letak :

Johan Reimon Batmetan

Penerbit :

Unima Press

Redaksi :

Jl. Kampus Unima Tondano 95618,
Minahasa, Sulawesi Utara, Indonesia
Tel +620431321845, +6281243226311
Fax +620431321866
Email : verryronnypalilingan@unima.ac.id

Distribusi Tunggal :

UPT Percetakan Universitas Negeri Manado
Jl. Kampus Unima Tondano 95618,
Minahasa, Sulawesi Utara, Indonesia
Tel +620431321845, +6281243226311
Fax +620431321866
Email : verryronnypalilingan@unima.ac.id

Cetakan pertama, Januari 2015

Hak cipta dilindungi undang-undang

Dilarang memperbanyak karya tulis ini dalam bentuk dan dengan cara apapun
tanpa ijin dari penerbit

PRAKATA

Puji syukur kepada Tuhan Yang Maha Esa karena bimbinganNya maka laporan kemajuan ini dapat terselesaikan. Tujuan buku referensi ini adalah pertama, untuk mendapatkan referensi hasil pengembangan metode semi-implisit tiga dimensi untuk aliran air dangkal dari Casulli dan Chen (1992) menjadi model numerik baru yang mana hasil model simulasi numerik dari model numerik baru berupa simulasi distribusi kecepatan arus laut dan sungai di teluk Manado Propinsi Sulawesi Utara, Indonesia yang nantinya akan dipakai sebagai sumber data dalam perancangan pembuatan dermaga lokal di sekitar muara sungai; kedua, untuk bahan referensi bagi mahasiswa dan pembaca dalam bidang mekanika fluida dan konversi energi.

Pada kesempatan ini kami penulis menyampaikan terima kasih yang sebesar-besarnya kepada semua pihak yang telah membantu dalam penulisan buku ini dan lebih khusus kepada:

1. Pemberi dana yaitu pemerintah Indonesia dalam hal ini Kementerian Pendidikan dan Kebudayaan Republik Indonesia melalui dana DIPA Ditlitabmas Dirjendikti Jakarta.
2. Rektor, Dekan Fatek, Ketua Jurusan dan Kepala Laboratorium serta Dosen-dosen Jurusan PTM pada Universitas Negeri Manado di Tondano yang telah memfasilitasi dan memberi kesempatan kepada kami penulis untuk membuat buku ini dengan tujuan untuk meningkatkan profesionalisme dosen dalam mendidik dan mengajar di UNIMA melalui kegiatan penulisan buku referensi.

Besar harapan kami kepada semua pihak kiranya dapat memberikan kritik dan saran guna penyempurnaan buku ini.

Tondano, Januari 2015
Penulis,

DAFTAR ISI

HALAMAN JUDUL	2
PRAKATA.....	3
DAFTAR ISI.....	5
DAFTAR GAMBAR.....	7
DAFTAR TABEL.....	9
DAFTAR LAMPIRAN.....	11
BAB 1. PENDAHULUAN	13
A. Latar Belakang.....	13
B. Tujuan Penelitian.....	16
C. Manfaat Penelitian.....	16
BAB 2. GAMBARAN SELAT MANADO DAN DOMAIN KAJIAN	19
A. Letak geografi.....	19
B. Domain kajian.....	20
BAB 3. MODEL MATEMATIKA.....	23
A. Persamaan Navier-Stokes	23
1. Parameter Coriolis	23
2. Difusi efektif.....	24
B. Kondisi batas	24
1. Kondisi-kondisi batas pada permukaan dan dasar laut. 24	
2. Kondisi-kondisi batas pada dinding dan saluran keluar system.	25
BAB 4. MODEL NUMERIK	27
A. Langkah adveksi	27

B.	Langkah difusi	28
C.	Langkah tekanan kontinuitas	31
D.	Energi kinetik.....	32
BAB 5.	METODE.....	33
A.	Data set	34
B.	Pembuatan grid	35
C.	Pembuatan indeks	35
D.	Kondisi awal	37
E.	Cetak hasil kalkulasi	38
F.	Validasi hasil penelitian.....	38
BAB 6.	HASIL DAN PEMBAHASAN	39
A.	Sebuah model numerik	39
B.	Model program numerik.....	40
1.	Program utama.....	40
2.	Program subrutin	50
C.	Prediksi pasang surut air laut di teluk Manado.....	169
D.	Uji coba simulasi 2D distribusi kecepatan arus laut dan sungai di teluk Manado.....	169
E.	Hasil simulasi numerik 2D dan 3D di teluk Manado	174
1.	Hasil-hasil simulasi numerik 2D	175
2.	Hasil-hasil simulasi numerik 3D	180
BAB 7.	KESIMPULAN.....	185
DAFTAR PUSTAKA	187
LAMPIRAN.....		191

DAFTAR GAMBAR

Gambar 1. Pengembangan produksi energi listrik di tahun 2025 di Indonesia	14
Gambar 2. Roadmap listrik arus laut di Indonesia.....	15
Gambar 3. Peta teluk Manado dan lokasi numerik	20
Gambar 4. Skema diagram mesh dan notasi komputasional	29
Gambar 5. Diagram alir penelitian.....	33
Gambar 6. Gambar alir kalkulasi numerik.....	36
Gambar 7. Peta teluk Manado dan sungai Singkil	37
Gambar 8. Prediksi pasang surut air laut di teluk Manado	169
Gambar 9. Bathymetry lokasi numerik.....	171
Gambar 10. Garis-garis aliran air pada kondisi arus pasang dan 1 m kolom air laut	171
Gambar 11. Garis-garis aliran air pada kondisi arus surut dan 1 m kolom air laut	172
Gambar 12. Distribusi streamline arus laut pada 1 m kolom air	175
Gambar 13. Distribusi kecepatan arus laut pada 1 m kolom air	176
Gambar 14. Distribusi gelombang air pada 1 m kolom air.....	177
Gambar 15. Distribusi energi kinetik pada 1 m kolom air.....	178
Gambar 16. Distribusi streamline arus laut pada 1 m kolom air	178
Gambar 17. Distribusi kecepatan arus laut pada 1 m kolom air	179
Gambar 18. Distribusi gelombang air pada 1 m kolom air.....	180
Gambar 19. Distribusi energi kinetik pada 1 m kolom air.....	180
Gambar 20. Distribusi kecepatan arus laut pada 1 m kolom air	181
Gambar 21. Distribusi gelombang air pada 1 m kolom air.....	181
Gambar 22. Distribusi energi kinetik pada 1 m kolom air.....	182

Gambar 23. Distribusi kecepatan arus laut pada 1 m kolom air	182
Gambar 24. Distribusi gelombang air pada 1 m kolom air.....	183
Gambar 25. Distribusi energi kinetik pada 1 m kolom air.....	183

DAFTAR TABEL

Tabel 1. Parameter numerik untuk uji coba simulasi 2D	170
Tabel 2. Parameter numerik untuk simulasi 2D	174
Tabel 3. Parameter numerik untuk simulasi 3D	174

DAFTAR LAMPIRAN

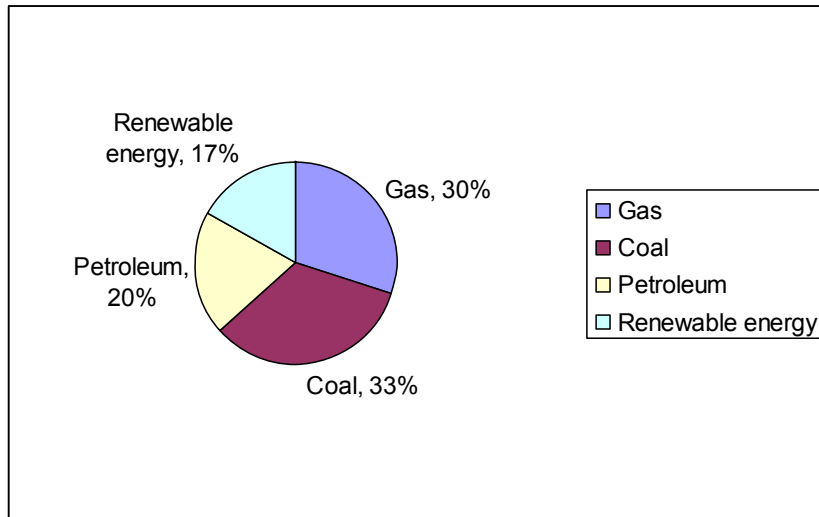
Lampiran A: Validasi hasil kalkulasi numerik	193
Lampiran B: Artikel ilmiah nasional yang dimuat dalam prosiding	195

BAB 1. PENDAHULUAN

A. Latar Belakang

Kekurangan pasokan listrik di daerah perdesaan sangat mungkin terjadi karena jauh dari perkotaan dan jaringan listrik, namun tidak menutup kemungkinan daerah perkotaan juga mengalami hal yang sama seperti kota Manado dan sekitarnya. Kenyataannya kota Manado dan sekitarnya sering terjadi pemadaman listrik secara bergilir karena kekurangan pasokan energi listrik.

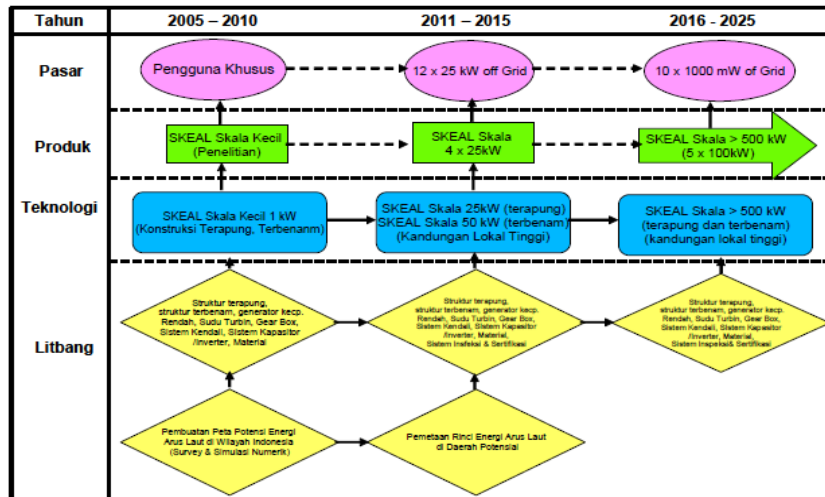
Untuk mengantisipasi masalah-masalah itu, maka potensial tenaga arus hasil pertemuan arus laut dan arus sungai di Teluk Manado Propinsi Sulawesi Utara, Indonesia diperkirakan akan menjadi suatu solusi dari berbagai permasalahan itu dengan dibangunnya pembangkit listrik tenaga arus laut dan sebagai penggerak mula generator listrik adalah turbin air laut dan itu sesuai dengan rencana pemerintah pada tahun 2013 dan 2016 dalam penelitian dan pengembangan (lihat gambar 2). Rancangan turbin itu harus efisien dan efektif yang mana dalam pembuatannya memerlukan data-data kecepatan arus dan energi kinetik yang tersedia (BC Hydro, 2002). Data-data itu dapat dikaji melalui penelitian tentang simulasi numerik yang secara teori dapat menentukan besarnya energi kinetik yang tersedia di suatu daerah yang airnya dangkal (Backhaus, 1983; Stelling, 1984; Casulli, 1990; Casulli dan Cheng, 1992; Stansby, 1997; Casulli dan Walters, 2000; Broomans, 2003).



Gambar 1. Pengembangan produksi energi listrik di tahun 2025 di Indonesia

(sumber: Peraturan Presiden No. 5 Tahun 2006 tentang Kebijakan Energi Nasional Indonesia)

Kajian simulasi numerik energi arus laut sudah sesuai dengan roadmap dari pemerintah Indonesia dalam bidang penelitian dan pengembangan seperti terlihat pada gambar 2. Pada periode jangka menengah dan jangka panjang, pemerintah telah merencanakan peran pemerintah melakukan penelitian dan pengembangan (R&D) yaitu mulai dari melaksanakan pemetaan energi arus laut di daerah yang berpotensi sampai pada melaksanakan penelitian dan pengembangan sektor energi arus laut skala menengah dalam jumlah banyak (turbin arus laut model *tidal farm*). Begitu pula dalam peluang pasar yang mana pada tahun 2014 peran pemerintah melanjutkan pemetaan potensi energi arus laut. Gambar 1 menunjukkan bahwa pengembangan energi terbarukan di tahun 2025 di Indonesia sebesar 17%.



Gambar 2. Roadmap listrik arus laut di Indonesia (sumber: BUKU PUTIH, Indonesia 2005-2025, Penelitian, Pengembangan dan Penerapan Ilmu Pengetahuan dan Teknologi Bidang Sumber Energi Baru dan Terbarukan untuk Mendukung Keamanan Ketersediaan Energi Tahun 2025, Kementerian Negara Riset dan Teknologi Republik Indonesia, Jakarta: 2006)

Selain itu kajian hasil pertemuan arus laut dan arus sungai dapat dijadikan dasar kajian dalam pembangunan dermaga lokal untuk keperluan transportasi kapal laut. Dalam pembangunan dermaga itu sangat dibutuhkan data-data kecepatan arus laut dan arus sungai untuk menghindari terjadinya kecelakaan kapal saat masuk dan keluar dermaga. Hasil temuan menunjukkan bahwa metode numerik dari Casulli dan Cheng (1992) melalui persamaan matematik (1), (2), dan (3) perlu dikembangkan menjadi model matematika dan numerik baru untuk menghasilkan simulasi numerik (secara teori) yang lebih mendekati pada keadaan yang sebenarnya tentang pemetaan distribusi kecepatan arus dan potensi energi arus laut yang tersedia di teluk Manado Propinsi Sulawesi Utara, Indonesia dan nantinya hasil itu akan dipakai sebagai sumber data dalam perencanaan pembangunan dermaga lokal dan turbin

pembangkit listrik arus laut. Persamaan itu dapat ditulis sebagai berikut:

$$\frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} + v \frac{\partial u}{\partial y} + w \frac{\partial u}{\partial z} = -g \frac{\partial \eta}{\partial x} + \mu \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right) + \frac{\partial}{\partial z} \left(\nu_i \frac{\partial u}{\partial z} \right) + f.v \quad (1)$$

$$\frac{\partial v}{\partial t} + u \frac{\partial v}{\partial x} + v \frac{\partial v}{\partial y} + w \frac{\partial v}{\partial z} = -g \frac{\partial \eta}{\partial y} + \mu \left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} \right) + \frac{\partial}{\partial z} \left(\nu_i \frac{\partial v}{\partial z} \right) + f.u \quad (2)$$

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} = 0 \quad (3)$$

dimana u, v , dan w adalah masing-masing komponen kecepatan dalam arah x, y , dan z ; ρ adalah densitas dan ρ_0 densitas referensi, p adalah tekanan, ν_N viskositas kinematik, g adalah kecepatan gravitasi konstan dan f_x, f_y , dan f_z adalah masing-masing gaya coriolis per satuan massa.

B. Tujuan Penelitian

Tujuan khusus dalam penelitian ini adalah untuk mendapatkan hasil pengembangan metode semi-implisit tiga dimensi untuk aliran air dangkal dari Casulli dan Chen (1992) menjadi model numerik baru.

Tujuan umum dalam penelitian ini adalah untuk mendapatkan model simulasi numerik dari model numerik baru berupa simulasi distribusi kecepatan arus laut dan energi kinetik di teluk Manado Propinsi Sulawesi Utara, Indonesia yang nantinya akan dipakai sebagai sumber data dalam perancangan turbin pembangkit listrik arus laut.

C. Manfaat Penelitian

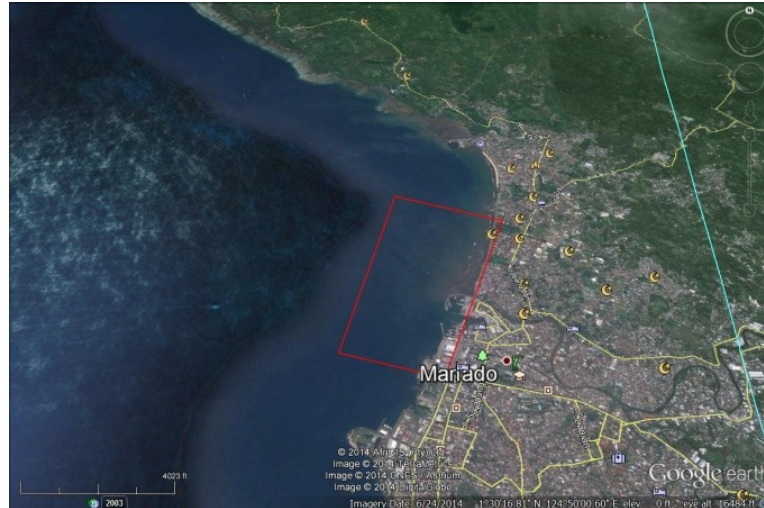
Penelitian ini bermanfaat bagi pemerintah daerah kota Manado dan propinsi Sulawesi Utara sebagai bahan informasi dan bahan kajian dalam perencanaan pembangunan pembangkit listrik arus laut di teluk Manado guna pemenuhan kebutuhan energi listrik melalui energi baru terbarukan dan bagi masyarakat sekitar teluk Manado jika terealisasi pembangunan pembangkit listrik arus

lalu maka akan mendapatkan keuntungan dari segi pekerjaan dan prioritas penyaluran aliran listrik guna pemenuhan kebutuhan listrik keluarga.

BAB 2. GAMBARAN SELAT MANADO DAN DOMAIN KAJIAN

A. Letak geografi

Letak geografis teluk Manado berada pada $1^{\circ}30'60.81''$ N dan $124^{\circ}50'00.60''$ E. Di kota Manado terdapat satu dermaga transportasi lokal dengan tujuan ke kabupaten-kabupaten Sitaro, Sangir dan Talaud yang letaknya di bagian utara propinsi Sulawesi Utara. Juga, terdapat beberapa dermaga kecil untuk kapal-kapal sebagai transportasi menuju taman laut Bunaken di pulau Bunaken. Semuanya itu terletak berdekatan dengan muara sungai Singkil yang berasal dari danau Tondano kabupaten Minahasa (lihat Gambar 3 dan Gambar 5). Pada bagian keluar sungai itu bertemu dengan laut (teluk Manado) yang mana pada lokasi ini terjadi pertemuan dua arus yaitu arus sungai dan arus laut. Biasanya arus teluk Manado terjadi dua kali pasang dan dua kali surut yang mana pertama; pada pagi sampai siang terjadi arus surut (sekitar 0,5 m) dan pada siang sampai sore menjelang malam terjadi arus pasang (sekitar 0,5 m), dan kedua; pada sore menjelang tengah malam terjadi lagi arus surut dan pada tengah malam sampai pagi hari terjadi arus pasang (pasang dan surut sekitar masing-masing 0,5 m).



Gambar 3. Peta teluk Manado dan lokasi numerik

Sampai saat ini telah dibangun beberapa dermaga kecil di sekitar muara sungai Singkil untuk tempat kapal-kapal kecil sebagai transportasi kapal pesiar bagi turis-turis. Juga, di sungai Singkil sudah lama sebagai tempat berlabuh kapal-kapal kecil sebagai transportasi lokal bagi masyarakat yang menuju ke pulau-pulau Manadotua, Bunaken, Mantehang, dan Naeng-besar. Suatu saat nanti akan nada lagi tambahan pembangunan beberapa dermaga kecil di sekitar sungai Singkil bagian Utara, sehingga pemerintah harus cepat mengantisipasi pembangunan itu agar terhindar dari situasi yang buruk seperti terhindar dari ancaman arus laut.

B. Domain kajian

Hasil pertemuan arus laut dan arus sungai dapat dijadikan dasar kajian dalam pembangunan dermaga lokal untuk keperluan transportasi kapal laut di kota Manado. Pembangunan dermaga sangat membutuhkan data-data kecepatan arus-arus laut dan sungai untuk menghindari terjadinya kecelakaan kapal saat masuk dan keluar dermaga. Kajian kecepatan arus-arus itu dalam prakteknya sangat membutuhkan seperti peralatan-peralatan yang sangat mahal, waktu yang dibutuhkan sangat

lama, dan biaya yang diperlukan sangat mahal. Salah satu contoh dalam pengukuran kecepatan arus-arus laut membutuhkan kapal laut, alat pengukur kecepatan arus hydrometer yang banyak tergantung ada berapa titik-titik pengukuran yang diperlukan, tenaga kerja yang banyak, waktu yang diperlukan untuk beberapa titik-titik pengukuran, dan biaya yang dikeluarkan untuk sewa dan pembelian alat termasuk biaya tenaga kerja.

Untuk efisiensi dan efektifitas pengukuran kecepatan arus-arus laut dan sungai di selat Manado maka digunakan sebuah model yaitu model numerik yang mana model ini sangat efisien dan efektif dalam penggunaannya karena menghemat dalam misalnya penggunaan alat, tenaga kerja, waktu, dan biaya. Hasil yang didapat dari model numerik juga sangat efektif karena nilai-nilainya sangat mendekati nilai observasi langsung.

Peralatan yang digunakan dalam penggunaan model numerik seperti seperangkat komputer, printer, mesin pemindai, perangkat lunak (autocad 2014, fortran 90, argusone, dan tecplot 10), dan alat ukur kecepatan arus sederhana yang dibuat sendiri. Dari segi tenaga kerja hanya dibutuhkan 10 orang yaitu 4 orang dalam pemrograman komputer dan 6 orang dalam validasi hasil kalkulasi numerik.

BAB 3. MODEL MATEMATIKA

A. Persamaan Navier-Stokes

Sebelum mendapatkan persamaan-persamaan model numerik sebagai dasar dalam pemodelan, maka dibuat dahulu persamaan-persamaan model matematika. Pers. 1, 2, dan 3 dikembangkan menjadi persamaan Navier-Stokes rata-rata Reynolds yang diasumsikan di bawah tekanan hidrostatik:

$$\frac{\partial \bar{u}}{\partial t} + \bar{u} \frac{\partial \bar{u}}{\partial x} + \bar{v} \frac{\partial \bar{u}}{\partial y} + \bar{w} \frac{\partial \bar{u}}{\partial z} = -g \frac{\partial \eta}{\partial x} + \text{div} \left(v_{\text{eff}} \overrightarrow{\text{grad}}(\bar{u}) \right) + f_{\text{cor}} \bar{v} \quad (4)$$

$$\frac{\partial \bar{v}}{\partial t} + \bar{u} \frac{\partial \bar{v}}{\partial x} + \bar{v} \frac{\partial \bar{v}}{\partial y} + \bar{w} \frac{\partial \bar{v}}{\partial z} = -g \frac{\partial \eta}{\partial y} + \text{div} \left(v_{\text{eff}} \overrightarrow{\text{grad}}(\bar{v}) \right) + f_{\text{cor}} \bar{u} \quad (5)$$

$$\frac{\partial \bar{u}}{\partial x} + \frac{\partial \bar{v}}{\partial y} + \frac{\partial \bar{w}}{\partial z} = 0 \quad (6)$$

$$\frac{\partial \eta}{\partial t} + \frac{\partial}{\partial x} \left(\int_{-h}^{\eta} \bar{u} dz \right) + \frac{\partial}{\partial y} \left(\int_{-h}^{\eta} \bar{v} dz \right) = 0 \quad (7)$$

dimana $\bar{u}(x,y,z,t)$, $\bar{v}(x,y,z,t)$ dan $\bar{w}(x,y,z,t)$ adalah masing-masing komponen kecepatan dalam arah x,y horizontal, dan z vertikal, t adalah waktu, $\eta(x,y,t)$ adalah elevasi permukaan air, g is percepatan gravitasi, f_{cor} adalah parameter Coriolis. v_{eff} adalah sebuah difusi efektif.

1. Parameter Coriolis

Parameter Coriolis diasumsikan konstan dan persamaannya adalah:

$$f_{\text{cor}} = 2\omega \sin(\varphi) \quad (8)$$

dimana ω adalah kecepatan sudut dari bumi ($\omega = 7.29212 \times 10^{-5} \text{ s}^{-1}$) dan ϕ adalah latitut. Pengaruh gaya Coriolis di dalam persamaan Navier-Stokes diatur melalui bilangan Rossby:

$$R_0 = \frac{\bar{u}_{ref}}{f_{cor}l} \quad (9)$$

where \bar{u}_{ref} and l adalah masing-masing kecepatan referensi dan panjang dari aliran. Itu penting hanya jika $R_0 \leq 1$.

2. Difusi efektif

Difusi efektif adalah jumlah dari viskositas kinematik turbulen (ν_t) dan viskositas kinematik molekuler (ν_m). Viskositas kinematik turbulen digunakan formulasi kedalaman yang dirata-ratakan dari Stansby (2003) yaitu:

$$\nu_t = \left(l_h^4 \left[2 \left(\frac{\partial \bar{u}}{\partial x} \right)^2 + 2 \left(\frac{\partial \bar{v}}{\partial y} \right)^2 + \left(\frac{\partial \bar{v}}{\partial x} + \frac{\partial \bar{u}}{\partial y} \right)^2 \right] + (\gamma \bar{u}_f h)^2 \right)^{1/2} \quad (10)$$

B. Kondisi batas

Sejauh ini daerah teluk Manado bentuknya lebih kompleks untuk aliran permukaan bebas. Itu dibatasi:

1. Kondisi-kondisi batas pada permukaan dan dasar laut.

Pada permukaan laut, persamaan yang digunakan adalah:

$$\nu_t \frac{\partial \bar{u}}{\partial z} = C_D \rho_{udara} W_{10x} \|W_{,10x}\| \quad (11)$$

$$\nu_t \frac{\partial \bar{v}}{\partial z} = C_D \rho_{udara} W_{10y} \|W_{,10y}\| \quad (12)$$

dimana $C_D = (0.75 + 0.067W_{10})10^{-3}$ adalah sebuah koefisien gesek dari formula Garratt (1977) yang merupakan fungsi dari

kecepatan angin. $\rho_{\text{udara}} = 1.178 \text{ kg/m}^3$ (pada 27 C dan 1 bar) dan kecepatan angin 10 m di atas permukaan laut pada arah x dan y adalah $W_{10,x}$ dan $W_{10,y}$.

Pada dasar laut, persamaan tegangan dasar laut bisa dihubungkan kepada hukum turbulen dinding dasar laut, sebuah koefisien gesek yang diasosiasikan dengan sebuah formula Chezy. Persamaan itu menjadi:

$$v_t \frac{\partial \bar{u}}{\partial z} = - \frac{g \sqrt{(\bar{u}^2 + \bar{v}^2)}}{C_z^2} \bar{u} \quad (13)$$

$$v_t \frac{\partial \bar{v}}{\partial z} = - \frac{g \sqrt{(\bar{u}^2 + \bar{v}^2)}}{C_z^2} \bar{v} \quad (14)$$

dimana C_z adalah koefisien Chezy.

2. Kondisi-kondisi batas pada dinding dan saluran keluar system.

Pada dinding, tegangan sama dengan nol, sehingga kecepatan-kecepatan dalam semua arah pada dinding sama dengan nol.

Pada saluran keluar sistem, ada dua metode yang digunakan yaitu metode von Neumann dan pengembangan kondisi batas Adaptatif. Persamaan-persamaan itu adalah:

Metode pertama,

$$\frac{\partial \varphi}{\partial n} = 0 \quad (15)$$

Metode kedua,

jika ($C_{\varphi x} > 0$) maka;

$$\frac{\partial \varphi}{\partial t} = -C_{\varphi x} \frac{\partial \varphi}{\partial x} + \frac{(\varphi_c - \varphi)}{\tau_o} \quad (16)$$

jika ($C_{\varphi x} \leq 0$) maka;

$$\frac{\partial \varphi}{\partial t} = \frac{(\varphi_c - \varphi)}{\tau_i} \quad (17)$$

dimana $C_{\varphi x}$ dan $C_{\varphi y}$ kecepatan-kecepatan fase pada arah x dan y ke batas di dalam koordinat kartesius local. φ adalah variabel yang menjadi perlakuan untuk kondisi batas yang ditentukan. φ_c adalah sebuah perkiraan klimatologi atau observasi dari φ pada batas di saluran keluar. τ_0 dan τ_i adalah skala waktu relaksasi pada kondisi masuk dan keluar sistem.

BAB 4. MODEL NUMERIK

Persamaan-persamaan model numerik untuk menghitung kecepatan-kecepatan dalam arah x, y, dan z diturunkan dari persamaan-persamaan model matematika. Penyelesaian numerik tiga dimensi digunakan metode beda hingga semi implisit. Ada tiga langkah dalam penyelesaian model numerik dari persamaan-persamaan model matematika Pers. 1, Pers. 2, Pers. 3, dan Pers. 4 yaitu sebagai berikut:

A. Langkah adveksi

Langkah adveksi menggunakan diskretisasi *Eulerian-Lagrangian* dari hubungan konveksi dan viskos. Persamaannya adalah sebagai berikut:

$$\begin{aligned}
 C_{i-a,j-b,k-d}^n &= (1-r)\{(1-p) \\
 &[(1-q)C_{i-l,j-m,k-n}^n + qC_{i-l,j-m,k-n}^n] \\
 &+ p[(1-q)C_{i-l-1,j-m,k-n}^n + qC_{i-l-1,j-m-1,k-n}^n]\} \\
 &+ r\{(1-p)[(1-q)C_{i-l,j-m,k-n-1}^n + qC_{i-l,j-m-1,k-n-1}^n] \\
 &+ p[(1-q)C_{i-l-1,j-m,k-n-1}^n + qC_{i-l-1,j-m-1,k-n-1}^n]\} \quad (18)
 \end{aligned}$$

dengan kondisi stabilnya adalah:

$$\Delta t \leq \left[2\mu \left(\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2} \right) \right]^{-1} \quad (19)$$

jika $\mu=0$, maka kondisi menjadi tidak stabil.

B. Langkah difusi

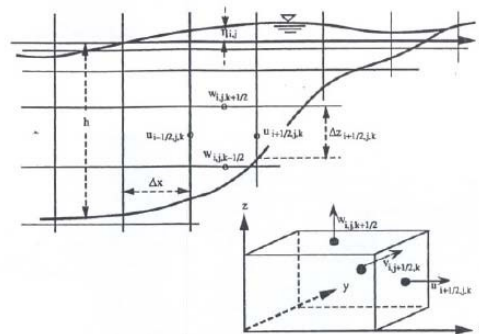
Untuk mengetahui gejala-gejala dari aliran air dari bawah sampai permukaan laut seperti kecepatan longitudinal (u), transversal (v), vertikal (w), dan permukaan bebas (z_1) maka digunakan metode numerik persamaan air dangkal dua dan tiga dimensi (Cea, French, dan Vazquez-Cendon, 2006; Hervouet, 2007).

Beberapa metode numerik untuk persamaan air dangkal dua dan tiga dimensi dengan bergantung pada waktu telah diketahui dalam literatur dan sekarang digunakan dalam aplikasi-aplikasi praktis. Pada tahun 1992, simulator aliran air dangkal dua dimensi adalah kompetitif secara ekonomi dengan metode ADI yang telah dikembangkan dan diterapkan (Casulli dan Cheng, 1992; Cheng dan Casulli, 1992). Metode ini termasuk semi-implisit yang lebih baik dari metode pemecahan implisit yang dikembangkan oleh Stelling (1984). Di dalam metode semi-implisit hanya penurunan tekanan barotropik dalam persamaan momentum dan divergen kecepatan dalam persamaan kontinuitas diambil secara implisit. Secara komputasional, pada tiap step waktu sistem lima diagonal linier diselesaikan dalam elevasi permukaan air yang baru untuk daerah masuk yang tidak diketahui. Koefisien matrix untuk tiap sistem adalah simetri dan pasti positif dan penyelesaiannya bisa ditentukan secara unik dan efisien dengan menggunakan suatu metode penurunan konjugasi. Metode pemisahan waktu implisit menggunakan dua atau lebih step waktu kecil yang secara esensial tidak dipasang operator propagasi dari konveksi dan difusi. Tiap operator ini kemudian didiskrit secara implisit.

Dalam analisis numerik untuk aliran air laut yang dangkal di teluk Manado Propinsi Sualwesi Utara, Indonesia dikembangkan sebagai fase satu dari pengembangan suatu model tiga dimensi (TRIM-D) secara umum. Tujuan analisis ini adalah untuk membuat dasar matematika yang kuat untuk skema numerik dan algoritme komputasional dalam penyelesaian numerik dari masalah-masalah aliran geofisik dua atau tiga dimensi.

Persamaan-persamaan Navier-Stokes adalah persamaan-persamaan umum yang bisa digunakan untuk memodelisasi gerakan dari air. Setiap kali kita mempertimbangkan suatu masalah khusus seperti aliran dalam air dangkal dimana skala horisontal lebih besar dari skala vertikal, itu memerlukan pertimbangan hipotesa pasti. Persamaan-persamaan Navier-Stokes yang dikonsiderasikan hasil penelitian Casulli dan Cheng (1992) yang bisa kita lihat pada pers. 1, 2, dan 3 valid jika lengkapi dengan hipotesa dari Boussinesq, itu berlaku jika perubahan densitasnya kecil, densitas itu harus mempertimbangkan konstanta dalam semua titik yang menerima gaya gravitasi. Begitu juga persamaan-persamaan yang sama itu telah dikembangkan oleh Cornett, Cousineau, dan Nistor (2013) yang mana mereka mengembangkan kecepatan u dan v menjadi kecepatan-kecepatan rata-rata kedalaman (*depth-averaged velocities*).

Dalam aturan tiga dimensi itu, persamaan-persamaan variabel primitif menggambarkan densitas konstan, aliran permukaan bebas di dalam embayments dan lautan-lautan yang berhubungan dengan pantai bisa diturunkan dari persamaan-persamaan Navier-Stokes setelah merata-ratakan turbulen dan dibawah asumsi penyederhanaan bahwa tekanan adalah hidrostatik (Hervouet, 2007).



Gambar 4. Skema diagram mesh dan notasi komputasional

Suatu analisis karakteristik dari dua dimensi, secara vertikal diintegrasikan persamaan air dangkal yang menunjukkan bahwa faktor \sqrt{gH} dalam persamaan itu

mempunyai tujuan untuk memunculkan karakteristik berbentuk kerucut dari penurunan tekanan barotropik di dalam persamaan momentum dan dari turunan kecepatan di dalam persamaan permukaan bebas. Suatu analisis stabilitas yang setepat-tepatnya juga ditetapkan dengan menggunakan metode von Neumann pada skema hubungan linieritas. Hasil dari analisis ini mempunyai peranan penting pada metode semi-implisit praktis dari penyelesaian untuk persamaan air dangkal tiga dimensi yang mempunyai jaminan dalam beberapa aplikasi. Langkah-langkah yang akan dilakukan adalah pertama-tama pers. 1, 2, dan 3 akan diturunkan dalam bentuk penurunan elevasi permukaan di dalam persamaan momentum dan pers. 1, 2, dan 3 akan didiskritisasi secara implisit. Konveksi, *Coriolis*, dan faktor kecepatan horisontal di dalam persamaan momentum akan didiskritisasi secara eksplisit untuk menghilangkan kondisi stabilitas yang disebabkan oleh viskositas pusar vertikal, faktor pencampuran vertikal akan didiskritisasi secara implisit (Casulli dan Cheng, 1992).

