VORtech Computing

Experts in Technisch Rekenwerk

Postbus 260
2600 AG DELFT

tel. 015-285 0125
fax. 015-285 0126
vortech@vortech.nl

Detailontwerp verbetering droogvallen en onderlopen in WAQUA/TRIWAQ

Technisch Rapport TR04-02 versie 1.3

Datum

15 november 2004

Auteur(s)

dr.ir. E.A.H. Vollebregt
dr.ir. B. van 't Hof
ir. J.A.Th.M. van Kester (WL|Delft Hydraulics)

In opdracht van

Rijkswaterstaat/RIKZ (overeenkomst RKZ-1407)

Log-sheet

Versie	Auteur	Datum	Opmerkingen	Review
0.1	EV	10-05-2004	Eerste werkversie	
0.2	EV	26-05-2004	Werkversie activiteiten 3-8, 3-7, 3-4	JvK
0.3	EV	01-06-2004	Werkversie activiteiten 3-2, 3-5	JvK
0.4	BvtH, EV	03-06-2004	Werkversie activiteiten 2-4, 2-1, 3-3, 3-1, 3-6	JvK
0.5	EV, BvtH	04-06-2004	Werkversie activiteiten 2-3, 2-2, 2-5	JvK
0.9	EV	07-06-2004	Eindversie t.b.v. bespreking fase 1	
0.95	EV	09-06-2004	Overzicht van keuzes en opties toegevoegd	JvK
1.0	EV	24-06-2004	Aanpassingen van hoofdstuk 2 n.a.v. bespreking opdrachtgever	
1.1	EV	31-08-2004	Aanpassingen van hoofdstuk 2 n.a.v. implementatie	
1.2	EV	11-10-2004	Aanpassingen van hoofdstuk 3 n.a.v. implementatie 3-1, 3-7, 3-8, 3-4 (TRIWAQ)	
1.3	EV	15-11-2004	Aanpassingen van hoofdstuk 3 n.a.v. implementatie 3-2, 3-5, 3-6 (WAQUA)	
Bestandslokatie:		/v3/E05k_droogval/doc/detail_droog		

Inhoudsopgave

Hoofdstuk 1

Inleiding

1.1 Achtergrond van het huidige project

Simulatie van droogvallen en onderlopen van intertijdegebieden, uiterwaarden en polders etc. is al jarenlang een zorgenkind bij de SIMONA gebruikers. De huidige algoritmes en implementatie van droogvallen en onderlopen leiden soms tot instabiel gedrag van de modelresultaten. Bij kleine aanpassingen in de modelparameters, bodem of schematisatie kan er plots een groot verschil in waterstand en zoutconcentratie optreden [11]. De gebruikers hebben zich in 2003 dan ook vrijwel unaniem uitgesproken voor de noodzaak van robuuste droogval- en onderlooproutines.

Deze gevoeligheid van de modellen is niet alleen een probleem voor de gebruikers maar ook voor de beheerders en ontwikkelaars van de programmatuur, omdat kleine modelwijzigingen in sommige gevallen grote gevolgen hebben voor de modelresultaten. Dit leidde tot extra inspanningen om vast te stellen of de geconstateerde instabiliteit zijn oorsprong had in de nieuwe ontwikkeling dan wel in het droogvallen en onderlopen. Hierdoor moesten vaak (onnodig) meerkosten worden gemaakt en werd de invoering van nieuwe ontwikkelingen vertraagd.

In 2002 hebben V$\mathcal{O}$Rtech Computing en WL|Delft Hydraulics een inventarisatie gemaakt van hoe er verbeteringen tot stand kunnen worden gebracht [9]. Op basis daarvan heeft Rijkswaterstaat/RIKZ op 29 maart 2004 middels haar brief met kenmerk RIKZ/2004/05308 offerte aangevraagd voor de implementatie van de meeste van de gesuggereerde verbeteringen voor de korte termijn. V$\mathcal{O}$Rtech heeft hiervoor samen met WL een aanbieding gedaan en is op 27 april de opdracht gegund.

1.2 Doel van het huidige project

Het doel van het huidige project is te komen tot een nieuwe versie van WAQUA/TRIWAQ:

- die sterk is voorbereid op de overgang naar OMS,

- waarin de "beste" droogvalmethode is geïmplementeerd die ook wordt nagestreefd binnen OMS,
- waarin WAQUA en TRIWAQ sterk zijn geüniformeerd,
- waarin een aantal ongewenste gedragingen is geëlimineerd,
- en waarin de gevoeligheid voor verstoringen enigszins verminderd is.

1.3 Doel van het huidige rapport

Het onderhavige rapport bevat het technisch ontwerp van de beoogde wijzigingen aan WAQUA/TRIWAQ. Het wordt gebruikt om in een vroeg stadium de wenselijkheid en uitvoerbaarheid van de voorgestelde aanpak te kunnen evalueren. Daarnaast geeft het houvast bij de daadwerkelijke implementatie van de wijzigingen. Tenslotte geeft het na afloop van het project een overzicht van wat er in het kader hiervan is aangepast. Voor dit laatste doel zullen wijzigingen in de aanpak die gedurende de implementatiefasen worden gemaakt in het rapport worden verwerkt.

1.4 Indeling van dit rapport

In het huidige project wordt onderscheid gemaakt tussen wijzigingen die geen of wel gevolgen hebben voor bestaande simulaties. De eerste groep aanpassingen wordt "herstructureren van de code" genoemd en wordt geïmplementeerd in fase 2 van het project (fase 1 betreft het schrijven van het onderhavige detailontwerp). Deze wijzigingen worden beschreven in Hoofdstuk 2 van dit rapport. De tweede groep wijzigingen betreft "verbeteringen van de huidige algoritmen". Deze wijzigingen worden uitgevoerd in fase 3 van het project en worden beschreven in Hoofdstuk 3.

Hoofdstuk 2

Fase 2: herstructurering van de programmatuur

2.1 Overzicht

In de tweede fase worden de activiteiten van Tabel 1 van het werkplan uitgevoerd, hier herhaald als Tabel 2.1.

We beginnen met activiteit 2-4 omdat dit de andere werkzaamheden vergemakkelijkt. Daarna wordt activiteit 2-3 uitgevoerd. Hierbij wordt in alle rekenroutines het array KHU vervangen door KFU, KCU. Dit is een vrij mechanische bewerking. Ook worden waar dat van toepassing is de arrays DPU en DPV ingevoerd. Vervolgens wordt het informatica-technische gedeelte van activiteit 2-2 gedaan: vervangen van KFU door KFUP en KFUH. Tenslotte worden activiteiten 2-1 en 2-5 gedaan.

2.2 Activiteit 2-4: Verwijderen `lgrid`

2.2.1 Achtergrond

Het verwijderen van `lgrid` uit WAQUA betreft een relatief omvangrijke activiteit waarin de programmatuur intern stevig wordt gereorganiseerd. Dit is nuttig omdat hiermee WAQUA en TRIWAQ sterk op elkaar worden afgestemd. De onderhoudbaarheid en uitbreidbaarheid

Tabel 2.1: *Overzicht van de verschillende activiteiten in Fase 2.*

Nr.	Wat?	Soort	Rapport
2-1	*Gebruik DUPWND*	*Herstructureren Code*	*Hfd 6 - wens 3*
2-2	*Tijdsniveau toevoegen aan kenmerkarrays*	*Herstructureren Code*	*Hfd 6 - wens 7*
2-3	*Arrays DPU, DPV, KCU, KCV*	*Herstructureren Code*	*Hfd 6 - wens 8*
2-4	*Verwijderen LGRID uit WAQUA*	*Herstructureren Code*	-
2-5	*Dieptecijfers opgeven waterstandspunten*	*Nieuwe functionaliteit*	*Hfd 6 - wens 13*

van de programmatuur wordt hier een stuk beter door. Bijvoorbeeld kunnen nieuwe features die voor het ene model worden geïmplementeerd (maximale waterstand, QH-randvoorwaarde, droogvalalgoritmes, ...) veel gemakkelijker ook in het andere model beschikbaar worden gemaakt.

Het verwijderen van `lgrid` is ook prettig voor het echte rekenwerk. Dat komt omdat er veel meer met de zogenaamde "`fullbox`-arrays" dan met "`mnmaxk`-arrays" zal worden gewerkt. Het `fullbox`-formaat is op directe adressering gebaseerd: roosterpunt (m, n) wordt afgebeeld op array-element `(n,m)`. Het `mnmaxk`-formaat gebruikt juist indirecte adressering via `lgrid`: $(m, n) \rightarrow$ array-index `nm=lgrid(n,m)`.

Nadelen van het verwijderen van `lgrid` zijn dat het `mnmaxk`-formaat efficiënter met het geheugen omgaat, en dat ermee in een aantal situaties sneller mee gerekend wordt. Het `mnmaxk`-formaat is efficiënt qua geheugenopslag omdat hierin alleen met het natte gedeelte van het rooster wordt gewerkt. Afhankelijk van de vullingsgraad van een gebiedsschematisatie scheelt dit tot een factor 5 in de benodigde hoeveelheid geheugen (buffersize). Wanneer dit verschil voor een grote schematisatie belangrijk wordt dan kan het geheugengebruik via parallel rekenen weer worden teruggebracht.

Het performanceverschil ontstaat ook doordat alleen met het natte gedeelte van het rooster wordt gewerkt. Daardoor wordt er op verschillende computerplatformen het cache-geheugen beter gebruikt. Experimenten hebben laten zien dat dit effect een paar procent bedraagt op een Linux PC, en dat dit kan oplopen tot 15% op computers die op MIPS processoren zijn gebaseerd. Dit verlies van performance zal naar verwachting volledig met het aanpassen van de massacorrectiestap in activiteit 3-7 worden gecompenseerd.

2.2.2 Aanpak van het verwijderen van `lgrid`

Een eerste keuze die wordt gemaakt bij het uitwerken van deze activiteit is dat ze alleen zal worden uitgevoerd voor het programma WAQPRO. De inhoud van de SDS-file wordt niet gewijzigd, bij de opslag van gegevens zal het `mnmaxk`-formaat blijven worden gebruikt. Dit is nodig omdat bij SDS-files de omvang wel degelijk belangrijk is, en vanwege compatibiliteit met alle andere programmatuur. Ook levert de activiteit binnen WAQPRE niet zo veel voordeel op.

De werkzaamheden worden in een aantal stappen uitgevoerd. Na elke stap zal een volledig functionerende tussenversie van WAQPRO gemaakt worden, die met behulp van de regressietestbank zal worden gevalideerd. De verschillen die optreden in de simulatieresultaten mogen niet groter zijn dan de machinenauwkeurigheid. We verwachten zelfs dat de meeste stappen totaal geen verschillen opleveren.

De stappen in de uitvoering van deze activiteit zijn:

1. **Ontwerp: hoe wordt omgegaan met WAQGEN**

   Een lastig punt bij het verwijderen van het rekenen met `lgrid` betreft het deelsysteem WAQGEN. Routines hiervan kunnen niet zomaar worden aangepast omdat ze ook in andere programma's worden gebruikt, zoals WAQPRE en COEXEC.

In Sectie 2.2.3 wordt besproken hoe er met het gebruik van `lgrid` in WAQGEN wordt omgegaan.

2. **Ontwerp: Space Varying Wind and Pressure**

   De berekeningen voor een aantal speciale functies worden momenteel in het geheel op `mnmaxk`-arrays uitgevoerd, ook wanneer er met TRIWAQ gerekend wordt. Dit betreft bijvoorbeeld de berekening voor variabele wind en druk (SVWP) en de harmonische analyse.

   Sectie 2.2.4 beschrijft hoe er in deze stukken van WAQPRO met `lgrid` wordt omgegaan.

3. **Ontwerp: overzicht `mnmaxk` en corresponderende `fullbox`-arrays**

   Van veel variabelen in de berekening bestaan reeds twee varianten: een `mnmaxk`-array en een `fullbox`-array, waarvan de laatste vaak alleen maar bestaat wanneer TRIWAQ wordt gebruikt. Hierin wordt zo veel mogelijk eenheid gebracht.

   In Sectie 2.2.5 wordt een compleet overzicht van alle `mnmaxk`- en `fullbox`-arrays gemaakt (huidige datamodel, gecodeerd in `awasin.i`), en wordt beschreven welke `fullbox`-arrays in WAQUA zullen worden gebruikt.

4. **Ontwerp: welke vereenvoudigingen uitvoeren**

   In eerste instantie zal de verwijdering van `lgrid` uit de WAQUA-routines recht-toe-recht-aan worden uitgevoerd, met zo min mogelijk veranderingen aan de code. Daarna zullen er verschillende vereenvoudigingen worden doorgevoerd. In het bijzonder zullen meer regels code gedeeld kunnen worden door TRIWAQ en WAQUA.

   In Sectie 2.2.6 wordt bepaald welke vereenvoudigingen zullen worden doorgevoerd.

5. **Ontwerp: welke uitbreidingen aan TRIWAQ uitvoeren**

   Zodra er in WAQUA niet meer via `lgrid` wordt geadresseerd kunnen diverse stukken code ook voor TRIWAQ geschikt worden gemaakt. In een aantal gevallen kan hiermee met weinig werk nieuwe functionaliteit voor TRIWAQ beschikbaar worden gesteld, eventueel met de beperking dat ze alleen in 2D-berekeningen (1 laag) mag worden gebruikt.

   In Sectie 2.2.7 wordt uitgewerkt welke extra functionaliteit voor TRIWAQ zal worden gerealiseerd.

6. **Initialiseren alle benodigde arrays**

   De eerste stap van de daadwerkelijke programmeerwerkzaamheden betreft het correct aanmaken en initialiseren van de nieuwe data-structuur. Dit gaat vooral om `fullbox`-arrays die ook in WAQUA nodig zijn. Hiervoor moeten vooral subroutines `wasggd`, `wasfgd` en `wastgd` worden aangepast.

7. **`lgrid` verwijderen uit rekenroutines**

   Vervolgens moet `lgrid` uit de rekenroutines worden verwijderd. Hiervoor is een incrementele aanpak bedacht waarmee steeds een routine apart kan worden aangepast en

getest. Deze aanpak bestaat uit het (vooralsnog) handhaven van alle `mnmaxk`-arrays, voorafgaand aan de call van een rekenroutine converteren van zijn argumenten van `mnmaxk`- naar `fullbox`-formaat (`was1fr`), en na afloop terugconverteren van de resultaten naar de `mnmaxk`-arrays (`wasf1r`). De `mnmaxk`-arrays blijven zo steeds de actuele informatie bevatten.

De aanpassingen van de rekenroutines zelf zijn redelijk recht-toe-recht-aan. Een aandachtspunt betreft de communicaties die voor parallel rekenen en domein decompositie worden gedaan. Hierin worden per routine gelijk de goede index-sets ingevuld. Deze zullen in eerste instantie echter nog niet zijn gedefiniëerd, zodat er tussentijds alleen met sequentiële runs kan worden getest.

Er wordt middels de testbank gekeken of de aanpassingen correct zijn gemaakt. Hierbij worden geen verschillen in de simulatieresultaten verwacht.

8. **Koproutines aanpassen**

   Alle in punt 7 geïntroduceerde conversies (`was1fr`, `wasf1r`) worden weggehaald. Op de weinige plaatsen waar er nog met `mnmaxk`-arrays wordt gewerkt worden juist nieuwe conversies toegevoegd.

   Wanneer alle koproutines zijn aangepast, wordt middels regressietesten gekeken of de aanpassingen correct zijn gemaakt. Hierbij wordt nog steeds alleen met sequentiële runs gewerkt.

9. **Aanpassen initialisaties voor parallel rekenen**

   Vervolgens zullen de initialisaties van de communicatiebibliotheek COCLIB worden aangepakt. In simulaties met WAQUA zullen hier in plaats van index-sets voor `mnmaxk`-arrays nu index-sets voor `fullbox`-arrays worden gedefiniëerd. Dit betreft vooral aanpassing van het commentaar in subroutines `wasixs`, `wasifc`, `wasico` e.d.. Voor het overige is het onderscheid tussen de namen die in WAQUA en TRIWAQ nodig zijn momenteel via een variabele "`namhix`" geparametriseerd. Deze functionaliteit voor het onderscheiden van WAQUA en TRIWAQ wordt overbodig en zal ten behoeve van de leesbaarheid worden verwijderd uit de programmatuur.

   Na deze aanpassingen zullen de testen uit de testbank met parallel rekenen en domein decompositie worden uitgevoerd. Wederom worden er geen verschillen in de resultaten verwacht.

10. **Doorvoeren van vereenvoudigingen**

    De in Sectie 2.2.6 besproken vereenvoudigingen van de programmatuur zullen nu worden doorgevoerd en getest.

11. **WAQUA-functionaliteit beschikbaar maken in TRIWAQ-berekeningen**

    De speciale functies van WAQUA die gemakkelijk naar TRIWAQ kunnen worden omgezet zijn beschreven in Sectie 2.2.7. In deze stap worden de hiervoor benodigde werkzaamheden uitgevoerd. Deze omvatten beperkte testen van de nieuwe functionaliteit.

12. **Eindtesten met definitieve versie, rapportage en testverslag**

    Tenslotte worden er met de uiteindelijke versie van de programmatuur definitieve testen uitgevoerd. Onder andere wordt hierbij ook op de performance gelet. Deze testen worden in Sectie 2.2.8 verder uitgewerkt. De resultaten van deze testen worden in de rapportage en het testverslag verwerkt.

2.2.3 Aanpassingen in WAQGEN

WAQGEN is een bibliotheek van routines van WAQUA die in meerdere systemen worden gebruikt. Veel van de routines van WAQGEN worden zowel in WAQPRE als WAQPRO gebruikt, maar ook zijn er routines die in WAQPRO en COEXEC worden gebruikt, en routines die in allerlei andere programma's zoals SIMPAR, WAQPAN, WAQVIEW, OBS2SDS, SDS2MAT e.d. worden gebruikt. De laatste twee genoemde programma's horen niet bij SIMONA, maar worden onderhouden in het KALMINA-project.

Omdat de routines van WAQGEN in meerdere programma's worden gebruikt kunnen ze niet zomaar worden aangepast. Toch moet er met de routines die nu `lgrid` gebruiken iets worden gedaan. Hiervoor is een complete inventarisatie gemaakt van het gebruik van routines van WAQGEN in SIMONA, WAQVIEW en de KALMINA-programmatuur. Daaruit volgt dat de routines van WAQGEN op de volgende manier kunnen worden aangepast.

- `wagadb` en `wagcsb` vullen administratie-arrays m.b.t. barriers en worden alleen in WAQPRO gebruikt. `Lgrid` kan worden verwijderd uit deze routines.

- `wagadw` is een triviale routine die administratie-array `lgrovl` voor overlaten vult voor WAQPRE en WAQPRO. Hier zijn twee kopieën van gemaakt: subroutine `wapadw` voor WAQPRE werkt met `lgrid`, subroutine `wagadw` voor WAQPRO gebruikt geen `lgrid`.

- Subroutines `wagmt2`, `wagov1`, `wagov4` en `wagovp` betreffen het matchen van roosters voor domein decompositie met horizontale verfijning. Deze routines worden in WAQPRO en COEXEC gebruikt. Ze kunnen het beste op `mnmaxk` blijven werken (zie paragraaf 2.2.4).

- Subroutine `wagkni` voert de k-Nikuradse berekening uit. Deze routine wordt in WAQPRE, WAQPRO en WAQVIEW gebruikt. Het is een complexe berekening, waarvoor het naast elkaar onderhouden van meerdere implementaties duidelijk niet is gewenst. Verder is er de nodige rekentijd mee gemoeid. Daarom is het gewenst dat er in WAQPRO geen conversies tussen `mnmaxk` en `fullbox` nodig zijn. De routine moet dus naar `fullbox` worden omgezet.

  Een verfijning van de berekening die in dit project is onderkend betreft het gebruik van arrays `xzeta` en `yzeta`. Deze worden gebruikt om roosterafstanden te berekenen, terwijl die afstanden in WAQPRO ook al in arrays `guv` en `gvu` beschikbaar zijn. In overleg is besloten om deze arrays in `wagkni` te gaan gebruiken, ondanks dat ze in WAQPRE en WAQVIEW zullen moeten worden aangemaakt. Bij de implementatie bleek dit echter op problemen te stuiten: `guv` en `gvu` worden in WAQPRO berekend door het middelen

(de-staggeren) van `guu` en `gvv`, dit kan in WAQPRE en WAQVIEW moeilijk worden nagedaan, en dit leidt tot verschillen in (verslechtering van ?) de rekenresultaten. Het kan worden overwogen om `guv` en `gvu` in WAQPRO uit de coordinaten te gaan berekenen in plaats van te de-staggeren.

- De routines `wagcof` en `wagcot` betreffen de berekening van time-histories. Ze worden in WAQPRE gebruikt voor het initialiseren van de histories. Hierbij worden veld-arrays `sep`, `up`, `hu`, e.d. gebruikt.

  Van deze routines bestaat reeds een `fullbox`-variant, onder de namen `trscof` en `trscot`. Die routines van TRIWAQ kunnen echter niet zomaar geschikt worden gemaakt voor WAQUA. Bijvoorbeeld omdat ze op arrays voor de laagposities zijn gebaseerd die in WAQUA niet bestaan.

  We kiezen er vooralsnog voor om de routines `wagcof` en `wagcot` naar `fullbox` om te zetten. In WAQPRE worden de benodigde veld-arrays dan naar `fullbox` werkarrays gekopiëerd. Dit betreft 9 arrays voor `wagcof` en 12 arrays voor `wagcot`.

- Subroutines `wagcva` en `wagwen` behoren tot de berekening van het overlatenmodel, welke in WAQPRE en in WAQPRO wordt uitgevoerd. Hiervoor wordt dezelfde aanpak gevolgd als voor `wagcof` en `wagcot`.

  Er is ook overwogen om de berekeningen voor het initiële tijdstip te verhuizen naar de initialisaties in WAQPRO. Dat heeft echter als nadeel dat de waardes niet op de initiële SDS-file worden gezet. Dat is lastig voor parallel rekenen (partitioneren/collecteren) en voor de implementatie van de restart-functionaliteit.

Om ook gemakkelijk in andere programma's dan WAQPRO heen en weer te kunnen converteren tussen `mnmaxk` en `fullbox` zullen de subroutines `was1fr` en `wasf1r` worden verhuisd naar WAQGEN en worden hernoemd naar `wag1fr` en `wagf1r`.

2.2.4 Aanpassing van afzonderlijke functionaliteit

De berekeningen van TRIWAQ die op `mnmaxk`- in plaats van `fullbox`-arrays zijn gebaseerd zijn als volgt:

- de k-Nikuradse ruwheidsberekening (onderdeel van WAQGEN);
- het zogenaamde "matchen van roosters" voor domein decompositie met horizontale verfijning;
- harmonische analyse;
- space varying wind and pressure (SVWP).

De k-Nikuradse berekening is al in de vorige paragraaf besproken. Deze wordt zowel voor WAQUA als voor TRIWAQ naar `fullbox` omgezet.

Bij de SVWP-berekening wordt ervoor gekozen om deze op `mnmaxk`-arrays te blijven laten werken. Dat is voordelig omdat er anders een dertiental nieuwe `fullbox`-arrays moet worden geïntroduceerd. Verder moet dan ook de aansturing van het SIMONA interpolatietool worden aangepast. Verder gebruikt het SVWP-gedeelte geen arrays van de hydrodynamica die hiervoor naar het `mnmaxk`-formaat moeten worden geconverteerd. Met het aanpassen wordt daarom geen vereenvoudiging van het beheer en onderhoud bereikt.

Een aanpassing die wel aan de SVWP-berekening zal worden gemaakt betreft de conversie van de resultaten (arrays `pres`, `windu` en `windv`, zie Tabel 2.3) naar het `fullbox`-formaat. Deze conversie wordt nu in de koproutines `wasspu/v` gedaan wanneer er met TRIWAQ gerekend wordt. Omdat de conversie nu ook voor WAQUA nodig wordt is het elegant wanneer de berekening in het SVWP-gedeelte opgenomen wordt (subroutine `wasvwp`). Hiervoor worden drie `fullbox` werkarrays in semi-permanente arrays omgezet.

Ook de andere berekeningen zullen we grotendeels op `mnmaxk` laten staan. Voor het matchen is dit omdat er hier een aantal arrays wordt gebruikt die direct van de SDS-file afkomstig zijn (`x/ydep`, `khu/v`). Verder is het matchen in het verleden juist aangepast om het geheugengebruik van COEXEC te verminderen. Het gebruik van `lgrid` zit hier niet echt in de weg en WAQUA en TRIWAQ zijn hier al volledig op elkaar afgestemd.

Voor de harmonische analyse is de motivatie dat WAQUA en TRIWAQ al goed op elkaar zijn afgestemd en dat er in dit gedeelte loops `nm=1,mnmaxk` worden uitgevoerd. Wanneer deze berekeningen naar `fullbox` worden omgezet dan moeten alle loops worden aangepast zodanig dat er voor droge punten geen rekenwerk wordt gedaan.

In geval van de harmonische analyse kiezen we ervoor om de berekening van het rechter lid wèl, en de analyse zelf niet op `fullbox` te doen. Alleen array `rhstid` en de berekeningen in subroutines `wasuwl` en `washrh` worden dan aangepast. In eerstgenoemde subroutine wordt nu namelijk geen loop over `mnmaxk` maar over `mmax` en `nmax` uitgevoerd, en deze wordt sterk verbeterd door `nmloop` te gebruiken. De tweede routine wordt aangeroepen vanuit routines die veel met het FLOW-gedeelte van WAQPRO te maken hebben waardoor toevoegen van arrays zoals `irogeo` of `nmloop` aan de header niet bezwaarlijk is. Tenslotte bespaart deze keuze het heen en weer converteren van de arrays die voor de berekening van het rechterlid worden gebruikt.

Er bestaat ook een berekening die in een WAQUA-simulatie wordt uitgevoerd op `fullbox`-arrays: de user-transportroutine `wasust`. Momenteel worden voorafgaand aan deze routine de `mnmaxk`-arrays van WAQUA naar `fullbox`-formaat geconverteerd. Deze conversie komt te vervallen. Een verwarrende situatie die hierbij optreedt is dat er twee arrays in WAQUA met lengte `mnmaxk` en in TRIWAQ met afmetingen van `fullbox` zijn gedefiniëerd: `gsour` en `gsink` (`intgda`-entries 33,3 en 34,3). Dit verschil wordt opgeheven.

2.2.5 Overzicht van `mnmaxk`- en corresponderende `fullbox`-arrays

Omdat de berekeningen in WAQUA op `fullbox`-arrays moeten worden uitgevoerd moet worden uitgezocht welke `fullbox`-arrays moeten worden aangemaakt. Dat gaan we nu doen. We laten de arrays van WAQUA daarbij zoveel mogelijk samenvallen met de arrays die al in

TRIWAQ worden gebruikt.

Van veel variabelen in de berekening bestaan reeds twee varianten: een `mnmaxk`-array voor WAQUA en van/voor de SDS-file, en een `fullbox`-array dat vaak alleen voor TRIWAQ bestaat. In TRIWAQ worden de `fullbox`-arrays en `mnmaxk`-arrays vaak allebei gebruikt. De `mnmaxk`-arrays zijn ten eerste nodig voor het inlezen van en wegschrijven naar de SDS-file, en worden daarnaast ook in speciale berekeningen die op `mnmaxk` zijn gebaseerd gebruikt (bijv. SVWP). Hierbij wordt er overigens niet moeilijk gedaan over het geheugen dat dit kost; mogelijk kan er op de geheugenbehoefte van TRIWAQ worden bespaard door de `mnmaxk`-arrays steeds alleen tijdelijk te laten bestaan (in de initialisatiefase van WAQPRO, bij het wegschrijven naar SDS-file e.d.). Omgekeerd hoeven we ons bij het verwijderen van `lgrid` dus geen zorgen te maken over het dubbel in-core houden van alle WAQUA-arrays: WAQUA zal voor een schematisatie steeds minder geheugen gebruiken dan TRIWAQ nu al voor die schematisatie nodig heeft.

