DG Rijkswaterstaat
Rijksinstituut voor Kust en Zee

Haalbaarheidsstudie HYDRA-K

Fase 5 – Inbouwen van een additionele stochast en additionele faalmechanismen in het demonstratiemodel

F.L.M. Diermanse, H.R.A. Jagers, M. Klein Breteler, A.C.W.M. Vrouwenvelder, H.M.G.M. Steenbergen

rapport

maart 2004

H4304

delft hydraulics

Inhoud

1 Inleiding

Achtergrondinformatie

In het kader van de voorbereidingen voor het hydraulische randvoorwaardenboek wordt momenteel een rekenprogramma ontwikkeld door het Rijksinstituut voor Kust en Zee (RIKZ). De voornaamste functionaliteiten van dit rekenprogramma (HYDRA-K) zijn:

- het berekenen van hydraulische randvoorwaarden langs de kust, die bestaan uit een combinatie van de waterstand en golfparameters (het zogenaamde “ontwerppunt” of “toetspunt”) en dienen als invoer voor een toets op de veiligheid;
- het berekenen van een faalkans van een waterkering; en
- het berekenen van de benodigde sterkte en/of dimensie van de waterkering voor een gegeven veiligheidscriterium.

Genoemde functionaliteiten zijn momenteel binnen HYDRA-K ontwikkeld voor drie faalmechanismen:

- golfoploop;
- golfoverslag; en
- stabiliteit (van steenzetting).

In het kader van de voorbereidingen op het hydraulische randvoorwaardenboek zal HYDRA-K uitgebreid worden met een aantal faalmechanismen, behorende bij de volgende bekledingstypen:

- steenzetting (een uitbreiding van de huidige functionaliteit);
- asfaltbekleding; en
- grasbekleding.

Op langere termijn worden de faalmechanismen door middel van dll’s in HYDRA-K ingebouwd. Tijdens de uitvoering van deze studie zullen de dll’s nog niet beschikbaar zijn, maar de rekenregels waarop ze gebaseerd zijn, staan beschreven in de leidraad toetsen op veiligheid (TAW, 1999).

Probleemstelling

Het toevoegen van de in de inleiding genoemde additionele faalmechanismen en bekledingstypen aan HYDRA-K behelst meer dan alleen het inbouwen van een aantal dll’s. De belastingelementen van de additionele faalmechanismen/bekledingen verschillen namelijk van de belastingelementen van de faalmechanismen/bekledingstypen die reeds zijn ingevoerd in HYDRA-K. Eerst dient daarom vastgesteld te worden of de additionele faalmechanismen/bekledingstypen binnen het bestaande concept van HYDRA-K passen.

Daarbij spelen de volgende vraagstukken:

- welke aanpassingen zijn nodig binnen HYDRA-K om de additionele faalmechanismen in te passen?
- hoeveel inspanning vergen deze aanpassingen?
- zijn alle gewenste aanpassingen mogelijk binnen het concept van HYDRA-K?
- is het mogelijk om middels work-arounds alternatieve oplossingen te vinden voor dié aanpassingen die niet binnen het concept van HYDRA-K passen?
- is het op langere termijn technisch haalbaar om, middels enige wijzigingen aan het concept van HYDRA-K, alsnog alle gewenste aanpassingen te realiseren?
- welke inspanningen zijn nodig om een derde (of eventueel vierde) stochast aan HYDRA-K toe te voegen?

Doel van het project

Het doel van dit project is om de haalbaarheid van de inbouw van de genoemde faalmechanismen/bekledingstypen in HYDRA-K te onderzoeken. Afhankelijk van de uitkomst van de haalbaarheidsstudie zal een aantal van deze faalmechanismen in een demonstratiemodel van HYDRA-K ingebouwd worden. De nauwkeurigheid van de rekenresultaten van deze faalmechanismen zal geanalyseerd worden.

Voor de faalmechanismen die niet op korte termijn ingebouwd kunnen worden, wordt in de eerste plaats een work-around opgesteld en vervolgens wordt onderzocht of het op langere termijn technische haalbaar is om, middels enige wijzigingen aan het concept van HYDRA-K, deze faalmechanismen alsnog in te bouwen.

Fasering

Het project is opgedeeld in vijf fasen:

Fase 1: Plan van aanpak.
Fase 2: Inbouwen van additionele faalmechanismen en bekledingstypen in het demonstratiemodel.
Fase 3: Indicatieve bepaling van de nauwkeurigheid.
Fase 4: Alternatieven (work-arounds) voor het inbouwen van faalmechanismen.
Fase 5: Inbouwen van een additionele stochast en additionele faalmechanismen in het demonstratiemodel.

Het voorliggende rapport vormt één van de eindproducten van **fase 5**: "*Inbouwen van een additionele stochast en additionele faalmechanismen in het demonstratiemodel*". Het andere product is een CD met daarop een versie van het demonstratiemodel.

Opgemerkt wordt dat de titel en uitgevoerde activiteiten van deze fase gewijzigd zijn ten opzichte van de oorspronkelijke opzet zoals geformuleerd in de offerte-aanvraag en de offerte. In de oorspronkelijke opzet is fase 5 gedefinieerd om een *werkplan* op te zetten voor het op lange termijn inbouwen van dié faalmechanismen waarvoor in de eerste fasen van het project is vastgesteld dat ze niet binnen het concept van HYDRA-K passen. Verder diende

het werkplan een aanpak te bevatten over het op termijn invoeren van een additionele stochast (golfperiode) in HYDRA-K. Om een aantal redenen is tijdens het verloop van het project duidelijk geworden dat een andere insteek voor fase 5 gekozen diende te worden. In de eerste plaats is tijdens fase 1 t/m 3 voor geen van de ingebouwde faalmechanismen vastgesteld dat ze niet binnen het concept passen. Verder zijn in fase 4 voor een aantal additionele faalmechanismen work-arounds opgesteld die eveneens binnen het concept van HYDRA-K passen en in het demonstratiemodel ingebouwd kunnen worden. Tot slot is op basis van een analyse met gestileerde modellen in fase 3 vastgesteld dat het toevoegen van een additionele stochast aan HYDRA-K goed mogelijk is zonder daarbij op een ander concept dan het huidige (methode de Haan) over te moeten stappen.

Het gevolg is nu dat fase 5 niet meer het karakter heeft van een werkplan, maar een stap verder gaat: de onderdelen die oorspronkelijk bedoeld waren voor het werkplan zijn nu grotendeels geïmplementeerd. De volgende activiteiten zijn uitgevoerd in fase 5:

1. Implementatie van de additionele stochast (golfperiode);
2. Implementatie van de in fase 4 beschreven additionele faalmechanismen;
3. Implementatie van additionele typen steenzettingen; en
4. Uitvoeren van testberekeningen.

Auteurs en organisatie

Dit rapport is opgesteld door dr. ir. F. L. M. Diermanse (WL | Delft Hydraulics, projectleider), dr. ir. H.R.A. Jagers (WL | Delft Hydraulics), ir. M. Klein Breteler (WL | Delft Hydraulics), Prof. ir. A.C.W.M. Vrouwenvelder (TNO-Bouw) en ir. H.M.G.M. Steenbergen (TNO-bouw).

Het voorliggende rapport is één van de producten van het project, zoals verwoord in RIKZ-opdrachtnummer RKZ-1345 d.d. 15 september 2003.

2 Aanpassingen aan het demonstratiemodel en gevolgen

2.1 Inleiding

In fase 4 is een plan van aanpak beschreven voor het opzetten van de finale versie van het demonstratiemodel. Het werkplan beschreef de volgende aanpassingen ten opzichte van voorgaande versies van het demonstratiemodel:

1. additionele faalmechanismen en grenstoestandsfuncties (de zogenoemde work-arounds);
2. de golfperiode als stochast; en
3. additionele typen steenzettingen.

In de huidige fase is het werkplan geïmplementeerd. Aangezien niet is afgeweken van het werkplan wordt deze verder niet beschreven en verwijzen we de geïnteresseerde lezer naar het fase 4 rapport (WL/TNO, 2004b). De aangepast code is wél beschreven in dit rapport (in de bijlage).

Dit hoofdstuk beschrijft de berekeningen die zijn uitgevoerd na implementatie van het werkplan en de analyses van de resultaten. Er zijn berekeningen uitgevoerd om het effect te bepalen van het gebruik van de additionele stochast (punt 2). Verder is in vergelijking met voorgaande versies de berekeningswijze van de helling van het talud aangepast en de gevolgen daarvan zijn gekwantificeerd en in de eerstvolgende paragraaf beschreven.

2.2 Aangepaste berekeningswijze helling talud

In de eerste implementatie van de stabiliteitsformuleringen voor de bekledingstypen gras en steenzettingen werd de helling bepaald door middel van een gewogen gemiddelde van de taludhelling tussen h+1.5H_s en h-1.5H_s (waarbij h het niveau van de stilwaterlijn is en H_s de golfhoogte). Daarbij werd overigens geen rekening gehouden met de berm. Voor het bekledingstype asfalt werd de lokale helling ter plaatse van h genomen (eveneens zonder rekening te houden met de berm).

In de meest recente versie van het demonstratiemodel wordt bij de formuleringen voor asfalt en steenzetting op de berm gesteld dat de helling van het ondertalud maatgevend is. Hierbij bestaat het gevaar voor een inconsistente toepassing van de berekening van de helling. Als namelijk een stuk bekleding ónder de berm ligt en de stilwaterlijn juist erboven, dan wordt de helling van het boventalud als representatief beschouwd terwijl bij een bekleding op de berm altijd de helling van het ondertalud zou worden genomen. Om deze inconsistentie op te lossen is ervoor gekozen om altijd de helling van het ondertalud te nemen in het geval van bekleding op het ondertalud of berm, en de helling van het boventalud bij bekleding op het boventalud.

Het gevolg is dat de rekenresultaten van fase 3 (WL/TNO, 2004a) niet per definitie exact gereproduceerd kunnen worden met de laatste versie van het demonstratiemodel. De verschillen zijn echter zeer klein, zoals aan de hand van testberekeningen is aangetoond. De resultaten van deze berekeningen zijn opgenomen in Tabel 2.1 (versie fase 3) en Tabel 2.2 (versie fase 5). In Tabel 2.3 en Tabel 2.4 zijn bovendien de verschillen weergegeven voor respectievelijk steenbekleding en grasbekleding. Uit de tabellen blijkt dat voor steenbekledingen het verschil tussen de twee modelversies in de regel gelijk is aan 0 of 1 cm. Uitzondering op deze regel vormt de locatie Petten bij een herhalingstijd van 10.000 jaar waar een iets groter verschil (3 cm) optreedt. Dit laatste wordt veroorzaakt door de dominantie van het faalmechanisme afschuiving bij deze locatie en herhalingstijd. Voor grasbekleding zijn de uitkomsten gelijk gebleven, op twee uitzonderingen na, beide bij locatie Petten. In deze twee uitzonderingsgevallen heeft de aangepast berekeningswijze er voor gezorgd dat de uitkomst net over de grens gebracht worden tussen “goede grasmat vereist” een “geen enkele kwaliteit grasmat voldoet”.

Oorzaak onvoldoende faalwaarnemingen

In de tabellen komt een aantal keer de afkorting “O.F.” voor. Dit staat voor “onvoldoende faalwaarnemingen” en het kan verschillende oorzaken hebben. Bij locatie Vlissingen op 1 m+NAP ligt het getoetste deel van het talud te laag om belast te worden. In dit geval ligt de waterstand gedurende elk uur van alle opgeschaalde stormgebeurtenissen (ruimschoots) hoger dan 1.5 m+NAP zodat dit stuk bekleding niet belast wordt. In feite volgt hieruit dat dit deel van de dijk niet zal falen, ongeacht de sterkte van de bekleding. Dit is uiteraard niet realistisch. Hier wreekt zich het feit dat na opschaling van de stormgebeurtenissen de waterstand aan het begin en eind van de storm relatief hoog ligt en dat het feitelijke begin en eind van de storm eigenlijk wordt overgeslagen. Dit probleem is opgelost door toepassing van zgn. “stormflanken” (zie paragraaf 2.4).

Bij locatie Den Helder voor lage herhalingstijden op 5 m+NAP zijn eveneens onvoldoende faalwaarnemingen beschikbaar. In dit geval ligt het getoetste deel van het talud juist te hoog om belast te worden. In tegenstelling tot de hiervoor beschreven situatie bij Vlissingen is dit niet in strijd met de intuïtie: stormgebeurtenissen met lage herhalingstijden zijn minder extreem dan stormgebeurtenissen met hoge herhalingstijden, en dus kan het gebeuren dat de hogere delen van het talud onbelast blijven.

Tabel 2.1 Benodigde steendikte (m) en graskwaliteit volgens de het demonstratiemodel van **fase 3** voor 3 locaties, 5 verschillende dijkdelen en 8 herhalingstijden (T).

Vlissingen					
T (jaar)	**steen, 1 m+NAP**	**steen, 3 m+NAP**	**steen, 5 m+NAP**	**gras, 5 m+NAP**	**gras, 7 m+NAP**
10	O.F.	0.22	0.17	slecht	slecht
30	O.F.	0.24	0.19	matig	slecht
100	O.F.	0.26	0.21	goed	slecht
300	O.F.	0.28	0.22	goed	slecht
1000	O.F.	0.30	0.24	voldoet niet	slecht
2000	O.F.	0.31	0.26	voldoet niet	slecht
4000	O.F.	0.32	0.30	voldoet niet	slecht
10000	O.F.	0.33	0.32	voldoet niet	matig
Den Helder					
T (jaar)	**steen, 1 m+NAP**	**steen, 3 m+NAP**	**steen, 5 m+NAP**	**gras, 5 m+NAP**	**gras, 7 m+NAP**
10	0.21	0.17	O.F.	slecht	slecht
30	0.22	0.21	O.F.	slecht	slecht
100	0.24	0.24	O.F.	slecht	slecht
300	0.25	0.26	0.20	matig	slecht
1000	0.26	0.28	0.22	matig	slecht
2000	0.26	0.29	0.23	matig	slecht
4000	0.27	0.30	0.24	goed	slecht
10000	0.28	0.31	0.25	goed	slecht
Petten					
T (jaar)	**steen, 1 m+NAP**	**steen, 3 m+NAP**	**steen, 5 m+NAP**	**gras, 5 m+NAP**	**gras, 7 m+NAP**
10	0.37	0.30	0.30	matig	slecht
30	0.41	0.41	0.32	goed	matig
100	0.44	0.44	0.35	goed	matig
300	0.47	0.47	0.38	voldoet niet	goed
1000	0.51	0.51	0.41	voldoet niet	goed
2000	0.53	0.53	0.42	voldoet niet	goed
4000	0.55	0.55	0.44	voldoet niet	voldoet niet
10000	0.61	0.61	0.46	voldoet niet	voldoet niet

O.F. betekent: onvoldoende faalwaarnemingen
“voldoet niet” betekent: zelfs een grasmat met goede kwaliteit zal niet voldoen

Tabel 2.2 Benodigde steendikte (m) en graskwaliteit volgens de het demonstratiemodel van **fase 5** voor 3 locaties, 5 verschillende dijkdelen en 8 herhalingstijden (T).

Vlissingen					
T (jaar)	**steen, 1 m+NAP**	**steen, 3 m+NAP**	**steen, 5 m+NAP**	**gras, 5 m+NAP**	**gras, 7 m+NAP**
10	O.F.	0.22	0.17	slecht	slecht
30	O.F.	0.24	0.19	matig	slecht
100	O.F.	0.26	0.21	goed	slecht
300	O.F.	0.27	0.22	goed	slecht
1000	O.F.	0.29	0.24	voldoet niet	slecht
2000	O.F.	0.30	0.25	voldoet niet	slecht
4000	O.F.	0.31	0.29	voldoet niet	slecht
10000	O.F.	0.32	0.31	voldoet niet	matig
Den Helder					
T (jaar)	**steen, 1 m+NAP**	**steen, 3 m+NAP**	**steen, 5 m+NAP**	**gras, 5 m+NAP**	**gras, 7 m+NAP**
10	0.21	0.18	O.F.	slecht	slecht
30	0.23	0.21	O.F.	slecht	slecht
100	0.24	0.24	O.F.	slecht	slecht
300	0.25	0.25	0.20	matig	slecht
1000	0.26	0.27	0.22	matig	slecht
2000	0.27	0.28	0.23	matig	slecht
4000	0.27	0.29	0.23	goed	slecht
10000	0.28	0.30	0.24	goed	slecht
Petten					
T (jaar)	**steen, 1 m+NAP**	**steen, 3 m+NAP**	**steen, 5 m+NAP**	**gras, 5 m+NAP**	**gras, 7 m+NAP**
10	0.37	0.30	0.30	matig	slecht
30	0.41	0.41	0.33	goed	matig
100	0.45	0.45	0.36	goed	matig
300	0.48	0.48	0.38	goed	goed
1000	0.51	0.51	0.41	voldoet niet	goed
2000	0.53	0.53	0.43	voldoet niet	goed
4000	0.55	0.55	0.44	voldoet niet	voldoet niet
10000	0.58	0.58	0.47	voldoet niet	voldoet niet

O.F. betekent: onvoldoende faalwaarnemingen
"voldoet niet" betekent: zelfs een grasmat met goede kwaliteit zal niet voldoen

Tabel 2.3 Verschil tussen de benodigde steendikte (m) bij steenbekleding volgens het demonstratiemodel van fase 3 en het demonstratiemodel van fase 5 voor 3 locaties, 3 verschillende dijkdelen en 8 herhalingstijden (T). De verschillen worden veroorzaakt door een andere hellingdefinitie. Daar waar een streepje staat was geen vergelijking mogelijk. Positieve getallen duiden op een stijging in vergelijking met fase 3, negatieve getallen duiden op een daling ten opzichte van fase 3.

Vlissingen			
T (jaar)	**steen, 1 m+NAP**	**steen, 3 m+NAP**	**steen, 5 m+NAP**
10	-	0.00	0.00
30	-	0.00	0.00
100	-	0.00	0.00
300	-	-0.01	0.00
1000	-	-0.01	0.00
2000	-	-0.01	-0.01
4000	-	-0.01	-0.01
10000	-	-0.01	-0.01
Den Helder			
T (jaar)	**steen, 1 m+NAP**	**steen, 3 m+NAP**	**steen, 5 m+NAP**
10	0.00	0.01	-
30	0.01	0.00	-
100	0.00	0.00	-
300	0.00	-0.01	0.00
1000	0.00	-0.01	0.00
2000	0.01	-0.01	0.00
4000	0.00	-0.01	-0.01
10000	0.00	-0.01	-0.01
Petten			
T (jaar)	**steen, 1 m+NAP**	**steen, 3 m+NAP**	**steen, 5 m+NAP**
10	0.00	0.00	0.00
30	0.00	0.00	0.01
100	0.01	0.01	0.01
300	0.01	0.01	0.00
1000	0.00	0.00	0.00
2000	0.00	0.00	0.01
4000	0.00	0.00	0.00
10000	-0.03	-0.03	0.01

Tabel 2.4 Verschil tussen de benodigde kwaliteit van de grasmat voor het demonstratiemodel van fase 3 en het demonstratiemodel van fase 5 voor 3 locaties, 2 verschillende dijkdelen en 8 herhalingstijden (T). De berekeningen waar verschillen zijn gevonden tussen zijn grijs gemarkeerd.

Vlissingen				
	gras, 5 m+NAP		**gras, 7 m+NAP**	
T (jaar)	**Fase 3**	**Fase 5**	**Fase 3**	**Fase 5**
10	slecht	slecht	slecht	slecht
30	matig	matig	slecht	slecht
100	goed	goed	slecht	slecht
300	goed	goed	slecht	slecht
1000	voldoet niet	voldoet niet	slecht	slecht
2000	voldoet niet	voldoet niet	slecht	slecht
4000	voldoet niet	voldoet niet	slecht	slecht
10000	voldoet niet	voldoet niet	matig	matig
Den Helder				
	gras, 5 m+NAP		**gras, 7 m+NAP**	
T (jaar)	**Fase 3**	**Fase 5**	**Fase 3**	**Fase 5**
10	slecht	slecht	slecht	slecht
30	slecht	slecht	slecht	slecht
100	slecht	slecht	slecht	slecht
300	matig	matig	slecht	slecht
1000	matig	matig	slecht	slecht
2000	matig	matig	slecht	slecht
4000	goed	goed	slecht	slecht
10000	goed	goed	slecht	slecht
Petten				
	gras, 5 m+NAP		**gras, 7 m+NAP**	
T (jaar)	**Fase 3**	**Fase 5**	**Fase 3**	**Fase 5**
10	matig	matig	slecht	slecht
30	goed	goed	matig	matig
100	goed	goed	matig	matig
300	voldoet niet	goed	goed	goed
1000	voldoet niet	voldoet niet	goed	goed
2000	voldoet niet	voldoet niet	voldoet niet	goed
4000	voldoet niet	voldoet niet	voldoet niet	voldoet niet
10000	voldoet niet	voldoet niet	voldoet niet	voldoet niet
“voldoet niet” betekent: zelfs een grasmat met goede kwaliteit zal niet voldoen				

2.3 Golfperiode als additionele stochast

Inleiding

De golfperiode is als additionele stochast toegevoegd aan het demonstratiemodel. Overigens is dit “slechts” een optie in het model, dus het is ook nog steeds mogelijk om de golfperiode deterministisch gerelateerd aan de wind op diep water te kiezen. Aangenomen is dat de golfperiode asymptotisch afhankelijk is van de bestaande stochasten windsnelheid en waterstand. Als gevolg van deze aanname kan de golfperiode op dezelfde wijze opgeschaald worden (methode de Haan) als de waterstand en de windsnelheid.

Een praktisch probleem dat hierbij ontstaat is dat in de SWAN-database geen rekening is gehouden met de golfperiode als extra stochast. Deze database bevat de uitvoer van berekeningen voor verschillende waarden voor de windrichting, windsnelheid en waterstand. De berekeningen vormen aldus een 3-dimensionaal rooster waarmee het relevante bereik voor windrichting, windsnelheid en waterstand wordt omvat. Bij het toevoegen van de golfperiode als additionele stochast dient dit rooster uitgebreid te worden. Weliswaar is bij elke berekening een golfperiode op diep water gebruikt, maar deze is één op één gerelateerd met de windsnelheid en windrichting en vormt dus niet de gewenste vierde dimensie van het rooster.

Het past niet binnen de tijdspanne van het huidige project om de extra benodigde SWAN-berekeningen uit te voeren. Echter, omdat het huidige project een haalbaarheidsstudie is, wordt het acceptabel geacht om de database op kunstmatige wijze uit te breiden. Dit wordt gerealiseerd door uitkomsten van de benodigde *additionele* SWAN-berekeningen te schatten op basis van de *beschikbare* SWAN-berekeningen. Daarbij wordt als vuistregel gehanteerd dat de relatie tussen de offshore piekperiode en de nearshore piekperiode van de additionele SWAN-berekeningen gelijk is aan die van de beschikbare SWAN-berekeningen.

Bepalen van fictieve SWAN-resultaten

In het kader van de analyses in de huidige haalbaarheidsstudie volstaat het om het “uitbreiden” van de database in de Matlab-code van HYDRA-K uit te voeren tijdens een berekening en derhalve niet op voorhand in de database zelf. Voor elke tijdstap van de doorgerekende stormgebeurtenissen wordt ter plekke de nearshore golfperiode geschat op basis van de eerder genoemde verhouding offshore/nearshore:

$$T_{p,near} = \frac{T^*_{p,near}}{T^*_{p,off}} T_{p,off}$$

Hierin is:

$T_{p,near}$ = de gezochte waarde van de nearshore golfperiode van de doorgerekende tijdstap

$T^*_{p,near}$ = referentie-waarde van de nearshore golfperiode

$T^*_{p,off}$ = referentie-waarde van de offshore golfperiode

$T_{p,off}$ = de offshore golfperiode van de doorgerekende tijdstap

De twee referentie-waarden worden dus gebruikt om de verhouding tussen de offshore periode en de nearshore periode te bepalen. De drie waarden van het rechterlid in bovenstaande vergelijking worden als volgt bepaald

1. De referentie-waarde $T^*_{p,near}$ is de nearshore golfperiode zoals die in de “oude methode” (waarin de golfperiode geen stochast is) bepaald wordt. Dat betekent dat op basis van de heersende windrichting, windsnelheid en waterstand de golfperiode bepaald wordt door middel van 3-dimensionale interpolatie van SWAN-resultaten uit de database. In de SWAN-database vormen de windrichting, windsnelheid en waterstand namelijk als het ware een 3-dimensionaal rooster.
2. De referentie-waarde $T^*_{p,off}$ wordt op dezelfde wijze bepaald als de referentie-waarde van de nearshore golfperiode. De SWAN-database bevat immers niet alleen de berekende nearshore golfperioden, maar ook de gehanteerde offshore golfperiode. Middels de eerder genoemde 3-dimensionale interpolatie kan hieruit een offshore (referentie-)waarde bepaald worden op basis van de heersende windrichting, windsnelheid en waterstand.
3. De offshore golfperiode $T_{p,off}$ wordt bepaald door opschaling van metingen uit het bestand met simultane waarnemingen. De golfperiode is nu immers een stochast binnen de methode de Haan en wordt derhalve op dezelfde wijze opgeschaald als de windsnelheid en de waterstand.

Resultaten

Het effect van het toevoegen van de additionele stochast is gekwantificeerd op basis van vergelijkende berekeningen, d.w.z. berekeningen met en zonder additionele stochast (vanaf nu worden deze berekeningen aangeduid als met de termen “deterministische golfperiode” resp. “stochastische golfperiode”). De resultaten van de berekeningen met extra stochast zijn opgenomen in Tabel 2.5, terwijl de resultaten zonder extra stochast eerder al zijn weergegeven in Tabel 2.2. In Tabel 2.6 en Tabel 2.7 zijn bovendien de verschillen weergegeven voor respectievelijk steenbekleding en grasbekleding.

Uit de tabellen blijkt dat voor grasbekledingen de uitkomsten merendeels gelijk zijn; in 7 van de 48 gevallen wordt een verschil gevonden (zie Tabel 2.7). Uit de verschillen is af te leiden dat bij de locaties Vlissingen en Den Helder het gebruik van de extra stochast aanleiding geeft tot minder scherpe eisen aan de bekleding, terwijl het voor locatie Petten juist tot scherpere eisen leidt. Bij steenbekledingen is het verschil tussen de twee opties in de regel gelijk aan 0, 1 of 2 cm. Uitzondering op deze regel vormt locatie Den Helder op niveau 1 m+NAP, waar verschillen kunnen oplopen tot maximaal 6 cm. Het blijkt dat bij relatief lage waterstanden (die maatgevend zijn voor het stuk bekleding op 1 m+NAP) te Den Helder er een relatief groot verschil is in de deterministische gekozen golfperiode en de stochastisch gekozen golfperiode. Bij hogere waterstanden zijn deze verschillen beduidend kleiner, hetgeen verklaart waarom op 3 m+NAP en 5 m+NAP nauwelijks verschillen in ontwerpdikten gevonden worden.

Tabel 2.5 Benodigde steendikte (m) en graskwaliteit met inachtneming van de golfperiode als stochast voor 3 locaties, 5 verschillende dijkdelen en 8 herhalingstijden (T).