Gambar 4. menunjukkan suatu ruang berlubang yang terdiri dari sel-sel persegi empat dari panjang Δx , lebar Δy dan tinggi Δz sebagai langkah awal. Kemudian tiap sel diberi nomor pada tengahnya dengan tanda i, j dan k . Diskrit kecepatan u kemudian di definisikan pada setengah integer i, j dan k ; v di definisikan pada integer i, k , dan setengah integer j ; w di definisikan pada integer i, j , dan setengah integer k . Terakhir, η di definisikan pada integer i dan j . Kedalaman air $h(x,y)$ dispesifikasikan pada titik-titik u dan v horisontal. Kemudian pengdiskritisasi semi-implisit secara umum dari persamaan momentum dari pers. 1, 2, dan 3 berbentuk sebagai berikut

$$\begin{aligned}
 & \text{(Casulli dan Cheng, 1992):} \\
 & \mathbf{A}_{i,j+1/2}^{n+1} \mathbf{U}_{i,j+1/2}^{n+1} = \mathbf{G}_{i,j+1/2}^{n+1} - g \frac{\Delta t}{\Delta y} (\eta_{i,j+1}^{n+1} - \eta_{i,j}^{n+1}) \mathbf{AZ}_{i,j+1/2}^n \\
 & \mathbf{A}_{i+1/2,j}^{n+1} \mathbf{V}_{i+1/2,j}^{n+1} = \mathbf{G}_{i+1/2,j}^{n+1} - g \frac{\Delta t}{\Delta x} (\eta_{i+1,j}^{n+1} - \eta_{i,j}^{n+1}) \mathbf{AZ}_{i+1/2,j}^n \\
 & \mathbf{A}_{i,j+1/2}^{n+1} \mathbf{W}_{i,j+1/2}^{n+1} = \mathbf{G}_{i,j+1/2}^{n+1} - g \frac{\Delta t}{\Delta z} (\eta_{i,j+1}^{n+1} - \eta_{i,j}^{n+1}) \mathbf{AZ}_{i,j+1/2}^n
 \end{aligned} \tag{20}$$

$$\mathbf{U}_{i,j+1/2}^{n+1} = \begin{bmatrix} \bar{u}_{i+1/2,j,m}^{n+1} \\ \vdots \\ \bar{u}_{i+1/2,j,m}^{n+1} \end{bmatrix}, \quad \mathbf{V}_{i+1/2,j}^{n+1} = \begin{bmatrix} \bar{v}_{i,j+1/2,m-2}^{n+1} \\ \vdots \\ \bar{v}_{i,j+1/2,m-2}^{n+1} \end{bmatrix}, \quad \mathbf{W}_{i,j+1/2}^{n+1} = \begin{bmatrix} \bar{w}_{i,j+1/2,m}^{n+1} \\ \vdots \\ \bar{w}_{i,j+1/2,m}^{n+1} \end{bmatrix}$$

dimana $\mathbf{U}, \mathbf{V}, \mathbf{W}, \mathbf{AZ}$ dan \mathbf{G} didefinisikan sebagai:

$$\mathbf{U}_{i,j+1/2}^{n+1} = \begin{bmatrix} \frac{(\bar{v}_{i,j+1/2,m-2})_{i,j+1/2}^{n+1} \Delta t}{\Delta x} + \frac{(\bar{v}_{i,j+1/2,m-2})_{i,j+1/2}^{n+1} \Delta t}{\Delta x} \\ \vdots \\ \frac{(\bar{v}_{i,j+1/2,m-2})_{i,j+1/2}^{n+1} \Delta t}{\Delta x} + \frac{(\bar{v}_{i,j+1/2,m-2})_{i,j+1/2}^{n+1} \Delta t}{\Delta x} \end{bmatrix}, \quad \mathbf{V}_{i+1/2,j}^{n+1} = \begin{bmatrix} \bar{v}_{i,j+1/2,m-2}^{n+1} \\ \vdots \\ \bar{v}_{i,j+1/2,m-2}^{n+1} \end{bmatrix}, \quad \mathbf{W}_{i,j+1/2}^{n+1} = \begin{bmatrix} \bar{w}_{i,j+1/2,m}^{n+1} \\ \vdots \\ \bar{w}_{i,j+1/2,m}^{n+1} \end{bmatrix}$$

Persamaan 4 dan 5 adalah sistem tridiagonal linier yang dikopel ke elevasi permukaan air laut η^{n+1} pada waktu t_{n+1} .

Untuk menghitung gelombang air (permukaan bebas) $\eta_{i,j}^{n+1}$ bisa ditulis dalam notasi matriks sebagai berikut:

$$\eta_{i,j}^{n+1} = \eta_{i,j}^{n+1} - \frac{\Delta t}{\Delta x} [(\Delta Z_{i+1/2,j})^T \mathbf{U}_{i+1/2,j}^{n+1} - (\Delta Z_{i-1/2,j})^T \mathbf{U}_{i-1/2,j}^{n+1}] \quad (22)$$

C. Langkah tekanan kontinuitas

Persamaan yang digunakan adalah pengembangan dari Pers. 4, Pers. 17, dan Pers. 18 menjadi persamaan baru:

$$\begin{aligned} & \eta_{i,j}^{n+1} - g \frac{\Delta t^2}{\Delta x^2} \left\{ (\Delta \Delta Z^T A^{-1} \Delta Z)_{j+1/2,j}^T (\eta_{i+1,j}^{n+1} - \eta_{i,j}^{n+1}) - [(\Delta \Delta Z^T A^{-1} \Delta Z)_{j-1/2,j}^T (\eta_{i,j}^{n+1} - \eta_{i-1,j}^{n+1})] \right\} \\ & - g \frac{\Delta t^2}{\Delta y^2} \left\{ (\Delta \Delta Z^T A^{-1} \Delta Z)_{j+1/2,j}^T (\eta_{i,j+1}^{n+1} - \eta_{i,j}^{n+1}) - [(\Delta \Delta Z^T A^{-1} \Delta Z)_{j-1/2,j}^T (\eta_{i,j}^{n+1} - \eta_{i,j-1}^{n+1})] \right\} \\ & = \eta_{i,j}^n - \frac{\Delta t}{\Delta x} \left\{ (\Delta Z)^T A^{-1} G_{j+1/2,j}^T - [(\Delta Z)^T A^{-1} G_{j-1/2,j}^T] \right\} \\ & - \frac{\Delta t}{\Delta y} \left\{ (\Delta Z)^T A^{-1} G_{j+1/2,j}^T - [(\Delta Z)^T A^{-1} G_{j-1/2,j}^T] \right\} \end{aligned} \quad (23)$$

Untuk mendapatkan kecepatan \bar{w} dalam arah z (arah ke dasar laut) dikembangkan dari Pers. 3 menjadi:

$$\begin{aligned} \bar{w}_{i,j,k+1/2}^{n+1} &= \bar{w}_{i,j,k-1/2}^{n+1} - \frac{\Delta z_{i+1/2,j,k}^n \bar{u}_{i+1/2,j,k}^{n+1} - \Delta z_{i-1/2,j,k}^n \bar{u}_{i-1/2,j,k}^{n+1}}{\Delta x} \\ & - \frac{\Delta z_{i,j+1/2,k}^n \bar{v}_{i,j+1/2,k}^{n+1} - \Delta z_{i,j-1/2,k}^n \bar{v}_{i,j-1/2,k}^{n+1}}{\Delta y} \end{aligned} \quad (24)$$

dimana $k=m, m+1, \dots, M$, dan $\bar{w}_{i,j,m-1/2}^{n+1} = 0$ artinya tidak ada aliran melintasi daerah ke dasar laut.

D. Energi kinetik

Tenaga atau energi kinetik yang tersedia per satuan luas tegak lurus (BC Hydro, 2002; Luquet, Bellevre, Fréchu, Perdon, dan Guinard, 2013) adalah:

$$P_A = \frac{1}{2} \rho v^3 10^{-3} \quad (25)$$

dimana P_A dalam kW/m^2 , v adalah resultan kecepatan arus laut (m/s) dan ρ densitas air laut (kg/m^3). Persamaan 7 akan diubah menjadi persamaan numerik saat dibuat program komputasi numerik menjadi:

$$P_A = \frac{1}{2} \rho (v_{i,j,k}^{n+1})^3 10^{-3} \quad (26)$$

dimana P_A adalah energi kinetik per satuan luas dalam kW/m^2 dan $v_{i,j,k}^{n+1} = \sqrt{\bar{u}^2 + \bar{v}^2 + \bar{w}^2}$ adalah resultan kecepatan dengan

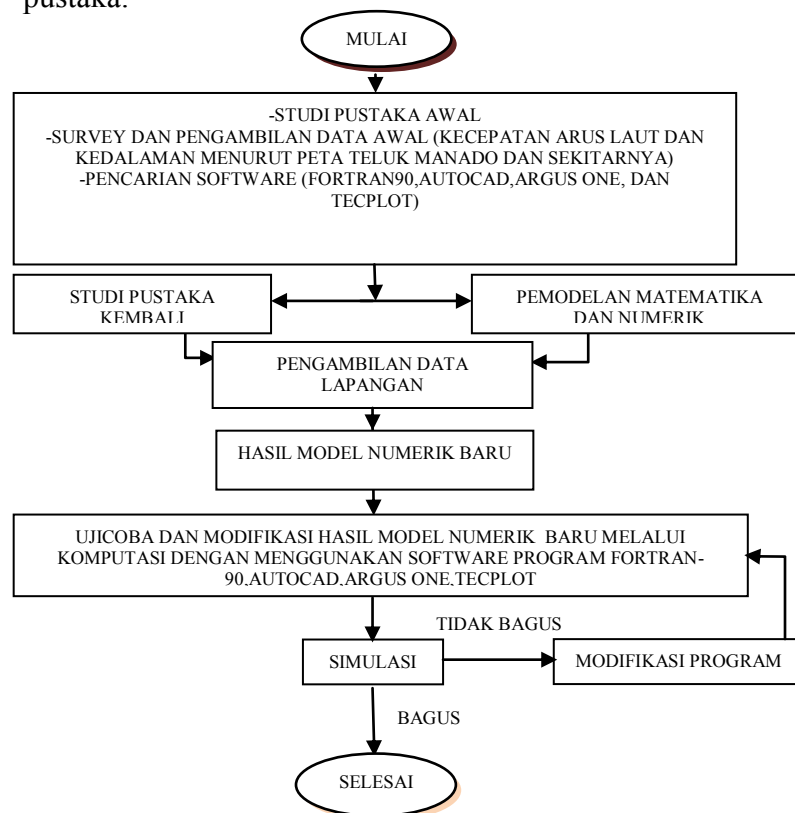
$$\bar{u} = \frac{1}{2} (\bar{u}_{i,j,k}^{n+1} + \bar{u}_{i+1,j,k}^{n+1}), \quad \bar{v} = \frac{1}{2} (\bar{v}_{i,j,k}^{n+1} + \bar{v}_{i,j+1,k}^{n+1}) \quad \text{dan}$$

$$\bar{w} = \frac{1}{2} (\bar{w}_{i,j,k}^{n+1} + \bar{w}_{i,j,k+1}^{n+1}) \quad \text{masing-masing adalah skalar.}$$

BAB 5. METODE

Metode yang akan digunakan dalam penelitian ini adalah:

- Studi pustaka yaitu penelusuran bahan *software* (program-program komputer), peta lokasi penelitian, dan teori melalui pustaka.



Gambar 5. Diagram alir penelitian

- Pengambilan data input untuk ujicoba dan modifikasi model numerik baru dilakukan secara langsung di lokasi penelitian yaitu teluk Manado (lihat gambar 7) kota Manado propinsi Sulawesi Utara, Indonesia (sebelum pengukuran data maka diadakan kalibrasi alat ukur). Data generasi grid (*mesh*) dan

notasi dilakukan dengan cara mengkonversi data menggunakan *software autocad 2010* dari peta teluk Manado dengan ukuran 10440 m X 19080 m X 859 m dengan jumlah grid 174 X 318 X 4 = 221328 artinya ukuran *mesh* 174 untuk sumbu *x*, 318 untuk sumbu *y*, dan 4 untuk sumbu *z*. Tiap elemen horizontal berukuran 7 m x 7 m untuk 2D dan dibuat 4 lapis elemen vertikal untuk 3D yang mana kesemuanya menjadi data input menggunakan *software argus one* yang kemudian digunakan dalam analisis komputasi. Analisis komputasi menggunakan *software fortran 90* dan simulasinya menggunakan *software tecplot 9*.

- Metode matematika, numerik dan komputasi yaitu membuat model matematik kemudian dirubah menjadi model numerik yang kemudian dianalisis komputasi untuk mendapatkan model numerik baru sebagai indikator capaian yang terukur.
- Metode simulasi numerik yaitu hasil model numerik baru yang telah diujicobakan dan telah mendapatkan hasil yang diharapkan kemudian disimulasikan dan hasilnya adalah sebagai target dan tujuan khusus penelitian ini.

Desain lengkap penelitian utama dapat dilihat pada gambar 5 dalam bentuk bagan diagram alir. Penelitian dimulai dari hal-hal yang telah dikerjakan, kemudian kegiatan persiapan sebelum masuk dalam kegiatan ujicoba hasil pemrograman berupa komputasi model numerik. Setelah itu jika hasil simulasi bagus maka penelitian selesai tetapi jika tidak maka harus dilakukan modifikasi pemrogramannya kemudian diujicobakan lagi dan seterusnya.

A. Data set

Kalkulasi numerik 3D untuk pemodelan hasil pertemuan arus- arus yang berasal dari laut teluk Manado dan sungai Singkil (lihat gambar 7) digunakan program fortran 90. Kalkulasi terdiri dari dua kondisi yaitu pada kondisi aliran air pasang dan surut. Data-data awal yang dimasukkan adalah $\Delta t = 1$ s, $g = 9,81$ m/s²,

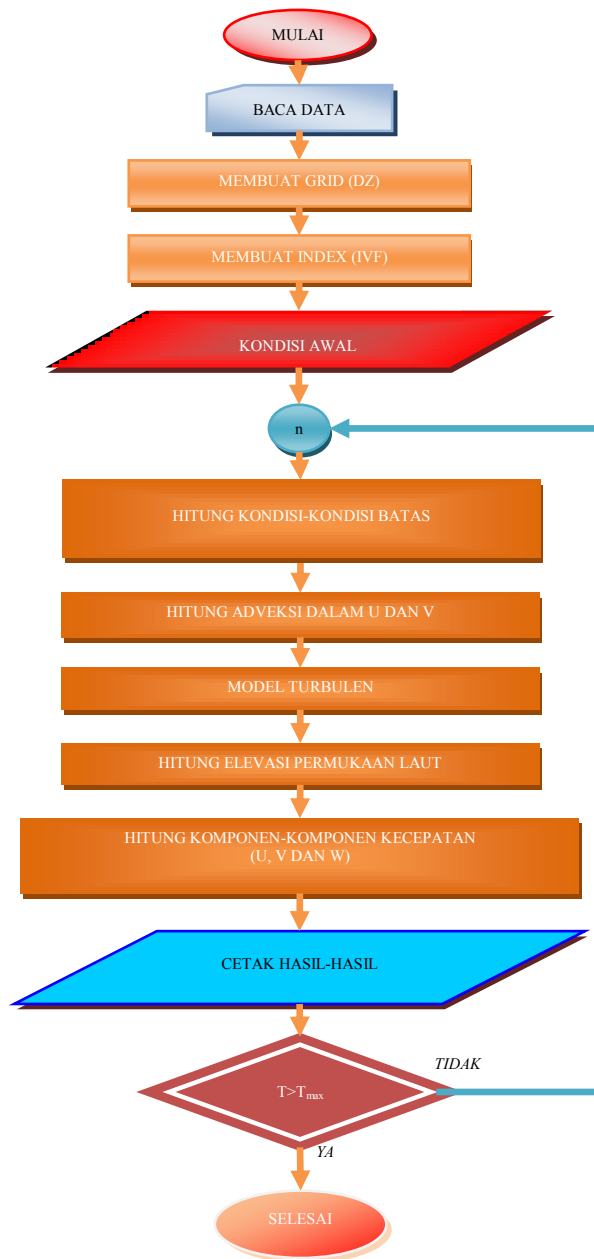
$Cz = 48,04$, $W_{10} = 1 \text{ m/s}^2$, iterasi maksimum $T_{\max} = 720000$, densitas udara $1,178 \text{ kg/m}^3$, $\rho_{\text{air}} = 1024 \text{ kg/m}^3$, dx , dy , dan dz masing-masing 7 m , 7 m , dan 1 m , $\omega = 7.29212 \times 10^{-5} \text{ s}^{-1}$ dengan sudut $1,45^\circ$, $\tau_0 = 2 \text{ hari}$ dan $\tau_i = 1 \text{ hari}$, debit air laut saat masuk adalah $100 \times 10^3 \text{ m}^3/\text{s}$.

B. Pembuatan grid

Grid dibuat dengan langkah-langkah sebagai berikut: pertama-tama peta lokasi numerik dalam bentuk *bathymetry* dibuat format DXF kemudian di import ke program Argus ONE. Pembuatan grid membutuhkan langkah-langkah pengerjaan sampai pada tahap ekspor grid dalam bentuk format EXP yang dipakai sebagai data-data grid dalam kalkulasi yang berisi seperti jumlah grid arah x dan y , indeks 0 untuk daratan dan 1 untuk lautan, dan terakhir data kedalaman laut hasil interpolasi yang dihitung dari program Argus ONE. Hasilnya adalah grid untuk arah x , y , dan z masing-masing adalah 174, 318, dan 2. Luas bidang numerik adalah lebar $1,218 \text{ km}$ x panjang $2,228 \text{ km}$ (lihat Gambar 7). Maksimum kedalaman laut adalah 79 m .

C. Pembuatan indeks

Indeks dibuat dengan memberi penomoran yaitu 99, 88, dan 77. Jika $dz = 0$ maka diberi 99 sedangkan jika $dz \neq 0$ maka diberi tanda 1. Juga, penomoran 99 diberi tanda bahwa tidak ada kecepatan-kecepatan air arah x , y , dan z . Penomoran 88 dipakai sebagai tanda untuk kecepatan-kecepatan air dan elevasi permukaan air pada daerah masuk dan keluar sistem. Terakhir penomoran 77 dipakai sebagai tanda pada perhitungan kecepatan-kecepatan arah x dan y dalam langkah adveksi dan dalam perhitungan elevasi permukaan laut.



Gambar 6. Gambar alir kalkulasi numerik

D. Kondisi awal

Data-data kondisi awal yang dimasukkan ada dua kondisi yaitu pertama, saat kalkulasi baru dan kedua, saat sudah pernah kalkulasi dan akan dilanjutkan lagi. Pada kondisi pertama, semua kecepatan air sama dengan nol baik lama maupun baru, elevasi permukaan laut juga sama dengan nol baik , dan viskositas-viskositas turbulen arah x dan y sama dengan viskositas kinematik molekuler. Pada kondisi kedua, hanya data-data lama hasil kalkulasi sebelumnya seperti kecepatan-kecepatan air dan elevasi permukaan laut yang menjadi kondisi awal.



Gambar 7. Peta teluk Manado dan sungai Singkil

E. Cetak hasil kalkulasi

Sementara kalkulasi dilakukan maka data-data direkam dan dicetak dan ketika iterasi mencapai maksimum (T_{\max}) maka kalkulasi selesai. Data-data hasil kalkulasi kemudian disimulasikan dengan menggunakan program *Tecplot 9*.

F. Validasi hasil penelitian

Validasi data hasil penelitian dilakukan dengan cara membandingkan kecepatan-kecepatan arus laut hasil kalkulasi numerik dan hasil observasi di sekitar muara sungai Singkil (pada 1 m di bawah permukaan laut). Data-data observasi diukur bersamaan dengan pengukuran kecepatan arus sungai masuk ke laut. Juga, diobservasi fenomena aliran di lokasi tersebut. Didapat hasil kalkulasi numerik mendekati hasil observasi maka hasil kalkulasi numerik dapat diterima.

BAB 6. HASIL DAN PEMBAHASAN

Hasil yang dicapai adalah sebuah model numerik berupa program untuk menghitung distribusi kecepatan arus laut dan sungai di Teluk Manado propinsi Sulawesi Utara, Indonesia dan simulasi distribusi kecepatan arus laut dan sungai (2D dan 3D) hasil perhitungan program.

A. Sebuah model numerik

Sebuah model numerik yang dibuat berdasarkan pengembangan persamaan permukaan bebas dari persamaan-persamaan Navier-Stokes yang dikonsiderasikan hasil penelitian Casulli dan Cheng (1992) kemudian diturunkan setelah merata-ratakan turbulen dan dibawah asumsi penyederhanaan bahwa tekanan adalah hidrostatis (Hervouet, 2007). Kemudian dibuat dalam bentuk notasi matrik dan disusun sebuah program dengan pengerjaan sesuai diagram alir seperti yang ditunjukkan pada gambar 8 dan menggunakan bahasa program *fortran 90*.

Pembuatan program isinya terdiri dari dua bagian yaitu program utama dan program subrutin. Program utama adalah program keseluruhan yang terdiri dari subrutin-subrutin sedangkan program subrutin adalah program perhitungan bagian-bagian.

Proses pembuatan program dimulai dari pembuatan membaca data (subrutin) dimana data-data input yang akan digunakan dalam komputasi harus dimasukkan dalam bagian ini (data input dapat dilihat pada tabel 1). Kemudian memasukkan banyaknya *grid/mesh* (subrutin) yang sebelumnya telah dibuat memasukkan data daerah kalkulasi numerik (lihat gambar 7 pada bagian persegiempat warna merah) seperti nilai x (panjang ke arah x), y (panjang ke arah y), dan z (kedalaman air) di teluk Manado seperti pada gambar 6 dan setelah itu dibuat subrutin tentang pembuatan index (lihat gambar 4). Berikut membuat nilai kondisi awal dengan nilai nol untuk semua kondisi awal.

Proses berikut adalah pembuatan kondisi-kondisi batas (subrutin) dan *advectons* dalam kecepatan arah x (U) dan y (V) dalam hal ini notasi matriks A seperti pada persamaan-persamaan 4 dan 5 (subrutin). Kemudian membuat program menghitung model turbulensi (subrutin) dan elevasi permukaan (subrutin). Selanjutnya dibuat perhitungan komponen-komponen kecepatan dalam arah x (U), y (V), dan z (W) dan energi kinetik arus laut per satuan luas (subrutin).

Setelah semua pembuatan subrutin-subrutin itu selesai maka dibuatlah program utama dan selanjutnya program diujicobakan untuk dilihat hal-hal apa saja yang masih kurang dan ketika program dieksekusi maka terdapat kekurangan-kekurangan yang dapat ditunjukkan lewat indikator "*error*" sehingga harus dilihat kesalahan apa yang dimaksudkan oleh program itu dan harus dimodifikasi sampai program sudah tidak lagi *error* dan hasil kalkulasi sudah bagus maka eksekusi selesai dan dapat di cetak hasilnya untuk dilihat apakah sudah sesuai dengan yang diharapkan dan jika belum maka dilakukan lagi modifikasi terutama pada bagian data masuk dan keluar. Setelah hasil komputasi melalui hasil simulasi menggunakan program *teplot 8* sudah cukup bagus (dengan membandingkan data observasi kecepatan arus di sekitar sungai).

B. Model program numerik

1. Program utama

```
c YAXUM "telukmanadokanan.f" version parrallel
c kode menggunakan "equations de Shallow Water"
c untuk menghubungkan evolusi dari permukaan bebas dan
c kecepatan dari écoulement satuan air
c (lake, océan, etc.)
```

```
c=====
c=====
c      Menghitung dengan kondisi batas:
c      - 3 lapis (nk=3)
```



```

c      - Pada dt= 1 detik
c      - Sudut coriolis = 1,45 derajat
c      - Rho udara/Rho air = 0,001225
c      - Model Turbulen adalah modlm3D (Mixing-Length
model by Peter K STANSBY)
c      - Menggunakan Free Pass by Orlanski's (1976)
c      - Perhitungan model (dari kanan ke kiri)
c      - Menggunakan GRID-7 m x 7 m x 1 m

```

```

c=====
=====

```

```

c=====
      module physique

```

```

c=====
      implicit none

```

```

          integer, dimension(3), parameter :: n=(/176,320,5/) !grid
x,y,z

```

```

          integer ::

```

```

iter,itermax,iterken,nimp,nimpar,nimtp,niter,ncourant

```

```

          integer ::

```

```

imtur,icor,iforma,idz,icond1,icond0,itemp,isal,iconc,

```

```

&      iec,ied,ni,nj,nk,jkia1,jkia2,ikib1,ikib2,jkan1,jkan2,

```

```

&iata1,iata2,iatb1,iatb2,itr,ipr,ncour,nfp,ncc,nc,kq,nci,nprint,

```

```

&JTs,jst,jst2,jx,is,ist,ist2,kz,nbz,nbe,nbc,nbd,njt,ijb,iib,jbt,

```

```

&jat,jfa,jea,jhb,jjb,nd0,itera,nr,nrr,nbal,ncl,iterasi,jkan3,jkan4

```

```

      real(kind=8) :: t, dt, tmax      !waktu, selisih waktu, waktu
maksimum

```

```

      real(kind=8) :: rho0, chezy, g, fii, cnum, kp, ctehx, ctehy

```

```

      real(kind=8) :: tol

```

```

      real(kind=8) :: wxx,wyy,densr,cdv

```

```

      real(kind=8) :: zini,uini,vini,wini,tini,sini,cini

```

```

      real(kind=8) :: hzmin,rres,rn

```

```

      real(kind=8) ::

```

```

dzmin,dz0,ua,ub,ve,vg,ui,usri,usre,usrg,usra,usrb

```

```

    real(kind=8) ::
ua1,va1,ud1,vd1,ra1,rb1,dbal,ficg,fici,fica,ficb
    real(kind=8) ::
da,db,dg,di,dtot1,dtot2,da12,da34,dbal12,dbal34
    real(kind=8) :: ag,ai,xg,xi,rata,ratb,ratg,rati
    real(kind=8) :: aa,ab,xa,xb,dtot3,dtot4
    real(kind=8) ::
alp,blp,clp,dlp,elp,flp,glp,olp,plp,qlp,rlp,slp,
&tlp,ulp,vlp,wlp,xlp,ylp,zlp,aalp,ablp,aclp,adlp,aelp,aflp
    real(kind=8) ::
uta,vtb,uti,vtg,debmas,aaa,dnimtp,dnimpar

    integer, dimension(n(1),n(2)) :: nnkz,nnku,nnkv
    integer, dimension(n(1),n(2),n(3)) :: ivf,ivfu,ivfv

    real(kind=8), dimension(n(1),n(2),n(3))
::u,v,w,un,vn,wn,upy,upz,
& vpx,vpz,wpx,wpy ! vitesses
    real(kind=8), dimension(n(1),n(2),n(3)) ::
fu,fv,ekps,ekph,ekpj
    real(kind=8), dimension(n(1),n(2),n(3)) ::
vort,ekpm,ekpa,r,ek

    real(kind=8), dimension(n(1),n(2),n(3)) ::
cnue,cnueu,cnuev,cnuew

    real(kind=8), dimension(n(1),n(2)) :: z1,zu,zv,hz,hu,hv,
& tz,tu,tv
    real(kind=8), dimension(n(1),n(2),n(3)) :: dz,dzu,dzv,aam
! mailles
    real(kind=8), dimension(n(1)) :: dx,xx,dx0
    real(kind=8), dimension(n(2)) :: dy,yy,dy0
    real(kind=8), dimension(n(1),n(2)) :: wx,wy,ctevie,cor
!vent et coriolis
    real(kind=8), dimension(n(1),n(2),n(3)) :: u0,v0,w0

```

```

contains
c=====
=====
c*****
=
      program manado3d                               !=
c*****
=
c=====
=====
      use physique

      Call CPU_TIME(time_begin)
      write(*,*)'KOMPUTASI    NUMERIK    TELUK
MANADO'
      write(*,*)'KOMPUTASI DIMULAI DARI KANAN KE
KIRI'
      write(121,*)'iter  ,dbal  ,dtot1  ,da  ,db  ,uta  ,vtb'
      nbal=1
      nrr=0
      nc=1
      nci=0
      ntp=0

c  lee datos generales y calcula parametros y constantes
      call donnees
c  initialise toutes les variables
      call initialise
c  calcule le longueur du maille verticale
      if (idz.eq.1) call caldz1
      if (idz.eq.2) call caldz2

c  generacion de la variable ivf para saber si es frontera seca,
mojada
      call genivf
c  subrutina para asignar fronteras y promedios

```

```

call fronte
call prome

if (icond0.eq.1) then
  iterasi=1
endif

  write(201,*)' iter'
  write(202,*)' dtot1'
  write(203,*)' dtot3'
  write(204,*)' dbal'
c commence la bucle du temps****
do iter = iterasi, itermax

c write(*,*) ' calcula adveccion en u', iter
call advecu

c write(*,*) ' calcula adveccion en v', iter
call advecv

c subrutina para calcular la velocidad w
call calcw

c subrutina para asignar fronteras y promedios
call fronte
call prome

c write(*,*) ' modelo de turbulencia', iter
if (imtur.eq.1) call modlm1
if (imtur.eq.2) call modlm2
if (imtur.eq.3) call modlm2d
if (imtur.eq.4) call modlm3d

c write(*,*) ' calcula sup libre z', iter
call super5

```

```

c  write(*,*) ' calcula velocidades u ', iter
    call calcu

c  write(*,*) ' calcula velocidades v ', iter
    call calcv

c  subrutina para calcular la velocidad w
    call calcw

c  subrutina para asignar fronteras y promedios
    call fronte
    call prome

c  Actualization :
    un(:,:,) = u(:,:,)
    vn(:,:,) = v(:,:,)
    wn(:,:,) = w(:,:,)

c-----
c----->resultados e impresion de archivos-----
c-----
        nci=nci+1
        if(mod(iter,100).eq.0)then
write(*,600)iter,nci,nc,dbal,uti,rati,vtb,ratb
write(100,600)iter,nci,nc,dbal,uti,rati,vtb,ratb
write(201,700)iter
write(202,800)dtot1
write(203,900)dtot3
write(204,950)dbal
        endif

c-----
        if(mod(iter,nimtp).eq.0) then
            write(*,*)'Buat tecplot pada ',nc,' JAM',' ITER
',iter
            call vortic

```

```

call tecplo(iter,ntp)
endif
c-----
      if(mod(iter,nimtp).eq.0)then
write(10,*)'KONDISI SETELAH CALCUL'
write(10,*)'ITERASI =',iter
write(10,*)'=====AKHIR=====
=====
write(10,*)'=====
=====
write(10,*)'debit masuk total berikut =',dtot1
write(10,*)'luas masuk di ag      =',ag
write(10,*)'luas masuk di ai      =',ai
write(10,*)'luas masuk di aa      =',aa
write(10,*)'luas masuk di ab      =',ab
write(10,*)'kecepatan masuk di ratg =',ratg
write(10,*)'kecepatan masuk di rati =',rati
write(10,*)'kecepatan masuk di rata =',rata
write(10,*)'kecepatan masuk di ratb =',ratb
write(10,*)'debit masuk di dg      =',dg
write(10,*)'debit masuk di di      =',di
write(10,*)'debit masuk di da      =',da
write(10,*)'debit masuk di db      =',db
write(10,*)'=====
=====
write(10,*)'debit total keluar di db (db) =',dtot3
write(10,*)'luas keluar di ab      =',ab
write(10,*)'kecepatan keluar di ratb =',ratb
write(10,*)'selisih debit masuk-keluar =',dbal
write(*,*)'KONDISI SETELAH CALCUL'
write(*,*)'ITERASI =',iter
write(*,*)'=====AKHIR=====
=====
write(*,*)'=====
=====
write(*,*)'debit masuk total berikut =',dtot1

```

```

write(*,*)'luas masuk di ag      = ',ag
write(*,*)'luas masuk di ai     = ',ai
write(*,*)'luas masuk di aa     = ',aa
write(*,*)'luas masuk di ab     = ',ab
write(*,*)'kecepatan masuk di ratg = ',ratg
write(*,*)'kecepatan masuk di rati = ',rati
write(*,*)'kecepatan masuk di rata = ',rata
write(*,*)'kecepatan masuk di ratb = ',ratb
write(*,*)'debit masuk di dg     = ',dg
write(*,*)'debit masuk di di     = ',di
write(*,*)'debit masuk di da     = ',da
write(*,*)'debit masuk di db     = ',db
write(*,*)'=====
=====
write(*,*)'debit total keluar di db (db) = ',dtot3
write(*,*)'luas keluar di ab      = ',ab
write(*,*)'kecepatan keluar di ratb = ',ratb
write(*,*)'selisih debit masuk-keluar = ',dbal
endif

c-----
      if(mod(iter,nimtp).eq.0) then
write(121,111)iter,dbal,dtot1,dtot3,db,ratb,v(10,5,nk)
write(*,111)iter,dbal,dtot1,dtot3,db,ratb,v(10,5,nk)
endif

c-----
if(mod(iter,nimpar).eq.0) then
open (60, file='z1.dat')
open (61, file='u1.dat')
open (62, file='v1.dat')
open (63, file='w1.dat')
do j=1,nj
  write(60,300) (z1(i,j),i=1,ni)
end do
do k=1,nk
  do j=1,nj
    write(61,300) (u(i,j,k),i=1,ni)

```

```

        write(62,300) (v(i,j,k),i=1,ni)
        write(63,300) (w(i,j,k),i=1,ni)
    end do
end do
close(60)
close(61)
close(62)
close(63)
endif

```

```

c=====
=====

```

```

if(mod(iter,(nimpar-1)).eq.0) then
    open (701, file='un.dat')
    open (702, file='vn.dat')
    open (703, file='wn.dat')
    do k=1,nk
        do j=1,nj
            write(701,300) (un(i,j,k),i=1,ni)
            write(702,300) (vn(i,j,k),i=1,ni)
            write(703,300) (wn(i,j,k),i=1,ni)
        end do
    end do
    close(701)
    close(702)
    close(703)
    open (801, file='iter.dat')
        write(801,*)iter
        close(801)
    open (802, file='nc.dat')
        write(802,*)nc,nci
        close(802)
    endif

```

```

c=====
=====

```

```

        if(mod(iter,nimtp).eq.0)then
            nc=nc+1

```



```

                                nci=0
                                endif
c=====
=====
                                if(iter.eq.itermax) go to 1234
                                enddo  !****fin de la boucle du temps****

1234  write(*,*)
      write(*,*)
      print*, '          kalkulasi selesai!!!'
      write(*,*)
      write(*,*)
      Call CPU_TIME(time_end)
      Print*, 'Time of Operation was',time_end-time_begin,'
seconds'
111  format(i7,4f10.0,2f6.2)
300  format(1000e20.10)
500  format(6f10.4)
600  format(' i',i7,' n,h',i5,i4,' d= ',f7.0,' ti,ri,tb=',3f7.4,
      &' rb=',f7.4,' m/s')
1000  format(i7,6f7.4,f10.0)
700  format(i7)
800  format(f10.0)
900  format(f10.0)
950  format(f10.0)

                                end program manado3d

```

2. Program subrutin

```
c*****
  subroutine donnees
c*****
c  sub rutin untuk memasukkan data-data
  integer :: i, j, nbasura0, nbasura1
  integer, dimension (n(1),n(2)) :: nflag
  real(kind=8) :: x0, y0

  character(len=30)linea0, linea2, linea9, linea3, linea4

  character(len=10)frase01,frase02,frase03,frase04,frase05,frase0
6

  character(len=10)frase07,frase08,frase09,frase10,frase11,frase1
2

  character(len=10)frase13,frase14,frase15,frase16,frase17,frase1
8

  character(len=10)frase19,frase20,frase21,frase22,frase23,frase2
4

  character(len=10)frase25,frase26,frase27,frase28,frase29
  character(len=10)frase30,frase31,frase32

  open(1, file ='data_gen.dat',status='old')

  read (1,'(a)') linea0
  read (1,'(a)') linea2
  open (2,file=linea2)
  write (2,*) linea0
  read (1,'(a)') linea9
  read (1,'(a)') linea3
  read (1,'(a)') linea4
  open (9,file=linea9)
```

```

write (9,*) 'title="",linea0,'''
read (1,*) frase01,dt
read (1,*) frase02,g
read (1,*) frase03,fii
read (1,*) frase04,chezy
read (1,*) frase05,tol
read (1,*) frase06
read (1,*) frase06,itermax
read (1,*) frase07,nimp
read (1,*) frase08,nimtp
read (1,*) frase09,nimpar
read (1,*) frase10,niter
read (1,*) frase11
read (1,*) frase11,cdv
read (1,*) frase12,densr
read (1,*) frase13,wxx
read (1,*) frase14,wyy
read (1,*) frase15,ctehx
read (1,*) frase16,ctehy
read (1,*) frase17,nk
read (1,*) frase18,dz0
read (1,*) frase19,dzmin
read (1,*) frase19
read (1,*) frase17,imtur !1->lm1, 2->lm2
read (1,*) frase18,icor !1->on, 0->off
read (1,*) frase19,iforma !3->argus
read (1,*) frase20,idz !1->cte, 2->var
read (1,*) frase21,icond1
read (1,*) frase16
read (1,*) frase22,icond0
read (1,*) frase23,itemp
read (1,*) frase24,isal
read (1,*) frase25,iconc
read (1,*) frase26,iec
read (1,*) frase27,ied
close (1)