Het op deze manier besparen van geheugen in TRIWAQ lijkt moeilijk te zijn. Het probleem is dat de gebruikte `mnmaxk`-arrays zijn geassocieerd met arrays die op de SDS-file zijn opgeslagen. Het is niet duidelijk in hoeverre het deleten en opnieuw aanmaken van de in-core arrays tot problemen in het SIMONA memory management en SDS-gedeelte leidt.

Verder is het aantal arrays dat dubbel in-core wordt gehouden (zowel `mnmaxk` als `fullbox`-variant) relatief beperkt. Op basis van Tabellen 2.2-2.5 schatten we dat er in een typische WAQUA-run in totaal zo'n 60 `fullbox`-arrays zullen worden gebruikt: 20 arrays die ook op de SDS-file worden bewaard, 20 semi-permanente arrays die gedurende de gehele run bestaan, en 20 werkarrays die alleen in de flow-routines worden gebruikt (subroutines `wasspu/v`). Alleen van de eerste categorie wordt er ook een `mnmaxk`-variant gemaakt. Voor een model met vullingsgraad 0.5 wordt de berekening dan als volgt:

$$\text{huidige situatie:} \quad 60 * \texttt{mnmaxk} \tag{2.1}$$

$$\text{nieuw, efficient:} \quad 60 * \texttt{fullbox} + 5 * \texttt{mnmaxk} = 125 * \texttt{mnmaxk} \tag{2.2}$$

$$\text{nieuw, dubbel in-core:} \quad 60 * \texttt{fullbox} + 20 * \texttt{mnmaxk} = 140 * \texttt{mnmaxk} \tag{2.3}$$

Het verwijderen van `lgrid` kost voor dit model dus ruwweg een factor 2 extra geheugen binnen WAQPRO (1/vullingsgraad), en het dubbel in-core houden van arrays die op de SDS-file worden gezet kost hier nog 12% meer. Dit loopt op tot maximaal 33% voor een model met vullingsgraad 1. Deze hoeveelheid extra geheugenbeslag lijkt het niet waard om een risicovolle aanpassing van de SIMONA tools te maken. Wel kan er met een korte test worden onderzocht of het deleten van arrays die op de SDS-file zijn gezet problemen geeft.

Voor bijna alle `mnmaxk`-arrays die worden gebruikt in de berekeningen bestaat reeds een `fullbox`-equivalent, zie Tabellen 2.2 t/m 2.5. Hierbij zal in WAQUA het array van TRIWAQ worden gebruikt. De arrays waarvoor er nog geen `fullbox`-equivalent bestaat zijn:

- `xzeta`, `yzeta`

  Deze arrays worden alleen in de k-Nikuradse berekening gebruikt (subroutine `wagkni`), zie paragraaf 2.2.3. Ze worden gebruikt voor het berekenen van roosterafstanden.

mnmaxk-array		fullbox-array	
naam	intgda-entry	naam	intgda-entry
Roostercoordinaten			
xdep	204,1 (3) (S)	xdepf	263,1
ydep	204,1 (4) (S)	ydepf	264,1
xzeta	204,1 (5) (S)	-	
yzeta	204,1 (6) (S)	-	
guu (geu)	204,1 (1) (S)	guuf	251,1
gvv (gkv)	204,1 (2) (S)	gvvf	252,1
guv (gev)	25,1 (1)	guv	253,1
gvu (gku)	25,1 (2)	gvu	254,1
gsqslg	25,1 (3)	gsqs	255,1
gsqsi	25,1 (4)	-	
gsqdi	25,1 (5)	-	
Kenmerk- en administratiearrays			
kcslg	262,2	kcs	262,2
khu	3,1 (1) (S)	khuf	259,1
khv	3,1 (2) (S)	khvf	260,1
-		kfu	256,1
-		kfv	257,1
-		kfs	258,1
lgrovl	23,1	lgrovf	265,1
lgubar	34,2	-	
lgvbar	35,2	-	
-		kfbu	296,2
-		kfbv	294,2
Waterstanden en stroomsnelheden			
sep, sepold	201,2 (S), 30,2	sepf, seh	(252,256),2
up	202,2 (S)	upf, uh	(253,257),2
vp	203,2 (S)	vpf, vh	(254,258),2
w	233,2 (S)	wf	255,2
wphys	234,2 (S)	wphysf	werk array
qx	31,2	qxk	259,2
qy	40,2	qyk	260,2
-		qzk	261,2
-		umean	278,2
-		vmean	279,2
tetau	werk array	tetauf	werk array
tetav	werk array	tetavf	werk array

Tabel 2.2: mnmaxk-arrays en de overeenkomstige fullbox-arrays in uitgangsversie van WAQPRO (wordt vervolgd in Tabellen 2.3 t/m 2.5). De toevoeging (S) geeft aan dat het om arrays van/voor de SDS-file gaat.

mnmaxk-array		fullbox-array	
naam	intgda-entry	naam	intgda-entry
Dieptes, laaginterfaces en laagdiktes			
h (dp)	201,1 (S)	hf	261,1
dps	10,2 (S)	hdry	274,2
zks	7,1 (1) (S)	zksp, zksh	263,2, 266,2
zku	7,1 (2) (S)	zkup, zkuh	264,2, 267,2
zkv	7,1 (3) (S)	zkvp, zkvh	265,2, 268,2
hu	381,2 (S)	hkup, hkuh	werk arrays
hv	382,2 (S)	hkvp, hkvh	werk arrays
Bodemruwheid, viscositeit			
czu	204,2 (1) (S)	czuf	270,2
czv	204,2 (2) (S)	czvf	271,2
cmanu	214,2 (1) (S)	roughu	272,2
cmanv	214,2 (2) (S)	roughv	273,2
cmanue	214,2 (3) (S)	rougue (cmanue)	389,2
cmanve	214,2 (4) (S)	rougve (cmanve)	390,2
cczu, cczv	werk arrays	-	
metrou	388,2 (S)	metrou	388,2
vicow	235,2 (S)	vicowf	269,2
vicowg	- (S)	vicowg (vicwgf)	295,2
-		rich	275,2
-		ustbu	276,2
-		ustbv	277,2
-		z0wl	291,2
-		cfrowl	292,2
Wind en Pressure en Coriolis			
ffzeta	211,1 (S)	ffzetf	293,2
pres	313,2	presf	werk array
windu	309,2	winduf	werk array
windv	310,2	windvf	werk array
Arrays van het SVWP-gedeelte			
cdv1	212,1 (S)	-	
cdv2	213,1 (S)	-	
wstr	214,1 (S)	-	
cd	werk array	-	
cdu	werk array	-	
cdv	werk array	-	
windsx	werk array	-	
windsy	werk array	-	

Tabel 2.3: Vervolg van Tabel 2.2: mnmaxk-arrays en de overeenkomstige fullbox-arrays in de uitgangsversie van WAQPRO.

mnmaxk-array		fullbox-array	
naam	intgda-entry	naam	intgda-entry
Arrays van het SVWP-gedeelte (vervolg)			
wndu1	301,2	-	
wndu2	302,2	-	
wndv1	303,2	-	
wndv2	304,2	-	
wnspu1	305,2	-	
wnspu2	306,2	-	
wnspv1	307,2	-	
wnspv2	308,2	-	
pres1	311,2	-	
pres2	312,2	-	
wnduwl	werk array	-	
wndvwl	werk array	-	
Transport en turbulentie			
difco	206,1 (S)	difcof	262,1
difcw	217,3 (S)	difcwf	252,3
rp	5,3 (S)	rpf	251,3
ener	219,3 (S)	turbf, turbh	(257,258),3 (1)
diss	220,3 (S)	turbf, turbh	(257,258),3 (2)
rho	werk array	rho	werk array
gsink	34,3	gsink	34,3
gsour	33,3	gsour	33,3
User-transportroutine			
solusr	13,3 (S)	solusf	werk-array
spainp	340,3 (S)	spainf	werk-array
...		...	
Euleriaanse tijdsintegralen			
sepnow	227,2 (S)	-	
depint	228,2 (S)	-	
upint	229,2 (S)	upintf	284,2
vpint	230,2 (S)	vpintf	285,2
disunt	231,2 (S)	disunf	288,2
disvnt	232,2 (S)	disvnf	289,2
wphint	319,2 (S)	wphinf	287,2
wint	320,2 (S)	wintf	286,2
zkint	321,2 (S)	zkintf	290,2
zk	322,2 (S)	-	

Tabel 2.4: Vervolg van Tabel 2.2: mnmaxk-arrays en de overeenkomstige fullbox-arrays in de uitgangsversie van WAQPRO.

`mnmaxk`-array		`fullbox`-array	
naam	intgda-entry	naam	intgda-entry
Lagrangiaanse tijdsintegralen			
-		`fx`	351,2
-		`fy`	352,2
-		`fz`	353,2
-		`kpos`	354,2
-		`mpos`	355,2
-		`npos`	356,2
-		`uintp`	357,2
-		`vintp`	358,2
-		`wintp`	359,2
`xdlagr`	391,2 (S)	`xdlagf`	360,2
`ydlagr`	392,2 (S)	`ydlagf`	361,2
`zdlagr`	393,2 (S)	`zdlagf`	362,2
Harmonische getijdeanalyse			
`asplt`	- (S)	-	
`azero`	- (S)	-	
`ampl`	- (S)	-	
`phase`	- (S)	-	
`rhstid`	457,2	-	
Speciale uitvoer			
`valmax`	436,2 (S)	-	
`iclswl`	437,2	-	

Tabel 2.5: Vervolg van Tabel 2.2: `mnmaxk`-arrays en de overeenkomstige `fullbox`-arrays in de uitgangsversie van WAQPRO.

Van deze arrays zullen `fullbox`-equivalenten worden gemaakt waarvan de `infor`-indices worden opgeslagen in de `intgda`-entries (268,1) en (269,1).

- `gsqdi`, `gsqsi`

  Deze arrays bevatten de inverses van de oppervlaktes van roostercellen (bijv. `1./gsqs`). Ze zijn geïntroduceerd bij de parallellisatie van WAQUA, waarbij het performanceverlies van de daarvoor benodigde herstructurering zo veel mogelijk moest worden gecompenseerd.

  Van deze arrays zullen `fullbox`-equivalenten worden gemaakt waarvan de `infor`-indices worden opgeslagen in de `intgda`-entries (266,1) en (267,1). `Intgda`-entry (25,1) wordt hierdoor vrijgemaakt. De arrays zouden ook in TRIWAQ kunnen worden gebruikt, of ten koste van wat performance kunnen worden verwijderd.

  Overigens is het ongewenst dat er voor de arrays met roosterafstanden `guu`, `guv`, `gvu` en

`gvv` in WAQUA ook de namen `geu`, `gev`, `gku`, `gkv` worden gehanteerd. Laatstgenoemde namen zijn bij de parallellisatie van WAQUA geïntroduceerd, onder het mom van het beter leesbaar maken van de formules waarin deze variabelen worden gebruikt. De letters van de nieuwe namen maken onderscheid in welke richting er gemeten wordt (`k`=ξ, `e`=η), en in welke punten de afstand berekend is (u- en v-punten). Deze namen zouden in de hele programmatuur moeten worden ingevoerd, of overal zouden de oude namen moeten worden gebruikt. De keuze is ook van belang voor OMS.

In de bespreking van het detail-ontwerp is vastgesteld dat de namen `gku` en `gev` op zich meer zeggen dan `gvu` en `guv`, dat een gedeelte van de programmeurs aan laatstgenoemde namen is gewend, en dat de oude namen `guu` en `gvv` op de SDS-file staan, in veel gerelateerde programma's en in Delft3D-FLOW worden gebruikt. Daarom is invoering van de nieuwe namen vooralsnog niet gewenst. Ten behoeve van de uniformiteit zullen ze worden vervangen, en zullen overal de oorspronkelijke namen worden gebruikt.

- `lgubar`, `lgvbar`

  Dit betreft kenmerkarrays van WAQUA voor barriers. In barrierpunten kan aan dit array worden gezien dat er een barrier staat, wat het volgnummer is, en of de stroming over de barrier op ieder tijdstip sub- of superkritisch is.

  Deze arrays zullen alleen in `fullbox`-formaat nodig zijn. Ze worden direct in dit formaat aangemaakt via aanpassing van subroutines `wagadb` en `wagcsb` (zie paragraaf 2.2.3). De definitie van de bestaande arrays zal hiervoor worden aangepast.

- `cdv1`, `cdv2`, `wstr`, e.d.

  Omdat de SVWP-berekening niet wordt aangepast zijn deze arrays nooit nodig in `fullbox`-formaat.

  Arrays `presf`, `winduf` en `windvf` worden semi-permanente arrays. De overeenkomstige arrays `pres`, `windu` en `windv` kunnen juist via werk-arrays worden geïmplementeerd. Hiervan zullen de `intgda`-entries worden hergebruikt.

- `sepnow`, `depint`, `zk`

  De arrays `sepnow` en `depint` worden alleen in de Euleriaanse tijdsintegratie in WAQUA gebruikt, array `zk` wordt alleen in TRIWAQ gebruikt. `Sepnow` bevat een kopie van array `sep` en hoeft daarom niet in `fullbox`-formaat beschikbaar te zijn. Op dezelfde manier is `zk` een kopie van FLOW-array `zksp`. Array `depint` bevat de integraal van de totale waterdiepte. Dit array zou kunnen worden vervangen door het TRIWAQ-equivalent `zkint`, waarin de dikte per laag wordt geïntegreerd (verdere afstemming 2D en 3D berekeningen). Vooralsnog wordt er echter een `fullbox`-versie van `depint` geintroduceerd. De `infor`-index hiervan wordt opgeslagen in `intgda`-entry (323,2).

- `rhstid`

  Dit array is van de harmonische analyse van het getij. Omdat die berekening grotendeels op `mnmaxk` blijft werken wordt hier slechts weinig aangepast. Merk op dat van de

vier SDS-arrays van de harmonische analyse geen `infor`-pointers worden opgeslagen in `intgda`. Dat is omdat er in het analyse-gedeelte een andere (mooiere) programmeerstijl wordt toegepast waarin karakteristieke namen worden gebruikt.

Array `rhstid` wordt omgezet naar `fullbox`. Binnen het analysegedeelte wordt een lokaal werkarray aangemaakt met `mnmaxk`-formaat. `Intgda`-positie (457,2) wordt voor het `fullbox`-array hergebruikt.

- `iclswl`, `valmax`

  Dit zijn de arrays van de berekening van maximale waardes gedurende een simulatie en van de bepaling van incrementele uitvoer op basis van een klasse-verdeling. Deze functionaliteit is momenteel alleen voor WAQUA geïmplementeerd maar zal ook in TRIWAQ beschikbaar worden gemaakt.

  Een moeilijkheid bij de berekening van maximale waardes in geval KMAX>1 is dat de indeling van het array `valmax` lastig te beschrijven is. Belangrijker is echter dat er ook snelheden op het moment van de hoogste waterstand worden bewaard, wat complexere aanpassingen vergt en mogelijk minder zinvol is wanneer KMAX>1. Daarom wordt deze berekening in TRIWAQ vooralsnog alleen uitgevoerd wanneer KMAX=1.

  Entry (436,2) bleek dubbel te worden gebruikt, zowel voor array `grefdd` als voor `valmax`. Laatstgenoemde array is naar fullbox geconverteerd en de `infor`-index hiervan wordt nu opgeslagen in entry (438,2). Het array `iclswl` wordt ook naar `fullbox` geconverteerd. Van deze arrays zijn geen `mnmaxk`-versies nodig gedurende de berekeningen.

Behalve `mnmaxk`-arrays waarmee geen enkel `fullbox`-array overeenkomt, bestaan er ook enkele `mnmaxk`-arrays waar twee `fullbox`-arrays mee overeenkomen. Het gaat om de volgende arrays:

- `sep`, `sepold`, `hu`, `hv`, `up` en `vp`

  Met deze arrays komen steeds twee `fullbox`-arrays overeen, waarvan de ene “leeft” op de hele tijdstappen (achtervoegsel `p`), en de andere op de halve tijdstappen (`h`).

  Dit verschil hangt samen met een verschil in de filosofie van WAQUA en TRIWAQ in de manier waarop er met halve tijdstippen wordt omgegaan. TRIWAQ gebruikt aparte arrays voor “halve” en “hele” tijdstippen, WAQUA onderscheidt daarentegen de “oude” en “nieuwe” arrays. De oude arrays zijn in WAQUA slechts af en toe nodig en worden dan via werkarrays geïmplementeerd. Hierdoor wordt er een beperkte hoeveelheid geheugen bespaard.

  WAQUA zal op dezelfde manier met deze variabelen omgaan als TRIWAQ nu doet. Dat betekent dat bij elke routine-call goed in de gaten moet worden gehouden welke van de twee beschikbare arrays bedoeld wordt. Daarvoor kan steeds worden gekeken naar de overeenkomstige call die in TRIWAQ wordt gedaan.

  Overigens wordt in Delft3D-FLOW de huidige filosofie van WAQUA gebruikt. Het is niet haalbaar of gewenst om hier nu in heel WAQUA/TRIWAQ op over te gaan.

2.2.6 Vereenvoudigingen in de code

In eerste instantie worden de `fullbox`-berekeningen geïntroduceerd met minimale aanpassing van de routines. Nadat de aanpassingen zijn doorgevoerd zullen daarom nog verschillende verbeteringen mogelijk zijn. In de koproutines, waarin arrays worden geactiveerd en `ibuffr`-pointers worden opgehaald, zullen veel regels tweemaal voorkomen: eenmaal voor TRIWAQ en eenmaal voor WAQUA. Deze regels worden samengenomen.

Ook in de rekenroutines zullen vereenvoudigingen mogelijk zijn, in de vorm van loopstructuren die gestandaardiseerd kunnen worden. In de rekenroutines `waseva` en `waslvl` (dynamische barriersturing) kunnen ook vereenvoudigingen worden verkregen omdat daarin berekeningen zowel op `fullbox`- als op `mnmaxk`-arrays voorkomen. Overigens blijven hier verschillen tussen WAQUA en TRIWAQ bestaan vanwege verschillende in de arrays die moeten worden gebruikt (HU versus ZKU, etc.).

Er worden ook vergelijkbare vereenvoudigingen in de initialisaties van de communicatiebibliotheek verwacht. Veel initialisaties zullen niet meer nodig zijn omdat veel index sets, interfaces, interpolatiemethoden en coëfficiëntensets niet meer worden gebruikt. Welke dat zijn zal worden bepaald wanneer de vereenvoudingen uitgevoerd worden, omdat op dat moment veel efficiënter kan worden gezocht in de code welke elementen niet meer worden gebruikt.

Ten slotte kunnen vereenvoudigingen worden bereikt door TRIWAQ en WAQUA-routines samen te voegen. Hierin zal terughoudend te werk worden gegaan. Alleen de volgende routines zullen worden samengevoegd:

- `wasguv` en `trsguv`: Berekening van afstanden en oppervlakten in het rooster.
- `waskcs` en `trskcs`: Berekening van het kenmerkarray `kcs`.

Een overzicht van andere overeenkomstige routines van WAQUA en TRIWAQ wordt weergegeven in Tabel 2.6.

2.2.7 Uitbreidingen voor TRIWAQ

In WAQUA-berekeningen zijn bepaalde faciliteiten beschikbaar die niet in TRIWAQ beschikbaar zijn. De routines die deze faciliteiten bieden kunnen echter eenvoudig gebruikt worden voor TRIWAQ (zeker voor TRIWAQ met één laag) wanneer WAQUA op `fullbox`-arrays is gebaseerd. De volgende berekeningen zullen beschikbaar gemaakt worden voor TRIWAQ met één laag:

- QH- en QAD-randen;
- berekening van de maximale waterstand;
- berekening van incrementele uitvoer (klasseverdeling) van de waterstand.

Voor het toestaan van QAD-randen is een voordeel dat het gebruik hiervan momenteel in TRIWAQ niet verboden is. WAQPRE controleert niet of de invoer wel correct is, want in

`wasdry/trsdry`	droogval en onderlopen
`wastrd/trstrd`	oplossen van tridiagonale stelsels met BJ-TWGE
`wasvlu/wasvlv/trsczw`	berekening van de bodemruwheid
`trsdfv`, `trsdfz`, `trsdhv`	gelijktrekken van verschillende schotjesvariabelen
`wasfit/trsfit`	Euleriaanse tijdsintegralen
`wapdps/trshdr`	Berekening van de bodemligging in waterstandspunten
`waspcf/trspcf`	Afdrukken naar report-file
`waspci/trspci`	Afdrukken van een integer array
`waspcr/trspcr`	Afdrukken van een real array
`waspct/trspct`	Afdrukken naar report file
`wasphf/trsphf`	Afdrukken naar report file
`waspht/trspht`	Afdrukken naar report file
...	...

Tabel 2.6: Routines van WAQUA en TRIWAQ die een overeenkomstige functie hebben en die misschien samengevoegd kunnen worden.

TRIWAQ wordt er een onzinberekening uitgevoerd. Twee problemen met de huidige implementatie zijn dat de TRIWAQ-arrays (snelheden, bodemwrijving) niet naar `mnmaxk`-formaat worden geconverteerd, en dat de arrays `hu` en `hv` in TRIWAQ überhaupt niet worden gebruikt. Het eerste probleem wordt door het verwijderen van `lgrid` uit WAQUA automatisch opgelost. Voor het tweede hoeft alleen de berekening van de totale doorstroomhoogte in TRIWAQ-berekeningen te worden toegevoegd. Dit lijkt zowieso een nuttige uitbreiding te zijn.

De berekening van de maximale waterstand is direct voor TRIWAQ met 1 laag te gebruiken. Voor verdere generalisatie moeten er extra array-dimensies en loops voor de vertikaal worden toegevoegd. Verder verandert hiermee de definitie van het array dat op de SDS-file wordt gezet. Dit zal nog niet worden uitgevoerd omdat er momenteel bij MX.*Systems* aan deze berekening wordt gewerkt (instelbaar maken van de optie via de simulatie-invoerfile).

De berekening van de incrementele uitvoer betreft momenteel alleen de waterstand en is dus direct voor zowel WAQUA als TRIWAQ geschikt. Bij V$\mathcal{O}$Rtech Computing bestaat wel een testversie waarin ook uitvoer voor de snelheden kan worden verkregen. Deze hebben we in opdracht van Meander Advies gemaakt. Deze uitbreidingen zullen niet in het huidige project in WAQUA worden geïntegreerd.

2.2.8 Testen

Omdat het verwijderen van `lgrid` uit WAQUA een omvangrijke ingreep is zal ze uitvoerig worden getest met veel modellen met allerlei verschillende opties aan- en uitgezet. Bij deze testen zullen compileroptimalisaties worden uitgezet. De verwachting is dat er dan in de meeste modellen helemaal geen verschillen in de numerieke resultaten ontstaan. Wanneer er toch verschillen optreden dan zal de oorzaak hiervan worden getraceerd.

De performance zal worden onderzocht op het Linux platform met de GNU en Intel compilers en op de Teras van SARA (omdat deze een heel ander cachemechanisme gebruikt en omdat V$\mathcal{O}$Rtech Computing hier gemakkelijk toegang tot heeft). Dit wordt gedaan zodra ook activiteit 3-7 is uitgevoerd (massacorrectiestap). Het netto effect van die en de huidige activiteit zal zijn dat het performanceverlies in praktisch alle gevallen minder dan 5% bedraagt.

2.3 Activiteit 2-3: Invoeren van arrays DPU, DPV, KCU en KCV

2.3.1 Achtergrond van de aanpassingen

Een aantal problemen met de huidige implementatie van het algoritme van droogvallen en onderlopen hangt samen met gegevens die niet worden opgeslagen maar steeds worden herberekend. Dit beperkt de uitbreidbaarheid van de programmatuur, omdat wijzigingen in de algoritmiek hierdoor ingrijpen op veel verschillende plaatsen in de programmatuur. Met name de koppeling van waterbeweging met stoftransport en morfologie heeft hieronder te lijden.

Dit is onder andere onderkend in het OMS project bij het maken van keuzes voor de hydrodynamische module van OMS [13]. Met name worden er keuzes gemaakt voor de definitie van kenmerkarrays (KCU, KCV, KFU, e.d., voor de permanente (C) en tijdsafhankelijke (F) aspecten van de geometrie) en voor het beschrijven van waterstanden en dieptes (DPU, ZKU, HKU e.d.).

In de huidige activiteit wordt een aantal van deze voorstellen geïmplementeerd, namelijk het invoeren van DPU en DPV, en het vervangen van KHU, KHV door KCU, KFU, KCV en KFV. Dit zijn vooral de voorstellen die van direct belang zijn voor andere activiteiten uit dit project (2-5, 3-8). Verder wordt met de aanpassingen een verschil in administratie tussen WAQUA en TRIWAQ weggewerkt (KHU versus KFU), wat de onderhoudbaarheid en uitbreidbaarheid van WAQUA/TRIWAQ ten goede komt.

Belangrijke verschillen tussen WAQUA en TRIWAQ die bij de uitwerking van activiteit 2-4 naar voren zijn gekomen zijn het alleen bestaan van de arrays HU en HV in WAQUA en van ZKS, ZKU en ZKV in TRIWAQ. Deze verschillen staan in een aantal gevallen het uitwisselen van code in de weg. Bijvoorbeeld werken het droogvalalgoritme in WAQUA met controles `hu(nm).le.htrsh` en in TRIWAQ met `zkuh(nm,0)-zkuh(nm,kmax).le.htrsh`. Voor de uitbreidingen van paragraaf 2.2.7 wordt HU ook in TRIWAQ toegevoegd, naast het bestaande array HKU, misschien niet permanent maar in ieder geval als werkarray. Het toevoegen van de ZK-arrays in WAQUA is lastiger; hiervoor moeten de waterstanden steeds worden gekopieerd, wat mogelijk te veel performance kost. Het gelijktrekken van WAQUA en TRIWAQ op deze punten wordt daarom niet binnen het huidige project gedaan.

Activiteit 2-3 moet worden uitgevoerd voorafgaand aan 2-2 (toevoegen tijdsniveau aan schotjesarrays) omdat het veel mechanischer werk is en hiermee alvast wat inzicht wordt vergaard. Verder wordt WAQUA in deze activiteit afgestemd op TRIWAQ waardoor de vervolgactiviteit 2-2 ook gemakkelijker wordt.

2.3.2 Invoering van arrays DPU en DPV

De arrays DPU en DPV worden toegevoegd aan de SDS-file. Dit is nodig voor activiteit 2-5, waarbij de waardes hierin op een andere manier berekend gaan worden. Verder is dit gewenst voor het kunnen plotten van de dieptes in verschillende lokaties en voor de koppeling met andere modules.

Op dit moment zijn de dieptes op de SDS-file niet netjes georganiseerd. De dieptes in de hoekpunten van roostercellen worden opgeslagen in array MESH_H, in WAQUA-runs worden de dieptes in de cellen opgeslagen in DPS_FLOW, en in TRIWAQ-runs worden alle dieptes opgeslagen via de LAYER_INTERFACES, arrays ZKS, ZKU en ZKV. Hierin wordt bij de invoering van de nieuwe arrays gelijk wat lijn gebracht.

De dieptes worden opgeslagen in arrays MESH_DEPTH_DPD (vervangt MESH_H), MESH_DEPTH_DPU, MESH_DEPTH_DPV en MESH_DEPTH_DPS (vervangt DPS_FLOW). Op deze manier worden alle dieptegegevens in dezelfde compound-array samengevoegd en wordt de naamgeving geüniformeerd. Vooralsnog laten we de arrays MESH_H en DPS_FLOW bestaan. Dit is vooral van belang voor compatibiliteit. Niet alle afgeleide programmatuur moet dan gelijk worden aangepast, en er ontstaan minder beperkingen aan de versies van WAQPRO en afgeleide programmatuur die samen kunnen worden gebruikt. Na een of twee jaar zouden MESH_H en DPS_FLOW echter wel moeten worden verwijderd van de SDS-file. Verder kan er op termijn naar een manier worden gezocht om de diepte-arrays te laten samenvallen met de onderste laag van de ZK-arrays. Iets dergelijks is ook besproken bij het datamodel voor OMS [13, par. 5.4].