Vlissingen					
T (jaar)	**steen, 1 m+NAP**	**steen, 3 m+NAP**	**steen, 5 m+NAP**	**gras, 5 m+NAP**	**gras, 7 m+NAP**
10	O.F.	0.21	0.15	slecht	slecht
30	O.F.	0.23	0.17	matig	slecht
100	O.F.	0.24	0.19	matig	slecht
300	O.F.	0.26	0.21	goed	slecht
1000	O.F.	0.27	0.22	voldoet niet	slecht
2000	O.F.	0.28	0.25	voldoet niet	slecht
4000	O.F.	0.29	0.28	voldoet niet	slecht
10000	O.F.	0.30	0.29	voldoet niet	slecht
Den Helder					
T (jaar)	**steen, 1 m+NAP**	**steen, 3 m+NAP**	**steen, 5 m+NAP**	**gras, 5 m+NAP**	**gras, 7 m+NAP**
10	0.22	0.18	O.F.	slecht	slecht
30	0.25	0.21	O.F.	slecht	slecht
100	0.27	0.23	O.F.	slecht	slecht
300	0.29	0.25	0.20	slecht	slecht
1000	0.31	0.26	0.21	matig	slecht
2000	0.32	0.27	0.22	matig	slecht
4000	0.33	0.28	0.23	matig	slecht
10000	0.34	0.29	0.24	goed	slecht
Petten					
T (jaar)	**steen, 1 m+NAP**	**steen, 3 m+NAP**	**steen, 5 m+NAP**	**gras, 5 m+NAP**	**gras, 7 m+NAP**
10	0.37	0.30	0.30	matig	slecht
30	0.41	0.41	0.33	goed	matig
100	0.44	0.44	0.36	goed	goed
300	0.48	0.48	0.38	voldoet niet	goed
1000	0.52	0.52	0.41	voldoet niet	goed
2000	0.53	0.53	0.43	voldoet niet	voldoet niet
4000	0.55	0.55	0.44	voldoet niet	voldoet niet
10000	0.58	0.58	0.47	voldoet niet	voldoet niet

O.F. betekent: onvoldoende faalwaarnemingen
"voldoet niet" betekent: zelfs een grasmat met goede kwaliteit zal niet voldoen

Tabel 2.6 Verschil tussen de benodigde steendikte (m) bij steenbekleding zonder inachtneming van de golfperiode als stochast (Tabel 2.2) en mét inachtneming van de golfperiode als stochast (Tabel 2.5) voor 3 locaties, 3 verschillende dijkdelen en 8 herhalingstijden (T). Daar waar een streepje staat was geen vergelijking mogelijk. Positieve getallen duiden op een stijging als gevolg van de extra stochast, negatieve getallen duiden op een daling als gevolg van de extra stochast.

Vlissingen			
T (jaar)	**steen, 1 m+NAP**	**steen, 3 m+NAP**	**steen, 5 m+NAP**
10	-	-0.01	-0.02
30	-	-0.01	-0.02
100	-	-0.02	-0.02
300	-	-0.01	-0.01
1000	-	-0.02	-0.02
2000	-	-0.02	0.00
4000	-	-0.02	-0.01
10000	-	-0.02	-0.02
Den Helder			
T (jaar)	**steen, 1 m+NAP**	**steen, 3 m+NAP**	**steen, 5 m+NAP**
10	0.01	0.00	-
30	0.02	0.00	-
100	0.03	-0.01	-
300	0.04	0.00	0.00
1000	0.05	-0.01	-0.01
2000	0.05	-0.01	-0.01
4000	0.06	-0.01	0.00
10000	0.06	-0.01	0.00
Petten			
T (jaar)	**steen, 1 m+NAP**	**steen, 3 m+NAP**	**steen, 5 m+NAP**
10	0.00	0.00	0.00
30	0.00	0.00	0.00
100	-0.01	-0.01	0.00
300	0.00	0.00	0.00
1000	0.01	0.01	0.00
2000	0.00	0.00	0.00
4000	0.00	0.00	0.00
10000	0.00	0.00	0.00

Tabel 2.7 Verschil tussen de benodigde kwaliteit van de grasmat zonder inachtneming van de golfperiode als stochast (Tabel 2.2) en met inachtneming van de golfperiode als stochast (Tabel 2.5) voor 3 locaties, 2 verschillende dijkdelen en 8 herhalingstijden (T). De berekeningen waar verschillen zijn gevonden tussen zijn grijs gemarkeerd.

Vlissingen

	gras, 5 m+NAP		gras, 7 m+NAP	
T (jaar)	**T_p geen stochast**	**T_p stochast**	**T_p geen stochast**	**T_p stochast**
10	slecht	slecht	slecht	slecht
30	matig	matig	slecht	slecht
100	goed	matig	slecht	slecht
300	goed	goed	slecht	slecht
1000	voldoet niet	voldoet niet	slecht	slecht
2000	voldoet niet	voldoet niet	slecht	slecht
4000	voldoet niet	voldoet niet	slecht	slecht
10000	voldoet niet	voldoet niet	matig	slecht

Den Helder

	gras, 5 m+NAP		gras, 7 m+NAP	
T (jaar)	**T_p geen stochast**	**T_p stochast**	**T_p geen stochast**	**T_p stochast**
10	slecht	slecht	slecht	slecht
30	slecht	slecht	slecht	slecht
100	slecht	slecht	slecht	slecht
300	matig	slecht	slecht	slecht
1000	matig	matig	slecht	slecht
2000	matig	matig	slecht	slecht
4000	goed	matig	slecht	slecht
10000	goed	goed	slecht	slecht

Petten

	gras, 5 m+NAP		gras, 7 m+NAP	
T (jaar)	**T_p geen stochast**	**T_p stochast**	**T_p geen stochast**	**T_p stochast**
10	matig	matig	slecht	slecht
30	goed	goed	matig	matig
100	goed	goed	matig	goed
300	goed	voldoet niet	goed	goed
1000	voldoet niet	voldoet niet	goed	goed
2000	voldoet niet	voldoet niet	goed	voldoet niet
4000	voldoet niet	voldoet niet	voldoet niet	voldoet niet
10000	voldoet niet	voldoet niet	voldoet niet	voldoet niet

"voldoet niet" betekent: zelfs een grasmat met goede kwaliteit zal niet voldoen

Verschillen tussen deterministische en stochastische golfperiode

Om meer grip te krijgen op de oorzaken van de verschillen tussen de stochastische en deterministisch gekozen golfperiode is de berekening voor Den Helder op 1 m+NAP nader geanalyseerd. Voor een herhalingstijd van 4000 jaar is de berekening uitgevoerd met beide opties en zijn verscheidene variabelen tegen elkaar uitgezet en geanalyseerd voor elk uur (21.343 in totaal) van de doorgerekende stormgebeurtenissen. In eerste instantie zijn de deterministische en stochastische (nearshore) golfperiode tegen elkaar uitgezet (Figuur 2.1) en is gekeken hoe dicht de waarden rond de lijn y=x liggen. Uit deze figuur valt gelijk de grote afwijking op van de bijna verticale puntenwolk rond de lijn x=5. Hieruit blijkt dat in situaties waar de deterministische piekperiode iets groter is dan 5 s de stochastisch gekozen golfperiode kan oplopen tot wel 9 s, een verschil dus van bijna 4 s.

Deze aparte “aftakking” van de puntenwolk in Figuur 2.1 doet vermoeden dat hier een relatie is met de onderliggende windrichting(en). Het verschil tussen de deterministische en de stochastische (nearshore) piekperiode is daarom uitgezet tegen de windrichting (Figuur 2.2) en er blijkt inderdaad een sterke relatie te zijn. De grote verschillen treden vooral op bij windrichtingen rond de 60°. Omdat de keuzeprocedure voor de golfperiode (stochastisch en deterministisch) in eerste instantie op diep water plaats vindt is eenzelfde vergelijking gemaakt voor offshore golfperiodes (Figuur 2.3). Het patroon van de beide figuren is vrijwel exact gelijk, met dien verstande dat de verschillen offshore een stuk groter zijn.

Om de oorzaak van de piek bij 60° te achterhalen zijn de stochastische en deterministische (offshore) piekperiodes individueel uitgezet tegen de windrichting (Figuur 2.4 en Figuur 2.5). De stochastisch gekozen piekperiode daalt redelijk lineair van ± 17 s in noordelijke richtingen naar ± 9 s in zuidelijke richtingen. De deterministisch gekozen piekperiode kent daarentegen een opvallend verval bij de overgang van westelijke windrichtingen (210° – 30°) naar oostelijke richtingen (60° t/m 180°). Omdat het verval zich inzet bij 60° is daar juist het verschil tussen de stochastische en deterministische piekperiode het grootst, zoals blijkt uit de pieken van Figuur 2.2 en Figuur 2.3.

Behalve het verval in Figuur 2.5 valt ook op dat de piekperiode constant is voor alle oostelijke richtingen, ongeacht de windsnelheid. De deterministische golfperiode wordt bepaald op basis van de deterministische offshore relatie tussen de wind en waterstand enerzijds en golfhoogte, golfperiode anderzijds. Deze relatie is in tabelvorm beschikbaar voor combinaties van windrichting, windsnelheid en waterstand (zie Tabel 2.8). In het geval van Den Helder is deze relatie gebaseerd op offshore-metingen in het station “Eierlandse Gat”. Uit de tabel blijkt dat voor oostelijke richtingen telkens een golfhoogte van 4 m en een golfperiode van 9.15 s verondersteld wordt op te treden (een enkele keer staat er 9.10 s in plaats van 9.15 s, maar dat verschil is voor de huidige analyse verwaarloosbaar). Kennelijk is voor de oostelijke windrichtingen onvoldoende informatie beschikbaar en is de tabel opgevuld met “dummy-waarden”.

Normaal gesproken hebben de genoemde “dummy-waarden” voor Oostelijke windrichtingen geen invloed op het resultaat van ontwerpberekeningen, omdat de relevante stormgebeurtenissen uit westelijke windrichting komen. Echter met de introductie van de additionele stochast bestond de noodzaak om de SWAN-database uit te breiden met fictieve

rekenresultaten en de wijze waarop dat gebeurd is heeft “per toeval” ervoor gezorgd dat de dummy-waarden plotseling wel invloed hebben op de uitkomsten. Dat is als volgt te verklaren:

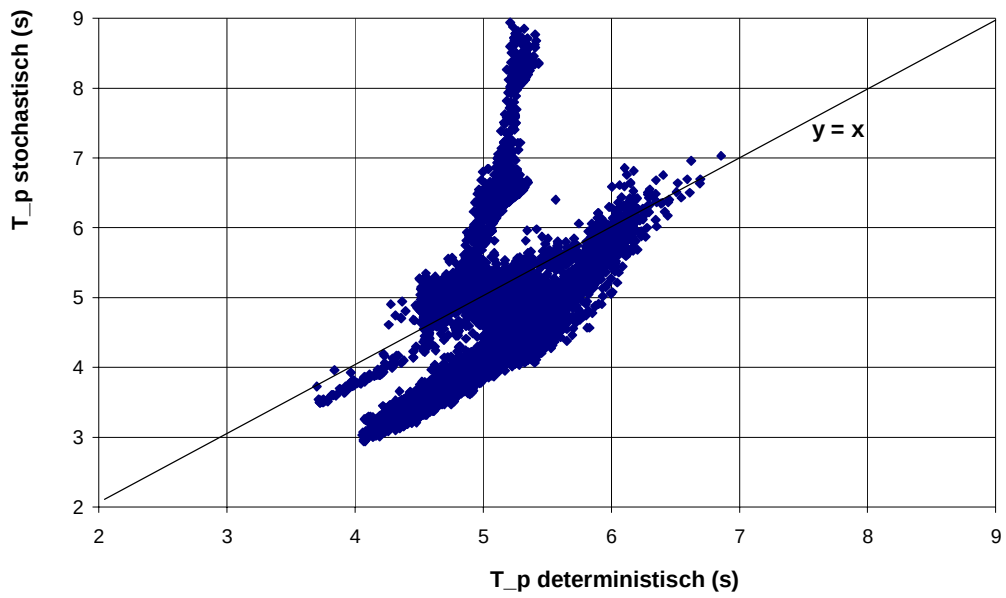
- Bij het uitbreiden van de database is als vuistregel gehanteerd dat de relatie tussen de offshore piekperiode en de nearshore piekperiode van de additionele SWAN-berekeningen gelijk is aan die van de beschikbare SWAN-berekeningen.
- Als gevolg van deze aanname is de verhouding tussen de stochastische golfperiode en deterministische golfperiode offshore hetzelfde als nearshore.
- Bij oostelijke richtingen (60° t/m 180°) is de offshore golfperiode uit de database onafhankelijk van de windsnelheid en relatief laag (9.15 s). Met name bij 60° is de stochastisch gekozen offshore piekperiode relatief groot ten opzichte van de deterministisch gekozen offshore piekperiode.
- Doordat de verhouding stochastisch/deterministisch gelijk is voor offshore en nearshore is bij windrichtingen van rond de 60° de stochastische nearshore golfperiode relatief groot ten opzichte van de deterministische nearshore golfperiode.
- De nearshore deterministische golfperiode volgt uit SWAN-berekeningen en ondanks de relatief lage offshore dummy-waarde van 9.15 s is de nearshore golfperiode voor windrichtingen van rond de 60° niet significant lager dan voor andere windrichtingen (zie Figuur 2.6).
- De combinatie van de voorgaande twee bullets leidt er toe dat de stochastische nearshore golfperiodes voor windrichtingen van rond de 60° significant groter zijn dan voor andere windrichtingen (zie Figuur 2.7).
- Bij het toetsen van een stuk bekleding op 1 m+NAP zijn de oostelijke windrichtingen het meest relevant omdat de (opgeschaalde) stilwaterlijn bij deze windrichtingen rond deze hoogte liggen (zie Figuur 2.8) terwijl de stilwaterlijn voor westelijke windrichtingen hoger liggen.

De Oostelijke windrichtingen zijn dus bepalend voor de belasting op 1 m+NAP en bij deze richtingen kunnen (met name bij 60°) de verschillen tussen de deterministische en stochastische piekperiode hoog oplopen. Vandaar dat het effect van de additionele stochast op de ontwerpdikte relatief groot is op 1 m+NAP (6 cm, zie Tabel 2.3) terwijl het verschil op 3 m+NAP en 5 m+NAP verwaarloosbaar is.

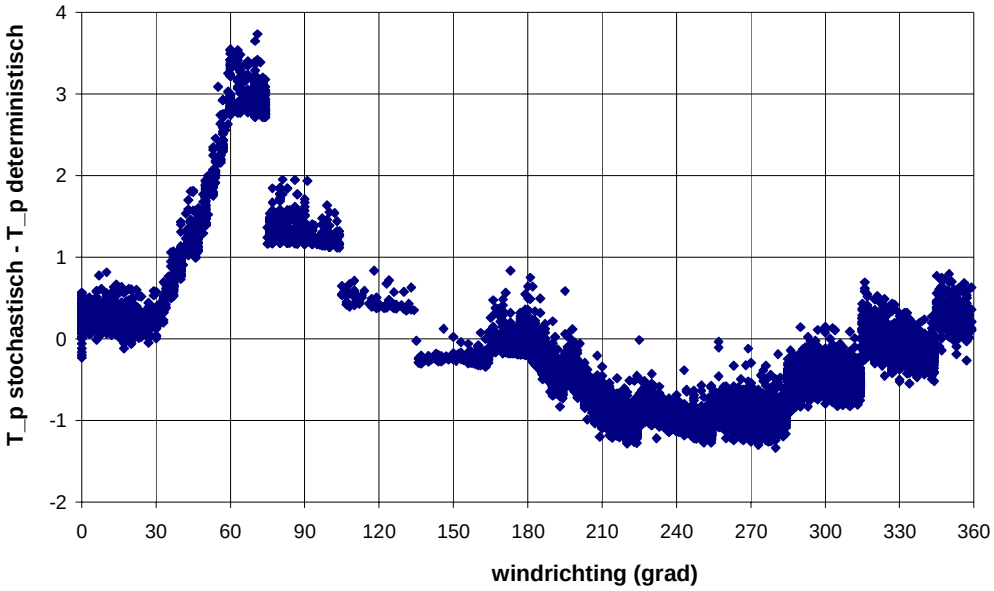
Voor locatie Petten zijn bij een zelfde analyse dezelfde bevindingen gedaan, met uitzondering van de laatste bullet hierboven. Bij Petten zijn namelijk de westelijke windrichtingen het meest relevant, ook voor steenbekledingen op 1 m+NAP. Dit komt doordat de golfhoogtes bij Petten significant groter zijn dan bij Den Helder, waardoor ook bij hogere waterstanden de bekleding op 1 m+NAP nog belast wordt. Aangezien voor westelijke windrichtingen het verschil tussen de stochastische en deterministische piekperiode veel kleiner is dan voor oostelijke windrichtingen heeft bij Petten het gebruik van de additionele stochast vrijwel geen effect op de ontwerpdiktes van steenbekledingen. Voor locatie Vlissingen kan het effect van de additionele stochast op de bekleding op 1 m+NAP niet bepaald worden omdat er onvoldoende faalwaarnemingen zijn (zie Tabel 2.2 en Tabel 2.5).

Resumerend kan gesteld worden dat het uitbreiden van de SWAN-database met fictieve rekenresultaten voor locatie Den Helder op “ongelukkige wijze” heeft geleid tot onrealistisch hoge nearshore golfperioden voor oostelijke windrichtingen. Oorzaak hiervan ligt in het feit dat de bestaande database bij oostelijke richtingen “dummy-waarden” voor de golfhoogte en golfperiode bevat. In plaats van de hier gebruikte methode is het derhalve raadzaam om additionele SWAN-berekeningen uit te voeren opdat de SWAN-database op reguliere wijze wordt uitgebreid.

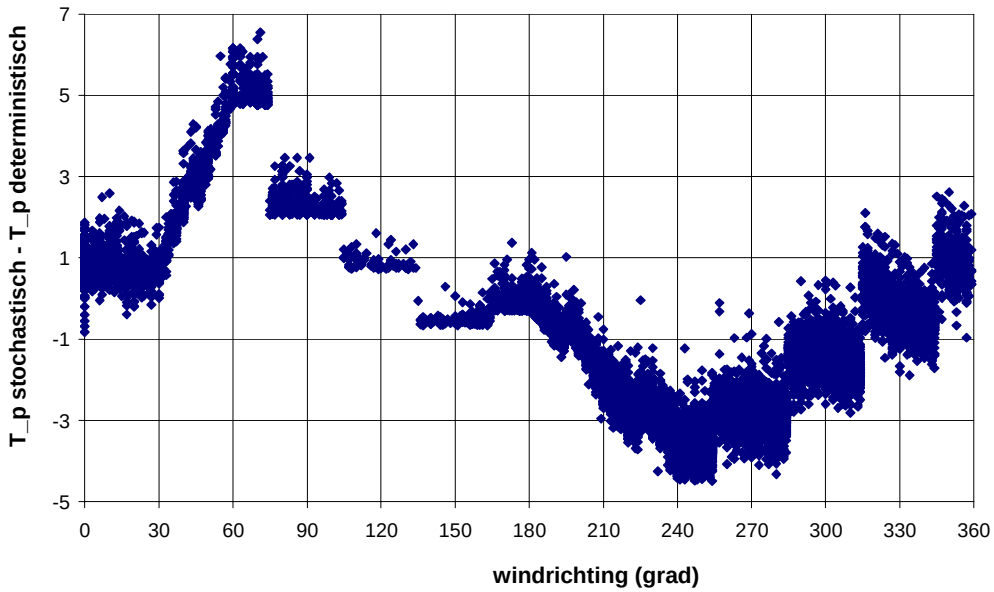
Opgemerkt wordt dat ook bij gebruik van de deterministische golfperiode de oostelijke windrichtingen het meest belastend zijn op 1 m+NAP bij Den Helder, hetgeen toch contra-intuïtief is. Dit wordt veroorzaakt door het feit dat de waterstanden bij westelijke windrichtingen na opschalen veel hoger liggen dan 1 m+NAP en derhalve niet belastend zijn op de lage delen van het talud. Dit euvel wordt ook niet verholpen door toevoeging van stormflanken (zie de analyse van paragraaf 2.4 verderop). Dit bevestigt eens te meer dat het opschalen van waargenomen stormgebeurtenissen ten behoeve van de toetsing van laaggelegen delen van een dijk een precaire kwestie is. In eerdere fases (met name fase 3, zie WL/TNO [2004a]) is dit ook al aan het licht gekomen.



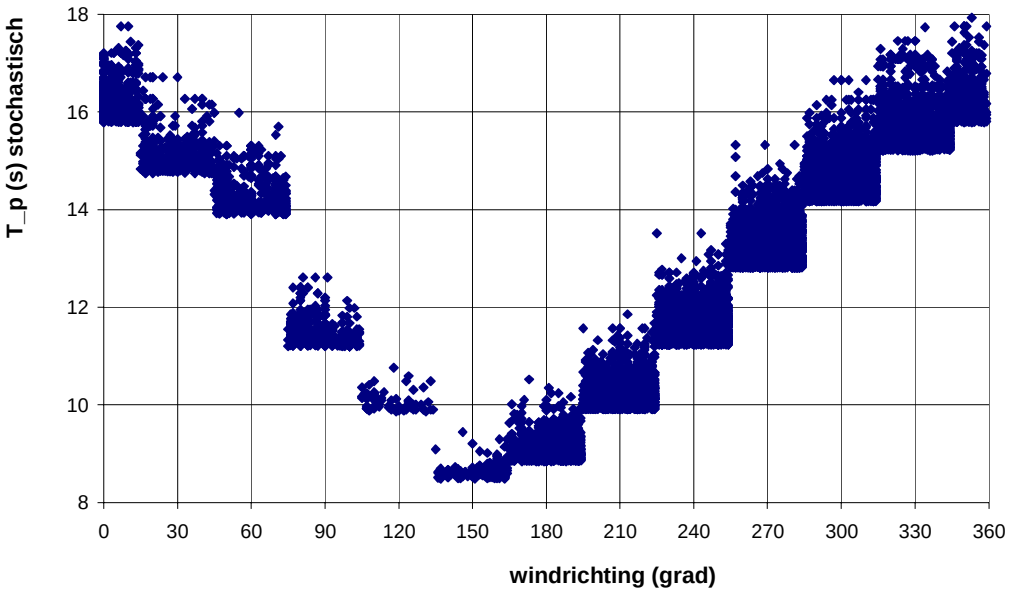
Figuur 2.1 Stochastisch gekozen **nearshore** piekperiode vs. deterministisch gekozen nearshore piekperiode voor alle uurwaarnemingen. Locatie Den Helder, herhalingstijd 4.000 jaar.



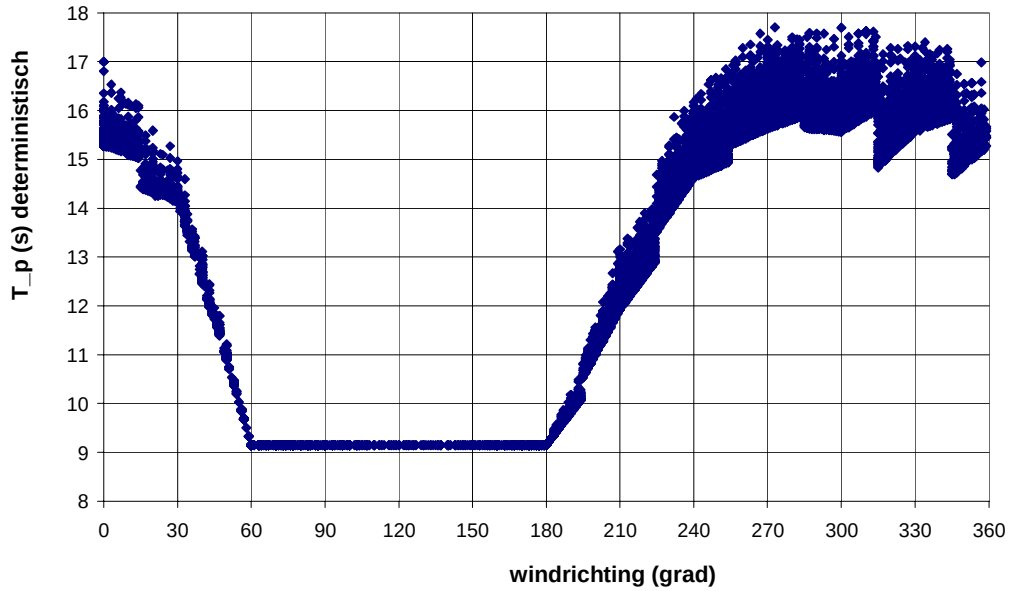
Figuur 2.2 Stochastisch gekozen **nearshore** piekperiode minus deterministisch gekozen nearshore piekperiode voor alle uurwaarnemingen uitgezet tegen de windrichting. Locatie Den Helder, herhalingstijd 4.000 jaar.



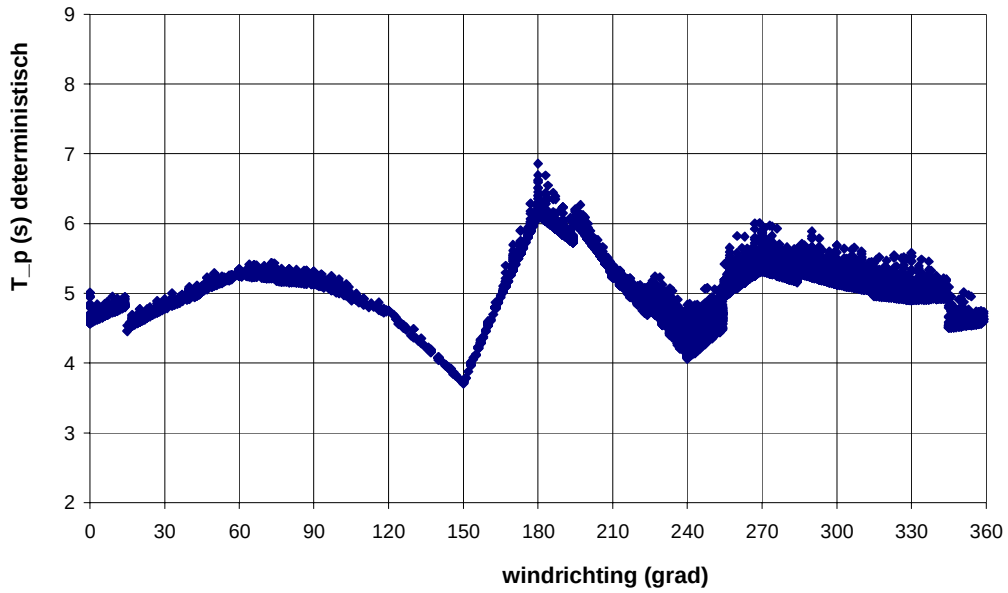
Figuur 2.3 Stochastisch gekozen **offshore** piekperiode minus deterministisch gekozen nearshore piekperiode voor alle uurwaarnemingen uitgezet tegen de windrichting. Locatie Den Helder, herhalingstijd 4.000 jaar.



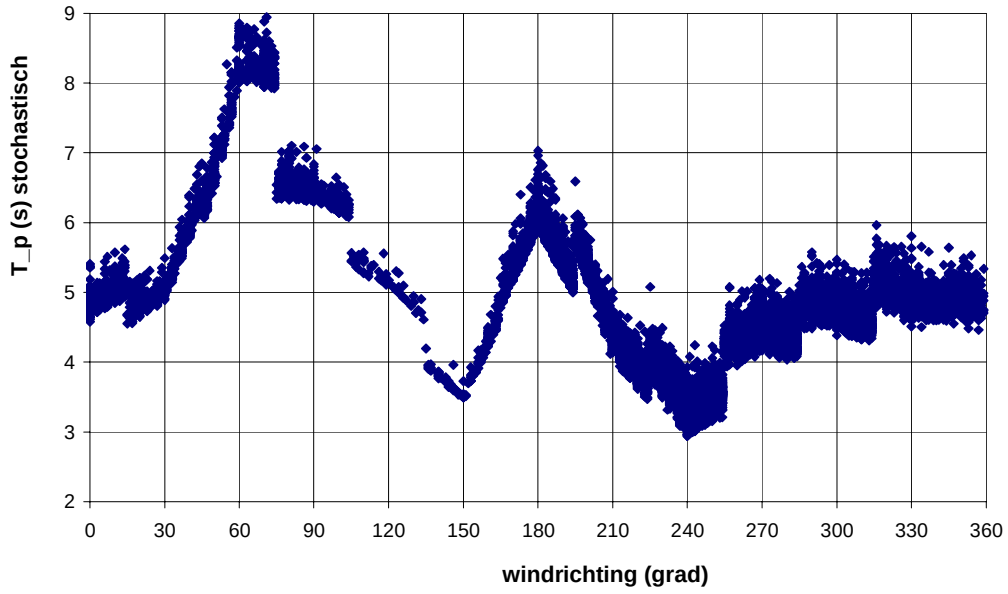
Figuur 2.4 **Stochastisch** gekozen *offshore* piekperiode voor alle uurwaarnemingen uitgezet tegen de windrichting. Locatie Den Helder, herhalingstijd 4.000 jaar.