```

```

c  nilai kode umum
rho0 = 1024.0d0    !densité zéro
cnum = 0.000001d0 !vicosité moléculaire
kp  = 0.41d0      !cte von Karman
c    membuka data grid dan kedalaman laut dari program
argusone
open (11,file=linea3, status='old')
c  code
read(11,*) nj, ni, nbasura0, nbasura1
c  grid arah y
read(11,*) y0
do j=1,nj
  read(11,*) dy0(j)
  dy(j) = dy0(j)*4.60d0
enddo
c  grid arah x
read(11,*) x0
do i=1,ni
  read (11,*) dx0(i)
  dx(i) = dx0(i)*4.60d0
enddo
c  index
do j=1,nj
  read (11,*) (nflag(i,j),i=1,ni)
enddo
c  kedalaman laut
do j=1,nj
  read (11,*) (hz(i,j),i=1,ni)
enddo
do j=1,nj
  do i=1,ni
    if(nflag(i,j).eq.0) then
      hz(i,j)=0.0d0
    endif
  enddo
enddo

```

```

close(11)
open (1, file ='condawal.dat',status='old')
  read (1,*) frase01,zini
  read (1,*) frase02,uini
  read (1,*) frase03,vini
  read (1,*) frase04,wini
  read (1,*) frase05,tini
  read (1,*) frase06,sini
  read (1,*) frase07,cini
close (1)
c
=====
=====
c   kondisi batas awal pada daerah masuk dan keluar
c
=====
=====
c   ==== MASUK ====
c   batas daerah sebelah atas (g)
      iatb1=1
      iatb2=159
c   batas daerah sebelah kiri (i)
      jkan1=182
      jkan2=194
c   batas daerah sebelah kiri bawah (a)
      jkia1=1
      jkia2=318
c   ==== KELUAR ====
c   batas daerah sebelah kiri atas (b)
      ikib1=1
      ikib2=141
c   batas dermaga manado
      jkan3=129
      jkan4=137

```

```

c
=====
=====
c      Debit Masuk
c
=====
=====
      debmas=100000.0d0
c
=====
=====
c
=====
=====
      end subroutine donnees

c
*****
*****
      subroutine initialise
c
*****
*****
      integer :: i, j, k
      integer, dimension (n(1),n(2)) :: nflag
c-----

c coriolis (latitud)
  do j=1,nj
    do i=1,ni
      cor(i,j)=0.000145d0*sin(fii*3.141592d0/180.0d0)
    enddo
  enddo

c vent
  do j=1,nj

```

```

do i=1,ni
  wx(i,j)=wxx
  wy(i,j)=wyy

  ctevie(i,j)=((0.75d0+0.067d0*sqrt(wx(i,j)*wx(i,j)
&      +wy(i,j)*wy(i,j)))*0.001d0)
&      *densr
  enddo
enddo

```

c initialisation des vitesses et surface libre pour icond0 = 1

```

if (icond0.eq.1) then
  z1(:,:) = zini
  u(:,,:) = uini
  v(:,,:) = vini
  w(:,,:) = wini
  iterken=1
endif

```

c initialisation des vitesses moyennes

```

vpx(:,,:) = 0.0d0
upy(:,,:) = 0.0d0
vpz(:,,:) = 0.0d0
upz(:,,:) = 0.0d0
wpx(:,,:) = 0.0d0
wpy(:,,:) = 0.0d0

```

c initialisation du cnue

```

cnueu(:,,:) = cnum
cnuev(:,,:) = cnum
cnue(:,,:) = cnum
un(:,,:) = u(:,,:)
vn(:,,:) = v(:,,:)
wn(:,,:) = w(:,,:)

```

```

c      initialisation des vitesses et surface libre pour icond0 = 2
      if (icond0.eq.2) then
          write(*,*) ' HITUNG LANJUTAN KALKULASI
NUMERIK TELUK MANADO'
          open (801, file='iter.dat')
          read(801,*)iter
          iterasi=iter+1
          close(801)
          open (802, file='nc.dat')
          read(802,*)nc,nci
          close(802)
          write(*,*)'lanjut iterasi ke_',iterasi,' lanjut ke_',nc,
&' hit ke_',nci
          open(3, file='z1.dat')
          do j=1,nj
              read(3,*) (z1(i,j),i=1,ni)
          end do
          write(*,*) 'termine z1'
          close (3)

          open(3, file='u1.dat')
              open(5,file='un.dat')
          do k=1,nk
              do j=1,nj
                  read(3,*) (u(i,j,k),i=1,ni)
                  read(5,*) (un(i,j,k),i=1,ni)
              end do
          end do
          write(*,*) 'termine u1'
          write(*,*) 'termine un'
          close (3)
          close (5)

          open(3, file='v1.dat')
              open(5,file='vn.dat')
          do k=1,nk

```



```

do j=1,nj
  read(3,*) (v(i,j,k),i=1,ni)
  read(5,*) (vn(i,j,k),i=1,ni)
end do
end do
write(*,*) 'termine v1'
write(*,*) 'termine vn'
close (3)
close (5)
open(3, file='w1.dat')
open(5,file='wn.dat')

do k=1,nk
do j=1,nj
  read(3,*) (w(i,j,k),i=1,ni)
  read(5,*) (wn(i,j,k),i=1,ni)
end do
end do
write(*,*) 'termine w1'
write(*,*) 'termine wn'
close (3)
close (5)

endif
c----->Coordendas para tecplot
xx(1)=dx(1)*0.5d0
yy(1)=dy(1)*0.5d0
do i=2,ni
  xx(i)=xx(i-1)+dx(i-1)*0.5d0+dx(i)*0.5d0
end do

do j=2,nj
  yy(j)=yy(j-1)+dy(j-1)*0.5d0+dy(j)*0.5d0
end do

end subroutine initialise

```

```

c
*****
*****
      subroutine caldz1
c
*****
*****
c subrutina que calcula los diferentes dz
  integer :: i, j, k
  real(kind=8), dimension(n(3)) :: dz9

c----->hu
  do j=1,nj
    do i=1,ni
      if(i.eq.1) hu(i,j)=hz(i,j)
      if (i.eq.ni+1) hu(i,j)=hz(ni,j)
      if(i.gt.1.and.i.lt.ni+1) hu(i,j)=0.5d0*(hz(i-1,j)+hz(i,j))
    end do
  end do

c----->hv
  do j=1,nj
    do i=1,ni
      if (j.eq.1) hv(i,j)=hz(i,j)
      if (j.gt.1.and.j.lt.nj+1) hv(i,j)=0.5d0*(hz(i,j)+hz(i,j-1))
      if (j.eq.nj+1) hv(i,j)=hz(i,j-1)
    end do
  end do

c----->zu
  do j=1,nj
    do i=1,ni
      if (i.gt.1.and.i.lt.ni+1) then
        zu(i,j)=0.5d0*(z1(i,j)+z1(i-1,j))
      endif
      if(i.eq.1) zu(i,j)=z1(1,j)
    end do
  end do

```

```

        if (i.eq.ni+1) zu(i,j)=z1(ni,j)
    end do
end do

c----->zv
do j=1,nj
do i=1,ni
    if(j.gt.1.and.j.lt.nj+1) then
        zv(i,j)=0.5d0*(z1(i,j)+z1(i,j-1))
    endif
    if(j.eq.1)    zv(i,j)=z1(i,1)
    if (j.eq.nj+1) zv(i,j)=z1(i,nj)
end do
end do

c----->dz
do i=1,ni
do j=1,nj
    tz(i,j)=z1(i,j)-hz(i,j)
    call detdz(n,tz(i,j),dz0,dzmin,nnkz(i,j),dz9,nk)
    do k=1,nk
        dz(i,j,k)=dz9(k)
    end do
end do
end do

c----->dzu
do j=1,nj
do i=1,ni
    tu(i,j)=zu(i,j)-hu(i,j)
    call detdz(n,tu(i,j),dz0,dzmin,nnku(i,j),dz9,nk)
    do k=1,nk
        dzu(i,j,k)=dz9(k)
    end do
end do
end do

```

```

c----->dzv
  do j=1,nj
    do i=1,ni
      tv(i,j)=zv(i,j)-hv(i,j)
      call detdz(n,tv(i,j),dz0,dzmin,nkv(i,j),dz9,nk)
      do k=1,nk
        dzv(i,j,k)=dz9(k)
      end do
    end do
  end do
end do

300 format(1000e10.2)

end subroutine caldz1

c
*****
*****
  subroutine caldz2
c
*****
*****
c      subrutina que calcula los diferentes dz que van a ser
variables
c      de acuerdo al problema que se presente
integer      :: i, j, k, nkefe, kk
real(kind=8) :: dzacum,acum
real(kind=8),dimension(n(3)) :: dz9
character(len=10)frase01

c      archivo donde se indican los espaciamentos verticales
variables
c      los deltas z se dan de arriba para abajo
open (1, file ='delta-z.dat',status='old')

```

```

c
    read (1,*) frase01
    read (1,*) frase01
    read (1,*) frase01
    read (1,*) nkefe

c
    do k=1,nkefe
        read (1,*) dz9(k)
    end do

c
    close (1)
c----->hu
    do j=1,nj
        do i=1,ni
            if(i.eq.1) hu(i,j)=hz(i,j)
            if (i.eq.ni+1) hu(i,j)=hz(ni,j)
            if(i.gt.1.and.i.lt.ni+1) hu(i,j)=0.5d0*(hz(i-1,j)+hz(i,j))
        end do
    end do

c----->hv
    do j=1,nj
        do i=1,ni
            if (j.eq.1) hv(i,j)=hz(i,j)
            if (j.gt.1.and.j.lt.nj+1) hv(i,j)=0.5d0*(hz(i,j)+hz(i,j-1))
            if (j.eq.nj+1) hv(i,j)=hz(i,j-1)
        end do
    end do

c----->zu
    do j=1,nj
        do i=1,ni
            if (i.gt.1.and.i.lt.ni+1) then
                zu(i,j)=0.5d0*(z1(i,j)+z1(i-1,j))
            endif
            if(i.eq.1) zu(i,j)=z1(1,j)
        end do
    end do

```

```

        if (i.eq.ni+1) zu(i,j)=z1(ni,j)
    end do
end do

c----->zv
do j=1,nj
do i=1,ni
    if(j.gt.1.and.j.lt.nj+1) then
        zv(i,j)=0.5d0*(z1(i,j)+z1(i,j-1))
    endif
    if(j.eq.1)    zv(i,j)=z1(i,1)
    if (j.eq.nj+1) zv(i,j)=z1(i,nj)
end do
end do

c----->dz
dzacum=dz9(1)
acum=0.0d0
kk=0
do k=1,nkefe
do j=1,nj
do i=1,ni
    tz(i,j)=z1(i,j)-hz(i,j)

    if(tz(i,j).ge.dzacum)then
        dz(i,j,nk-kk)=dz9(k)
    else
        if(tz(i,j)-acum.lt.dzmin)dz(i,j,nk-kk)=0.0d0
        if(tz(i,j)-acum.ge.dzmin)dz(i,j,nk-kk)=tz(i,j)-acum
    end if

end do
end do

acum=dzacum
dzacum=dzacum+dz9(k+1)

```

```

        kk=kk+1

    end do

c----->nnkz
    do j=1,nj
        do i=1,ni
            kk=0
            if (dz(i,j,nk).ne.0.0d0) then
                do k=nk,1,-1
                    if(dz(i,j,k).ne.0d0) kk=kk+1
                end do
                nnkz(i,j)=nk-kk+1
            else
                nnkz(i,j)=99
            end if
        end do
    end do

c----->dzu
    dzacum=dz9(1)
    acum=0.0
    kk=0
    do k=1,nkefe
        do j=1,nj
            do i=1,ni
                tu(i,j)=zu(i,j)-hu(i,j)

                if(tu(i,j).ge.dzacum)then
                    dzu(i,j,nk-kk)=dz9(k)
                else
                    if(tu(i,j)-acum.lt.dzmin)dzu(i,j,nk-kk)=0.0d0
                    if(tu(i,j)-acum.ge.dzmin)dzu(i,j,nk-kk)=tu(i,j)-acum
                end if
            end do
        end do
    end do

```

```

end do

acum=dzacum
dzacum=dzacum+dz9(k+1)
kk=kk+1
end do

c----->nnku
do j=1,nj
do i=1,ni
kk=0
if (dzu(i,j,nk).ne.0d0) then
do k=nk,1,-1
if(dzu(i,j,k).ne.0d0) kk=kk+1
end do
nnku(i,j)=nk-kk+1
else
nnku(i,j)=99
end if
end do
end do

c----->dzv
dzacum=dz9(1)
acum=0.0
kk=0
do k=1,nkefe
do j=1,nj
do i=1,ni
tv(i,j)=zv(i,j)-hv(i,j)

if(tv(i,j).ge.dzacum)then
dzv(i,j,nk-kk)=dz9(k)
else
if(tv(i,j)-acum.lt.dzmin)dzv(i,j,nk-kk)=0.0d0
if(tv(i,j)-acum.ge.dzmin)dzv(i,j,nk-kk)=tv(i,j)-acum

```



```

        end if

        end do
    end do

    acum=dzacam
    dzacam=dzacam+dz9(k+1)
    kk=kk+1
end do

c----->nnkv
do j=1,nj
do i=1,ni
    kk=0
    if (dzv(i,j,nk).ne.0d0) then
        do k=nk,1,-1
            if(dzv(i,j,k).ne.0d0) kk=kk+1
        end do
        nnkv(i,j)=nk-kk+1
    else
        nnkv(i,j)=99
    end if
end do
end do

300 format (1000e10.2)

end subroutine caldz2

c
*****
*****
subroutine genivf

```

```

c
*****
*****
c subroutine que calcula los diferentes dz
  integer :: i, j, k

c----->ivf dans 99
  do i=1,ni
    do j=1,nj
      do k=1,nk
        if(dz(i,j,k).eq.0) then
          ivf(i,j,k)=99    !significa que es tierra
        else
          ivf(i,j,k)=1    !significa que la celda tiene agua
        end if
      end do
    end do
  end do

c----->ivfu et ivfv dans 99
  do i=1,ni
    do j=1,nj
      do k=1,nk
        if(ivf(i,j,k).eq.99) then
          ivfu(i,j,k)=99
          ivfu(i+1,j,k)=99
          ivfv(i,j,k)=99
          ivfv(i,j+1,k)=99
        end if
      end do
    end do
  end do

  do i=1,ni
    do j=1,nj
      do k=1,nk

```

```

        if(ivfu(i,j,k).ne.99)ivfu(i,j,k)=1
        if(ivfv(i,j,k).ne.99)ivfv(i,j,k)=1
    end do
end do
end do

```

```

c----->dzu et dzv dans nk+1
do i=1,ni
do j=1,nj
dz(i,j,nk+1)=dz(i,j,nk)
dzu(i,j,nk+1)=dzu(i,j,nk)
dzv(i,j,nk+1)=dzv(i,j,nk)
end do
end do

```

```

c===== Generalisation ivfu & ivfv =====
c----- Orlanski's (1976) Condition -----
        call ivfright_fp
c=====
==

```

```

c-----> imprime ivf
        open(17, file ='ivf.dat')
write(17,*)'ivf'
do k=nk,1,-1
write(17,*)'ivf capa k=',k
do j=1,nj
write(17,200) (ivf(i,j,k),i=1,ni)
end do
end do

```

```

c-----> imprime ivfu ou ivfv
        open(18, file ='ivfu.dat')
write(18,*)'ivfu'
do k=nk,1,-1

```

```

write(18,*)'ivfu capa k=',k
do j=1,nj
write(18,200) (ivfu(i,j,k),i=1,ni)
end do
end do
c-----> imprime ivfu ou ivfv
open(19, file ='ivfv.dat')
write(19,*)'ivfv'
do k=nk,1,-1
write(19,*)'ivfv capa k=',k
do j=1,nj
write(19,200) (ivfv(i,j,k),i=1,ni)
end do
end do

```

```

100 format (200i4,1f8.2)
200 format (225i4)
300 format(1000e10.2)

```

```

end subroutine genivf

```

```

c=====
=====

```

```

c===== SUBROUTINE IVFU & IVFV DE DROITE
=====

```

```

subroutine ivfright_fp

```

```

c=====
=====

```

```

integer :: i, j, k

```

```

c----->ivf dans 88 pour z1

```

```

c_____ entrée

```

```

do k=1,nk

    do j=jkan1,jkan2
        ivf(ni+1,j,k)=88
        ivf(ni,j,k)=88
    enddo

    do i=iatb1,iatb2
        ivf(i,nj+1,k)=88
        ivf(i,nj,k)=88
    enddo

c_____ sortir-OPEN SEA
    do i=ikib1,ikib2
        ivf(i,1,k)=88
        ivf(i,2,k)=88
    enddo

c_____ VON NEUMANN
    do j=jkia1,jkia2
        ivf(1,j,k)=88
    enddo

    do j=jkan3,jkan4
        ivf(ni+1,j,k)=88
    enddo

enddo

c----->ivfu dan ivfv dans 88
c_____ sortir_free
    do k=1,nk
    do j=jkia1,jkia2
        ivfu(1,j,k) =88
    enddo

```

```
do j=jkia1+1,jkia2
  ivfv(1,j,k) =88
enddo
enddo
```

```
do k=1,nk
do j=jkan3,jkan4
  ivfu(ni+1,j,k) =88
enddo
```

```
do j=jkan3+1,jkan4
  ivfv(ni+1,j,k) =88
enddo
enddo
```

c----->ivfu dan ivfv dans 77

```
c_____ entrée
do k=1,nk
do i=iatb1,iatb2
  ivfv(i,nj+1,k) =77
  ivfv(i,nj,k) =77
enddo
```

```
do i=iatb1+1,iatb2
  ivfu(i,nj+1,k) =77
  ivfu(i,nj,k) =77
enddo
enddo
```

```
do k=1,nk
do j=jkan1,jkan2
  ivfu(ni+1,j,k) =77
  ivfu(ni,j,k) =77
enddo
```

```
do j=jkan1+1,jkan2
```

```

        ivfv(ni+1,j,k) =77
        ivfv(ni,j,k) =77
    enddo
enddo

c_____ sortir
do k=1,nk
do i=ikib1,ikib2
    ivfv(i,1,k) =77
    ivfv(i,2,k) =77
enddo

do i=ikib1+1,ikib2
    ivfu(i,1,k) =77
    ivfu(i,2,k) =77
enddo
enddo

end subroutine ivfright_fp

c=====
=====

c
*****
*****
    subroutine fronte
c
*****
*****
    integer :: i,j,k

c----->vitesse 99
do k=1,nk

```

```

do j=1,nj
do i=1,ni
  if(ivfu(i,j,k).eq.99) then
    u(i,j,k)=0.0d0
  endif

  if(ivfv(i,j,k).eq.99) then
    v(i,j,k)=0.0d0
  endif

  if(ivf(i,j,k).eq.99) then
    w(i,j,k)=0.0d0
    w(i,j,k+1)=0.0d0
  endif

end do
end do
end do

```

```

c----->vitesse 88
do k=1,nk
do j=1,nj
do i=1,ni
c      para el sentido x
  if(ivfu(i+1,j,k).eq.88.and.ivfu(i,j,k).eq.1) then
    u(i+1,j,k)=u(i,j,k)
  endif

  if(ivfu(i+1,j,k).eq.1.and.ivfu(i,j,k).eq.88) then
    u(i,j,k)=u(i+1,j,k)
  endif

  if(ivfv(i+1,j,k).eq.88.and.ivfv(i,j,k).eq.1) then
    v(i+1,j,k)=v(i,j,k)
  endif

```



```

    if(ivfv(i+1,j,k).eq.1.and.ivfv(i,j,k).eq.88) then
      v(i,j,k)=v(i+1,j,k)
    endif

c    para el sentido y
    if(ivfv(i,j+1,k).eq.88.and.ivfv(i,j,k).eq.1) then
      v(i,j+1,k)=v(i,j,k)
    endif

    if(ivfv(i,j+1,k).eq.1.and.ivfv(i,j,k).eq.88) then
      v(i,j,k)=v(i,j+1,k)
    endif

    if(ivfu(i,j+1,k).eq.88.and.ivfu(i,j,k).eq.1) then
      u(i,j+1,k)=u(i,j,k)
    endif

    if(ivfu(i,j+1,k).eq.1.and.ivfu(i,j,k).eq.88) then
      u(i,j,k)=u(i,j+1,k)
    endif

    end do
  end do
end do

c----->elevation 88 et 99
do j=1,nj
do i=1,ni
  if(ivf(i,j,nk).eq.88.and.ivf(i+1,j,nk).eq.1) then
    z1(i,j)=z1(i+1,j)
  end if

  if(ivf(i,j,nk).eq.1.and.ivf(i+1,j,nk).eq.88) then
    z1(i+1,j)=z1(i,j)
  end if

```

```

    if(ivf(i,j,nk).eq.88.and.ivf(i,j+1,nk).eq.1) then
        z1(i,j)=z1(i,j+1)
    end if

    if(ivf(i,j,nk).eq.1.and.ivf(i,j+1,nk).eq.88) then
        z1(i,j+1)=z1(i,j)
    end if

    if(ivf(i,j,nk).eq.99) then
        z1(i,j)=0.0d0
    end if

end do
end do
c=====
=====
        call fronteright
c=====
=====
        end subroutine fronte
c=====
=====

c===== SUBROUTINE FRONTE DE DROIT
=====
        subroutine fronteright
c=====
=====
        integer :: i,j,k,ka,kb,nkp,nkr,nkk,kk
        integer,dimension(:),allocatable :: itera,iterb
        real(kind=8),dimension(:),allocatable :: uva,vva,uvb,vvb

c===== data entrée debit et calcul freepass =====
c----- daerah kiri -----
        aa=0.0d0
        do j=jkia1,jkia2

```

```

aa=aa+dy(j)*(z1(1,j)-hz(1,j)+0.5d0)
enddo
c----- daerah bawah -----
ab=0.0d0
do i=ikib1,ikib2
ab=ab+dx(i)*(z1(i,1)-hz(i,1)+0.5d0)
enddo
c----- daerah atas -----
ag=0.0d0
do i=iatb1,iatb2
ag=ag+dx(i)*(z1(i,nj)-hz(i,nj)+0.5d0)
enddo
c----- daerah kanan -----
ai=0.0d0
do j=jkan1,jkan2
ai=ai+dy(j)*(z1(ni,j)-hz(ni,j)+0.5d0)
enddo

c      debit masuk
vtg=(debmas*ag/(ag+ai))/ag
uti=(debmas*ai/(ag+ai))/ai
uta=0.0d0          !((debmas*aa/(ag+ai+aa))/aa)
vtb=debmas/ab
dg=ag*vtg
di=ai*uti
da=aa*uta
db=ab*vtb
dtot1=dg+di
if(iter.lt.iterasi.or.iter.lt.1)then
write(10,*)'KONDISI SEBELUM CALCUL'
write(10,*)'=====AWAL=====
=====
write(10,*)'debit total masuk      = ',dtot1
write(10,*)'luas awal masuk di ag  = ',ag
write(10,*)'luas awal masuk di ai  = ',ai
write(10,*)'luas awal masuk di aa  = ',aa

```

```

write(10,*)'luas awal masuk di ab      = ',ab
write(10,*)'kecepatan awal masuk di vtg = ',vtg
write(10,*)'kecepatan awal masuk di uti = ',uti
write(10,*)'kecepatan awal masuk di uta = ',uta
write(10,*)'kecepatan awal masuk di vtb = ',vtb
write(10,*)'debit awal masuk di dg     = ',dg
write(10,*)'debit awal masuk di di     = ',di
write(10,*)'debit awal masuk di da     = ',da
write(10,*)'debit awal masuk di db     = ',db
write(*,*)'KONDISI SEBELUM CALCUL'
write(*,*)'=====AWAL=====
=====
write(*,*)'debit total masuk           = ',dtot1
write(*,*)'luas awal masuk di ag       = ',ag
write(*,*)'luas awal masuk di ai       = ',ai
write(*,*)'luas awal masuk di aa       = ',aa
write(*,*)'luas awal masuk di ab       = ',ab
write(*,*)'kecepatan awal masuk di vtg = ',vtg
write(*,*)'kecepatan awal masuk di uti = ',uti
write(*,*)'kecepatan awal masuk di uta = ',uta
write(*,*)'kecepatan awal masuk di vtb = ',vtb
write(*,*)'debit awal masuk di dg      = ',dg
write(*,*)'debit awal masuk di di      = ',di
write(*,*)'debit awal masuk di da      = ',da
write(*,*)'debit awal masuk di db      = ',db
endif
c===== Panggil kecepatan masuk =====
      call masuk
c===== Panggil ORLANSKI'S Condition=====
      call fpneg1
c===== perhitungan debit masuk dan keluar =====
      call debit
c=====

end subroutine fronteright

```

```

c
=====
=
c      Kecepatan Masuk
c
=====
=
      subroutine masuk
c
=====
=
integer :: i,j,k,ka,kb,nkp,nkr,nkk,kk
integer,dimension(:),allocatable :: itera,iterb
real(kind=8),dimension(:),allocatable :: uva,vva,uvb,vvb

      do k=1,nk
        do i=iatb1,iatb2
          v(i,nj+1,k) =-vtg
          v(i,nj,k) =-vtg
          z1(i,nj+1)=z1(i,nj)
        enddo
        do i=iatb1+1,iatb2
          u(i,nj+1,k) =0.0d0
          u(i,nj,k) =0.0d0
        enddo
      enddo

      do k=1,nk
        do j=jkan1,jkan2
          u(ni+1,j,k) =-uti
          u(ni,j,k) =-uti
          z1(ni+1,j)=z1(ni,j)
        enddo
        do j=jkan1+1,jkan2
          v(ni+1,j,k) =0.0d0
          v(ni,j,k) =0.0d0
        enddo
      enddo

```

```

        enddo
    enddo

    end subroutine masuk

c=====
c=====
c===== DEBIT
c=====
c=====
c=====

    subroutine debit

        integer :: i,j,k
        real(kind=8) :: uu0,vv0,ww0

        do k=1,nk
            do i=1,ni
                do j=1,nj
                    uu0=(u(i,j,k)+u(i+1,j,k))*0.5d0
                    vv0=(v(i,j,k)+v(i,j+1,k))*0.5d0
                    ww0=(w(i,j,k)+w(i,j,k+1))*0.5d0
                    r(i,j,k)=sqrt(uu0*uu0+vv0*vv0+ww0*ww0)
                enddo
            enddo
        enddo

        c=====luas=====
c----- daerah kanan -----
            rn=0.0d0
            nr=0
        do k=1,nk
            do j=jkan1,jkan2
                rn=rn+r(ni,j,k)
                nr=nr+1
            rati=rn/DFLOAT(nr)

```

```

        enddo
    enddo
    ai=0.0d0
    do j=jkan1,jkan2
        ai=ai+dy(j)*(z1(ni,j)-hz(ni,j)+0.5d0)
    enddo
    di=ai*rati
c----- daerah kiri -----
        rn=0.0d0
        nr=0
    do k=1,nk
        do j=jkia1,jkia2
            rn=rn+r(1,j,k)
            nr=nr+1
        rata=rn/DFLOAT(nr)
        enddo
    enddo
    aa=0.0d0
    do j=jkia1,jkia2
        aa=aa+dy(j)*(z1(1,j)-hz(1,j)+0.5d0)
    enddo
    da=aa*rata
c----- daerah atas -----
        rn=0.0d0
        nr=0
    do k=1,nk
        do i=iatb1,iatb2
            rn=rn+r(i,nj,k)
            nr=nr+1
        ratg=rn/DFLOAT(nr)
        enddo
    enddo
    ag=0.0d0
    do i=iatb1,iatb2
        ag=ag+dx(i)*(z1(i,nj)-hz(i,nj)+0.5d0)
    enddo

```

```

dg=ag*ratg
c----- daerah bawah -----
      rn=0.0d0
      nr=0
do k=1,nk
  do i=ikib1,ikib2
    rn=rn+r(i,1,k)
    nr=nr+1
    ratb=rn/DFLOAT(nr)
  enddo
enddo
ab=0.0d0
do i=ikib1,ikib2
  ab=ab+dx(i)*(z1(i,1)-hz(i,1)+0.5d0)
enddo
db=ab*ratb
c===== hitung variabel debit sisi kanan (i & g)dan kiri (a &
b) =====
      dtot3=db
      dbal=dtot1-dtot3
      if(iter.lt.iterasi.or.iter.lt.1)then
        write(10,*)'=====KELUAR=====
=====
        write(10,*)'debit total keluar di db   =',dtot3
        write(10,*)'luas keluar di ab       =',ab
        write(10,*)'kecepatan keluar di ratb  =',ratb
        write(10,*)'selisih debit masuk-keluar =',dbal
        write(*,*)'=====KELUAR=====
=====
        write(*,*)'debit total keluar di db   =',dtot3
        write(*,*)'luas keluar di ab       =',ab
        write(*,*)'kecepatan keluar di ratb  =',ratb
        write(*,*)'selisih debit masuk-keluar =',dbal
      endif
c===== hitung variabel debit sisi kanan (i & g)dan kiri (a &
b) =====

```


end subroutine debit

C+++++ ORLANSKY'S METHODE
+++++

c=====

subroutine fpneg1

c=====

integer :: i,j,k,ika1,ika2,ikb1,ikb2
real(kind=8) :: rx,ry,dfit,dfix,dfiy,test,
& tao0,taoi,fic

tao0=172800.0d0 ! 1 jam

taoi=86400.0d0 ! 1 detik

c=====batas j=55 sampai 178 (lihat peta
bagian kiri bawah)

do k=1,nk
do i=ikib1+1,ikib2

fic=-vtb

dfit=v(i,3,k)-vn(i,3,k)

dfiy=v(i,3,k)-v(i,4,k)

test=dfit*(vn(i+1,3,k)-vn(i-1,3,k))

if(test.gt.0)

& dfix=vn(i,3,k)-vn(i-1,3,k)

test=dfit*(vn(i+1,3,k)-vn(i-1,3,k))

if(test.lt.0)

& dfix=vn(i+1,3,k)-vn(i,3,k)

if(dfix*dfix+dfiy*dfiy.ne.0.0d0) then

```

    ry=-(dfit*dfiy)/(dfix*dfix+dfiy*dfiy)
  else
    ry=0.0d0
  endif

  if(ry.gt.0)
&   v(i,2,k)=1.0d0/(1.0d0+ry)*(
&     vn(i,2,k)+
&     ry*v(i,3,k)+
&     (1.0d0/tao0)*(fic-vn(i,2,k))
&   )

  if(ry.le.0)
&   v(i,2,k)=vn(i,2,k)+
&     dt*(1.0d0/taoi)*(fic-vn(i,2,k))

    v(i,1,k)=v(i,2,k)
    u(i,2,k)=0.0d0
    u(i,1,k)=u(i,2,k)
    z1(i,1)=z1(i,2)
  enddo
enddo
do k=1,nk
  v(1,2,k)=v(2,2,k)
  v(1,1,k)=v(1,2,k)
  u(1,2,k)=u(2,2,k)
  u(1,1,k)=u(1,2,k)
enddo

end subroutine fpneg1
c*****
*****
subroutine prome
c*****
*****

```

```

c      *calculo de los promedios de las velocidades que se
ocuparan*
      integer :: i,j,k

c----->vpx
      do k=1,nk
        do j=1,nj
          do i=2,ni
            if(ivfu(i,j,k).eq.1) then
              vpx(i,j,k)=(v(i-1,j,k)+v(i-1,j+1,k)+
&                v(i,j,k)+v(i,j+1,k))*0.25d0
            end if
          enddo
        enddo
      enddo

      do k=1,nk
        do j=1,nj
          do i=2,ni
            if(ivfu(i,j,k).eq.88.and.
&            ivfu(i-1,j,k).eq.1) then
              vpx(i,j,k)=vpx(i-1,j,k)
            end if

            if(ivfu(i,j,k).eq.88.and.
&            ivfu(i+1,j,k).eq.1) then
              vpx(i,j,k)=vpx(i+1,j,k)
            end if

            if(ivfu(i,j,k).eq.99) then
              vpx(i,j,k)=0.0d0
            end if
          enddo
        enddo
      enddo

```

```

c----->upy
  do k=1,nk
    do j=2,nj
      do i=1,ni
        if (ivfv(i,j,k).eq.1) then
          upy(i,j,k)=(u(i,j,k)+u(i+1,j,k)+
&          u(i,j-1,k)+u(i+1,j-1,k))*0.25d0
          end if
        enddo
      enddo
    enddo

    do k=1,nk
      do j=2,nj
        do i=1,ni
          if(ivfv(i,j,k).eq.88.and.
&          ivfv(i,j-1,k).eq.1) then
            upy(i,j,k)=upy(i,j-1,k)
          endif

          if(ivfv(i,j,k).eq.88.and.
&          ivfv(i,j+1,k).eq.1) then
            upy(i,j,k)=upy(i,j+1,k)
          endif

          if(ivfv(i,j,k).eq.99) then
            upy(i,j,k)=0.0d0
          endif
        enddo
      enddo
    enddo

c----->vpz
  do j=2,nj-1
    do i=2,ni-1
      if (nnkz(i,j).ne.99) then

```

```

k=nnkz(i,j)
vpz(i,j,k)=0.0d0

do k=nnkz(i,j)+1,nk
  vpz(i,j,k)=(v(i,j,k)+v(i,j+1,k)+
&      v(i,j,k-1)+v(i,j+1,k-1))*0.25d0
enddo

k=nk+1
vpz(i,j,k)=(v(i,j,k-1)+v(i,j+1,k-1))*0.5d0
endif
enddo
enddo

c----->upz
do j=2,nj-1
  do i=2,ni-1
    if (nnkz(i,j).ne.99) then
      k=nnkz(i,j)
      upz(i,j,k)=0.0d0

      do k=nnkz(i,j)+1,nk
        upz(i,j,k)=(u(i,j,k)+u(i+1,j,k)+
&      u(i,j,k-1)+u(i+1,j,k-1))*0.25d0
      end do

      k=nk+1
      upz(i,j,k)=(u(i,j,k-1)+u(i+1,j,k-1))*0.5d0
    end if
  end do
end do

c----->wpx
do k=1,nk
  do j=1,nj
    do i=2,ni

```

```

        if (ivfu(i,j,k).eq.1) then
            wpx(i,j,k)=(w(i-1,j,k)+w(i-1,j,k+1)+
&                w(i,j,k)+w(i,j,k+1))*0.25d0
            endif
        enddo
    enddo
enddo

do k=1,nk
    do j=1,nj
        do i=2,ni
            if(ivfu(i,j,k).eq.88.and.
&            ivfu(i-1,j,k).eq.1) then
                wpx(i,j,k)=wpx(i-1,j,k)
            endif

            if(ivfu(i,j,k).eq.88.and.
&            ivfu(i+1,j,k).eq.1) then
                wpx(i,j,k)=wpx(i+1,j,k)
            endif

            if(ivfu(i,j,k).eq.99) then
                wpx(i,j,k)=0.0d0
            endif
        enddo
    enddo
enddo

c----->wpy
do k=1,nk
    do j=2,nj
        do i=1,ni
            if (ivfv(i,j,k).eq.1) then
                wpy(i,j,k)=(w(i,j,k)+w(i,j,k+1)+
&                w(i,j-1,k)+w(i,j-1,k+1))*0.25d0
            endif
        enddo
    enddo
enddo