Alternatieven hiervoor zijn om de dieptes op te slaan onder de compounds COEFF_FLOW of COEFF_GENERAL. In MESH worden er naast de coördinaten van het kromlijnige rooster namelijk verder eigenlijk alleen maar integer administratie-arrays bewaard, zoals de (m, n)-lokaties van dampunten, overlaten, barriers e.d. De afmetingen (hoogtes, breedtes e.d.) van overlaten en barriers staan in compound COEFF_FLOW. Een extra argument voor het onderbrengen van de dieptearrays in COEFF_FLOW is dat de berekeningen die ermee worden uitgevoerd sterk aan het droogvalalgoritme zijn gerelateerd, terwijl dit algoritme typisch tot FLOW gerekend wordt. Aan de andere kant zijn de dieptes ook nadrukkelijk een invoerparameter voor het transportmodel. Vanuit dat licht bezien zouden ze ook goed onder compound COEFF_GENERAL kunnen worden gezet.

De arrays kunnen onder een gemeenschappelijk tussenniveau in de LDS van WAQUA worden geplakt ("MESH_DEPTH"). Een nadeel daarvan is dat er meer veranderingen in de structuur worden gemaakt. Verder worden de verschillende niveaus van compound-arrays door de invoering van karakteristieke namen steeds minder relevant. Daarom wordt ervoor gekozen om alle vier de arrays op hetzelfde niveau te plakken:

```
MESH
  |
  +------- KMESH1
             |
             +----- JMESHA        general part
             |
```

```
+----- JMESHB        specific part
         |
         +---- KMESHA    = IDIMEN
         :
         :
         +---- KMESHH    = MESH_H
         :
         :
         +---- KMESHS    = DEPTH_DPS
         |
         +---- KMESHT    = DEPTH_DPD (kopie van MESH_H)
         |
         +---- KMESHU    = DEPTH_DPU
         |
         +---- KMESHV    = DEPTH_DPV
```

De invoering van de arrays in WAQPRE vereist de volgende aanpassingen:

`wapg02` - de descriptor van de MESH-array wordt aangepast,

`wapg03` - er worden extra subarrays van MESH aangemaakt, de call van `wap0mi` (vullen van de diepte-arrays) wordt aangepast,

`wapf13` - wordt aangepast: kopiëren van DPS in plaats van aanroepen `wapdps`,

`wapdps` - wordt verwijderd, deze berekening wordt opgenomen in `wap0mi`,

`wap0mi` - deze routine wordt sterk uitgebreid, en gaat alle dieptecijfers invullen in plaats van alleen DPD (H),

`wapwrt` - de aanpassingen in de data-structuur worden verwerkt.

Andere aanpassingen die worden gemaakt betreffen het gebruik van de nieuwe arrays. Overal waar nu de-staggering van DPD wordt gebruikt zullen de arrays DPU en DPV worden ingevoerd. Dit betreft de rekenroutines van WAQPRO, initialisaties van ZKU/V in TRIWAQ, en een enkele plek in WAQPRE.

2.3.3 Vervanging van KHU, KHV door KCU, KFU, KCV en KFV

Het is mogelijk om KCU, KFU, KCV en KFV (en de bestaande arrays KCS, KFS) ook toe te voegen aan de SDS-file, maar dit is niet noodzakelijk. Het lijkt ons in ieder geval niet verstandig om de arrays KHU en KHV die overbodig worden gemaakt van de SDS-file te verwijderen. Deze arrays worden op ontelbaar veel plaatsen in gerelateerde programmatuur gebruikt, zodat die programmatuur dan onvermijdelijk ook direct moet worden aangepast.

Wanneer KCU e.d. al in WAQPRE worden ingevoerd dan is het logisch om ze gelijk binnen WAQPRE op zo veel mogelijk plaatsen te gaan gebruiken. Maar verder wordt WAQPRE in

dit project niet zo veel aangepast. Er zijn ook geen directe toepassingen bekend waarvoor de nieuwe arrays op de SDS-file moeten worden gezet. Daarom worden de nieuwe arrays vooralsnog alleen binnen het programma WAQPRO geïntroduceerd. In de rekenroutines zal er consequent met KFU, KCU en KFV, KCV worden gewerkt, op de SDS-file worden KHU en KHV gezet.

De arrays zullen in de initialisatiefase van WAQPRO worden aangemaakt en geïnitialiseerd. Dit gebeurt in subroutine `wasggd` (initialisaties van het general gedeelte), omdat KFS, KFU en KFV daar nu ook al worden aangemaakt voor TRIWAQ runs. De arrays KFU en KFV van TRIWAQ kunnen eenvoudig ook voor WAQUA worden gebruikt, de arrays KCU en KCV zijn nieuw en worden op een nog te kiezen plek in `intgda` toegevoegd.

De definitie van KCU en KCV zegt dat de waarde nul is in punten waar er een permanent schotje staat. Deze punten worden herkend aan de conditie KHU/V=0. In alle andere punten wordt de waarde KCU/V=1 gebruikt. In het OMS keuzesverhaal [13] zijn verder de codes 2 en 3 gereserveerd voor open randen en punten bij subdomeininterfaces. Het is nog niet duidelijk of deze codes nuttig en nodig zijn.

Vervolgens worden de nieuwe arrays op alle relevante plekken in WAQPRO ingevoerd. Logische condities op basis van `khu` en `khv` worden als volgt aangepast:

```
khu(nm).gt.0   -->   kfu(nm).eq.1
khu(nm).ge.0   -->   wordt niet gebruikt
khu(nm).eq.0   -->   kcu(nm).eq.0
khu(nm).ne.0   -->   kcu(nm).ge.1
khu(nm).le.0   -->   kfu(nm).eq.0
khu(nm).lt.0   -->   kcu(nm).ge.1 .and. kfu(nm).eq.0
```

Het aanpassen van `khu` en `khv` wordt conform de volgende regels omgezet:

```
khu(nm) =  kmax          -->  kfu(nm) = 1
khu(nm) = -kmax          -->  kfu(nm) = 0
khu(nm) = -khu(nm)       -->  kfu(nm) = 0 of 1, afhankelijk van context
khu(nm) = -abs(khu(nm))  -->  kfu(nm) = 0
```

Tenslotte worden alle subroutineheaders en de corresponderende calls en koproutines aangepast. In principe betekent dit het vervangen van `khu` en `khv` door `kcu`, `kfu`, `kcv` en `kfv`, maar in praktijk zullen er veel routines zijn waar `kcu` en `kcv` niet nodig zijn. `khu` en `khv` komen nu op zo'n 580 plekken voor in 30 rekenroutines en 10 koproutines van WAQPRO.

De arrays `khu` en `khv` worden ook op een paar plekken in TRIWAQ gebruikt (subroutines `trsdfv`, `trsdfz`, `trskhi`). De arrays `kfu` en `kfv` gelden momenteel als werkarrays die handig voor de berekeningen zijn, `khu` en `khv` zijn de "echte" arrays die ook op de SDS-file worden gezet. Dit wordt aangepast. In het vervolg zijn `kfu` en `kfv` binnen WAQPRO de echte arrays waaraan de nat/droogstatus wordt vastgelegd, en worden `khu` en `khv` alleen voor het lezen van en schrijven naar de SDS-file gebruikt.

Bij het schrijven naar de SDS-file worden de volgende conversies gebruikt:

```
kcu(nm).eq.0                       -->  khu(nm) = 0
kcu(nm).gt.0 .and. kfu(nm).eq.0    -->  khu(nm) = -kmax
kcu(nm).gt.0 .and. kfu(nm).eq.1    -->  khu(nm) =  kmax
```

2.3.4 Testen

De aanpassingen aan de dieptearrays en schotjesarrays zullen apart van elkaar worden getest. Bij de vervanging van `khu` en `khv` verwachten we totaal geen verschillen in de rekenresultaten.

In geval van de dieptearrays zijn er wel verschillen mogelijk. Omdat de de-staggering (`0.5*(dp(nm)+dp(ndm))`) door array-lookup (`dpu(nm)`) wordt vervangen zijn er verschillen in de volgorde van evaluatie mogelijk, met andere afronding als gevolg. Wanneer dit optreedt dan zullen de verschillen tussen de oude en nieuwe versies worden vergeleken met de verschillen die met optie QUANTF_RANDOM worden behaald. Ook kunnen de compiler-optimalisaties tijdelijk worden uitgezet om te kijken of er dan nog steeds verschillen zijn.

2.3.5 Documentatie

De aanpassingen van deze activiteit hebben geen gevolgen voor de gebruikersdocumentatie van WAQUA. De aanpassingen aan de lokale data-structuur (dieptearrays DPS-DPD) worden in de LDS-beschrijving verwerkt. Het gebruik van `kfu` in plaats van `khu` wordt in de data-analyse van WAQUA en TRIWAQ verwerkt.

2.4 Activiteit 2-2: Toevoegen tijdsniveau aan kenmerkarrays

2.4.1 Achtergrond

Activiteit 2-2 uit het werkplan bij de offerteaanvraag voor het huidige project wordt in twee stukken gesplitst: enerzijds het op een zodanige manier invoeren van aparte kenmerkarrays voor halve en hele tijdstippen dat het numerieke algoritme ongewijzigd blijft, anderzijds het benutten van de nieuwe arrays voor verbetering van de discretisatie van de advectieve termen. Deze laatste activiteit is in dit detailontwerp “3-8” genoemd en wordt in fase 3 uitgevoerd.

2.4.2 Uitwerking van de activiteit

De bestaande arrays KFU en KFV worden hernoemd naar KFUP en KFVP, en hiernaast worden nieuwe arrays KFUH en KFVH geïntroduceerd. Deze veranderingen worden overal doorgevoerd (1100 plekken binnen 55 rekenroutines en 18 koproutines). Steeds moet er worden gekeken of de waardes van halve of hele tijdstippen moeten worden gebruikt. Om dit correct uit te kunnen voeren is een data-analyse uitgevoerd voor de schotjesarrays. Hierbij zijn de volgende zaken opgevallen:

- Alleen in de routines `wasuxc`, `wassuc` en `trssuw` zijn beide tijdsniveaus van de schotjesarrays nodig.

  In `wasuxc` (eerste helft van de tijdstap) wordt het array `khvh` gevuld met de waarden van `khvp`, en worden de waarden waar nodig veranderd op de fysische randen. De routine `wassuc` doet het zelfde voor de schotjes in de u-richting in het array `khuh`.

  In TRIWAQ worden beide nieuwe schotjesarrays pas gevuld in `trssuw`. Er bestaat wat dit betreft dus een verschil tussen TRIWAQ en WAQUA. Dit verschil zal in activiteit 3-2 van dit project ongedaan worden gemaakt (zie paragraaf 3.5).

- In de transportroutines `wasdfc` en `trsdif` worden de nieuwe waarden van de schotjesarrays in de v-richting gebruikt. Het lijkt hier echter logischer om de oude waarden te gebruiken. Dit is afhankelijk van hoe er precies met de debieten wordt omgegaan. Conceptueel kunnen hiervoor namelijk ook waardes op hele (`qxkp`) en halve tijdstippen (`qxkh`) worden onderkend. Ook dit zal pas in fase 3 van dit project worden aangepakt.

- In subroutine `wascht` worden in TRIWAQ bij de berekening van transport-histories de oude waardes voor de stroomsnelheden gebruikt (array `up`). In WAQUA worden de nieuwe waarden uit array `uh` gebruikt. Dit lijkt een programmeerfout in TRIWAQ te zijn. Deze wordt vooralsnog niet gecorrigeerd, vooral omdat er anders verschillen in de resultaten worden geïntroduceerd. Dit probleem is bij de SIMONA-beheerder MX.*Systems* aangemeld.

2.4.3 Testen

Het testen van deze uitbreiding is relatief eenvoudig. De twee versies van een array zijn in geval van droogvallen steeds verschillend van elkaar, dus als de verkeerde wordt gebruikt dan is dit in de simulatieresultaten te zien.

2.4.4 Documentatie

De veranderingen hebben geen effect op de beschrijving van de numerieke methodes in de technische documentatie. Wel zal de data-analyse van WAQUA en TRIWAQ in [7] worden aangepast.

2.5 Activiteit 2-1: Gebruik DUPWND

2.5.1 Achtergrond en uitwerking van de nieuwe methode

De keuze tussen de “gemiddelde” en “upwind” aanpak voor de berekening van doorstroomhoogtes `hu` en `hv` wordt in WAQUA en TRIWAQ gestuurd met de invoerparameter DUPWND, zie [9, vgl. (3.8)]. Het gebruik van een drempelwaarde kan ertoe leiden dat een kleine verandering in de waterstand tot een groot verschil in de doorstroomhoogte leidt. Dit verhoogt de

Algorithm 1 - Afhandeling van keywords UPWIND_ZETA en DUPWND in WAQPRE

if (UPWIND_ZETA opgegeven door gebruiker) **then**
  lees string, converteer naar hoofdletters
  if (string is YES of Y) **then**
    `dupwnd = +99999.`
  elseif (string is NO of N) **then**
    `dupwnd = -99999.`
  else
    foutmelding en stop
  end if
  if (DUPWND opgegeven door gebruiker) **then**
    waarschuwing dat DUPWND wordt genegeerd
  end if
elseif (DUPWND opgegeven door gebruiker) **then**
  waarschuwing dat DUPWND wordt afgeraden
  lees waarde van `dupwnd` in
else
  zet default NO: `dupwnd = -99999.`
end if

gevoeligheid van het model voor kleine verstoringen. Daarom is het gewenst een vlag in te voeren waarmee de upwind-aanpak overal wordt aangezet of uitgezet.

Deze aanpassing betreft een kleine uitbreidng van de preprocessor WAQPRE. In de referentietabel wordt onder het keyword FLOW - PROBLEM - DRYING het nieuwe keyword UPWIND_ZETA toegevoegd. Dit is een optioneel keyword met type=3 (string). Toegestane waardes zijn YES en NO (niet-case sensitive), alleen de eerste letter hiervan is verplicht, en de default is NO. Deze default is consistent met de huidige parameter DUPWND met default -99999.

Het nieuwe keyword wordt afgehandeld in subroutine `wap073` van WAQPRE, waar momenteel ook de waarde van `dupwnd` wordt bepaald. Dit gebeurt met de logica van Algoritme 1. We kiezen ervoor om de variabele `dupwnd` met dezelfde betekenis op dezelfde plek op de SDS-file te laten staan. Aan WAQPRO hoeft dan niets te worden aangepast.

2.5.2 Testen

Om de aanpassingen te testen hoeven geen complete simulaties te worden uitgevoerd. Aan WAQPRO verandert immers niets. Wel moeten alle invoercombinaties worden aangeboden aan WAQPRE:

- UPWIND_ZETA en DUPWND beiden niet opgegeven: `dupwnd=-99999.`
- Alleen DUPWND opgegeven: waarde wordt verwerkt, warning.

- Alleen UPWIND_ZETA wordt opgegeven: test of alle mogelijke strings op de goede manier worden afgehandeld (goed: YES, Y, yes, no, N, fout: ye2, noy).
- Beide keywords opgegeven: waarschuwing over DUPWND.

2.5.3 Documentatie

De nieuwe optie moet worden beschreven in de Users Guide WAQUA. Zowel in het general gedeelte (paragraaf 3.6.2) als in de invoerbeschrijving van WAQPRE (paragraaf 2.8.1.4).

2.6 Activiteit 2-5: Opgeven van dieptecijfers in waterstandspunten

2.6.1 Achtergrond van de nieuwe optie

In [9, par. 3-2] wordt beargumenteerd dat het voordelig is om de dieptecijfers in waterstandspunten (middens van roostercellen) te kunnen opgeven in plaats van in de dieptepunten van het rooster (hoekpunten van cellen) zoals nu gebeurt. Geultjes en dijken kunnen dan met één roostercel worden weergegeven in plaats van dat er twee rijen van cellen voor nodig zijn. Met deze optie is reeds ervaring opgedaan in het kader van inundatieberekeningen met Delft-FLS.

In tegenstelling tot andere activiteiten van fase 2 van dit project betreft deze activiteit niet het herstructureren van de bestaande programmatuur, maar betreft het het toevoegen van nieuwe functionaliteit. De activiteit is toch aan fase 2 toegevoegd omdat ze geen gevolgen heeft voor de resultaten van simulaties met de reeds bestaande functionaliteit.

2.6.2 Wiskundige methoden en vergelijkingen

De nieuwe optie betreft het inlezen van dieptecijfers in de preprocessor WAQPRE en het vertalen hiervan naar de dieptes in verschillende soorten roosterpunten. In de huidige versie van de programmatuur wordt array DPD door de gebruiker gespecificeerd en worden DPS, DPU en DPV hiervan afgeleid, in de nieuwe versie wordt DPS opgegeven en worden DPU, DPV en DPD hieruit afgeleid.

Voor het berekenen van dieptes in snelheidspunten bij gebruik van de nieuwe optie wordt in Delft-FLS een "MIN"-algoritme gebruikt:

$$\text{METH\_DPUV} = \text{'MIN\_DPS'} : d^{(u)}_{m,n} = \min(d^{(s)}_{m,n}, d^{(s)}_{m+1,n}) \tag{2.4}$$

Hierin staat $d^{(u)}_{m,n}$ voor `dpu(m,n)`, $d^{(s)}$ voor `dps` etc. Merk verder op de dieptes in arrays `dps`, `dpu` etc. *positief naar beneden* gemeten worden, zodat MIN op de kleinste diepte slaat. De analoge formule voor de definitie van $d^{(v)}$ wordt via spiegeling van m en n bepaald.

De MIN-aanpak kan goed worden gevisualiseerd door de bodem te tekenen als een verzameling horizontale "tegels" per roostercel. Deze aanpak heeft als voordeel boven lineair

interpoleren dat het afstromen van een plaat in de richting van de bodemgradiënt plaatsvindt. Verder is het met deze aanpak mogelijk om een methode te ontwikkelen die postieve waterdieptes oplevert (cf. Delft-FLS). Aan de andere kant suggereert de tegelbenadering een lagere orde nauwkeurigheid dan wanneer lineaire interpolatie wordt gebruikt. Daarom worden er verschillende mogelijkheden naast elkaar geïmplementeerd.

$$\text{METH\_DPUV = 'MEAN\_DPS'} : d^{(u)}_{m,n} = (d^{(s)}_{m,n} + d^{(s)}_{m+1,n})/2 \tag{2.5}$$

$$\text{METH\_DPUV = 'MEAN\_DPD'} : d^{(u)}_{m,n} = (d^{(d)}_{m,n} + d^{(d)}_{m,n+1})/2 \tag{2.6}$$

Wanneer de dieptes in waterstandspunten worden opgegeven dan lijken de waardes in dieptepunten (array DPD) nauwelijks van belang. In aansluiting met de tegelaanpak van de MIN-optie voor DPU en DPV kan hier goed het minimum van de omliggende vier cellen worden gebruikt. Wanneer voor de berekening van DPU en DPV echter lineaire interpolatie wordt gebruikt dan ligt het gemiddelde van de omliggende vier cellen meer voor de hand.

$$\text{METH\_DPUV = 'MIN\_DPS'} : d^{(d)}_{m,n} = \min(d^{(s)}_{m,n}, d^{(s)}_{m+1,n}, d^{(s)}_{m,n+1}, d^{(s)}_{m+1,n+1}) \tag{2.7}$$

$$\text{METH\_DPUV = 'MEAN\_DPS'} : d^{(d)}_{m,n} = (d^{(s)}_{m,n} + d^{(s)}_{m+1,n} + d^{(s)}_{m,n+1} + d^{(s)}_{m+1,n+1})/4 \tag{2.8}$$

2.6.3 Uitbreiding van de invoerfile

Met betrekking tot de benodigde uitbreiding van de structuur van simulatie-invoerfiles zijn er verschillende mogelijkheden:

1. De nieuwe opties kunnen onder het hoofdstukje DEPTH_CONTROL worden toegevoegd. Ze passen hier op zich wel goed thuis, maar staat dan wel ver verwijderd van de opgegeven waardes zelf.

2. De nieuwe opties kunnen bij het opgeven van de dieptecijfers onder MESH - BATHYMETRY worden toegevoegd. Hiermee worden de waardes en de betekenis ervan dicht bijelkaar gezet. Aan de andere kant zijn de bestaande keywords onder BATHYMETRY heel weinig op de betekenis gericht (DEPTAG, DEPMULTIPL, etc.).

3. De nieuwe opties kunnen bij het specificeren van het droogvalalgoritme onder FLOW - PROBLEM - DRYING worden toegevoegd. Ze lijken immers veel op de bestaande optie IDRYFLAG. Maar verder horen de dieptes niet specifiek bij het FLOW-gedeelte. Zelfs optie IDRYFLAG hoort hier eigenlijk niet langer thuis omdat ze alleen effect heeft op de permanente geometrie.

Wat hierin te zien is is dat het droogvallen zowel als onderdeel van de hydrodynamica als als onderdeel van de meer algemene geometriebeschrijving kan worden beschouwd. Dat is een terugkerend probleem dat zich bijvoorbeeld ook uit in de plek waar de dieptecijfers worden opgeslagen, en hoe er met schotjes en de posities van laaginterfaces wordt omgegaan.

We kiezen vooralsnog voor alternatief 2. De ideale structuur van de invoerfile wordt dan:

MESH
BATHYMETRY
GLOBAL
| DPD_GIVEN
<
| DPS_GIVEN
METH_DPS = 'MIN_DPUV' | 'MEAN_DPD' | 'MAX_DPUV' | 'MAX_DPD'
METH_DPUV = 'MIN_DPS' | 'MEAN_DPS' | 'MEAN_DPD'

FLOW
PROBLEM
DRYING
CHECK_WL = 'YES' | 'NO'
IDRYFLAG = 0 | 1 | 2 | 3 (voor compatibiliteit)
TRESH_UV_FLOODING = *[val]*
TRESH_WL_FLOODING = *[val]*
DEPCRIT = *[val]* (voor compatibiliteit)
UPWIND_ZETA = 'YES' | 'NO'
DUPWND = *[val]* (voor compatibiliteit)

Hierbij worden de volgende opmerkingen gemaakt:

- Met de keywords DPD_GIVEN en DPS_GIVEN wordt geselecteerd of de opgegeven waardes gelden voor dieptepunten of waterstandspunten. De eerste is de default.
- Keyword METH_DPS vervangt het huidige keyword IDRYFLAG. Het grote voordeel hiervan is dat er met strings wordt gewerkt in plaats van cryptische waardes, zie wens 2 in [9]. Verder is de optie verhuisd van het hoofdstuk FLOW naar de geometriebeschrijving in MESH. De waarde 'MAX_DPUV' is de default, corresponderend met IDRYFLAG=2.

  Wanneer dieptes in waterstandspunten worden opgegeven (DPS_GIVEN) dan hoort METH_DPS niet te worden gebruikt. Gebeurt dat toch dan wordt een waarschuwing gegeven en wordt de opgegeven waarde genegeerd.

  Een idee was om hier met zogenaamde XOR-keywords te werken. Dat zou als voordeel hebben dat de herkenning automatisch door de generieke preprocessor van SIMONA wordt gedaan. Dit is echter niet mogelijk in de structuur van de invoerfile (referentietabel). Verder kan er met XOR-keywords geen default-waarde worden gehanteerd.
- Bij keyword METH_DPS wordt onderscheid gemaakt tussen de MAX-opties van Rijkswaterstaat ('MAX_DPUV') en WL|Delft Hydraulics ('MAX_DPD'). De eerste verwijst naar de bestaande formule in WAQUA/TRIWAQ, de tweede naar de strengere formule van Delft3D. Deze levert grotere dieptes op, en wordt toegevoegd in WAQPRE.
- Keyword METH_DPU is een verfijning ten opzichte van de bestaande methodes voor berekening van DPU. De keywords verwijzen naar formules (2.4), (2.5) en (2.6). Wan-

neer DPD_GIVEN wordt gebruikt dan is 'MEAN_DPD' de default (de huidige methode voor berekening van dieptes), in geval van DPS_GIVEN is 'MIN_DPS' de default (tegelaanpak).

Merk op dat het ook wordt toegestaan om 'MIN_DPS' (formule (2.4)) te gebruiken in combinatie met de oude methode van opgeven van dieptecijfers in dieptepunten (DPD_GIVEN).

- Het keyword IDRYFLAG wordt in principe overbodig gemaakt door de invoering van METH_DPS. Alleen de huidige waarde 3 ("NO") hoort specifiek bij het algoritme in plaats van bij de geometrie. Hiervoor is het keyword CHECK_WL toegevoegd; hiermee wordt de extra droogvalcontrole in waterstandspunten aan- en uitgezet.

  De optie IDRYFLAG wordt vooralsnog gehandhaafd vanwege compatibiliteit. Gebruikers worden via een waarschuwing op de wijzigingen gewezen, bijvoorbeeld:

```
WARNING xxxx:

You are using input-option IDRYFLAG=1, which has become obsolete.
Please use the following settings instead:
    MESH  BATHYMETRY  GLOBAL : METH_DPS = 'MEAN_DPD'
and
    FLOW  PROBLEM     DRYING : CHECK_WL = 'YES'
```

  Wanneer opties METH_DPS en/of CHECK_WL worden gebruikt dan is het gebruik van IDRYFLAG niet toegestaan.

  Wanneer IDRYFLAG=3 wordt opgegeven door de gebruiker ("NO") dan wordt dit geinterpreteerd als CHECK_WL='NO' en METH_DPS=MEAN_DPD.

- Het keyword TRESH_WL_FLOODING wordt ingevoerd zodat experts met verfijningen van het droogvalalgoritme kunnen spelen. Hiermee kan de grenswaarde voor de droogvalcontrole in waterstandspunten apart worden ingesteld. Voor de duidelijkheid van de invoerfile wordt TRESH_UV_FLOODING ingevoerd als synoniem voor DEPCRIT.

  DEPCRIT geldt als default voor TRESH_UV en heeft zelf als default de waarde 0.3. De waarde die hiermee voor TRESH_UV wordt bepaald wordt gebruikt als default voor TRESH_WL. Wanneer TRESH_UV en DEPCRIT beiden worden gespecificeerd dan wordt er een waarschuwing gegeven dat de laatste geen invloed heeft. Wanneer CHECK_WL='NO' dan heeft TRESH_WL geen effect.

Van al deze opties zijn alleen de keywords DPS_GIVEN en DPD_GIVEN begroot in dit project. De overige verbeteringen worden als meerwerk uitgevoerd.

2.6.4 Afhandeling van de opties in WAQPRE

De nieuwe keywords m.b.t. dieptes in de MESH worden afgehandeld in WAQPRE bij het inlezen van de dieptecijfers in subroutine `wap0mi`. Deze subroutine wordt (logisch gezien?) in drie stukken verdeeld: het bepalen van opties, het lezen van het opgegeven array van waardes, en het vertalen hiervan naar de vier arrays DPS, DPU, DPV en DPD.