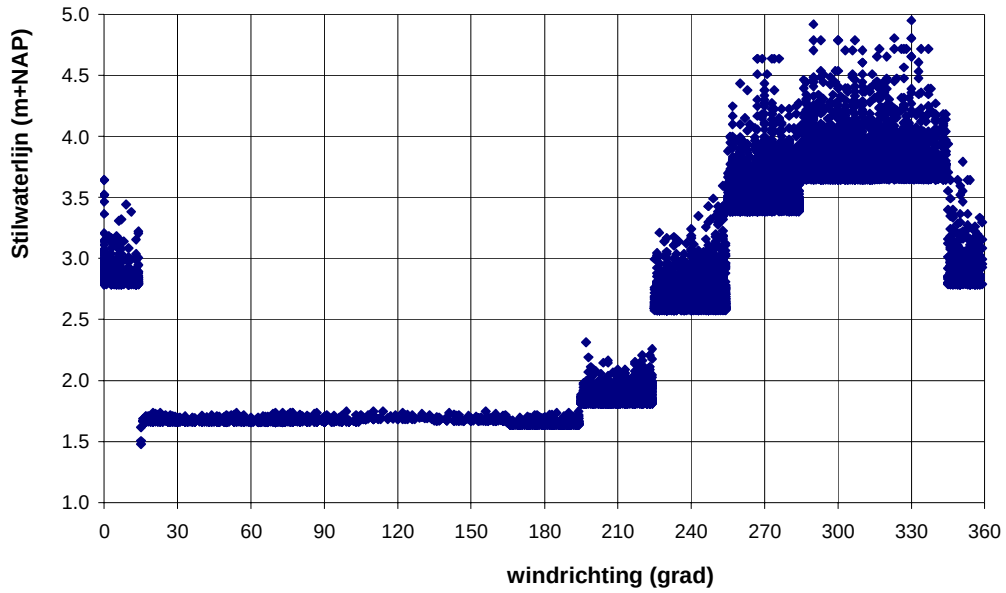
Figuur 2.5 **Deterministisch** gekozen *offshore* piekperiode voor alle uurwaarnemingen uitgezet tegen de windrichting. Locatie Den Helder, herhalingstijd 4.000 jaar.



Figuur 2.6 **Deterministisch** gekozen *nearshore* piekperiode voor alle uurwaarnemingen uitgezet tegen de windrichting. Locatie Den Helder, herhalingstijd 4.000 jaar.



Figuur 2.7 **Stochastisch** gekozen *nearshore* piekperiode voor alle uurwaarnemingen uitgezet tegen de windrichting. Locatie Den Helder, herhalingstijd 4.000 jaar.



Figuur 2.8 Stilwaterlijn voor alle uurwaarnemingen uitgezet tegen de windrichting. Locatie Den Helder, herhalingstijd 4.000 jaar.

Tabel 2.8 Relatie tussen de offshore wind en waterstand enerzijds en golfhoogte, golfperiode anderzijds voor locatie Den Helder.

windrichting	windsnelheid.	waterstand	golfhoogte	golfperiode
30	15	1	3.25	8.81
30	15	3	3.25	8.81
30	15	5	3.25	8.81
30	20	1	5.57	11.77
30	20	3	5.57	11.77
30	20	5	5.57	11.77
30	25	1	7.14	13.92
30	25	3	7.14	13.92
30	25	5	7.14	13.92
30	35	1	10.20	16.76
30	35	3	10.20	16.76
30	35	5	10.20	16.76
60	16	1	4.00	9.15
60	16	3	4.00	9.15
60	16	5	4.00	9.15
60	26	1	4.00	9.15
60	26	3	4.00	9.15
60	26	5	4.00	9.15
60	36	1	4.00	9.15
60	36	3	4.00	9.15
60	36	5	4.00	9.15
90	15	1	4.00	9.15
90	15	3	4.00	9.15
90	15	5	4.00	9.15
90	25	1	4.00	9.15
90	25	3	4.00	9.15
90	25	5	4.00	9.15
90	35	1	4.00	9.15
90	35	3	4.00	9.15
90	35	5	4.00	9.15
120	13	1	4.00	9.15
120	13	3	4.00	9.15
120	13	5	4.00	9.15
120	23	1	4.00	9.15
120	23	3	4.00	9.15
120	23	5	4.00	9.15
120	24	1	4.00	9.15
120	24	3	4.00	9.15
120	24	5	4.00	9.15
120	33	1	4.00	9.15
120	33	3	4.00	9.15
120	33	5	4.00	9.15
150	15	1	4.00	9.15
150	15	3	4.00	9.15
150	15	5	4.00	9.15
150	25	1	4.00	9.15
150	25	3	4.00	9.15
150	25	5	4.00	9.15
150	35	1	4.00	9.15
150	35	3	4.00	9.15
150	35	5	4.00	9.15
180	13	1	4.00	9.10
180	13	3	4.00	9.10
180	13	5	4.00	9.10
180	23	1	4.00	9.15
180	23	3	4.00	9.15

windrichting	windsnelheid.	waterstand	golfhoogte	golfperiode
180	23	5	4.00	9.15
180	28	1	4.00	9.15
180	28	3	4.00	9.15
180	28	5	4.00	9.15
180	29	1	4.00	9.15
180	29	3	4.00	9.15
180	29	5	4.00	9.15
180	38	1	4.00	9.15
180	38	3	4.00	9.15
180	38	5	4.00	9.15
210	18	1	3.66	9.36
210	18	3	3.66	9.36
210	18	5	3.66	9.36
210	28	1	5.09	11.19
210	28	3	5.09	11.19
210	28	5	5.09	11.19
210	33	1	5.84	12.41
210	33	3	5.84	12.41
210	33	5	5.84	12.41
210	34	1	5.99	12.66
210	34	3	5.99	12.66
210	34	5	5.99	12.66
210	43	1	7.49	13.95
210	43	3	7.49	13.95
210	43	5	7.49	13.95
240	16	1	4.10	9.94
240	16	3	4.10	9.94
240	16	5	4.10	9.94
240	21	1	4.81	10.84
240	21	3	4.81	10.84
240	21	5	4.81	10.84
240	26	1	6.18	12.49
240	26	3	6.18	12.49
240	26	5	6.18	12.49
240	31	1	7.53	13.98
240	31	3	7.53	13.98
240	31	5	7.53	13.98
240	36	1	8.61	15.53
240	36	3	8.61	15.53
240	36	5	8.61	15.53
240	37	1	8.88	15.92
240	37	3	8.88	15.92
240	37	5	8.88	15.92
240	46	1	10.91	17.46
240	46	3	10.91	17.46
240	46	5	10.91	17.46
270	17	1	4.99	11.06
270	17	3	4.99	11.06
270	17	5	4.99	11.06
270	22	1	5.85	12.10
270	22	3	5.85	12.10
270	22	5	5.85	12.10
270	27	1	7.15	13.58
270	27	3	7.15	13.58
270	27	5	7.15	13.58
270	32	1	8.45	14.97
270	32	3	8.45	14.97
270	32	5	8.45	14.97
270	37	1	9.53	16.51
270	37	3	9.53	16.51
270	37	5	9.53	16.51

windrichting	windsnelheid.	waterstand	golfhoogte	golfperiode
270	38	1	9.82	16.91
270	38	3	9.82	16.91
270	38	5	9.82	16.91
270	47	1	11.70	18.24
270	47	3	11.70	18.24
270	47	5	11.70	18.24
285	17	1	5.30	11.45
285	17	3	5.30	11.45
285	17	5	5.30	11.45
285	22	1	6.21	12.52
285	22	3	6.21	12.52
285	22	5	6.21	12.52
285	27	1	7.52	13.97
285	27	3	7.52	13.97
285	27	5	7.52	13.97
285	32	1	8.77	15.30
285	32	3	8.77	15.30
285	32	5	8.77	15.30
285	37	1	9.71	16.70
285	37	3	9.71	16.70
285	37	5	9.71	16.70
285	38	1	10.00	17.10
285	38	3	10.00	17.10
285	38	5	10.00	17.10
285	47	1	12.03	18.56
285	47	3	12.03	18.56
285	47	5	12.03	18.56
300	16	1	4.28	10.16
300	16	3	4.28	10.16
300	16	5	4.28	10.16
300	21	1	5.42	11.59
300	21	3	5.42	11.59
300	21	5	5.42	11.59
300	26	1	6.98	13.39
300	26	3	6.98	13.39
300	26	5	6.98	13.39
300	31	1	8.35	14.87
300	31	3	8.35	14.87
300	31	5	8.35	14.87
300	36	1	9.44	16.42
300	36	3	9.44	16.42
300	36	5	9.44	16.42
300	37	1	9.76	16.85
300	37	3	9.76	16.85
300	37	5	9.76	16.85
300	46	1	11.85	18.39
300	46	3	11.85	18.39
300	46	5	11.85	18.39
315	14	1	2.84	8.24
315	14	3	2.84	8.24
315	14	5	2.84	8.24
315	24	1	7.39	13.84
315	24	3	7.39	13.84
315	24	5	7.39	13.84
315	29	1	7.98	14.47
315	29	3	7.98	14.47
315	29	5	7.98	14.47
315	34	1	9.36	16.33
315	34	3	9.36	16.33
315	34	5	9.36	16.33
315	35	1	9.69	16.78

windrichting	windsnelheid.	waterstand	golfhoogte	golfperiode
315	35	3	9.69	16.78
315	35	5	9.69	16.78
315	44	1	11.75	18.29
315	44	3	11.75	18.29
315	44	5	11.75	18.29
330	16	1	3.83	9.57
330	16	3	3.83	9.57
330	16	5	3.83	9.57
330	21	1	5.85	12.10
330	21	3	5.85	12.10
330	21	5	5.85	12.10
330	26	1	6.72	13.09
330	26	3	6.72	13.09
330	26	5	6.72	13.09
330	31	1	9.21	16.18
330	31	3	9.21	16.18
330	31	5	9.21	16.18
330	32	1	9.53	16.61
330	32	3	9.53	16.61
330	32	5	9.53	16.61
330	41	1	12.15	18.68
330	41	3	12.15	18.68
330	41	5	12.15	18.68
360	13	1	2.19	7.24
360	13	3	2.19	7.24
360	13	5	2.19	7.24
360	18	1	5.10	11.20
360	18	3	5.10	11.20
360	18	5	5.10	11.20
360	23	1	7.15	13.57
360	23	3	7.15	13.57
360	23	5	7.15	13.57
360	28	1	8.57	15.50
360	28	3	8.57	15.50
360	28	5	8.57	15.50
360	38	1	11.78	18.31
360	38	3	11.78	18.31
360	38	5	11.78	18.31

2.4 Effect van het gebruik van stormflanken

Bij locatie Vlissingen worden voor geen van de doorgerekende herhalingstijden voldoende faalwaarnemingen waargenomen indien het dijkdeel tussen 1,0 m+NAP en 1,5 m+NAP wordt getoetst (zie Tabel 2.1, Tabel 2.2 en Tabel 2.5). Gedurende elk uur van alle opgeschaalde stormgebeurtenissen ligt de waterstand hoger dan 1,5 m+NAP en dat betekent dat dit stuk bekleding niet belast wordt. In feite volgt hieruit dat dit deel van de dijk niet zal falen, ongeacht de sterkte van de bekleding, hetgeen niet realistisch is. Hier wreekt zich het feit dat na opschaling van de stormgebeurtenissen de waterstand aan het begin en eind van de storm relatief hoog ligt en dat het feitelijke begin en eind van de storm eigenlijk wordt overgeslagen. In fase 3 van dit project is daarom een optie toegevoegd aan het demonstratiemodel waarmee het begin en eind van de opgeschaalde stormen verlengd worden met extra “flanken” waarbij het verloop van de waterstand naar 0 wordt doorgetrokken.

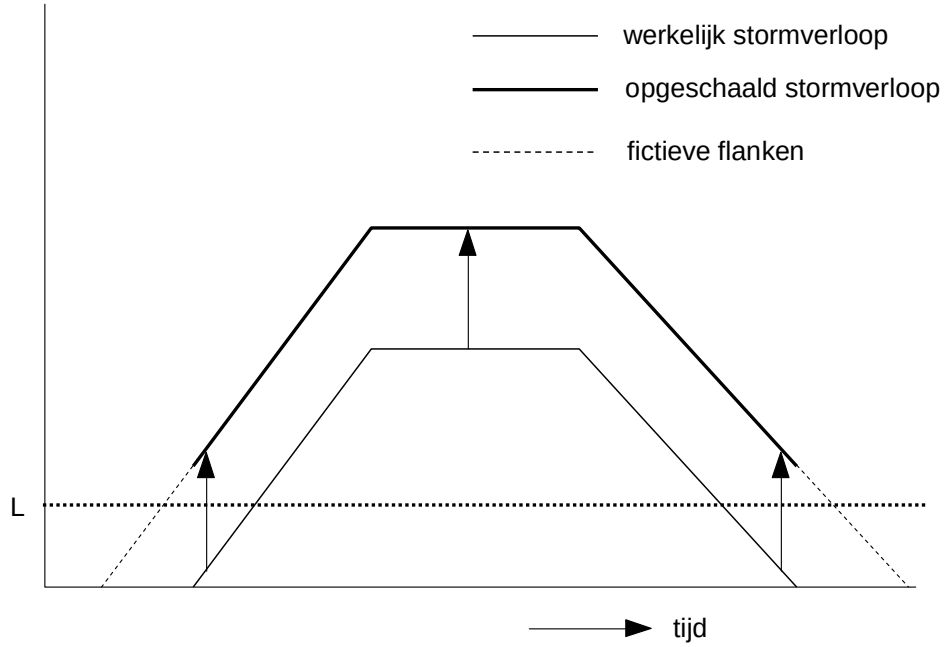
Ten behoeve van de volledigheid van het onderhavige rapport zijn de exercities van paragraaf 2.2 en paragraaf 2.3 herhaald, maar dan mét gebruik van deze stormflanken. De duur van de stormflanken is daarbij gelijk gekozen aan 15 uur. In fase 3 van deze studie zijn ook exercities uitgevoerd met stormflanken van 5 en 10 uur, maar met name bij 5 uur waren de uitkomsten inconsistent. Voor herhalingstijden van 4.000 en 10.000 jaar was de berekende steendikte namelijk kleiner dan voor herhalingstijden van 1.000 en 2.000 jaar (0,12 m resp. 0,16 m). Daaruit bleek dat de grootte van de belasting gedurende de flanken enigszins willekeurig is. Het kan bijvoorbeeld zijn dat bij een herhalingstijd van 10.000 jaar de flank zo steil is dat gedurende de tijdstap van een uur (die als basis dient in HYDRA-K) de waterstand als het ware over het getoetste deel heen “springt”, terwijl dat bij een herhalingstijd van 1.000 jaar wellicht niet gebeurt. Maar ook als de waterstand er niet overheen springt, is het van belang te weten waar exact de waterstand op het getoetste dijkdeel terecht komt, omdat dat de grootte van de belasting bepaalt. Door een langere flankduur te nemen (van 15 uur bijvoorbeeld) wordt dit effect minder relevant omdat de stormflanken minder steil worden en er zo gedurende de flank meerdere tijdstappen sprake is van belasting op het getoetste dijkdeel.

De resultaten van de exercities staan weergegeven in Tabel 2.9 (piekperiode niet als stochast) en Tabel 2.10 (piekperiode wel als stochast). Om de invloed van het gebruik van de stormflanken te bepalen moeten deze tabellen vergeleken worden met Tabel 2.2 resp. Tabel 2.5. Onderstaand zijn de belangrijkste bevindingen uit die vergelijking weergegeven.

- Door toevoeging van de flanken wordt telkens het primaire doel bereikt, nl. dat voor Vlissingen op 1 m+NAP nu voldoende waarnemingen beschikbaar zijn en derhalve een ontwerpdikte voor steenbekleding bepaald wordt.
- De overige resultaten voor steenbekledingen worden in het geheel niet beïnvloed door toevoeging van de flanken; de resultaten voor Den Helder en Petten zijn onveranderd gebleven en datzelfde geldt voor locatie Vlissingen op 3 m+NAP en 5 m+NAP.
- Voor grasbekledingen is het effect van de stormflanken ook klein, maar in 2 gevallen heeft het wel tot een andere uitkomst in het ontwerp geleid (zie Tabel 2.11 en Tabel 2.12). In beide gevallen betrof het de grasmat op 5 m+NAP bij locatie Petten.

Met name de combinatie van de laatste twee bevindingen is interessant: Voor relatief laaggelegen steenbekledingen (op 1 m+NAP en 3 m+NAP) hebben de toegevoegde stormflanken bij de locatie Petten geen effect op de ontwerpdikte, maar bij de hoger gelegen grasmat op 5 m+NAP een enkele keer wel. Om dit te verklaren moet rekening gehouden worden met het feit dat de flanken als volgt gekarakteriseerd zijn:

- De duur van de flanken (voor en na) is 15 uur.
- De waterstand loopt op van 0 m+NAP tot het niveau van de eerste resp. laatste (opgeschaalde) waarneming van de storm.
- De windsnelheid loopt op van 0 m/s tot het niveau van de eerste resp. laatste (opgeschaalde) waarneming van de storm.



Figuur 2.9 Schematische weergave van het opschalen van het stormverloop, hetgeen er toe leidt dat taludniveau L niet belast wordt. Na toevoegen van de fictieve flanken wordt niveau L wel belast.

Voor steenbekleding is de piekbelasting doorslaggevend voor het ontwerp. Deze vindt in de regel niet plaats gedurende de stormflanken omdat de windsnelheid zich dan in de “opbouwende” (afbouwende) fase bevindt en hetzelfde geldt voor de golfhoogte. De duur van de belasting wordt mogelijk wel verlengd door gebruik van de stormflanken en dat is relevant voor grasbekleding. Bij locatie Petten zijn de golfhoogten dusdanig groot dat dit effect ook op 5 m+NAP merkbaar is en dus kan het gebeuren dat de benodigde kwaliteit van de grasmat wijzigt als gevolg van het gebruik van de stormflanken.

Tabel 2.9 Benodigde steendikte (m) en graskwaliteit voor 3 locaties, 5 verschillende dijkdelen en 8 herhalingstijden (T). Stormgebeurtenissen zijn verlengd met flanken van 15 uur. De golfperiode is niet als stochast meegenomen.

Vlissingen					
T (jaar)	**steen, 1 m+NAP**	**steen, 3 m+NAP**	**steen, 5 m+NAP**	**gras, 5 m+NAP**	**gras, 7 m+NAP**
10	0.15	0.22	0.17	slecht	slecht
30	0.16	0.24	0.19	matig	slecht
100	0.16	0.26	0.21	goed	slecht
300	0.17	0.27	0.22	goed	slecht
1000	0.17	0.29	0.24	voldoet niet	slecht
2000	0.17	0.30	0.25	voldoet niet	slecht
4000	0.18	0.31	0.29	voldoet niet	slecht
10000	0.17	0.32	0.31	voldoet niet	matig
Den Helder					
T (jaar)	**steen, 1 m+NAP**	**steen, 3 m+NAP**	**steen, 5 m+NAP**	**gras, 5 m+NAP**	**gras, 7 m+NAP**
10	0.21	0.18	O.F.	slecht	slecht
30	0.23	0.21	O.F.	slecht	slecht
100	0.24	0.24	O.F.	slecht	slecht
300	0.25	0.25	0.20	matig	slecht
1000	0.26	0.27	0.22	matig	slecht
2000	0.27	0.28	0.23	matig	slecht
4000	0.27	0.29	0.23	goed	slecht
10000	0.28	0.30	0.24	goed	slecht
Petten					
T (jaar)	**steen, 1 m+NAP**	**steen, 3 m+NAP**	**steen, 5 m+NAP**	**gras, 5 m+NAP**	**gras, 7 m+NAP**
10	0.37	0.30	0.30	matig	slecht
30	0.41	0.41	0.33	goed	matig
100	0.45	0.45	0.36	goed	matig
300	0.48	0.48	0.38	voldoet niet	goed
1000	0.51	0.51	0.41	voldoet niet	goed
2000	0.53	0.53	0.43	voldoet niet	goed
4000	0.55	0.55	0.44	voldoet niet	voldoet niet
10000	0.58	0.58	0.47	voldoet niet	voldoet niet

O.F. betekent: onvoldoende faalwaarnemingen
“voldoet niet” betekent: zelfs een grasmat met goede kwaliteit zal niet voldoen

Tabel 2.10 Benodigde steendikte (m) en graskwaliteit voor 3 locaties, 5 verschillende dijkdelen en 8 herhalingstijden (T). Stormgebeurtenissen zijn verlengd met flanken van 15 uur. De golfperiode is als stochast meegenomen.

Vlissingen					
T (jaar)	**steen, 1 m+NAP**	**steen, 3 m+NAP**	**steen, 5 m+NAP**	**gras, 5 m+NAP**	**gras, 7 m+NAP**
10	0.14	0.21	0.15	slecht	slecht
30	0.15	0.23	0.17	matig	slecht
100	0.17	0.24	0.19	matig	slecht
300	0.17	0.26	0.21	goed	slecht
1000	0.18	0.27	0.22	voldoet niet	slecht
2000	0.19	0.28	0.25	voldoet niet	slecht
4000	0.19	0.29	0.28	voldoet niet	slecht
10000	0.19	0.30	0.29	voldoet niet	slecht
Den Helder					
T (jaar)	**steen, 1 m+NAP**	**steen, 3 m+NAP**	**steen, 5 m+NAP**	**gras, 5 m+NAP**	**gras, 7 m+NAP**
10	0.22	0.18	O.F.	slecht	slecht
30	0.25	0.21	O.F.	slecht	slecht
100	0.27	0.23	O.F.	slecht	slecht
300	0.29	0.25	0.2	slecht	slecht
1000	0.31	0.26	0.21	matig	slecht
2000	0.32	0.27	0.22	matig	slecht
4000	0.33	0.28	0.23	matig	slecht
10000	0.34	0.29	0.24	goed	slecht
Petten					
T (jaar)	**steen, 1 m+NAP**	**steen, 3 m+NAP**	**steen, 5 m+NAP**	**gras, 5 m+NAP**	**gras, 7 m+NAP**
10	0.37	0.30	0.3	goed	slecht
30	0.41	0.41	0.33	goed	matig
100	0.44	0.44	0.36	goed	goed
300	0.48	0.48	0.38	voldoet niet	goed
1000	0.52	0.52	0.41	voldoet niet	goed
2000	0.53	0.53	0.43	voldoet niet	voldoet niet
4000	0.55	0.55	0.44	voldoet niet	voldoet niet
10000	0.58	0.58	0.47	voldoet niet	voldoet niet

O.F. betekent: onvoldoende faalwaarnemingen
“voldoet niet” betekent: zelfs een grasmat met goede kwaliteit zal niet voldoen

Tabel 2.11 Verschil tussen de benodigde kwaliteit van de grasmat zonder gebruik van stormflanken (Tabel 2.2) en met gebruik van stormflanken (Tabel 2.9) voor 3 locaties, 2 verschillende dijkdelen en 8 herhalingstijden (T). De berekeningen waar verschillen zijn gevonden tussen zijn grijs gemarkeerd. De golfperiode is niet als stochast meegenomen.

Vlissingen				
	gras, 5 m+NAP		**gras, 7 m+NAP**	
T (jaar)	**geen stormflanken**	**wel stormflanken**	**geen stormflanken**	**wel stormflanken**
10	slecht	slecht	slecht	slecht
30	matig	matig	slecht	slecht
100	goed	goed	slecht	slecht
300	goed	goed	slecht	slecht
1000	voldoet niet	voldoet niet	slecht	slecht
2000	voldoet niet	voldoet niet	slecht	slecht
4000	voldoet niet	voldoet niet	slecht	slecht
10000	voldoet niet	voldoet niet	matig	matig
Den Helder				
	gras, 5 m+NAP		**gras, 7 m+NAP**	
T (jaar)	**geen stormflanken**	**wel stormflanken**	**geen stormflanken**	**wel stormflanken**
10	slecht	slecht	slecht	slecht
30	slecht	slecht	slecht	slecht
100	slecht	slecht	slecht	slecht
300	matig	matig	slecht	slecht
1000	matig	matig	slecht	slecht
2000	matig	matig	slecht	slecht
4000	goed	goed	slecht	slecht
10000	goed	goed	slecht	slecht
Petten				
	gras, 5 m+NAP		**gras, 7 m+NAP**	
T (jaar)	**geen stormflanken**	**wel stormflanken**	**geen stormflanken**	**wel stormflanken**
10	matig	matig	slecht	slecht
30	goed	goed	matig	matig
100	goed	goed	matig	matig
300	goed	voldoet niet	goed	goed
1000	voldoet niet	voldoet niet	goed	goed
2000	voldoet niet	voldoet niet	goed	goed
4000	voldoet niet	voldoet niet	voldoet niet	voldoet niet
10000	voldoet niet	voldoet niet	voldoet niet	voldoet niet
"voldoet niet" betekent: zelfs een grasmat met goede kwaliteit zal niet voldoen				

Tabel 2.12 Verschil tussen de benodigde kwaliteit van de grasmat zonder gebruik van stormflanken (Tabel 2.5) en met gebruik van stormflanken (Tabel 2.10) voor 3 locaties, 2 verschillende dijkdelen en 8 herhalingstijden (T). De berekeningen waar verschillen zijn gevonden tussen zijn grijs gemarkeerd. De golfperiode is als stochast meegenomen.

Vlissingen				
	gras, 5 m+NAP		**gras, 7 m+NAP**	
T (jaar)	**geen stormflanken**	**wel stormflanken**	**geen stormflanken**	**wel stormflanken**
10	slecht	slecht	slecht	slecht
30	matig	matig	slecht	slecht
100	matig	matig	slecht	slecht
300	goed	goed	slecht	slecht
1000	voldoet niet	voldoet niet	slecht	slecht
2000	voldoet niet	voldoet niet	slecht	slecht
4000	voldoet niet	voldoet niet	slecht	slecht
10000	voldoet niet	voldoet niet	slecht	slecht
Den Helder				
	gras, 5 m+NAP		**gras, 7 m+NAP**	
T (jaar)	**geen stormflanken**	**wel stormflanken**	**geen stormflanken**	**wel stormflanken**
10	slecht	slecht	slecht	slecht
30	slecht	slecht	slecht	slecht
100	slecht	slecht	slecht	slecht
300	slecht	slecht	slecht	slecht
1000	matig	matig	slecht	slecht
2000	matig	matig	slecht	slecht
4000	matig	matig	slecht	slecht
10000	goed	goed	slecht	slecht
Petten				
	gras, 5 m+NAP		**gras, 7 m+NAP**	
T (jaar)	**geen stormflanken**	**wel stormflanken**	**geen stormflanken**	**wel stormflanken**
10	matig	goed	slecht	slecht
30	goed	goed	matig	matig
100	goed	goed	goed	goed
300	voldoet niet	voldoet niet	goed	goed
1000	voldoet niet	voldoet niet	goed	goed
2000	voldoet niet	voldoet niet	voldoet niet	voldoet niet
4000	voldoet niet	voldoet niet	voldoet niet	voldoet niet
10000	voldoet niet	voldoet niet	voldoet niet	voldoet niet
"voldoet niet" betekent: zelfs een grasmat met goede kwaliteit zal niet voldoen				

3 Beknopte gebruikershandleiding voor de nieuwe faalmechanismen

3.1 Inleiding

In het rapport van fase 2 is een (beknopte) gebruikershandleiding beschreven van het destijds opgeleverde demonstratiemodel van HYDRA-K. In de huidige fase is het rekenmodel uitgebreid met een aantal functionaliteiten. In dit hoofdstuk wordt beschreven hoe de nieuwe functionaliteiten door de gebruiker aangestuurd kunnen worden. Daarbij zijn tevens de functionaliteiten van fase 2 beschreven opdat de lezer zo één compacte gebruikershandleiding ter beschikking heeft. We merken op dat we ervan uitgaan dat de gebruiker bekend is met voorgaande ('reguliere') versies van HYDRA-K.