```

```

        enddo
    enddo
enddo

do k=1,nk
do j=2,nj
do i=1,ni
    if(ivfv(i,j,k).eq.88.and.
&      ivfv(i,j-1,k).eq.1) then
        wpy(i,j,k)=wpy(i,j-1,k)
    endif

    if(ivfv(i,j,k).eq.88.and.
&      ivfv(i,j+1,k).eq.1) then
        wpy(i,j,k)=wpy(i,j+1,k)
    endif

    if(ivfv(i,j,k).eq.99) then
        wpy(i,j,k)=0.0d0
    endif
enddo
enddo
enddo

```

300 format(65e8.1)

end subroutine prome

```

c
*****
*****
    subroutine modlm1
c
*****
*****
c -----

```

```

c se calcula el coeficiente de mezclado vertical
c modelo de longitud de mezcla
c basado en "computational hydraulics vol ii", pag 83
c  $c_{nut} = l^2 \sqrt{du/dz \cdot du/dz + dv/dz \cdot dv/dz}$ 
c  $l = k_p \cdot z$  si  $z < \delta$   $k_p = 0.4$ 
c  $l = k_p \cdot \delta$  si  $z > \delta$   $\delta = .2 \cdot \text{tirante}$ 
c-----
integer :: i, j, k
real(kind=8) :: alfa,lv,zeta,delta,duz2,dvz2,duz,dvz,
& denozu,denozv

alfa=0.8d0

c nuzeta en sentido longitudinal
do j=2,nj-1
do i=2,ni-1
c write (*,*) 'nuzeta longitudinal', i, j
if (ivfu(i,j,nk).ne.1) then
do k=1,nk
cnueu(i,j,k)=0.0d0
end do
else
delta=zu(i,j)-alfa*tu(i,j)-hu(i,j)
do k=1,nnku(i,j)
cnueu(i,j,k)=0.0d0
end do
do k=nnku(i,j)+1,nk-1
denozu=dzu(i,j,k)+0.5d0*(dzu(i,j,k+1)+dzu(i,j,k-1))
duz=(u(i,j,k+1)-u(i,j,k-1))/denozu
dvz=(vpx(i,j,k+1)-vpx(i,j,k-1))/denozu
zeta=zu(i,j)-dzu(i,j,k)*0.5d0-(nk-k)*dz0-hu(i,j)
if(zeta.lt.delta) lv=kp*zeta
if(zeta.ge.delta) lv=kp*delta
duz2=duz*duz
dvz2=dvz*dvz
cnueu(i,j,k)=lv*lv*sqrt(duz2+dvz2)

```



```

end do
c nuzeta en el elemento superior k=nk
denozu=0.5d0*(dzu(i,j,nk)+dzu(i,j,nk-1))
duz=(u(i,j,nk)-u(i,j,nk-1))/denozu
dvz=(vpx(i,j,nk)-vpx(i,j,nk-1))/denozu
zeta=zu(i,j)-dzu(i,j,nk)*0.5d0-hu(i,j)
if(zeta.lt.delta) lv=kp*zeta
if(zeta.ge.delta) lv=kp*delta
duz2=duz*duz
dvz2=dvz*dvz
cnueu(i,j,nk)=lv*lv*sqrt(duz2+dvz2)
c nuzeta en el elemento inferior liquido k=nnku(i,j)
k=nnku(i,j)
denozu=0.5d0*(dzu(i,j,k+1)+dzu(i,j,k))
duz=(u(i,j,k+1)-u(i,j,k))/denozu
dvz=(vpx(i,j,k+1)-vpx(i,j,k))/denozu
zeta=zu(i,j)-dzu(i,j,k)*0.5d0-(nk-k)*dz0-hu(i,j)
if(zeta.lt.delta) lv=kp*zeta
if(zeta.ge.delta) lv=kp*delta
duz2=duz*duz
dvz2=dvz*dvz
cnueu(i,j,k)=lv*lv*sqrt(duz2+dvz2)
end if
end do
end do

c nuzeta en el sentido transversal
do j=2,nj-1
do i=2,ni-1
if (ivfv(i,j,nk).ne.1) then
do k=1,nk
cnuev(i,j,k)=0.0d0
end do
else
delta=zv(i,j)-alfa*tv(i,j)-hv(i,j)
c write (*,*) 'nuzeta transversal', i ,j, delta

```

```

do k=1,nnkv(i,j)
  cnuev(i,j,k)=0.0d0
end do
do k=nnkv(i,j)+1,nk-1
  denozv=dzv(i,j,k)+0.5d0*(dzv(i,j,k+1)+dzv(i,j,k-1))
  dvz=(v(i,j,k+1)-v(i,j,k-1))/denozv
  duz=(upy(i,j,k+1)-upy(i,j,k-1))/denozv
  zeta=zv(i,j)-dzv(i,j,k)*0.5d0-(nk-k)*dz0-hv(i,j)
  if(zeta.lt.delta) lv=kp*zeta
  if(zeta.ge.delta) lv=kp*delta
  duz2=duz*duz
  dvz2=dvz*dvz
  cnuev(i,j,k)=lv*lv*sqrt(duz2+dvz2)
end do
c nuzeta en el elemento superior k=nk
denozv=0.5d0*(dzv(i,j,nk)+dzv(i,j,nk-1))
dvz=(v(i,j,nk)-v(i,j,nk-1))/denozv
duz=(upy(i,j,nk)-upy(i,j,nk-1))/denozv
zeta=zv(i,j)-dzv(i,j,nk)*0.5d0-hv(i,j)
if(zeta.lt.delta) lv=kp*zeta
if(zeta.ge.delta) lv=kp*delta
duz2=duz*duz
dvz2=dvz*dvz
cnuev(i,j,nk)=lv*lv*sqrt(duz2+dvz2)
c nuzeta en el elemento inferior liquido k=nnku(i,j)
k=nnkv(i,j)
denozv=0.5d0*(dzv(i,j,k+1)+dzv(i,j,k))
dvz=(v(i,j,k+1)-v(i,j,k))/denozv
duz=(upy(i,j,k+1)-upy(i,j,k))/denozv
zeta=zv(i,j)-dzv(i,j,k)*0.5d0-(nk-k)*dz0-hv(i,j)
if(zeta.lt.delta) lv=kp*zeta
if(zeta.ge.delta) lv=kp*delta
duz2=duz*duz
dvz2=dvz*dvz
cnuev(i,j,k)=lv*lv*sqrt(duz2+dvz2)
end if

```

```

        end do
    end do

c    calculo del coeficiente de viscosidad turbulenta al centro de
la celda
    do k=2,nk
        do j=2,nj-1
            do i=2,ni-1
                cnue(i,j,k)=(cnueu(i,j,k)+cnueu(i+1,j,k)
&                +cnuev(i,j,k)+cnuev(i,j+1,k))*0.25d0
            end do
        end do
    end do

c    calculo de los operadores fu y fv
c    por medio del modelo de longitud de mezclado
    call opfulm
    call opfvlm

110 format (3i4,5f8.2)
500 format(100e20.10)

    end subroutine modlm1
c
*****
*****
        subroutine modlm2
c
*****
*****
c -----
c se calcula el coeficiente de mezclado vertical
c modelo de longitud de mezcla
c basado en "Lagrangian Scheme for 3D Shallow Water Flow"
pag 119

```

```

c cnut=coeficiente viscoso turbulento
c cnum=coeficiente viscoso molecular
c cnut=cnutm+l**2.*
c l=kp*z    si    z < alfa
c l=lambda*h  si alfa < z < h
c where
c alfa=lambda*h/kp
c lambda = 0.09
c kp = 0.43
c-----
integer    :: i, j, k
real(kind=8) :: lv,lambda,zeta,alfa,denozu,duz,dvz,
&          denozv

lambda=0.09d0

c  en sentido longitudinal
do i=1,ni
do j=1,nj
if (ivfu(i,j,nk).ne.1) then
do k=1,nk
cneu(i,j,k)=cnum
end do
else
do k=1,nnku(i,j)
cneu(i,j,k)=cnum
end do
alfa=(lambda*tu(i,j))/kp

do k=nnku(i,j)+1,nk-1
denozu=dzu(i,j,k)+0.5d0*(dzu(i,j,k+1)+dzu(i,j,k-1))
duz=(u(i,j,k+1)-u(i,j,k-1))/denozu
dvz=(vpx(i,j,k+1)-vpx(i,j,k-1))/denozu
zeta=zu(i,j)-dzu(i,j,k)*0.5d0-(nk-k)*dz0-hu(i,j)
if(zeta.lt.alfa) lv=kp*zeta
if(zeta.ge.alfa) lv=lambda*tu(i,j)

```

```

&      if((u(i,j,k)*u(i,j,k)+
&          vpx(i,j,k)*vpx(i,j,k)).gt.0) then
&          cnueu(i,j,k)=cnum + lv*lv
&          *abs((u(i,j,k)*duz+vpx(i,j,k)*dvz)
&              *(1.0d0/(sqrt(u(i,j,k)*u(i,j,k)
&                  +vpx(i,j,k)*vpx(i,j,k))))))
&      else
&          cnueu(i,j,k)=cnum
&      end if
end do

c  en el elemento superior k=nk
denozu=dzu(i,j,nk)+0.5d0*(dzu(i,j,nk)+dzu(i,j,nk-1))
duz=(u(i,j,nk)-u(i,j,nk-1))/denozu
dvz=(vpx(i,j,nk)-vpx(i,j,nk-1))/denozu
zeta=z(u(i,j))-dzu(i,j,nk)*0.5d0-hu(i,j)
if(zeta.lt.alfa) lv=kp*zeta
if(zeta.ge.alfa) lv=lambda*tu(i,j)

&      if((u(i,j,nk)*u(i,j,nk)+
&          vpx(i,j,nk)*vpx(i,j,nk)).gt.0d0) then
&          cnueu(i,j,nk)=cnum + lv*lv
&          *abs((u(i,j,nk)*duz+vpx(i,j,nk)*dvz)
&              *(1.0d0/(sqrt(u(i,j,nk)*u(i,j,nk)
&                  +vpx(i,j,nk)*vpx(i,j,nk))))))
&      else
&          cnueu(i,j,nk)=cnum
&      end if

c  en el elemento inferior liquido k=nnku(i,j)
k=nnku(i,j)
denozu=dzu(i,j,k)+0.5d0*(dzu(i,j,k+1)+dzu(i,j,k))
duz=(u(i,j,k+1)-(-u(i,j,k)))/denozu
dvz=(vpx(i,j,k+1)-(-vpx(i,j,k)))/denozu
zeta=z(u(i,j))-dzu(i,j,k)*0.5d0-(nk-k)*dz0-hu(i,j)

```

```

if(zeta.lt.alfa) lv=kp*zeta
if(zeta.ge.alfa) lv=lambda*tu(i,j)

if((u(i,j,k)*u(i,j,k)+
& vpx(i,j,k)*vpx(i,j,k)).gt.0d0) then
  cnueu(i,j,k)=cnum + lv*lv
& *abs((u(i,j,k)*duz+vpx(i,j,k)*dvz)
& *(1.0d0/(sqrt(u(i,j,k)*u(i,j,k)
& +vpx(i,j,k)*vpx(i,j,k))))))
else
  cnueu(i,j,k)=cnum
end if
end if
end do
end do

```

```

c en el sentido transversal
do i=1,ni
do j=1,nj
if (ivfv(i,j,nk).ne.1) then
do k=1,nk
cnuev(i,j,k)=cnum
end do
else
do k=1,nnkv(i,j)
cnuev(i,j,k)=cnum
end do
alfa=(lambda*tv(i,j))/kp

do k=nnkv(i,j)+1,nk-1
denozv=dzv(i,j,k)+0.5d0*(dzv(i,j,k+1)+dzv(i,j,k-1))
duz=(upy(i,j,k+1)-upy(i,j,k-1))/denozv
dvz=(v(i,j,k+1)-v(i,j,k-1))/denozv
zeta=zv(i,j)-dzv(i,j,k)*0.5d0-(nk-k)*dz0-hv(i,j)
if(zeta.lt.alfa) lv=kp*zeta
if(zeta.ge.alfa) lv=lambda*tv(i,j)

```

```

&      if((upy(i,j,k)*upy(i,j,k)+
&          v(i,j,k)*v(i,j,k)).gt.0d0) then
&          cnuev(i,j,k)=cnum + lv*lv
&          *abs((upy(i,j,k)*duz+v(i,j,k)*dvz)
&              *(1.0d0/(sqrt(upy(i,j,k)*upy(i,j,k)
&                  +v(i,j,k)*v(i,j,k))))))
&      else
&          cnuev(i,j,k)=cnum
&      end if
end do

c  en el elemento superior k=nk
denozv=dzv(i,j,nk)+0.5d0*(dzv(i,j,nk)+dzv(i,j,nk-1))
duz=(upy(i,j,nk)-upy(i,j,nk-1))/denozv
dvz=(v(i,j,nk)-v(i,j,nk-1))/denozv
zeta=zv(i,j)-dzv(i,j,nk)*0.5d0-hv(i,j)
if(zeta.lt.alfa) lv=kp*zeta
if(zeta.ge.alfa) lv=lambda*tv(i,j)

&      if((upy(i,j,nk)*upy(i,j,nk)+
&          v(i,j,nk)*v(i,j,nk)).gt.0d0) then
&          cnuev(i,j,nk)=cnum + lv*lv
&          *abs((upy(i,j,nk)*duz+v(i,j,nk)*dvz)
&              *(1.0d0/(sqrt(upy(i,j,nk)*upy(i,j,nk)
&                  +v(i,j,nk)*v(i,j,nk))))))
&      else
&          cnuev(i,j,nk)=cnum
&      end if

c  en el elemento inferior liquido k=nnkv(i,j)
k=nnkv(i,j)
denozv=dzv(i,j,k)+0.5d0*(dzv(i,j,k+1)+dzv(i,j,k))
duz=(upy(i,j,k+1)-(-upy(i,j,k)))/denozv
dvz=(v(i,j,k+1)-(-v(i,j,k)))/denozv
zeta=zv(i,j)-dzv(i,j,k)*0.5d0-(nk-k)*dz0-hv(i,j)

```

```

        if(zeta.lt.alfa) lv=kp*zeta
        if(zeta.ge.alfa) lv=lambda*tv(i,j)

        if((upy(i,j,k)*upy(i,j,k)+
&          v(i,j,k)*v(i,j,k)).gt.0) then
          cnuev(i,j,k)=cnum + lv*lv
&          *abs((upy(i,j,k)*duz+v(i,j,k)*dvz)
&          *(1.0d0/(sqrt(upy(i,j,k)*upy(i,j,k)
&          +v(i,j,k)*v(i,j,k))))))
        else
          cnuev(i,j,k)=cnum
        end if
      end if
    end do
  end do
end do

```

c calculo del coeficiente de viscosidad turbulenta al centro de la celda

```

    do i=1,ni
      do j=1,nj
        do k=1,nk
          cnue(i,j,k)=(cnueu(i,j,k)+cnueu(i+1,j,k)
&          +cnuev(i,j,k)+cnuev(i,j+1,k))*0.25d0
        end do
      end do
    end do
end do

```

c Primer Criterio

c la viscosidad turbulenta dentro de la derivadas

```
call opfulm
```

```
call opfvlm
```

```
110 format (3i4,5f8.2)
```

```
end subroutine modlm2
```



```

c
*****
*****
      subroutine modlm2d
c
*****
*****
c -----
c  Subrutina NUEVA
c  modelo de longitud de mezclado en DOS dimensiones
c  (promediado en la vertical)
c  Limitations of depth-average modelling for shallow wakes
c  Peter Stansby
c  para 2D
c
vt=sqrt(lh**4*(2*(du/dx)**2+2*(dv/dy)**2+(dv/dx+du/dy)**2)
+(gama*Uetoil*h)**2)
c  where:
c  gama=0.067                               =>coef de Elder
c  Uetoil=sqrt(|taub|/dens)                   =>vitesse al cortante
c          (taubx,tauby)=dens*Cf*(u,v)*sqrt(u**2+v**2)
=>contrainte
c  Cf=0.0559/Reh**0.25                       =>coef de rugosidad
c  Reh=sqrt(u**2+v**2)*h/cnum                 =>Reynolds
c
c -----
      integer :: i, j, k, im1, jm1
      real(kind=8) :: beta,lambda,gama,up,vp,lh,cf,reh,taubx,
&          tauby,uetoil,dudx,dvdy,dvdx,dudy,cnut,
&          velp,rehl

c  Constantes:
      beta=1.d0
      lambda=0.09d0
      gama=0.067d0

```

```

do k=1,nk
  do j=1,nj
    do i=1,ni

      if (ivf(i,j,k).eq.1) then
        up=(u(i,j,k)+u(i+1,j,k))*0.5d0
        vp=(v(i,j,k)+v(i,j+1,k))*0.5d0
        lh=beta*lambda*tz(i,j)

        cf=(2.d0*g)/(chezy*chezy)

        velp = up*up+vp*vp
        if(velp.gt.1.d-50) then
          reh=(sqrt(velp)*tz(i,j))/cnum

          cf=0.0559d0/((reh)**0.25d0)
        endif
        taubx=cf*up*sqrt(up*up+vp*vp)
        tauby=cf*vp*sqrt(up*up+vp*vp)

        uetoil=sqrt(sqrt(taubx*taubx+tauby*tauby))

c-----dudx
        dudx=(u(i+1,j,k)-u(i,j,k))/dx(i)

c-----dvdy
        dvdy=(v(i,j+1,k)-v(i,j,k))/dy(j)

c-----dvdx
        dvdx=(vpx(i+1,j,k)-vpx(i,j,k))/dx(i)

c-----dudy
        dudy=(upy(i,j+1,k)-upy(i,j,k))/dy(j)

c-----cnut
        cnut=sqrt(lh**4*(2.d0*dudx*dudx+2.d0*dvdy*dvdy)

```

```

&          +(dvdxdudy)*(dvdxdudy))
&          +(gama*uetoil*tz(i,j))
&          *(gama*uetoil*tz(i,j))
&          )
c-----cnue
      cnue(i,j,k)=cnum + cnut
      else
      cnue(i,j,k)=cnum
      end if
      end do
      end do
      end do

do k=1,nk
do j=1,nj
do i=1,ni
if (ivf(i,j,k).eq.1.and.ivf(i+1,j,k).eq.88)
&   cnue(i+1,j,k)=cnue(i,j,k)

if (ivf(i,j,k).eq.88.and.ivf(i+1,j,k).eq.1)
&   cnue(i,j,k)=cnue(i+1,j,k)

if (ivf(i,j,k).eq.1.and.ivf(i,j+1,k).eq.88)
&   cnue(i,j+1,k)=cnue(i,j,k)

if (ivf(i,j,k).eq.88.and.ivf(i,j+1,k).eq.1)
&   cnue(i,j,k)=cnue(i,j+1,k)

if (ivf(i,j,k).eq.99)cnue(i,j,k)=cnum
end do
end do
end do

c   los cnut centrados en u, v, w

```

```

c.....para cnutu.....
do k=1,nk
do j=1,nj
do i=2,ni
if (ivfu(i,j,k).eq.1) then
im1=i-1
if(im1.le.0)im1=1
cnueu(i,j,k)=(cnue(i-1,j,k)*dx(im1)*0.5d0+
& cnue(i,j,k)*dx(i)*0.5d0)/
& (dx(i)*0.5d0+dx(im1)*0.5d0)
end if
end do
end do
end do

```

```

do k=1,nk
do j=1,nj
do i=1,ni
if(ivfu(i,j,k).eq.99) cnueu(i,j,k)=cnum
end do
end do
end do

```

```

c.....para cnutv.....
do k=1,nk
do j=2,nj
do i=1,ni
if (ivfv(i,j,k).eq.1) then
jm1=j-1
if(jm1.le.0)jm1=1
cnuev(i,j,k)=(cnue(i,j-1,k)*dy(jm1)*0.5d0+
& cnue(i,j,k)*dy(j)*0.5d0)/
& (dy(j)*0.5d0+dy(jm1)*0.5d0)
end if
end do
end do

```

```

        end do
    end do

    do k=1,nk
        do j=1,nj
            do i=1,ni
                if(ivfv(i,j,k).eq.99) cnuev(i,j,k)=cnum
            end do
        end do
    end do

c.....para cnutw.....
    do j=1,nj
        do i=1,ni
            do k=2,nk
                if (ivf(i,j,k).eq.1) then
                    cnuew(i,j,k)=(cnue(i,j,k-1)*dz(i,j,k-1)*0.5d0+
&                    cnue(i,j,k)*dz(i,j,k)*0.5d0)/
&                    (dz(i,j,k-1)*0.5d0+dz(i,j,k)*0.5d0)
                else
                    cnuew(i,j,k)=cnum
                end if
            end do

            if (ivf(i,j,nk).eq.1)then
cnuew(i,j,nk+1)=(cnue(i,j,nk)*dz(i,j,nk)*0.5d0+
&                cnue(i,j,nk)*dz(i,j,nk)*0.5d0)/
&                (dz(i,j,nk)*0.5d0+dz(i,j,nk)*0.5d0)
            endif
        end do
    end do

c    Primer Criterio: (la viscosidad turbulenta dentro de la
derivadas)
    call opfulm

```

```

call opfvlm

110 format (3i4,5f8.2)
500 format(100e20.10)

end subroutine modlm2d

c
*****
*****
      subroutine modlm3d
c
*****
*****
c -----
c  Subrutina NUEVA
c  modelo de longitud de mezclado en DOS dimensiones
c  (promediado en la vertical)
c  Limitations of depth-average modelling for shallow wakes
c  Peter Stansby
c -----
c  para calcular l (longitud de mezclado)
c  para 3D
c
vt=sqrt(lh**4*(2*(du/dx)**2+2*(dv/dy)**2+(dv/dx+du/dy)**2)
+lv**4*((du/dz)**2+(dv/dz)**2))
c  lh=beta*lv
c  lv=kp*z      si    z < alfa
c  lv=lambda*h  si alfa < z < h
c  where:
c  alfa=lambda*h/kp
c  lambda = 0.09
c  kp = 0.43
c -----

```

```

integer    :: i, j, k, mm, im1, jm1
real(kind=8) :: beta, lambda, lh, up, vp, wp, lv, alfa, zeta, coefsma,
&
dudx, dvdy, dvdx, dudy, dudz, dvdz, cnut, kp, reh, rehl, velp

```

```

c  Constantes:

```

```

beta =6.d0

```

```

lambda=0.09d0

```

```

kp=0.41d0

```

```

c    coefsma=0.1d0

```

```

do k=1,nk

```

```

  do j=1,nj

```

```

    do i=1,ni

```

```

      if (ivf(i,j,k).eq.1) then

```

```

        alfa=(lambda*tz(i,j))/kp

```

```

        zeta=0.0d0

```

```

        do mm=1,k

```

```

          zeta=zeta+dz(i,j,mm)

```

```

        enddo

```

```

        zeta=zeta-dz(i,j,k)*0.5d0

```

```

        if(zeta.lt.alfa) lv=kp*zeta

```

```

        if(zeta.ge.alfa) lv=lambda*tz(i,j)

```

```

        lh=beta*lv

```

```

c-----dudx

```

```

  dudx=(u(i+1,j,k)-u(i,j,k))/dx(i)

```

```

c-----dvdy

```

```

  dvdy=(v(i,j+1,k)-v(i,j,k))/dy(j)

```

```

c-----dvdx

```

```

        dwdx=(vpx(i+1,j,k)-vpx(i,j,k))/dx(i)
c-----dudy
        dudy=(upy(i,j+1,k)-upy(i,j,k))/dy(j)
c-----dudz
        dudz=(upz(i,j,k+1)-upz(i,j,k))/dz(i,j,k)
c-----dvdz
        dvdz=(vpz(i,j,k+1)-vpz(i,j,k))/dz(i,j,k)
c-----cnum
        cnum=sqrt(lh**4*(2.d0*dwdx*dwdx+2.d0*dvdy*dvdy
&          +(dwdx+dudy)*(dwdx+dudy))
&          +lv**4*(dudz*dudz+dvdz*dvdz))

        cnue(i,j,k)=cnum + cnut

    else
        cnue(i,j,k)=cnum
    end if
end do
end do
end do

do k=1,nk
do j=1,nj
do i=1,ni
    if (ivf(i,j,k).eq.1.and.ivf(i+1,j,k).eq.88)
&    cnue(i+1,j,k)=cnue(i,j,k)

    if (ivf(i,j,k).eq.88.and.ivf(i+1,j,k).eq.1)
&    cnue(i,j,k)=cnue(i+1,j,k)

    if (ivf(i,j,k).eq.1.and.ivf(i,j+1,k).eq.88)

```



```

&   cnue(i,j+1,k)=cnue(i,j,k)

      if (ivf(i,j,k).eq.88.and.ivf(i,j+1,k).eq.1)
&   cnue(i,j,k)=cnue(i,j+1,k)

      if (ivf(i,j,k).eq.99)cnue(i,j,k)=cnum
      end do
    end do
  end do

c   los cnut centrados en u, v, w
c.....para cnutu.....
  do k=1,nk
    do j=1,nj
      do i=2,ni
        if (ivfu(i,j,k).eq.1) then
          im1=i-1
          if(im1.le.0)im1=1
          cnueu(i,j,k)=(cnue(i-1,j,k)*dx(im1)*0.5d0+
&   cnue(i,j,k)*dx(i)*0.5d0)/
&   (dx(i)*0.5d0+dx(im1)*0.5d0)
        end if
      end do
    end do
  end do

  do k=1,nk
    do j=1,nj
      do i=1,ni
        if(ivfu(i,j,k).eq.99) cnueu(i,j,k)=cnum
      end do
    end do
  end do

c.....para cnutv.....

```

```

do k=1,nk
  do j=2,nj
    do i=1,ni
      if (ivfv(i,j,k).eq.1) then
        jm1=j-1
        if(jm1.le.0)jm1=1
        cnuev(i,j,k)=(cnue(i,j-1,k)*dy(jm1)*0.5d0+
&          cnue(i,j,k)*dy(j)*0.5d0)/
&          (dy(j)*0.5d0+dy(jm1)*0.5d0)
        end if
      end do
    end do
  end do
end do

```

```

do k=1,nk
  do j=1,nj
    do i=1,ni
      if(ivfv(i,j,k).eq.99) cnuev(i,j,k)=cnum
    end do
  end do
end do

```

```

c.....para cnutw.....
do j=1,nj
  do i=1,ni
    do k=2,nk
      if (ivf(i,j,k).eq.1) then
        cnuew(i,j,k)=(cnue(i,j,k-1)*dz(i,j,k-1)*0.5d0+
&          cnue(i,j,k)*dz(i,j,k)*0.5d0)/
&          (dz(i,j,k-1)*0.5d0+dz(i,j,k)*0.5d0)
      else
        cnuew(i,j,k)=cnum
      end if
    end do
  end do
end do

```

```

        if (ivf(i,j,nk).eq.1)then
cnuew(i,j,nk+1)=(cnue(i,j,nk)*dz(i,j,nk)*0.5d0+
&                cnue(i,j,nk)*dz(i,j,nk)*0.5d0)/
&                (dz(i,j,nk)*0.5d0+dz(i,j,nk)*0.5d0)
        endif
    end do
end do

c    Primer Criterio: (la viscosidad turbulenta dentro de la
derivadas)
c    call opfuch
c    call opfvch

    call opfulm
    call opfvlm

c    call opfulm3d
c    call opfvlm3d

110 format (3i4,5f8.2)
500 format(100e20.10)

end subroutine modlm3d

c
*****
*****
    subroutine opfulm
c
*****
*****
c calcula el valor de la funcion fu
c considera conveccion, difusion y coriolis

```

```

integer    :: i,j,k,im1,jm1
real(kind=8) :: difx,dudx1,dudx2,dudy1,dudy2,dify1,
&          dvdx1,dvdx2,dify2,coru

do k=1,nk
  do j=2,nj-1
    do i=2,ni-1
      if(ivfu(i,j,k).eq.1) then

c      difusion en x 2.0*d/dx*(vE*du/dx)
        im1=i-1
        if(im1.le.0) im1=1

        jm1=j-1
        if(jm1.le.0) jm1=1

        dudx1=(u(i,j,k)-u(i-1,j,k))/dx(im1)

        dudx2=(u(i+1,j,k)-u(i,j,k))/dx(i)

        difx=2.0d0*
&        (cnue(i,j,k)*dudx2-cnue(i-1,j,k)*dudx1)
&        /(0.5d0*dx(i)+0.5d0*dx(im1))

c      difusion en y1 d/dy*(vE*du/dy)
        if(ivfu(i,j+1,k).ne.99.and.ivfu(i,j-1,k).ne.99) then
          dudy1=(u(i,j,k)-u(i,j-
1,k))/(0.5d0*dy(j)+0.5d0*dy(jm1))
          dudy2=(u(i,j+1,k)-
u(i,j,k))/(0.5d0*dy(j+1)+0.5d0*dy(j))

          dify1=(cnuev(i,j+1,k)*dudy2-cnuev(i,j,k)*dudy1)/dy(j)
        end if

        if(ivfu(i,j+1,k).eq.99.and.ivfu(i,j-1,k).eq.1) then

```

```

        dudy1=(u(i,j-1,k)-
u(i,j,k))/(0.5d0*dy(j)+0.5d0*dy(jm1))
        dudy2=(g*u(i,j,k)*u(i,j,k))/(chezy*chezy)

        dify1=(cnuev(i,j,k)*dudy1 - dudy2)/dy(j)
        end if

        if(ivfu(i,j+1,k).eq.1.and.ivfu(i,j-1,k).eq.99) then
            dudy1=(g*u(i,j,k)*u(i,j,k))/(chezy*chezy)
            dudy2=(u(i,j+1,k)-
u(i,j,k))/(0.5d0*dy(j+1)+0.5d0*dy(j))

            dify1=(cnuev(i,j+1,k)*dudy2 - dudy1)/dy(j)
            end if

c        difusion en y2 d/dy*(vE*dv/dx)
            dvdx1=(v(i,j,k)-v(i-
1,j,k))/(0.5d0*dx(i)+0.5d0*dx(im1))
            dvdx2=(v(i,j+1,k)-v(im1,j+1,k))/
            &      (0.5d0*dx(i)+0.5d0*dx(im1))

            dify2=(cnuev(i,j+1,k)*dvdv2-cnuev(i,j,k)*dvdv1)/dy(j)

c        coriolis en x
            coru=cor(i,j)*vpx(i,j,k)*icor

            fu(i,j,k)=u(i,j,k)+dt*(difx+dify1+dify2+coru)
        else
            fu(i,j,k)=0.0d0
        end if
    end do
end do
end do

200 format (225i4)
300 format(1000e10.2)

```

```
500 format(100e20.10)
```

```
end subroutine opfulm
```

```
c
*****
*****
subroutine opfvlm
c
*****
*****
c calcula el valor de la funcion fv
c considera conveccion, difusion y coriolis
integer :: i,j,k,im1,jm1
real(kind=8) :: dify,dvdy1,dvdy2,dvdx1,dvdx2,
& difx1,dudy1,dudy2,difx2,corv

do k=1,nk
do j=2,nj-1
do i=2,ni-1
if(ivfv(i,j,k).eq.1) then

im1=i-1
if(im1.le.0) im1=1

jm1=j-1
if(jm1.le.0) jm1=1

c difusion en y 2.0*d/dy*(vE*dv/dy)
dvdy1=(v(i,j,k)-v(i,j-1,k))/dy(jm1)

dvdy2=(v(i,j+1,k)-v(i,j,k))/dy(j)

dify=2.0*
& (cnue(i,j,k)*dvdy2 - cnue(i,j-1,k)*dvdy1)
& /(0.5d0*dy(j)+0.5d0*dy(jm1))
```

```

c      difusion en x1 d/dx*(vE*dv/dx)
      if(ivfv(i+1,j,k).ne.99.and.ivfv(i-1,j,k).ne.99) then
        dwdx1=(v(i,j,k)-v(i-
1,j,k))/(0.5d0*dx(i)+0.5d0*dx(im1))
        dwdx2=(v(i+1,j,k)-
v(i,j,k))/(0.5d0*dx(i+1)+0.5d0*dx(i))

        difx1=(cnueu(i+1,j,k)*dwdx2-cnueu(i,j,k)*dwdx1)/dx(i)
      end if

      if(ivfv(i+1,j,k).eq.99.and.ivfv(i-1,j,k).eq.1) then
        dwdx1=(v(i-1,j,k)-
v(i,j,k))/(0.5d0*dx(i)+0.5d0*dx(im1))
        dwdx2=(g*v(i,j,k)*v(i,j,k))/(chezy*chezy)

        difx1=(cnueu(i,j,k)*dwdx1 - dwdx2)/dx(i)
      end if

      if(ivfv(i+1,j,k).eq.1.and.ivfv(i-1,j,k).eq.99) then
        dwdx1=(g*v(i,j,k)*v(i,j,k))/(chezy*chezy)
        dwdx2=(v(i+1,j,k)-
v(i,j,k))/(0.5d0*dx(i+1)+0.5d0*dx(i))

        difx1=(cnueu(i+1,j,k)*dwdx2 - dwdx1)/dx(i)
      end if

c      difusion en x2 d/dx*(vE*du/dy)
        dudy1=(u(i,j,k)-u(i,j-
1,k))/(0.5d0*dy(j)+0.5d0*dy(jm1))
        dudy2=(u(i+1,j,k)-u(i+1,j-1,k))
&          /(0.5d0*dy(j)+0.5d0*dy(jm1))

        difx2=(cnueu(i+1,j,k)*dudy2-cnueu(i,j,k)*dudy1)/dx(i)

c      coriolis en x

```

```

corv=cor(i,j)*upy(i,j,k)*icor

fv(i,j,k)=v(i,j,k)+dt*(dify+difx1+difx2-corv)
else
fv(i,j,k)=0.0d0
end if
end do
end do
end do

```

```

300 format(100e20.10)
500 format(100e20.10)

```

```

end subroutine opfvlm

```

```

=====
=====
!           difusión centrada en "u"
!=====
=====
c
*****
*****
      subroutine opfulm3d
c
*****
*****
c calcula el valor de la funcion fu
c considera conveccion, difusion y coriolis
      integer    :: i,j,k
c   real(kind=8) :: difx,dudx1,dudx2,dudy1,dudy2,dify1,
c   &            dvdx1,dvdx2,dify2,coru
c   real(8)      :: dderux,dderuy,ddervx,ddervy

      do j=2,nj-1

```



```

do i=2,ni-1
  if (ivfu(i,j,nk).eq.1) then
    do k=nnku(i,j),nk
!      para la dderux no hay problema
      dderux=((u(i+1,j,k)-u(i,j,k))/dx(i)-
(u(i,j,k)-u(i-1,j,k))
&      /dx(i-1))/(dx(i)*0.5e0+dx(i-1)*0.5e0)

!      para la dderuy si hay problema
      if(ivfu(i,j-
1,k).ne.99.and.ivfu(i,j+1,k).ne.99) then
        dderuy=((u(i,j+1,k)-
u(i,j,k))/(dy(j+1)*0.5e0
&      +dy(j)*0.5e0)-(u(i,j,k)-u(i,j-
1,k))/(dy(j)*0.5e0
&      +dy(j-1)*0.5e0))/(dy(j))
        end if
        if(ivfu(i,j-
1,k).eq.99.and.ivfu(i,j+1,k).ne.99) then
          dderuy=((u(i,j+1,k)-
u(i,j,k))/(dy(j+1)*0.5e0+dy(j)
&      *0.5e0)-(u(i,j,k)-(-
1.e0*u(i,j,k)))/(dy(j)
&      *0.5e0+dy(j)*0.5e0))/(dy(j))
          end if

        if(ivfu(i,j+1,k).eq.99.and.ivfu(i,j-
1,k).ne.99) then
          dderuy=(((-1.e0*u(i,j,k))-
u(i,j,k))/(dy(j+1)
&      *0.5e0+dy(j)*0.5e0)-(u(i,j,k)-
u(i,j-1,k))/
&      (dy(j)*0.5e0+dy(j-
1)*0.5e0))/(dy(j))
          end if