De opties worden opgeslagen in vier integer vlaggen in array MESH_IDIMEN:

```
idimen(34) = idploc = location where depth values are specified by the
                      user; 1 = wl-points, 4 = dp-points
idimen(35) = imtdps = method by which dps has been computed;
                      0 = given,    1 = min_dpd, 2 = mean_dpd,
                      3 = max_dpuv, 4 = max_dpd
idimen(36) = imtdpu = method by which dpu and dpv have been computed;
                      2 = mean_dpd, 5 = min_dps, 6 = mean_dps
idimen(37) = imtdpd = method by which dpd has been computed;
                      0 = given,    5 = min_dps, 6 = mean_dps
```

Verder worden de volgende waardes gewijzigd en toegevoegd in CONTROL_FLOW_ICONTB:

```
icontb( 1) = idryfl = selection flag for drying-flooding procedure
                      ...
                      4 = extra drying control at wl-location with
                          'minimum criterion' with local depths at
                          dp-location
                      5 = extra drying control at wl-location with
                          values DPS specified by user
icontb(11) = idrywl = selection flag for drying checks in waterlevel
                      points:
                      0 = no drying control at wl location
                      1 =    drying control at wl location using depth-
                          values from array DPS
```

De variabele `idryfl` wordt alleen gehandhaafd voor compatibiliteit, ze is niet langer nodig binnen WAQPRO. In een aantal combinaties van de nieuwe keywords is er geen echt geschikte waarde voor `idryfl` mogelijk. Dan wordt altijd waarde `3` gebruikt wanneer CHECK_WL= 'NO'. Anders wordt de waarde van METH_DPS vertaald volgens de bestaande codering met bovenstaande uitbreidingen.

Tenslotte worden de volgende entries van array CONTROL_FLOW_RCONTA aangepast:

```
rconta(11) = trshuv = marginal depth value in flooding checks at
                      velocity points; drying checks use htrshu =
                      trshuv/2.
rconta(14) = trshwl = marginal depth value in flooding checks at
                      waterlevel points; drying checks use htrshw =
                      trshwl/2.
```

Zodra de waardes van DPS, DPU, DPV en DPD berekend zijn en op de SDS-file zijn gezet is er verder geen verschil meer ten opzichte van de oude aanpak waarin dieptecijfers in de hoekpunten worden gespecificeerd. Aan het programma WAQPRO hoeven daarom geen aanpassingen te worden gemaakt.

2.6.5 Testen

De nieuwe methode zal worden getest met een testmodel van een droogvallende plaat met verschillende gradiënten in x- en y-richtingen (conform paragraaf 7.1 van [9]).

Verder zal er vooral worden getest of de opgegeven bodemligging en daarbij betrokken opties goed door WAQPRE worden verwerkt. Hierbij kan mogelijk ook een kanaal met constante bodemhelling worden gebruikt, waarbij er met verschillende opties mogelijk hetzelfde resultaat kan worden bereikt. Hiermee worden gelijk eventuele verschillen in de randafhandeling in kaart gebracht.

2.6.6 Documentatie

Voor deze optie worden aanpassingen gemaakt in de gebruikershandleiding van WAQUA. Dit betreft zowel het general gedeelte als de invoerbeschrijving van WAQPRE. Verder wordt de nieuwe methode in de technische documentatie verwerkt.

Hoofdstuk 3

Fase 3: verbetering van het algoritme

3.1 Overzicht

In fase 3 worden de verbeteringen aan de numerieke algoritmen van Tabel 2 uit het werkplan geïmplementeerd, hier herhaald als Tabel 3.1, en worden de numerieke aspecten van activiteit 2-2 doorgevoerd.

De volgorde die we hier kiezen is om eerst het droogvalalgoritme van TRIWAQ compleet uit te werken en pas daarna met WAQUA aan de gang te gaan. De activiteiten 3-8 (tijdsniveaus schotjes) en 3-7 (massacorrectiestap) vergemakkelijken het verdere werk en worden daarom als eerste uitgevoerd. Bij WAQUA geldt hetzelfde voor activiteit 3-2. Vervolgens wordt het feitelijke nieuwe droogvalalgoritme geïmplementeerd in stap 3-4. In geval van WAQUA wordt deze stap direct gecombineerd met stap 3-5.

Activiteit 3-3 wordt als laatste uitgevoerd. We hebben onze twijfels over de wenselijkheid van deze stap. Ze geeft misschien ongewenste interacties met de massacorrectiestap (complexe code) of met de benodigde communicaties voor domein decompositie en parallel rekenen (performance). Deze consequenties zijn pas goed te bepalen zodra de andere stappen zijn

Tabel 3.1: *Overzicht van de verschillende activiteiten in Fase 3.*

Nr.	**Wat?**	**Soort**	**Rapport**
3-1	*Begrenzing H in diffusie/anti-creep*	*Verbeteren algoritme*	*Hfd 6 - wens 6*
3-2	*Verwijderen 4 verschillen droogval WAQUA en TRIWAQ*	*Verbeteren algoritme*	*Hfd 6 - wens 9*
3-3	*Verwijderen 1 verschil droogval met Delft-3D-FLOW*	*Verbeteren algoritme*	*Hfd 6 - wens 10*
3-4	*Verbeteren droogvalalgoritme mbt dwarsrichting*	*Verbeteren algoritme*	*Hfd 6 - wens 11*
3-5	*Negatieve controle volumes in WAQUA*	*Verbeteren algoritme*	*Hfd 6 - wens 12*
3-6	*Aanpassing advectieve term*	*Verbeteren algoritme*	*Hfd 6 - wens 15*
3-7	*Massacorrectie alleen op subdomeinranden*	*Verbeteren algoritme*	*Hfd 6 - wens 16*

uitgevoerd.

De activiteiten 3-1 en 3-6 staan los van de aanpassingen aan het droogvalalgoritme en worden onafhankelijk van de rest gedaan.

3.2 Activiteit 3-8: Aanpassen tijdsniveaus schotjes in advectieve termen

De eerste aanpassing aan de numerieke formuleringen van WAQUA/TRIWAQ betreft het gebruik van schotjes op het oude tijdsniveau bij het discretiseren van de impulsvergelijking. Dit is een relatief kleine aanpassing in de rekenroutines `wassuc` en `trsumo`. Deze aanpassing is mogelijk gemaakt door het introduceren van aparte schotjesarrays voor halve en hele tijdstappen in activiteit 2-2.

Alleen de discretisatie van de impulsvergelijking die samen met de continuïteitsvergelijking wordt opgelost wordt aangepakt. Alleen hierin speelt namelijk het probleem van het aanpassen van de geometrie. Bijvoorbeeld wordt in subroutine `trssuw` van TRIWAQ eerst `trsdfv` aangeroepen (schotjes weghalen en bijplaatsen → aanpassen van de geometrie), en daarna `trsumo` aangeroepen waarin de impulsvergelijking wordt gediscretiseerd. Omdat de schotjes (nieuwe geometrie) hierin dan niet kloppen met de snelheden die in de discretisaties worden gebruikt (oude tijdsniveau) worden bepaalde randafhandelingen anders gedaan dan wordt beoogd.

3.2.1 Advectieve termen in subroutine `trsumo` (TRIWAQ)

We beschrijven de berekening op de halve tijdstippen in de u-richting. De schotjes op het oude tijdsniveau worden aangeduid met `kfup`, de beginschatting voor de schotjes op het nieuwe tijdsniveau met `kfuh`.

Er kunnen vier soorten roosterpunten worden onderscheiden:

1. punten die droog waren en zijn gebleven (`kfup` = `kfuh` = 0). In deze punten wordt geen impulsvergelijking gediscretiseerd en de invloed van deze punten op de omliggende punten is constant in de tijd.

2. punten die nat waren en droog zijn geworden (`kfup` = 1, `kfuh` = 0). Voor deze punten zelf is de geometrie niet van belang. In omliggende punten zal de snelheid `up` van deze punten in discretisaties worden gebruikt (nu gebeurt dit niet).

3. punten die droog waren en nat zijn geworden (`kfup` = 0, `kfuh` = 1). In deze punten moet een impulsvergelijking worden opgesteld. Hierin moeten zo mogelijk ook advectieve en visceuze termen worden meegenomen. Door de nieuwe administratie wordt de mogelijkheid geïntroduceerd om hiervoor speciale benaderingen te gaan gebruiken.

4. punten die nat waren en nat zijn gebleven (`kfup` = `kfuh` = 1). In deze punten wordt een impulsvergelijking opgesteld die afhankelijk is van de actuele geometrie. Daarbij

ligt het voor de hand om afgeleides op het oude tijdsniveau ook de oude geometrie te gebruiken. Bijvoorbeeld bevat de viscositeitsterm een bijdrage $\texttt{kfuh}_{m-1} \cdot (\texttt{up}_m - \texttt{up}_{m-1})$ welke tegengesteld werkt aan de snelheid $\texttt{up}_m$. Om te voorkomen dat het onderlopen van punt $m-1$ leidt tot een tijdelijk grote afremming in punt m kan hierin $\texttt{kfup}_{m-1}$ worden gebruikt. De term wordt dan uitgeschakeld wanneer het buurpunt op het oude tijdstip droog was. Hiermee wordt wel een beperkte asymmetrie geïntroduceerd: in het buurpunt wordt conform punt 3 hierboven wel viscositeit in rekening gebracht.

Bij het discretiseren van de impulsvergelijking worden de schotjes op twee manieren gebruikt:

1. om te bepalen voor welke punten er moet worden gediscretiseerd, en

2. om te bepalen hoe er moet worden gediscretiseerd.

Voor het eerste doel zijn steeds de nieuwe schotjes van belang. Dit betreft onder andere de schotjes die worden doorgegeven aan subroutine `trsbar`. Voor het tweede doel moet goed naar het tijdstip van de te discretiseren term worden gekeken. Dit is eenvoudig voor termen waarin alleen u-snelheden worden gebruikt (oude tijdsniveau), maar lastiger voor termen waarin ook v-snelheden worden gebruikt (y-advectie, viscositeit).

Als voorbeeld beschouwen we de advectieterm $\overline{v}^{xy} u_y$. De v-snelheid wordt op het nieuwe tijdsniveau genomen, u_y wordt expliciet geëvalueerd. In de discretisatie van deze term worden de schotjes van de vier omliggende v-punten gebruikt. Hiervoor zijn nu twee keuzes mogelijk: oude waardes `kfvp` of nieuwe waardes `kfvh`.

- als je `kfvp` gebruikt dan sta je y-advectie toe wanneer de vier v-punten nat waren. Als er een of meer daarvan droogvallen in de huidige stap dan wordt de y-advectie verminderd door afname van $\overline{v}^{xy}$ (variabele `vvv`).

- als je `kfvh` gebruikt dan wordt y-advectie uitgeschakeld wanneer een of meer v-punten droogvallen in de huidige stap.

Er zijn geen argumenten bekend waarom de ene variant numeriek gezien beter zou zijn dan de andere. De keuze is daarom op implementatie-technische gronden bepaald. Het is voordelig om de oude schotjes te gebruiken omdat de discretisaties dan buiten de iteratieloop kunnen worden bepaald. Droogvallen van v-punten in de huidige stap heeft dan geen consequenties voor de coëfficiënten van de impulsvergelijking.

Een vergelijkbare analyse is ook gemaakt voor de overige termen waarin de v-snelheden worden gebruikt. In alle gevallen lijkt het gebruik van de schotjes `kfvp` acceptabel te zijn.

Bij het opstellen van de discretisaties wordt geen onderscheid gemaakt tussen punten van categorieën 3 en 4. De advectieve en visceuze termen worden dus volledig toegepast in punten die net nat zijn geworden.

3.2.2 Advectieve termen in subroutine `wassuc` (WAQUA)

De aanpassingen van WAQUA worden op dezelfde principes gebaseerd. Ook worden de discretisaties van dezelfde termen aangepast: x-advectie, y-advectie, kromlijnigheidstermen, viscositeit (excl. kruistermen).

Het discretiseren van de gekoppelde impuls- en continuïteitsvergelijking wordt hier in meerdere subroutines gedaan: subroutine `wasrv1` voor randpunten, `wassbc` voor barrierpunten en `wassuc` voor interne u-punten.

Een extra stap in WAQUA betreft het tijdelijk zetten van schotjes in superkritische barrierpunten in de dwarsrichting. Hiermee worden de discretisaties van de advectieve termen gestuurd: de snelheden in superkritische barrierpunten kunnen veel afwijken van het omliggende gebied en kunnen daarom beter niet worden gebruikt in het bepalen van ruimtelijke afgeleiden. Deze stap moet werken op de schotjes `kfvp` op het oude tijdsniveau. De extra schotjes worden aan het einde van `wassuc` weer weggehaald.

Overigens is het niet zo elegant dat het schotjesarray wordt gebruikt voor het doorgeven van informatie over de status van barriers. Het zou misschien beter zijn wanneer hiervoor een apart maskerarray wordt geïntroduceerd.

3.2.3 Testen

Het is moeilijk om deze aanpassingen te testen omdat moeilijk te bepalen is wat "juist" is en wat niet. De methode die hierbij wordt gevolgd is om de testbank te draaien, het model te bepalen dat de grootste verschillen te zien geeft, waarna die verschillen in detail worden onderzocht. We verwachten vooral verschillen in lokale stroombeelden, bijvoorbeeld bij droogvallende en onderlopende platen.

3.2.4 Documentatie

De beschrijvingen van de numerieke formuleringen in de technische documentatie kunnen mogelijk op details worden aangepast. Verder moeten de aanpassingen worden verwerkt in de data-analyse van WAQUA/TRIWAQ.

3.3 Activiteit 3-7: Aanpassen massacorrectiestap

3.3.1 Achtergrond van de massacorrectiestap

In WAQUA en TRIWAQ wordt de continuïteitsvergelijking zodanig gediscretiseerd en opgelost dat de oplossing na iedere iteratie massabehoudend is. Hierdoor kan het iteratieproces op ieder moment worden afgebroken in plaats van dat er moet worden doorgeïtereerd totdat machinenauwkeurigheid is bereikt. Een veronderstelling die hieraan ten grondslag ligt is echter wel dat de tridiagonale stelsels voor de waterstanden exact worden opgelost. Bij gebruik van parallel rekenen is dit niet langer het geval. Wanneer er zogenaamde blok-Jacobi randvoorwaarden worden gebruikt dan worden de tridiagonale stelsels niet exact opgelost.

Blok-Jacobi randvoorwaarden komen voor wanneer een rij van het rekenrooster over drie of meer subdomeinen is verdeeld.

Bij de parallellisatie van WAQUA/TRIWAQ is een speciale correctiestap geïntroduceerd om toch massabehoud te kunnen garanderen in parallelle runs. Deze correctiestap wordt uitgevoerd nadat het iteratieproces voor de continuïteitsvergelijking beëindigd is. Ze bestaat uit het fixeren van de debieten die door het iteratieproces zijn bepaald, uitrekenen van de waterstanden die hieruit volgen, en uitrekenen van de stroomsnelheid die daarbij het gefixeerde debiet teruggeeft. In WAQUA wordt daarna nog een droogvalcontrole toegepast. Als er punten droogvallen door de correctiestap dan worden de debieten hierin op nul gezet, en wordt de correctiestap herhaald.

Uit diverse onderzoekjes is gebleken dat de tridiagonale stelsels voor de waterstanden ook in sequentiële runs niet exact worden opgelost. In een aantal situaties is de oplossing sterk gevoelig voor afrondfouten, zijn de stelsels slecht geconditioneerd [9]. Daarom wordt de correctiestap voor het hele rooster en ook in sequentiële runs uitgevoerd. Theoretisch gezien zou ze dan niet nodig zijn. In praktijk blijken er door de correctiestap wel degelijk aanpassingen aan de waterstanden te worden gemaakt.

Aan de andere kant blijkt de massacorrectiestap in bepaalde gevallen afrondfouten juist te versterken. Met name bij grote Courantgetallen heeft de correctie in het binnengebied een averechts effect. Verder kost ze performance: in veel gevallen naar schatting 5-10%, oplopend tot 18% in testen met het MHW Maasmodel op een Linux cluster [6]. Om deze twee nadelen weg te nemen of te verminderen wordt de massacorrectiestap aangepast.

3.3.2 Massacorrectiestap in TRIWAQ

De beoogde aanpassingen aan TRIWAQ worden schematisch weergegeven in Algoritme 2.

Er wordt vanuitgegaan dat er alleen over de debieten op subdomeininterfaces nog onduidelijkheid bestaat. De twee betrokken subdomeinen zijn van verschillende waardes uitgegaan. Na afloop van het iteratieproces berekent de eigenaar van de interface het "echte" debiet en wordt dit gecommuniceerd. Vervolgens worden de waterstanden aangepast in roostercellen die van deze debieten afhankelijk zijn. Dit zijn de eerste en laatste interne cellen per rij: `mfu` en `ml`. Hierna worden de laagposities herberekend in u-punten die van deze waterstanden afhangen. Tenslotte worden de u-snelheden in deze punten geschaald zodat de eerder bepaalde debieten worden bereikt.

Voor het loopen over de eerste en laatste interne cellen van een rij worden de volgende statements gebruikt:

```
do 200 irk = 1, norows
   ...
   if (lcoupl(1,irk) .and. mfu.le.ml) then
      ...
   endif
   if (lcoupl(2,irk) .and. mfu.le.ml) then
      ...
```

Algorithm 2 - Uitwerking van de massacorrectiestap op subdomeinranden in TRIWAQ (subroutines `trssuw` en `trsmss`)

iteratie q: berekening $\zeta^{[q]}$, $\mathtt{hu}^{[q]}$, $u^{[q]}$
na afloop iteratieproces: berekening definitieve debieten op halve tijdstap (`qxk`)
→ automatisch massabehoudend in binnengebied
if (`lmassc`) **then**
  → massacorrectie kan met hard-coded parameter `lmassc` worden gedeactiveerd
  berekening waterstanden ζ': alleen punten `mfu` en `ml`
  communicatie waterstanden ζ' voor stencil stenc_{max}
  aanpassing laagposities in u-punten op nieuwe tijdstip (`zkuh`): punten `mf`, `mfu`, `mld`, `ml`
  aanpassing snelheden u' op subdomeininterfaces: punten `mf`, `mfu`, `mld` en `ml`
else
  communicatie waterstanden ζ' voor stencil stenc_{max}
  berekening laagposities in u-punten op nieuwe tijdstip (`zkuh`)
  → deze berekening van `zkuh` (call `trslav`) lijkt overbodig ?!
end if

```
            endif
 200      continue
```

Het eerste gedeelte van de tests die hierin worden uitgevoerd bepaalt of er een subdomeingrens zit aan het begin (`lcoupl(1,irk)`) of einde (`lcoupl(2,irk)`) van een rij. Het tweede gedeelte is nodig voor samenvallende subdomein/fysische randen. Rijtjes van een subdomein waarvoor `mfu.gt.ml` bestaan alleen uit een randpunt van het globale domein en moeten dus worden overgeslagen in de correctiestap.

Merk op dat er in dit algoritme niet op droogvallen wordt gecontroleerd. Dit werd in TRIWAQ nooit gedaan en wordt in het huidige project uit WAQUA verwijderd (zie onder). In principe is het hierdoor mogelijk dat de laagdiktes in een snelheidspunt negatief worden zonder dat er een schotje staat, of dat negatieve volumina ontstaan. In praktijk valt dit mee, vooral omdat de correcties klein zijn ten opzichte van de droogvalgrenswaarde $\delta/2$.

3.3.3 Massacorrectiestap in WAQUA

In WAQUA wordt vrijwel hetzelfde algoritme geïmplementeerd als in Algoritme 2 voor TRIWAQ is voorgesteld. Dit betekent dat er wel meer wordt aangepast, omdat er in WAQUA momenteel in de massacorrectiestap op droogvallen wordt gecontroleerd en omdat hiervoor een extra iteratieproces wordt gebruikt.

De motivatie voor het verwijderen van de droogvalcontrole uit de massacorrectiestap is als volgt. Deze controle kost relatief veel communicatie in parallelle runs en bemoeilijkt de code. Er moet een apart iteratieproces worden geïntroduceerd waarin steeds wordt gekeken of er nog droogval heeft plaatsgevonden, waarna een nieuwe iteratie moet worden uitgevoerd.

Zowel voor het stopcriterium voor dit iteratieproces als voor de schotjesarrays moet er extra worden gecommuniceerd.

Problemen ten gevolge van het verwijderen van de droogvalcontrole worden voorkomen door een aanpassing die ten behoeve van domein decompositie is gemaakt. Bij gebruik van horizontale verfijning worden de waterstanden `seh`, doorstroomhoogtes `hu` en schotjes `khu` onafhankelijk van elkaar gecommuniceerd (van een fijn naar een grover buurdomein). Dat blijkt tot op heden de meest praktische manier van werken te zijn. De schotjes moeten worden gecommuniceerd in plaats van berekend door het grove domein vanwege de strikte eis dat er in het grove domein geen schotje staat als er in tenminste een van de overeenkomstige interfacepunten van het fijne domein geen schotje staat. De doorstroomhoogtes moeten worden gecommuniceerd in plaats van berekend door het grove domein omdat er vooral bij de overgang nat/droog anders geen goede overeenstemming tussen de waardes van beide domeinen te bereiken is. En de doorstroomhoogtes in de interfacepunten van het grove domein moeten tenslotte ergens op slaan wanneer er in deze punten geen schotje staat.

Om dit laatste te bereiken wordt `hu` overal op minimaal `0.499`*DEPCRIT gezet (in TRIWAQ wordt in subroutine `trslav` voor de totale diepte de minimumwaarde `0.002` gebruikt, zie verder de vierde alinea op pagina 5-5 van [9]). Hiermee worden de numerieke algoritmes van WAQUA en TRIWAQ potentiëel beïnvloed. Maar tot op heden kon de waarde van `hu` in alle droge punten sowieso niet als een zinvolle waarde worden gebruikt. Het effect van de aanpassing is voor domein decompositie dat het fijne domein in alle interfacepunten positieve waardes gebruikt, en dat middeling van deze waardes een zinvolle (positieve) doorstroomhoogte voor het grove domein geeft. Voor de aanpassing van de massacorrectiestap is het effect dat het array `hu` geen nullen of negatieve waardes bevat. Daardoor kan het geen kwaad wanneer er tijdelijk geen schotje staat in een punt waar dat op basis van de droogvalcontroles eigenlijk wel nodig zou zijn. Waar tot op heden geen rekening mee is gehouden is dat door de correctiestap in theorie ook waterstanden onder de bodem kunnen komen: negatieve volumina. Om te kijken of en hoe vaak dit voorkomt zal er gedurende de testen in dit project een waarschuwing in de code worden ingebouwd. Op basis hiervan zal worden bepaald of er aanvullende maatregelen nodig zijn.

3.3.4 Testen

Het is moeilijk om deze aanpassingen te testen. De methode die hierbij wordt gevolgd is om de testbank te draaien, het model te bepalen dat de grootste verschillen te zien geeft, waarna die verschillen in detail worden onderzocht.

Er zou een testmodel kunnen worden gemaakt waarin massabehoud een belangrijke rol speelt. Dit kan dan sequentiëel en parallel worden gedraaid met volledige massacorrectie, alleen op subdomeinranden, of zonder massacorrectie.

3.3.5 Documentatie

De beschrijving van de massacorrectiestap in de data-analyse moet worden aangepast. Verder is onduidelijk in hoeverre de correctie wordt beschreven in de technische documentatie, en

hoeveel beschrijving hier is gewenst.

3.4 Activiteit 3-4: Verbeteren algoritme TRIWAQ mbt dwarsrichting

Zodra de tijdsniveaus van schotjes veranderd zijn dan is het niet langer nodig om de impulsvergelijking opnieuw te discretiseren wanneer er droogvallen in de dwarsrichting plaatsvindt. Hiermee kunnen de binnen- en buiteniteratie in `trssuw` worden samengevoegd. Verder kunnen er bij droogvallen van waterstandspunten in een keer vier schotjes worden gezet. Al met al wordt hiermee een eenvoudiger en efficiënter droogvalalgoritme voor TRIWAQ gerealiseerd.

3.4.1 Alternatieven bij uitwerking van het nieuwe algoritme

De werking van het nieuwe algoritme is globaal al in [9, par.5.1] gepresenteerd.

Ons uitgangspunt is dat subroutine `trssuw` de enige plaats is waar de waterstanden worden aangepast, en dat hierbij gelijk alle informatie met betrekking tot de geometrie met de nieuwe waterstanden consistent kan worden gemaakt. Voor de berekening van `seh` op halve tijdsniveaus (berekening in x-richting) zijn dit de posities van laaginterfaces `zksh`, `zkuh`, `zkvh` en de schotjes `kfuh` en `kfvh`. Verder worden de snelheden en debieten in x-richting (`qxk`, `uh`) consistent met `seh` bepaald. Tenslotte worden de debieten in dwarsrichting (`qyk`) aangepast (op nul gezet) wanneer dit nodig is om negatieve waterdieptes te voorkomen.

De moeilijkste keuzes in dit algoritme zitten in hoe er precies wordt omgegaan met de grootheden voor de dwarsrichting. Momenteel wordt de snelheid `vp` van het oude tijdstip op nul gezet wanneer `qyk` op nul wordt gezet, om de relatie $\texttt{qyk} = \texttt{gvv} \cdot \delta\texttt{zkvp} \cdot \texttt{vp}$ te handhaven. Dit is vreemd omdat hiermee snelheden uit het verleden worden aangepast die mogelijk al op de SDS-file zijn gezet. Verder is de aanpassing niet nodig. `Vp` wordt alleen nog in het berekenen van time-histories gebruikt, en daarbij wordt de nuance van droogvallen in de dwarsrichting sowieso niet correct verwerkt. Het is overigens niet duidelijk waarom hier niet gewoon de variabele `qyk` wordt gebruikt in plaats van `vp` en `hv` (`wagcot`) of `zkvp` (`trscot`).

De snelheid `vh` op het halve tijdstip die reeds in `trscue` berekend is wordt na `trssuw` nog wèl op allerlei plekken gebruikt: in de berekening van overlaatkarakteristieken, in het $k-\epsilon$ turbulentiemodel, bij QAD-randen, in de Chezy-correctie voor saliniteitsgradiënten, en in de flow-berekeningen in de tweede halve stap. In een aantal gevallen wordt hierbij eerst gekeken of er een schotje (`kfvh`) staat voordat `vh` wordt gebruikt; wanneer er gedestaggerd wordt ($\texttt{vvv} = \overline{v}^{xy}$) dan wordt dit echter niet gedaan. Het is een mogelijkheid om `vh` op nul te zetten wanneer er in het betreffende v-punt op het halve tijdstip een schotje wordt geplaatst.

Een complicatie die in [9, par.5.1] nog niet was ontdekt is dat de discretisaties van de impulsvergelijking in `trsumo` gedeeltelijk ook blijven gebaseerd op de *nieuwe* geometrie, namelijk waar het de viscositeitskruistermen betreft. Wanneer er in de dwarsrichting in `kfvh` extra schotjes worden geplaatst en al of niet de snelheid `vh` wordt gereset, dan zou je deze termen eigenlijk opnieuw moeten discretiseren. Een alternatief is om de afhankelijkheid te

negeren. Dit is vooral te verdedigen met het feit dat de kruistermen slecht functioneren en op termijn uit de code zouden moeten worden verwijderd [10].

Tenslotte is de strategie in subroutine `trssuw` nu dat er zowel op de waterdiepte in wl-punten als op de doorstroomhoogte in v-punten wordt gecontroleerd, en dat in beide gevallen `qyk` op nul wordt gezet en de iteratie wordt herstart. Het voelt inefficiënt om te herstarten als er een punt door de controle in v-punten wordt drooggezet, omdat het schotje voor wat betreft de waterstanden niet nodig is. Verder wil je aan de geometrie in dwarsrichting zo min mogelijk aanpassingen maken. In ieder geval wil je voorkomen dat er in het iteratieproces schotjes worden bijgeplaatst die achteraf niet nodig blijken te zijn. Eventueel zou je de controle op doorstroomhoogtes in v-punten daarom na afloop van het iteratieproces willen doen. Het probleem hierbij is echter dat de schotjes, debieten en continuïteitsvergelijking dan niet langer consistent zijn met elkaar. Je wilt dat een schotje `kfvh` $= 0$ steeds impliceert dat het debiet `qyk` $= 0$, en je wilt ook dat de debieten die in `qyk` staan gelijk zijn aan de debieten die in de continuïteitsvergelijking zijn gebruikt. Dit kan alleen maar als de controle in v-punten binnen het iteratieproces wordt gedaan.