3.2 User interface

Het hoofdscherm in de user interface is uitgebreid met één extra faalmechanisme (zie Figuur 3.1) genaamd "*Algemene instabiliteit*". Onder deze noemer vallen de nieuwe faalmechanismen voor bekledingstypen gras, steenzetting en asfalt. Zodra de gebruiker dit faalmechanisme selecteert verschijnt een balk in beeld met daarop de tekst "wijzig bekleding" (zie Figuur 3.1). Zodra deze aangeklikt wordt met de muis verschijnt een nieuw scherm in beeld, waarin de bekleding van de dijk gedefinieerd en/of aangepast kan worden (zie Figuur 3.2). Dit scherm bestaat uit twee delen:

1. aan de linker zijde zijn alle knoppen en edit vensters geplaatst.
2. aan de rechter zijde is het geselecteerde profiel weergegeven met daarop aangegeven met een rode lijn de locatie van het beschouwde bekledingssegment.

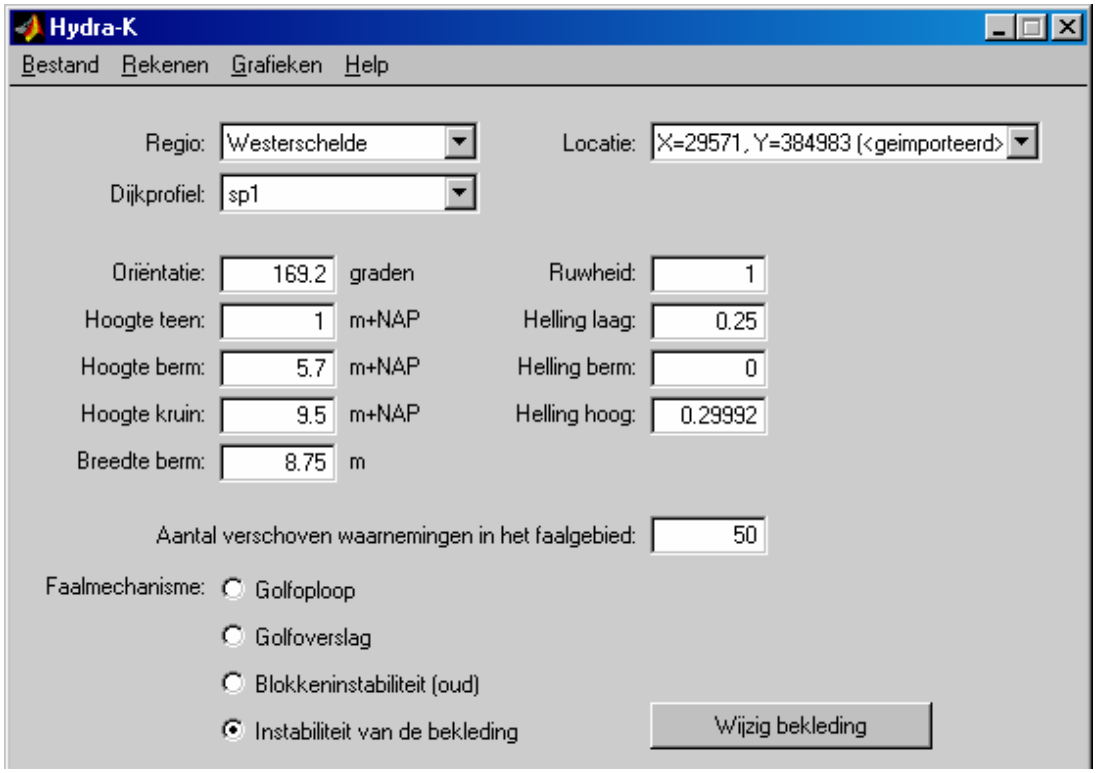
De gebruiker kan aan de linkerzijde de parameters van de bekleding, het type bekleding en de locatie van het bekledingssegment wijzigen. Tevens is het mogelijk om de bekledingssegmenten toe te voegen en de totale opbouw van de dijkbekleding naar een bestand weg te schrijven. De optie van het wegschrijven naar een bestand is in de eerste plaats toegevoegd om later eenvoudig nieuwe berekeningen met hetzelfde profiel uit te kunnen voeren. Daartoe kan het opgeslagen bestand automatisch ingelezen worden. Een bijkomende functie van het wegschrijven van de gegevens van het dijkprofiel is dat deze als invoer van de batchberekeningen vereist zijn (zie volgende paragraaf).

Opmerkingen:

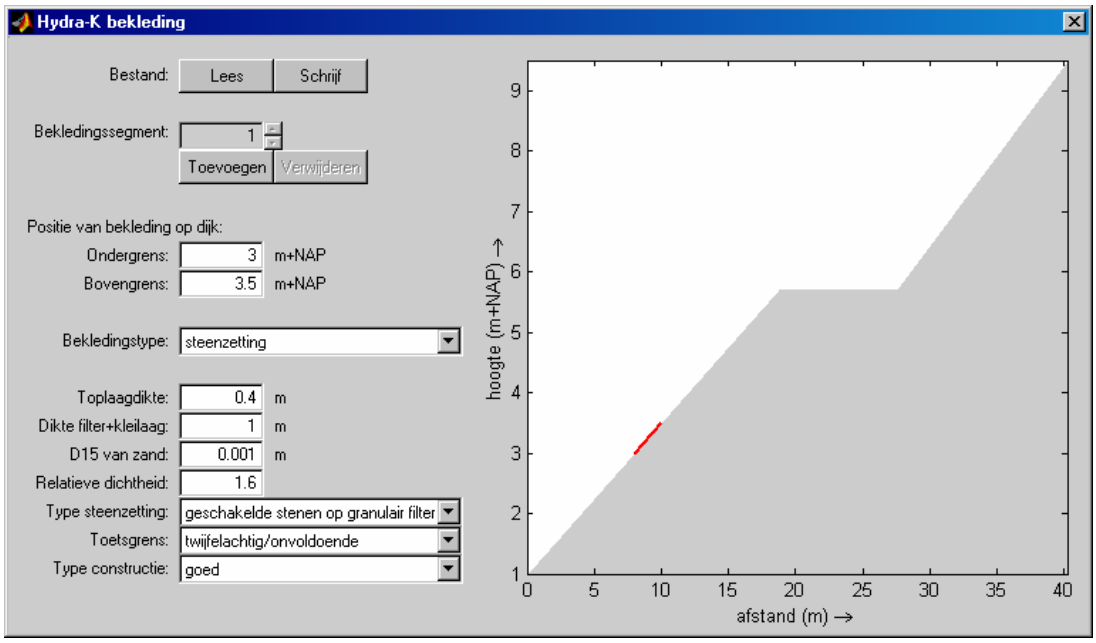
- Het demonstratiemodel biedt de mogelijkheid om faalkansen voor dijken met meerdere bekledingstypen door te rekenen. De leidraad toetsen (TAW, 1999) doet echter geen uitspraken over faalkansen van dijken die uit meerdere bekledingen bestaan. Volgens de leidraad moeten de afzonderlijke delen ook afzonderlijk getoetst worden en zolang elk van deze delen aan de gestelde eisen voldoet is voor de dijk als geheel voldoende. Om de berekeningen met HYDRA-K zoveel mogelijk te laten stroken met de leidraad, wordt

daarom aangeraden om per berekening slechts één bekledingstype door te rekenen. Voor dijken met bijvoorbeeld drie verschillende bekledingstypen impliceert dit dat in totaal drie berekeningen (ontwerpberekeningen of faalkansberekeningen) uitgevoerd moeten worden.

- De bekleding op bermen dient als apart onderdeel getoetst te worden. Dit heeft gevolgen voor de door de gebruiker in te voeren ondergrens en bovengrens van het deel van de bekleding dat getoetst gaat worden. Het is namelijk niet wenselijk dat de ondergrens onder de berm ligt en de bovengrens erboven, omdat dan feitelijk drie delen van de dijk getoetst worden die beter apart beschouwd kunnen worden. Daarom zal een foutmelding gegeven worden zodra dit het geval is en wordt de gebruiker verzocht om de invoer aan te passen.
- In het geval van tijdsduurafhankelijke belasting (d.w.z. in het geval van grasbekleding) is het ontwerppunt niet gedefinieerd. De bepaling van het ontwerppunt wordt in dat geval overgeslagen.
- Indien de gebruiker de *combinatie* van standaard profielen heeft geselecteerd, zal het eerste profiel worden weergegeven in de interface.



Figuur 3.1 Bovenste deel van het hoofdscherm van de user-interface van HYDRA-K



Figuur 3.2 De user interface voor het wijzigen van de dijkbekleding.

3.3 Invoer via het ini-bestand

In HYDRA-K wordt een aantal parameters ingesteld via het ini-bestand. In de huidige studie zijn aan dit bestand de volgende parameters toegevoegd:

- *duur_stormflanken*. In HYDRA-K is de mogelijkheid aangebracht om de (opgeschaalde) stormgebeurtenissen te verlengen, om er zorg voor te dragen dat ook de lage delen van de dijkbekleding belast worden. Het fase 3 rapport beschrijft uitgebreid waarom deze zogenaamde “stormflanken” zijn toegevoegd en wat de gevolgen zijn. Middels de parameter “duur_stormflanken” kan de lengte van deze flanken (in uren) ingevoerd worden. Indien men geen gebruik wenst te maken van deze optie dient de waarde 0 ingevuld te worden.
- *piekperiode_stochast*. Deze parameter krijgt de waarde 1 indien de piekperiode als stochast moet worden behandeld in de berekeningen en 0 indien de piekperiode niet als stochast moet worden behandeld in de berekeningen.

3.4 Batchberekeningen

In verband met het karakter van deze studie is ervoor gekozen om de parameters van de dijkbekleding door de gebruiker in een apart bestand (dus niet uit de database) te laten specificeren. Zoals eerder gezegd dient dit bestand middels de user interface aangemaakt te worden. De aansturing van een batchbestand gebeurt nog steeds via een invoerbestand, waarvan onderstaand en voorbeeld gegeven is:

```
Regio = Kust zuid
Kappa = 50
Faalmechanisme = BEKL
Herhalingstijd = 10 30 100 300 1000 2000 4000 10000
Profiel = ST,sp1
Afhankelijkheid = WAARnemingen, HOOGste waterstand
Stromingscorrectie = NIET
Vaste stilwaterlijn = NIET
Dijkopbouwbestand = mijndijk.bekleding
Uitvoerbestand = D:\test\test.txt
```

In vergelijking met voorgaande versies van HYDRA-K zijn twee dingen veranderd:

- Het faalmechanisme ‘instabiliteit bekleding’ is toegevoegd als optie en kan worden geselecteerd door BEKL te specificeren bij het keyword *Faalmechanisme*.
- De opbouw van de dijkbekleding dient gespecificeerd te worden door middel van het keyword *Dijkopbouwbestand*, bijvoorbeeld: Dijkopbouwbestand = mijndijk.bekleding waarbij het bestand mijndijk.bekleding gemaakt is met de user interface.

4 Conclusies

In fase 5 is invulling gegeven aan het werkplan dat in fase 4 is opgesteld. De volgende activiteiten zijn uitgevoerd in fase 5:

1. Implementatie van de additionele stochast (golfperiode);
2. Implementatie van de in fase 4 beschreven additionele faalmechanismen;
3. Implementatie van additionele typen steenzettingen; en
4. Uitvoeren van testberekeningen.

De testberekeningen waren in eerste instantie gericht op het bepalen van het effect van de additionele stochast (punt 1). Daarnaast zijn extra berekeningen uitgevoerd om het effect te bepalen van de berekeningswijze van de helling van het talud. Deze berekeningswijze is namelijk aangepast in vergelijking met eerdere versies van het demonstratiemodel. Aanleiding om de berekeningswijze van de helling aan te passen was het feit dat er inconsistenties dreigden te ontstaan bij implementatie van de additionele faalmechanismen (punt 2.). Bij het opstellen van het werkplan in fase 4 is namelijk gesteld dat voor een stuk bekleding op de berm de helling gelijk is aan die van het ondertalud. In de voorgaande versies van het demonstratiemodel was het zo dat als een stuk bekleding onder de berm ligt, maar de stilwaterlijn juist erboven dan zou juist de helling van het boventalud gebruikt worden. Om te voorkomen dat deze inconsistentie in het demonstratiemodel zou komen, is ervoor gekozen om altijd de helling van het ondertalud te nemen in het geval van bekleding op het ondertalud of berm, en de helling van het boventalud bij bekleding op het boventalud.

Het gevolg van de gewijzigde berekeningswijze voor de helling van een talud is dat de berekeningen van fase 3 niet meer per definitie exact gereproduceerd kunnen worden met de laatste versie van het demonstratiemodel. De verschillen zijn echter zeer klein, zoals aan de hand van testberekeningen is aangetoond. Voor steenbekledingen is het verschil tussen de twee modelversies in de regel gelijk aan 0 of 1 cm. Uitzondering op deze regel vormt de locatie Petten bij een herhalingstijd van 10.000 jaar waar een iets groter verschil van 3 cm optreedt. Dit laatste wordt veroorzaakt door de dominantie van het faalmechanisme afschuiving bij deze locatie en herhalingstijd. Voor grasbekleding zijn de uitkomsten gelijk gebleven, op twee uitzonderingen na bij locatie Petten.

Het effect van het toevoegen van de additionele stochast (golfperiode) is geschat op basis van vergelijkende berekeningen (berekeningen mét en zonder additionele stochast). Uit de berekeningen blijkt dat voor grasbekleding de uitkomsten merendeels gelijk zijn gebleven. Voor steenbekledingen bedraagt het verschil tussen de twee opties in de regel 0, 1 of 2 cm. Uitzondering op deze regel vormt locatie Den Helder op niveau 1 m+NAP, waar de verschillen kunnen oplopen tot maximaal 6 cm. Nadere analyse laat zien dat bij relatief lage waterstanden (die maatgevend zijn voor het stuk bekleding op 1 m+NAP) te Den Helder er een relatief groot verschil is in de deterministische gekozen en de stochastisch gekozen golfperiode. Bij hogere waterstanden zijn deze verschillen beduidend kleiner, hetgeen verklaart waarom voor 3 m+NAP en 5 m+NAP nauwelijks verschillen in ontwerpdikte bestaan.

De diepere oorzaak van de grote verschillen bij locatie Den Helder ligt in het feit dat de bestaande database bij oostelijke richtingen “dummy-waarden” voor de golfhoogte en golfperiode bevat. Dit heeft namelijk op “ongelukkige wijze” geleid tot onrealistisch hoge nearshore golfperioden voor oostelijke windrichtingen bij het uitbreiden van de SWAN-database met fictieve rekenresultaten ten behoeve van de berekeningen met de additionele stochast. Bij het toetsen van een stuk bekleding op 1 m+NAP bij locatie Den Helder zijn juist deze oostelijke windrichtingen het meest relevant omdat de (opgeschaalde) stilwaterlijn bij deze windrichtingen rond deze hoogte liggen, terwijl de stilwaterlijn voor westelijke windrichtingen hoger liggen.

Aangeraden wordt om in de toekomst ten behoeve van de additionele stochast de uitbreiding van de SWAN-database op reguliere wijze te voltrekken middels “echte” SWAN-berekeningen in plaats van de fictieve rekenresultaten die in de huidige studie gebruikt zijn. Op die manier zullen de nearshore piekperiodes voor oostelijke windrichtingen bij locatie Den Helder niet meer zo onrealistisch hoog zijn. Het feit blijft echter dat ook bij gebruik van de deterministische golfperiode (zonder uitgebreide database) de oostelijke windrichtingen het meest belastend zijn op 1 m+NAP bij Den Helder, hetgeen toch contra-intuïtief is. Dit wordt veroorzaakt door het feit dat de waterstanden bij westelijke windrichtingen na opschalen (veel) hoger liggen dan 1 m+NAP en derhalve niet belastend zijn op de lage delen van het talud. Dit euvel wordt ook niet verholpen door toevoeging van stormflanken, gezien het feit dat deze geen invloed hebben op de ontwerpdikte van de steenbekleding. Dit bevestigt eens te meer dat het opschalen van waargenomen stormgebeurtenissen ten behoeve van de toetsing van laaggelegen delen van een dijk een precaire kwestie is en nader onderzoek behoeft.

5 Referenties

WL/TNO, 2003a: *Haalbaarheidsstudie HYDRA-K; fase 1: Plan van aanpak*. WL | Delft Hydraulics en TNO-Bouw in opdracht van RIKZ, oktober 2003.

WL/TNO, 2003b: *Haalbaarheidsstudie HYDRA-K; fase 2: Inbouwen van faalmechanismen en bekledingstypen in het demonstratiemodel*. WL | Delft Hydraulics en TNO-Bouw in opdracht van RIKZ, november 2003.

WL/TNO, 2004a: *Haalbaarheidsstudie HYDRA-K; fase 3: Indicatieve bepaling van de nauwkeurigheid*. WL | Delft Hydraulics en TNO-Bouw in opdracht van RIKZ, februari 2004.

WL/TNO, 2004b: *Haalbaarheidsstudie HYDRA-K; fase 4: Alternatieven (work arounds) voor het inbouwen van faalmechanismen*. WL | Delft Hydraulics en TNO-Bouw in opdracht van RIKZ, maart 2004.

A Aanpassingen in de code

A1 Overzicht

Deze bijlage geeft een overzicht van alle nieuwe respectievelijk gewijzigde Hydra-K routines ten behoeve van de implementatie van de nieuwe faalmechanismen voor bekledingen en het toevoegen van de golfperiode als extra stochast.

Codewijzigingen t.b.v. nieuwe faalmechanismen

Routines gerelateerd aan rekenwerk:

Nieuw:

- General/MeerdereBekledingen
- General/StormDuurAfhankelijk
- General/controleerBekleding
- Stat/StatBerekenBekleding
- Stat/StatBerekenOploopniveau
- Stat/StatBerekenZAsfalt
- Stat/StatBerekenZGras
- Stat/StatBerekenZSteenzetting
- Stat/stormFlanken

Gewijzigd:

- General/GlobalConstants
- General/InitialiseerGlobalConstants
- Stat/StatBerekenBloksterkte
- Stat/StatBerekenFaalfrequentie
- Stat/StatBerekenGolfoverslag
- Stat/StatBerekenKruinhoogte
- Stat/StatBerekenOntwerppunt
- Stat/StatBerekenZ
- Stat/StatBerekenZBlokinstabiliteit
- Stat/StatBerekenZuitNs
- Stat/StatDataHerhalingstijdFalen
- Stat/StatFaalFreqMinimizeFunc
- Stat/statGrensTContour
- Stat/StatKruinhBloksMinimizeFunc

Routines gerelateerd aan in- en uitvoer:

Nieuw:

- General/LeesBekleding
- General/printBekleding

Gewijzigd:

- General/PrintWaarden

Routines gerelateerd aan de opbouw en afhandeling van de user interface:

Nieuw:

- Ui/CBbekleding

Gewijzigd:

- Ui/CBfaalmechanisme
- Ui/FGhydra_k
- Ui/FGkruinhoogteBlokdikte\
- Ui/CBmenu

Routines specifiek voor de batch mode:

Gewijzigd:

- Batch/BatchLeesInvoerBestand
- Batch/BatchRekenen

Codewijzigingen t.b.v. extra stochast

Routines gerelateerd aan rekenwerk:

Nieuw:

- General/PiekPeriode

Gewijzigd:

- Stat/StatWeibullFreq
- Stat/StatWeibullInv
- Transformeer/Model
- Transformeer/Offshore2Nearshore
- Transformeer/RichtingTransformArray
- Transformeer/TransformeerNaarNearshore
- Transformeer/VulTransFormMatrix

Routines gerelateerd aan in- en uitvoer:

Gewijzigd:

- Stat/StatLaadStatistiek
- Transformeer/ControleerRand2001Data
- Transformeer/LeesDataVanRand2001

A2 Nieuwe routines

In deze bijlage wordt de code getoond van de nieuwe routines. Zie bijlage A1 voor de context waarin deze routines gebruikt worden.

General/controleerBekleding

```
function [Fout,Foutmelding] = controleerBekleding (Waarden)

%CONTROLEERBEKLEDING  controleer de ligging van de bekleding
%  AANROEP
%     controleerBekleding (Waarden)
%  INVOER
%     Waarden:    structure-array met waarden van faalmechanisme,
%                 profielen en bekleding

%  EXTERNE FUNCTIES
%     Uitvoer

% Copyright © 2000 Rijkswaterstaat/RIKZ All Rights Reserved.

GlobalConstants

Bekleding = [];
if isfield(Waarden,'faalmechanisme') & Waarden.faalmechanisme == FMINSTABBEKLED
   Bekleding = Waarden.dijkopbouw;
   Profiel = Waarden.profiel;
elseif isfield(Waarden,'naam') & strcmp(Waarden.naam,'instabiliteit van de
bekleding')
   Bekleding = Waarden.dijkopbouw;
   Profiel = Waarden.profiel;
end
Fout=0;
Foutmelding='';
VorigProfiel='';
if ~isempty(Bekleding)
   Bekleding = Waarden.dijkopbouw;
   for i=1:length(Bekleding)
      if Bekleding(i).ondergrens>Bekleding(i).bovengrens
         Foutmelding = sprintf('Ondergrens van segment %i boven bovengrens!',i);
         Fout=1;
      end
      BekledingsLocatie = '';
      for p=1:length(Profiel)
         z_berm_o = Profiel(p).hoogte_berm;
         z_berm_b = Profiel(p).hoogte_berm + ...
                     Profiel(p).breedte_berm*Profiel(p).helling_berm;
         if Bekleding(i).bovengrens<=z_berm_b & Bekleding(i).ondergrens>=z_berm_o
            if isempty(BekledingsLocatie)
               BekledingsLocatie = 'Berm';
               VorigProfiel=Profiel(p).ident;
            elseif ~strcmp(BekledingsLocatie,'Berm')
               Foutmelding = sprintf('Segment %i op berm voor profiel %s en op %s
voor profiel %s!',i,Profiel(p).ident,lower(BekledingsLocatie),VorigProfiel);
               Fout=1;
            end
         elseif Bekleding(i).ondergrens>=z_berm_b
            if isempty(BekledingsLocatie)
               BekledingsLocatie = 'Boventalud';
               VorigProfiel=Profiel(p).ident;
            elseif ~strcmp(BekledingsLocatie,'Boventalud')
               Foutmelding = sprintf('Segment %i op boventalud voor profiel %s en op
%s voor profiel %s!',i,Profiel(p).ident,lower(BekledingsLocatie),VorigProfiel);
               Fout=1;
            end
         elseif Bekleding(i).bovengrens<=z_berm_o
            if isempty(BekledingsLocatie)
               BekledingsLocatie = 'Ondertalud';
               VorigProfiel=Profiel(p).ident;
            elseif ~strcmp(BekledingsLocatie,'Ondertalud')
```

```
               Foutmelding = sprintf('Segment %i op ondertalud voor profiel %s en op
%s voor profiel %s!',i,Profiel(p).ident,lower(BekledingsLocatie),VorigProfiel);
               Fout=1;
            end
         else
            Foutmelding = sprintf('Segment %i verspreid over verschillende delen van
profiel %s!',i,Profiel(p).ident);
            Fout=1;
         end
         if Fout, break; end
      end
      if Fout, break; end
   end
end
%
if Fout
   if nargout==0
      error(Foutmelding);
   elseif nargout==1
      Uitvoer('p','FOUT: %s\n',Foutmelding);
   end
end
```

General/LeesBekleding

```
function Dijkopbouw = LeesBekleding(Bestandsnaam)
%
% Lees bekleding van file. Bij een aanroep vanuit BatchRekenen wordt de
% bestandsnaam gespecificeerd. Vanuit de user interface vindt de aanroep
% plaats zonder specificatie van de bestandsnaam en wordt de gebruiker
% eerst om een bestandsnaam gevraagd.
%
if nargin < 1
  [bestand,pad] = uigetfile('*.bekleding');
  if ~ischar(bestand)
     Dijkopbouw = [];
     return
  end
  Bestandsnaam = fullfile(pad,bestand);
end
%
% Lees bestand ...
%
fid = fopen(Bestandsnaam,'r');
fgetl(fid); %HYDRA-K DIJKOPBOUWBESTAND
AantalLagen = sscanf(fgetl(fid),'%i',1); %i LAGEN
for i = 1:AantalLagen
   Dijkopbouw(i).bekleding = deblank(fgetl(fid));
   Dijkopbouw(i).ondergrens = sscanf(fgetl(fid),'%f',1); %f Ondergrens in m+NAP
   Dijkopbouw(i).bovengrens = sscanf(fgetl(fid),'%f',1); %f Bovengrens in m+NAP
   switch Dijkopbouw(i).bekleding
   case 'asfalt'
      gegevens.laagdikte = sscanf(fgetl(fid),'%f',1); %f Laagdikte in m
      gegevens.ondergrondtype = lower(strtok(fgetl(fid))); %s Ondergrondtype
      gegevens.asfalttype = lower(strtok(fgetl(fid))); %s Asfalttype
      gegevens.filtertype = lower(strtok(fgetl(fid))); %s Filtertype
      gegevens.rho_water = sscanf(fgetl(fid),'%f',1); %s Dichtheid water in kg/m^3
      gegevens.rho_asfalt = sscanf(fgetl(fid),'%f',1); %s Dichtheid asfalt in kg/m^3
      gegevens.z_ogb = sscanf(fgetl(fid),'%f',1); %s Ondergrens gesloten bekleding in
m+NAP
   case 'blokken'
      gegevens.dichtheid_hoog = sscanf(fgetl(fid),'%f',1); %f Relatieve dichtheid
      gegevens.blokdikte_hoog = sscanf(fgetl(fid),'%f',1); %f Blokdikte in m
      gegevens.dichtheid_laag = gegevens.dichtheid_hoog;
      gegevens.blokdikte_laag = gegevens.blokdikte_hoog;
   case 'gras'
      gegevens.kwaliteit = lower(strtok(fgetl(fid))); %s Grasmat kwaliteit
   case 'steenzetting'
      gegevens.toplaagdikte = sscanf(fgetl(fid),'%f',1); %f Toplaagdikte in m
      gegevens.filterkleidikte = sscanf(fgetl(fid),'%f',1); %f Totale dikte van
filter- en kleilaag in m
      gegevens.D15 = sscanf(fgetl(fid),'%f',1); %f D15 van het zand in m
      gegevens.relatievedichtheid = sscanf(fgetl(fid),'%f',1); %f Relatieve dichtheid
```

```
      gegevens.bekledingstype = lower(fgetl(fid)); %s[met spaties!] Type bekleding
      label = findstr(gegevens.bekledingstype,'type');
      if ~isempty(label)
         gegevens.bekledingstype=deblank(gegevens.bekledingstype(1:label(1)-1));
      end
      gegevens.toetsgrens = lower(strtok(fgetl(fid))); %s Toetsgrens
      gegevens.constructie = lower(strtok(fgetl(fid))); %s Type constructie
   otherwise
      fclose(fid);
      uiwait(msgbox(sprintf('Onbekend bekledingstype:
''%s''.',Dijkopbouw(i).bekleding), ...
         'Foutmelding','modal'));
      Dijkopbouw = [];
      return;
   end
   Dijkopbouw(i).bekledingsgegevens = gegevens;
end
fclose(fid);
```

General/MeerdereBekledingen

```
function Ja = MeerdereBekledingen(UiWaarden)
%MEERDEREBEKLEDINGEN

Ja = 0;
if UiWaarden.faalmechanisme==4
   Ja = length(UiWaarden.dijkopbouw) > 1;
end
```

General/PiekPeriode

```
function Y=PiekPeriode

%
% Tijdelijke oplossing: piekperiode meenemen als extra stochast
%

persistent meenemen

if isempty( meenemen )
   meenemen = 0 ;
   if strcmp( '1', IniFile( 'LEES', 'general', 'piekperiode_stochast', '0' ) ),
      meenemen = 1 ;
   end
end

Y=meenemen ;
```