```

```

                                if(ivfu(i,j+1,k).eq.99.and.ivfu(i,j-
1,k).eq.99) then
                                dderuy=(-1.e0*u(i,j,k)-
u(i,j,k))/(dy(j)*0.5e0+dy(j)
                                &          *0.5e0)-(u(i,j,k)+u(i,j,k))/(dy(j)*0.5e0+dy(j)
                                &          *0.5e0))/(dy(j))
                                end if

                                fu(i,j,k)=aam(i,j,k)*(dderux+dderuy)

                                end do
                                else
                                fu(i,j,k)=0.0d0
                                end if
                                end do
                                end do

                                end subroutine opfulm3d

```

```

!=====
!=====
!           difusión centrada en "v"
!=====
!=====

```

```

c
*****
*****
subroutine opfvlm3d
c
*****
*****
c calcula el valor de la funcion fv
c considera conveccion, difusion y coriolis
integer :: i,j,k
c real(kind=8) :: dify,dvdy1,dvdy2,dvdx1,dvdx2,
c &          difx1,dudy1,dudy2,difx2,corv

```

```

real(8)          :: dderux, dderuy, ddervx, ddervy

do j=2,nj-1
do i=2,ni-1
    if (ivfv(i,j,nk).eq.1) then
        do k=nnkv(i,j),nk
!           para la ddervx si hay problema
            if(ivfv(i-
1,j,k).ne.99.and.ivfv(i+1,j,k).ne.99) then
                ddervx=((v(i+1,j,k)-
v(i,j,k))/(dx(i+1)*0.5e0
&
+dx(i)*0.5e0)-(v(i,j,k)-v(i-
1,j,k))/(dx(i)*0.5e0
&
+dx(i-1)*0.5e0))/(dx(i))
            end if

                if(ivfv(i-
1,j,k).eq.99.and.ivfv(i+1,j,k).ne.99) then
                    ddervx=((v(i+1,j,k)-
v(i,j,k))/(dx(i+1)*0.5e0
&
+dx(i)*0.5e0)-(v(i,j,k)-(-
1.e0*v(i,j,k)))/
&
(dx(i)*0.5e0+dx(i)*0.5e0))/(dx(i))
                end if

                    if(ivfv(i-
1,j,k).ne.99.and.ivfv(i+1,j,k).eq.99) then
                        ddervx=((-1.e0*v(i,j,k)-
v(i,j,k))/(dx(i+1)
&
*0.5e0+dx(i)*0.5e0)-(v(i,j,k)-
v(i-1,j,k))/
&
(dx(i)*0.5e0+dx(i-1)*0.5e0))/(dx(i))
                    end if

```

```

                                if(ivfv(i-
1,j,k).eq.99.and.ivfv(i+1,j,k).eq.99) then
                                ddervx=((-v(i,j,k)-
v(i,j,k))/(dx(i)*0.5e0+dx(i)
                                &
                                *0.5e0)-(v(i,j,k)+v(i,j,k))/(dx(i)
                                &
                                *0.5e0+dx(i)*0.5e0))/(dx(i))
                                end if

!                                para la ddervy no hay problema
                                ddervy=((v(i,j+1,k)-v(i,j,k))/dy(j)-
(v(i,j,k)-v(i,j-1,k))
                                &
                                /dy(j-1))/(dy(j)*0.5e0+dy(j-1)*0.5e0)

                                fv(i,j,k)=aam(i,j,k)*(ddervx+ddervy)

                                end do
                                else
                                fv(i,j,k)=0.0d0
                                end if
                                end do
                                end do
=====
110    format (3i4,5f8.2)

                                end subroutine opfvlm3d

```

```

c
*****
*****
                                subroutine opfuch
c
*****
*****
c calcula el valor de la funcion fu

```

```

c agrupa los términos de conveccion, difusion y coriolis
c cvh coeficiente de viscosidad horizontal calculado segun ke
  integer    :: i,j,k
  real(kind=8) :: ch,b,ks
  real(kind=8) :: vvdv, dderux, dderuy, difturu, coru, cvh

c  Datos
  ch=0.10d0
  b=33.d0
  ks=0.000216d0

do k=1,nk
do j=2,nj-1
do i=2,ni-1
  if (ivfu(i,j,k).eq.1) then

      vvdv=vpx(i,j,k)
      cvh=ch*tu(i,j)
&      *(sqrt(u(i,j,k)*u(i,j,k)+vvdv*vvdv))
&      /(5.75d0*log10((12.1d0*tu(i,j))/ks))

c  para la dderux no hay problema
      dderux=((u(i+1,j,k)-u(i,j,k))/dx(i)-
&      (u(i,j,k)-u(i-1,j,k))/dx(i-1))/
&      (dx(i)*0.5d0+dx(i-1)*0.5d0)

c  para la dderuy si hay problema
      if(ivfu(i,j-1,k).ne.99.and.ivfu(i,j+1,k).ne.99) then
          dderuy=((u(i,j+1,k)-u(i,j,k))
&      /(dy(j+1)*0.5d0+dy(j)*0.5d0)-
&      (u(i,j,k)-u(i,j-1,k))/
&      (dy(j)*0.5d0+dy(j-1)*0.5d0))/
&      (dy(j))
      end if

      if(ivfu(i,j-1,k).eq.99.and.ivfu(i,j+1,k).ne.99) then

```

```

        dderuy=((u(i,j+1,k)-u(i,j,k))/
&          (dy(j+1)*0.5d0+dy(j)*0.5d0)-
&          (u(i,j,k)-(-u(i,j,k)))/
&          (dy(j)*0.5d0+dy(j)*0.5d0))/
&          (dy(j))
    end if

    if(ivfu(i,j+1,k).eq.99.and.ivfu(i,j-1,k).ne.99) then
        dderuy=(((-u(i,j,k))-u(i,j,k))/
&          (dy(j+1)*0.5d0+dy(j)*0.5d0)-
&          (u(i,j,k)-u(i,j-1,k)))/
&          (dy(j)*0.5d0+dy(j-1)*0.5d0))/
&          (dy(j))
    end if

    if(ivfu(i,j+1,k).eq.99.and.ivfu(i,j-1,k).eq.99) then
        dderuy=((-u(i,j,k)-u(i,j,k))/(dy(j)*0.5d0+dy(j)*0.5d0)-
&          (u(i,j,k)+u(i,j,k))/(dy(j)*0.5d0+dy(j)*0.5d0))/
&          (dy(j))
    end if

c      difturu=cnueu(i,j,k)*(dderux+dderuy)
      difturu=cvh*(dderux+dderuy)

c      coriolis en x
      coru=cor(i,j)*vpx(i,j,k)*icor

c      operador fu
      fu(i,j,k)=u(i,j,k)+dt*(difturu+coru)

    end if
  end do
end do
end do

```

200 format (225i4)

```
300 format(1000e10.2)
```

```
500 format(100e20.10)
```

```
end subroutine opfuch
```

```
c
```

```
*****
```

```
*****
```

```
subroutine opfvch
```

```
c
```

```
*****
```

```
*****
```

```
c calcula el valor de la funcion fv
```

```
c considera conveccion, difusion y coriolis
```

```
c cvh coeficiente de viscosidad horizontal calculado segun ke
```

```
integer :: i,j,k
```

```
real(kind=8) :: ch,b,ks
```

```
real(kind=8) :: uudp,ddervx,ddervy,difturv,corv,cvh
```

```
c Datos
```

```
ch=0.10d0
```

```
b=33.d0
```

```
ks=0.000216d0
```

```
do k=1,nk
```

```
do j=2,nj-1
```

```
do i=2,ni-1
```

```
if (ivfv(i,j,k).eq.1) then
```

```
uudp=upy(i,j,k)
```

```
cvh=ch*tv(i,j)
```

```
& *(sqrt(uudp*uudp+v(i,j,k)*v(i,j,k)))
```

```
& /(5.75d0*log10((12.1d0*tv(i,j))/ks))
```

```
c para la ddervx si hay problema
```

```
if(ivfv(i-1,j,k).ne.99.and.ivfv(i+1,j,k).ne.99) then
```

```

      ddervx=((v(i+1,j,k)-v(i,j,k))/
&          (dx(i+1)*0.5d0+dx(i)*0.5d0)-
&          (v(i,j,k)-v(i-1,j,k))/
&          (dx(i)*0.5d0+dx(i-1)*0.5d0))/
&          (dx(i))
      end if

      if(ivfv(i-1,j,k).eq.99.and.ivfv(i+1,j,k).ne.99) then
      ddervx=((v(i+1,j,k)-v(i,j,k))/
&          (dx(i+1)*0.5d0+dx(i)*0.5d0)-
&          (v(i,j,k)-(-v(i,j,k)))/
&          (dx(i)*0.5d0+dx(i)*0.5d0))/(dx(i))
      end if

      if(ivfv(i-1,j,k).ne.99.and.ivfv(i+1,j,k).eq.99) then
      ddervx=(-v(i,j,k)-v(i,j,k))/
&          (dx(i+1)*0.5d0+dx(i)*0.5d0)-
&          (v(i,j,k)-v(i-1,j,k))/
&          (dx(i)*0.5d0+dx(i-1)*0.5d0))/
&          (dx(i))
      end if

      if(ivfv(i-1,j,k).eq.99.and.ivfv(i+1,j,k).eq.99) then
      ddervx=(-v(i,j,k)-v(i,j,k))/(dx(i)*0.5d0+dx(i)*0.5d0)-
&          (v(i,j,k)+v(i,j,k))/(dx(i)*0.5d0+dx(i)*0.5d0))/
&          (dx(i))
      end if

c      para la ddervy no hay problema
      ddervy=((v(i,j+1,k)-v(i,j,k))/dy(j)-
&          (v(i,j,k)-v(i,j-1,k))/dy(j-1))/
&          (dy(j)*0.5d0+dy(j-1)*0.5d0)

c      difturv=cnuev(i,j,k)*(ddervx+ddervy)
      difturv=cvh*(ddervx+ddervy)

```



```

c      coriolis en y
      corv=cor(i,j)*upy(i,j,k)*icor

c      operador fv
      fv(i,j,k)=v(i,j,k)+dt*(difturv-corv)

      end if
      end do
      end do
      end do

300 format(100e20.10)

      end subroutine opfvch

c*****
****
      subroutine advecu
c*****
****
      integer :: i,j,k,ki1,ji1,ii1

      real(kind=8)          :: p,q,r
      real(kind=8),dimension(n(1),n(2),n(3)) ::
a2,b2,d2,uad,up,adu

c      asignacion de la velocidad advectiva identificando si es
c      pared o si esta rodeada de flujo
      do k=1,nk
      do j=1,nj
      do i=1,ni
      if (ivfu(i,j,k).eq.99) uad(i,j,k)=0.0d0
      end do
      end do
      end do

```

```

do k=1,nk
  do j=1,nj
    do i=1,ni
      if (ivfu(i,j,k).eq.1) uad(i,j,k)=u(i,j,k)
      if (ivfu(i,j,k).eq.88)uad(i,j,k)=u(i,j,k)
      if (ivfu(i,j,k).eq.77)uad(i,j,k)=u(i,j,k)

      if (ivfu(i,j,k).eq.1.and.ivfu(i,j+1,k).eq.99)
&      uad(i,j+1,k)=-u(i,j,k)

      if (ivfu(i,j,k).eq.99.and.ivfu(i,j+1,k).eq.1)
&      uad(i,j,k)=-u(i,j+1,k)

      if (ivfu(i,j,k).eq.99.and.ivfu(i,j+1,k).eq.99)
&      uad(i,j,k)=0.0d0
c-----
      if (ivfu(i,j,k).eq.99.and.ivfu(i,j,k+1).eq.1)
&      uad(i,j,k)=-u(i,j,k+1)

      end do
    end do
  end do
c  en el fondo y en la superficie
do j=2,nj-1
  do i=2,ni-1
    if (ivfu(i,j,nk).eq.1) then
      uad(i,j,nku(i,j)-1)=-u(i,j,nku(i,j))
      uad(i,j,nk+1)=u(i,j,nk)
    end if
  end do
end do

do k=1,nk
  do j=1,nj
    do i=1,ni
      up(i,j,k)=(u(i,j,k)+u(i+1,j,k))*0.5d0

```

```

    if (ivfu(i,j,k).ne.99) then
      a2(i,j,k)=abs(up(i,j,k)*dt/dx(i))
      b2(i,j,k)=abs(vpx(i,j,k)*dt/dy(j))
      d2(i,j,k)=abs(wpx(i,j,k)*dt/dzu(i,j,k))
    else
      a2(i,j,k)=0.0d0
      b2(i,j,k)=0.0d0
      d2(i,j,k)=0.0d0
    endif
  enddo
enddo
enddo

```

- c se considera que se va a cumplir la condicion de courant
 c entonces $a=p$, $b=q$, $d=r$ y $l=0$ $m=0$ $n=0$

```

do k=1,nk
  do j=1,nj
    do i=1,ni
      if (ivfu(i,j,k).eq.1) then

```

- c conocer si la u, v o w son positivas o negativas

```

      if (wpx(i,j,k).ge.0) then
        ki1=k-1
      else
        ki1=k+1
        if(ki1.gt.nk) ki1=nk
      end if
      r=d2(i,j,ki1)

```

```

      if (vpx(i,j,k).ge.0) then
        ji1=j-1
      else
        ji1=j+1
      end if
      q=b2(i,ji1,k)

```

```

        if (up(i,j,k).ge.0) then
            ii1=i-1
        else
            ii1=i+1
        end if
        p=a2(ii1,j,k)

        adu(i,j,k)=(1.-r)*((1.-p)*
&      ((1.-q)*uad(i,j,k)+q*uad(i,ji1,k))
&      +p*((1.-q)*uad(ii1,j,k)+q*uad(ii1,ji1,k)))
&      +r*((1.-p)*((1.-q)*uad(i,j,ki1)+q*uad(i,ji1,ki1))
&      +p*((1.-q)*uad(ii1,j,ki1)+q*uad(ii1,ji1,ki1)))
        endif
    enddo
enddo
enddo

do k=1,nk
do j=1,nj
do i=1,ni
    if (ivfu(i,j,k).eq.1) then
        u(i,j,k)=adu(i,j,k)
    else
        u(i,j,k)=0.0d0
    end if
end do
end do
end do

300 format(1000e10.2)
500 format(500e20.10)
600 format(500e10.2)

end subroutine advecu

```

```

C*****
****
      subroutine advecv
C*****
****
      integer   :: i,j,k,ki1,ji1,ii1

      real(kind=8)           :: p,q,r
      real(kind=8),dimension(n(1),n(2),n(3))           ::
a3,b3,d3,vad,vp,adv

      do j=1,nj
      do k=1,nk
      do i=1,ni
      if (ivfv(i,j,k).eq.99) vad(i,j,k)=0.0d0
      end do
      end do
      end do

      do j=1,nj
      do k=1,nk
      do i=1,ni
      if (ivfv(i,j,k).eq.1) vad(i,j,k)=v(i,j,k)
      if (ivfv(i,j,k).eq.88)vad(i,j,k)=v(i,j,k)
      if (ivfv(i,j,k).eq.77)vad(i,j,k)=v(i,j,k)

      if (ivfv(i,j,k).eq.1.and.ivfv(i+1,j,k).eq.99) then
      vad(i+1,j,k)=-v(i,j,k)
      end if

      if (ivfv(i,j,k).eq.99.and.ivfv(i+1,j,k).eq.1) then
      vad(i,j,k)=-v(i+1,j,k)
      end if

      if (ivfv(i,j,k).eq.99.and.ivfv(i+1,j,k).eq.99) then
      vad(i,j,k)=0.0d0

```

```

end if

c-----
  if (ivfv(i,j,k).eq.99.and.ivfv(i,j,k+1).eq.1)
&    vad(i,j,k)=-v(i,j,k+1)
  end do
end do
end do

c  en el fondo y en la superficie
do j=2,nj-1
do i=2,ni-1
  if (ivfv(i,j,nk).eq.1) then
    vad(i,j,nkv(i,j)-1)=-v(i,j,nkv(i,j))
    vad(i,j,nk+1)=v(i,j,nk)
  end if
end do
end do

do j=1,nj
do k=1,nk
do i=1,ni
  vp(i,j,k)=(v(i,j,k)+v(i,j+1,k))*0.5d0
  if (ivfv(i,j,k).ne.99) then
    a3(i,j,k)=abs(upy(i,j,k)*dt/dx(i))
    b3(i,j,k)=abs(vp(i,j,k)*dt/dy(j))
    d3(i,j,k)=abs(wpy(i,j,k)*dt/dzv(i,j,k))
  else
    a3(i,j,k)=0.0d0
    b3(i,j,k)=0.0d0
    d3(i,j,k)=0.0d0
  endif
enddo
enddo
enddo
enddo

```

```

c   se considera que se va a cumplir la condicion de courant
c   entonces a=p, b=q, d=r y l=0 m=0 n=0
do i=1,ni
  do j=1,nj
    do k=1,nk
      if (ivfv(i,j,k).eq.1) then

c   conocer si la u, v o w son positivas o negativas
      if (wpy(i,j,k).ge.0) then
        ki1=k-1
      else
        ki1=k+1
        if(ki1.gt.nk) ki1=nk
      end if
      r=d3(i,j,ki1)

      if (vp(i,j,k).ge.0) then
        ji1=j-1
      else
        ji1=j+1
      end if
      q=b3(i,ji1,k)

      if (upy(i,j,k).ge.0) then
        ii1=i-1
      else
        ii1=i+1
      end if
      p=a3(ii1,j,k)

      adv(i,j,k)=(1.-r)*((1.-p)*
&      ((1.-q)*vad(i,j,k)+q*vad(i,ji1,k))
&      +p*((1.-q)*vad(ii1,j,k)+q*vad(ii1,ji1,k)))
&      +r*((1.-p)*((1.-q)*vad(i,j,ki1)+q*vad(i,ji1,ki1))
&      +p*((1.-q)*vad(ii1,j,ki1)+q*vad(ii1,ji1,ki1)))
    endif
  enddo
enddo

```

```

        enddo
    enddo
enddo

do j=1,nj
do k=1,nk
do i=1,ni
    if (ivfv(i,j,k).eq.1) then
        v(i,j,k)=adv(i,j,k)
    else
        v(i,j,k)=0.0d0
    end if
end do
end do
end do

end subroutine advecv

```

```

c
*****
*****
    subroutine calcu
c
*****
*****
c    subrutina calcu, calcula las velocidades u
c    modelado de 3d
    integer :: i, j, k, mm, nmat, ma, iu, ju
    real(kind=8) :: viento,cteff,ctep,dzeta,vvdp,vzdz0,
&          vzdz1,bet
    real(kind=8),dimension (n(3)) :: a,b,c,r,gam,x

do j=2,nj
do i=2,ni

    viento=dt*ctevie(i,j)*wx(i,j)*abs(wx(i,j))

```



```

cteff=dt*g/(chezy*chezy)
if(ivfu(i,j,nk).eq.1) then
  ctep=g*dt/(.5d0*(dx(i)+dx(i-1)))
  dzeta=ctep*(z1(i,j)-z1(i-1,j))

c      calculo de los coeficientes de la matriz
do k=nnku(i,j),nk
  mm=k-nnku(i,j)+1
  vdz0=(cnueu(i,j,k)*0.5d0+cnueu(i,j,k-1)*0.5d0)/
&      (dzu(i,j,k)*0.5d0+dzu(i,j,k-1)*0.5d0)

  vdz1=(cnueu(i,j,k)*0.5d0+cnueu(i,j,k+1)*0.5d0)/
&      (dzu(i,j,k)*0.5d0+dzu(i,j,k+1)*0.5d0)

  a(mm)=-dt*vdz0
  b(mm)=dzu(i,j,k)+dt*(vdz0+vdz1)
  c(mm)=-dt*vdz1
  r(mm)=dzu(i,j,k)*(fu(i,j,k)-dzeta)
end do

c-----para la superficie libre (k=nk)
mm=nk-nnku(i,j)+1
vdz0=(cnueu(i,j,nk)*0.5d0+cnueu(i,j,nk-1)*0.5d0)/
&      (dzu(i,j,nk)*0.5d0+dzu(i,j,nk-1)*0.5d0)

a(mm)=-dt*vdz0
b(mm)=dzu(i,j,nk)+dt*(vdz0)
c(mm)=0.0d0
r(mm)=dzu(i,j,nk)*(fu(i,j,nk)-dzeta)+
&      viento

c-----para el fondo (k=nnku)
mm=1
k=nnku(i,j)
vdz1=(cnueu(i,j,k)*0.5d0+cnueu(i,j,k+1)*0.5d0)/
&      (dzu(i,j,k)*0.5d0+dzu(i,j,k+1)*0.5d0)

```

```

vvdv=vpv(i,j,k)*vpv(i,j,k)
a(mm)=0.0d0
b(mm)=dzu(i,j,k)+dt*vzdz1+cteff*
&      sqrt(u(i,j,k)*u(i,j,k)+vvdv)
c(mm)=-dt*vzdz1
r(mm)=dzu(i,j,k)*(fu(i,j,k)-dzeta)

c-----caso cuando es dos dimensiones (nk=nnku)
  if(nk.eq.nnku(i,j)) then
    a(mm)=0.0
    b(mm)=dzu(i,j,nk)+cteff*
    &      sqrt(u(i,j,nk)*u(i,j,nk)+
    &      vpv(i,j,nk)*vpv(i,j,nk))
    c(mm)=0.0
    r(mm)=dzu(i,j,nk)*(fu(i,j,nk)-dzeta)+
    &      viento
  end if

c-----
c   calculo de u al resolver la matriz tridiagonal
c   write(4,130) k,m, a(m), b(m), c(m), r(m)
x = 0.
nmat=nk-nnku(i,j)+1
bet=b(1)
x(1)=r(1)/bet
do ma=2,nmat
  gam(ma)=c(ma-1)/bet
  bet=b(ma)-a(ma)*gam(ma)
c   if(bet.eq.0.) go to 20
  x(ma)=(r(ma)-a(ma)*x(ma-1))/bet
enddo

do ma=nmat-1,1,-1
  x(ma)=x(ma)-gam(ma+1)*x(ma+1)
enddo

```

```

c      call tridag(a,b,c,r,x,nmat)
c-----
      do k=1,nnku(i,j)-1
        u(i,j,k)=0.0d0
      end do
      do k=1,nmat
        mm=k-1+nnku(i,j)
        u(i,j,mm)=x(k)
      end do
    endif
  end do
end do

130 format('u=',2i4,4e10.2)
140 format(8f10.4)
300 format(100e20.10)

      end subroutine calculu

c
*****
*****
      subroutine calculv
c
*****
*****
c  subrutina calculu, calcula las velocidades v
c  modelado de 3d
      integer :: i, j, k, mm, nmat, ma, iv, jv
      real(kind=8) :: viento,cteff,ctep,dzeta,uudp,vzd0,
&          vzd1,bet
      real(kind=8),dimension (n(3)) :: a,b,c,r,gam,x

      cteff=dt*g/(chezy*chezy)

```

```

do j=2,nj
  do i=2,ni
    viento=dt*ctevie(i,j)*wy(i,j)*abs(wy(i,j))
    if(ivfv(i,j,nk).eq.1) then

      ctep=g*dt/(.5d0*(dy(j)+dy(j-1)))
      dzeta=ctep*(z1(i,j)-z1(i,j-1))

c      calculo de los coeficientes de la matriz
      do k=nnkv(i,j),nk
        mm=k-nnkv(i,j)+1
        vdz0=(cnuev(i,j,k)*0.5d0+cnuev(i,j,k-1)*0.5d0)/
&        (dzv(i,j,k)*0.5d0+dzv(i,j,k-1)*0.5d0)

        vdz1=(cnuev(i,j,k)*0.5d0+cnuev(i,j,k+1)*0.5d0)/
&        (dzv(i,j,k)*0.5d0+dzv(i,j,k+1)*0.5d0)

        a(mm)=-dt*vdz0
        b(mm)=dzv(i,j,k)+dt*(vdz0+vdz1)
        c(mm)=-dt*vdz1
        r(mm)=dzv(i,j,k)*(fv(i,j,k)-dzeta)
      end do

c-----para la superficie libre (k=nk)
      mm=nk-nnkv(i,j)+1
      vdz0=(cnuev(i,j,nk)*0.5d0+cnuev(i,j,nk-1)*0.5d0)/
&      (dzv(i,j,nk)*0.5d0+dzv(i,j,nk-1)*0.5d0)

      a(mm)=-dt*vdz0
      b(mm)=dzv(i,j,nk)+dt*(vdz0)
      c(mm)=0.0d0
      r(mm)=dzv(i,j,nk)*(fv(i,j,nk)-dzeta)+
&      viento

c-----para el fondo (k=nnkv)
      mm=1

```

```

k=nnkv(i,j)
vzdz1=(cnuev(i,j,k)*0.5d0+cnuev(i,j,k+1)*0.5d0)/
&      (dzv(i,j,k)*0.5d0+dzv(i,j,k+1)*0.5d0)

uudp=upy(i,j,k)*upy(i,j,k)
a(mm)=0.0d0
b(mm)=dzv(i,j,k)+dt*vzdz1+cteff*
&      sqrt(v(i,j,k)*v(i,j,k)+uudp)
c(mm)=-dt*vzdz1
r(mm)=dzv(i,j,k)*(fv(i,j,k)-dzeta)

c-----caso cuando es dos dimensiones (nk=nnkv)
if(nk.eq.nnkv(i,j)) then
  a(mm)=0.0
  b(mm)=dzv(i,j,nk)+cteff*
&      sqrt(v(i,j,nk)*v(i,j,nk)+
&      upy(i,j,nk)*upy(i,j,nk))
  c(mm)=0.0
  r(mm)=dzv(i,j,nk)*(fv(i,j,nk)-dzeta)+
&      viento
end if

c-----
c      calculo de v al resolver la matriz tridiagonal
x = 0.
nmat=nk-nnkv(i,j)+1
bet=b(1)
x(1)=r(1)/bet
do ma=2,nmat
  gam(ma)=c(ma-1)/bet
  bet=b(ma)-a(ma)*gam(ma)
c      if(bet.eq.0.) exit
  x(ma)=(r(ma)-a(ma)*x(ma-1))/bet
enddo

do ma=nmat-1,1,-1
  x(ma)=x(ma)-gam(ma+1)*x(ma+1)

```

```

        enddo
c-----

        do k=1,nnkv(i,j)-1
            v(i,j,k)=0.0d0
        end do
        do k=1,nmat
            mm=k-1+nnkv(i,j)
            v(i,j,mm)=x(k)
        end do
    endif
end do
end do

130 format('v=',5i4,7e17.7)
140 format(8f10.4)
300 format (100e17.7)
310 format (5i6,100e17.7)

    end subroutine calcv

c
*****
*****
    subroutine calcw
c
*****
*****
c  subrutina calcw, calcula las velocidades w
    integer    :: i,j,k
    real(kind=8) :: qu,qv

    do i=2,ni-1
        do j=2,nj-1
            if (ivf(i,j,nk).eq.1) then

```

```

do k=1,nnkz(i,j)
  w(i,j,k)=0.0d0
end do

do k=nnkz(i,j),nk
  qu=(dzu(i+1,j,k)*u(i+1,j,k)-dzu(i,j,k)*u(i,j,k))/dx(i)
  qv=(dzv(i,j+1,k)*v(i,j+1,k)-dzv(i,j,k)*v(i,j,k))/dy(j)
  w(i,j,k+1)=w(i,j,k)-qu-qv
end do
end if
end do
end do

end subroutine calcw

```

```

c
*****
****
  subroutine super5
c
*****
****
c  subrutina super5, calcula el nivel de superficie libre
c  formando una matriz pentadiagonal y resolverla por medio de
c  una sustitucion hacia atras
c  ref. casulli,et al 1992

integer :: i, j, k, nm, kr,mm,kk,im1,ip1,jm1,jp1
real(kind=8) :: lambda,teta,psi,vaux,gg1,hh1,gg2,ff2,ee1,ff1
real(kind=8),dimension (n(1),n(2)) ::
cmq,cmd,cmsi,cmsj,cmqi,cmqj

integer, dimension ( : ), allocatable :: indxi,indxj
real(kind=8),dimension( : ),allocatable :: dd,ee,ff,gg,hh,b,x

```

```

teta=0.50d0

do j=1,nj
  do i=1,ni
c    primero con la celda centrada en i-1/2
    if(ivfu(i,j,nk).eq.1) then

        nm=nk-nnku(i,j)+1
        call cfmatu(i,j,nm,nnku(i,j),ctevie(i,j),
&          wx(i,j),u(i,j,nnku(i,j)),
&          vpx(i,j,nnku(i,j)),cmsi(i,j),
&          cmqi(i,j),dx(i))

    end if

c    luego con la celda centrada en j-1/2
    if(ivfv(i,j,nk).eq.1) then

        nm=nk-nnkv(i,j)+1
        call cfmatv(i,j,nm,nnkv(i,j),ctevie(i,j),
&          wy(i,j),v(i,j,nnkv(i,j)),
&          upy(i,j,nnkv(i,j)),cmsj(i,j),
&          cmqj(i,j),dy(j))

    end if

c    para las celdas 77
c    primero con la celda centrada en i-1/2
    if(ivfu(i,j,nk).eq.77) then
        cmsi(i,j)=0.d0

        vaux=0.d0
        do k=nnku(i,j),nk
            vaux=vaux+dzu(i,j,k)*u(i,j,k)
        end do

```



```

        cmqi(i,j)=vaux
    end if

c    luego con la celda centrada en j-1/2
    if(ivfv(i,j,nk).eq.77) then
        cmsj(i,j)=0.d0

        vaux=0d0
        do k=nnkv(i,j),nk
            vaux=vaux+dzv(i,j,k)*v(i,j,k)
        end do
        cmqj(i,j)=vaux
    end if
end do

c    asignacion de fronteras para los coeficientes
do j=1,nj
do i=1,ni

    ip1=i+1
    if(ip1.ge.ni)ip1=ni

    im1=i-1
    if(im1.le.1)im1=1

    jp1=j+1
    if(jp1.ge.nj)jp1=nj

    jm1=j-1
    if(jm1.le.1)jm1=1

    if(ivfu(im1,j,nk).eq.1.and.ivfu(i,j,nk).eq.88) then
        cmsi(i,j)=cmsi(im1,j)
        cmqi(i,j)=cmqi(im1,j)
    end if
end do
end do

```

```

end if

if(ivfv(i,jm1,nk).eq.1.and.ivfv(i,j,nk).eq.88) then
  cmsj(i,j)=cmsj(i,jm1)
  cmqj(i,j)=cmqj(i,jm1)
end if

if(ivfu(ip1,j,nk).eq.1.and.ivfu(i,j,nk).eq.88) then
  cmsi(i,j)=cmsi(ip1,j)
  cmqi(i,j)=cmqi(ip1,j)
end if

if(ivfv(i,jp1,nk).eq.1.and.ivfv(i,j,nk).eq.88) then
  cmsj(i,j)=cmsj(i,jp1)
  cmqj(i,j)=cmqj(i,jp1)
end if
c-----
if(ivfu(i,j,nk).eq.99.and.ivfu(im1,j,nk).eq.1) then
  cmsi(i,j)=cmsi(im1,j)
end if

if(ivfv(i,j,nk).eq.99.and.ivfv(i,jm1,nk).eq.1) then
  cmsj(i,j)=cmsj(i,jm1)
end if

if(ivfu(ip1,j,nk).eq.1.and.ivfu(i,j,nk).eq.99) then
  cmsi(i,j)=cmsi(ip1,j)
end if

if(ivfv(i,jp1,nk).eq.1.and.ivfv(i,j,nk).eq.99) then
  cmsj(i,j)=cmsj(i,jp1)
end if
c-----
if(ivfu(i,j,nk).eq.99) then
  cmqi(i,j)=0.0d0
end if

```

```

        if(ivfv(i,j,nk).eq.99) then
            cmqj(i,j)=0.0d0
        end if

    end do
end do

cmd(:,:) = 1.0d0

c  formacion de las matrices cmq y cmd (d(i,j) y q(i,j))
nm=0
do j=2,nj-1
    do i=2,ni-1
        if(ivfv(i,j,nk).eq.1) then
            cmd(i,j)=1.+teta*cmsi(i+1,j)+teta*cmsi(i,j)
&             +teta*cmsj(i,j+1)+teta*cmsj(i,j)

            psi=z1(i,j)+(1.0d0-teta)*(
&             -(dt/dx(i))*(cmqi(i+1,j)-cmqi(i,j))
&             -(dt/dy(j))*(cmqj(i,j+1)-cmqj(i,j))

&             +cmsi(i+1,j)*(z1(i+1,j)-z1(i,j))
&             -cmsi(i,j)*(z1(i,j)-z1(i-1,j))

&             +cmsj(i,j+1)*(z1(i,j+1)-z1(i,j))
&             -cmsj(i,j)*(z1(i,j)-z1(i,j-1))
&             )
            cmq(i,j)=psi
&             -teta*(dt/dx(i))*(cmqi(i+1,j)-cmqi(i,j))
&             -teta*(dt/dy(j))*(cmqj(i,j+1)-cmqj(i,j))
            nm=nm+1
        end if
    end do
end do

```

```
allocate ( indxi(nm), indxj(nm) )
```

```
mm=0  
do j=2,nj-1  
  do i=2,ni-1  
    if(ivf(i,j,nk).eq.1) then  
      mm=mm+1  
      indxi(mm)=i  
      indxj(mm)=j  
    endif  
  enddo  
enddo
```

c asignacion de fronteras

```
do j=1,nj  
  do i=1,ni  
  
    ip1=i+1  
    if(ip1.ge.ni)ip1=ni  
  
    im1=i-1  
    if(im1.le.1)im1=1  
  
    jp1=j+1  
    if(jp1.ge.nj)jp1=nj  
  
    jm1=j-1  
    if(jm1.le.1)jm1=1  
  
    if(ivf(i,j,nk).eq.88.and.ivf(im1,j,nk).eq.1) then  
      cmd(i,j)=cmd(im1,j)  
      cmq(i,j)=cmq(im1,j)  
    end if  
  
    if(ivf(ip1,j,nk).eq.1.and.ivf(i,j,nk).eq.88) then
```

```

        cmd(i,j)=cmd(ip1,j)
        cmq(i,j)=cmq(ip1,j)
    end if

    if(ivf(i,j,nk).eq.88.and.ivf(i,jm1,nk).eq.1) then
        cmd(i,j)=cmd(i,jm1)
        cmq(i,j)=cmq(i,jm1)
    end if

    if(ivf(i,j,nk).eq.99) then
        cmd(i,j)=1.0d0
        cmq(i,j)=0.0d0
    end if

    if(ivf(i,jp1,nk).eq.1.and.ivf(i,j,nk).eq.88) then
        cmd(i,j)=cmd(i,jp1)
        cmq(i,j)=cmq(i,jp1)
    end if

end do
end do

do j=1,nj
do i=1,ni
    if(ivf(i,j,nk).eq.99) then
        cmd(i,j)=1.0d0
        cmq(i,j)=0.0d0
    end if
end do
end do