Andere keuzes die kunnen worden gemaakt betreffen de threshold die wordt gebruikt (0 of $\delta/2$) en of er bij de controle op waterstandspunten twee of vier schotjes tegelijk worden gezet. Eerstgenoemde keuze betreft activiteit 3-3 en wordt in paragraaf 3.7 verder uitgewerkt.

De tweede keuze betreft een verschil tussen hoe het nu gaat in TRIWAQ en in Delft3D-Flow. In TRIWAQ wordt de controle in wl-punten twee keer gedaan: apart voor u-punten en v-punten. Dat betekent dat als de waterstand te laag wordt er eerst twee schotjes in de u-punten worden gezet, en dat dan wordt gekeken of er nog twee schotjes in de v-punten nodig zijn. In Delft3D-Flow worden alle vier de schotjes tegelijkertijd gezet wanneer de waterstand te laag dreigt te worden. Dat is efficiënt omdat er dan minder teruggesprongen wordt, en ook hoeft er iets minder te worden gecontroleerd. Daarentegen worden er wel meer schotjes in de dwarsrichting gezet en blijft er meer water achter in een als droog gemarkeerde cel.

Wat hiermee samenhangt is dat schotjes die eenmaal zijn gezet gedurende het iteratieproces niet meer worden weggehaald. Verder is het uit de berekening nemen van een cel een stuk ingrijpender dan het zetten van een enkel schotje, waardoor de controle op wl-punten eerder dan de controle in u-punten tot extra slingeringen leidt. Daarom wil je dat het droogvallen het meeste op de u-punten gebeurt. Wat hiervoor in Delft3D-FLOW gebeurt is dat er in het iteratieproces alleen in u-punten wordt gecontroleerd (op $\delta/2$) en pas na afloop in wl-punten wordt gecontroleerd (op 0). Wanneer wl-punten droogvallen dan worden er gelijk vier schotjes gezet, worden de dwarsdebieten op nul gezet en wordt de berekening van de continuïteitsvergelijking opnieuw gedaan.

3.4.2 Keuzes met betrekking tot het nieuwe algoritme: "methode A"

Uitgangspunt bij het kiezen tussen de hierboven beschreven alternatieven voor details van het algoritme is dat de nieuwe methode geen concessies moet doen aan de eenvoud of nauwkeurigheid ten behoeve van de performance. De performance van TRIWAQ blijft sowieso op peil

omdat er bij droogvallen in de dwarsrichting minder werk opnieuw moet worden gedaan en omdat de massacorrectiestap verbeterd wordt. Met betrekking tot de eenvoud zijn echter wel diverse verbeteringen mogelijk:

1. Alle aanpassingen aan de schotjesarrays worden gemaakt binnen subroutine `trssuw` en onderliggende routines. Dit betreft met name ook de droogvalcontrole voor de dwarsrichting. Deze controle in v-punten wordt in `trssuw` uitgevoerd omdat de geometrie in de dwarsrichting (array `kfvh`) anders te lang achterloopt bij de waterstanden, en omdat ze conceptueel het beste bij de huidige berekening past (waterstanden op halve tijdstap).

2. Als een waarde eenmaal is vastgesteld dan wordt ze later niet meer veranderd. Dit betreft met name de aanpassing van de snelheid `vp` op het oude tijdsniveau in `trssuw`. Deze aanpassing wordt weggehaald.

3. Als er na afloop van `trssuw` een schotje staat dan zijn het bijbehorende debiet en de bijbehorende snelheid 0. Hiervoor wordt de controle op droogvallen in v-punten *in* het iteratieproces in `trssuw` gedaan, en als er in een v-punt een schotje wordt gezet dan worden de bijbehorende `qyk` en `vh` op nul gezet. Het op nul zetten van de snelheden `vh` wordt vooral vanuit het oogpunt van consistentie met de schotjes gedaan (snelheden/schotjes op SDS-file!).

4. De droogvalcontroles gebruiken als threshold de waarde $\delta/2$ in plaats van 0 (voorheen in Delft3D-FLOW gebruikt, zie paragraaf 3.7) of $\min(0.01, \delta/2)$, wat nu in TRIWAQ voor de dwarsrichting wordt gebruikt. De thresholds voor de checks in snelheidspunten en waterstandspunten kunnen apart worden gespecificeerd, maar zijn normaliter gelijk aan elkaar.

5. Wanneer er bij de controle in waterstandspunten in een cel te weinig water staat dan worden er gelijk vier schotjes rondom de cel geplaatst.

Dit laatste punt staat nog ter discussie omdat de nauwkeurigheid hierdoor mogelijk iets wordt aangetast omwille van de eenvoud van het algoritme. Dit wordt met gerichte testen verder onderzocht.

Verder worden ook de volgende aanpassingen gemaakt:

6. De droogvalcontrole bij aanvang van subroutine `trssuw` (in subroutine `trsdfv`) wordt verwijderd. De waterstanden en schotjes zijn namelijk niet veranderd ten opzichte van de vorige call van `trssuw` en er worden dezelfde grenswaardes gebruikt (zie punt 4), dus er valt niets droog.

7. Bij u-punten die droogvallen worden de coëfficiënten van de impulsvergelijking `aak`, `cck` en `ddk` niet op nul gezet. Dit is niet nodig omdat er bij het gebruik van deze coëfficiënten steeds eerst gekeken wordt of er een schotje staat (verticale integratie, berekening `uh`). Hiermee wordt een berekening en een aantal regels code uitgespaard.

8. Het "herstarten" van het iteratieproces in geval van droogvallen wordt verwijderd. In TRIWAQ betreft dit voor de u-richting feitelijk alleen het ophogen van het maximaal aantal iteraties, omdat er na het droogvallen met de nieuwste iterand doorgerekend wordt. Wanneer er in de derde iteratie droogvallen wordt geconstateerd, dan mogen er nu maximaal 3+ITERCON iteraties worden uitgevoerd. Het is eenvoudiger wanneer er altijd maximaal ITERCON iteraties worden gedaan. Hiermee wordt wel de veronderstelling ingebouwd dat ITERCON "groot genoeg" is (bijv. 6 of meer). Bij gebruik van te kleine ITERCON levert deze aanpassing mogelijk grotere verschillen in de resultaten op.

De volgende gewenste aanpassingen worden niet gemaakt:

9. Subroutine `trssuw` zou de waterstanden en laagposities in diagonale randpunten zodanig moeten berekenen dat `trsdif` hier niets aan aan te passen heeft.

10. Het maskerarray `kfs` dat nu in TRIWAQ wordt gebruikt zou kunnen worden verwijderd door aanpassing van de definitie van `kcs`. Punten met vier permanente schotjes eromheen (m.n. dampunten) worden dan in `kcs` met 0 gemarkeerd. Het array `kfs` kan dan worden gebruikt in de definitie van WL en kan in de droogvalcontrole in `trssuw` worden ingevuld.

11. De stabiliteitscheck ($|\zeta| < 200$) zou pas na afloop van het iteratieproces kunnen uitgevoerd. Dit scheelt wat rekenwerk, al moet er wel een keer extra voor worden gecommuniceerd. Verder wordt een verschil met WAQUA verwijderd en zou de controle in een gezamenlijke subroutine kunnen worden ondergebracht.

12. De berekening van informatie voor controlestations in subroutines `wagcot` en `trscot` zou op basis van debieten in plaats van snelheden moeten worden gedaan. Hiermee wordt deze berekening consistent gemaakt met de rekenroutines `wasdfc` en `trsdif`.

Het schema dat hieruit volgt wordt in detail uitgewerkt in Algoritme 3.

3.4.3 Een mogelijk alternatief: "methode B"

De keuzes die hierboven worden gemaakt gaan enigszins in tegen het principe dat in het OMS-project is gekozen [13, par. 5.1]:

1. A velocity point should be set dry as much as possible outside the iterative procedure where the nonlinear continuity equation is solved implicitly, because when a velocity point is set dry, the computation of the new water elevations should be restarted to preserve mass.

De argumenten die hierboven gegeven worden voor onze keuzes zijn dat we wel degelijk "zo veel mogelijk" buiten de continuïteitsvergelijking doen, en dat het binnen de iteratieve procedure uitvoeren van de checks niet veel performance kost.

Algorithm 3 - Ontwerp van het nieuwe droogvalalgoritme in subroutine `trssuw` van TRIWAQ

bereken `umean`, `tetau`, `vmean`, `tetav` t.b.v. upwindschema doorstroomhoogte
initialiseer $\texttt{seh}^{[0]}$, $\texttt{uh}^{[0]}$, $\texttt{kfuh}^{[0]}$, $\texttt{kfvh}^{[0]}$, $\texttt{zkuh}^{[0]}$
controleer op droogvallen en onderlopen (TRSDFV)
→ check op droogvallen in x-richting op $\delta/2$ wordt verwijderd
→ onderlopen in x-richting als $\texttt{hu} > \delta_{uv}$ *en*
$\max(\texttt{sep}_m, \texttt{sep}_{m+1}) - \max(\texttt{zksp}_{m,K}, \texttt{zksp}_{m+1,K}) > \delta_{wl}$
→ dit verandert $\texttt{kfuh}^{[0]}$ en laat `kfup` ongemoeid
bereken gediscretiseerde impulsvergelijking (`kfup` → `aak`, `cck`, `ddk`) (TRSUMO)
for iteratie $q = 1, 2, \ldots$ (maximaal ITERCON) **do**
in eerste iteratie en als er v-punten zijn drooggevallen:
bereken vaste deel continuïteitsvergelijking (`qyk` → `d0k`, `e`)
randvoorwaarden voor u-punten, verticale integratie ($\texttt{zkuh}^{[q-1]}$ → `aa`, `cc`, `dd`),
preconditioneren voor DDHOR, randvoorwaarden wl-punten
opstellen en oplossen tri-diagonale stelsels voor waterstanden $\texttt{seh}^{[q]}$
bereken `qxdd` t.b.v. DDHOR, communiceer $\texttt{seh}^{[q]}$
bereken $\texttt{zkuh}^{[q]}$, $\texttt{zkvh}^{[q]}$ (alleen bovenste laag?), controleer stabiliteit
droogvalcontroles in u-punten, v-punten en wl-punten (TRSDRY)
→ in u-punten op $\delta_{uv}/2$, aanpassing $\texttt{kfuh}^{[q]}$
→ in v-punten op $\delta_{uv}/2$, aanpassing $\texttt{kfvh}^{[q]}$, `qyk` en `vh`
→ in wl-punten op $\delta_{wl}/2$, aanpassing $\texttt{kfuh}^{[q]}$ *en* $\texttt{kfvh}^{[q]}$, `qyk`, `vh`
i.g.v. convergentiecontrole op snelheden: bereken $\texttt{uh}^{[q]}$, $\texttt{qxk}^{[q]}$
communiceer t.b.v. stopcriterium (WASCHK)
end for
i.g.v. convergentiecontrole op waterstanden: bereken $\texttt{uh}^{[q]}$, $\texttt{qxk}^{[q]}$
ververs coëfficiëntenset `sepu_full` met $\texttt{zkuh}^{[q]}$ t.b.v. verticale interpolatie
communiceer $\texttt{qxk}^{[q]}$
voer de massacorrectiestap uit, incl. berekening `zkuh` (TRSMSS)
bereken `zksh`, ververs coëfficiëntensets m.b.t. laagposities
bereken verticale snelheden en debieten
bepaal maskerarray `khs` en ververs coëfficiëntenset t.b.v. horizontale interpolaties

Vanwege twijfels omtrent de performance van "methode A" is er in de ontwerpfase van het project een alternatief bedacht dat tot minder rekenwerk zou moeten leiden. Dat alternatief wordt "methode B" genoemd. Het voldoet meer aan het principe van het OMS keuzesverhaal. Gaandeweg het project is echter gebleken dat de performance van TRIWAQ met "methode A" beter in plaats van slechter wordt. Daarom wordt methode B niet gebruikt en wordt ze ook niet in veel detail uitgewerkt.

Methode B wordt op de volgende manier geconstrueerd:

1. In het iteratieproces voor de continuïteitsvergelijking wordt niet op de doorstroomhoogtes in de dwarsrichting (`zkvh`) gecontroleerd.

2. Bij het oplossen van de impulsvergelijking aan het begin van een halve tijdstap wordt wel extra gecontroleerd. Daarbij worden de doorstroomhoogtes van een halve tijdstap eerder gebruikt (`zkvp`). Wanneer er schotjes worden gezet dan worden ook de debieten (`qyk`) gereset.

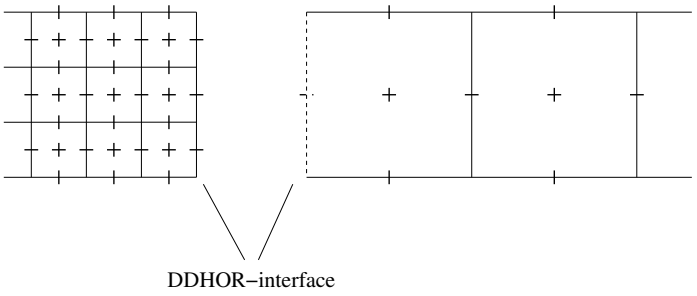
3. "Rommelen aan het verleden" wordt voorkomen door hierbij schotjes te zetten op het nieuwe halve tijdstip (`kfvh`). Hetzelfde geldt voor de debieten `qyk`, waarvoor conceptueel waardes per halve tijdstap kunnen worden onderscheiden. Niet de debieten `qykp` uit de continuïteitsvergelijking maar de debieten `qykh` voor de volgende halve stap worden gereset.

Merk op dat de droogvalcontrole voor de dwarsrichting hiermee *naar voren* wordt gehaald. In de eerste halve tijdstap wordt de continuïteitsvergelijking in de x-richting opgelost (berekening `uh`, `zkuh`, `kfuh`), en de tweede halve tijdstap begint dan met de impulsvergelijking voor de x-richting inclusief droogvalcontrole voor `kfup`.

Dit alternatief is nog niet in detail uitgewerkt, maar lijkt een goed werkbaar algoritme te kunnen leveren. Het grootste bezwaar lijkt dat er negatieve doorstroomhoogtes `zkvh` en `zkup` kunnen voorkomen zonder dat er in betreffende punten een schotje wordt gezet. De consistentie tussen `zkvh` en `kfvh` gaat dus verloren. Daar staat tegenover dat `zkvh` en `zkup` de minder belangrijke posities van laaginterfaces in het ADI-schema zijn. In het transportgedeelte van TRIWAQ worden deze waardes niet gebruikt, in WAQUA worden de overeenkomstige arrays `hu` en `hv` zelfs alleen uitgerekend om de halve stap. Toch kunnen er hierdoor problemen ontstaan: in `trsumo` wordt `zkup` voor het discretiseren van de impulsvergelijking gebruikt, en wordt er bij de vertikale viscositeit, de bodemwrijving en de windstress door de laagdiktes gedeeld. Hiervoor moet dan aan het begin van `trssuw` bij de onderloopcontroles in `trsdfv` een extra droogvalcontrole worden geïntroduceerd.

3.4.4 Verdere uitwerking van subroutine `trsdry`

De meest ingrijpende aanpassingen om Algoritme 3 te bereiken betreffen subroutine `trsdry`. Deze subroutine wordt momenteel gebruikt om de droogvalcontroles in u- en v-richtingen gedurende het iteratieproces te implementeren (via twee aparte calls), inclusief de benodigde communicaties en aanvullende berekeningen voor subdomeinranden en horizontale verfijning.



Figuur 3.1: *Schematische weergave van een DDHOR-interface tussen twee domeinen. De interface loopt door snelheidspunten (cellfaces), en hoort bij het fijne domein. Het snelheidspunt van het grove domein op de interface is een "interpolatiepunt". De waardes hierin (snelheden, debieten, schotje) worden bepaald door interpolatie van waardes uit het fijne domein.*

Er zijn een aantal speciale aanpassingen nodig ten behoeve van domein decompositie. De strategie die hier wordt gebruikt is als volgt:

- De interface tussen twee domeinen loopt door snelheidspunten en hoort bij het fijne domein, zie Figuur 3.1. Het snelheidspunt van het grove domein op de interface is een "interpolatiepunt". Waardes hierin worden niet uit een differentiaalvergelijking bepaald, maar via communicatie/interpolatie verkregen van het fijne domein.

- Ieder (sub-)domein voert alle gebruikelijke controles uit in zijn eigen (u, v, wl-) roosterpunten. Het fijne domein bepaalt dus of er droogvallen/onderlopen plaatsvindt op de interface.

- Voor de interpolatie van schotjes wordt de "max"-functie gebruikt. Dat betekent dat er in het interfacepunt van het grove domein alleen dan een schotje staat (0) wanneer er in alle overeenkomstige interfacepunten van het fijne domein ook een schotje staat. Is tenminste een van de punten van het fijne domein nat (1) dan is ook het punt van het grove domein nat.

- Wanneer er in de eerste roostercel van het grove domein te weinig water staat dan wil het grove domein twee schotjes plaatsen in de rekenrichting (zie Figuur 3.1). Dit kan hij niet direct doen, schotjes op de interface moeten door het fijne domein worden gezet. Hiervoor worden de droogvallende wl-punten in een werkarray `idrywl` gemarkeerd, en wordt dit array via interpolatie aan het fijne domein toegestuurd. Het fijne domein zet dan schotjes in interfacepunten waarvoor de eerste guardbandcel in `idrywl` is gemarkeerd.

Het huidige algoritme van subroutine `trsdry` wordt geïllustreerd in Algoritme 4. De verschillende if-statements zijn geïntroduceerd om dezelfde routine te kunnen gebruiken voor zowel de rekenrichting (resetten impulsvergelijking) als de dwarsrichting in subroutine `trssuw` (resetten snelheden). Hiervoor is ook de droogvalgrenswaarde geparametriseerd: in de rekenrichting wordt $\delta/2$ gebruikt, in de dwarsrichting $\min(0.01, \delta/2)$.

Algorithm 4 - Huidige algoritme van droogvalcontroles in subroutine `trsdry` van TRIWAQ

```
controleer op droogvallen van u-punten (grenswaarde tresh/2), markeer in kfu
if ( controle in wl-punten gewenst ) then
    initialiseer idrywl=0
    controleer op droogvallen in wl-punten (grenswaarde tresh/2), markeer
        in kfu en idrywl
    communiceer kfu en idrywl
    zet extra schotjes in interfacepunten op basis van idrywl
else
    communiceer kfu
end if
if ( resetten van de impulsvergelijking gewenst ) then
    zet aak=cck=ddk=0 in alle punten met kfu=0
end if
if ( resetten van snelheden gewenst ) then
    zet u=qxk=0. in alle punten met kfu=0
end if
```

Een moeilijkheid bij het aanpassen van `trsdry` om het nieuwe droogvalalgoritme te implementeren zit in de benodigde communicatie van `vh` en `qyk`. Dit blijkt in de huidige implementatie ook niet 100% goed te gaan.

Het probleem is dat de debieten `qyk` op DDHOR-interfaces veranderen wanneer er in interfacepunten schotjes worden gezet. Deze debieten moeten in beide domeinen bekend zijn voor de continuïteitsvergelijking (array `e` in Algoritme 3). Voor dit "bekend zijn" wordt meestal communicatie gebruikt. In subroutine `trsdry` wordt echter een alternatief gebruikt: beide domeinen maken dezelfde aanpassingen. Ze zorgen er beiden voor dat het debiet op de interface op nul wordt gezet wanneer er op de interface een schotje wordt geplaatst, en omdat de debieten waarmee ze beiden begonnen gelijk waren is het debiet na afloop van `trsdry` ook gelijk. Dachten we. De moeilijkheid ontstaat wanneer het fijne domein schotjes zet op *een gedeelte* van de interface. Van de drie debieten van het fijne domein wordt er een op nul gezet. Hiermee verandert ook het geïnterpoleerde debiet van het grove domein. Maar die verandering wordt nu niet correct verwerkt.

Het nieuwe algoritme voor subroutine `trsdry` wordt gepresenteerd in Algoritme 5. Dit algoritme is als volgt tot stand gekomen:

- het resetten van de impulsvergelijking is weggehaald omdat dit door zorgvuldig gebruik van `aak-ddk` overbodig is gemaakt, zie punt 7 in paragraaf 3.4.2.
- de parameter `tresh` (met variabele inhoud) is vervangen door de variabelen `trshuv`$= \delta_{uv}$ en `trshwl`$= \delta_{wl}$ uit de gebruikersinvoer. Deze namen worden later ook in WAQUA ingevoerd in plaats van de eerder gebruikte namen `var` en `trsh`.

Algorithm 5 - Nieuwe algoritme van droogvalcontroles in subroutine `trsdry` van TRIWAQ

controleer op droogvallen van u-punten ($\delta_{uv}/2$), markeer in $\texttt{kfuh}^{[q]}$
controleer op droogvallen van v-punten ($\delta_{uv}/2$), markeer in $\texttt{kfvh}^{[q]}$, reset `vh`,`qyk`
if (controle in wl-punten gewenst) **then**
  initialiseer `idrywl=0`
  controleer op droogvallen in wl-punten ($\delta_{wl}/2$),
  → als er nog geen twee u-schotjes staan: zet u-schotjes in $\texttt{kfuh}^{[q]}$ en zet `idrywl=1`
  → anders, als er nog geen twee v-schotjes staan: zet v-schotjes in $\texttt{kfvh}^{[q]}$, reset `vh`,`qyk`, en zet `idrywl=2`
  communiceer $\texttt{kfuh}^{[q]}$, $\texttt{kfvh}^{[q]}$, `vh`, `qyk` en `idrywl` (COCUPD)
  zet extra schotjes op de interfaces van subdomeinen (parallel, DDHOR) op basis van `idrywl`
  → u-schotjes naast waterstandspunten met `idrywl.ge.1`
  → v-schotjes naast waterstandspunten met `idrywl.eq.2`, reset `vh`, `qyk`
else
  communiceer $\texttt{kfuh}^{[q]}$, $\texttt{kfvh}^{[q]}$, `vh` en `qyk` (COCUPD)
end if

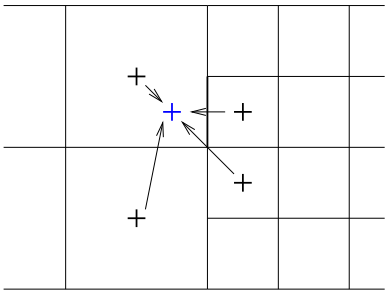
- De droogvalcontrole voor de snelheidspunten in de dwarsrichting is ingevoerd.

- Wanneer er schotjes in de dwarsrichting worden gezet dan worden gelijk de snelheden en debieten gereset. Hierdoor kunnen de goede debieten worden gecommuniceerd in parallelle berekeningen en bij gebruik van horizontale verfijning. De aparte loop voor het resetten van snelheden komt hiermee te vervallen.

- Er is een truc bedacht om met een simpele implementatie toch twee schotjes tegelijk te kunnen zetten. Hiervoor wordt in de droogvalcontrole voor wl-punten eerst gekeken of er al twee u-schotjes staan. Als dat niet zo is dan worden er geen v-schotjes gezet. Als de waterstand niet voldoende stijgt door de u-schotjes dan worden de v-schotjes een iteratie later alsnog gezet. Overigens leidt het zetten van twee schotjes tegelijk in bepaalde situaties tot een ongewenste hobbel in de tijdseries van waterstanden.

- De benodigde communicaties voor DDHOR zijn in detail uitgezocht. Daar is uitgekomen dat het hulparray `idrywl` zowel voor de rekenrichting als de dwarsrichting kan worden gebruikt. Wel moet er hierbij de goede interpolatiemethode worden gebruikt, zie onder. Verder is communicatie van `kfvh`, `vh` en `qyk` toegevoegd.

- Het laten doorwerken van veranderingen in `qyk` op het vaste gedeelte van de continuiteitsvergelijking (arrays `d0k` en `e`) wordt in subroutine `trssuw` gedaan, zie Algoritme 3.



Figuur 3.2: *Interactiepatroon voor bilineaire interpolatie bij* 1 : 2 *verfijning. De zwarte plussen geven "discretisatiepunten" aan waarvan de waardes worden gebruikt voor het interpoleren naar het in blauw getekende "interpolatiepunt".*

Bij de gedegen analyse van het gebruik van array `idrywl` in domein decompositie runs is een complicatie in de oude implementatie opgespoord die tot onterechte extra schotjes op de DDHOR-interface leidt. Deze complicatie wordt geïllustreerd in Figuur 3.2 waarin het interactiepatroon voor bilineaire interpolatie wordt getoond.

In de bilineaire interpolatie wordt in vier kwadranten het dichtstbijzijnde punt geselecteerd. In het getekende geval zou ook het bovenste punt van het fijne domein kunnen worden gekozen afhankelijk van de precieze definitie van de kwadranten. Dat is voor uitleg van het huidige probleem niet van belang. In de bilineaire interpolatie zelf worden op basis van de coördinaten van de vier omliggende punten gewichten bepaald. Bij de interpolatie van array `idrywl` worden deze gewichten echter niet gebruikt, omdat een "max-versie" van de bilineaire interpolatie wordt gebruikt. De waarde in het interpolatiepunt wordt dan bepaald als het maximum van de waardes in de getoonde discretisatiepunten.

Het probleem betreft nu het tweede snelheidspunt van boven op de interface. In de figuur is deze cellface vet getekend. Het fijne domein zal in dit snelheidspunt een schotje neerzetten als dit domein uit de communicatie een waarde `idrywl.ge.1` verkrijgt. Maar in de figuur is te zien dat dat ook gebeurt als de onderste cel van het grove domein droogvalt! In dat geval is het schotje in de beschouwde cellface onterecht en ongewenst. Zulke ongewenste schotjes blijken inderdaad regelmatig voor te komen in testen met domein decompositie waarin droogvallen plaatsvindt.

Gelukkig is er een eenvoudige oplossing voor dit probleem. Deze oplossing is om een andere interpolatiemethode te gebruiken in de communicatie van `idrywl`. Een geschikte interpolatiemethode is "`max_cellavg`". Deze methode is op cel-gemiddelden gebaseerd. Interpolatiepunten van het fijne domein krijgen hierin de waarde van de omhullende discretisatie-cel van het grove domein, en voor de interpolatie van fijn naar grof worden gemiddeldes over meerdere cellen bepaald. Van de vier pijlen in Figuur 3.2 blijft alleen die links-boven over. Het is evident dat er hiermee geen onterechte schotjes meer kunnen worden gezet.

3.4.5 Testen

Het nieuwe droogvalalgoritme voor TRIWAQ wordt op twee manieren getest. Ten eerste wordt de testbank gedraaid om te bepalen welke modellen de grootste verschillen geven. Met deze modellen wordt onderzocht of de implementatie correct is gedaan. Ten tweede wordt er een geschikt testmodel voor droogvallen en onderlopen gezocht. Hierbij wordt gedacht aan een model van de Oosterschelde of de Waddenzee. Hiermee wordt gekeken of de resultaten van de simulatie merkbaar verbeterd of verslechterd zijn. Het in het testverslag van RIKZ gebruikte Kustzuidmodel lijkt voor deze excercitie minder geschikt.

Verder wordt ook de performance gerapporteerd. Met betrekking tot de performance kan vooral keuze 3 enigszins ter discussie worden gesteld. Een test die hiervoor zou kunnen worden uitgevoerd is er een met een lineair oplopende bodem, waarin het droogvallen van v-punten nadrukkelijk aan de orde zou moeten zijn.

3.4.6 Documentatie

De droogval- en onderloopprocedures worden beschreven in paragraaf 7.4 van de technische documentatie van TRIWAQ [12]. Deze beschrijvingen moeten her en der worden aangepast ten aanzien van de aanpassingen. Verder moet de data-analyse van subroutine `trssuw` in [7] worden aangepast.