General/printBekleding

```
function Fout = printBekleding (Waarden)

%PRINTBEKLEDING   afdrukken van de bekleding
%  AANROEP
%     printBekleding (Waarden)
%  INVOER
%     Waarden:    structure-array met waarden van faalmechanisme,
%                 profielen en bekleding

%  EXTERNE FUNCTIES
%     Uitvoer

% Copyright © 2000 Rijkswaterstaat/RIKZ All Rights Reserved.

GlobalConstants

Bekleding = [];
if isfield(Waarden,'faalmechanisme') & Waarden.faalmechanisme == FMINSTABBEKLED
   Bekleding = Waarden.dijkopbouw;
```

```
   Profiel = Waarden.profiel;
elseif isfield(Waarden,'naam') & strcmp(Waarden.naam,'instabiliteit van de
bekleding')
   Bekleding = Waarden.dijkopbouw;
   Profiel = Waarden.profiel;
end
Fout=0;
if ~isempty(Bekleding)
   Uitvoer('p','\n%18s\n','BEKLEDING');
   Bekleding = Waarden.dijkopbouw;
   Uitvoer('p','%25s%i\n','Aantal segmenten: ',length(Bekleding));
   for i=1:length(Bekleding)
      Uitvoer('p','%28s%i\n','SEGMENT: ',i);
      Uitvoer('p','%28s%s\n','Bekleding: ',upper(Bekleding(i).bekleding));
      Uitvoer('p','%28s%5.2f%s\n','Ondergrens: ',Bekleding(i).ondergrens,' m+NAP');
      Uitvoer('p','%28s%5.2f%s\n','Bovengrens: ',Bekleding(i).bovengrens,' m+NAP');
      gegevens = Bekleding(i).bekledingsgegevens;
      switch Bekleding(i).bekleding
      case 'asfalt'
         Uitvoer('p','%28s%5.2f%s\n','Laagdikte: ',gegevens.laagdikte,' m');
         Uitvoer('p','%28s%s\n','Ondergrondtype: ',gegevens.ondergrondtype);
         Uitvoer('p','%28s%s\n','Asfalttype: ',gegevens.asfalttype);
         Uitvoer('p','%28s%s\n','Filtertype: ',gegevens.filtertype);
         Uitvoer('p','%28s%5.0f%s\n','Dichtheid water: ',gegevens.rho_water,'
kg/m^3');
         Uitvoer('p','%28s%5.0f%s\n','Dichtheid asfalt: ',gegevens.rho_asfalt,'
kg/m^3');
         Uitvoer('p','%28s%5.2f%s\n','Ondergrens gesl. bekleding: ',gegevens.z_ogb,'
m+NAP');
      case 'blokken'
         Uitvoer('p','%28s%5.2f\n','Relatieve dichtheid: ',gegevens.dichtheid_hoog);
         Uitvoer('p','%28s%5.2f%s\n','Blokdikte: ',gegevens.blokdikte_hoog,' m');
      case 'gras'
         Uitvoer('p','%28s%s\n','Grasmat kwaliteit: ',gegevens.kwaliteit);
      case 'steenzetting'
         Uitvoer('p','%28s%5.2f%s\n','Toplaagdikte: ',gegevens.toplaagdikte,' m');
         Uitvoer('p','%28s%5.2f%s\n','Dikte filter- en kleilaag:
',gegevens.filterkleidikte,' m');
         Uitvoer('p','%28s%7.4f%s\n','D15 van zand: ',gegevens.D15,' m');
         Uitvoer('p','%28s%5.2f\n','Relatieve dichtheid:
',gegevens.relatievedichtheid);
         Uitvoer('p','%28s%s\n','Type bekleding: ',gegevens.bekledingstype);
         Uitvoer('p','%28s%s\n','Toetsgrens: ',gegevens.toetsgrens);
         Uitvoer('p','%28s%s\n','Type constructie: ',gegevens.constructie);
      end
   end
end
F = controleerBekleding(Waarden);
if nargout>0
   Fout = F;
end
```

General/StormDuurAfhankelijk

```
function Ja = StormDuurAfhankelijk(UiWaarden)
%STORMDUURAFHANKELIJK

Ja = 0;
if isfield(UiWaarden,'faalm') & UiWaarden.faalm == 4
   %
   % Batch.invoer
   %
   Ja = 0.5;
elseif isfield(UiWaarden,'faalmechanisme') & UiWaarden.faalmechanisme == 4
   %
   % UiWaarden
   %
   Ja = 0.5;
elseif isfield(UiWaarden,'naam') & strcmp(UiWaarden.naam,'instabiliteit van de
bekleding')
   %
   % FaalMechanisme
   %
```

```
   Ja = 0.5;
end
if Ja == 0.5
   Ja = 0;
   AantalLagen = length(UiWaarden.dijkopbouw);
   for i = 1:AantalLagen
      if strcmp(UiWaarden.dijkopbouw(i).bekleding,'gras')
         Ja = 1;
      end
   end
end
```

Stat/StatBerekenBekleding

```
function [paramWaarde,IndexFalen] = ...
   StatBerekenBekleding (Regio, Locatie, correctieRVB, WaarnOverschrFactor,
Waarneming, WaarnStormSectorIndex, ...
   WaarnStormIndex, Stat, Profiel, HerhalingsTijd , Kappa , SelectieWaarnemingen,
StroomCor, faalMechID, dijkopbouw)

%STATBEREKENBEKLEDING         In StatBerekenBekleding wordt een eigenschap van de
bekleding
%                             bepaald gegeven de herhalingstijd van falen en het
dijkprofiel.
%                             Deze functie is conceptueel gelijk aan
StatBerekenKruinhoogte.
%
%                             Voor continue parameters worden eerst een ondergrens en
een
%                             bovengrens voor de te varieren parameter bepaald.
Daarna wordt de
%                             benodigde waarde van de parameter bepaald met een
minimalisatie-
%                             routine.

%                             Voor discrete parameters worden alle waarden doorlopen
beginnend
%                             in het minimum.
%
%                             Bij elke waarde komt een aantal waarnemingen in het
faalgebied.
%                             Het stopcriterium is als het aantal waarnemingen in het
faal-
%                             gebied gelijk is aan kappa. (Kappa is eenduidig bepaald
door de
%                             herhalingstijd.)
%
%  AANROEP
%     [paramWaarde,IndexFalen] = ...
%        StatBerekenBekleding (Regio, Locatie, correctieRVB, WaarnOverschrFactor, ...
%           Waarneming, WaarnStormSectorIndex, WaarnStormIndex, Stat, Profiel, ...
%           HerhalingsTijd , Kappa , SelectieWaarnemingen)
%  INVOER
%     Regio:                  string met naam van de regio
%     Locatie:                de x,y coordinaten van de locatie
%     correctieRVB:           de correctie op de waterstand voor verschillen met het
Randvoorwaardenboek
%     WaarnOverschrFactor:    formule: -
log(waarnemingenoverschrijdingsaantallen/kappa)
%                             (waterstand, windsnelheid, piekperiode, sign.
golfhoogte)
%     Waarneming:             de golfrichting en de windrichting van de waarnemingen
%     WaarnStormSectorIndex:  de sectorindex van de storm (windrichting van de storm)
%     WaarnStormIndex:        de index van de storm behorend bij de waarneming
%     Freqbound:              bovengrens voor de frequentie
%     Stat:                   de Weibull parameters voor de waterstand, windsnelheid,
%                             piekperiode en sign. golfhoogte
%     Profiel:                het dijkprofiel
%     Herhalingstijd:         de ingevoerde herhalingstijd
%     Kappa:                  het aantal verschoven waarnemingen in het faalgebied
%     SelectieWaarnemingen:   criterium voor de selectie van de waarnemingen: zie
stap 2
```

```
%                               Methode de Haan op basis van de hoogste waterstand of
op basis
%                               van de hoogste belasting
%  UITVOER
%     paramWaarde:              de benodigde waarde van de parameter bij de
herhalingstijd;
%                               de parameter is afhankelijk van de bekleding
%     IndexFalen:               een vector met de indices van de waarnemingen die in
het
%                               faalgebied liggen

%  EXTERNE FUNCTIES
%     StatWeibullInv
%     Offshore2Nearshore
%     StatBerekenZuitNs
%     SelecteerMaxZ
%     StatKruinhBloksMinimizeFunc

% Copyright © 2000 Rijkswaterstaat/RIKZ All Rights Reserved.

% Auteur: Bert Jagers, WL | Delft Hydraulics

IndexFalen= [];

% bepalen offshorewaarden (waterstand met wind- en golfrichting)
offshoreWaarden= [StatWeibullInv(WaarnOverschrFactor+log(HerhalingsTijd),...
      WaarnStormSectorIndex,Stat,Regio) Waarneming];

% transformatie van offshore naar nearshore
[nearshoreWaarden,WaarnStormIndex] = Offshore2Nearshore ( ...
   Regio, Locatie.x,Locatie.y,
correctieRVB,offshoreWaarden,StroomCor,faalMechID,WaarnStormIndex);

faalMechanisme.naam = 'instabiliteit van de bekleding';
faalMechanisme.dijkopbouw = dijkopbouw;

%
% Onderstaande gaat alleen goed als er maar 1 bekledingssegment is ....
%
discreteparameter = 0;
switch dijkopbouw.bekleding
case 'asfalt'
   ondergrens = 0.01;
   bovengrens = 50;
   parameter = 'laagdikte';
case 'blokken'
   ondergrens = 0.01;
   bovengrens = 50;
   parameter = 'dichtheid_hoog';
   faalMechanisme.dijkopbouw.bekledingsgegevens.blokdikte_laag = 1;
   faalMechanisme.dijkopbouw.bekledingsgegevens.blokdikte_hoog = 1;
case 'gras'
   parameter = 'kwaliteit';
   discreteparameter = 1;
   parameterwaarden = {'slecht','matig','goed'};
case 'steenzetting'
   ondergrens = 0.01;
   bovengrens = 50;
   parameter = 'relatievedichtheid';
   faalMechanisme.dijkopbouw.bekledingsgegevens.toplaagdikte = 1;
end


if discreteparameter
   for i = 1:length(parameterwaarden)
      faalMechanisme.dijkopbouw.bekledingsgegevens = ...

setfield(faalMechanisme.dijkopbouw.bekledingsgegevens,parameter,parameterwaarden{i});
      %
      % Hier worden altijd alle waarnemingen (dus hele stormen voor zover relevant)
      % doorgegeven.
      %
      z = StatBerekenZuitNs (faalMechanisme,
Profiel,nearshoreWaarden,WaarnStormIndex);
      [aantalZkleinerNul, Jm, zOorsprIndex] = ...
```

```
            SelecteerMaxZ ( z, Kappa, WaarnStormIndex, SelectieWaarnemingen);

         if aantalZkleinerNul <= Kappa,
            IndexFalen= zOorsprIndex(1:aantalZkleinerNul);
            paramWaarde = parameterwaarden{i};
            return
         end
      end
      %
      % zelf bovengrens voldoet niet...
      %
      IndexFalen = [];
      paramWaarde = Inf;

   else % continue parameter
      faalMechanisme.ontwerpparameter = parameter;

      % kijk eerst of de ondergrens wel voldoende is
      faalMechanisme.dijkopbouw.bekledingsgegevens = ...
         setfield(faalMechanisme.dijkopbouw.bekledingsgegevens,parameter,ondergrens);
      if strcmp(parameter,'dichtheid_hoog')
         faalMechanisme.dijkopbouw.bekledingsgegevens.dichtheid_laag = ondergrens;
      end

      z = StatBerekenZuitNs (faalMechanisme, Profiel,nearshoreWaarden,WaarnStormIndex);
      [aantalZkleinerNul, Jm] = ...
         SelecteerMaxZ ( z, Kappa, WaarnStormIndex, SelectieWaarnemingen);

      if aantalZkleinerNul < Kappa,
         % IndexFalen= zOorsprIndex(1:aantalZkleinerNul);
         IndexFalen= [];
         paramWaarde = -Inf;

         return
      end

      % kijk eerst of de bovengrens wel voldoende is
      faalMechanisme.dijkopbouw.bekledingsgegevens = ...
         setfield(faalMechanisme.dijkopbouw.bekledingsgegevens,parameter,bovengrens);
      if strcmp(parameter,'dichtheid_hoog')
         faalMechanisme.dijkopbouw.bekledingsgegevens.dichtheid_laag = bovengrens;
      end

      z = StatBerekenZuitNs (faalMechanisme, Profiel,nearshoreWaarden,WaarnStormIndex);
      [aantalZkleinerNul, Jm] = ...
         SelecteerMaxZ ( z, Kappa, WaarnStormIndex, SelectieWaarnemingen);

      if aantalZkleinerNul > Kappa,
         % IndexFalen= zOorsprIndex(1:aantalZkleinerNul);
         IndexFalen= [];
         paramWaarde = Inf;

         return
      end

      % de oplossing ligt ergens tussen ondergrens en bovengrens
      % Nu de bovengrens  zodanig maken dat aantalZkleinerNul < Kappa
      %
      % stapgrootte tussen minimum en maximum zodanig dat deze voor blokken
      % ongeveer op 4 uitkomt net zoals in StatBerekenBloksterkte
      %
      stap = (bovengrens-ondergrens)/12.5;

      while 1

         faalMechanisme.dijkopbouw.bekledingsgegevens = ...

setfield(faalMechanisme.dijkopbouw.bekledingsgegevens,parameter,ondergrens+stap);
         if strcmp(parameter,'dichtheid_hoog')
            faalMechanisme.dijkopbouw.bekledingsgegevens.dichtheid_laag =
ondergrens+stap;
         end

         z = StatBerekenZuitNs (faalMechanisme,
Profiel,nearshoreWaarden,WaarnStormIndex);
```

```
      [aantalZkleinerNul, Jm] = ...
         SelecteerMaxZ ( z, Kappa, WaarnStormIndex, SelectieWaarnemingen);
      if aantalZkleinerNul > Kappa,
         ondergrens = ondergrens+stap;
      elseif aantalZkleinerNul == 0,
         stap = stap/4;
      else,
         break;
      end

   end
   bovengrens = ondergrens+stap;

   count = 0;
   stormduurafh = StormDuurAfhankelijk(faalMechanisme);
   while 1

      if stormduurafh
         %
         % In het geval van stormduurafhankelijke bekleding moet de hele storm worden
         % doorgegeven ...
         %
         heleStorm=WaarnStormIndex==WaarnStormIndex(Jm);
         paramWaarde = fminbnd('StatKruinhBloksMinimizeFunc', ondergrens, bovengrens,
...
            optimset('TolX',1e-4,'Display','off'),...
            faalMechanisme, Profiel, nearshoreWaarden(heleStorm,:),WaarnStormIndex);
      else
         paramWaarde = fminbnd('StatKruinhBloksMinimizeFunc', ondergrens, bovengrens,
...
            optimset('TolX',1e-4,'Display','off'),...
            faalMechanisme, Profiel, nearshoreWaarden(Jm,:),WaarnStormIndex);
      end

      faalMechanisme.dijkopbouw.bekledingsgegevens = ...

setfield(faalMechanisme.dijkopbouw.bekledingsgegevens,parameter,paramWaarde);
      if strcmp(parameter,'dichtheid_hoog')
         faalMechanisme.dijkopbouw.bekledingsgegevens.dichtheid_laag = paramWaarde;
      end

      count = count + 1;
      if count > 100,
         Uitvoer('p', 'lus afgebroken');
         break;
      end

      z = StatBerekenZuitNs (faalMechanisme,
Profiel,nearshoreWaarden,WaarnStormIndex);
      [aantalZkleinerNul, Jm, zOorsprIndex] = ...
         SelecteerMaxZ ( z, Kappa, WaarnStormIndex, SelectieWaarnemingen);

      if aantalZkleinerNul> Kappa,
         if abs(bovengrens-ondergrens) < 0.0001,
            % staat een beetje om Kappa heen te springen
            break;
         elseif abs(ondergrens-paramWaarde) < 0.0001,
            ondergrens = ondergrens+min(0.01*abs(ondergrens-bovengrens),0.0005);
         else
            ondergrens= paramWaarde ;
         end

      elseif aantalZkleinerNul< Kappa,
         if abs(bovengrens-paramWaarde) < 0.0001,
            bovengrens = bovengrens-min(0.01*abs(ondergrens-bovengrens),0.0005);
         else
            bovengrens= paramWaarde ;
         end

      else
         % hier is de oplossing gevonden
         break;
      end

   end
```

```
   IndexFalen= zOorsprIndex(1:Kappa);
end
```

Stat/StatBerekenOploopniveau

```
function z_q  = StatBerekenOploopniveau(Profiel, Qc, H_s, T_p, sl, theta_0)

%STATBEREKENOPLOOPNIVEAU      In StatBerekenOploopniveau wordt het oploopniveau
%        behorende bij het opgegeven debiet Qc uitgerekend gebruikmakend van de
%        formules voor het faalmechanisme golfoverslag. Het eerste deel van de
%        code komt overeen met de routine StatBerekenZGolfoverslag. Het tweede
%        deel berekent vanuit het (overslag)debiet het bijbehorende oploopniveau
%        of de kritische kruinhoogte in plaats van uit de kruinhoogte het over-
%        slaand debiet zoals StatBerekenZGolfoverslag dat berekent.
%
%  AANROEP
%     z_q = StatBerekenOploopniveau (Profiel, Qc, H_s, T_p, sl, theta_0)
%  INVOER
%     Profiel:    het dijkprofiel
%     Qc:         het kritieke overslagdebiet (dat overslagdebiet dat nog
%                 toelaatbaar is)
%     H_s:        significante golfhoogte (nearshore)
%     T_p:        de piekperiode of de spectrale periodemaat (nearshore)
%     sl:         waterstand (nearshore)
%     theta_0:    golfrichting (nearshore) (t.o.v. het noorden, in radialen)
%  UITVOER
%     z_q:        het oploopniveau (ook wel de kritieke overslag kruinhoogte h_k
%                 bij het gespecificeerde kritieke overslagdebiet).

%  EXTERNE FUNCTIES
%     StatBerekenZGolfoploop

% Auteur: Bert Jagers, WL | Delft Hydraulics

% Gebaseerd op StatBerekenZGolfoverslag (ref. versie Hydra-K 2.05)
%    Copyright © 2000 Rijkswaterstaat/RIKZ All Rights Reserved.

%
%--- Vanaf hier tot aan soortgelijke eindmarker is de code identiek aan
%    StatBerekenZGolfoverslag.
%


%
% Originele en nieuwe formulering hebben dezelfde functionele vorm,
% alleen de coefficienten verschillen enigzins. Vang ze daarom hier op.
%
if ~NieuweGolfFormules,
   Q_b_coeff = 0.06  ;
   R_b_coeff = 4.7   ;
else
   Q_b_coeff = 0.067 ;
   R_b_coeff = 4.3   ;
end

% Voor brekende golven geldt:
% q = Q_b * gamma_b * sqrt( g*H_s^3*tan(alfa) / s_op )
% waarbij:
% q = overslaand debiet [l/s/m]
% Q_b = dimensieloos overslagdebiet voor brekende golven
% gamma_b = reductiefactor voor invloed berm
% g = zwaartekrachtversnelling [m/s2]
% H_s = significante golfhoogte [m]
% alfa = helling van het talud [- is al een tangens]
% s_op = golfsteilheid

% Definitie golfsteilheid:
% s_op = 2 * pi * H_s / ( g * T_p^2)
% waarbij:
% T_p = piekperiode [s]

% Definitie dimensieloos overslagdebiet bij brekende golven:
% Q_b = 0,06 * exp (-4,7 * R_b)
```

```
% waarbij:
% R_b = dimensieloze kruinhoogte bij brekende golven

% Definitie dimensieloze kruinhoogte bij brekende golven:
% R_b = h_k * sqrt(s_op) / (H_s * tan(alfa) * gamma_b * gamma_f * ...
%     gamma_beta * gamma_s * gamma_v)
% waarbij:
% h_k = vrije kruinhoogte boven stilwaterlijn
% gamma_f = reductiefactor voor invloed van ruwheid
% gamma_beta = reductiefactor voor invloed van hoek van golfaanval
% gamma_s = reductiefactor voor invloed van hoek van golfaanval
% gamma_v = reductiefactor voor invloed van verticale wand


% Voor niet-brekende golven geldt:
% q = Q_n * sqrt(g * H_s^3)
% waarbij Q_n = maximum dimensieloos overslagdebiet bij niet-brekende golven

% Definitie dimensieloos overslagdebiet bij niet-brekende golven:
% Q_n = 0,2 * exp(-2,3*R_n)
% waarbij:
% R_n = dimensieloze kruinhoogte bij niet-brekende golven

% Definitie dimensieloze kruinhoogte bij niet-brekende golven:
% R_n = h_k / (H_s * gamma_f * gamma_beta * gamma_s)

hoogte_kruin = Profiel.hoogte_kruin;
theta_dijk   = Profiel.theta_dijk * pi / 180;

g = 9.81;

h_k = hoogte_kruin - sl;
verschilhoek = abs(normaliseerHoek2(theta_0-theta_dijk)) *180/pi;
% normaliseerHoek0 normaliseert hoeken, zodat ze liggen tussen -pi en pi

[alfa, gamma_b, gamma_f, hoogte_2pr, s_op, zeta, z_2pr] ...
   = StatBerekenZGolfoploop (Profiel, H_s, T_p, sl, theta_0);

% alfa = berekend door Golfoploop
% gamma_b = berekend door Golfoploop
% gamma_f = berekend door Golfoploop

max_gamma_beta = 0.736;
min_gamma_s = 0;
max_gamma_s = 1;
gamma_beta = max ( max_gamma_beta, 1-verschilhoek .* 0.0033);
gamma_s = max ( min_gamma_s, min (max_gamma_s, 1-(verschilhoek-80)./30));

% Dus
% gamma_beta = 0.736                   als verschilhoek < 80
% gamma_beta = 1-verschilhoek * o,0022 als verschilhoek > 80

% Wat is de juiste waarde van gamma_v?
gamma_v = 1;

gamma_t = max(0.4, gamma_beta .* gamma_f .* gamma_b .* gamma_v);
gamma_totaal = gamma_t.* gamma_s;

mini = 1.e-10;
%
%--- Tot hier is de code identiek aan StatBerekenZGolfoverslag
%

%
% In plaats van het uitrekenen van het overslag debiet bij de
% kruinhoogte van het dijkprofiel reken terug vanuit het overslag
% debiet naar de minimale benodigde kruinhoogte.
%

%
% Corrigeer voor negatieve golfhoogtes (komen een enkele keer
% voor indien de stormflanken worden toegevoegd.
%
H_s=max(H_s,1e-10);
T_p=max(T_p,1e-10);
```

```
%
% Bepaal de coefficienten voor de eerste relatie tussen z_q en q.
%
alfa_b1 = 1000 .* sqrt(g .* H_s.^3) .* Q_b_coeff .*zeta .* gamma_b ./sqrt(alfa);
alfa_b2 = -R_b_coeff ./ max(mini, zeta .* H_s .* max(gamma_totaal, mini));

%
% Bepaal de coefficienten voor de tweede relatie tussen z_q en q.
%
alfa_n1 = 0.2 * 1000 .* sqrt(g .* H_s.^3);
alfa_n2 = -2.3./ max(mini, H_s .* gamma_f .* gamma_beta .* gamma_s);

%
% Bepaal voor beide vergelijkingen het niveau en neem het minimum.
%
z_qb = log(Qc./alfa_b1)./alfa_b2;
z_qn = log(Qc./alfa_n1)./alfa_n2;
z_q  = min(z_qb,z_qn);

%
% Controle code: bereken overslaand debiet ...
%
%h_k = z_q;
%q_b = alfa_b1 .* exp(alfa_b2 .* h_k);
%q_n = alfa_n1 .* exp(alfa_n2 .* h_k);
%q = min(q_b,q_n);

%
% Begrens oploophoogte tot waarden groter of gelijk 0. Let op: na deze
% correctie is het resterend overslagdebiet kleiner dan of gelijk aan
% de gewenste waarde.
%
z_q=max(z_q,0);
```

Stat/StatBerekenZAsfalt

```
function [Z,q] = StatBerekenZAsfalt (Profiel, ZSegment, H_s, T_p, sl, theta_0)

%STATBEREKENZASFALT   In StatBerekenZAsfalt wordt de Z-functie uitgerekend voor
%                     het faalmechanisme instabiliteit voor asfalt.
%  AANROEP
%     [Z,q] = BerekenZAsfalt (Profiel, ZSegment, H_s, T_p, sl, theta_0)
%  INVOER
%     Profiel:    het dijkprofiel
%     ZSegment:   karakteristiek van het dijksegment (ondergrens & bovengrens,
%                 en bekledingsgegevens.
%     H_s:        significante golfhoogte (nearshore)
%     T_p:        de piekperiode (nearshore)
%     sl:         waterstand (nearshore)
%     theta_0:    golfrichting (nearshore) (t.o.v. het noorden, in radialen)
%  UITVOER
%     Z:          Z is gelijk aan de asfaltsterkte minus de belasting
%     q:          de belasting q

% Auteur: Bert Jagers, WL | Delft Hydraulics

% 17-10-2003
%
% Corrigeer voor negatieve golfhoogtes (komen een enkele keer
% voor indien de stormflanken worden toegevoegd.
%
H_s=max(H_s,1e-10);
T_p=max(T_p,1e-10);

%
% Bekledingsgegevens ...
%
laagdikte      = ZSegment.bekledingsgegevens.laagdikte;
ondergrondtype = ZSegment.bekledingsgegevens.ondergrondtype; %'zand','klei'
asfalttype     = ZSegment.bekledingsgegevens.asfalttype;
%'waterbouwasfaltbeton','opensteenasfalt'
filtertype     = ZSegment.bekledingsgegevens.filtertype; %'geotextiel','zandasfalt'
```

```
rho_water      = ZSegment.bekledingsgegevens.rho_water;
rho_asfalt     = ZSegment.bekledingsgegevens.rho_asfalt;
z_ogb          = ZSegment.bekledingsgegevens.z_ogb; %ondergrens gesloten bekleding
(z_ogb< z_ondergrens)
%
% Verticale locatie van het segment in het dijkprofiel ...
%
z_ondergrens   = ZSegment.ondergrens;
z_bovengrens   = ZSegment.bovengrens;
%
% Bepaal helling ...
%
%helling        = BepaalHelling(1.5,H_s,sl,Profiel);
helling              = repmat(Profiel.helling_laag,size(sl));
z_berm_o             = Profiel.hoogte_berm;
z_berm_b             = Profiel.hoogte_berm +
Profiel.breedte_berm*Profiel.helling_berm;
bekleding_op_berm    = z_ondergrens>=z_berm_o & z_bovengrens<=z_berm_b;
bekleding_boven_berm = z_ondergrens>=z_berm_b & z_bovengrens>z_berm_b;
helling(bekleding_boven_berm) = Profiel.helling_hoog;

%
% De van de asfaltlaag is de maatgevende grootheid voor de
% sterkte van de asfaltbekleding voor zowel golfklappen als
% het opdrukken van de asfaltlaag.
%

sterkte          = repmat(laagdikte,size(sl));
belasting        = repmat(0,size(sl));

%
% Hoogtecriterium voor de toetsing op golfklappen. Let op:
% opdrukken asfalt gebruikt ander hoogtecriterium (zie lokaal
% bij de code voor de bereking van dit faalmechanisme).
%

if bekleding_op_berm
   sltelaag         = sl < z_ondergrens;
   sltehoog         = sl > (z_bovengrens + 1.5 *H_s);
   resterend        = ~sltehoog & ~sltelaag;
else
   sltelaag         = sl < (z_ondergrens - 0.25*H_s);
   sltehoog         = sl > (z_bovengrens + 2.0 *H_s);
   resterend        = ~sltehoog & ~sltelaag;
end

switch asfalttype
   case 'waterbouwasfaltbeton'
      %----------------------
      %   Golfklappen
      %----------------------

      switch ondergrondtype
         case 'zand'
            minimum_laagdikte1    = (-0.18401+0.125056*H_s+0.0014*H_s.^2).*(-
1.69947+5.174361*helling+0.098738*H_s);
         case 'klei'
            minimum_laagdikte1    = (-0.06927+0.084858*H_s+0.013069*H_s.^2).*(-
0.93087+5.108631*helling+0.060252*H_s);
      end
      minimum_laagdikte2    = 0.1 + 0.0167*H_s;
      minimum_laagdikte     = max(minimum_laagdikte1,minimum_laagdikte2);
      belasting(resterend)  = minimum_laagdikte(resterend);

      tehogegolven          = resterend & (H_s>6.0);
      sterkte(tehogegolven) = -inf;

      %----------------------
      %   Opdrukken Asfalt
      %----------------------
      %
      % Bepaal maatgevende grondwaterstand op basis van de waterstand
      % (maximale waterstand tijdens storm) en de gemiddelde waterstand.
      %
      z_GWS=0; % gemiddelde waterstand
```

```
      %
      z_MGW = (sl + z_GWS)/2;
      %
      % Bepaal v: het verschil tussen de maatgevende grondwaterstand
      % en de maatgevende buitenwaterstand welke begrensd is door de
      % gemiddelde waterstand.
      %
      v = min(0.53*(z_MGW-z_ogb),z_MGW-z_GWS);
      %
      % Reductiefactor i.v.m. buitenwatersand
      %
      R_w = max(0,1-3.64*(0.53-(v./(z_MGW-z_ogb))).^2);
      %
      % Taludfactor
      %
      alpha=atan(helling);
      Qn = 0.96*(cos(alpha)).^(-1.4);
      %
      % Bepaal minimaal benodigde dikte van het asfalt tegen opdrukken.
      %
      dichtheid=rho_water/(rho_asfalt-rho_water);
      minimum_laagdikte = 0.21*Qn.*(z_MGW-z_ogb)*dichtheid.*R_w;
      zMGW_voldoende_hoog         = z_MGW > z_ondergrens;
      %
      % Belasting = maximum van de belasting t.g.v. golfklappen
      %
      belasting(zMGW_voldoende_hoog) = max(belasting(zMGW_voldoende_hoog), ...
                                    minimum_laagdikte(zMGW_voldoende_hoog));

      %----------------------
   case 'opensteenasfalt'
      %----------------------
      %   Golfklappen
      %----------------------
      switch ondergrondtype
         case 'zand'
            minimum_laagdikte1    = (-0.05785+0.025545*H_s+0.026866*H_s.^2).*(-
0.93087+5.108631*helling+0.060252*H_s);
         case 'klei'
            minimum_laagdikte1    = (-0.1229+0.17821*H_s+0.03541978*H_s.^2).*(-
1.36632+5.138124*helling+0.128026*H_s);
      end
      switch filtertype
         case 'geotextiel'
            c3     = 0.2;
            c4     = 0.025;
         case 'zandasfalt'
            c3     = 0.1;
            c4     = 0.025;
      end
      minimum_laagdikte2    = c3 + c4*H_s;
      minimum_laagdikte     = max(minimum_laagdikte1,minimum_laagdikte2);
      belasting(resterend)  = minimum_laagdikte(resterend);

      tehogegolven          = resterend & (H_s>4.0);
      sterkte(tehogegolven) = -inf;
end

q = belasting;
Z = sterkte - belasting;
```