```

c formacion de las 5 diagonales que van a tener valores
c dd->diagonal principal; ee->diagonal superior 1; ff->diagonal superior 2
c hh->diagonal inferior 1; gg->diagonal inferior 2

c $e_{ij} = \sqrt{d_{i,j}} * z_{1(i,j)}$, esto con el fin de normalizar la matriz a resolver
 c y convertirla en diagonalmente dominante para que el metodo converja

```

allocate (
dd(nm),ee(nm),ff(nm),gg(nm),hh(nm),b(nm),x(nm) )
      dd = 0.d0
      ee = 0.d0
      ff = 0.d0
      gg = 0.d0
      hh = 0.d0
      b = 0.d0
      x = 0.d0

```

```

c primer renglon
i=indxi(1)
j=indxj(1)

dd(1)=1.0d0

      ee(1)=-cmsj(i,j+1)*teta
&      /(sqrt(cmd(i,j)*cmd(i,j+1)))

      ff(1)=-cmsi(i+1,j)*teta
&      /(sqrt(cmd(i,j)*cmd(i+1,j)))

      hh(1)=0.0d0
      hh1=-cmsj(i,j)*teta
&      /(sqrt(cmd(i,j)*cmd(i,j-1)))

      gg(1)=0.0d0
      gg1=-cmsi(i,j)*teta
&      /(sqrt(cmd(i,j)*cmd(i-1,j)))

```

b(1)=cmq(i,j)/(sqrt(cmd(i,j)))

c segundo renglon

i=indxi(2)

j=indxj(2)

dd(2)=1.0d0

ee(2)=-cmsj(i,j+1)*teta
& / (sqrt(cmd(i,j)*cmd(i,j+1)))

ff(2)=-cmsi(i+1,j)*teta
& / (sqrt(cmd(i,j)*cmd(i+1,j)))

hh(2)=-cmsj(i,j)*teta
& / (sqrt(cmd(i,j)*cmd(i,j-1)))

gg(2)=0.0d0
gg2=-cmsi(i,j)*teta
& / (sqrt(cmd(i,j)*cmd(i-1,j)))

b(2)=cmq(i,j)/(sqrt(cmd(i,j)))

do kr=3,nm-2

i=indxi(kr)

j=indxj(kr)

dd(kr)=1.0d0

hh(kr)=-cmsj(i,j)*teta
& / (sqrt(cmd(i,j)*cmd(i,j-1)))

gg(kr)=-cmsi(i,j)*teta

```

&      /(sqrt(cmd(i,j)*cmd(i-1,j)))

      ee(kr)=-cmsj(i,j+1)*teta
&      /(sqrt(cmd(i,j)*cmd(i,j+1)))

      ff(kr)=-cmsi(i+1,j)*teta
&      /(sqrt(cmd(i,j)*cmd(i+1,j)))

      b(kr)=cmq(i,j)/(sqrt(cmd(i,j)))

end do

c  penultimo renglon
i=indxi(nm-1)
j=indxj(nm-1)

dd(nm-1)=1.0d0

      hh(nm-1)=-cmsj(i,j)*teta
&      /(sqrt(cmd(i,j)*cmd(i,j-1)))

      gg(nm-1)=-cmsi(i,j)*teta
&      /(sqrt(cmd(i,j)*cmd(i-1,j)))

      ee(nm-1)=-cmsj(i,j+1)*teta
&      /(sqrt(cmd(i,j)*cmd(i,j+1)))

      ff(nm-1)=0.d0
      ff2=-cmsi(i+1,j)*teta
&      /(sqrt(cmd(i,j)*cmd(i+1,j)))

      b(nm-1)=cmq(i,j)/(sqrt(cmd(i,j)))

c  ultimo renglon

```



```

i=indxi(nm)
j=indxj(nm)

dd(nm)=1.0d0

  hh(nm)=-cmsj(i,j)*teta
&      /(sqrt(cmd(i,j)*cmd(i,j-1)))

  gg(nm)=-cmsi(i,j)*teta
&      /(sqrt(cmd(i,j)*cmd(i-1,j)))

  ee(nm)=0.d0
  ee1=-cmsj(i,j+1)*teta
&      /(sqrt(cmd(i,j)*cmd(i,j+1)))

  ff(nm)=0.d0
  ff1=-cmsi(i+1,j)*teta
&      /(sqrt(cmd(i,j)*cmd(i+1,j)))

  b(nm)=cmq(i,j)/(sqrt(cmd(i,j)))

c   resolver el sistema pentadiagonal que se forma por gauss-
seidel penta
c   con lambda igual a cero

  lambda =1.5d0
  call gseid5(dd,ee,ff,gg,hh,b,nm,x,lambda,niter,
&          gg1,hh1,gg2,ff2,ee1,ff1)

c   calculo de la elevacion
do kk=1,nm
  z1(indxi(kk),indxj(kk))=x(kk)
&      /(sqrt(cmd(indxi(kk),indxj(kk))))

end do

```

```

150 format(//,'niveles de superficie libre, i=',i4)
152 format(i4,10f10.4)
300 format(1000e15.3)
400 format(3i10,10e17.8)

```

```

end subroutine super5

```

```

c
*****
*****
      subroutine cfmatu(i,j,nm,nnk,ctevien,wxy,
&          vel,velp,cms,cmq,dxy)
c
*****
*****
c   subrutina para calcular los coeficientes de la matriz
c   (a) que se utilizará en el método de gauss seidel
integer      :: nm,i,j,nnk,kr,l,m,k,
&   ii,jj

real(kind=8)      :: ctevien,wxy,dxy,
&          vel,velp,dzeuv,visdzm1,visdzp1

real(kind=8)      :: eps1,eps2,cms,cmq
real(kind=8),dimension (1,1)      :: s,q
real(kind=8),dimension (n(3),n(3)) :: a
real(kind=8),dimension (n(3),1)   :: mdz,mg
real(kind=8),dimension (1,n(3))   :: mdzt
real(kind=8),dimension (n(3))     :: alf,bet,gam

mdzt= 0.d0
mdz = 0.d0
mg = 0.d0

```

```

alf = 0.d0
bet = 0.d0
gam = 0.d0

```

```

c=====>formacion de las matrices "a", deltaz y mg
c   fronteras para la superficie libre

```

```

mg(1,1)=fu(i,j,nk)*dzu(i,j,nk)
&      +(dt*ctevien*wxy*abs(wxy))

```

```

mdz(1,1)=dzu(i,j,nk)

```

```

dzeuv=dzu(i,j,nk)*0.5d0 + dzu(i,j,nk-1)*0.5d0
visdzm1=((cnueu(i,j,nk)+cnueu(i,j,nk-1))*0.5d0)/
&      dzeuv

```

```

alf(1)=0.d0

```

```

bet(1)=dzu(i,j,nk)+(dt*visdzm1)

```

```

gam(1)=-dt*visdzm1

```

```

k=nk
do kr=2,nm-1
  k=k-1

```

```

dzeuv=dzu(i,j,k)*0.5d0 + dzu(i,j,k-1)*0.5d0
visdzm1=((cnueu(i,j,k)+cnueu(i,j,k-1))*0.5d0)/
&      dzeuv

```

```

dzeuv=dzu(i,j,k)*0.5d0 + dzu(i,j,k+1)*0.5d0
visdzm1=((cnueu(i,j,k)+cnueu(i,j,k+1))*0.5d0)/
&      dzeuv

```

```

mdz(kr,1)=dzu(i,j,k)

mg(kr,1)=fu(i,j,k)*dzu(i,j,k)

alf(kr)=-dt*visdzp1

bet(kr)=dzu(i,j,k)+dt*(visdzp1+visdzm1)

gam(kr)=-dt*visdzm1
end do

```

c frontera para el fondo para la matriz "a"

```

dzeuv=dzu(i,j,nnk)*0.5d0 + dzu(i,j,nnk+1)*0.5d0
visdzp1=((cnueu(i,j,nnk)+cnueu(i,j,nnk+1))*0.5d0)/
& dzeuv

mdz(nm,1)=dzu(i,j,nnk)

mg(nm,1)=fu(i,j,nnk)*dzu(i,j,nnk)

alf(nm)=-dt*visdzp1

bet(nm)=dzu(i,j,nnk)+(dt*visdzp1)+
& (g*dt/(chezy*chezy))*
& sqrt(vel*vel+velp*velp)

gam(nm)=0.d0

```

c caso especial 2d
if(nnk.eq.nk) then

```

mdz(1,1)=dzu(i,j,nk)

mg(1,1)=fu(i,j,nk)*dzu(i,j,nk)
& +(dt*ctevien*wxy*abs(wxy))

```

```

alf(1)=0.d0

bet(1)=dzu(i,j,nk)+
&      (g*dt/(chezy*chezy))*
&      sqrt(vel*vel+velp*velp)

gam(1)=0.d0

endif

```

c=====>formacion de la matriz inversa de "a" por el metodo de gauss-jordan

```

a = 0.d0
do l=1,nm
  a(l,l)=1.0d0
enddo

```

```

alf(1) =0.d0
gam(nm)=0.d0

```

```

do l=2,nm
  eps1=bet(l-1)
  eps2=alf(l)
  alf(l)=eps1*alf(l)-eps2*bet(l-1)
  bet(l)=eps1*bet(l)-eps2*gam(l-1)
  gam(l)=eps1*gam(l)

```

```

do m=1,nm
  a(l,m)=eps1*a(l,m)-eps2*a(l-1,m)
enddo
enddo

```

```

do l=nm-1,1,-1
  eps1=gam(l)

```

```

eps2=bet(l+1)
gam(l)=eps2*gam(l)-eps1*bet(l+1)
bet(l)=eps2*bet(l)

do m=nm,1,-1
  a(l,m)=eps2*a(l,m)-eps1*a(l+1,m)
enddo
enddo

do m=1,nm
  do l=1,nm
    a(l,m)=a(l,m)/bet(l)
  enddo
enddo

c=====>formacion de la matriz transpuesta de deltaz(i-
1/2,j)
do kr=1,n(3)
  mdzt(1,kr) = mdz(kr,1)
enddo

c=====>formacion de los coeficientes de la matriz
s = g*(dt*dt)/(dxy*dxy)
& *(matmul(mdzt(1:1,1:nm) ,
& matmul(a(1:nm,1:nm) , mdz(1:nm,1:1))))

cms=s(1,1)

q = (matmul(mdzt(1:1,1:nm) ,
& matmul(a(1:nm,1:nm) , mg(1:nm,1:1))))

cmq=q(1,1)

300 format(100e12.3)
400 format(i3,100e12.3)

```

```
end subroutine cfmatu
```

```
c
*****
*****
      subroutine cfmatv(i,j,nm,nnk,ctevien,wxy,
&          vel,velp,cms,cmq,dxy)
c
*****
*****
c  subrutina para calcular los coeficientes de la matriz
c  (a) que se utilizará en el método de gauss seidel
integer      :: nm,i,j,nnk,kr,l,m,k
& ii,jj

real(kind=8)      :: ctevien,wxy,dxy,
&          vel,velp,dzeuv,visdzm1,visdzp1

real(kind=8)      :: eps1,eps2,cms,cmq
real(kind=8),dimension (1,1)      :: s,q
real(kind=8),dimension (n(3),n(3)) :: a
real(kind=8),dimension (n(3),1)   :: mdz,mg
real(kind=8),dimension (1,n(3))   :: mdzt
real(kind=8),dimension (n(3))     :: alf,bet,gam

mdzt= 0.d0
mdz = 0.d0
mg = 0.d0
alf = 0.d0
bet = 0.d0
gam = 0.d0

c=====>formacion de las matrices "a", deltaz y mg
c  fronteras para la superficie libre
```

```

mg(1,1)=fv(i,j,nk)*dzv(i,j,nk)
&      +(dt*ctevien*wxy*abs(wxy))

mdz(1,1)=dzv(i,j,nk)

dzeuv=dzv(i,j,nk)*0.5d0 + dzv(i,j,nk-1)*0.5d0
visdzm1=((cnuev(i,j,nk)+cnuev(i,j,nk-1))*0.5d0)/
&      dzeuv

alf(1)=0.d0

bet(1)=dzv(i,j,nk)+(dt*visdzm1)

gam(1)=-dt*visdzm1

k=nk
do kr=2,nm-1
  k=k-1

  dzeuv=dzv(i,j,k)*0.5d0 + dzv(i,j,k-1)*0.5d0
  visdzm1=((cnuev(i,j,k)+cnuev(i,j,k-1))*0.5d0)/
&      dzeuv

  dzeuv=dzv(i,j,k)*0.5d0 + dzv(i,j,k+1)*0.5d0
  visdzp1=((cnuev(i,j,k)+cnuev(i,j,k+1))*0.5d0)/
&      dzeuv

mdz(kr,1)=dzv(i,j,k)

mg(kr,1)=fv(i,j,k)*dzv(i,j,k)

alf(kr)=-dt*visdzp1

bet(kr)=dzv(i,j,k)+dt*(visdzp1+visdzm1)

```



```

        gam(kr)=-dt*visdzm1
    end do

c    frontera para el fondo para la matriz "a"

    dzeuv=dzv(i,j,nnk)*0.5d0 + dzv(i,j,nnk+1)*0.5d0
    visdzp1=((cnuev(i,j,nnk)+cnuev(i,j,nnk+1))*0.5d0)/
&        dzeuv

    mdz(nm,1)=dzv(i,j,nnk)

    mg(nm,1)=fv(i,j,nnk)*dzv(i,j,nnk)

    alf(nm)=-dt*visdzp1

    bet(nm)=dzv(i,j,nnk)+(dt*visdzp1)+
&        (g*dt/(chezy*chezy))*
&        sqrt(vel*vel+velp*velp)

    gam(nm)=0.d0

c    caso especial 2d
    if(nnk.eq.nk) then

        mdz(1,1)=dzv(i,j,nk)

        mg(1,1)=fv(i,j,nk)*dzv(i,j,nk)
&        +(dt*ctevien*wxy*abs(wxy))

        alf(1)=0.d0

        bet(1)=dzv(i,j,nk)+
&        (g*dt/(chezy*chezy))*
&        sqrt(vel*vel+velp*velp)

        gam(1)=0.d0

```

```

endif

c=====>formacion de la matriz inversa de "a" por el
metodo de gauss-jordan
a = 0.d0
do l=1,nm
  a(l,l)=1.0d0
enddo

alf(1) =0.d0
gam(nm)=0.d0

do l=2,nm
  eps1=bet(l-1)
  eps2=alf(l)
  alf(l)=eps1*alf(l)-eps2*bet(l-1)
  bet(l)=eps1*bet(l)-eps2*gam(l-1)
  gam(l)=eps1*gam(l)

  do m=1,nm
    a(l,m)=eps1*a(l,m)-eps2*a(l-1,m)
  enddo
enddo

do l=nm-1,1,-1
  eps1=gam(l)
  eps2=bet(l+1)
  gam(l)=eps2*gam(l)-eps1*bet(l+1)
  bet(l)=eps2*bet(l)

  do m=nm,1,-1
    a(l,m)=eps2*a(l,m)-eps1*a(l+1,m)
  enddo
enddo

```

```

do m=1,nm
do l=1,nm
a(l,m)=a(l,m)/bet(l)
enddo
enddo

```

c=====>formacion de la matriz transpuesta de deltaz(i-1/2,j)

```

do kr=1,n(3)
mdzt(1,kr) = mdz(kr,1)
enddo

```

c=====>formacion de los coeficientes de la matriz

```

s = g*(dt*dt)/(dxy*dxy)
& *(matmul(mdzt(1:1,1:nm) ,
& matmul(a(1:nm,1:nm) , mdz(1:nm,1:1))))

```

```

cms=s(1,1)

```

```

q = (matmul(mdzt(1:1,1:nm) ,
& matmul(a(1:nm,1:nm) , mg(1:nm,1:1))))

```

```

cmq=q(1,1)

```

```

300 format(100e12.3)
400 format(i3,100e12.3)

```

```

end subroutine cfmatv

```

```

c
*****
*****
subroutine vievar

```

```

c
*****
*****
c   subrutina para calcular el coeficiente del viento
c   por medio de la formula de garrat
integer   :: i,j

c   calculo de la cdv de acuerdo a la formula de garrat(1977)
do i=1,ni
  do j=1,nj
    wx(i,j)=wxx
    wy(i,j)=wyy

    ctevie(i,j)=((0.75+0.067*sqrt(wx(i,j)*wx(i,j)
&      +wy(i,j)*wy(i,j)))*0.001d0)
&      *densr

    end do
  end do

  end subroutine vievar

c
*****
*****
  subroutine imprime(ntie)
c
*****
*****
c   subrutina imprime, para dibujo de resultados
integer :: i, j, k, nii,nnk,nd,ntie
real(kind=8) :: tiempo

tiempo=ntie*dt
write(2,*)'niveles de sup libre, tiempo (s) t=',tiempo,ntie

```

```

c  por exploración, imprimo solo .... capas superiores
   nd=nk

c  hasta que capa imprimir
   nnk=nk

   nii=ni

c  niveles de superficie libre
   do j=nj,1,-1
     write(2,300) (z1(i,j),i=1,nii)
   end do

c  velocidades longitudinales
   write(2,*)'velocidades u,tiempo (s) t=',tiempo,ntie
   do k=nnk,nd,-1
     write(2,*)'u capa k=',k
     do j=1,nj
       write(2,300) (u(i,j,k),i=1,nii)
     end do
   end do

c  velocidades transversales
   write(2,*)'velocidades v,tiempo (s) t=',tiempo,ntie
   do k=nnk,nd,-1
     write(2,*)'v capa k=',k
     do j=nj,1,-1
       write(2,300) (v(i,j,k),i=2,nii)
     end do
   end do

c
c  para ahorrar bytes, dejo de imprimir en matriz a las vel w
c  velocidades verticales
   write(2,*)'velocidades w,tiempo (s) t=',tiempo
   do k=nnk+1,nd,-1
     write(2,*)'w capa k=',k

```

```

        do j=nj,1,-1
            write(2,300) (w(i,j,k),i=1,ni)
        end do
    end do

300 format(1000e10.2)

end subroutine imprime

c*****
*****
    subroutine vortic
c*****
*****
c    subrutina para calcular la vorticidad
    integer          :: i,j,k,km

    real(kind=8)      :: vor1,vor2,vor3
    real(kind=8),dimension(n(1),n(2),n(3)) :: dux,dvx,dwx,dvy,
    &                duy,dwy,dwz,duz,dvz,
    &                vort

c=====
=====
c    gradientes con respecto a x
c=====
=====
    do i=1,ni
        do j=1,nj
            do k=1,nk
                if(ivf(i,j,k).eq.1) then
                    dux(i,j,k)=(u(i+1,j,k)-u(i,j,k))/dx(i)
                    dvx(i,j,k)=(vpx(i+1,j,k)-vpx(i,j,k))/dx(i)
                    dwx(i,j,k)=(wpx(i+1,j,k)-wpx(i,j,k))/dx(i)
                else
                    dux(i,j,k)=0.0d0

```

```

        dvx(i,j,k)=0.0d0
        dwx(i,j,k)=0.0d0
    end if
end do
end do
end do

```

```

c=====
=====
c   gradientes con respecto a y
c=====
=====

```

```

do i=1,ni
do j=1,nj
do k=1,nk
if(ivf(i,j,k).eq.1) then
    dvy(i,j,k)=(v(i,j+1,k)-v(i,j,k))/dy(j)
    duy(i,j,k)=(upy(i,j+1,k)-upy(i,j,k))/dy(j)
    dwy(i,j,k)=(wpy(i,j+1,k)-wpy(i,j,k))/dy(j)
else
    dvy(i,j,k)=0.0d0
    duy(i,j,k)=0.0d0
    dwy(i,j,k)=0.0d0
end if
end do
end do
end do

```

```

c=====
=====
c   gradientes con respecto a z
c=====
=====

```

```

do j=1,nj
do i=1,ni

```

```

do k=1,nk
  if(ivf(i,j,k).eq.1) then
    km=k+1
    dwz(i,j,k)=(w(i,j,km)-w(i,j,k))/(dz(i,j,k))
    duz(i,j,k)=(upz(i,j,km)-upz(i,j,k))/(dz(i,j,k))
    dvz(i,j,k)=(vpz(i,j,km)-vpz(i,j,k))/(dz(i,j,k))
  else
    dwz(i,j,k)=0.0d0
    duz(i,j,k)=0.0d0
    dvz(i,j,k)=0.0d0
  end if
end do
end do
end do

do j=1,nj
  do i=1,ni
    do k=1,nk
      if(ivf(i,j,k).eq.1) then
        vor1=(dwy(i,j,k)-dvz(i,j,k))
        vor2=(duz(i,j,k)-dwx(i,j,k))
        vor3=(dvx(i,j,k)-duy(i,j,k))
        vort(i,j,k)=sqrt(vor1*vor1+vor2*vor2+vor3*vor3)
      end if
    end do
  end do
end do
end do

300 format(1000e10.2)

end subroutine vortic

```



```

c
*****
*
      subroutine tecplo(ntie,ntp)
c
*****
*
c  imprime para que dibuje tecplot
integer :: i, j, k, ntie,ntp,nil
real(kind=8) :: tiempo,uu0,vv0,ww0,xtp,ytp,zz
real(kind=8),dimension (n(1),n(2),n(3)) :: ctr,ekt

tiempo=(ntie)*dt
ntp=ntp+1
if (ntp.eq.1) then
  do i=1,ni+1
    do j=1,nj+1
      do k=1,nk
        if(ivf(i,j,k).eq.1) then
          ctr(i,j,k)=0
        else
          ctr(i,j,k)=5
        end if
        if(ivf(i,j,k).eq.88) then
          ctr(i,j,k)=0
        end if
      end do
    end do
  end do

  write(9,*)'variables="x(m)", "y(m)", "z(m)", "u(m/s)",
& "v(m/s)", "w(m/s)", "r(m/s)",
& "ctr", "elevat(m)",
"depth(m)", "KE(kW/m2)", "KEt(kWH)""

  write(9,100) tiempo, ni, nj, nk

```

```

do k=1,nk
  zz=dz0*(k-nk)-dz0*0.5d0
  do j=1,nj
    do i=1,ni
      uu0=(u(i,j,k)+u(i+1,j,k))*0.5d0
      vv0=(v(i,j,k)+v(i,j+1,k))*0.5d0
      ww0=(w(i,j,k)+w(i,j,k+1))*0.5d0
      r(i,j,k)=sqrt(uu0*uu0+vv0*vv0+ww0*ww0)
    
```

```

=====
=====

```

=

```

c      Energy Kinetic      = [1/2 m v2]
                                           [Joules]

```

```

c      = [(1/2)*(Rho*Volume)*r2]

```

```

c      Energy Kinetic/Area = [(1/2)*(Rho*r2)]/(1/r)

```

```

c      Energy Kinetic/m2  = [(1/2)*(Rho*r3)]/1000
                                           [kW/m2]

```

```

c      Energy Kinetic/m2  = [(1/2)*(Rho*dx(i)*dy(j)*-
hz(i,j))*r2]/(1000*3600) [kWH]

```

```

c      Rho= 1024 kg/m3 ==>Density constant of sea water
under condition temperature and

```

```

c      deep of 20C and 100m

```

```

      ek(i,j,k)=(0.5d0*1024.0d0*r(i,j,k)*r(i,j,k)*r(i,j,k))
&      /(1000.d0)

```

```

! [kW/m2]

```

```

      ekt(i,j,k)=(0.5d0*1024.0d0*dx(i)*dy(j)*(tz(i,j))
&      *r(i,j,k)*r(i,j,k))/(1000.d0*3600.d0)

```

```

! [kWH]

```

```

=====
=====

```

=

```

        xtp=xx(i)
        ytp=yy(j)
        write(9,110)xtp,ytp,zz,uu0,vv0,ww0,r(i,j,k),
&          ctr(i,j,k),z1(i,j),hz(i,j),ek(i,j,k),ekt(i,j,k)

        end do
    end do
end do

```

```

else
write(9,200) tiempo, ni, nj, nk
do k=1,nk
do j=1,nj
do i=1,ni
    uu0=(u(i,j,k)+u(i+1,j,k))*0.5d0
    vv0=(v(i,j,k)+v(i,j+1,k))*0.5d0
    ww0=(w(i,j,k)+w(i,j,k+1))*0.5d0
    r(i,j,k)=sqrt(uu0*uu0+vv0*vv0+ww0*ww0)

```

=====

=

c Energy Kinetic = [1/2 m v²] [Joules]

c = [(1/2)*(Rho*Volume)*r²]

c Energy Kinetic/Area = [(1/2)*(Rho*r²)/(1/r)

c Energy Kinetic/m² = [(1/2)*(Rho*r³)/1000 [kW/m²]

c Energy Kinetic/m² = [(1/2)*(Rho*dx(i)*dy(j)*- hz(i,j))*r²]/(1000*3600) [kWH]

c Rho= 1024 kg/m³ ==>Density constant of sea water under condition temperature and

c deep of 20C and 100m

$$ek(i,j,k) = (0.5d0 * 1024.0d0 * r(i,j,k) * r(i,j,k) * r(i,j,k))$$

```

&          /(1000.d0)
&          ! [kW/m2]
&          ekt(i,j,k)=(0.5d0*1024.0d0*dx(i)*dy(j)*(tz(i,j))
&          *r(i,j,k)*r(i,j,k))/(1000.d0*3600.d0)
&          ! [kWH]

c=====
=====
=

&          write(9,210) uu0,vv0,ww0,r(i,j,k),
&          ctr(i,j,k),z1(i,j),hz(i,j),ek(i,j,k),ekt(i,j,k)
&          end do
&          end do
&          end do
&          endif
100 format('zone t=""',f10.0,'sec", i=', i3,', j=',i3,', k=',
&          *i3,', f=point')
110 format (12e12.4)

200 format('zone t=""',f10.0,'sec", i=', i3,', j=',i3,', k=',
&          *i3,', f=point, d=(1,2,3)')
210 format (9e12.4)
300 format(1000e20.10)

&          end subroutine tecplo

&          end module physique

c
*****
*****
&          subroutine detdz(n,tir,dz0,dzmin,nnkd,dz9,nk)

```

```

c
*****
*****
implicit none
integer, dimension(3)      :: n
integer                    :: k,nk,nkd,nelez
real(kind=8)               :: tir,dz0,dzmin,fracc
real(kind=8),dimension(n(3)) :: dz9

      if (tir.le.dzmin) then
        do k=1,nk
          dz9(k)=0.d0
        end do
        nkd=99
      else
        nelez=int((tir-0.0001)/dz0)
        fracc=tir-nelez*dz0
c aqui es donde redefine si es menor que dzmin
        if (fracc.lt.dzmin.and.fracc.gt.0.0d0) then
          nelez=nelez-1
          fracc=dz0
        endif
        nkd=nk-nelez
      end if
c especifica el espesor de cada capa
      do k=1,nk
        if (k.lt.nkd) dz9(k)=0.0d0
        if (k.eq.nkd) dz9(k)=fracc
        if (k.gt.nkd) dz9(k)=dz0
      end do

      end subroutine detdz

c*****
*****

```

```

subroutine gseid5(d,e,f,g,h,b,nm,x,lambda,niter,
&      gg1,hh1,gg2,ff2,ee1,ff1)
c*****
*****
c  subrutina para calcular por medio del metodo
c      iterativo de gauss-seidel las elevaciones pero solo
utilizando
c  las 5 diagonales que tiene valores
implicit none
integer          :: i,nm,imax,sentinel,iter,niter
real(kind=8)     :: lambda,gg1,hh1,gg2,ff2,ee1,ff1
real(kind=8)     :: dummy,old,sum,es,ea
real(kind=8),dimension(nm) :: d,e,f,g,h,x,b

c  d-> diagonal principal
c  e-> diagonal superior 1
c  f-> diagonal superior 2
c  h-> diagonal inferior 1
c  g-> diagonal inferior 2
c  b-> terminos independientes
c  x-> variable a resolver
c  valores impuestos
h(1) = hh1
g(1) = gg1
g(2) = gg2

f(nm-1) = ff2
f(nm) = ff1
e(nm) = ee1

c  tolerancia
es=1.e-07

c  maxima iteración
imax=niter

```

```

do i=1,nm
  dummy=d(i)

  e(i)=e(i)/dummy
  f(i)=f(i)/dummy
  g(i)=g(i)/dummy
  h(i)=h(i)/dummy

  b(i)=b(i)/dummy

end do

do i=1,nm
  sum=b(i)

  sum=sum -(e(i) +f(i) +h(i) +g(i))*x(i)

  x(i)=sum

end do

iter=1
do
  sentinel=1
  do i=1,nm
    old=x(i)
    sum=b(i)

    sum=sum -(e(i) +f(i) +h(i) +g(i))*x(i)

    x(i)=lambda*sum+(1.-lambda)*old

    if(sentinel.eq.1.and.abs(x(i)).gt.1.d-50) then

```

```
        ea=abs((x(i)-old)/x(i))*100.d0
        if(ea.gt.es) sentinel=0
    end if

end do

iter = iter + 1

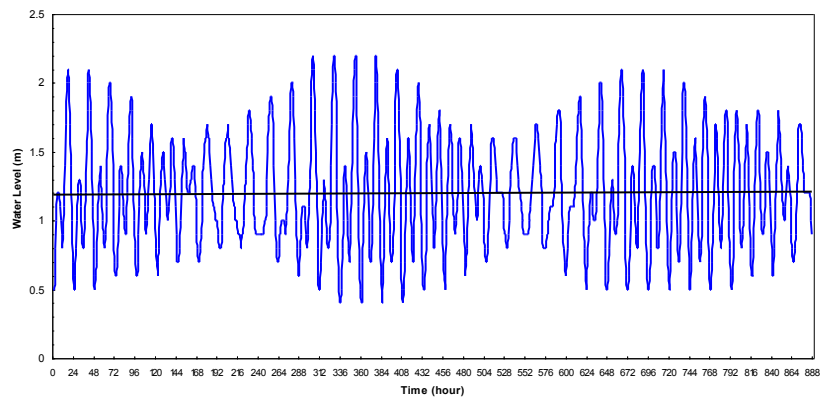
if ( (iter > imax) .or. sentinel == 1 ) exit
end do

if (iter.gt.imax) then
    write(*,*) 'se paso del num de iter',iter,ea
    pause
end if

end subroutine gseid5
```


C. Prediksi pasang surut air laut di teluk Manado

Hasil prediksi pasang surut air laut di teluk Manado menurut Dinas Hidro-Oceanografi TNI AL dari tanggal 16 Januari sampai 21 Februari 2012 ditunjukkan dalam gambar 8. Hal itu menunjukkan bahwa dalam sehari terjadi dua kali pasang surut dan waktu transisi dari air pasang ke air surut sekitar 0,5 jam serta level air maksimum rata-rata sekitar 1,5-2 m. Juga, pergerakan arus ketika air pasang menuju ke arah Selatan dan sebaliknya ketika air surut menuju ke arah Utara.



Gambar 8. Prediksi pasang surut air laut di teluk Manado

D. Uji coba simulasi 2D distribusi kecepatan arus laut dan sungai di teluk Manado

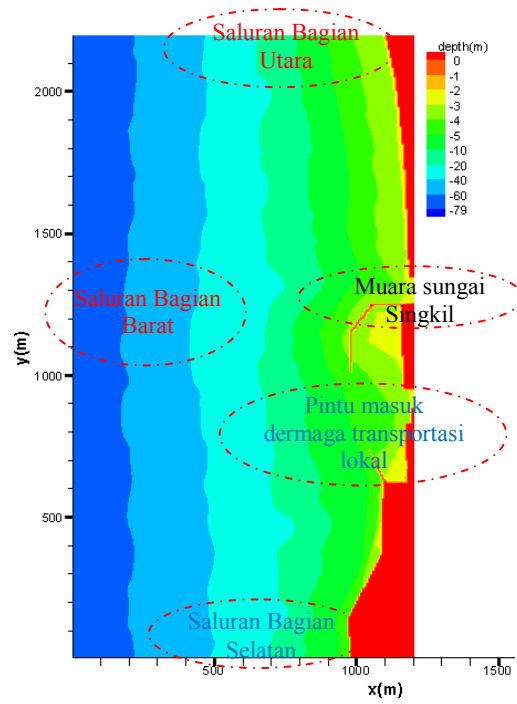
Uji coba dilakukan pada daerah khusus teluk Manado (lihat gambar 10 dan 11) untuk melihat distribusi kecepatan arus laut saat terjadi arus pasang dan surut.

Hasil simulasi numerik 2D arus laut berupa distribusi kecepatan arus dapat dilihat pada gambar 10 dan 11 yang mana parameter numerik sebagai data input dapat dilihat pada tabel 1.

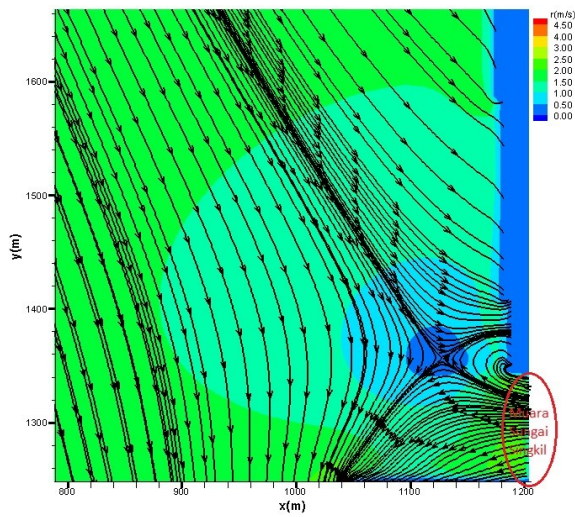
Tabel 1. Parameter numerik untuk uji coba simulasi 2D

Parameter	Nilai	Parameter	Nilai
<i>Gravitasi</i>	9.81 m s ⁻²	<i>Densitas air laut</i>	1024 kg/m ³
<i>Konstanta Chezy</i>	48	<i>Sel pada sumbu x</i>	7 m
<i>Skala waktu relaksasi kondisi keluar</i>	2 days	<i>Sel pada sumbu y</i>	7 m
<i>Skala waktu relaksasi kondisi masuk</i>	1 day	<i>Waktu</i>	0,4 s
<i>Debit</i>	0,1 Sv.		

Gambar 9 menunjukkan *bathymetry* dari lokasi numerik, Terdapat 2 sisi aliran air masuk dan 3 sisi aliran air keluar sistem saat arus pasang. Pada sisi masuk, pertama berada pada daerah aliran sungai Singkil dan kedua pada bagian utara. Pada sisi keluar, pertama pada daerah bagian timur di dekat dari aliran sungai Singkil bagian selatan (ada dermaga transportasi lokal), kedua, pada daerah bagian barat, dan ketiga, pada daerah bagian selatan. Saat arus surut, terdapat 2 sisi aliran air masuk dan 3 sisi keluar. Di sisi masuk yaitu pada daerah-daerah aliran sungai Singkil dan bagian selatan, sedangkan di sisi keluar yaitu pada daerah-daerah bagian timur dan utara serta bagian timur dekat aliran sungai Singkil. Pada sisi keluar di daerah dekat sungai Singkil tidak terjadi pengeluaran debit air karena di situ terdapat dermaga transportasi lokal yang mana air tidak mengalir keluar. Kedalaman maksimum yaitu 79 m berada pada daerah bagian barat. Kedalaman rata-rata di muara sungai Singkil adalah 2,5 m pada level air normal.