3.5 Activiteit 3-2: Verwijderen van 4 verschillen tussen TRIWAQ en WAQUA

In het droogvalrapport [9] wordt een viertal kleinere verschillen tussen de algoritmes van WAQUA en TRIWAQ geconstateerd. Deze verschillen zijn historisch ontstaan, zonder dat er een goede motivatie voor te geven is. Het algoritme van TRIWAQ lijkt op deze punten beter te functioneren. Uit beheersoogpunt is het daarom gewenst om het algoritme van WAQUA op deze punten aan te passen aan dat van TRIWAQ:

1. Verplaatsen van droogval/onderloopcontroles uit subroutine `wasuxc` naar `wassuc`.

2. Toevoegen van een droogvalcontrole voorafgaand aan het iteratieproces in subroutine `wassuc`, conform de overeenkomstige controle in `trsdfv`.

3. Toevoegen van de truc van TRIWAQ bij de onderloopcontrole in `wassuc`, om te voorkomen dat punten nat worden gezet die in het iteratieproces weer droog worden gemaakt.

4. Niet compleet herstarten van het iteratieproces, maar doorrekenen met de nieuwste iterand.

In dit project zijn er ook extra verschillen opgevallen die ook ongedaan zijn gemaakt:

5. Bij het onderlopen van randpunten in WAQUA (subroutine `wasuxc`) wordt in het randpunt een "startsnelheid" ingesteld, in TRIWAQ gebeurt dit niet. Deze verfijning wordt uit WAQUA verwijderd.

6. De randvoorwaarde voor de waterstand verschilt tussen WAQUA en TRIWAQ. Hier wordt de methode van WAQUA (spiegelen) ook in TRIWAQ geïmplementeerd.

7. De droogvalcontrole in *wl*-punten wordt in WAQUA ook in punten van de open rand gedaan terwijl dat in TRIWAQ niet gebeurt. Deze controle wordt ook in TRIWAQ ingevoerd.

Tenslotte is er nog een kleine aanpassing gemaakt in de randafhandeling van zowel WAQUA als TRIWAQ:

8. In droge randpunten wordt de vergelijking voor de waterstand aangepast als deze anders onder de bodem valt.

Deze punten worden in de volgende paragrafen verder uitgewerkt.

3.5.1 Droogvallen en de randafhandeling

De droogval- en onderloopcontroles in subroutine `wasuxc` (punt 1 hierboven) betreffen alleen de afhandeling van randpunten. Hiermee zijn diverse problemen geweest. Een nieuw probleem van Martin Scholten onderstreept deze moeilijkheden en geeft extra motivatie voor het gelijktrekken van WAQUA met TRIWAQ op dit punt.

Het probleem van Martin Scholten betreft een kanaal van 1 cel breed met vlakke bodem, debietrand (met automatische verdeling) aan de bovenstroomse kant en waterstandsrand benedenstrooms. De opgegeven waterstand loopt op vanaf de beginwaarde tot $8\,m$. Wanneer de initiële waterstand net boven $\delta/2$ wordt gekozen (kanaal is initiëel geheel nat) dan verloopt de simulatie goed. Wordt de initiële waterstand echter net onder $\delta/2$ gekozen (kanaal initiëel droog) dan blijft het kanaal gedurende de hele simulatie droog. Een simulatie met TRIWAQ voor deze situatie blijkt wel goed te gaan.

Dit probleem is herleid tot de onderloopcontrole die in subroutine `wasuxc` wordt gedaan. In deze controle wordt gekeken of de doorstroomhoogte `hu` in het randpunt groter dan δ is, *èn* of de waterdiepte in de twee naburige waterstandspunten groter dan δ is. Deze laatste controles zijn afgeleid van de droogvalcontroles in waterstandspunten. Ze zijn bedoeld om te voorkomen dat schotjes worden weggehaald naast cellen met een negatieve waterdiepte, wat in de transportsolver van WAQUA tot antidiffusie leidt. Dat laatste trad op in januari 2001 in het Zeedelta-model met concentraties oplopend tot 10^{45} en afbreken van de simulatie als gevolg.

In het huidige model zorgen deze extra controles echter voor een probleem. In de eerste interne cel kan de waterstand nooit groter dan δ worden voordat het schotje wordt weggehaald. Daarmee zorgt de controle ervoor dat het kanaal nooit onderloopt. Wanneer de onderloopcontrole conform TRIWAQ naar subroutine `wassuc` wordt verhuisd dan wordt dit probleem

opgelost. Daar is het namelijk niet nodig om bij het onderlopen gelijk op de waterstanden in naburige cellen te controleren. Die controles worden namelijk bij de droogvalcontroles gedurende het iteratieproces uitgevoerd. Overigens zou het probleem mogelijk ook met een verfijning van de controle in `wasuxc` kunnen worden opgelost, *à la* de truc die in de onderloopcontroles in TRIWAQ wordt toegepast (punt 3).

Het verhuizen van de check betekent het verwijderen van loop 210 uit subroutine `wasuxc` en aanpassen van loops 190 en 220 in subroutine `wassuc`. Aangezien TRIWAQ het voorbeeld is zal voor de nieuwe code in `wassuc` subroutine `trsdfv` als voorbeeld worden gebruikt. `Trsdfv` is niet geheel te gebruiken voor WAQUA omdat er omtrent barriers nog andere verschillen zijn (wel/geen array `lgubar`).

Wanneer een schotje wordt weggehaald in subroutine `wasuxc` dan wordt daar momenteel voor waterstandsrandpunten ook de snelheid op de helft van de snelheid van het eerste interne snelheidspunt gezet (punt 5). Deze "startsnelheid" beïnvloedt termen in de impulsvergelijking van het punt zelf (advectie, bodemwrijving) en van punten eromheen (advectie). In de meeste gevallen komt de stroming hierdoor gladder op gang en wordt het omliggende gebied minder verstoord. Wel gaat dit om een tijdelijk effect. In TRIWAQ wordt geen vergelijkbare aanpassing gemaakt. Merk op dat dit een speciale discretisatie is voor punten van type 3 uit paragraaf 3.2.1. In principe kunnen er met de nieuwe schotjesarrays meer van dit soort aanpassingen aan de discretisaties worden gemaakt. Omwille van de eenvoud kiezen we er vooralsnog echter voor om het aanpassen van de snelheid uit WAQUA te verwijderen.

Een klein verschil tussen WAQUA en TRIWAQ dat bij het onderzoeken van het probleem van Martin Scholten is geconstateerd betreft de waterstand in drooggevallen randpunten (punt 6). In waterstandsrandpunten wordt hier in zowel WAQUA als TRIWAQ de echte randvoorwaarde gehanteerd, maar voor snelheids-, debiet- en Riemannranden worden verschillende formules gehanteerd. WAQUA gebruikt voor droge randpunten nog steeds spiegeling van de waarde van het eerste interne waterstandspunt: $\zeta_1' = \zeta_2'$, net zoals ook voor natte randpunten wordt gebruikt. TRIWAQ handhaaft echter de oude waarde in het drooggevallen punt: $\zeta_1' = \zeta_1$.

De methode van WAQUA zal hier in het algemeen leiden tot een hogere waterstand. Daarmee kan het randpunt gemakkelijker (sneller) weer onderlopen. Dat is sterk gewenst, vooral voor instroomranden met automatische debietverdeling. De waterstandstoename in het eerste interne punt kan alleen toenemen door zijdelingse toestroming, en dit komt altijd langzamer op gang dan de toestroom in de werkelijkheid. Daarom zal de randafhandeling van TRIWAQ op dit punt op die van WAQUA worden afgestemd. Dit betreft een kleine aanpassing aan subroutine `trsrv2`.

Tijdens het testen bleken er echter spiegelpunten voor te komen waar de waterstand onder de bodem kwam. Hiervoor is nog een kleine aanpassing aan deze randen gemaakt: de spiegeling wordt alleen toegepast wanneer de waterstand in het interne punt boven de bodem in het randpunt ligt, anders wordt de waterstand net boven de bodem (de helft van de drempelwaarde `trshwl`) voorgeschreven. Op deze manier wordt nu gegarandeerd dat de waterstand altijd boven de bodem blijft.

Bij de testen die zijn uitgevoerd in dit project is ook regelmatig gekeken naar de hoeveelheid water die achterblijft in droge gebieden. Daarvoor wordt de grootheid "`sep+dps`" geplot. In droge punten hoort deze grootheid altijd positief te zijn (als er een controle in dwarsrichting wordt gebruikt), maar dat bleek niet zo te zijn. In punten van waterstandsranden kan de waterstand onder de bodem komen (punt 8). Dat komt omdat de door de gebruiker opgegeven waterstand wordt opgelegd, zonder dat daarbij de bodemhoogte wordt beschouwd. Hiervoor is een kleine aanpassing bij het opdrukken van de randvoorwaardes gemaakt in subroutines `wasrv2` en `trsrv2`. In plaats van de waarde `cirbou` van de gebruiker direct toe te passen wordt nu de volgende waarde gebruikt:

```
d(nmf) = max( cirbou(1,ic), -dps(nmf)+htrswl )
```

Door het spiegelen van waterstanden dat bij snelheidsranden wordt gebruikt kan de waterstand ook onder de bodem komen. Daarom is ook hier een kleine aanpassing gemaakt. Als de waterstand in het interne punt (`nmfu` of `nml`) voldoende boven de bodem in het randpunt ligt dan wordt spiegelen gebruikt, anders wordt de minimale waterstand opgelegd.

Een laatste punt dat met de randafhandeling te maken heeft betreft punt 7 hierboven: wel/geen droogvalcontrole in waterstandsrandpunten. Het is gewenst dat WAQUA en TRIWAQ op dit punt op elkaar worden afgestemd. Vanuit consistentie, de controle is of in allebei de modellen nodig of in geen van beiden, en omdat hiermee mogelijk op termijn dezelfde subroutine voor de droogvalcontroles kan worden gebruikt.

Het verschil uit zich in de mogelijkheid van randpunten met een negatieve waterdiepte zonder schotjes eromheen in TRIWAQ terwijl zulke punten in WAQUA niet voorkomen. Het type randvoorwaarde is hierbij nagenoeg niet van belang. Voor het gemak kunnen we ons daarom beperken tot waterstandsranden, recht en diagonaal. Omdat er hier geen spiegeling van waterstanden wordt gebruikt kan het hierbij gemakkelijk gebeuren dat het randpunt droog moet worden beschouwd terwijl er in de eerste interne roostercel nog voldoende water staat.

Dat negatieve waterdieptes in TRIWAQ niet tot problemen leiden komt waarschijnlijk door de randvoorwaarden die in het transportgedeelte worden gehanteerd. Bij instroom is dit een Dirichlet-voorwaarde waarbij de randwaarde met de methode van Tatcher-Harlemann wordt bepaald. Bij uitstroom wordt de oude concentratie en pure advectie gebruikt. In beide gevallen is de randvoorwaarde op een zodanige manier gediscretiseerd dat de waterdiepte in het randpunt er geen rol in speelt. Daarmee wordt de controle in randpunten effectief overbodig gemaakt. In WAQUA worden dezelfde randvergelijkingen in het transportgedeelte gebruikt. Het eerder genoemde probleem met enorme concentraties in het Zeedeltamodel speelde dan ook in de eerste inwendige cel. Het is daarom mogelijk om de droogvalcontrole in waterstandspunten over te slaan voor de randpunten. Dat is vooral voor de eenvoud van de code gewenst.

Op dit moment neemt de hele controle in *wl*-punten in TRIWAQ 13 relatief eenvoudige code-regels in beslag. In WAQUA komen hier voor de controle voor randpunten 26 relatief lastige regels bij. In Delft3D-FLOW worden de randpunten echter via een andere aanpassing veel gemakkelijker meegenomen in het proces. Daar is de waarde van `dps` in de randpunten

zodanig gedefiniëerd dat er in de controle geen onderscheid tussen interne en randpunten hoeft te worden gemaakt.

De extra controle in randpunten is iets netter en veiliger en kan via de aanpak van Delft3D-FLOW relatief eenvoudig in WAQUA en TRIWAQ worden geïmplementeerd. Hiervoor worden subroutines `wasdry` en `trsdry` aangepast. Eventueel kan ook de berekening van `dps` in `wapdps` en `hdry` in `trshdr` worden verfijnd. Daarin wordt nu niet echt rekening gehouden met diagonale openingen. In een diagonaal openingspunt wordt de diepte van het v-snelheidsrandpunt gehanteerd (`0.5*(h(nfm)+h(nfmd))`) in plaats van dat het gemiddelde van de dieptes van de u- en v-randpunten wordt gebruikt. Deze verfijning zal vooralsnog niet binnen het huidige project worden gemaakt.

3.5.2 Details van het droogvalalgoritme en het iteratieproces

De andere drie punten hierboven (2, 3, 4) betreffen de precieze details van het droogvalalgoritme en het iteratieproces.

Punt 2 betreft de droogvalcontrole die in TRIWAQ wel en in WAQUA niet wordt uitgevoerd voorafgaand aan het iteratieproces. Het voordeel van deze controle is dat er gelijk al punten droog worden gemarkeerd, zodat er hiervoor geen nutteloze extra iteratie moet worden gedaan. Een mogelijk nadeel is dat punten droog worden gemarkeerd waarvoor er in de eerste iteratie voldoende water zou kunnen worden aangevoerd om ze nat te houden. Het onderloopproces wordt hierdoor mogelijk iets vertraagd.

De overeenkomstige droogvalcontrole in TRIWAQ komt te vervallen bij activiteit 3-4. Ze wordt overbodig gemaakt doordat er aan het einde van het iteratieproces in beide richtingen volledig op droogvallen is gecontroleerd. Aan het begin van de volgende halve tijdstap hoeven er daarom nooit schotjes te worden bijgeplaatst. Het verschil tussen WAQUA en TRIWAQ wordt daarom op een andere manier geëlimineerd, en in WAQUA zal geen extra droogvalcontrole worden ingevoerd.

Punt 3 betreft de onderloopcontroles die ook voorafgaand aan het iteratieproces worden uitgevoerd. TRIWAQ gebruikt hierbij een truc om zo min mogelijk punten nat te laten worden die in het iteratieproces weer droog worden gemaakt. De truc is dat een schotje alleen wordt weggehaald wanneer er in een van beide naburige cellen voldoende water aanwezig is. Dat is van belang voor de performance en is van belang voor de discretisatie van de impulsvergelijking zolang de tijdsniveaus van schotjes hierbij niet zijn aangepast. Aan de andere kant vertraagt deze truc mogelijk ook weer het onderloopproces, omdat er van een rij droge punten steeds (hooguit) 1 schotje per tijdstap wordt weggehaald.

De vertraging van het onderloopproces door de truc betreft waarschijnlijk een klein effect. Ook zonder te truc zijn er mechanismen waardoor de snelheid tot 1 cel per tijdstap is beperkt. Bijvoorbeeld in het testkanaal van Martin Scholten: alle `hu`'s zijn initiëel kleiner dan $\delta/2$, dus moet er eerst aan de ene kant van een cel water aangevoerd worden voordat aan de andere kant het volgende schotje kan worden weggehaald. Dit kan als een tijdstapbeperking worden gezien op basis van het Courantgetal voor advectie: $U\Delta t/\Delta x < 1$. Voor een werkelijke verbetering van het onderloopproces moeten daarom grootschaligere aanpassingen worden gemaakt, zoals

via de "artificial porosity" methode van V$\mathcal{O}$Rtech Computing [1] (in [9] "pseudowaterstand" genoemd) of via de methode van DELFT-FLS van WL|Delft Hydraulics [4].

Het performance-effect is wel degelijk reëel. Wanneer in het model van de Bergse Maas (Maasdemo-model) optie MAX wordt gebruikt dan worden er in de eerste paar uur van de simulatie per tijdstap 1-10 schotjes weggehaald en heeft de truc geen effect. Wanneer echter de droogvaloptie MEAN wordt gebruikt dan worden er nu ruim 5200 schotjes per stap weggehaald en in het iteratieproces weer teruggezet. Wanneer de truc wordt toegepast dan worden vrijwel alle onterechte schotjes goed herkend en worden er slechts 10-20 per tijdstap weggehaald. Dit geeft duidelijk aan waarom de truc ook in WAQUA moet worden toegepast. Dit vergt een aanpassing van een paar regels code in loop 220 in subroutine `wassuc`.

Het laatste van de behandelde verschillen tussen WAQUA en TRIWAQ betreft tenslotte punt 4. Wanneer er in TRIWAQ een punt droog valt in een rij dan wordt de beginschatting voor de waterstand in die rij (waterstand van vorige halve tijdstap) niet teruggezet. In WAQUA gebeurt dit wel. Dit kan leiden tot verschillen tussen simulaties die met WAQUA en TRIWAQ zijn uitgevoerd.

Voor de berekende waterstanden in droge punten is het terugzetten van de begintoestand niet nodig; de oude waterstand wordt door het oplossen van de continuïteitsvergelijking bepaald. Voor natte punten lijkt de methode van TRIWAQ iets efficiënter: de al uitgevoerde iteraties worden niet ongedaan gemaakt. Wel wordt het resultaat iets meer afhankelijk van de precieze linearisaties die worden gebruikt. Tenslotte is een verschil dat er in TRIWAQ een loop van 24 regels code wordt uitgespaard.

Ten behoeve van de eenvoud en performance wordt het terugzetten van de beginschatting uit WAQUA verwijderd. Hiervoor wordt loop 3700 verwijderd uit subroutine `wassuc`. Daarbij moet wel op het resetten van de coëfficiënten `aa-dd` van de impulsvergelijking worden gelet, zie punt 7 in paragraaf 3.4.2.

3.5.3 Testen

Het is lastig om de wijzigingen voor deze activiteit goed te testen. Enerzijds om te bewijzen dat het werk goed is gedaan, anderzijds om de invloed van de wijzigingen op de modelresultaten te bepalen. Voor beide doelen zal de eerder beschreven aanpak met de testbank worden gebruikt: dat model in detail bestuderen dat de grootste verschillen laat zien.

3.5.4 Documentatie

Documentatie die met deze activiteit te maken heeft betreft beperkte stukken van de technische documentatie van WAQUA ([2, Hfdst 8], de data-analyse van het rekenhart van WAQUA, en het testverslag dat in dit project wordt gemaakt.

3.6 Activiteiten 3-4 en 3-5: Algoritme dwarsrichting en negatieve controlevolumes in WAQUA

Nadat het nieuwe droogvalalgoritme van paragraaf 3.4 in TRIWAQ is geïmplementeerd en getest en de verschillen tussen WAQUA en TRIWAQ (paragraaf 3.5) verminderd zijn kan het nieuwe algoritme ook in WAQUA worden geïmplementeerd. Dit is voor WAQUA een ingrijpendere stap dan voor TRIWAQ omdat hierbij in WAQUA voor het eerst een droogvalcontrole in de dwarsrichting wordt gemaakt. Deze controle dient ter voorkoming van negatieve controlevolumes (activiteit 3-5).

De droogvalcontrole in de dwarsrichting is lastig te implementeren in WAQUA omdat het aantal iteraties in WAQUA per rij apart wordt bepaald. De ene rij kan eerder uit de berekening worden genomen dan de andere, en moet weer in de berekening actief worden gemaakt wanneer er in de dwarsrichting droogvallen plaatsvindt. Dit kan met name een complexe aangelegenheid worden in parallelle berekeningen wanneer ook nog het "minit-schema per rij" wordt gebruikt.

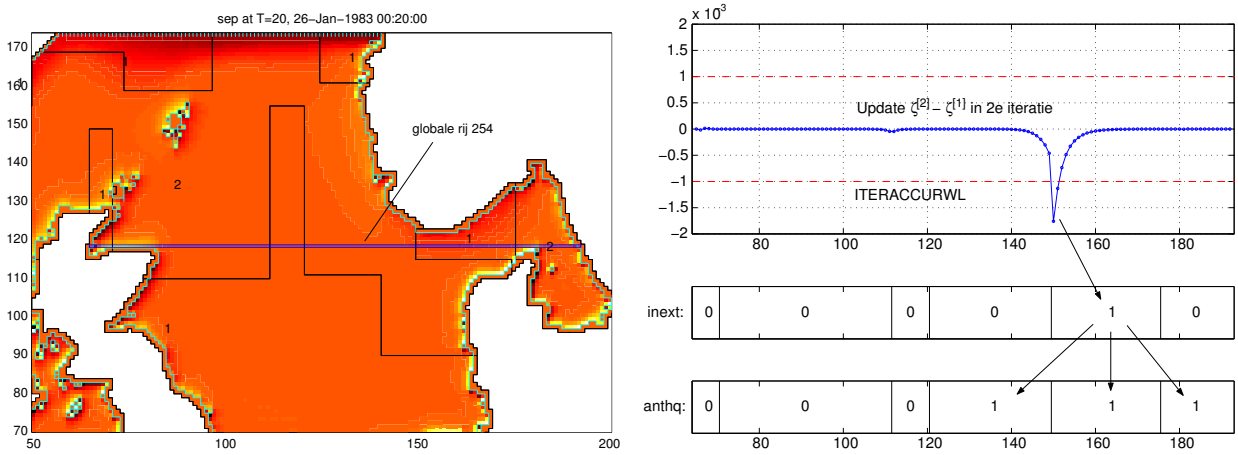
In het kader van een opdracht van Rijkswaterstaat/RIZA heeft VØRtech Computing een nieuw stopcriterium voor parallel WAQUA geprogrammeerd. Een gedeelte van de aanpassingen daarvoor wordt in het huidige project overgenomen om de droogvalcontrole in de dwarsrichting te vergemakkelijken. Dit betreft vooral het verwijderen van het complexe "minit-schema per rij". De volgende paragrafen beschrijven de achtergronden van het nieuwe criterium en de aanpassingen die aan het stopcriterium zullen worden gemaakt (3.6.1-3.6.3). Vervolgens wordt in paragraaf 3.6.4) de implementatie van de droogvalcontrole in de dwarsrichting besproken. Tenslotte wordt het gehele nieuwe droogvalalgoritme voor subroutine `wassuc` gepresenteerd in paragraaf 3.6.5.

3.6.1 Achtergrond van het nieuwe stopcriterium

Een belangrijke factor in de efficiëntie van WAQUA in vergelijking tot TRIWAQ is dat er per rij apart wordt bepaald hoeveel iteraties er nodig zijn. In veel gevallen scheelt dit de helft van het rekenwerk in het iteratieproces van subroutine `wassuc`, en dit is de meest rekenintensieve stap van WAQUA.

De huidige implementatie van het stopcriterium per rij kost echter in parallelle runs veel globale communicatie. Dit blijkt de *bottle-neck* te zijn voor goede performance op grotere aantallen (≥ 8) Linux PC's. Verder pakt de huidige methode ongunstig uit in berekeningen met domein decompositie met horizontale verfijning (DDHOR). In DDHOR-berekeningen hebben grotere gedeeltes van de verschillende domeinen interactie met elkaar, en wordt daarvoor hetzelfde aantal iteraties uitgevoerd. Opgeteld bij het feit dat er voor domeindecompositieranden toch al meer iteraties nodig zijn leidt dit tot een grote toename van de hoeveelheid rekenwerk in DDHOR-berekeningen ten opzichte van dezelfde modellen zonder domein decompositie.

In het kader van het project DDH+V, domein decompositie met horizontale èn vertikale verfijning, is een alternatief voor het huidige stopcriterium bedacht. Dit alternatief is begin 2004 in een opdracht van Rijkswaterstaat/RIZA door VØRtech Computing in een testversie



Figuur 3.3: *Illustratie van het nieuwe stopcriterium voor een globale rij van CSM8 die in zes stukken is verdeeld (zie linker figuur). In de lokale rij waar de update van de waterstanden in een iteratie groter is dan de iteratienauwkeurigheid is nog een iteratie gewenst (*`inext=1`*), in die lokale rij en zijn buren wordt er nog een iteratie uitgevoerd (*`ianthq=1`*).*

geïmplementeerd [6]. Dit alternatief bestaat uit het niet langer voor alle "deelrijen" van een "globale rij" hetzelfde aantal iteraties uitvoeren. Alleen als een deelrij het zelf wil of als een van z'n buurrijen dat wil dan wordt er nog een volgende iteratie voor een deelrij uitgevoerd, maar niet als er alleen deelrijen verderop nog niet zijn geconvergeerd. Dit principe wordt geïllustreerd in Figuur 3.3 voor een situatie uit het CSM8-model.

Met dit nieuwe stopcriterium worden de volgende doelen bereikt:

- De hoeveelheid globale communicatie wordt sterk verminderd. Dit verbetert de performance van parallel WAQUA op grotere aantallen Linux PC's.
- In domeindecompositieberekeningen worden veel meer deelrijen van elkaar ontkoppeld, waardoor er in deze simulaties een stuk minder rekenwerk wordt gedaan.
- De code wordt vereenvoudigd, voornamelijk door het overbodig maken van het "minit-schema per rij".
- De nauwkeurigheid blijft gewaarborgd, omdat de uiteindelijke oplossing nog steeds overal aan het oorspronkelijke criterium voldoet.

3.6.2 Toelichting op de iteratiemethode in subroutine `wassuc`

Het stopcriterium per rij wordt op twee plaatsen in WAQUA gebruikt: bij het oplossen van de continuïteitsvergelijking in subroutine `wassuc` en voor de transportvergelijking in subroutine `wasdfc`. We richten ons in deze beschrijving vooralsnog alleen op de eerste hiervan, omdat die in veel opzichten meer omvat dan de andere.

Subroutine `wassuc` bepaalt de oplossing ζ', u' van de continuïteitsvergelijking en u-impulsvergelijking op het halve tijdsniveau $t + \delta t/2$. Doordat er in y-richting alleen expliciete discretisaties worden gebruikt (zie [2, 3]) zijn de vergelijkingen die moeten worden opgelost alleen gekoppeld binnen rijen van het grid. Tussen de rijen hebben de onbekenden (waterstanden en u-snelheden) geen invloed op elkaar.

De continuïteitsvergelijking wordt zodanig gediscretiseerd dat er een niet-lineair stelsel moet worden opgelost. Daarom wordt het stelsel steeds gelineariseerd rond de huidige schatting voor de oplossing $\zeta^{[q-1]}$, met q het iteratieniveau. De linearisatie leidt tot tridiagonale stelsels per rekenrij die met de double sweep methode kunnen worden opgelost voor $\zeta^{[q]}$.

Een aantal speciale aspecten van de iteratiemethode in subroutine `wassuc` is nu als volgt.

1. Stopcriterium. Er wordt van oudsher in WAQUA geïtereerd totdat de verschillen in u-snelheden klein geworden zijn, $\|u^{[q]} - u^{[q-1]}\|_\infty < \epsilon_{vel}$, of totdat het maximale aantal iteraties ITERCON is bereikt. Bij de parallellisatie van WAQUA is ook als optimalisatie het stopcriterium $\|\zeta^{[q]} - \zeta^{[q-1]}\|_\infty < \epsilon_{wl}$ geïmplementeerd. Afhankelijk van de instellingen van ϵ_{vel} en ϵ_{wl} wordt hiermee exact hetzelfde aantal iteraties uitgevoerd (dus dezelfde oplossing worden bepaald), maar kan er per iteratie wat rekenwerk worden uitgespaard.

2. Droogvallen. In iedere iteratie wordt voor alle u-punten en optioneel ook waterstandspunten gecontroleerd of er nog voldoende water staat. Als dit niet zo is dan worden er extra schotjes toegevoegd. In WAQUA wordt dan de beginschatting $\zeta^{[0]}$ teruggezet en wordt het iteratieproces herstart vanaf $q = 1$. Dit betekent ook dat er opnieuw maximaal ITERCON iteraties mogen worden uitgevoerd.