Stat/StatBerekenZGras

```
function [Z,q] = StatBerekenZGras (Profiel, ZSegment, WaarnStormIndex, ...
   H_s, T_p, sl, theta_0)

%STATBEREKENZGRAS     In StatBerekenZGras wordt de Z-functie uitgerekend voor
%                     het faalmechanisme instabiliteit voor grasbekleding.
%  AANROEP
%     [Z,q] = BerekenZGras (Profiel, ZSegment, WaarnStormIndex, H_s, T_p, sl,
theta_0)
%  INVOER
%     Profiel:    het dijkprofiel
```

```
%     ZSegment:   karakteristiek van het dijksegment (ondergrens & bovengrens,
%                 en bekledingsgegevens.
%     WaarnStormIndex:        de index van de storm behorend bij de waarneming
%     H_s:        significante golfhoogte (nearshore)
%     T_p:        de piekperiode (nearshore)
%     sl:         waterstand (nearshore)
%     theta_0:    golfrichting (nearshore) (t.o.v. het noorden, in radialen)
%  UITVOER
%     Z:          Z is gelijk aan de asfaltsterkte minus de belasting
%     q:          de belasting q

% Auteur: Bert Jagers, WL | Delft Hydraulics

% 3-11-2003
%
% Corrigeer voor negatieve golfhoogtes (komen een enkele keer
% voor indien de stormflanken worden toegevoegd.
%
H_s=max(H_s,1e-10);
T_p=max(T_p,1e-10);

%
% Bekledingsgegevens ...
%
grasmat         = ZSegment.bekledingsgegevens.kwaliteit; % 'goed','matig','slecht'
z_q             = StatBerekenOploopniveau(Profiel, 0.1, H_s, T_p, sl, theta_0);
%
% Verticale locatie van het segment in het dijkprofiel ...
%
z_ondergrens    = ZSegment.ondergrens;
z_bovengrens    = ZSegment.bovengrens;
%
% Bepaal helling ...
%
helling             = repmat(Profiel.helling_laag,size(sl));
z_berm_o            = Profiel.hoogte_berm;
z_berm_b            = Profiel.hoogte_berm +
Profiel.breedte_berm*Profiel.helling_berm;
bekleding_op_berm   = z_ondergrens>=z_berm_o & z_bovengrens<=z_berm_b;
bekleding_boven_berm = z_ondergrens>=z_berm_b & z_bovengrens>z_berm_b;
helling(bekleding_boven_berm) = Profiel.helling_hoog;


sterkte         = repmat(1,size(sl));
belasting       = repmat(0,size(sl));

%
% Bereken ts/tsmax voor alle waarnemingen.
%
tijdstap         = 1; % tijdstap van 1 uur

grasbovengolven  = z_ondergrens > sl + z_q;
resterend        = ~grasbovengolven;

ToetsGolfklappen = resterend & (z_ondergrens < sl);
ToetsStroming    = resterend & ~ToetsGolfklappen;

%
% Toets op golfklappen ...
%
karakParam = 2*(H_s.^0.75).*(T_p.^0.5).*helling;
switch grasmat
case 'goed'
   C1=1.0;
case 'matig'
   C1=0.7;
case 'slecht'
   C1=0.2;
end
belasting(ToetsGolfklappen & karakParam>=C1) = 2;
%belasting(ToetsGolfklappen & karakParam<0.2) = 0;
ToetsGolfklappen = ToetsGolfklappen & karakParam<C1 & karakParam>=0.2;

if ~strcmp(grasmat,'slecht')
   switch grasmat
```

```
   case 'goed'
      C2 = 2.3;
      C3 = -0.52;
   case 'matig'
      C2 = 1.6;
      C3 = -0.52;
   end
   maximale_belastingduur      = (karakParam/C2).^(1/C3);
   belasting(ToetsGolfklappen) = tijdstap./maximale_belastingduur(ToetsGolfklappen);
end

%
% Toets op stroming ...
%
% STAP 1
%
z=z_ondergrens-sl;
%
% Voorkom deling door nul. Het maakt niet uit welke waarde z_q krijgt
% want z ligt tussen 0 en z_q (als ToetsStroming waar is).
%
z_q(z_q==0)=1e-10;
%
g2pi=9.81/(2*pi);
rekensnelheid = 700*(H_s./T_p).*(0.085 - H_s./(g2pi*T_p.^2)).*sqrt(1-
(z./z_q)).*helling;
%
%STAP 2
%
switch grasmat
case 'goed'
    C1 = 1560;
    C2 = -4.8;
case 'matig'
    C1 = 276;
    C2 = -4.2;
case 'slecht'
    C1 = 50;
    C2 = -3.6;
end
maximale_belastingduur = C1*rekensnelheid.^C2;
%
%STAP 3
%
z=z_ondergrens-sl;
%
effectieve_tijdstap = (1 - z./z_q) * tijdstap;
%
%STAP 4
%
belasting(ToetsStroming) = ...
   effectieve_tijdstap(ToetsStroming) ./ maximale_belastingduur(ToetsStroming);

%
% Bepaal begin en eind van alle stormen in dataset en
% sommeer belasting (=ts/tsmax) voor elke storm.
%
Start=cat(1,1,find(diff(WaarnStormIndex))+1,length(WaarnStormIndex)+1);
for storm = 1:length(Start)-1
   Storm=Start(storm):Start(storm+1)-1;
   belasting(Storm)=cumsum(belasting(Storm));
end

q = belasting;
Z = sterkte - belasting;
```

Stat/StatBerekenZSteenzetting

```
function [Z,q] = StatBerekenZSteenzetting (Profiel, ZSegment, H_s, T_p, sl, theta_0)

%STATBEREKENZSTEENZETTING   In StatBerekenZSteenzetting wordt de Z-functie
%     uitgerekend voor het faalmechanisme instabiliteit voor steenzettingen.
%  AANROEP
```

```
%     [Z,q] = BerekenZSteenzetting (Profiel, Blok, H_s, T_p, sl, theta_0)
%  INVOER
%     Profiel:    het dijkprofiel
%     ZSegment:   karakteristiek van het dijksegment (ondergrens & bovengrens,
%                 en bekledingsgegevens.
%     H_s:        significante golfhoogte (nearshore)
%     T_p:        de piekperiode (nearshore)
%     sl:         waterstand (nearshore)
%     theta_0:    golfrichting (nearshore) (t.o.v. het noorden, in radialen)
%  UITVOER
%     Z:          Z is gelijk aan de sterkte van de steenzetting minus de
%                 belasting
%     q:          de belasting q

% Auteur: Bert Jagers, WL | Delft Hydraulics

% 4-11-2003
%
% Corrigeer voor negatieve golfhoogtes (komen een enkele keer
% voor indien de stormflanken worden toegevoegd.
%
H_s=max(H_s,1e-10);
T_p=max(T_p,1e-10);
%
% Globale constanten ...
%
g2pi=9.81/(2*pi);
%
% Bekledingsgegevens ...
%
toplaagdikte    = ZSegment.bekledingsgegevens.toplaagdikte;
filterkleidikte = ZSegment.bekledingsgegevens.filterkleidikte;
D15             = ZSegment.bekledingsgegevens.D15;
reldichtheid    = ZSegment.bekledingsgegevens.relatievedichtheid;
bekledingstype  = ZSegment.bekledingsgegevens.bekledingstype; %bijv. 'ingeklemde
stenen op granulair filter'
toetsgrens      = ZSegment.bekledingsgegevens.toetsgrens;
%'goed/twijfelachtig','twijfelachtig/onvoldoende'
constructie     = ZSegment.bekledingsgegevens.constructie; %'normaal','ongunstig'

%
% Oploop niveau t.o.v. SWL ...
%
z_q             = StatBerekenOploopniveau(Profiel, 0.1, H_s, T_p, sl, theta_0);
%
% Verticale locatie van het segment in het dijkprofiel ...
%
z_ondergrens    = ZSegment.ondergrens;
z_bovengrens    = ZSegment.bovengrens;
%
% Bepaal helling ...
%
%helling        = BepaalHelling(1.5,H_s,sl,Profiel);
helling             = repmat(Profiel.helling_laag,size(sl));
z_berm_o            = Profiel.hoogte_berm;
z_berm_b            = Profiel.hoogte_berm +
Profiel.breedte_berm*Profiel.helling_berm;
bekleding_op_berm   = z_ondergrens>=z_berm_o & z_bovengrens<=z_berm_b;
bekleding_boven_berm = z_ondergrens>=z_berm_b & z_bovengrens>z_berm_b;
helling(bekleding_boven_berm) = Profiel.helling_hoog;

%
% Stap 0: Afschuiving
%    Alleen als de bekleding niet op de berm ligt.
%
if bekleding_op_berm
   sterkte         = repmat(1,size(sl));
   belasting       = repmat(0,size(sl));
   resterend       = repmat(logical(1),size(sl));
else
   %
   sterkte         = reldichtheid*toplaagdikte+filterkleidikte;
   sterkte         = repmat(sterkte,size(sl));
   %
   % De volgende formule geeft de grens tussen GOED en TWIJFELACHTIG
```

```
   %
   belastingbovensegment   = sl - 2*H_s > z_bovengrens;
   belastingondersegment   = sl < z_ondergrens;
   belasting       = min(0.16*(H_s.^0.2).*(T_p.^1.6).*(helling.^0.8),1.5*H_s)-
1334*D15*(1-1.19*helling).*sqrt(T_p);
   %
   % Als sterkte > belasting, dan is de dijk GOED met betrekking tot
   % afschuiving en moeten we dus kijken naar de toplaag stabiliteit.
   % Als sterkte < belasting, dan is de dijk TWIJFELACHTIG of ONVOLDOENDE
   % en dus zullen we deze hier interpreteren als zijnde gefaald.
   %
   resterend       = (sterkte > belasting) | belastingbovensegment |
belastingondersegment;
   %
end
%
% Stap 1: Toplaag stabiliteit
%
% Bepaal hoogte waar de golfklappen de dijkbekleding raken.
%
impacthoogte     = sl - min(0.11*H_s.*(g2pi * (T_p.^2) .* helling ./
H_s).^0.8,1.5*H_s);
%
% Controleer of de locatie van golfklap of de effectieve golfoploop
% het dijksegment met de steenzetting omvat.
%
belastingbovensegment   = impacthoogte > z_bovengrens;
belastingondersegment   = sl + z_q/2 < z_ondergrens;
%
% Niet in bereik ---> gebruik de hierboven gedefinieerde sterkte en
% belasting betreffende het proces afschuiving.
%
resterend        = resterend & ~belastingbovensegment & ~belastingondersegment;
%
% Bij belasting rond de ondergrens van het dijksegment met steen-
% zetting vergroot de effectieve toplaagdikte en ga door met de
% toets op toplaagstabiliteit.
%
belastingrondondergrens = sl < z_ondergrens & ~belastingondersegment;
toplaagdikte     = repmat(toplaagdikte,size(sl));
toplaagdikte(belastingrondondergrens) = toplaagdikte(1)/0.8;
%
% Bij steenzetting op de berm wordt de toplaagdikte ook aangepast.
%
if bekleding_op_berm
   dB            = sl - z_berm_o;
   Cberm         = 0.85*exp(-0.8*(-0.9+dB./H_s).^2)+0.7*exp(-0.5*(-2.1+dB./H_s).^2);
   toplaagdikte  = toplaagdikte./Cberm;
end

%
% Stap 2: Berekenen stabiliteitsgetal en golfbrekerparameter
%
stabiliteitsgetal = H_s./(reldichtheid*toplaagdikte);
golfbrekerparam   = helling./sqrt(H_s./(g2pi*T_p.^2));

%
% Stap 3: Bepaal de grenswaarde voor het stabiliteitsgetal
%
% De grenswaarde "n_s;onder" gebruikt hier is de grens tussen
% GOED en TWIJFELACHTIG (of TWIJFELACHTIG en ONVOLDOENDE). Deze
% grenswaarde wordt hier aangeduid als bovengrens_SG omdat het
% de bovengrens vormt voor de range van stabiliteitsgetallen
% waarin de toplaag stabiel is.
%
bt_indices=[1 2 3 6 7 8];
bt = strmatch(bekledingstype,{'ingeklemde stenen op geotextiel op zand/klei', ...
                              'ingeklemde stenen direct op goede klei', ...
                              'ingeklemde stenen op granulair filter', ...
                              'geschakelde stenen op geotextiel op zand/klei', ...
                              'geschakelde stenen direct op goede klei', ...
                              'geschakelde stenen op granulair filter'});
bti = bt_indices(bt);
switch bekledingstype
```

```
case {'ingeklemde stenen op granulair filter', 'geschakelde stenen op granulair
filter'}
   switch constructie
   case 'goed'
   case 'normaal'
      bti=bti+1;
   case 'ongunstig'
      bti=bti+2;
   end
end

switch toetsgrens
case 'goed/twijfelachtig'
   tbl = [ 4.31    -0.926   11     -4     0.09     1.38   2.2
           3.75    -1.001   8      -4     0.02     1.25   2.4
           4.58    -0.903   14.5   -4     0.17     1.27   2.2
           4.08    -1.014   11.0   -4     0.03     1.25   2.0
           3.07    -1.014   6.5    -4     0.02     1.09   2.0
           5.192   -0.817   21     -4     0.33     1.18   2.3
           4.31    -0.926   11     -4     0.09     1.38   2.2
           5.06    -0.783   23     -4     0.33     1.1    2.4
           4.53    -0.886   15.0   -4     0.14     1.28   2.4
           3.97    -0.96    12     -4     0.06     1.18   2.6];
case 'twijfelachtig/onvoldoende'
   tbl = [ 6.78    -0.588   17     -2     1.84    -3.25   2.2
           6.1     -0.75    11     -2     0.98    -1.0    2.1
           7.12    -0.539   17.8   -1.5   2.54    -6.32   2.2
           6.68    -0.723   12.0   -1.5   1.5     -3.12   2.1
           5.08    -0.785   13.8   -4     0.26     1.53   2.3
           8.1     -0.47    26     -0.5   3.8    -20.03   1.8
           6.78    -0.588   17     -2     1.84    -3.25   2.2
           7.97    -0.435   30     -0.5   4.2    -23.6    2.0
           7.3     -0.6     28.0   -0.5   3.4    -21.68   2.0
           6.5     -0.7     12     -1     1.62    -5.23   2.0];
end
gbpGTgrens                = golfbrekerparam>tbl(bti,7);
bovengrens_SG             = tbl(bti,1)*golfbrekerparam.^tbl(bti,2);
bovengrens_SG(gbpGTgrens) = tbl(bti,3)*golfbrekerparam(gbpGTgrens).^tbl(bti,4) ...
                            + tbl(bti,5)*golfbrekerparam(gbpGTgrens) ...
                            + tbl(bti,6);

sterkte(resterend) = bovengrens_SG(resterend) ./ max(stabiliteitsgetal(resterend),1e-
20);
belasting(resterend) = 1;

%
% Uitvoer grootheden ...
%
q = belasting;
Z = sterkte - belasting;
```

Stat/stormFlanken

```
function Toevoegen=stormFlanken
%Tijdelijke oplossing voor het toevoegen van de flanken
%warning('Stormflanken worden toegevoegd.')
%De geretourneerde waarde is meteen de te gebruiken flankduur.
persistent flankduur

if isempty( flankduur )
   inistring = IniFile( 'LEES', 'general', 'duur_stormflanken', '0' );
   flankduur = str2num(inistring);
end

Toevoegen=flankduur;
```