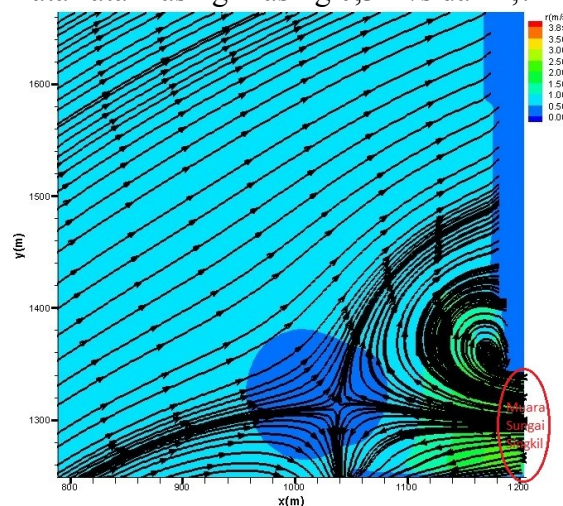
Gambar 9. Bathymetry lokasi numerik



Gambar 10. Garis-garis aliran air pada kondisi arus pasang dan 1 m kolom air laut

Gambar 10 menunjukkan garis-garis aliran air pada saat arus pasang. Daerah pertemuan arus sungai Singkil dan arus laut terdapat perempatan aliran yang berada pada sisi laut arah barat ke utara dengan jarak kira-kira 80 m sampai 108 m dari muara sungai (pada sumbu x dan y masing-masing 1126 m dan 1359 m sampai radius dari sumbu itu 14 m) dengan kecepatan arus dan kedalaman rata-rata masing-masing 0,28 m/s dan 3,6 m. Hal itu menunjukkan bahwa sangat aman jika kapal-kapal melewati daerah itu baik masuk maupun keluar sungai.

Gambar 11 sama dengan Gambar 10 tetapi pada kondisi arus surut. Itu menunjukkan bahwa daerah aman untuk pelayaran kapal-kapal adalah pada jarak 170 m sampai 220 m dari muara sungai (pada sumbu x dan y masing-masing 1040 m dan 1310 m sampai radius dari sumbu itu 28 m) dengan kecepatan air dan kedalaman rata-rata masing-masing 0,3 m/s dan 4,7 m.



Gambar 11. Garis-garis aliran air pada kondisi arus surut dan 1 m kolom air laut

Fenomena aliran air yang terjadi pada kondisi arus pasang dan arus surut ternyata sama yaitu membentuk seperti perempatan tetapi letak kejadiannya berbeda (lihat Gambar 10 dan 11). Hal itu dipengaruhi oleh arah aliran air yang mana pada kondisi arus pasang alirannya dari arah Utara sedangkan pada kondisi arus surut dari arah Selatan, saat bersamaan juga aliran sungai dari

arah Timur lebih besar kecepatannya dibandingkan kecepatan air dari laut teluk Manado. Kecepatan air sungai tidak sama pada saat arus pasang 2,43 m/s dan 2,50 m/s saat arus surut. Itu disebabkan karena saat arus pasang maka kecepatan air sungai menjadi kecil berhubung debit air dari laut teluk Manado naik sehingga menahan lajunya arus dari sungai yang menuju ke laut. Sebaliknya, saat arus surut tidak terjadi demikian sehingga arus sungai menuju ke laut sangat bebas.

E. Hasil simulasi numerik 2D dan 3D di teluk Manado

Hasil simulasi numerik 2D dan 3D arus laut berupa distribusi kecepatan arus, energi kinetik, dan gelombang air dapat dilihat pada gambar 12 dan 25 yang mana parameter numerik sebagai data input dapat dilihat pada tabel 1 dan 2.

Tabel 2. Parameter numerik untuk simulasi 2D

Parameter	Nilai	Parameter	Nilai
<i>Gravitasi</i>	9.81 m s ⁻²	<i>Densitas air laut</i>	1024 kg/m ³
<i>Konstanta Chezy</i>	48	<i>Sel pada sumbu x</i>	7 m
<i>Skala waktu relaksasi pada kondisi keluar</i>	2 days	<i>Sel pada sumbu y</i>	7 m
<i>Skala waktu relaksasi pada kondisi masuk</i>	1 day	<i>Waktu</i>	0,5 sec
<i>Debit</i>	0,1 Sv.		

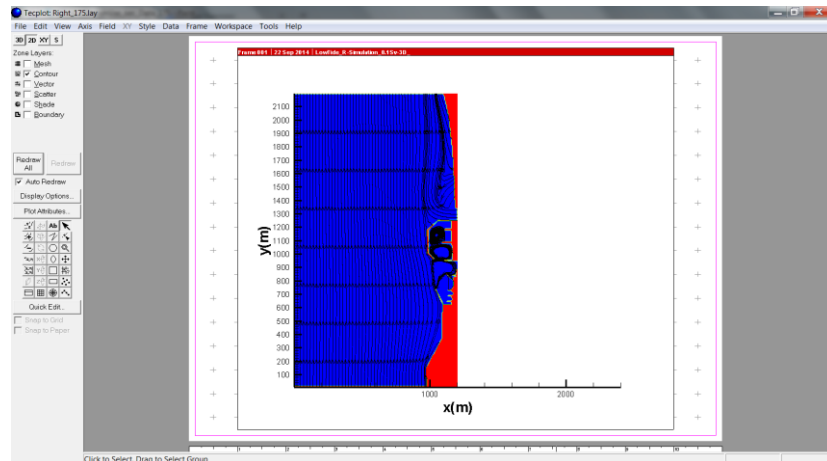
Tabel 3. Parameter numerik untuk simulasi 3D

Parameter	Nilai	Parameter	Nilai
<i>Gravitasi</i>	9.81 m s ⁻²	<i>Densitas air laut</i>	1024 kg/m ³
<i>Konstanta Chezy</i>	48	<i>Sel pada sumbu x</i>	7 m
<i>Skala waktu relaksasi pada kondisi keluar</i>	2 days	<i>Sel pada sumbu y</i>	7 m
<i>Skala waktu relaksasi pada kondisi masuk</i>	1 day	<i>Sel pada sumbu z</i>	1 m
<i>Debit</i>	0,1 Sv.	<i>Waktu</i>	0,5 sec

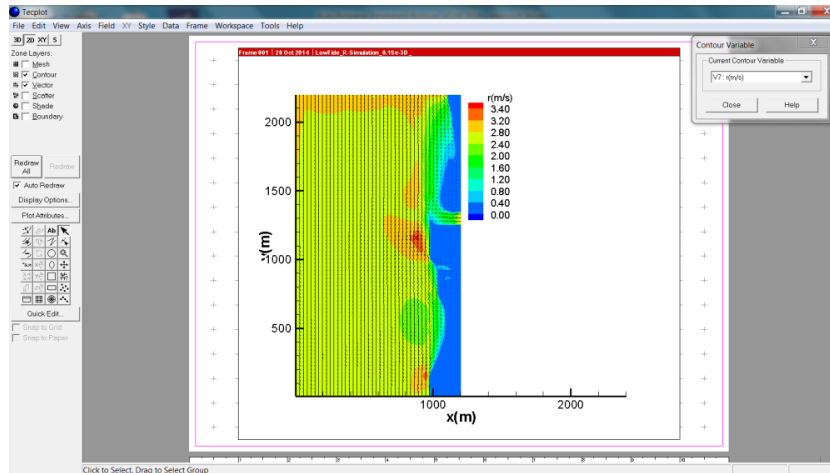
1. Hasil-hasil simulasi numerik 2D

a. Saat arus laut surut

Gambar 12 menunjukkan distribusi *streamline* arus laut pada 1 m kolom air yang terjadi saat arus laut sedang surut. Arus laut di selat Manado bergerak dari Selatan menuju ke Utara dan sebagian mengalir masuk dermaga lokal dengan membentuk turbulen-turbulen kecil kemudian keluar menuju Utara. Pada daerah pertemuan arus-arus laut dan sungai terjadi arus sungai lebih kuat dari arus laut sehingga air dari sungai keluar muaranya dan berbelok menuju Utara kemudian berputar balik menuju pinggiran muara sungai melewati pesisir pantai dan berbalik menuju Utara lagi, dan ini terjadi berulang-ulang selama arus masih surut.



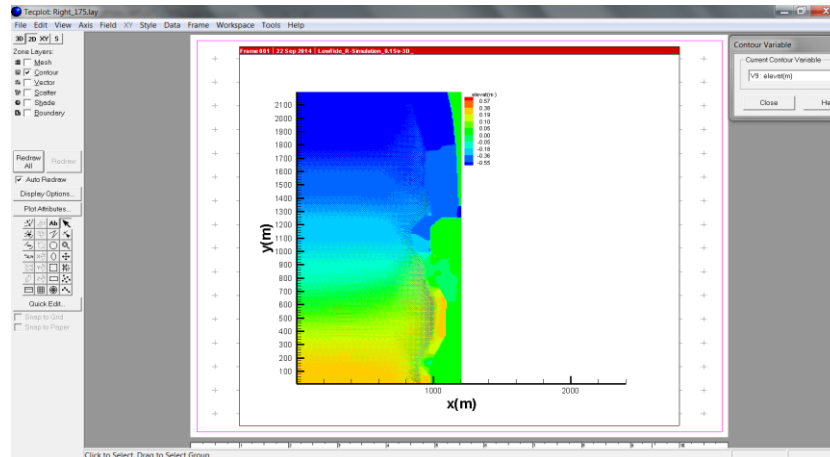
Gambar 12. Distribusi streamline arus laut pada 1 m kolom air



Gambar 13. Distribusi kecepatan arus laut pada 1 m kolom air

Gambar 13 menunjukkan distribusi kecepatan arus laut pada 1 m kolom air yang mana terjadi pada saat arus laut sedang surut. Kecepatan arus-arus laut sangat kuat kira-kira 3,20-3,40 m/s pada daerah depan dermaga lokal (lihat pada gambar 13 warna merah). Itu disebabkan karena pada daerah tersebut kedalaman lautnya yang rata-rata 5 m (dangkal) sehingga arus begitu kuat melewati daerah tersebut. Sebaliknya, kecepatan arus-arus pada daerah di dalam dermaga lokal sangat kecil sekitar 0,00-0,40 m/s dan membentuk turbulen-turbulen kecil. Hal itu diakibatkan karena arus-arus yang masuk kedalam dermaga diredam kecepatannya, sehingga sangat baik untuk tempat parkir kapal-kapal laut.

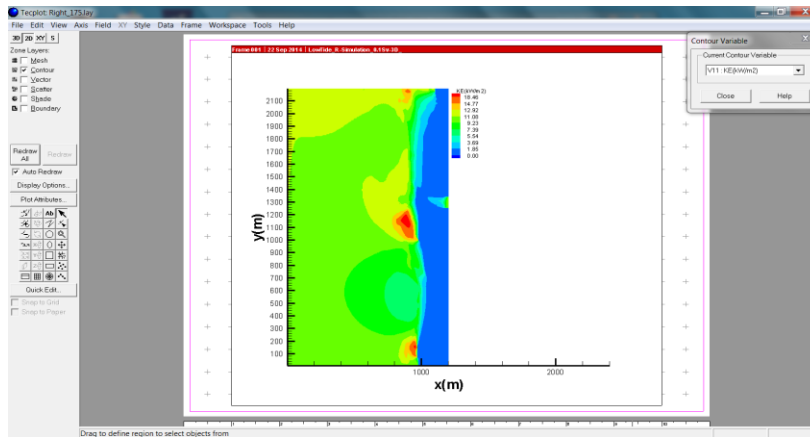
Gelombang air pada 1 m kolom air (gambar 14) di dalam dermaga lokal air mendekati tidak ada sama sekali, tetapi pada muara sungai terjadi gelombang air sekitar 0,55 m rendahnya, sehingga cukup aman untuk kapal-kapal laut melewati muara sungai tersebut.



Gambar 14. Distribusi gelombang air pada 1 m kolom air

Gambar 15 menunjukkan distribusi energi kinetik pada 1 m kolom air yang mana terjadi pada saat arus laut surut. Energi kinetik yang tersedia paling besar kira-kira $14,77-18,46 \text{ kW/m}^2$ adalah pada daerah di depan dermaga lokal. Hal itu disebabkan karena kecepatan arus-arus laut sangat kuat di daerah tersebut. Sebaliknya, energi kinetik sebesar $1,00-1,85 \text{ kW/m}^2$ pada daerah di dalam dermaga lokal. Itu menunjukkan energi kinetik sangat kecil karena kecepatan arus-arusnya sangat kecil pada daerah tersebut. Pada daerah muara sungai, energi kinetik sekitar $3,69-11,08 \text{ kW/m}^2$ adalah energi kinetik dalam taraf menengah ketersediaannya dibandingkan dengan daerah lain.

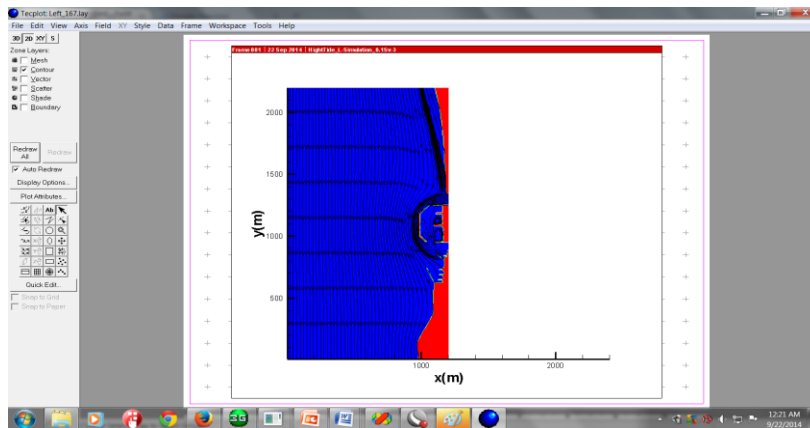
Pada saat arus laut surut, dapat disimpulkan bahwa di daerah bagian dalam dermaga yang mana sebagai tempat kapal-kapal laut berlabuh adalah paling kecil kecepatan arus-arus yang terjadi dan gelombang air sangat kecil ukurannya. Sebaliknya, pada daerah di depan dermaga lokal terjadi kecepatan arus-arus yang kuat, sehingga disarankan agar kapal-kapal laut jangan melewati daerah tersebut untuk menghindari kecelakaan akibat arus laut yang sangat kuat di tempat itu. Pada daerah muara sungai masih dalam taraf aman jika kapal-kapal laut melewati tempat itu.



Gambar 15. Distribusi energi kinetik pada 1 m kolom air

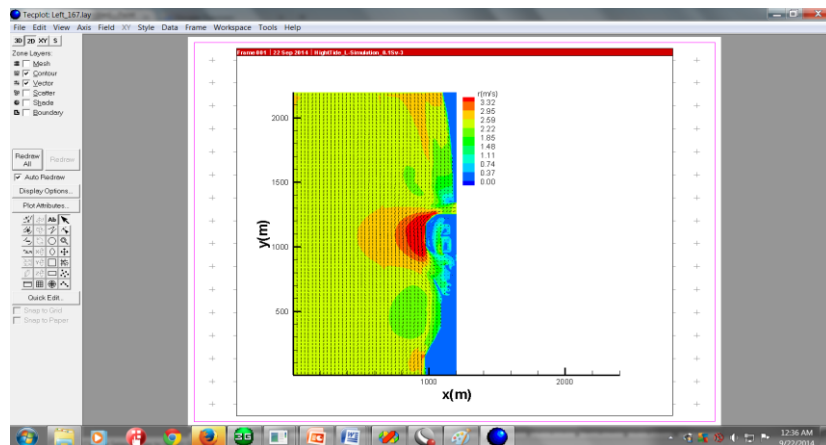
b. Saat arus laut pasang

Distribusi *streamline* arus laut pada 1 m kolom air (gambar 16) yang mana terjadi saat arus laut pasang. Itu terlihat bahwa arus bergerak dari Utara menuju Selatan. Arus laut yang bergerak disekitar muara sungai sangat kuat mendorong air sungai yang mengalir ke laut kearah Selatan dan campuran air laut dan sungai menelusuri depan dermaga lokal kemudian masuk ke dalam dermaga berbentuk turbulen-turbulen kecil.

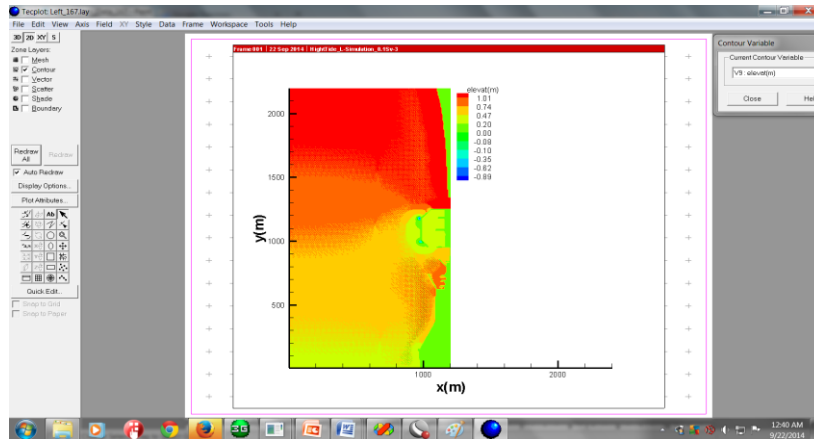


Gambar 16. Distribusi streamline arus laut pada 1 m kolom air

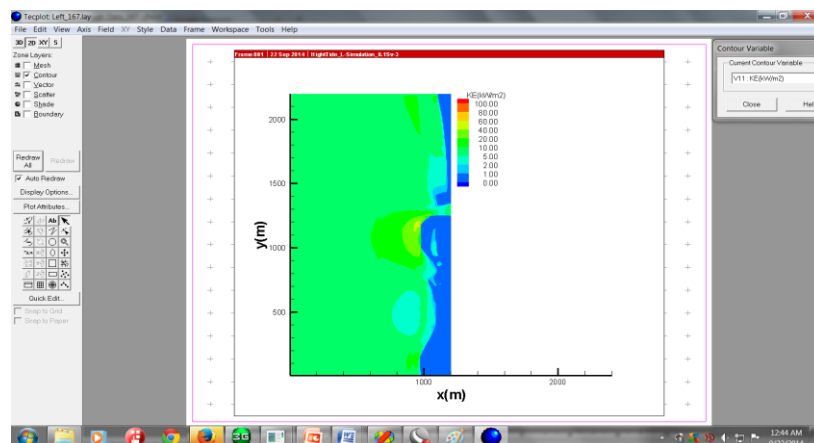
Gambar 17 menunjukkan distribusi kecepatan arus laut pada 1 m kolom air saat arus laut sedang pasang. Kecepatan arus-arus laut dan sungai secara bersama-sama melewati depan dermaga lokal dan sangat kuat yang besarnya kira-kira 2,95-3,32 m/s. Itu disebabkan karena tidak hanya kedalaman laut yang berkisar 5 m (dangkal) tetapi juga ada dua arus yaitu arus laut dan arus sungai secara bersama-sama mengalir melalui tempat itu. Gelombang air (gambar 18) pada dermaga tingginya berkisar 0,20-0,74 m sedangkan pada sekitar muara sungai tingginya 1,01 m. Energi kinetik (gambar 19) berkisar 20-100 kW/m² terjadi di depan dermaga lokal, sebaliknya di dalam dermaga sekitar 1-5 kW/m² dan di muara sungai kira-kira 10-20 kW/m².



Gambar 17. Distribusi kecepatan arus laut pada 1 m kolom air



Gambar 18. Distribusi gelombang air pada 1 m kolom air

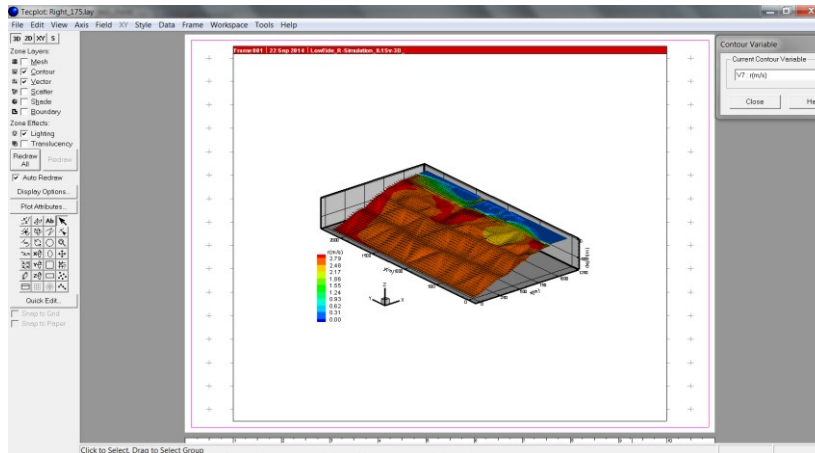


Gambar 19. Distribusi energi kinetik pada 1 m kolom air

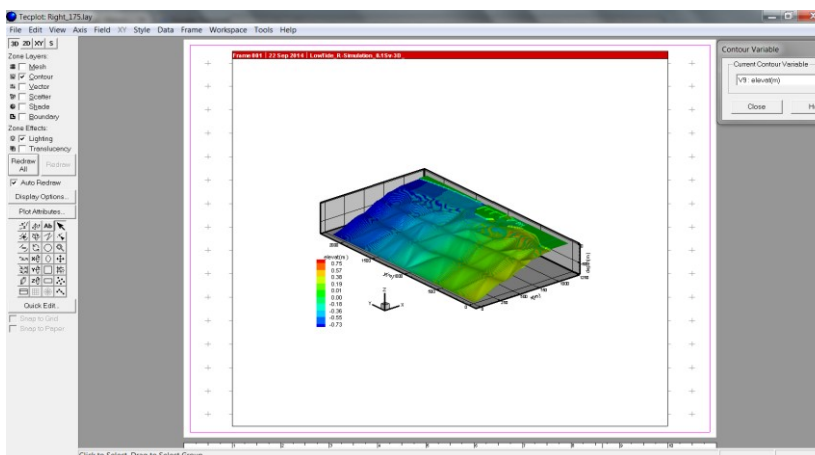
2. Hasil-hasil simulasi numerik 3D

a. Saat arus laut surut

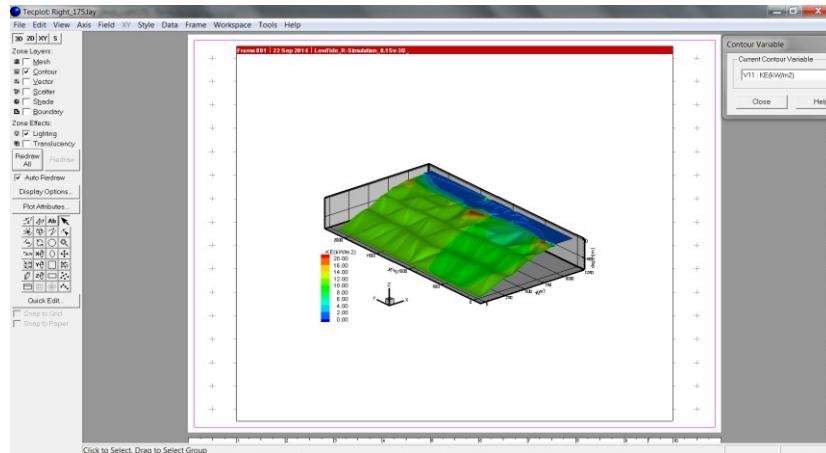
Hasil-hasil simulasi numerik 3D saat arus laut surut (gambar 20-22) tidak berbeda jauh dengan 2D. Itu karena kecepatan arus-arus dalam arah vertikal sangat kecil dibandingkan arah horizontal.



Gambar 20. Distribusi kecepatan arus laut pada 1 m kolom air



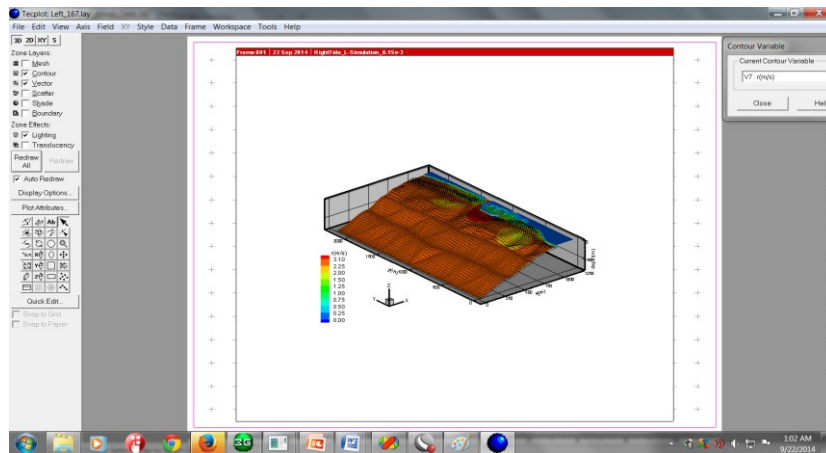
Gambar 21. Distribusi gelombang air pada 1 m kolom air



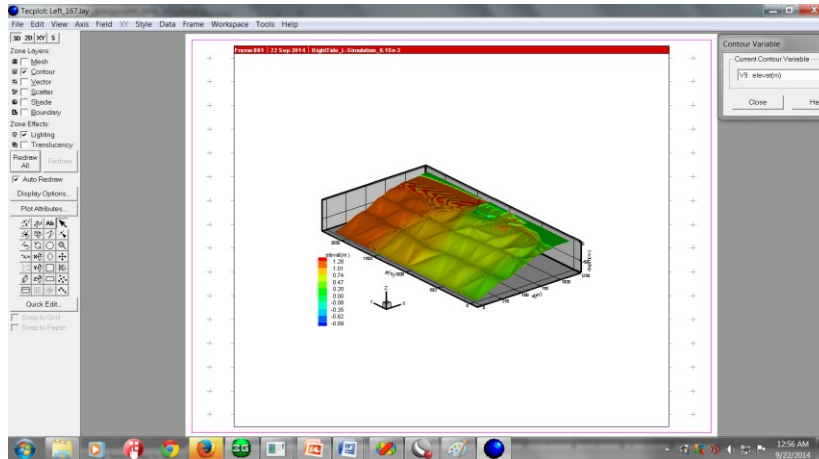
Gambar 22. Distribusi energi kinetik pada 1 m kolom air

b. Saat arus laut pasang

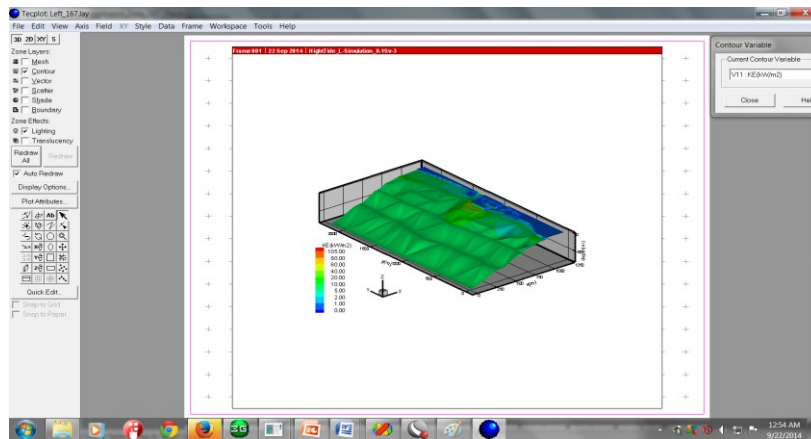
Hasil-hasil simulasi numerik 3D saat arus laut pasang (gambar 23-25) sama dengan saat arus laut surut yaitu tidak berbeda jauh dengan hasil-hasil simulasi numerik 2D. Itu karena kecepatan arus-arus dalam arah vertikal sangat kecil juga bila dibandingkan arah horizontal.



Gambar 23. Distribusi kecepatan arus laut pada 1 m kolom air



Gambar 24. Distribusi gelombang air pada 1 m kolom air



Gambar 25. Distribusi energi kinetik pada 1 m kolom air

BAB 7. KESIMPULAN

Sebuah model numerik arus laut dan sungai di teluk Manado, Sulawesi Utara, Indonesia dapat disimpulkan bahwa:

1. Tidak ada perbedaan yang besar antara simulasi numerik 2D dan 3D karena pengaruh kecepatan arus dalam arah vertikal sangat kecil sekali.
2. Distribusi kecepatan arus berkisar antara 0,00-3,32 m/s, gelombang laut berfluktuasi antara -0,89-1,01 m, dan energi kinetik berkisar antara 0,01-105 kW/m² yang mana semuanya terjadi pada kedalaman 1 m kolom air ketika air pasang dan surut pada daerah teluk Manado. Hal itu menunjukkan bahwa kecepatan dan gelombang air masih dalam batas-batas aman untuk lalu lintas perairan laut dan sungai, dan memungkinkan juga dibangun dermaga lokal disekitar pertemuan arus dan sungai, serta energi kinetik per satuan luas yang tersedia dapat digunakan dalam perancangan turbin untuk pembangkit listrik arus laut.

DAFTAR PUSTAKA

- Backhaus, J.O., 1983, A Semi-Implicit Scheme for the Shallow Water Equations for Application to Shelf Sea Modeling, *Continental Shelf Res.*, Vol. 2, pp. 243-254.
- Broomans, P., 2003, Numerical accuracy in solution of the shallow-water equations: *Master thesis*, TU Delft & WL, Delft Hydraulics.
- BC Hydro, 2002, Green Energy Study for British Columbia-Phase 2- Mainland Tidal Current Energy, Triton Consultants Ltd., <http://www.llbc.leg.bc.ca/public/PubDocs/bcdocs/357590/environment3928.pdf>, diakses tanggal 10 Februari 2012.
- Buku Putih, Indonesia 2005-2025, Penelitian, Pengembangan dan Penerapan Ilmu Pengetahuan dan Teknologi Bidang Sumber Energi Baru dan Terbarukan untuk Mendukung Keamanan Ketersediaan Energi Tahun 2025, Kementerian Negara Riset dan Teknologi Republik Indonesia, Jakarta: 2006, pp. iv-100.
- Casulli, V., 1990, Semi-Implicit Finite Difference Methods for the Two-Dimensional Shallow Water Equations, *J. Comput. Phys.*, Vol.86, pp. 56-74.
- Casulli, V. and Cheng, R.T., 1992, Semi-implicit finite difference methods for three-dimensional shallow water flow. *International Journal for Numerical Methods in Fluids*, Vol.15, pp. 629-648.
- Casulli, V. and Walters, R.A., 2000, An unstructured grid, three-dimensional model based on the shallow water equations. *International Journal for Numerical Methods in Fluids*, Vol.32, pp. 331-348.
- Cea, L., French, J.R., and Vazquez-Cendon, M.E., 2006, Numerical modelling of tidal flows in complex estuaries including turbulence: An unstructured finite volume solver and experimental validation. *International Journal for Numerical Methods in Engineering*, Vol.67, pp. 1909-1932.

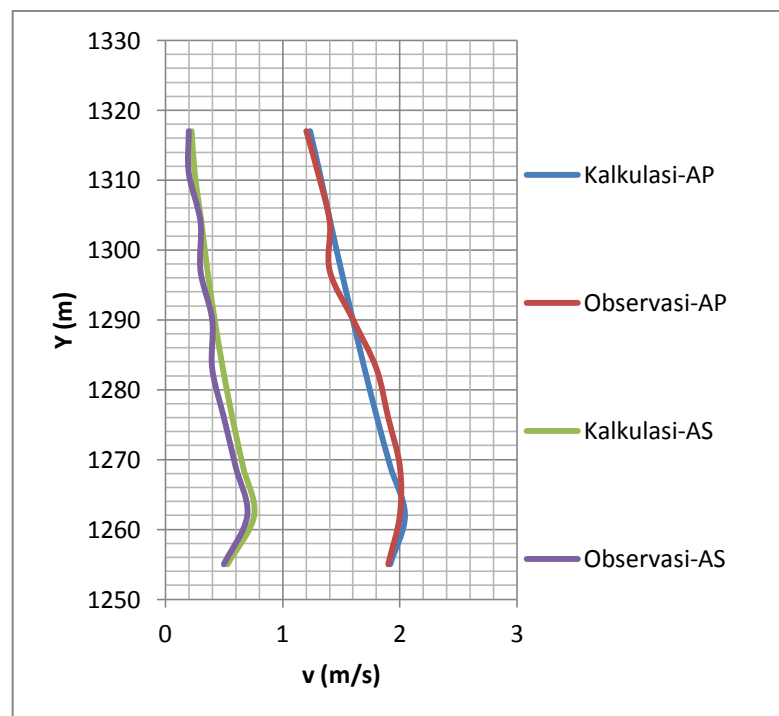
- Cheng, R.T. and Casulli, V., 1992, Tidal, Residual, Inter-Tidal Mud-Flat (TRIM) Model, Using Semi-Implicit Eulerian-Lagrangian Method, *USGS Open-File Rep.* Pp. 92-62.
- Cheng, R.T. and Smith, P.E., 1990, A Survey of Three-Dimensional Numeric Estuarine Models, dalam M.L. Spaulding (ed.), *Estuarine Coastal Modeling*, ASCE, New York, pp. 1-15.
- Cornett, A., Cousineau, J., and Nistor, L., 2013, Assessment of Hydrodynamic Impacts from Tidal Power Lagoons in the Bay of Fundy. *International Journal of Marine Energy*, Vol. 1, pp. 33-54.
- Dinas Hidro-Oceanografi TNI AL. *Daftar Pasang Surut (Tide Tables), Kepulauan Indonesia (Indonesian Archipelago)*, Jakarta: 2012, 672p.
- Fraenkel, P.L. 2002, Power from Marine Currents, Proceedings of the Institution of Mechanical Engineers. Part A: *J. Power and Energy*, Vol. 216, No. 1, pp. 1-14.
- Hervouet, J.M., 2007, *Hydrodynamics of free surface flows: Modelling with the finite element method*. John Wiley & Sons, Ltd., Englang: cop, ISBN 978-0-470-03558-0 (HB),pp. xiv-341.
- Inpres No. 4 Tahun 2003 tentang *Pengkoordinasian Perumusan dan Pelaksanaan Kebijakan Strategis Pembangunan Nasional Ilmu Pengetahuan dan Teknologi*.
- Kebijakan Energi Nasional, Kebijakan Energi yang Terpadu untuk Mendukung Pembangunan Nasional Berkelanjutan, Departemen Energi dan Sumber Daya Mineral, Jakarta, 2004.
- Luquet, R., Bellevre, D., Fréchet, D., Perdon P., and Guinard, P., 2013, Design and Model Testing of An Optimized Ducted Marine Current Turbine. *International Journal of Marine Energy*, Vol. 2, pp. 61-80.
- Peraturan Presiden No. 5 Tahun 2006 tentang *Kebijakan Energi Nasional Indonesia*.

- PT. PLN (Persero) Wilayah SULUT-TENGGGO, 2010, *Laporan Bulanan*, Manado. <http://www.pln.co.id/>. Diakses pada 21 Februari 2012
- Rodriguez, C., Serre, E., Rey, C., and Ramirez, H., 2005, A numerical model for shallow-water flows: dynamics of the eddy shedding. *WSEAS Transactions on Environment and Development*, Vol.1, pp. 280-287.
- Stansby, P.K., 1997, Semi-implicit finite volume shallow-water flow and solute transport solver with k- ϵ turbulence model. *International Journal for Numerical Methods in Fluids*, Vol.25, pp. 285-313.
- Stansby, P.K., 2003, A mixing-length model for shallow turbulent wakes. *Journal of Fluid Mechanics*, Vol.495, pp. 369-384.
- Stelling, G.S., 1984, On the Construction of Computational Methods for Shallow Water Flow Problems, *Rijkswaterstaat Communications*, No. 35, The Hague.
- Undang-undang No. 18 tahun 2002 tentang *Sistem Nasional Penelitian, Pengembangan dan Penerapan Ilmu Pengetahuan dan Teknologi*.
- Zarrati, A.R. and Jin, Y.C., 2004, Development of a generalized multi-layer model for 3-D simulation of free surface flows. *Int. J. Numer. Meth. Fluids*, Vol.46, pp. 1049-1067.