3. BJ-TWGE methode. Wanneer parallel rekenen wordt gebruikt dan wordt het rekenrooster in subdomeinen verdeeld. Hiermee worden ook rijen van het globale rooster ("globale rijen") in stukken geknipt ("deelrijen"). De double sweep methode voor tridiagonale stelsels kan dan niet meer zomaar worden toegepast. Hiervoor is bij de parallellisatie van TRIWAQ de "blok-Jacobi met two-way Gaussian Elimination" methode (BJ-TWGE) bedacht [8]. Daarbij worden de deelrijen paarsgewijs gecombineerd. De twee betrokken processoren "vegen" naar elkaar toe, en communiceren in het midden wat coëfficiënten met elkaar. Hiermee wordt het tridiagonale stelsel voor de twee deelrijen samen exact opgelost. Op de buitenranden van het stelsel en ook bij domein decompositie met horizontale verfijning worden blok-Jacobi randvoorwaarden gebruikt.

4. Massa-correctie. Wanneer er in het iteratieproces blok-Jacobi randvoorwaarden worden gebruikt dan wordt het gelineariseerde stelsel voor alle subdomeinen samen niet exact opgelost. Hiermee verliest de gebruikte methode de eigenschap dat de oplossing na iedere iteratie massabehoudend is. Daarom wordt er in geval van parallel rekenen na afloop van het iteratieproces een correctiestap uitgevoerd waarmee massabehoud wordt hersteld.

5. Implementatie stopcriterium per rij. Bij gebruik van parallel rekenen moet het aantal communicaties tussen processoren zo veel mogelijk worden geminimaliseerd. Daarom

worden de rijen niet een voor een opgelost, maar wordt er voor alle rijen tegelijk geitereerd. Hierbij worden de volgende soort loops gebruikt:

```
    do 2000 ic=1,norows            ! loop over alle rijen
       if (ianthq(ic).ne.0) then   !    als rij nog niet geconvergeerd
          ...work for row ic...    !       doe berekening
       endif
200 continue
```

Array `ianthq(norows)` geeft per rij aan of er nog geen (`1`) of al wel convergentie is bereikt (`0`). De naam staat voor *"another_q_iteration"* en verwijst naar de iteratieteller q.

Iedere processor bepaalt in een array `inext(norows)` of er voor zijn eigen deelrijen aan het stopcriterium wordt voldaan of dat er droogvallen is geweest. De "black box" subroutine `wasck2` zorgt er dan voor dat deze informatie tussen de verschillende processoren wordt gecommuniceerd en dat de eigen `ianthq` hiermee wordt ingevuld.

6. Minit-schema. Om de hoeveelheid communicatie voor het evalueren van stopcriteria te verminderen, is er voor parallel TRIWAQ het zogenaamde *"minit-schema"* bedacht [8]. De essentie hiervan is dat je eerst het aantal iteraties uitvoert dat je op basis van de vorige tijdstap verwacht nodig te hebben, en pas naderhand bepaalt of je er misschien te veel hebt gedaan. In de eerste `minit-1` iteraties hoef je dan niet te controleren of convergentie is bereikt, want er worden sowieso nog een of meer iteraties uitgevoerd.

   Dit schema is uitgebreid naar het stopcriterium per rij dat in parallel WAQUA wordt gebruikt, en zit ook in subroutine `wasck2` verwerkt. Maar enerzijds is dit "minit-schema per rij" een uitermate complex verhaal, anderzijds levert het nauwelijks een besparing in de communicatietijd op.

Het uitgangspunt van de huidige implementatie van het stopcriterium per rij is dat er voor alle deelrijen die horen bij dezelfde globale rij evenveel iteraties worden gedaan. Deze eigenschap wordt bereikt door globale communicatie te gebruiken. Binnen `wasck2` wordt een array `inextg(nglrow)` gebruikt, met `nglrow` het totaal aantal globale rijen van het gehele model. Ieder subdomein vult in dit array voor een aantal rijen zijn eigen waardes in, en deze arrays per subdomein worden uitgewisseld en gecombineerd totdat iedere processor voor alle globale rijen weet of er wel of geen nieuwe iteratie voor moet worden uitgevoerd. Het performanceprobleem van parallel WAQUA zit er nu in dat het aantal boodschappen dat hiervoor moet worden gecommuniceerd minimaal toeneemt met $2 \cdot \log_2(p)$, met p het aantal processoren dat wordt gebruikt, en dat de lengte van deze boodschappen (`nglrow`) toeneemt met de grootte van het model.

3.6.3 Uitwerking van het nieuwe stopcriterium

Het idee van het nieuwe stopcriterium per rij is dat het niet nodig is om verder te itereren voor een deelrij als er in die deelrij en in de buurt daarvan niets veranderd is. Daarom hoeft

er voor een deelrij alleen dan een nieuwe iteratie te worden uitgevoerd als er in de deelrij zelf of in een van z'n directe buren nog geen convergentie is bereikt.

Dit op zichzelf simpele idee brengt het volgende te weeg:

1. Het communiceren van informatie over de convergentie per deelrij kan met de update-operatie worden bereikt. Hiervoor wordt een veld-array `inextf(mnmmaxk)` gebruikt. Ieder subdomein vult in de randpunten per rij in of er convergentie is bereikt, en weet dan na communicatie (en interpolatie in geval van DDHOR) wat de status van de directe buren is.

2. Het selectief uitschakelen van deelrijen heeft invloed op de implementatie van BJ-TWGE. Alleen als er voor beide deelrijen waarvoor TWGE zou worden gebruikt nog een iteratie wordt uitgevoerd dan kan TWGE worden toegepast. Als een van beiden niet meer meedoet dan wordt een blok Jacobi randvoorwaarde gebruikt. Voor welke buurrijen er nog een iteratie wordt uitgevoerd moet hiervoor worden opgeslagen. Daartoe wordt array `iaqngb(2,norows)` geïntroduceerd.

3. Het alleen met directe buren communiceren zorgt ervoor dat een subdomein het niet weet als er verderop iets droogvalt. Daarom kan de beginschatting $\zeta^{[0]}$ niet in de hele globale rij worden teruggezet. Het lijkt ons niet gewenst om alleen in een gedeelte van de deelrijen het iteratieproces te resetten. Dit is een extra motivatie voor het kiezen van de aanpak van TRIWAQ bij punt 4 van paragraaf 3.5.

4. Na de communicatie van `inextf` in subroutine `wasck2` is er nog steeds globale communicatie nodig om te bepalen of er *überhaupt* nog een nieuwe iteratie moet worden opgestart. Alle subdomeinen moeten in de pas met elkaar blijven lopen, en moeten het daarvoor met elkaar eens zijn over het totaal aantal iteraties dat wordt uitgevoerd.

   Voor deze globale communicatie kan goed de bestaande subroutine `waschk` worden aangeroepen vanuit `wasck2`. Hiermee wordt het grootste residu en de lokatie daarvan uitgewisseld tussen alle processoren, wordt het globale minit-schema geïmplementeerd, en wordt gecontroleerd of het iteratieproces niet stagneert of in een cycle terecht is gekomen.

Omdat de aanpassing van het stopcriterium in principe een ingrijpende aangelegenheid is moet hier zorgvuldig mee worden omgegaan. Daarom is het niet gewenst dat het nieuwe criterium in het huidige project in WAQUA wordt geïmplementeerd. Om het huidige project te vergemakkelijken worden wel de nodige voorbereidingen hiervoor gemaakt:

- Het minit-schema per rij wordt verwijderd uit WAQUA. Dit betreft het verwijderen van enkele arrays uit `intgda`, van het aanmaken en initialisatie hiervan in `waspif`, van het doorgeven aan de rekenroutines `wassuc`, `wasdfc` en onderliggende routines, en het verwijderen van het gebruik hiervan in `wasck2`.

  Uitschakelen van dit schema betekent dat het huidige stopcriterium preciezer wordt nageleefd. Nu wordt er soms voor een rij een iteratie teveel gedaan omdat daarmee

mogelijk communicatie kan worden uitgespaard. Deze aanpassing heeft wel effect op de rekenresultaten, maar de validiteit van de aanpassing kan hiermee niet ter discussie worden gesteld.

- Nieuwe arrays voor het nieuwe stopcriterium worden toegevoegd: `iaqngb(2,norows)`, `inextf(lenhor)`. Dit zijn werkarrays die in de koproutines `wasspu/v`, `wastru/v` worden aangemaakt en doorgegeven aan `wassuc` en `wasdfc`. In deze rekenroutines worden de arrays correct geïnitialiseerd en bijgewerkt en doorgegeven naar onderliggende routines zoals `wastrd` voor het BJ-TWGE schema en `wasck2` voor het stopcriterium.

- Subroutine `wasck2` blijft het oude stopcriterium implementeren. Alleen wordt het minitschema per rij eruit verwijderd en wordt het vullen van array `iaqngb` eraan toegevoegd. Hiermee wordt de header gelijk gemaakt aan de versie van `wasck2` voor het nieuwe criterium. Daarmee wordt het gemakkelijk om verschillende versies van de programmatuur te onderhouden en kan ook met het nieuwe criterium worden geëxperimenteerd.

3.6.4 Implementatie van de droogvalcontrole in dwarsrichting in subroutine `wasdry`

De grootste moeilijkheid bij het implementeren van een droogvalcontrole in de dwarsrichting betreft het weer actief maken van rijen die eerder al waren uitgeconvergeerd. Verder moet er in parallelle en domein decompositie berekeningen over de nieuwe schotjes in de dwarsrichting worden gecommuniceerd. Dit laatste is sterk gerelateerd aan de uitwerking van subroutine `trsdry` van TRIWAQ in paragraaf 3.4.4.

De methode die voor deze twee punten is bedacht bestaat uit het in een werkarray `idryv` markeren van v-punten die worden drooggezet, communiceren van het werkarray, en in een loop over alle rijen zetten van schotjes in de dwarsrichting en resetten van de convergentievlag per rij. De convergentievlaggen per rij kunnen zowel met het oude als het nieuwe stopcriterium naar de buurprocessen worden gecommuniceerd. Dit algoritme heeft sterke overeenkomst met wat er momenteel vanwege domein decompositie ook voor de rekenrichting wordt gebruikt. De methode wordt verder uitgewerkt in Algoritme 6.

Een kleine complicatie in dit algoritme is dat de update in twee aparte calls van `cocupd` moet worden verdeeld. Dat lijkt sowieso nodig te worden omdat momenteel in subroutine `wasdry` ook het array `hu` wordt gecommuniceerd. Welke arrays er moeten worden gecommuniceerd en welke stencils hierbij nodig zijn zal later heel precies worden uitgezocht.

Met dit algoritme wordt de efficiëntie van WAQUA gewaarborgd: wanneer er droogvallen van v-punten optreedt dan wordt alleen voor de direct daarbij betrokken rijen extra iteraties uitgevoerd. Verder leidt dit algoritme wat ons betreft tot een code die goed te begrijpen is. Een vergelijkbare methode wordt al gebruikt voor het zetten van schotjes op DDHOR-interfaces vanwege de droogvalcontrole in waterstandspunten (maskerarray `idrywl` in subroutine `wasdry`, zie ook paragraaf 3.4.4). Bovendien blijft het hele algoritme gelokaliseerd (verstopt) binnen subroutine `wasdry`.

Algorithm 6 - Ontwerp van de droogvalcontrole in dwarsrichting voor subroutine `wasdry` van WAQUA

for all points `nm=jstart to mnmaxj` **do**
  `idryv(nm)=0`
end for
for computational columns `ic=1 to nocols` **do**
  check drying of v-points `nf`,...,`nl`, mark in `kfv`, `qy`, `vh`, `idryv`
end for
update: `call cocupd(idrywl, 'fullbox_s', 'stc1', 'max_meth_s',`
`idryv, 'fullbox_v', 'stcvs', 'max_meth_uv',`
`kfu, 'fullbox_u', 'stcus', 'max_meth_uv',`
`kfv, 'fullbox_v', 'stcvs', 'max_meth_uv',`
`qy, 'fullbox_v', 'stcvs*ddh', 'q_distr')`
for computational rows `ic=1 to norows` **do**
  for all wl-points `m=mf to mlu` **do**
    check whether v-points above/below cell `m` have dried (using `idryv`)
    if so: set `inext(ic)=2` (another iteration required)
  end for
end for

3.6.5 Uitwerking van het droogvalalgoritme voor subroutine `wassuc`

Het nieuwe algoritme voor heel subroutine `wassuc` wordt in Algoritme 7 verder uitgewerkt. Hierin zijn diverse extra aanpassingen aan WAQUA verwerkt:

1. In WAQUA worden de waterstanden en snelheden op hele en halve tijdstippen op een andere manier opgeslagen. In plaats van aparte arrays `seh` en `sep` zoals in TRIWAQ gebruikt WAQUA arrays `s` en `seold`. In de koproutines `wasspu` en `wasspv` wordt er met deze arrays gemanipuleerd.

   Vanuit het oogpunt van consistentie is het gewenst dat dezelfde namen als in TRIWAQ worden gebruikt, zie ook activiteit 2-4. Verder is het uitvoeren van rekenwerk in koproutines ongewenst. Daarom worden de namen `seh` en `sep`, `uh` en `up` en `vh`, `vp` ook in `wassuc` geïntroduceerd. Een van de eerste stappen betreft dan het kopiëren van de waardes van de vorige halve tijdstap, ofwel het zetten van de beginschatting voor de waardes van de huidige halve stap.

2. In WAQUA worden momenteel de doorstroomhoogtes maar een keer per tijdstap uitgerekend: namelijk bij het oplossen van de continuïteitsvergelijking. Om de droogvalcontroles op dezelfde manier als in TRIWAQ te kunnen doen moeten de doorstroomhoogtes ook in de andere halve tijdstap worden bepaald. Hiervoor worden de bestaande arrays `hu` en `hv` hernoemd naar `hkuh` en `hkvp`, en worden er daarnaast nieuwe arrays `hkup` en `hkvh` geïntroduceerd. Met de naamgeving hiervan wordt gelijk aangesloten op de

overeenkomstige werkarrays die in TRIWAQ worden gebruikt. Omdat de arrays van de continuïteitsvergelijking het meest betekenisvol zijn en vanwege achterwaartse compatibiliteit zullen de arrays `hkuh` en `hkvp` als HU en HV in SOLUTION_FLOW op de SDS-file worden gezet.

3. Een klein puntje is dat de berekening van `h0i` iets kan worden versneld. Omdat `hu` ten behoeve van domein decompositie altijd op minimaal `0.499`*DEPCRIT wordt gesteld is hier geen maximum met `htrsh` vereist.

3.6.6 Testen

Naast de gebruikelijke aanpak om de testbank te gebruiken voor het bepalen van een geschikt testmodel is hier meer mogelijk. Het oorspronkelijke probleem van negatieve volumina is gemakkelijk te reproduceren, en hiermee kan goed worden gedemonstreerd dat het in de nieuwe versie verholpen is. Hiervoor zal het Kustzuid of het Scalwest model worden gebruikt.

3.6.7 Documentatie

De droogvalalgoritmes van WAQUA worden beschreven in [2, Hfdst 8] en zal worden aangepast. Verder moet de data-analyse van WAQUA in [7] worden bijgewerkt.

3.7 Activiteit 3-3: Verwijderen van een verschil met Delft3D-FLOW

In het droogvalrapport [9, pag.3-26] is geconstateerd er verschillen zijn tussen de droogvalcontroles in TRIWAQ en Delft3D-FLOW. In TRIWAQ worden de controles in u-punten en wl-punten in iedere iteratie uitgevoerd, wordt de grenswaarde $\delta/2$ gebruikt, en worden er in geval van droogvallen een of twee schotjes gezet. Na afloop van het iteratieproces wordt dezelfde controle voor de dwarsrichting uitgevoerd. In Delft3D-FLOW wordt in het iteratieproces alleen in u-punten gecontroleerd, op grenswaarde 0. Na afloop van het iteratieproces wordt er in wl-punten gecontroleerd, op grenswaarde 0, en worden er in geval van droogvallen gelijk vier schotjes gezet. Tenslotte wordt er in Delft3D-FLOW aan het begin van de volgende halve stap wederom in u-punten gecontroleerd, maar nu op grenswaarde $\delta/2$.

In het droogvalalgoritme van [9, par.5.1] is gekozen om de methode van Delft3D-FLOW te gaan gebruiken voor wat betreft de grenswaarde van 0 in plaats van $\delta/2$. De motivatie daarvoor is dat controleren op 0 tot minder herstarten van het iteratieproces leidt. Deze afstemming op Delft3D-FLOW is als activiteit 3-3 opgenomen in het werkplan voor het huidige project.

Inmiddels heeft WL|Delft Hydraulics echter haar implementatie aangepast. De controle in u-punten in het iteratieproces wordt nu ook op $\delta/2$ gedaan. Dit is gedaan vanwege een inconsistentie die door de controle aan het begin van de volgende halve tijdstap werd geintroduceerd. Daarbij werden wel nieuwe schotjes gezet maar werden de bijbehorende debieten

Algorithm 7 - Ontwerp van het nieuwe droogvalalgoritme in subroutine `wassuc` van WAQUA

initialisaties voor barriers
initialiseer $\texttt{seh}^{[0]}, \texttt{uh}^{[0]}, \texttt{kfuh}^{[0]}, \texttt{kfvh}^{[0]}$
bereken $\texttt{tetau}, \texttt{hkuh}^{[0]}, \texttt{tetav}, \texttt{hkvh}^{[0]}$ (upwindschema) (WASSHU)
controleer op droogvallen en onderlopen
→ droogvallen in x-richting op $\delta/2$ wordt momenteel alleen voor randpunten gedaan, wordt overbodig
→ onderlopen in x-richting als $\texttt{hkuh}^{[0]} \geq \delta$ *en* $\max(\texttt{sep}_m, \texttt{sep}_{m+1}) + \min(\texttt{dps}_m, \texttt{dps}_{m+1}) \geq \delta$
→ dit verandert $\texttt{kfuh}^{[0]}$ en laat `kfup` ongemoeid
bereken gediscretiseerde impulsvergelijking (`kfup` → `aa`, `cc`, `dd`)
bereken preconditionering van vergelijkingen op DDHOR-interfaces
bereken vaste deel continuïteitsvergelijking (`qy` → `e`)
initialiseer arrays `ianthq` en `iaqngb` voor het stopcriterium-per-rij
for iteratie $q = 1, 2, \ldots$ (maximaal ITERCON) **do**
bereken nieuwe impulsvergelijking voor barrierpunten (WASSBC)
preconditioneren voor DDHOR, randvoorwaarden wl-punten (WASRV2)
opstellen en oplossen tri-diagonale stelsels voor waterstanden $\texttt{seh}^{[q]}$ (WASTRD)
bereken `qx` in DDHOR-randpunten, communiceer $\texttt{seh}^{[q]}$
bereken $\texttt{hkuh}^{[q]}$, $\texttt{hkvh}^{[q]}$
droogvalcontroles in u-punten, v-punten en wl-punten (WASDRY)
→ in u-punten op $\delta/2$, aanpassing $\texttt{kfuh}^{[q]}$
→ in v-punten op $\delta/2$, aanpassing $\texttt{kfvh}^{[q]}$, `qy` en `vh`
→ in wl-punten op $\delta/2$, aanpassing $\texttt{kfuh}^{[q]}$ *en* $\texttt{kfvh}^{[q]}, \texttt{qy}, \texttt{vh}$
→ inclusief aanpassing van `e` m.b.t. wijzigingen `qy`
i.g.v. convergentiecontrole op snelheden: bereken $\texttt{uh}^{[q]}, \texttt{qx}^{[q]}$
communiceer t.b.v. stopcriterium per rij (WASCK2)
update dischargerandvoorwaarden
end for
i.g.v. convergentiecontrole op waterstanden: bereken $\texttt{uh}^{[q]}, \texttt{qx}^{[q]}$
communiceer $\texttt{qx}^{[q]}$
voer de massacorrectiestap uit, incl. berekening `hkuh`
controleer op stabiliteit
herbereken de barrierarrays
bepaal maskerarray `khs` en ververs coëfficiëntenset t.b.v. horizontale interpolaties

niet aangepast. Dit leidde tot moeilijkheden in de transportsolver: daar konden de schotjes niet zondermeer worden vertrouwd, maar moest ook op de waarde van de debieten worden gecontroleerd. Dit soort problemen met inconsistenties tussen verschillende waardes wordt met het door ons ontworpen algoritme van paragraaf 3.4.2 zo veel mogelijk omzeild.

In Delft3D-FLOW wordt de grenswaarde van 0 nog wel in de controle in *wl*-punten gebruikt. Dat zou ook in het huidige nieuwe algoritme kunnen worden verwerkt. Een probleem hiermee is wel dat er roostercellen met extreem dunne laagjes water kunnen ontstaan. De rekenmethodes voor het stoftransport en turbulentiemodel moeten daarop zijn voorbereid. Dat is in Delft3D-FLOW gedaan, maar is niet triviaal. Het controleren op 0 wordt daarom in het huidige project niet in WAQUA/TRIWAQ geïmplementeerd.

Een idee om in de toekomst met deze optie te kunnen spelen is als volgt. Er zou naast DEPCRIT nog een tweede parameter DEPCRIT_WL kunnen worden geïntroduceerd. In de huidige situatie is deze steeds identiek aan DEPCRIT, in Delft3D-FLOW is deze permanent ingesteld op 0. Door haar instelbaar te maken kan er met verschillende algoritmes worden gespeeld en kan de robuustheid van verschillende methodes worden onderzocht.

3.8 Activiteit 3-1: Flux-limiter in transport

3.8.1 Wiskundige methoden en vergelijkingen

Activiteit 3-1 (begrenzing H in diffusie/anti-creep) staat in dit project min of meer op zichzelf. Zoals in onze offerte is vermeld hebben we via een uitbreiding van de Kalmanfilteringprogrammatuur uitgebreide ervaring opgedaan met de flux-limiter voor het transportgedeelte van WAQUA. De gevoeligheid van de berekende concentraties bleek funest voor het RRSQRT Kalman filter, en blijkt na verfijning van het mechanisme goed met de fluxlimiter te kunnen worden aangepakt. Hetzelfde mechanisme lijkt toepasbaar op het anti-creep gedeelte van TRIWAQ.

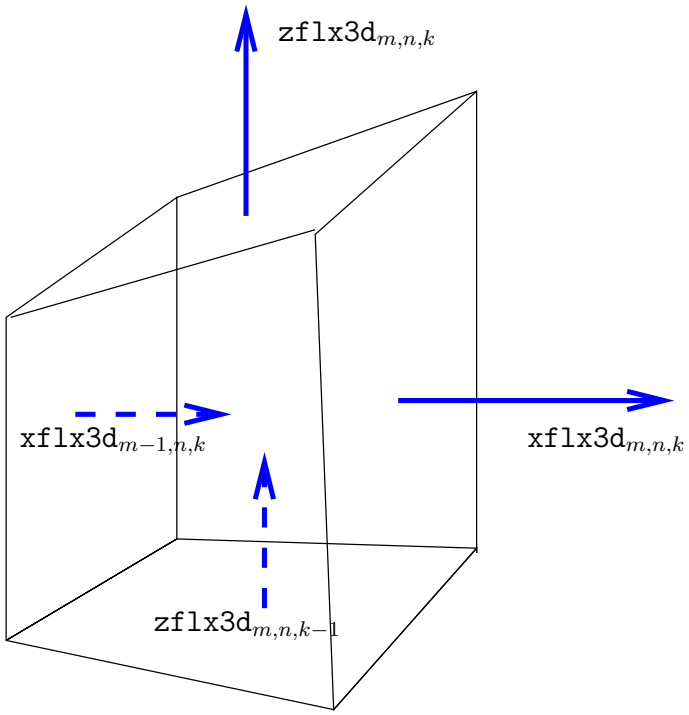
Tijdens het werk aan de Kalman-programmatuur is een aanpassing aan de routine `wasdfc` ontwikkeld die de resultaten van de transportberekening veel gladder doet verlopen. Deze aanpassing betreft het expliciete deel van het transport. Dit transport is diffusief en advectief van aard. Bij gebruik van maar één laag is er bovendien een term afkomstig van bodemwrijving.

De aanpassing is in de volgende stappen geconstrueerd:

1. De genoemde 'spikkels' bestaan uit grote pieken in de berekende velden van zoutconcentraties.

2. De verandering in de concentraties wordt beschreven door de transportvergelijking. Deze vergelijking verklaart de verandering van de hoeveelheid getransporteerde stof in een *controlevolume* uit de transportfluxen door de *cellfaces* van het controlevolume, zoals in Figuur 3.4 wordt geïllustreerd.

   In eerdere analyses is gebleken dat de pieken vooral ontstaan in roosterpunten waar maar weinig water staat. Een mogelijk mechanisme is, dat de de doorstroomhoogte in het snelheidspunt, waar de transportflux wordt berekend, veel groter is dan in het



Figuur 3.4: *De getransporteerde stof wordt door* cellfaces *in- en uit het* controlevolume *getransporteerd. Fluxen in y-richting zijn uit de figuur weggelaten.*

waterstandspunt, waarin de getransporteerde stof zich bevindt. Door de celfaces kan daardoor een relatief grote hoeveelheid stof worden getransporteerd.

In de huidige activiteit worden transportfluxen daarom alleen maar aangepast in snelheidspunten waarin de doorstroomhoogte veel groter is dan de doorstroomhoogte in het aangrenzende waterstandspunt. Er is voor gekozen om de grens te leggen bij een factor 3: er wordt alleen ingegrepen wanneer de doorstroomhoogte in het snelheidspunt minstens 3 maal zo groot is als de doorstroomhoogte in het waterstandspunt.

3. In de berekening wordt onderscheid gemaakt tussen *expliciete* en *impliciete* fluxen.

   De waarden van de expliciete transportfluxen wordt berekend aan het begin van de berekening in `wasdfc/trsdif`. De waarde van de impliciete fluxen wordt pas gaandeweg de berekening in `waspnd/trsjac` bekend.

   In eerdere analyses gebleken dat pieken in het concentratieveld meestal worden veroorzaakt door de expliciete fluxen.

   Een aanpassing aan de expliciete fluxen kan dus veel eenvoudiger worden doorgevoerd dan een aanpassing aan de impliciete fluxen.

4. De expliciete transportfluxen bestaan uit een combinatie van advectieve en diffusieve fluxen. Bekend is dat expliciete berekening van diffusie kan leiden tot over- en undershoots. Een van de maatregelen die soms worden genomen is dan ook het uitschakelen van de diffusie.

   Ook de advectieve termen kunnen over- en undershoots veroorzaken, omdat centrale differenties worden gebruikt. Zelfs bij gebruik van upwind differenties kunnen over- en undershoots optreden, wanneer het advectieve CFL-getal groter is dan 1. In de huidige aanpak wordt daarom geen gebruik gemaakt van upwind-discretisatie, maar van een discretisatie die puur uitgaat van de toestand in het roosterpunt waar problemen lijken op te treden. Ook deze aanpak heeft een aantal nadelen, die in punt 7 hieronder worden besproken.