Ui/CBbekleding

```
function Uitvoer = CBbekleding (Commando,Bestandsnaam)

%CBBEKLEDING            afhandelen callback van de pushbutton voor het
```

```
%                       wijzigen van de dijkbekleding voor de instabiliteit
%                       van de bekleding.
%  AANROEP EXTERN
%     CBbekleding ('SHOW')
%     Datastructuur = CBbekleding ('LEES',Bestandsnaam)
%  AANROEP INTERN
%     CBbekleding (Commando)
%  INVOER
%     Commando: Voor extern gebruik 'SHOW' en 'LEES', de overige commando's
%               zijn voor intern gebruik.
%     Bestandsnaam: In het geval van 'LEES' dient de aanroepende functie de
%                   naam van een dijkopbouwbestand mee te geven.
%     Datastructuur: In het geval van 'LEES' wordt een dijkopbouw structure
%                    teruggegeven.

% Auteur: Bert Jagers, WL | Delft Hydraulics

switch upper(Commando)
case 'SHOW'
   ToonVenster(gcbo)
case 'CLOSE'
   set(gcbf,'closerequestfcn','closereq')
case 'SEGMENT'
   SelecteerAnderSegment(gcbo)
case 'TOEVOEGEN'
   ToevoegenSegment
case 'VERWIJDEREN'
   VerwijderSegment
case 'BEKLEDING'
   SelecteerAndereBekleding
case 'EDIT'
   HaalData
case 'LEES'
   if isempty(gcbf)
      % Vanuit batch mode
      Uitvoer = LeesBekleding(Bestandsnaam);
   else
      % Vanuit user interface
      UiData = get(gcbf,'userdata');
      Dijkopbouw = LeesBekleding;
      if ~isempty(Dijkopbouw)
         UiData.dijkopbouw = Dijkopbouw;
      end
      set(gcbf,'userdata',UiData)
      VernieuwVenster(gcbf)
   end
case 'SCHRIJF'
   UiData = get(gcbf,'userdata');
   SchrijfBekleding(UiData.dijkopbouw);
otherwise
   fprintf('Onbekend commando: %s\n',Commando)
end


function SelecteerAndereBekleding
%
% Gebruiker heeft een ander type bekleding geselecteerd. Verander de
% eigenschappen van dit bekledingssegment overeenkomstig het nieuwe
% bekledingstype en update de user interface.
%
UiData = get(gcbf,'userdata');
i = get(UiData.handles.segmentscroller,'value');
bekledingStrings = get(UiData.handles.type,'string');
bekledingNr = get(UiData.handles.type,'value');
bekleding = bekledingStrings{bekledingNr};
%
% Als hetzelfde bekledingstype ook al elders op de dijk voorkomt, neem dan
% initieel dezelfde instellingen voor de bekledingsgegevens. Waarschijnlijk
% komen die instellingen beter overeen dan de default waarden die hier hard
% in de code staan.
%
gegevens = [];
for ii = 1:length(UiData.dijkopbouw)
   if strcmp(UiData.dijkopbouw(ii).bekleding,bekleding)
      gegevens = UiData.dijkopbouw(ii).bekledingsgegevens;
```

```
      break
   end
end
%
% Als hetzelfde bekledingstype nog niet elders op de dijk voorkomt, neem
% dan de onderstaande standaard waarden.
%
if isempty(gegevens)
   switch bekleding
   case 'asfalt'
      gegevens.laagdikte = 0.15;
      gegevens.ondergrondtype = 'klei';
      gegevens.asfalttype = 'waterbouwasfaltbeton';
      gegevens.filtertype = 'geotextiel';
      gegevens.rho_water  = 1025;
      gegevens.rho_asfalt = 2100;
      gegevens.z_ogb = 0;
   case 'blokken'
      gegevens.dichtheid_hoog = 1.6;
      gegevens.dichtheid_laag = 1.6;
      gegevens.blokdikte_hoog = 0.4;
      gegevens.blokdikte_laag = 0.4;
   case 'gras'
      gegevens.kwaliteit = 'goed';
   case 'steenzetting'
      gegevens.toplaagdikte = 0.4;
      gegevens.filterkleidikte = 3;
      gegevens.D15 = 0.001;
      gegevens.relatievedichtheid = 1.6;
      gegevens.bekledingstype = 'ingeklemde stenen op granulair filter';
      gegevens.toetsgrens = 'goed/twijfelachtig';
      gegevens.constructie = 'normaal';
   end
end
%
% Sla de nieuwe gegevens op in de datastructuur ...
%
UiData.dijkopbouw(i).bekleding = bekleding;
UiData.dijkopbouw(i).bekledingsgegevens = gegevens;
set(gcbf,'userdata',UiData);
%
% ... en update de user interface.
%
VernieuwVenster(gcbf)


function HaalData
%
% Lees data uit user interface objecten (met uitzondering van het nummer
% van het bekledingssegment en het bekledingstype).
%
UiData = get(gcbf,'userdata');
i = get(UiData.handles.segmentscroller,'value');
%
% Bepaal de locatie van het bekledingssegment op de dijk.
%
UiData.dijkopbouw(i).ondergrens =
HaalWaarde(UiData.handles.grenzen(2),UiData.dijkopbouw(i).ondergrens);
UiData.dijkopbouw(i).bovengrens =
HaalWaarde(UiData.handles.grenzen(5),UiData.dijkopbouw(i).bovengrens);
%
% Bereid het lezen van de bekledingsgegevens voor door indices in de
% UiData.handles array ...
%
controls=indextabel(UiData.dijkopbouw(i).bekleding);
%
% Generieke code om alle edit velden van het geselecteerde bekledingstype
% uit te lezen.
%
H = getfield(UiData.handles,UiData.dijkopbouw(i).bekleding);
gegevens = UiData.dijkopbouw(i).bekledingsgegevens;
for j = 1:size(controls)
   HuidigeWaarde = getfield(gegevens,controls{j,1});
   NieuweWaarde = HaalWaarde(H(controls{j,2}),HuidigeWaarde);
   gegevens = setfield(gegevens,controls{j,1},NieuweWaarde);
```

```
end
%
% Update data die afgeleid kan worden uit de gelezen data.
%
switch UiData.dijkopbouw(i).bekleding
case 'blokken'
   gegevens.dichtheid_laag = gegevens.dichtheid_hoog;
   gegevens.blokdikte_laag = gegevens.blokdikte_hoog;
end
UiData.dijkopbouw(i).bekledingsgegevens = gegevens;
set(gcbf,'userdata',UiData)
%
% Update de user interface (in het bijzonder de plot van het dijkprofiel).
%
VernieuwVenster(gcbf)


function Waarde = HaalWaarde(control,standaardWaarde)
%
% Hulpfunctie om de waarde van een enkele edit control in de user interface
% uit te lezen.
%
switch get(control,'style')
case 'edit'
   Waarde = sscanf(get(control,'string'),'%f',1);
   Grenzen = get(control,'userdata');
   if ~isempty(Grenzen)
      if Waarde<Grenzen(1) | Waarde>Grenzen(2)
         Waarde = standaardWaarde;
      end
   end
   set(control,'string',num2str(Waarde))
case 'popupmenu'
   Strings = get(control,'string');
   i = get(control,'value');
   Waarde = Strings{i};
end


function VerwijderSegment
%
% Verwijder het geselecteerde segment.
%
UiData = get(gcbf,'userdata');
i = get(UiData.handles.segmentscroller,'value');
UiData.dijkopbouw(i) = [];
set(gcbf,'userdata',UiData);
%
% Kies een ander te editen segment. Pas de segmentscroller waarde aan; deze
% is maatgevend bij het updaten van de user interface.
%
AantalLagen = length(UiData.dijkopbouw);
if i>AantalLagen
   i = i-1;
end
set(UiData.handles.segmentscroller,'max',AantalLagen,'value',i)
%
% Update de user interface.
%
VernieuwVenster(gcbf)


function ToevoegenSegment
%
% Voeg een nieuw segment toe en laat dat een kopie zijn van het
% huidige segment.
%
UiData = get(gcbf,'userdata');
i = get(UiData.handles.segmentscroller,'value');
UiData.dijkopbouw(end+1) = UiData.dijkopbouw(i);
set(gcbf,'userdata',UiData);
%
% Pas de range en de waarde van de segment scroller aan (deze is maatgevend
% bij het updaten van de user interface).
%
```

```
AantalLagen = length(UiData.dijkopbouw);
set(UiData.handles.segmentscroller,'max',AantalLagen,'value',AantalLagen)
%
% Update de user interface.
%
VernieuwVenster(gcbf)


function SelecteerAnderSegment(control)
%
% De gebruiker heeft een ander segment geselecteerd hetzij via het edit
% venster, hetzij via de segment scroller. Welke van de twee de gebruiker
% gebruikt heeft, wordt bepaald door het invoerargument "control".
%
UiData = get(gcbf,'userdata');
if control==UiData.handles.segment
   Laag = sscanf(get(control,'string'),'%i',1);
else
   Laag = round(get(control,'value'));
end
AantalLagen = length(UiData.dijkopbouw);
if Laag<1 | Laag>AantalLagen
   Laag = Data(1);
end
%
% Pas de segmentscroller aan (deze is maatgevend bij het updaten van de
% user interface).
%
set(UiData.handles.segmentscroller,'max',AantalLagen,'value',Laag)
%
% Update de user interface.
%
VernieuwVenster(gcbf)


function SchrijfBekleding(Dijkopbouw)
%
% Schrijf bekleding naar file. Laat de gebruiker een bestandsnaam selecteren.
%
[bestand,pad] = uiputfile('*.bekleding');
if ~ischar(bestand)
   return
end
%
% Voeg de extensie .bekleding toe indien er geen extensie gespecificeerd
% is.
%
[p,b,e] = fileparts(bestand);
if isempty(e)
   bestand = [bestand '.bekleding'];
end
bestand = fullfile(pad,bestand);
%
% Schrijf het bestand ...
%
fid = fopen(bestand,'wt');
fprintf(fid,'HYDRA-K DIJKOPBOUWBESTAND\n');
AantalLagen = length(Dijkopbouw);
if AantalLagen == 1
   fprintf(fid,'1 LAAG\n');
else
   fprintf(fid,'%i LAGEN\n',AantalLagen);
end
for i = 1:AantalLagen
   fprintf(fid,'%s\n',Dijkopbouw(i).bekleding);
   fprintf(fid,'%f Ondergrens in m+NAP\n',Dijkopbouw(i).ondergrens);
   fprintf(fid,'%f Bovengrens in m+NAP\n',Dijkopbouw(i).bovengrens);
   gegevens = Dijkopbouw(i).bekledingsgegevens;
   switch Dijkopbouw(i).bekleding
   case 'asfalt'
      fprintf(fid,'%f Laagdikte in m\n',gegevens.laagdikte);
      fprintf(fid,'%s Ondergrondtype\n',gegevens.ondergrondtype);
      fprintf(fid,'%s Asfalttype\n',gegevens.asfalttype);
      fprintf(fid,'%s Filtertype\n',gegevens.filtertype);
      fprintf(fid,'%s Dichtheid water in kg/m^3\n',gegevens.rho_water);
```

```
      fprintf(fid,'%s Dichtheid asfalt in kg/m^3\n',gegevens.rho_asfalt);
      fprintf(fid,'%s Ondergrens gesloten bekleding in m+NAP\n',gegevens.z_ogb);
   case 'blokken'
      fprintf(fid,'%f Relatieve dichtheid\n',gegevens.dichtheid_hoog);
      fprintf(fid,'%f Blokdikte in m\n',gegevens.blokdikte_hoog);
   case 'gras'
      fprintf(fid,'%s Grasmat kwaliteit\n',gegevens.kwaliteit);
   case 'steenzetting'
      fprintf(fid,'%f Toplaagdikte in m\n',gegevens.toplaagdikte);
      fprintf(fid,'%f Totale dikte van filter- en kleilaag in
m\n',gegevens.filterkleidikte);
      fprintf(fid,'%f D15 van zand in m\n',gegevens.D15);
      fprintf(fid,'%f Relatieve dichtheid\n',gegevens.relatievedichtheid);
      fprintf(fid,'%s Type bekleding\n',gegevens.bekledingstype);
      fprintf(fid,'%s Toetsgrens\n',gegevens.toetsgrens);
      fprintf(fid,'%s Type constructie\n',gegevens.constructie);
   end
end
fclose(fid);


function VernieuwGrafiek(UiFig)
%
% Vernieuw de plot van het dijkprofiel.
%
UiData = get(UiFig,'userdata');
segment = get(UiData.handles.segmentscroller,'value');
%
% Haal de constanten uit de structure.
%
breedte_berm = UiData.profiel(1).breedte_berm;

helling_laag = UiData.profiel(1).helling_laag;
helling_berm = UiData.profiel(1).helling_berm;
helling_hoog = UiData.profiel(1).helling_hoog;

hoogte_teen  = UiData.profiel(1).hoogte_teen;
hoogte_berm  = UiData.profiel(1).hoogte_berm;
hoogte_berm2 = hoogte_berm+breedte_berm*helling_berm;
if hoogte_berm2 == hoogte_berm
   hoogte_berm2u = hoogte_berm+0.00001;
else
   hoogte_berm2u = hoogte_berm;
end
hoogte_kruin = UiData.profiel(1).hoogte_kruin;

breedte_laag = (hoogte_berm-hoogte_teen)/helling_laag;
breedte_hoog = (hoogte_kruin-hoogte_berm2u)/helling_hoog;
%
% Teken het dijkprofiel.
%
xx = cumsum([0 breedte_laag breedte_berm breedte_hoog]);
zz = [hoogte_teen hoogte_berm hoogte_berm2u hoogte_kruin];
area(xx,zz,hoogte_teen,'facecolor',[.8 .8
.8],'edgecolor','none','parent',UiData.assenstelsel);
%
% Voeg de teksten langs de assen toe.
%
set(UiData.assenstelsel,'ylim',[hoogte_teen hoogte_kruin])
set(get(UiData.assenstelsel,'xlabel'),'string','afstand (m) \rightarrow');
set(get(UiData.assenstelsel,'ylabel'),'string','hoogte (m+NAP) \rightarrow');
set(UiData.assenstelsel,'layer','top')

%
% Teken de bekledingssegmenten ... waarbij het actieve segment als een wat
% dikkere rode lijn wordt getekend.
%
AantalLagen = length(UiData.dijkopbouw);
Segment = UiData.dijkopbouw(segment);

for i = 1:AantalLagen
   z = sort([UiData.dijkopbouw(i).ondergrens UiData.dijkopbouw(i).bovengrens]);
   z = [max(hoogte_teen,z(1)) min(hoogte_kruin,z(2))];
   %
   % Houd rekening met de berm ...
```

```
   %
   if hoogte_berm == hoogte_berm2
      if z(1) == z(2) & z(1)==hoogte_berm
         z(2)=hoogte_berm2u;
      elseif z(1) == hoogte_berm
         % berm is ondergrens: verplaats in de figuur naar de landzijde van de berm
         z(1)=hoogte_berm2u;
      elseif z(2) == hoogte_berm
         % berm is bovengrens: geen aanpassing nodig
      elseif z(1) < hoogte_berm2u & hoogte_berm2u < z(2)
         z = [z(1) hoogte_berm2u z(2)];
      end
   elseif z(1) < hoogte_berm2u & hoogte_berm2u < z(2)
      z = [z(1) hoogte_berm2u z(2)];
   end
   if z(1) < hoogte_berm & hoogte_berm < z(2)
      z = [z(1) hoogte_berm z(2:end)];
   end
   %
   % Teken lijn ...
   %
   L = line(interp1(zz,xx,z),z);
   if i == segment
      set(L,'color','r','linewidth',2)
   end
end


function VernieuwVenster(UiFig)
%
% Vernieuw het dialoogvenster.
%
UiData = get(UiFig,'userdata');
Actief = [1 1 1];
NietActief = get(UiFig,'color');
%
% Zet de selectie van het actieve bekledingssegment goed.
%
AantalLagen = length(UiData.dijkopbouw);
if AantalLagen == 1
   set(UiData.handles.segment,'string','1','enable','inactive', ...
      'backgroundcolor',NietActief,'tooltip','')
   set(UiData.handles.segmentscroller,'value',1,'max',2,'enable','off')
   set(UiData.handles.segmentverwijder,'enable','off');
   segment = 1;
else
   segment = get(UiData.handles.segmentscroller,'value');
   if segment > AantalLagen
      segment = 1;
   end

set(UiData.handles.segment,'string',num2str(segment),'enable','on','backgroundcolor',
Actief,'tooltip',sprintf('1 tot %i',AantalLagen))

set(UiData.handles.segmentscroller,'value',segment,'max',AantalLagen,'enable','on','s
liderstep',[min(1/(AantalLagen-1),0.9) 1])
   set(UiData.handles.segmentverwijder,'enable','on');
end
%
% Selecteer het betreffende bekledingssegment.
%
Segment = UiData.dijkopbouw(segment);
%
% Toon de locatie en het bekledingstype
%
set(UiData.handles.grenzen(2),'string',num2str(Segment.ondergrens))
set(UiData.handles.grenzen(5),'string',num2str(Segment.bovengrens))
bekledingstypen = get(UiData.handles.type,'string');
set(UiData.handles.type,'value',strmatch(Segment.bekleding,bekledingstypen))
%
% Verberg de edit objecten voor de niet geselecteerde bekledingstypes.
%
for i=1:length(bekledingstypen)
   bekledingstype=bekledingstypen{i};
   switch bekledingstype
```

```
      otherwise
         field=bekledingstype;
   end
   onoff='off';
   Htemp = getfield(UiData.handles,field);
   if strcmp(bekledingstype,Segment.bekleding)
      onoff='on';
      H = Htemp;
   end
   set(Htemp,'visible',onoff)
end
%
% Toon voor het geselecteerde bekledingstype de edit objecten.
%
controls=indextabel(Segment.bekleding);
for j = 1:size(controls)
   HuidigeWaarde = getfield(Segment.bekledingsgegevens,controls{j,1});
   Hcontrol = H(controls{j,2});
   switch get(Hcontrol,'style')
   case 'edit'
      set(Hcontrol,'string',num2str(HuidigeWaarde));
   case 'popupmenu'
      Strings = get(Hcontrol,'string');
      Waarde = strmatch(HuidigeWaarde,Strings);
      set(Hcontrol,'value',Waarde);
   end
end
%
% Vernieuw de plot van het dijkprofiel.
%
VernieuwGrafiek(UiFig)


function ToonVenster(PushButtonHandle)
%TOONVENSTER Toon het edit venster voor dijkbekleding.
%
% Initialisatie van het venster en vraag het profiel en de bekleding op.
%
parent = get(PushButtonHandle,'Parent');
uiWaarden = FGhydra_k ('WAARDEN', parent);
UiFig = figure('numbertitle','off','integerhandle','off','name','Hydra-K
bekleding','menubar','none','visible','off','resize','off');
set(UiFig,'position',get(UiFig,'position')+[0 0 200 0]);
UiData.assenstelsel = axes('units','normalized','position',[.48 .1 .50 .86]);
UiData.profiel = uiWaarden.profiel;
UiData.dijkopbouw = uiWaarden.dijkopbouw;
%
% Initialiseer de weergave van de edit objecten.
%
set(UiFig,'units','points');
UiFigPositie = get(UiFig,'position');
voffset = UiFigPositie(4)-30;
%
% Lees en schrijf knoppen (het is niet mogelijk om in een "modal"
% dialoogvenster menu's weer te geven, vandaar dat we knoppen gebruiken).
%
set(maakTextControl(UiFig,'right',[10 voffset 75 15],''),'string','Bestand:')
ph=maakPushbuttonControl(UiFig,[90 voffset],[],'CBbekleding LEES','Lees bekleding uit
een bestand');
set(ph,'string','Lees','position',get(ph,'position')-[0 0 4 0])
ph=maakPushbuttonControl(UiFig,[140 voffset],[],'CBbekleding SCHRIJF','Schrijf
bekleding naar een bestand');
set(ph,'string','Schrijf','position',get(ph,'position')-[0 0 4 0])
%
% Objecten voor de selectie van het bekledingssegment (edit venster en
% scroller).
%
voffset = voffset-30;
set(maakTextControl(UiFig,'right',[10 voffset 75
15],''),'string','Bekledingssegment:')
UiData.handles.segment = maakEditControl(UiFig,'right',[90 voffset 45
15],'CBbekleding SEGMENT',1,'Selecteer het bekledingssegment');
UiData.handles.segmentscroller =
uicontrol('style','slider','parent',UiFig,'units','points','position',[135 voffset 10
15],'min',1,'max',2,'value',1,'enable','off','callback','CBbekleding SEGMENT');
```

```
%
% Toevoeg en verwijder knoppen voor bekledingssegmenten.
%
voffset = voffset-18;
ph=maakPushbuttonControl(UiFig,[90 voffset],[],'CBbekleding TOEVOEGEN','Voeg een
segment toe');
set(ph,'string','Toevoegen','position',get(ph,'position')-[0 0 4 0])
ph=maakPushbuttonControl(UiFig,[140 voffset],[],'CBbekleding VERWIJDEREN','Verwijder
dit segment');
set(ph,'string','Verwijderen','position',get(ph,'position')-[0 0 4 0])
UiData.handles.segmentverwijder=ph;
%
% Positie van de bekleding op het profiel.
%
voffset = voffset-30;
set(maakTextControl(UiFig,'left',[10 voffset 120 13],'A'),'string','Positie van
bekleding op dijk:')
velden={'Ondergrens:','Ondergrens in m+NAP',[],'m+NAP'
   'Bovengrens:','Bovengrens in m+NAP',[],'m+NAP'};
[UiData.handles.grenzen,voffset] = maakEditVelden (UiFig,voffset,velden,'on');
%
% Bekledingstype selectie lijst.
%
voffset = voffset-30;
set(maakTextControl(UiFig,'right',[10 voffset 75 13],''),'string','Bekledingstype:')
UiData.handles.type = uicontrol('style','popupmenu', ...
   'parent',UiFig, ...
   'units','points', ...
   'position',[90 voffset 150 15], ...
   'string',{'asfalt','blokken','gras','steenzetting'}, ...
   'backgroundcolor','w', ...
   'callback','CBbekleding BEKLEDING');
voffsetref = voffset;
%
% Bekledingstype specifieke gegevens.
%
%----- ASFALT
voffset = voffsetref-15;
velden={'Laagdikte:','',[],'m'};
[H1,voffset] = maakEditVelden (UiFig,voffsetref-15,velden,'off');
velden={'Ondergrondtype:',{'zand','klei'},'ondergrondtype'
   'Asfalttype:',{'waterbouwasfaltbeton','opensteenasfalt'},'asfalttype'
   'Filtertype:',{'geotextiel','zandasfalt'},'filtertype (alleen bij
opensteenasfalt)'};
[H2,voffset] = maakPopupmenuVelden (UiFig,voffset,velden,'off');
velden={'Dichtheid water:','Ten behoeve van toets op opdrukken',[],'kg/m^3'
   'Dichtheid asfalt:','Ten behoeve van toets op opdrukken',[],'kg/m^3'
   'Ondergr. gesl. bekl.:','Ondergrens gesloten bekleding ten behoeve van toets op
opdrukken',[],'m+NAP'};
[H3,voffset] = maakEditVelden (UiFig,voffset,velden,'off');
UiData.handles.asfalt = cat(2,H1,H2,H3);
%
%----- BLOKKEN
voffset = voffsetref-15;
velden={'Relatieve dichtheid:','1 <= dichtheid <= 5',[1 5],''
   'Blokdikte:','0 <= blokdikte <= 10',[0 10],'m'};
UiData.handles.blokken = maakEditVelden (UiFig,voffset,velden,'off');
%
%----- GRAS
voffset = voffsetref-15;
velden={'Kwaliteit:',{'goed','matig','slecht'},'kwaliteit van de grasmat'};
UiData.handles.gras = maakPopupmenuVelden (UiFig,voffset,velden,'off');
%
%----- STEENZETTING
voffset = voffsetref-15;
velden={'Toplaagdikte:','',[],'m'
   'Dikte filter+kleilaag:','',[],'m'
   'D15 van zand:','',[],'m'
   'Relatieve dichtheid:','1 <= dichtheid <= 5',[1 5],''};
[H1,voffset] = maakEditVelden (UiFig,voffset,velden,'off');
velden={'Type steenzetting:',{'ingeklemde stenen op geotextiel op zand/klei', ...
                          'ingeklemde stenen direct op goede klei', ...
                          'ingeklemde stenen op granulair filter', ...
                          'geschakelde stenen op geotextiel op zand/klei', ...
                          'geschakelde stenen direct op goede klei', ...
```

```
                          'geschakelde stenen op granulair filter'}, ...
                          'type vande steenzetting'
        'Toetsgrens:',{'goed/twijfelachtig','twijfelachtig/onvoldoende'},'grens te
gebruiken voor toetsing'
        'Type constructie:',{'goed','normaal','ongunstig'},'kwaliteit van de
constructie (alleen bij stenen op granulair filter)'};
H2 = maakPopupmenuVelden (UiFig,voffset,velden,'off');
UiData.handles.steenzetting = cat(2,H1,H2);
%
%-----
%
% Toon venster en wacht op venster afsluiting.
%
set(UiFig,'userdata',UiData,'windowstyle','modal','closerequestfcn','CBbekleding
CLOSE','visible','on');
VernieuwVenster(UiFig)
waitfor(UiFig,'closerequestfcn')
UiData = get(UiFig,'userdata');
delete(UiFig);
if ~isempty(UiData)
   set(PushButtonHandle,'userdata',UiData.dijkopbouw)
   UiData.naam='instabiliteit van de bekleding';
   [Fout,Foutmelding]=controleerBekleding(UiData);
   if Fout
      uiwait(msgbox(Foutmelding))
   end
end


function [H,voffset] = maakEditVelden (UiFig,voffset,velden,visible)
% MAAKEDITVELDEN
%
% Hulpfunctie voor het tonen van edit controls.
%
b = 0;
for i = 1:size(velden,1)
   voffset = voffset-15;
   H(b+1) = maakTextControl(UiFig,'right',[10 voffset 75 13],'');
   set(H(b+1),'string',velden{i,1},'visible',visible)
   H(b+2) = maakEditControl(UiFig,'right',[90 voffset 45 15],'CBbekleding
EDIT',velden{i,3},velden{i,2});
   set(H(b+2),'visible',visible)
   H(b+3) = maakTextControl(UiFig,'left',[140 voffset 55 13],'');
   set(H(b+3),'string',velden{i,4},'visible',visible)
   b=b+3;
end


function [H,voffset] = maakPopupmenuVelden (UiFig,voffset,velden,visible)
% MAAKLISTVELDEN
%
% Hulpfunctie voor het tonen van popupmenu controls.
%
b = 0;
for i = 1:size(velden,1)
   voffset = voffset-15;
   H(b+1) = maakTextControl(UiFig,'right',[10 voffset 75 13],'');
   set(H(b+1),'string',velden{i,1},'visible',visible)
   H(b+2) = uicontrol('parent',UiFig,'style','popupmenu', ...
      'horizontalalignment','left','string',velden{i,2}, ...
      'units','points','position',[90 voffset 150 15],'visible',visible, ...
      'backgroundcolor','w','tooltip',velden{i,3}, ...
      'callback','CBbekleding EDIT');
   b=b+2;
end


function controls=indextabel(bekleding)
switch bekleding
case 'asfalt'
   controls={'laagdikte',2
      'ondergrondtype',5
      'asfalttype',7
      'filtertype',9
      'rho_water',11
```

```
      'rho_asfalt',14
      'z_ogb',17};
case 'blokken'
   controls={'dichtheid_hoog',2
      'blokdikte_hoog',5};
case 'gras'
   controls={'kwaliteit',2};
case 'steenzetting'
   controls={'toplaagdikte',2
      'filterkleidikte',5
      'D15',8
      'relatievedichtheid',11
      'bekledingstype',14
      'toetsgrens',16
      'constructie',18};
end
```

Overige routines…

A3 Gewijzigde routines t.b.v. de additionele faalmechanismen

In deze bijlage wordt de code getoond van de gewijzigde routines t.b.v. de nieuwe faalmechanismen. Zie bijlage A1 voor de context waarin deze routines gebruikt worden. In de onderstaande tekst zijn alleen de gewijzigde regels (met enkele voorafgaande en daarop volgende regels) weergegeven tenzij de gehele nieuwe code ongeveer even lang is in welk geval de nieuwe code wordt getoond. Bij het weergeven van de verschillen is de volgende codering gebruikt: + toegevoegde regels, - verwijderde regels, ! gewijzigde regels.

Batch/BatchLeesInvoerBestand

```
***************
*** 23,34 ****

  velden = struct(                ...
     'naam',{{'REGIO','KAPPA', 'FAALMECHANISME', 'OVERSLAGDEBIET', 'HERHALINGSTIJD'
, ...
!          'PROFIEL', 'AFHANKELIJKHEID', 'STROMINGSCORRECTIE', 'VASTE STILWATERLIJN',
'UITVOERBESTAND'}}, ...
     'mnemonic', {{'regio', 'kappa', 'faalm', 'qoverslag', 'herhtijd', ...
!          'profiel', 'afhank', 'stroomcor', 'stilwater', 'uitvoer'}});
  invoer = struct( ...
     'regio', '', 'kappa', '', 'faalm', '', 'qoverslag', '', 'herhtijd', '',...
!          'profiel', '', 'afhank', '','stroomcor', '', 'stilwater', '', 'uitvoer',
'');
  Batch = struct('velden',velden,'invoer',invoer );

  % de default waarden zetten
--- 23,34 ----

  velden = struct(                ...
     'naam',{{'REGIO','KAPPA', 'FAALMECHANISME', 'OVERSLAGDEBIET', 'HERHALINGSTIJD',
...
!          'PROFIEL', 'AFHANKELIJKHEID', 'STROMINGSCORRECTIE', 'VASTE STILWATERLIJN',
'UITVOERBESTAND', 'DIJKOPBOUWBESTAND'}}, ...
     'mnemonic', {{'regio', 'kappa', 'faalm', 'qoverslag', 'herhtijd', ...
!          'profiel', 'afhank', 'stroomcor', 'stilwater', 'uitvoer',
'dijkopbouwbestand'}});
  invoer = struct( ...
     'regio', '', 'kappa', '', 'faalm', '', 'qoverslag', '', 'herhtijd', '',...
!          'profiel', '', 'afhank', '','stroomcor', '', 'stilwater', '', 'uitvoer',
'', 'dijkopbouwbestand', '');
  Batch = struct('velden',velden,'invoer',invoer );

  % de default waarden zetten
***************
*** 36,41 ****
--- 36,42 ----
  Batch.invoer.qoverslag = '10';
  Batch.invoer.stroomcor = 'WEL';
  Batch.invoer.stilwater = 'NIET';
+ Batch.invoer.dijkopbouwbestand = 'NIET GEBRUIKT';

  LF = char(10); % linefeed
  CR = char(13); % carriage-return
***************
*** 111,117 ****
--- 112,127 ----
     end
  elseif ~isempty([findstr(upper(Batch.invoer.faalm),'BL')]),
     Batch.invoer.faalm = FMBLOKKENINSTAB;
+ elseif ~isempty([findstr(upper(Batch.invoer.faalm),'BEKL')]),
+    Batch.invoer.faalm = FMINSTABBEKLED;
+    if strcmp(Batch.invoer.dijkopbouwbestand,'NIET GEBRUIKT'),
+       error ('Ontbrekend dijkopbouw bestand');
+    elseif ~exist(Batch.invoer.dijkopbouwbestand)
+       error (sprintf('Kan dijkopbouw bestand %s niet
lezen',Batch.invoer.dijkopbouwbestand));
```

```
     else
+       Batch.invoer.dijkopbouw = LeesBekleding(Batch.invoer.dijkopbouwbestand);
+    end
+ else
     error(['Onbekend faalmechanisme:' Batch.invoer.faalm]);
  end

***************
*** 173,175 ****
--- 183,188 ----
     error(['Onbekende stilwaterlijnkeuze:' Batch.invoer.stilwater]);
  end

+ if (Batch.invoer.afhank == AFVOLLEDIG | Batch.invoer.belver == SWHOOGSTEWS) &
StormDuurAfhankelijk(Batch.invoer)
+    error('Bekleding met stormduurafhankelijke belasting, selecteer AFHANKELIJKHEID
= WAARnemingen, ONGUnstigste belasting.')
+ end
```

Batch/BatchRekenen

```
***************
*** 7,15 ****
  %     BatchInvoer:   structure met de informatie, gelezen
  %                    uit het batch invoer-bestand

- %  LOKALE FUNCTIES
- %     printProfiel
-
  %  EXTERNE FUNCTIES
  %     StatLaadStatistiek
  %     HaalProfielen
--- 7,12 ----
***************
*** 18,23 ****
--- 15,21 ----
  %     Uitvoer
  %     StatBerekenKruinhoogte
  %     StatBerekenOntwerppunt
+ %     printProfiel

  % Copyright © 2000 Rijkswaterstaat/RIKZ All Rights Reserved.

***************
*** 96,101 ****
--- 94,111 ----
     Uitvoer('p','\n%s\n\n','Oude golfformules worden gebruikt')
  end

+ % Rapporteer de stormflanken-optie
+ if stormFlanken
+    Uitvoer('p','\n%s %i %s\n','Stormflanken toegevoegd van',stormFlanken,'uur')
+ end
+
+ % Rapporteer de piekperiode-optie
+ if PiekPeriode
+    Uitvoer('p','\n%s\n','Piekperiode meegenomen als extra stochast')
+ else
+    Uitvoer('p','\n%s\n','Piekperiode niet meegenomen als stochast (100% afhankelijk
van windsnelheid en -richting)')
+ end
+
  switch BatchInvoer.faalm

  case FMGOLFOPLOOP, % golfoploop
***************
*** 111,116 ****
--- 121,135 ----
     faalMechanisme.naam = 'blokkeninstabiliteit';
     Uitvoer('p','\n%s%s%s\n','Faalmechanisme: blokkeninstabiliteit');

+ case FMINSTABBEKLED
+    faalMechanisme.naam = 'instabiliteit van de bekleding';
```

```
+    faalMechanisme.dijkopbouw = BatchInvoer.dijkopbouw;
+    %
+    % Profiel toegevoegd om controle op ligging van de bekleding te controleren.
+    %
+    faalMechanisme.profiel=profiel;
+    Uitvoer('p','\n%s%s%s\n','Faalmechanisme: instabiliteit van de bekleding, naam =
',BatchInvoer.dijkopbouwbestand);
+
  end

  faalMechControle      = faalMechanisme ;
***************
*** 123,128 ****
--- 142,150 ----
  if BatchInvoer.profiel == PRSTANDAARD | BatchInvoer.profiel == PRCOMBINATIE,
     printProfiel (profiel);
  end
+ if BatchInvoer.faalm == FMINSTABBEKLED
+    Fout = printBekleding (faalMechanisme);
+ end

  Uitvoer ('p','\n%6s','loc.');
  Uitvoer ('p','%10s','X');
***************
*** 201,207 ****
        % 3) Bepaal de correctie op de waterstand =  toetspeil - berekende waarde
        correctieRVB.dToetsPeil=correctieRVB.toetspeil-kruinhZonder;

!
        switch BatchInvoer.faalm

        case {FMGOLFOPLOOP,FMGOLFOVERSLAG} % golfoploop
--- 223,229 ----
        % 3) Bepaal de correctie op de waterstand =  toetspeil - berekende waarde
        correctieRVB.dToetsPeil=correctieRVB.toetspeil-kruinhZonder;

!       ontwerp = 1;
        switch BatchInvoer.faalm

        case {FMGOLFOPLOOP,FMGOLFOVERSLAG} % golfoploop
***************
*** 232,237 ****
--- 254,261 ----
              end

           end
+          gefaald = isempty(indexFalen);
+
        case FMBLOKKENINSTAB
           % Berekenen van de bloksterkte
           [ bloksterkte, indexFalen] = ...
***************
*** 247,255 ****
              faalMechanisme.blokkeninstabiliteit.Blok.dichtheid_hoog = bloksterkte;
           end
           kruinhControle=[];
        end

!       if  ~isempty(indexFalen),
           % ontwerppunt berekenen
           [ ontwerppuntOffshore, ontwerppuntNearshore] = ...
              StatBerekenOntwerppunt (faalMechanisme, {BatchInvoer.regio}, lokatie,
correctieRVB, indexFalen, waarnOverschrFreq, ...
--- 271,322 ----
              faalMechanisme.blokkeninstabiliteit.Blok.dichtheid_hoog = bloksterkte;
           end
           kruinhControle=[];
+          gefaald = isempty(indexFalen);
+       case FMINSTABBEKLED
+          % Berekenen van de bloksterkte
+          [ paramWaarde, indexFalen] = ...
+             StatBerekenBekleding ({BatchInvoer.regio}, lokatie, correctieRVB,
waarnOverschrFreq, waarneming, ...
+             waarnStormSectorIndex, waarnStormIndex, stat, profiel, ...
```

```
+             herhalingstijd, kappa, BatchInvoer.belver,
BatchInvoer.stroomcor,BatchInvoer.faalm,BatchInvoer.dijkopbouw);
+
+          gefaald = ~isfinite(paramWaarde) & paramWaarde > 0;
+          if  ~gefaald & ~isfinite(paramWaarde) % -Inf
+             string2 = strcat(string2,' geen of onvoldoende faalwaarnemingen
gevonden');
+             ontwerp = 0;
+          elseif ~gefaald
+             if ischar(paramWaarde)
+                string2 = strcat (string2,sprintf('%10s',paramWaarde));
+             else
+                string2 = strcat (string2,sprintf('%10.2f',paramWaarde));
+             end
+             switch faalMechanisme.dijkopbouw.bekleding
+             case 'asfalt'
+                parameter = 'laagdikte';
+             case 'blokken'
+                parameter = 'dichtheid_hoog';
+                faalMechanisme.dijkopbouw.bekledingsgegevens.blokdikte_laag = 1;
+                faalMechanisme.dijkopbouw.bekledingsgegevens.blokdikte_hoog = 1;
+             case 'gras'
+                parameter = 'kwaliteit';
+                ontwerp = 0;
+             case 'steenzetting'
+                parameter = 'relatievedichtheid';
+                faalMechanisme.dijkopbouw.bekledingsgegevens.toplaagdikte = 1;
+             end
+             faalMechanisme.dijkopbouw.bekledingsgegevens = ...
+
setfield(faalMechanisme.dijkopbouw.bekledingsgegevens,parameter,paramWaarde);
+             if strcmp(parameter,'dichtheid_hoog')
+                faalMechanisme.dijkopbouw.bekledingsgegevens.dichtheid_laag =
paramWaarde;
+             end
+          end
+          kruinhControle=[];
        end
+       if ontwerp & stormFlanken
+          ontwerp=0;
+          string2 = strcat(string2,' geen ontwerp i.v.m. gebruik stormflanken.');
+       end
!       if  ~gefaald & ontwerp
          % ontwerppunt berekenen
          [ ontwerppuntOffshore, ontwerppuntNearshore] = ...
             StatBerekenOntwerppunt (faalMechanisme, {BatchInvoer.regio}, lokatie,
correctieRVB, indexFalen, waarnOverschrFreq, ...
***************
*** 272,277 ****
--- 339,348 ----
          end

          Uitvoer('p','%s\n', string2);
+       elseif ~gefaald
+          Uitvoer('p','%s\n', string2);
+       else
+          Uitvoer('p','%s%s\n', string2, ' geen adequate bekleding gevonden, faalt
altijd');
        end
     end
  end
```