LAMPIRAN

Lampiran A: Validasi hasil kalkulasi numerik

Hasil kalkulasi numerik perlu divalidasi terhadap hasil observasi lapangan. Validasi hasil kalkulasi numerik terhadap hasil observasi di sekitar muara sungai Singkil ketika arus pasang (AP) dan surut (AS) dapat dilihat pada Gambar A. Hasil itu menunjukkan bahwa data kecepatan-kecepatan hasil kalkulasi numerik mendekati data observasi di sekitar muara sungai Singkil (pada letak dan tempat yang sama yaitu titik $X = 1580$ m dan titik Y bervariasi dari 1255-1317 m). Selain itu, telah diobservasi tentang fenomena aliran arus di permukaan laut pada lokasi yang sama dan hasilnya hampir sama dengan fenomena pada hasil kalkulasi numerik. Sehingga dapat disimpulkan bahwa semua hasil kalkulasi numerik dapat diterima untuk disimpulkan lebih lanjut.



Gambar A. Perbandingan antara kecepatan-kecepatan arus laut hasil kalkulasi numerik dan hasil observasi di titik $X=1064$ m saat arus pasang (AP) dan surut (AS).

Lampiran B: Artikel ilmiah nasional yang dimuat dalam prosiding

**Pemodelan Numerik Hasil Pertemuan
Arus Laut dan Sungai Di Teluk Manado
Propinsi Sulawesi Utara, Indonesia**

Parabelem Rompas^{1, a *}, Jenly Manongko^{2, b}

¹Pendidikan Teknik Mesin FT Unima di Tondano Sulawesi
Utara 95168, Indonesia

²Pendidikan Teknik Mesin FT Unima di Tondano Sulawesi
Utara 95168, Indonesia

^aparabelem_rompas@yahoo.com,

^bjenly_manongko@yahoo.com

Abstrak

Pemodelan numerik pertemuan arus laut dan sungai di teluk Manado Propinsi Sulawesi Utara, Indonesia telah diteliti. Tujuan penelitian adalah untuk mendapatkan data-data kecepatan arus hasil pertemuan arus-arus laut dan sungai di teluk Manado Propinsi Sulawesi Utara, Indonesia melalui pembuatan sebuah model numerik. Metode yang dipakai adalah metode beda hingga semi-implisit untuk aliran air dangkal tiga dimensi dari Hervouet (2007) melalui pengembangan model matematika menjadi sebuah model numerik. Ditemukan kecepatan arus-arus di daerah pertemuan arus laut dan sungai berbeda dibandingkan dengan arus laut saja dan arus sungai saja. Hasil itu menunjukkan bahwa kecepatan arus-arus laut dan sungai masih aman pada debit air laut dan sungai yang dimodelkan, sehingga pembuatan dermaga lokal dapat terwujud.

Kata kunci : Pemodelan numerik, Arus laut, Arus sungai, Teluk Manado

Latar Belakang

Letak geografis teluk
Manado berada pada

1°30'60.81" N dan 124°50'00.60" E. Di kota Manado terdapat satu dermaga transportasi lokal dengan tujuan ke kabupaten-kabupaten Sitaro, Sangir dan Talaud yang letaknya di bagian utara propinsi Sulawesi Utara. Juga, terdapat beberapa dermaga kecil untuk kapal-kapal sebagai transportasi menuju taman laut Bunaken di pulau Bunaken. Semuanya itu terletak berdekatan dengan muara sungai Singkil yang berasal dari danau Tondano kabupaten Minahasa (lihat Gambar 1 dan Gambar 3). Pada bagian keluar sungai itu bertemu dengan laut (teluk Manado) yang mana pada lokasi ini terjadi pertemuan dua arus yaitu arus sungai dan arus laut. Biasanya arus teluk Manado terjadi dua kali pasang dan dua kali surut yang mana pertama; pada pagi sampai siang terjadi arus surut (sekitar 0,5 m) dan pada siang sampai sore menjelang malam terjadi arus pasang (sekitar 0,5 m), dan kedua; pada sore menjelang tengah malam terjadi lagi arus surut dan pada tengah malam sampai pagi hari terjadi arus

pasang (pasang dan surut sekitar masing-masing 0,5 m).

Sampai saat ini telah dibangun beberapa dermaga kecil di sekitar muara sungai Singkil untuk tempat kapal-kapal kecil sebagai transportasi kapal pesiar bagi turis-turis. Juga, di sungai Singkil sudah lama sebagai tempat berlabuh kapal-kapal kecil sebagai transportasi local bagi masyarakat yang menuju ke pulau-pulau Manadotua, Bunaken, Mantehang, dan Naeng-besar. Suatu saat nanti akan ada lagi tambahan pembangunan beberapa dermaga kecil di sekitar sungai Singkil bagian Utara, sehingga pemerintah harus cepat mengantisipasi pembangunan itu agar terhindar dari situasi yang buruk seperti terhindar dari ancaman arus laut.

Hasil pertemuan arus laut dan arus sungai dapat dijadikan dasar kajian dalam pembangunan dermaga lokal untuk keperluan transportasi kapal laut di kota Manado. Dalam pembangunan dermaga itu sangat dibutuhkan data-data kecepatan arus-arus laut dan sungai untuk menghindari

terjadinya kecelakaan kapal saat masuk dan keluar dermaga.

Tujuan Penelitian. Untuk mengantisipasi itu maka dilakukan penelitian melalui kajian pemodelan numerik pertemuan arus-arus laut dan sungai dengan tujuan adalah untuk mendapatkan fenomena kecepatan arus hasil pertemuan arus-arus laut dan sungai di teluk Manado Propinsi Sulawesi Utara, Indonesia agar aman untuk pelayaran keluar dan masuk sungai melalui hasil pemodelan numerik berupa simulasi distribusi kecepatan arus. Berdasarkan data-data itu maka pemerintah dapat mengambil sikap dalam mengatur pembangunan dermaga-dermaga kecil di sekitar muara sungai Singkil.

Data-data itu dapat dikaji melalui pemodelan numerik yang secara teori dapat menentukan distribusi kecepatan arus di suatu daerah yang airnya dangkal [1, 2, 3, 5, 7].



Gambar 1. Teluk Manado dan sungai Singkil

Metodologi

Model Matematika. Sebelum mendapatkan persamaan-persamaan model numerik sebagai dasar dalam pemodelan, maka dibuat dahulu persamaan-persamaan model matematika. Persamaan yang dipakai adalah persamaan Navier-Stokes rata-rata Reynolds yang diasumsikan di bawah tekanan hidrostatik [1]:

$$\frac{\partial \bar{u}}{\partial t} + \bar{u} \frac{\partial \bar{u}}{\partial x} + \bar{v} \frac{\partial \bar{u}}{\partial y} + \bar{w} \frac{\partial \bar{u}}{\partial z} = -g \frac{\partial \eta}{\partial x} + \text{div}(\mathbf{v}_{\text{eff}} \overrightarrow{\text{grad}}(\bar{u})) + f_{\text{cor}}$$

$$\frac{\partial \bar{v}}{\partial t} + \bar{u} \frac{\partial \bar{v}}{\partial x} + \bar{v} \frac{\partial \bar{v}}{\partial y} + \bar{w} \frac{\partial \bar{v}}{\partial z} = -g \frac{\partial \eta}{\partial y} + \text{div}(\mathbf{v}_{\text{eff}} \overrightarrow{\text{grad}}(\bar{v})) + f_{\text{cor}} \bar{u}$$

$$\frac{\partial \bar{u}}{\partial x} + \frac{\partial \bar{v}}{\partial y} + \frac{\partial \bar{w}}{\partial z} = 0$$

$$\frac{\partial \eta}{\partial t} + \frac{\partial}{\partial x} \left(\int_{-h}^{\eta} \bar{u} dz \right) + \frac{\partial}{\partial y} \left(\int_{-h}^{\eta} \bar{v} dz \right) = 0$$

dimana $\bar{u}(x,y,z,t)$, $\bar{v}(x,y,z,t)$ dan $\bar{w}(x,y,z,t)$ adalah masing-masing komponen kecepatan dalam arah x,y horizontal, dan z vertikal, t adalah waktu, $\eta(x,y,t)$ adalah elevasi permukaan air, g is percepatan gravitasi, f_{cor} adalah parameter Coriolis. \mathbf{v}_{eff} adalah sebuah difusi efektif.

Parameter Coriolis. Parameter Coriolis diasumsikan konstan dan persamaannya adalah [8]:

$$f_{\text{cor}} = 2\omega \sin(\varphi)$$

dimana ω adalah kecepatan sudut dari bumi ($\omega = 7.29212 \times 10^{-5} \text{ s}^{-1}$) dan φ adalah latitut.

Pengaruh gaya Coriolis di dalam persamaan Navier-Stokes diatur melalui bilangan Rossby:

$$R_0 = \frac{\bar{u}_{\text{ref}}}{f_{\text{cor}} l}$$

where \bar{u}_{ref} and l adalah masing-masing kecepatan referensi dan panjang dari aliran. Itu penting hanya jika $R_0 \leq 1$ [4].

Difusi efektif. Difusi efektif adalah jumlah dari viskositas kinematik turbulen (ν_t) dan viskositas kinematik molekuler (ν_m). Viskositas kinematik turbulen digunakan formulasi kedalaman yang dirata-ratakan dari Stansby [7] yaitu:

$$\nu_t = \left[l_h^4 \left[2 \left(\frac{\partial \bar{u}}{\partial x} \right)^2 + 2 \left(\frac{\partial \bar{v}}{\partial y} \right)^2 + \left(\frac{\partial \bar{v}}{\partial x} + \frac{\partial \bar{u}}{\partial y} \right)^2 \right] + (\gamma \bar{u}_f h)^2 \right]^{1/2}$$

Kondisi-kondisi batas.

Sejauh ini daerah teluk Manado bentuknya lebih kompleks untuk aliran

permukaan bebas. Itu dibatasi:

a. *Kondisi-kondisi batas pada permukaan dan dasar laut.*

Pada permukaan laut, persamaan yang digunakan adalah [3]:

$$v_t \frac{\partial \bar{u}}{\partial z} = C_D \rho_{\text{udara}} W_{10,x} \|W_{10,x}\|$$

$$v_t \frac{\partial \bar{v}}{\partial z} = C_D \rho_{\text{udara}} W_{10,y} \|W_{10,y}\|$$

dimana

$C_D = (0.75 + 0.067W_{10})10^{-3}$ adalah sebuah koefisien gesek dari formula Garratt (1977) yang merupakan fungsi dari kecepatan angin. $\rho_{\text{udara}} = 1.178 \text{ kg/m}^3$ (pada 27 C dan 1 bar) dan kecepatan angin 10 m di atas permukaan laut pada arah x dan y adalah $W_{10,x}$ dan $W_{10,y}$.

Pada dasar laut, persamaan tegangan dasar laut bisa dihubungkan kepada hukum turbulen dinding dasar laut, sebuah koefisien gesek yang diasosiasikan dengan sebuah formula Chezy. Persamaan itu menjadi [4]:

$$v_t \frac{\partial \bar{u}}{\partial z} = -\frac{g\sqrt{(\bar{u}^2 + \bar{v}^2)}}{C_z^2} \bar{u} \quad (10)$$

$$v_t \frac{\partial \bar{v}}{\partial z} = -\frac{g\sqrt{(\bar{u}^2 + \bar{v}^2)}}{C_z^2} \bar{v} \quad (11)$$

dimana C_z adalah koefisien Chezy.

b. *Kondisi-kondisi batas pada dinding dan saluran keluar system.* (8)

Pada dinding, tegangan sama dengan nol, sehingga kecepatan dalam semua arah pada dinding sama dengan nol.

Pada saluran keluar sistem, ada dua metode yang digunakan yaitu metode von Neumann [11] dan pengembangan kondisi batas Adaptatif [12]. Persamaan-persamaan itu adalah:

Metode pertama,

$$\frac{\partial \varphi}{\partial n} = 0 \quad (12)$$

Metode kedua,

jika ($C_{\varphi x} > 0$) maka;

$$\frac{\partial \varphi}{\partial t} = -C_{\varphi x} \frac{\partial \varphi}{\partial x} + \frac{(\varphi_c - \varphi)}{\tau_o} \quad (13)$$

jika ($C_{\varphi_x} \leq 0$) maka;

$$\frac{\partial \varphi}{\partial t} = \frac{(\varphi_c - \varphi)}{\tau_i}$$

dimana C_{φ_x} dan C_{φ_y} kecepatan-kecepatan fase pada arah x dan y ke batas di dalam koordinat kartesius local. φ adalah variabel yang menjadi perlakuan untuk kondisi batas yang ditentukan. φ_c adalah sebuah perkiraan klimatologi atau observasi dari φ pada batas di saluran keluar. τ_0 dan τ_i adalah skala waktu relaksasi pada kondisi masuk dan keluar sistem.

Model Numerik. Persamaan-persamaan model numerik untuk menghitung kecepatan-kecepatan dalam arah x, y, dan z diturunkan dari persamaan-persamaan model matematika. Penyelesaian numerik tiga dimensi digunakan metode beda hingga semi implisit [2,3,6]. Ada tiga langkah dalam penyelesaian model numerik dari persamaan-persamaan model matematika Pers. 1, Pers. 2, Pers. 3, dan Pers. 4 yaitu sebagai berikut:

1. *Langkah adveksi.*

Langkah adveksi menggunakan diskretisasi *Eulerian-Lagrangian* dari hubungan konveksi dan difusi. Persamaannya adalah sebagai berikut [2]:

$$C_{i-a,j-b,k-d}^n = (1-r)\{(1-p) [(1-q)C_{i-1,j-m,k-n}^n + qC_{i-1,j-m,k-n}^n] + p[(1-q)C_{i-l-1,j-m,k-n}^n + qC_{i-l-1,j-m-1,k-n}^n]\} + r\{(1-p)[(1-q)C_{i-l,j-m,k-n-1}^n + qC_{i-l,j-m-1,k-n-1}^n] + p[(1-q)C_{i-l-1,j-m,k-n-1}^n + qC_{i-l-1,j-m-1,k-n-1}^n]\}$$

dengan kondisi stabilnya adalah:

$$\Delta t \leq \left[2\mu \left(\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2} \right) \right]^{-1} \quad (16)$$

jika $\mu=0$, maka kondisi menjadi tidak stabil.

2. *Langkah difusi.*

Sebuah diskretisasi semi implisit umum dari Pers. 1 dan Pers. 2 bisa ditulis ke dalam bentuk matriks vector [2]:

$$A_{i+1/2,j}^n U_{i+1/2,j}^{n+1} = G_{i+1/2,j}^n - g \frac{\Delta t}{\Delta x} (\eta_{i+1,j}^{n+1} - \eta_{i,j}^{n+1}) \Delta Z_{i+1/2,j}^n \quad (17)$$

$$A_{i,j+1/2}^n V_{i,j+1/2}^{n+1} = G_{i,j+1/2}^n - g \frac{\Delta t}{\Delta y} (\eta_{i,j+1}^{n+1} - \eta_{i,j}^{n+1}) \Delta Z_{i,j}^n \bar{w}_{i,j,k+1/2}^{n+1} = \bar{w}_{i,j,k-1/2}^{n+1} - \frac{\Delta Z_{i+1/2,j,k}^n \bar{u}_{i+1/2,j,k}^{n+1} - \Delta Z_{i-1/2,j,k}^n \bar{u}_{i-1/2,j,k}^{n+1}}{\Delta x} - \frac{\Delta Z_{i,j+1/2,k}^n \bar{v}_{i,j+1/2,k}^{n+1} - \Delta Z_{i,j-1/2,k}^n \bar{v}_{i,j-1/2,k}^{n+1}}{\Delta y} \quad (18)$$

dimana **A**, **U**, **V**, **G**, dan ΔZ adalah vector-vektor yang dituangkan dalam bentuk matriks. Pers. 17 dan Pers. 8 adalah sistem tridiagonal linier yang mana di kopel ke elevasi permukaan laut η^{n+1} dan waktu t_{n+1} .

3. Langkah tekanan kontinuitas. Persamaan yang digunakan adalah pengembangan dari Pers. 4, Pers. 17, dan Pers. 18 menjadi persamaan baru [2]:

$$\eta_{i,j}^{n+1} - g \frac{\Delta t^2}{\Delta x^2} \left\{ (\Delta \Delta Z A^{-1} \Delta Z)_{j+1/2,j}^n (\eta_{i,j+1}^{n+1} - \eta_{i,j}^{n+1}) - [(\Delta \Delta Z A^{-1} \Delta Z)_{j-1/2,j}^n (\eta_{i,j}^{n+1} - \eta_{i,j+1}^{n+1})] \right\} - g \frac{\Delta t^2}{\Delta y^2} \left\{ (\Delta \Delta Z A^{-1} \Delta Z)_{j+1/2,j}^n (\eta_{i,j+1}^{n+1} - \eta_{i,j}^{n+1}) - [(\Delta \Delta Z A^{-1} \Delta Z)_{j-1/2,j}^n (\eta_{i,j}^{n+1} - \eta_{i,j+1}^{n+1})] \right\} = \eta_{i,j}^n - \frac{\Delta t}{\Delta x} \left\{ (\Delta Z)^T A^{-1} G \right\}_{j+1/2,j}^n - \left\{ (\Delta Z)^T A^{-1} G \right\}_{j-1/2,j}^n \left. \right\} - \frac{\Delta t}{\Delta y} \left\{ (\Delta Z)^T A^{-1} G \right\}_{j+1/2,j}^n - \left\{ (\Delta Z)^T A^{-1} G \right\}_{j-1/2,j}^n \quad (19)$$

Untuk mendapatkan kecepatan \bar{w} dalam arah z (arah ke dasar laut) dikembangkan dari Pers. 3 menjadi:

dimana $k=m, m+1, \dots, M$, dan $\bar{w}_{i,j,m-1/2}^{n+1} = 0$ artinya tidak ada aliran melintasi daerah ke dasar laut.

Metode. Metode yang dipakai adalah mula-mula studi pustaka, pengambilan data input berupa: peta lokasi numerik melalui penelusuran *website google earth*, peta teluk Manado dan sekitarnya beserta nilai-nilai kedalaman laut normal, hasil-hasil prediksi arus pasang-surut yang terjadi di teluk Manado dan sekitarnya [10], pengukuran-pengukuran lebar dan kedalaman sungai di sekitar muara sungai Singkil. Langkah-langkah dalam kalkulasi numerik dapat dilihat pada Gambar 1.

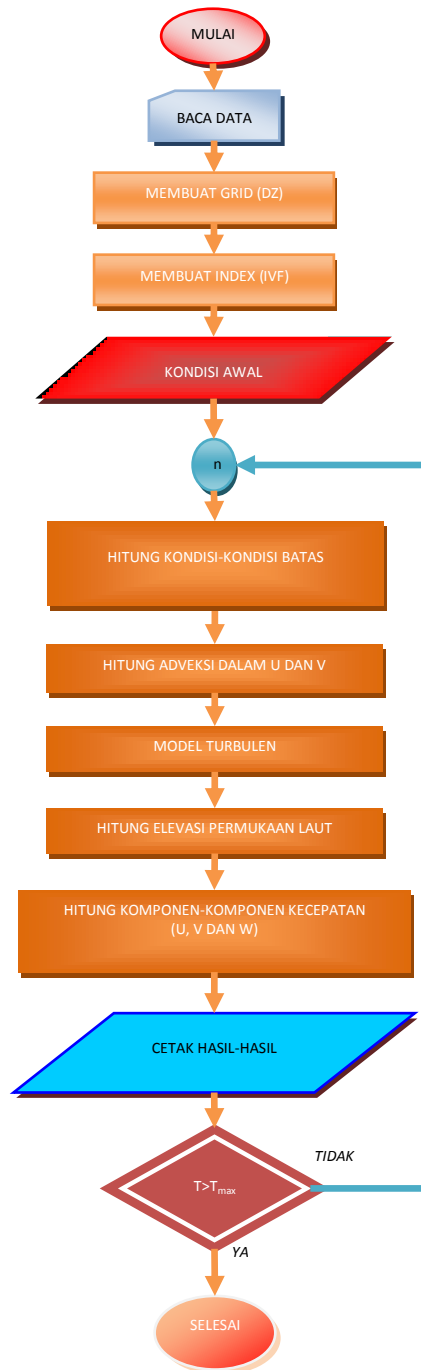
Data set. Kalkulasi numerik 3D untuk pemodelan hasil pertemuan arus-arus yang berasal dari laut teluk Manado dan sungai Singkil digunakan program fortran 90. Kalkulasi terdiri dari dua kondisi yaitu pada kondisi aliran air pasang dan surut.

Data-data awal yang dimasukkan adalah $\Delta t = 1$ s, $g = 9,81$ m/s², $Cz = 48,04$, $W_{10} = 1$ m/s², iterasi maksimum $T_{max} = 720000$, densitas udara $1,178$ kg/m³, $\rho_{air} = 1024$ kg/m³, dx , dy , dan dz masing-masing 7 m, 7 m, dan 1 m, $\omega = 7.29212 \times 10^{-5}$ s⁻¹ dengan sudut $1,45^\circ$, $\tau_0 = 2$ hari dan $\tau_i = 1$ hari, debit air laut saat masuk adalah 100×10^3 m³/s.

Pembuatan grid. Grid dibuat dengan langkah-langkah sebagai berikut: pertama-tama peta lokasi numerik dalam bentuk *bathymetry* dibuat format DXF kemudian di import ke program Argus ONE. Pembuatan grid membutuhkan langkah-langkah pengerjaan sampai pada tahap ekspor grid dalam bentuk format EXP yang dipakai sebagai data-data grid dalam kalkulasi yang berisi seperti jumlah grid arah x dan y , indeks 0 untuk daratan dan 1 untuk lautan, dan terakhir data kedalaman laut hasil interpolasi yang dihitung dari program Argus ONE. Hasilnya adalah grid untuk arah x , y , dan z masing-masing adalah 174 , 318 , dan 2 . Luas bidang numerik adalah

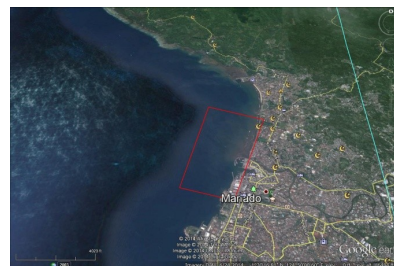
lebar $1,218$ km x panjang $2,228$ km (lihat Gambar 3). Maksimum kedalaman laut adalah 79 m.

Pembuatan indeks. Indeks dibuat dengan memberi penomoran yaitu 99 , 88 , dan 77 . Jika $dz = 0$ maka diberi 99 sedangkan jika $dz \neq 0$ maka diberi tanda 1 . Juga, penomoran 99 diberi tanda bahwa tidak ada kecepatan-kecepatan air arah x , y , dan z . Penomoran 88 dipakai sebagai tanda untuk kecepatan-kecepatan air dan elevasi permukaan air pada daerah masuk dan keluar sistem. Terakhir penomoran 77 dipakai sebagai tanda pada perhitungan kecepatan-kecepatan arah x dan y dalam langkah adveksi dan dalam perhitungan elevasi permukaan laut.



Gambar 2. Gambar alir kalkulasi numerik

Kondisi awal. Data-data kondisi awal yang dimasukkan ada dua kondisi yaitu pertama, saat kalkulasi baru dan kedua, saat sudah pernah kalkulasi dan akan dilanjutkan lagi. Pada kondisi pertama, semua kecepatan air sama dengan nol baik lama maupun baru, elevasi permukaan laut juga sama dengan nol, dan viskositas-viskositas turbulen arah x dan y sama dengan viskositas kinematik molekuler. Pada kondisi kedua, hanya data-data lama hasil kalkulasi sebelumnya seperti kecepatan-kecepatan air dan elevasi permukaan laut yang menjadi kondisi awal.



Gambar 3. Peta teluk Manado dan lokasi numerik

Cetak hasil kalkulasi. Sementara kalkulasi dilakukan maka data-data direkam dan dicetak dan ketika iterasi

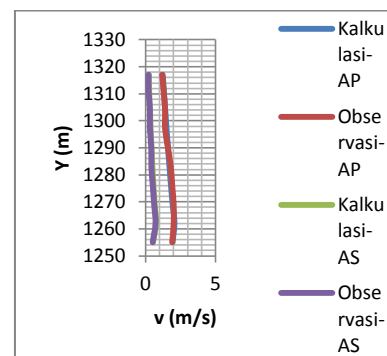
mencapai maksimum (T_{max}) maka kalkulasi selesai. Data-data hasil kalkulasi kemudian disimulasikan dengan menggunakan program Tecplot 360.

Validasi hasil penelitian. Validasi data hasil penelitian dilakukan dengan cara membandingkan kecepatan-kecepatan arus laut hasil kalkulasi numerik dan hasil observasi di sekitar muara sungai Singkil (pada 1 m di bawah permukaan laut). Data-data observasi diukur bersamaan dengan pengukuran kecepatan arus sungai masuk ke laut. Juga, diobservasi fenomena aliran di lokasi tersebut. Jika hasil kalkulasi numerik mendekati hasil observasi maka hasil kalkulasi numerik dapat diterima.

Hasil dan Pembahasan

Validasi hasil kalkulasi numerik terhadap hasil observasi di sekitar muara sungai Singkil ketika arus pasang (AP) dan surut (AS) dapat dilihat pada Gambar 4. Hasil itu menunjukkan bahwa data kecepatan-kecepatan hasil kalkulasi numerik mendekati data observasi di sekitar muara

sungai Singkil (pada letak dan tempat yang sama yaitu titik $X = 1580$ m dan titik Y bervariasi dari 1255-1317 m). Selain itu, telah diobservasi tentang fenomena aliran arus di permukaan laut pada lokasi yang sama dan hasilnya hampir sama dengan fenomena pada hasil kalkulasi numerik. Sehingga dapat disimpulkan bahwa semua hasil kalkulasi numerik dapat diterima untuk disimpulkan lebih lanjut.



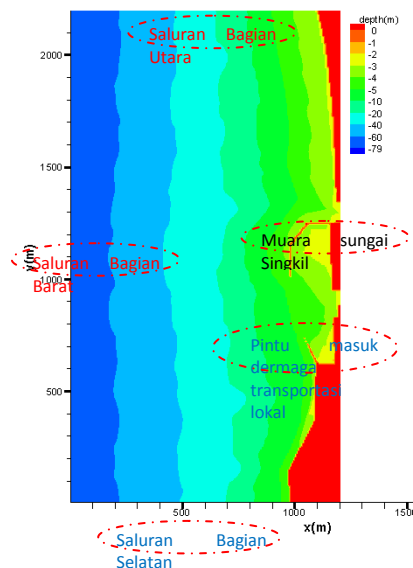
Gambar 4. Perbandingan antara kecepatan-kecepatan arus laut hasil kalkulasi numerik dan hasil observasi di titik $X=1064$ m saat arus pasang (AP) dan surut (AS).

Hasil kalkulasi numerik 3D dapat dilihat pada Gambar 5, 6 dan 7 dalam bentuk simulasi pada kondisi arus pasang dan

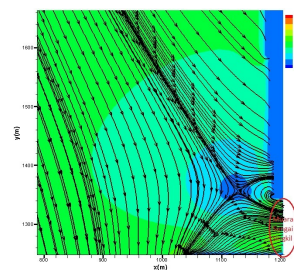
arus surut. Semua simulasi itu adalah hasil kalkulasi sampai pada saat debit masuk dan keluar sistem telah sama atau hanya selisih 0,01 %.

Gambar 5 menunjukkan *bathymetry* dari lokasi numerik, Terdapat 2 sisi aliran air masuk dan 3 sisi aliran air keluar sistem saat arus pasang. Pada sisi masuk, pertama berada pada daerah aliran sungai Singkil dan kedua pada bagian utara. Pada sisi keluar, pertama pada daerah bagian timur di dekat dari aliran sungai Singkil bagian selatan (ada dermaga transportasi lokal), kedua, pada daerah bagian barat, dan ketiga, pada daerah bagian selatan. Saat arus surut, terdapat 2 sisi aliran air masuk dan 3 sisi keluar. Di sisi masuk yaitu pada daerah-daerah aliran sungai Singkil dan bagian selatan, sedangkan di sisi keluar yaitu pada daerah-daerah bagian timur dan utara serta bagian timur dekat aliran sungai Singkil. Pada sisi keluar di daerah dekat sungai Singkil tidak terjadi pengeluaran debit air karena di situ terdapat dermaga transportasi lokal yang mana

air tidak mengalir keluar. Kedalaman maksimum yaitu 79 m berada pada daerah bagian barat. Kedalaman rata-rata di muara sungai Singkil adalah 2,5 m pada level air normal.



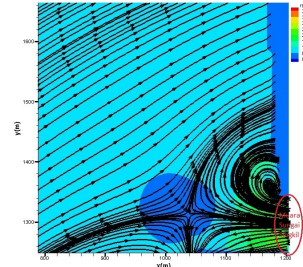
Gambar 5. *Bathymetry* lokasi numerik



Gambar 6. Garis-garis aliran air pada kondisi arus pasang dan 1 m kolom air laut

Gambar 6 menunjukkan garis-garis aliran air pada saat arus pasang. Daerah pertemuan arus sungai Singkil dan arus laut terdapat perempatan aliran yang berada pada sisi laut arah barat ke utara dengan jarak kira-kira 80 m sampai 108 m dari muara sungai (pada sumbu x dan y masing-masing 1126 m dan 1359 m sampai radius dari sumbu itu 14 m) dengan kecepatan arus dan kedalaman rata-rata masing-masing 0,28 m/s dan 3,6 m. Hal itu menunjukkan bahwa sangat aman jika kapal-kapal melewati daerah itu baik masuk maupun keluar sungai.

Gambar 7 sama dengan Gambar 6 tetapi pada kondisi arus surut. Itu menunjukkan bahwa daerah aman untuk pelayaran kapal-kapal adalah pada jarak 170 m sampai 220 m dari muara sungai (pada sumbu x dan y masing-masing 1040 m dan 1310 m sampai radius dari sumbu itu 28 m) dengan kecepatan air dan kedalaman rata-rata masing-masing 0,3 m/s dan 4,7 m.



Gambar 7. Garis-garis aliran air pada kondisi arus surut dan 1 m kolom air laut

Fenomena aliran air yang terjadi pada kondisi arus pasang dan arus surut ternyata sama yaitu membentuk seperti perampatan tetapi letak kejadiannya berbeda (lihat Gambar 6 dan 7). Hal itu dipengaruhi oleh arah aliran air yang mana pada kondisi arus pasang alirannya dari arah Utara sedangkan pada kondisi arus surut dari arah Selatan, saat bersamaan juga aliran sungai dari arah Timur lebih besar kecepatannya dibandingkan kecepatan air dari laut teluk Manado. Kecepatan air sungai tidak sama pada saat arus pasang 2,43 m/s dan 2,50 m/s saat arus surut. Itu disebabkan karena saat arus pasang maka kecepatan air sungai menjadi kecil terhubung debit air dari laut teluk Manado naik

sehingga menahan lajunya arus dari sungai yang menuju ke laut. Sebaliknya, saat arus surut tidak terjadi demikian sehingga arus sungai menuju ke laut sangat bebas. Disamping itu juga, kecepatan arus sungai dipengaruhi oleh elevasi permukaan air sungai akibat adanya gelombang laut di muara sungai.

Kesimpulan

Kecepatan-kecepatan air pada saat pertemuan arus-arus laut dan sungai yang letaknya berada pada jarak 70-220 m pada debit air laut $100 \times 10^3 \text{ m}^3/\text{s}$ baik saat arus pasang maupun arus surut adalah sangat aman untuk pelayaran keluar dan masuk sungai, sehingga pembuatan dermaga lokal di daerah aliran sungai dapat terhindar dari kecelakaan akibat pertemuan arus tersebut.

Referensi

[1] P. Broomans, Numerical accuracy in solution of the shallow-water equations, Master thesis, TU Delft & WL, Delft Hydraulics. 2003

[2] V. Casulli, R.T. Cheng, Semi-implicit finite difference methods for three-dimensional shallow water flow, *International Journal for Numerical Methods in Fluids*. 15 (1992) 629-648.

[3] X. Chen, A free-surface correction method for simulating shallow water flows, *Journal of Computational Physics*, 189 (2003) 557-578.

[4] J.M. Hervouet, *Hydrodynamics of free surface flows: Modelling with the finite element method*, John Wiley & Sons, Ltd., England: cop, ISBN 978-0-470-03558-0 (HB), (2007) xiv-341.

[5] C. Rodriguez, E. Serre, C. Rey, H. Ramirez, A numerical model for shallow-water flows: dynamics of the eddy shedding, *WSEAS Transactions on Environment and Development*, 1 (2005) 280-287.

[6] P.K. Stansby, Semi-implicit finite volume shallow-water flow and solute transport solver with k-ε turbulence model, *International Journal for*

**Proceeding Seminar Nasional Tahunan Teknik Mesin XIII
(SNTTM XIII)**

UI Depok, 15 – 16 Oktober 2014

- Numerical Methods in Fluids, 25 (1997) 285-313.
- [7] P.K. Stansby, Limitations of Depth-Averaged Modelling for Shallow Wakes. *Journal of Hydraulic Engineering*, 132 Issue 7 (2006) 737-740.
- [8] A.R. Zarrati, Y.C. Jin, Development of a generalized multi-layer model for 3-D simulation of free surface flows, *Int. J. Numer. Meth. Fluids*, 46 (2004) 1049-1067.
- [9] E. Brown, A. Colling, D. Park, *Ocean Circulation. Second Edition*, The Open University, Walton Hall, Milton Keynes, MK7 6AA and Butterworth-Heinemann, England: ISBN 0 7506 5278 0, cop. (2001) 286.
- [10] Dinas Hidro-Oceanografi TNI AL, *Daftar Pasang Surut (Tide Tables)*, Kepulauan Indonesia (Indonesian Archipelago), Jakarta: (2006) 672.
- [11] I. Orlanski, A Simple Boundary Condition for Unbounded Hyperbolic Flow, *J. Comput. Phys.*, 21 (1976) 251-269.
- [12] A.M. Treguier, B. Barnier, A.P. De Miranda, An Eddy-permitting Model of the Atlantic Circulation: Evaluating Open Boundary Conditions. *J. Geophys. Res.Oceans*, 106(C10):22115-22129 (2001) 1-23.



Parabelem T.D. ROMPAS adalah dosen mata kuliah: fisika, perpindahan kalor, termodinamika teknik, teknik pendingin, metode penelitian, pesawat kerja, pneumatik dan hidrolik, matematika struktur, aljabar linier, metode numerik, dan mekanika fluida pada jurusan Pendidikan Teknik Mesin dan Pendidikan Teknologi Informatika dan Komputer fakultas teknik Universitas Negeri Manado sejak Maret tahun 1992 sampai sekarang. Pada Agustus tahun 1991, lulus Sarjana Pendidikan Teknik Mesin di Institut Keguruan dan Ilmu Pendidikan (IKIP) Manado dengan skripsi tentang “*Peranan Pelaksanaan Responsi pada Mata Kuliah Matematika Dihubungkan dengan Hasil Belajar Matematika Mahasiswa Jurusan Pendidikan Teknik Mesin FPTK IKIP Manado*”. Pada Juli tahun 1999, lulus Magister Teknik Mesin di Universitas Gadjah Mada (UGM) Yogyakarta dengan tesis tentang “*Karakteristik Perubahan Perpindahan Kalor pada Sirip yang Dipengaruhi oleh Pembentukan Bunga Es pada Sirip Tersebut*”. Pada Desember tahun 2008 lulus Doktor Teknik Mesin di Aix-Marseille Universite, Marseille, France dengan disertasi tentang “*Sebuah Model Numerik untuk Studi Arus-arus Laut di Selat Bangka Sulawesi Utara Indonesia*” yang intinya menyelidiki potensi energi yang terkandung di selat bangka sebagai dasar untuk pembangunan *pembangkit listrik arus bawah laut*.

ISBN 978-602-1376-17-1