5. Om een geschikte formule te verkrijgen voor de limiter te krijgen, wordt de transportvergelijking gecombineerd met de continuïteitsvergelijking. Hiervoor wordt deze laatste eerst vermenigvuldigd met de oude waarde van de concentratie `rp`. Het resultaat wordt afgetrokken van de transportvergelijking. Dat levert voor WAQUA de volgende vergelijking op:

$$\begin{aligned} V'_{m,n}(\mathtt{rp}'_{m,n} - \mathtt{rp}_{m,n}) &= \Delta t/2((\mathtt{xflx2d}_{m-1,n} - \mathtt{qx}_{m-1,n}\mathtt{rp}_{m,n}) - \\ &\qquad (\mathtt{xflx2d}_{m,n} - \mathtt{qx}_{m,n}\mathtt{rp}_{m,n})) \\ &+ \Delta t/2((\mathtt{yflx2d}_{m,n-1} - \mathtt{qy}_{m,n-1}\mathtt{rp}_{m,n}) - \\ &\qquad (\mathtt{yflx2d}_{m,n} - \mathtt{qy}_{m,n}\mathtt{rp}_{m,n})). \end{aligned}$$

Voor TRIWAQ wordt de volgende vergelijking gevonden:

$$\begin{aligned} V'_{m,n,k}(\mathtt{rp}'_{m,n,k} - \mathtt{rp}_{m,n,k}) &= \Delta t/2((\mathtt{xflx3d}_{m-1,n,k} - \mathtt{qxk}_{m-1,n,k}\mathtt{rp}_{m,n,k}) - \\ &\qquad (\mathtt{xflx3d}_{m,n,k} - \mathtt{qxk}_{m,n,k}\mathtt{rp}_{m,n,k})) \\ &+ \Delta t/2((\mathtt{yflx3d}_{m,n-1,k} - \mathtt{qyk}_{m,n-1,k}\mathtt{rp}_{m,n,k}) - \\ &\qquad (\mathtt{yflx3d}_{m,n,k} - \mathtt{qyk}_{m,n,k}\mathtt{rp}_{m,n,k})) \\ &+ \Delta t/2((\mathtt{zflx3d}_{m,n,k-1} - \mathtt{qzk}_{m,n,k-1}\mathtt{rp}_{m,n,k}) - \\ &\qquad (\mathtt{zflx3d}_{m,n,k} - \mathtt{qzk}_{m,n,k}\mathtt{rp}_{m,n,k})). \end{aligned}$$

Merk op dat in deze vergelijking de nieuwe inhoud V' wordt gecombineerd met de oude concentratie `rp`.

6. In de bovenstaande vergelijkingen wordt de verandering in de concentratie gegeven als de som van 4 (WAQUA) of 6 (TRIWAQ) termen, die ieder het verschil zijn tussen de transportflux en het product van het debiet en de concentratie.

   In het aangepaste schema worden de transportfluxen vervangen door waarden die dicht bij het product van het debiet en de concentratie ligt. De aangepaste flux mag dan nog hoogstens 10% verandering in de concentratie veroorzaken. Zo moet voor de flux $\mathtt{xflx3d}_{m-1,n}$ bijvoorbeeld gelden:

$$|\mathtt{xflx2d}_{m-1,n} - \mathtt{qx}_{m-1,n}\mathtt{rp}_{m,n}| \leq 0.1\frac{V'_{m,n}\mathtt{rp}_{m,n}}{\Delta t/2}. \tag{3.1}$$

   Wanneer de berekende flux daaraan niet voldoet, dan wordt deze vervangen door:

$$\mathtt{xflx2d}_{m-1,n} = \mathtt{qx}_{m-1,n}\mathtt{rp}_{m,n} \pm 0.1\frac{V'_{m,n}\mathtt{rp}_{m,n}}{\Delta t/2}. \tag{3.2}$$

   Het teken $\pm$ wordt zodanig gekozen, dat de aanpassing zo klein mogelijk is (+ wanneer de flux te groot is, en - wanneer de flux te klein is).

7. De hier beschreven aanpak heeft een aantal nadelen die bekend zijn geworden tijdens het testen van de aanpassing:

   - Van elke transportflux wordt geëist dat deze (voldoende) dicht in de buurt liggen van twee verschillende waarden. Immers, de transportflux veroorzaakt een verandering in de concentratie in beide aangrenzende controlevolumes. Deze eis kan dus niet altijd en zeer streng worden afgedwongen, omdat er dan conflicten ontstaan.
   - De aangepaste fluxen worden niet verkregen uit een consistente benadering van de advectieve en diffusieve transporten die voorkomen in de transportvergelijkingen. Voldoende nauwkeurigheid kan dus alleen worden verkregen wanneer de aangepaste fluxen slechts zeer zelden worden gebruikt.

De aangepaste fluxen worden daarom alleen gebruikt in bijzondere gevallen, zoals in punt 3 is beschreven.

Verbeterde manieren van flux-aanpassing kunnen wellicht worden verkregen door in plaats van te eisen dat de concentratie maar weinig verandert te eisen dat de nieuwe concentratie een waarde heeft tussen de oude waarden in de omliggende roosterpunten.

Dit vergt nog enig uitzoekwerk.

De hierboven beschreven methode wordt schematisch weergegeven in Algoritme 8. Er wordt eerst gecontroleerd of de eerste situatie zich voordoet. Als de doorstroomhoogte op de randen van het controlevolume niet veel groter is dan de gemiddelde waterdiepte in de cel, is er geen aanpassing nodig in de expliciete tansportflux.

Vervolgens wordt er gekeken of de expliciete transporttermen gezamenlijk inderdaad een grote hoeveelheid transporteren, in verhouding tot wat er overblijft. Is dat niet het geval dan is er geen aanpassing nodig in de expliciete tansportflux. Is dat echter wel het geval, dan wordt de expliciete transportterm zodanig gemiddeld met de eerder beschreven, "onschuldige" advectieterm, dat er geen grote pieken worden veroorzaakt in het transportveld.

Het probleem van de pieken kan worden uitgelegd als dat de tijdstap te groot is voor een gedeelte van het ADI-schema dat wordt uitgevoerd. Met de flux-limiter wordt in dit geval effectief de diffusiecoëfficiënt teruggeschroefd. Verder is de aanpassing specifiek voor de MEAN-aanpak voor de berekening van dieptes van controlevolumes. Wanneer de MAX-aanpak wordt gebruikt dan is de verhouding `hv/(s+dps)` namelijk automatisch al beperkt.

Dezelfde methode is van toepassing voor de transporttermen in TRIWAQ, inclusief de expliciete anti-creeptermen van subroutine `trscre`. Het grote voordeel van deze methode ten opzichte van het limiten van bijvoorbeeld de laagdiktes `zkvp` is dat deze methode in een keer voor alle expliciete fluxen werkt, zonder dat heel precies moet worden uitgezocht welke factoren in welke termen voor pieken of instabiliteiten verantwoordelijk zijn.

De implementatie gebeurt hier via het introduceren van een werkarray waarin de expliciete fluxen tijdelijk worden bewaard. De aanpassingen betreffen de loops 2400, 2750 in `trsdif` en heel subroutine `trscre`. Na loop 2750 wordt een extra loop geïntroduceerd waarin de flux-limiter wordt berekend en de expliciete transporttermen worden toegepast. Omdat er geen TRIWAQ-simulatie (meer) gevonden kon worden waarin onrealistische concentraties een probleem vormden, is de limiter voorlopig niet-actief gemaakt.

3.8.2 Testen

De pieken in de resultaten van het transportgedeelte van WAQUA treden in ieder geval op in het Scalwestmodel maar mogelijk ook in het model Kustzuid. Een van beiden zal worden gebruikt om te laten zien dat de pieken door de limiter worden verkleind. Ook zal er enigszins met de instellingen `h2hfac` en `drmax` uit Algoritme 8 worden gevariëerd.

In geval van TRIWAQ zal er kort naar pieken in het transportgedeelte worden gekeken, maar zijn instabiliteiten door de anti-creeptermen van groter belang. Deze zijn in het verleden opgetreden in het bovenstroomse gedeelte van de riviersecties van RYMAMO en het Zeedelta-model, wanneer hierin veel lagen werden gebruikt. Mede op basis hiervan zal een geschikt

Algorithm 8 - Ontwerp van een flux-limiter voor `wasdfc` en `trsdif`

Instellingen voor de limiter:
`parameter (h2hfac=3.0)`
maximale verhouding tussen doorstroomhoogten
`parameter (drmax = 0.10)`
maximale relatieve verandering van de concentratie t.g.v. een flux in een ondiep punt

for alle v-punten **do**

bereken expliciete flux `expflx` zonder limiter

for beide naastgelegen waterstandspunten **do**

if ($|\texttt{hv}| \leq \texttt{h2hfac} * |\texttt{s} + \texttt{dps}|$) **then**

doorstroomhoogte in v-punt is niet veel groter dan in wl-punt: limiter is niet nodig

else : doorstroomhoogte in v-punt is veel groter dan in wl-punt

bereken de concentratie-veranderende flux `dr`:
`advflx` = advectie-flux m.b.v. upwind concentraties
`dr` = `advflx-expflx`

bereken de hoeveelheid stof bij ongewijzigde concentraties:
$\texttt{rprest} = \texttt{gsqs} * (\texttt{s} + \texttt{dps}) * \texttt{rp}$

if ($|\texttt{dr}| \leq \texttt{drmax} * \texttt{rprest}$) **then**

expliciete flux veroorzaakt geen overshoots: limiter is niet nodig

else : expliciete flux veroorzaakt overshoots:

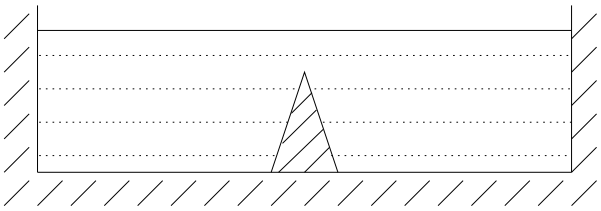
pas de limiter toe:
$\texttt{expflx} = \texttt{advflx} + \texttt{sign}(\texttt{dr}, \texttt{drmax} * \texttt{rprest})$

end if

end if

end for

end for



Figuur 3.5: *Testmodel voor de transportsolver. Basin met stilstaand water met een vertikale zoutgradiënt. Door de verhoging in de bodem worden de laaginterfaces gekromd, waardoor er bij de implementatie van de diffusie-operator speciale maatregelen zoals de anti-creeptermen nodig zijn.*

testmodel worden gezocht. Gedacht wordt aan het Waalmodel, of een andere uitsnede van het Zeedeltamodel.

Tenslotte zullen we kijken naar het functioneren van de anti-creeptermen als geheel. Hiervoor wordt een testmodel gebruikt van een basin met stilstaand water met een vertikale zoutgradient dat door een verhoging in de bodem in het midden in twee gebieden wordt verdeeld, zie Figuur 3.5. Door de verhoging van de bodem worden de laaginterfaces in het model gekromd. Wanneer de diffusie-operator dan langs σ-vlakken wordt gediscretiseerd dan wordt er water met verschillende zoutconcentraties gemengd en komt er een stroming op gang. De anti-creeptermen zouden dit moeten tegengaan, net als de aanpak van WL|Delft Hydraulics waarbij de diffusie-operator langs strikt horizontale vlakken wordt gediscretiseerd.

3.8.3 Documentatie

De discretisatie van de transportvergelijking wordt beschreven in paragraaf 4.5 van de technische documentatie van TRIWAQ [12], en in paragraaf 4.3 van de technische documentatie van WAQUA [2]. Aangezien er in deze activiteit aan de numerieke formuleringen gesleuteld wordt lijkt het ons gewenst wanneer dit in de documentatie wordt verwerkt.

3.9 Activiteit 3-6: Regularisatie van discrete stelsels in `wassuc`

3.9.1 Wiskundige methoden en vergelijkingen

Bij het oplossen van de gekoppelde impuls- en continuïteitsvergelijking in de routine `wassuc` treedt in sommige gevallen een grote gevoeligheid op in de oplossingen. Eén van de oorzaken is de discretisatie van de impulsvergelijking. Deze wordt benaderd met behulp van een lineaire vergelijking waarvan de diagonaalelementen soms zeer kleine of zelfs negatieve waarden kunnen aannemen. Dit probleem is voor het eerst beschreven in [5]. Nauwkeurige bestudering van het probleem wijst uit dat het niet mogelijk is om de conditie van het stelsel te verbeteren door upwinding toe te passen. Wel is het mogelijk om deze conditie te verbeteren door, in

sommige roosterpunten, de advectieve term (gedeeltelijk) expliciet te berekenen. De volgende twee mogelijkheden zijn hiervoor bestudeerd:

1. Wanneer de advectieterm `adx` negatief is dan wordt ze gedeeltelijk expliciet berekend (`dd+=fac*adx*up`) om te zorgen dat de diagonaal `bb` boven een ingestelde waarde blijft.

   Deze strategie wordt geconfigureerd met behulp van de parameter `bbmin`, bijvoorbeeld `parameter (bbmin = 0.15)`. Met `bbmin=-9999.` wordt het huidige schema ingesteld.

2. Wanneer `bb` te klein dreigt te worden dan nemen we aan dat de waarde van de advectieterm `adx` onrealistisch is. Een gedeelte van deze term wordt dan weggegooid, en `bb` wordt op de minimale waarde gesteld.

In de testen is gebleken dat de tweede manier de gevoeligheid van de uitkomsten het meest effectief onderdrukt. De aanpassing treedt in zeer weinig roosterpunten in werking. Het verwaarlozen van een gedeelte van de advectieve term in deze roosterpunten zal daarom geen negatief effect hebben op de nauwkeurigheid van de discretisatie, temeer omdat de discretisatie in de genoemde punten toch al niet zeer nauwkeurig kan zijn geweest. Een volgende aanpassing bestaat uit een lokale aanpassing van het advectie-schema van WAQUA op plekken waar de stroming van teken wisselt. In plaats van het centrale differentieschema

$$u\frac{\partial u}{\partial x} \approx u_m \frac{u_{m+1} - u_{m-1}}{2\delta x} \tag{3.3}$$

wordt het volgende alternatief gebruikt:

$$u_m > 0 \quad : \quad u\frac{\partial u}{\partial x} \approx u_m \frac{\max(0., u_{m+1}) - u_{m-1}}{2\delta x} \tag{3.4}$$

$$u_m < 0 \quad : \quad u\frac{\partial u}{\partial x} \approx u_m \frac{u_{m+1} - \min(0., u_{m-1})}{2\delta x} \tag{3.5}$$

Deze formules zouden in alle roosterpunten kunnen worden toegepast, en betekenen dan alleen een verandering ten opzichte van het oude schema wanneer $u_m > 0,\ u_{m+1} < 0$, of wanneer $u_m < 0,\ u_{m-1} > 0$, dus wanneer de snelheid u van teken wisselt in de benedenstroomse (downwind) richting.

Vooralsnog worden de formules alleen toegepast waar $u_m \cdot u_{m\pm1} < -0.2$. Maar er kunnen ook fysische overwegingen worden gegeven waarom deze of een vergelijkbare aanpassing overal zou moeten worden toegepast. Ook kunnen er andere verbeteringen aan het advectie-schema worden gemaakt. Dit verder uitwerken van het advectie-schema lijkt ons een nuttig onderzoek maar valt buiten de scope van het huidige project.

3.9.2 Testen

De strategieën zullen worden vergeleken via experimenten met het Grensmaasmodel, met de versie hiervan waarin het referentieniveau 42 m verschoven is. Daarbij zal de gevoeligheid van het model als belangrijke maatstaf worden gebruikt. Daarnaast moeten de verschillen met de uitgangsversie van WAQUA zo veel mogelijk worden beperkt.

3.9.3 Documentatie

De tijdsdiscretisatie van de impulsvergelijking wordt beschreven in paragraaf 5.1 van de technische documentatie van WAQUA [2]. Deze paragraaf behoeft een kleine aanpassing wanneer de advectieve term soms (gedeeltelijk) expliciet verrekend of verwaarloosd wordt.

De ruimtediscretisatie van de impulsvergelijking wordt beschreven in paragraaf 4.2.1 van de technische documentatie van WAQUA [2]. Deze paragraaf behoeft een kleine aanpassing wanneer de advectieve term soms (gedeeltelijk) wordt berekend met upwind-differenties.

Hoofdstuk 4

Overzicht van belangrijkste keuzes en opties

In dit rapport is een groot aantal keuzes voorgesteld cq. gemaakt. Ook worden er op diverse plekken werkzaamheden vermeld die vooralsnog niet zullen worden uitgevoerd in dit project. In dit hoofdstuk geven we hier een overzicht van. De meeste van deze keuzes zijn besproken en vastgelegd bij de bespreking van het detail ontwerp (versie 0.95) op 12 juni 2004 bij RIKZ.

4.1 De belangrijkste keuzes

De belangrijkste keuzes die in dit rapport zijn verwerkt als volgt:

1. Uitgangspunten voor het nieuwe droogvalalgoritme zijn dat alle aanpassingen aan waterstanden e.d. in subroutine `trssuw` worden gedaan en dat er zo veel mogelijk consistentie wordt nagestreefd (paragrafen 3.4.1 en 3.4.2).

2. In het nieuwe droogvalalgoritme zal de controle op *wl*-punten binnen het iteratieproces worden gedaan, wordt grenswaarde `0.5*trshwl` gebruikt (zie keuze 14), en worden er eerst twee schotjes in de rekenrichting en dan twee schotjes in de dwarsrichting gezet (paragraaf 3.4.1, laatste twee alineas).

3. Diverse gedetailleerde keuzes voor het nieuwe droogvalalgoritme worden in paragraaf 3.4.2 gespecificeerd.

4. De droogvalcontrole in de dwarsrichting wordt in WAQUA geïntegreerd met het stopcriterium per rij via de methode van Algoritme 6 (paragraaf 3.6.4).

5. In WAQUA zullen de arrays `hu` en `hv` ook wel `hkuh` en `hkvp` worden genoemd, en worden nieuwe arrays `hkup` en `hkvh` voor de andere halve tijdstappen geïntroduceerd. Deze nieuwe arrays worden in de droogvalcontroles in de dwarsrichting gebruikt (punt 2 in paragraaf 3.6.5).

6. In de massacorrectiestap zal niet op droogvallen worden gecontroleerd (paragrafen 3.3.2 en 3.3.3). In WAQPRE wordt een waarschuwing geven als ITERACCURACY te groot is ten opzichte van TRESH_WL_FLOODING, en in WAQPRO wordt een waarschuwing geven wanneer er mogelijk problemen kunnen ontstaan doordat er niet op droogvallen wordt gecontroleerd.

7. Het nieuwe stopcriterium wordt alleen voorbereid in de programmatuur, maar wordt nog niet aangezet (paragraaf 3.6.3).

8. Het huidige stopcriterium wordt aangepast door het verwijderen van het zogenaamde "minit-schema-per-rij" (paragraaf 3.6.3).

9. De aanpassingen van de upwind-methode worden gemaakt via een nieuwe optie UPWIND_ZETA conform paragraaf 2.5.1.

10. In paragraaf 2.6.3 wordt de zogenaamd ideale structuur van de inputfile m.b.t. dieptes en droogvallen gespecificeerd. Deze wordt volledig geïmplementeerd (meerwerk ten opzichte van het oorspronkelijke contract).

11. Bij het verwijderen van `lgrid` uit WAQUA maken we ons niet druk om de effecten van het dubbel in-core houden van arrays op het geheugengebruik (paragraaf 2.2.5).

4.2 Minder belangrijke keuzes

Minder belangrijke keuzes die zijn gemaakt zijn als volgt:

12. Pieken en instabiliteiten in het transportgedeelte van TRIWAQ worden aangepakt via het limiten van de "overshootflux", het verschil tussen de som van de expliciete fluxtermen en de concentratiebehoudende advectieve flux (paragraaf 3.8.1).

13. De gevoeligheid t.g.v. de advectieve termen in WAQUA wordt alleen onderzocht voor subroutine `wassuc`, en hierbij wordt het advectieschema alleen lokaal en tijdelijk aangepast (halverwege paragraaf 3.9.1).

14. Activiteit 3-3 wordt niet uitgevoerd (controleren op 0, paragraaf 3.7). Wel wordt er een aparte grenswaarde THRESH_WL_FLOODING geïntroduceerd voor het spelen met verschillende varianten van de droogvalalgoritmes (laatste alinea in paragraaf 3.7, paragraaf 2.6.3).

15. De minimumwaarde van `0.002` die bij de berekening van de posities van laaginterfaces wordt gebruikt wordt vervangen door `0.499`*DEPCRIT (paragraaf 3.3.3).

16. In WAQUA en TRIWAQ worden dezelfde namen voor de variabelen die droogvalgrenswaardes representeren ingevoerd: `trshuv` en `trshwl` (paragraaf 3.4.4).

17. De huidige "startsnelheid" die in WAQUA bij het onderlopen van open randpunten wordt gebruikt wordt verwijderd (punt 5 in paragraaf 3.5, paragraaf 3.5.1).

18. De vergelijkingen voor droge snelheids-, debiet- en Riemannranden wordt enigszins aangepast (punt 6 in paragraaf 3.5, paragraaf 3.5.1).

19. De droogvalcontrole in *wl*-punten wordt in TRIWAQ ook voor randpunten ingevoerd (punt 7 in paragraaf 3.5, paragraaf 3.5.1).

20. Het verwijderen van `lgrid` zal alleen worden gedaan voor het programma WAQPRO (1e alinea paragraaf 2.2.2). `Lgrid` wordt niet verwijderd uit het SVWP-gedeelte van WAQPRO, uit het matchen van roosters in WAQPRO en COEXEC, en uit de harmonische analyse in WAQPRO (paragraaf 2.2.4).

21. WAQUA zal op dezelfde manier gaan omgaan met tijdstippen van arrays: halve en hele tijdstippen i.p.v. oude en nieuwe waardes (laatste gedeelte van paragraaf 2.2.5, punt 1 in paragraaf 3.6.5).

22. Er worden geen aanpassingen gemaakt aan de namen van variabelen `guu`, `guv`, `geu`, `gev`, e.d. (paragraaf 2.2.5, in de opsomming bij `gsqdi`, `gsqsi`).

23. De DP-arrays worden opgeslagen onder compound-array MESH, naast de bestaande arrays MESH_H en DPS_FLOW. Deze laatste arrays kunnen na een overgangsperiode worden verwijderd uit de programmatuur (paragraaf 2.3.2).

4.3 De belangrijkste opties voor verdere verbetering

De belangrijkste ideeën voor verbetering van WAQUA en TRIWAQ zijn als volgt:

1. Het is gewenst om de ZK-arrays ook in WAQUA in te voeren op een zodanige manier dat dit geen performance kost. Hiermee kunnen WAQUA en TRIWAQ een stuk verder op elkaar worden afgestemd (paragraaf 2.3.1).

2. De definitie van kenmerkarray KCS zou moeten worden aangepast cf. OMS en Delft3D-FLOW, zodat het huidige array KFS overbodig wordt gemaakt. KCS en KFS kunnen dan in de betekenis van OMS op de SDS-file worden gezet (punt 10 in paragraaf 3.4.2).

3. Het transportgedeelte van TRIWAQ zou geen aanpassingen moeten maken aan de laagposities in diagonale randpunten (punt 9 in paragraaf 3.4.2)

4.4 Minder belangrijke opties voor verdere verbetering

Minder belangrijke opties voor verbetering van WAQUA en TRIWAQ zijn als volgt:

4. Er kunnen aanpassingen aan de discretisaties worden ontwikkeld voor net ondergelopen punten, bijvoorbeeld m.b.t. het zetten of schatten van een "startsnelheid" (punt 3 in paragraaf 3.2.1, punt 5 in paragraaf 3.5, paragraaf 3.5.1).

5. De berekening van DPS in diagonale randpunten zou moeten worden verfijnd (laatste alinea in paragraaf 3.5.1).

6. Op de SDS-file zou voor TRIWAQ array `zkuh` kunnen worden opgeslagen (laagposities uit oplossen continuïteitsvergelijking), of in WAQUA zou het opslaan van `hu` = `hkuh` kunnen worden vervangen door het opslaan van `hkup` (punt 2 in paragraaf 3.6.5).

7. De transport-histories lijken in TRIWAQ (`wascht`) op basis van de verkeerde waardes te worden bepaald. Dit zou gerepareerd moeten worden (laatste bullet in paragraaf 2.4.2, probleem is aangemeld by MX.*Systems*).

8. Bij de berekening van time-histories in subroutines `wagcot` en `trscot` zouden de debiet-arrays moeten worden gebruikt (paragraaf 3.4.1, punt 12 in paragraaf 3.4.2).

9. Er kan worden gezocht naar extra vereenvoudigingen van WAQUA en TRIWAQ die mogelijk zijn geworden door het verwijderen van `lgrid` (paragraaf 2.2.6).

10. De stabiliteitscheck kan in TRIWAQ na afloop van het iteratieproces worden gedaan, met een subroutine die ook in WAQUA wordt gebruikt (punt 11 in paragraaf 3.4.2).

11. De arrays `lgubar` en `lgvbar` zouden zodanig in TRIWAQ kunnen worden ingevoerd dat er meer code m.b.t. droogvalcontroles tussen WAQUA en TRIWAQ kan worden gedeeld (5e alinea in paragraaf 3.5.1).

12. De namen `gku` en `gev` kunnen overal in WAQPRO worden ingevoerd waar nu `gvu` en `guv` worden gebruikt (paragraaf 2.2.5, in de opsomming bij `gsqdi`, `gsqsi`).

13. De methode voor linearisatie van de continuïteitsvergelijking in WAQUA kan instelbaar worden gemaakt (methode-TRIWAQ vs. methode-Delft3D, niet besproken in het rapport).

14. De massacorrectiestap zou in de technische documentatie kunnen worden toegevoegd (paragraaf 3.3.5).

Referenties

[1] B. van 't Hof and E.A.H. Vollebregt. Modeling of drying in shallow water using artificial porosity. *??*, ??:???–???, to appear.

[2] J.A.Th.M. van Kester. Technical documentation WAQUA. Technical report, WL/RIKZ, February 1998. preprint.

[3] G.S. Stelling. *On the Construction of Computational Methods for Shallow Water Flow Problems*. PhD thesis, Delft University of Technology, 1983.

[4] G.S. Stelling and S.P.A. Duynmeyer. A numerical method for every Froude number in shallow water flows that might flood dry land. *??*, ??:??–??, 200?

[5] E.A.H. Vollebregt. Parallellisatie van de nieuwe overlaatroutines, September 2001. V$\mathcal{O}$Rtech Computing, Memo EV/M01.011.

[6] E.A.H. Vollebregt. Verbetering van de performance van parallel WAQUA op het Linux rekencluster van RIZA-Arnhem, Mei 2004. V$\mathcal{O}$Rtech Computing, Memo EV/M04.002, versie 1.1.

[7] E.A.H. Vollebregt, M.R.T. Roest, and B. van 't Hof. Detailontwerp domein decompositie met horizontale verfijning. Technical Report TR01-06, V$\mathcal{O}$Rtech Computing, Postbus 260, 2600 AG Delft, Nederland, April 2001. In opdracht van RIKZ.

[8] E.A.H. Vollebregt, M.R.T. Roest, H.H. ten Cate, and H.X. Lin. The PARALLEL project: Parallel simulation of 3D flow and transport models. In L. Dekker, W. Smit, and J.C. Zuidervaart, editors, *Int. EUROSIM Conference HPCN Challenges in telecomp and telecom*, pages 479–486, Amsterdam, the Netherlands, 1996. Elsevier Science Publishers.

[9] E.A.H. Vollebregt and J.A.Th.M. van Kester. Inventarisatie van problemen en oplosrichtingen m.b.t. droogvallen en onderlopen in WAQUA en TRIWAQ. Technical Report TR02-13, V$\mathcal{O}$Rtech Computing, P.O.Box 260, 2600 AG Delft, the Netherlands, November 2002.

[10] E.A.H. Vollebregt and C. van Velzen. Verdere experimenten met de viscositeitskruistermen, Augustus 2001. V$\mathcal{O}$Rtech Computing, Memo EV/M01.009.

[11] R.J. Vos, J. Soerdjbali, and R. van Dijk. Gevoeligheidsonderzoek voor droogvallen en onderlopen in WAQUA/TRIWAQ, beschrijving testresultaten zoet en zout. Technical Report xxx, Rijkswaterstaat/RIKZ, the Hague, the Netherlands, mei 2004.

[12] M. Zijlema. Technical documentation TRIWAQ. Technical Report SIMONA 99-01, National Institute for Coastal and Marine Management, the Hague, the Netherlands, 1999.

[13] M. Zijlema, J.A.Th.M. van Kester, E.A.H. Vollebregt, and E.D. de Goede. Choices for the hydrodynamic module in OMS. Technical report, Rijkswaterstaat/WL|Delft Hydraulics, the Hague, the Netherlands, 2001.