General/GlobalConstants

```
***************
*** 15,21 ****
  global ...        % FAALMECHANISMEN
     FMGOLFOPLOOP...      % golfoploop
     FMGOLFOVERSLAG...    % glofoverslag
!    FMBLOKKENINSTAB      % blokkeninstabiliteit

  global ...        % PROFIELKEUZE
     PRLOCAAL...          % lokaal dijkprofiel
```

```
--- 15,22 ----
  global ...        % FAALMECHANISMEN
     FMGOLFOPLOOP...      % golfoploop
     FMGOLFOVERSLAG...    % glofoverslag
!    FMBLOKKENINSTAB...   % blokkeninstabiliteit
!    FMINSTABBEKLED       % instabiliteit van de bekleding

  global ...        % PROFIELKEUZE
     PRLOCAAL...          % lokaal dijkprofiel
```

General/InitialiseerGlobalConstants

```
*** 18,23 ****
--- 18,24 ----
  FMGOLFOPLOOP      = 1;  % golfoploop
  FMGOLFOVERSLAG    = 2;  % glofoverslag
  FMBLOKKENINSTAB   = 3;  % blokkeninstabiliteit
+ FMINSTABBEKLED    = 4;  % instabiliteit van de bekleding

  PCODEFAULT        = 1;  % default oploop/overslag module
  PCODWW            = 2;  % DWW oploop/overslag module
```

General/PrintWaarden

```
***************
*** 49,54 ****
--- 49,55 ----
  Uitvoer('p','%25sX=%s, Y=%s\n','Locatie: ',num2str(Waarden.locatie.x),
num2str(Waarden.locatie.y))

  printProfiel (Waarden.profiel);
+ printBekleding (Waarden);

  Uitvoer('p','\n');
  Uitvoer('p','%50s%s\n','Aantal verschoven waarnemingen in het faalgebied:
',num2str(Waarden.kappa))
***************
*** 60,65 ****
--- 61,78 ----
     Uitvoer('p','\n%s\n','Oude golfformules worden gebruikt')
  end

+ % Rapporteer de stormflanken-optie
+ if stormFlanken
+    Uitvoer('p','\n%s %i %s\n','Stormflanken toegevoegd van',stormFlanken,'uur')
+ end
+
+ % Rapporteer de piekperiode-optie
+ if PiekPeriode
+    Uitvoer('p','\n%s\n','Piekperiode meegenomen als extra stochast')
+ else
+    Uitvoer('p','\n%s\n','Piekperiode niet meegenomen als stochast (100% afhankelijk
van windsnelheid en -richting)')
+ end
+
  Uitvoer('p','\n');

  switch Waarden.faalmechanisme
***************
*** 69,74 ****
--- 82,89 ----
     Uitvoer('p','%50s%s%s%s\n','Faalmechanisme: ','Golfoverslag, overslagdebiet:
',num2str(Waarden.overslagdebiet),' l/s/m');
  case FMBLOKKENINSTAB
     Uitvoer('p','%50s%s\n','Faalmechanisme: ','Blokkeninstabiliteit');
+ case FMINSTABBEKLED
+    Uitvoer('p','%50s%s\n','Faalmechanisme: ','Instabiliteit van de bekleding');
  end

  Uitvoer('p','\n');
```

Stat/StatBerekenBloksterkte

```
***************
*** 61,67 ****
        WaarnStormSectorIndex,Stat,Regio) Waarneming];

  % transformatie van offshore naar nearshore
! nearshoreWaarden = Offshore2Nearshore ( Regio,Locatie.x,Locatie.y, correctieRVB,
offshoreWaarden,StroomCor,faalMechID);

  faalMechanisme.naam = 'blokkeninstabiliteit';
  faalMechanisme.blokkeninstabiliteit.Blok.blokdikte_laag =1;
--- 61,68 ----
        WaarnStormSectorIndex,Stat,Regio) Waarneming];

  % transformatie van offshore naar nearshore
! [nearshoreWaarden,WaarnStormIndex] = Offshore2Nearshore ( ...
!    Regio, Locatie.x,Locatie.y,
correctieRVB,offshoreWaarden,StroomCor,faalMechID,WaarnStormIndex);

  faalMechanisme.naam = 'blokkeninstabiliteit';
  faalMechanisme.blokkeninstabiliteit.Blok.blokdikte_laag =1;
***************
*** 74,80 ****
  faalMechanisme.blokkeninstabiliteit.Blok.dichtheid_laag = ondergrens;
  faalMechanisme.blokkeninstabiliteit.Blok.dichtheid_hoog = ondergrens;

! z = StatBerekenZuitNs (faalMechanisme, Profiel,nearshoreWaarden);
  [aantalZkleinerNul, Jm] = ...
     SelecteerMaxZ ( z, Kappa, WaarnStormIndex, SelectieWaarnemingen);

--- 75,81 ----
  faalMechanisme.blokkeninstabiliteit.Blok.dichtheid_laag = ondergrens;
  faalMechanisme.blokkeninstabiliteit.Blok.dichtheid_hoog = ondergrens;

! z = StatBerekenZuitNs (faalMechanisme, Profiel,nearshoreWaarden, WaarnStormIndex);
  [aantalZkleinerNul, Jm] = ...
     SelecteerMaxZ ( z, Kappa, WaarnStormIndex, SelectieWaarnemingen);

***************
*** 89,95 ****
  faalMechanisme.blokkeninstabiliteit.Blok.dichtheid_laag = bovengrens;
  faalMechanisme.blokkeninstabiliteit.Blok.dichtheid_hoog = bovengrens;

! z = StatBerekenZuitNs (faalMechanisme, Profiel,nearshoreWaarden);
  [aantalZkleinerNul, Jm] = ...
     SelecteerMaxZ ( z, Kappa, WaarnStormIndex, SelectieWaarnemingen);

--- 90,96 ----
  faalMechanisme.blokkeninstabiliteit.Blok.dichtheid_laag = bovengrens;
  faalMechanisme.blokkeninstabiliteit.Blok.dichtheid_hoog = bovengrens;

! z = StatBerekenZuitNs (faalMechanisme, Profiel,nearshoreWaarden, WaarnStormIndex);
  [aantalZkleinerNul, Jm] = ...
     SelecteerMaxZ ( z, Kappa, WaarnStormIndex, SelectieWaarnemingen);

***************
*** 109,115 ****
     faalMechanisme.blokkeninstabiliteit.Blok.dichtheid_laag = ondergrens+stap;
     faalMechanisme.blokkeninstabiliteit.Blok.dichtheid_hoog = ondergrens+stap;

!    z = StatBerekenZuitNs (faalMechanisme, Profiel,nearshoreWaarden);
     [aantalZkleinerNul, Jm] = ...
        SelecteerMaxZ ( z, Kappa, WaarnStormIndex, SelectieWaarnemingen);
     if aantalZkleinerNul > Kappa,
--- 110,116 ----
     faalMechanisme.blokkeninstabiliteit.Blok.dichtheid_laag = ondergrens+stap;
     faalMechanisme.blokkeninstabiliteit.Blok.dichtheid_hoog = ondergrens+stap;

!    z = StatBerekenZuitNs (faalMechanisme, Profiel,nearshoreWaarden,
WaarnStormIndex);
     [aantalZkleinerNul, Jm] = ...
        SelecteerMaxZ ( z, Kappa, WaarnStormIndex, SelectieWaarnemingen);
     if aantalZkleinerNul > Kappa,
```

```
***************
*** 129,135 ****

     Bloksterkte = fminbnd('StatKruinhBloksMinimizeFunc', ondergrens, bovengrens, ...
        optimset('TolX',1e-4,'Display','off'),...
!       faalMechanisme, Profiel, nearshoreWaarden(Jm,:));

     faalMechanisme.blokkeninstabiliteit.Blok.dichtheid_laag = Bloksterkte;
     faalMechanisme.blokkeninstabiliteit.Blok.dichtheid_hoog = Bloksterkte;
--- 130,136 ----

     Bloksterkte = fminbnd('StatKruinhBloksMinimizeFunc', ondergrens, bovengrens, ...
        optimset('TolX',1e-4,'Display','off'),...
!       faalMechanisme, Profiel, nearshoreWaarden(Jm,:), WaarnStormIndex);

     faalMechanisme.blokkeninstabiliteit.Blok.dichtheid_laag = Bloksterkte;
     faalMechanisme.blokkeninstabiliteit.Blok.dichtheid_hoog = Bloksterkte;
***************
*** 140,146 ****
        break;
     end

!    z = StatBerekenZuitNs (faalMechanisme, Profiel,nearshoreWaarden);
     [aantalZkleinerNul, Jm, zOorsprIndex] = ...
        SelecteerMaxZ ( z, Kappa, WaarnStormIndex, SelectieWaarnemingen);

--- 141,147 ----
        break;
     end

!    z = StatBerekenZuitNs (faalMechanisme, Profiel,nearshoreWaarden,
WaarnStormIndex);
     [aantalZkleinerNul, Jm, zOorsprIndex] = ...
        SelecteerMaxZ ( z, Kappa, WaarnStormIndex, SelectieWaarnemingen);
```

Stat/StatBerekenFaalfrequentie

```
***************
*** 58,64 ****
  bovengrens = -log(FreqBound);

  z = StatBerekenZ (FaalMechanisme, Regio, Locatie, correctieRVB,
Profiel,WaarnOverschrFactor,WaarnStormSectorIndex,...
!    Waarneming,Stat,StroomCor,faalMechID);

  [aantalZkleinerNul, Jm] = ...
     SelecteerMaxZ ( z, Kappa, WaarnStormIndex, SelectieWaarnemingen);
--- 58,64 ----
  bovengrens = -log(FreqBound);

  z = StatBerekenZ (FaalMechanisme, Regio, Locatie, correctieRVB,
Profiel,WaarnOverschrFactor,WaarnStormSectorIndex,...
!    Waarneming,WaarnStormIndex,Stat,StroomCor,faalMechID);

  [aantalZkleinerNul, Jm] = ...
     SelecteerMaxZ ( z, Kappa, WaarnStormIndex, SelectieWaarnemingen);
***************
*** 72,78 ****

  % kijk even of we niet over de bovengrens heen gaan
  z=StatBerekenZ(FaalMechanisme, Regio, Locatie, correctieRVB,
Profiel,WaarnOverschrFactor+bovengrens,...
!    WaarnStormSectorIndex,Waarneming,Stat,StroomCor,faalMechID);
  [aantalZkleinerNul, Jm] = ...
     SelecteerMaxZ ( z, Kappa, WaarnStormIndex, SelectieWaarnemingen);

--- 72,78 ----

  % kijk even of we niet over de bovengrens heen gaan
  z=StatBerekenZ(FaalMechanisme, Regio, Locatie, correctieRVB,
Profiel,WaarnOverschrFactor+bovengrens,...
!    WaarnStormSectorIndex,Waarneming,WaarnStormIndex,Stat,StroomCor,faalMechID);
```

```
  [aantalZkleinerNul, Jm] = ...
     SelecteerMaxZ ( z, Kappa, WaarnStormIndex, SelectieWaarnemingen);

***************
*** 87,93 ****
  stap = 4;
  while 1,
     z=StatBerekenZ(FaalMechanisme, Regio, Locatie, correctieRVB,
Profiel,WaarnOverschrFactor+ondergrens+stap,...
!       WaarnStormSectorIndex,Waarneming,Stat,StroomCor,faalMechID);
     [aantalZkleinerNul, Jm] = ...
        SelecteerMaxZ ( z, Kappa, WaarnStormIndex, SelectieWaarnemingen);

--- 87,93 ----
  stap = 4;
  while 1,
     z=StatBerekenZ(FaalMechanisme, Regio, Locatie, correctieRVB,
Profiel,WaarnOverschrFactor+ondergrens+stap,...
!       WaarnStormSectorIndex,Waarneming,WaarnStormIndex,Stat,StroomCor,faalMechID);
     [aantalZkleinerNul, Jm] = ...
        SelecteerMaxZ ( z, Kappa, WaarnStormIndex, SelectieWaarnemingen);

***************
*** 94,99 ****
--- 94,100 ----
     if aantalZkleinerNul <= 0,
        ondergrens = ondergrens+stap;
     elseif aantalZkleinerNul > Kappa,
+       bovengrens = ondergrens+stap;
        stap = stap/4;
     else
        break
***************
*** 105,120 ****
  % Nu gaan we minimaliseren tussen de boven- en ondergrens

  lambda = ondergrens ;
  while bovengrens > ondergrens

-    lambda=fminbnd('StatFaalFreqMinimizeFunc',ondergrens,bovengrens...
-       ,optimset('TolX',1e-4,'Display','off')...
-       ,FaalMechanisme, Regio, Locatie, correctieRVB, WaarnOverschrFactor(Jm,:)...
-
,Waarneming(Jm,:),WaarnStormSectorIndex(Jm,:),Profiel,Stat,StroomCor,faalMechID);
-
-
     z=StatBerekenZ(FaalMechanisme, Regio, Locatie, correctieRVB,
Profiel,WaarnOverschrFactor+lambda,WaarnStormSectorIndex, ...
!          Waarneming,Stat,StroomCor,faalMechID);
          [aantalZkleinerNul, Jm, zOorsprIndex] = ...
          SelecteerMaxZ ( z, Kappa, WaarnStormIndex, SelectieWaarnemingen);

--- 106,137 ----
  % Nu gaan we minimaliseren tussen de boven- en ondergrens

  lambda = ondergrens ;
+ stormduurafh = StormDuurAfhankelijk(FaalMechanisme);
  while bovengrens > ondergrens
+    if stormduurafh
+       %
+       % In het geval van gras moet de hele storm worden doorgegeven ...
+       %
+       heleStorm=WaarnStormIndex==WaarnStormIndex(Jm);
+       lambda=fminbnd('StatFaalFreqMinimizeFunc',ondergrens,bovengrens, ...
+          optimset('TolX',1e-4,'Display','off'), ...
+          FaalMechanisme, Regio, Locatie, correctieRVB,
WaarnOverschrFactor(heleStorm,:), ...
+
Waarneming(heleStorm,:),WaarnStormSectorIndex(heleStorm,:),WaarnStormIndex(heleStorm)
, ...
+          Profiel,Stat,StroomCor,faalMechID);
+    else
+       lambda=fminbnd('StatFaalFreqMinimizeFunc',ondergrens,bovengrens, ...
+          optimset('TolX',1e-4,'Display','off'), ...
```

```
+          FaalMechanisme, Regio, Locatie, correctieRVB, WaarnOverschrFactor(Jm,:),
...
+          Waarneming(Jm,:),WaarnStormSectorIndex(Jm,:),[], ...
+          Profiel,Stat,StroomCor,faalMechID);
+    end
+    %
+    % Alternatief: zoek door het interval telkens in tweeen te splitsen:
+    % lambda = (ondergrens+bovengrens)/2;
+    %

     z=StatBerekenZ(FaalMechanisme, Regio, Locatie, correctieRVB,
Profiel,WaarnOverschrFactor+lambda,WaarnStormSectorIndex, ...
!       Waarneming,WaarnStormIndex,Stat,StroomCor,faalMechID);
     [aantalZkleinerNul, Jm, zOorsprIndex] = ...
        SelecteerMaxZ ( z, Kappa, WaarnStormIndex, SelectieWaarnemingen);

***************
*** 122,127 ****
--- 139,146 ----
        if (abs(bovengrens-ondergrens) < 0.0001),
           % staat en beetje om Kappa heen te springen door dubbele waarden in z
           break;
+       elseif (abs(bovengrens-lambda) < 0.0001),
+          bovengrens = bovengrens-0.01;
        else
           bovengrens= lambda;
        end
```

Stat/StatBerekenGolfoverslag

```
***************
*** 41,47 ****
        WaarnStormSectorIndex,Stat, Regio) Waarneming];

  % transformatie van offshore naar nearshore
! nearshoreWaarden = Offshore2Nearshore ( Regio,Locatie.x,Locatie.y, correctieRVB,
offshoreWaarden,StroomCor,faalMechID);

  ondergrens = 0.001;
  bovengrens = 1000;
--- 41,48 ----
        WaarnStormSectorIndex,Stat, Regio) Waarneming];

  % transformatie van offshore naar nearshore
! [nearshoreWaarden,WaarnStormIndex] = Offshore2Nearshore ( ...
!    Regio, Locatie.x,Locatie.y,
correctieRVB,offshoreWaarden,StroomCor,faalMechID,WaarnStormIndex);

  ondergrens = 0.001;
  bovengrens = 1000;
***************
*** 50,56 ****
  faalMechanisme.naam = 'golfoverslag';
  faalMechanisme.golfoverslag.Qc = ondergrens;

! z = StatBerekenZuitNs (faalMechanisme, Profiel,nearshoreWaarden);
  [aantalZkleinerNul, Jm] = ...
     SelecteerMaxZ ( z, Kappa, WaarnStormIndex, Selectiewaarnemingen);

--- 51,57 ----
  faalMechanisme.naam = 'golfoverslag';
  faalMechanisme.golfoverslag.Qc = ondergrens;

! z = StatBerekenZuitNs (faalMechanisme, Profiel,nearshoreWaarden, WaarnStormIndex);
  [aantalZkleinerNul, Jm] = ...
     SelecteerMaxZ ( z, Kappa, WaarnStormIndex, Selectiewaarnemingen);

***************
*** 64,70 ****
  % kijk eerst of de bovengrens wel voldoende is
  faalMechanisme.golfoverslag.Qc = bovengrens;

! z = StatBerekenZuitNs (faalMechanisme, Profiel,nearshoreWaarden);
```

```
  [aantalZkleinerNul, Jm] = ...
     SelecteerMaxZ ( z, Kappa, WaarnStormIndex, Selectiewaarnemingen);

--- 65,71 ----
  % kijk eerst of de bovengrens wel voldoende is
  faalMechanisme.golfoverslag.Qc = bovengrens;

! z = StatBerekenZuitNs (faalMechanisme, Profiel,nearshoreWaarden, WaarnStormIndex);
  [aantalZkleinerNul, Jm] = ...
     SelecteerMaxZ ( z, Kappa, WaarnStormIndex, Selectiewaarnemingen);

***************
*** 82,88 ****
  while 1
     faalMechanisme.golfoverslag.Qc = ondergrens+stap;

!    z = StatBerekenZuitNs (faalMechanisme, Profiel,nearshoreWaarden);
     [aantalZkleinerNul, Jm] = ...
        SelecteerMaxZ ( z, Kappa, WaarnStormIndex, Selectiewaarnemingen);
     if aantalZkleinerNul > Kappa,
--- 83,89 ----
  while 1
     faalMechanisme.golfoverslag.Qc = ondergrens+stap;

!    z = StatBerekenZuitNs (faalMechanisme, Profiel,nearshoreWaarden,
WaarnStormIndex);
     [aantalZkleinerNul, Jm] = ...
        SelecteerMaxZ ( z, Kappa, WaarnStormIndex, Selectiewaarnemingen);
     if aantalZkleinerNul > Kappa,
***************
*** 104,110 ****

    faalMechanisme.golfoverslag.Qc = Q;

!   z = StatBerekenZuitNs (faalMechanisme, Profiel,nearshoreWaarden);
     [aantalZkleinerNul, Jm, zOorsprIndex] = ...
        SelecteerMaxZ ( z, Kappa, WaarnStormIndex, Selectiewaarnemingen);

--- 105,111 ----

    faalMechanisme.golfoverslag.Qc = Q;

!   z = StatBerekenZuitNs (faalMechanisme, Profiel,nearshoreWaarden,
WaarnStormIndex);
     [aantalZkleinerNul, Jm, zOorsprIndex] = ...
        SelecteerMaxZ ( z, Kappa, WaarnStormIndex, Selectiewaarnemingen);
```

Stat/StatBerekenKruinhoogte

```
***************
*** 65,71 ****
        WaarnStormSectorIndex,Stat,Regio) Waarneming];

  % transformatie van offshore naar nearshore
! nearshoreWaarden = Offshore2Nearshore ( Regio, Locatie.x,Locatie.y,
correctieRVB,offshoreWaarden,StroomCor,faalMechID);

  % vaststellen onder- en bovengrens kruinhoogte
  ondergrens = min([Profiel.hoogte_teen]);
--- 65,72 ----
        WaarnStormSectorIndex,Stat,Regio) Waarneming];

  % transformatie van offshore naar nearshore
! [nearshoreWaarden,WaarnStormIndex] = Offshore2Nearshore ( ...
!    Regio, Locatie.x,Locatie.y,
correctieRVB,offshoreWaarden,StroomCor,faalMechID,WaarnStormIndex);

  % vaststellen onder- en bovengrens kruinhoogte
  ondergrens = min([Profiel.hoogte_teen]);
***************
*** 78,84 ****
     FaalMechanisme.golfoploop.Hc(i) = ondergrens;
  end
```

```
! z = StatBerekenZuitNs (FaalMechanisme, Profiel,nearshoreWaarden);
  [aantalZkleinerNul, Jm] = ...
     SelecteerMaxZ ( z, Kappa, WaarnStormIndex, SelectieWaarnemingen);

--- 79,85 ----
     FaalMechanisme.golfoploop.Hc(i) = ondergrens;
  end

! z = StatBerekenZuitNs (FaalMechanisme, Profiel,nearshoreWaarden, WaarnStormIndex);
  [aantalZkleinerNul, Jm] = ...
     SelecteerMaxZ ( z, Kappa, WaarnStormIndex, SelectieWaarnemingen);

***************
*** 92,98 ****
     FaalMechanisme.golfoploop.Hc(i) = bovengrens;
  end

! z = StatBerekenZuitNs (FaalMechanisme, Profiel,nearshoreWaarden);
  [aantalZkleinerNul, Jm] = ...
     SelecteerMaxZ ( z, Kappa, WaarnStormIndex, SelectieWaarnemingen);

--- 93,99 ----
     FaalMechanisme.golfoploop.Hc(i) = bovengrens;
  end

! z = StatBerekenZuitNs (FaalMechanisme, Profiel,nearshoreWaarden, WaarnStormIndex);
  [aantalZkleinerNul, Jm] = ...
     SelecteerMaxZ ( z, Kappa, WaarnStormIndex, SelectieWaarnemingen);

***************
*** 110,121 ****
        FaalMechanisme.golfoploop.Hc(i) = ondergrens+stap;
     end

!    z = StatBerekenZuitNs (FaalMechanisme, Profiel,nearshoreWaarden);
     [aantalZkleinerNul, Jm] = ...
        SelecteerMaxZ ( z, Kappa, WaarnStormIndex, SelectieWaarnemingen);
     if aantalZkleinerNul > Kappa,
        ondergrens = ondergrens+stap;
-
     elseif aantalZkleinerNul == 0,
        stap = stap/4;
     else,
--- 111,121 ----
        FaalMechanisme.golfoploop.Hc(i) = ondergrens+stap;
     end

!    z = StatBerekenZuitNs (FaalMechanisme, Profiel,nearshoreWaarden,
WaarnStormIndex);
     [aantalZkleinerNul, Jm] = ...
        SelecteerMaxZ ( z, Kappa, WaarnStormIndex, SelectieWaarnemingen);
     if aantalZkleinerNul > Kappa,
        ondergrens = ondergrens+stap;
     elseif aantalZkleinerNul == 0,
        stap = stap/4;
     else,
***************
*** 135,141 ****

     KruinHoogte = fminbnd('StatKruinhBloksMinimizeFunc', ondergrens, bovengrens, ...
        optimset('TolX',1e-4,'Display','off'),...
!       FaalMechanisme, Profiel, nearshoreWaarden(Jm,:));

     for i = 1:length(Profiel)
        Profiel(i).hoogte_kruin = KruinHoogte;
--- 135,141 ----

     KruinHoogte = fminbnd('StatKruinhBloksMinimizeFunc', ondergrens, bovengrens, ...
        optimset('TolX',1e-4,'Display','off'),...
!       FaalMechanisme, Profiel, nearshoreWaarden(Jm,:),WaarnStormIndex);

     for i = 1:length(Profiel)
        Profiel(i).hoogte_kruin = KruinHoogte;
***************
```

```
*** 142,148 ****
        FaalMechanisme.golfoploop.Hc(i) = KruinHoogte;
     end

!    z = StatBerekenZuitNs (FaalMechanisme, Profiel,nearshoreWaarden);

     [aantalZkleinerNul, Jm, zOorsprIndex] = ...
        SelecteerMaxZ ( z, Kappa, WaarnStormIndex, SelectieWaarnemingen);
--- 142,148 ----
        FaalMechanisme.golfoploop.Hc(i) = KruinHoogte;
     end

!    z = StatBerekenZuitNs (FaalMechanisme, Profiel,nearshoreWaarden,
WaarnStormIndex);

     [aantalZkleinerNul, Jm, zOorsprIndex] = ...
        SelecteerMaxZ ( z, Kappa, WaarnStormIndex, SelectieWaarnemingen);
```

Stat/StatBerekenOntwerppunt

```
***************
*** 158,166 ****
     else
        % transformatie van offshore naar nearshore
        nearshoreWaarden = Offshore2Nearshore ( Regio,Locatie.x,Locatie.y,...
!          tmpoffshoreWaarden,StroomCor,faalMechID);

!       c = StatBerekenZuitNs (FaalMechanisme, Profiel, nearshoreWaarden );

        % Bepaal begin en eind index van elke storm
                  %n= length(WaarnStormIndex);
--- 158,166 ----
     else
        % transformatie van offshore naar nearshore
        nearshoreWaarden = Offshore2Nearshore ( Regio,Locatie.x,Locatie.y,...
!          correctieRVB, tmpoffshoreWaarden,StroomCor,faalMechID);

!       c = StatBerekenZuitNs (FaalMechanisme, Profiel, nearshoreWaarden,
WaarnStormIndex);

        % Bepaal begin en eind index van elke storm
                  %n= length(WaarnStormIndex);
***************
*** 179,188 ****
  offshoreWaarden= [StatWeibullInv(trialf,ones(lt,1)*designc,Stat,Regio)
ones(lt,1)*designa];

  % transformatie van offshore naar nearshore
! nearshoreWaarden = Offshore2Nearshore ( Regio,Locatie.x,Locatie.y, correctieRVB,
offshoreWaarden, StroomCor,faalMechID);

  % alle bijbehorende kruinhoogtes worden uitgerekend
! z = StatBerekenZuitNs (FaalMechanisme, Profiel, nearshoreWaarden );

  % in z0 is de kruinhoogte die zo dicht mogelijk bij Z=0 ligt
  [zo,jo]= sort(abs(z));
--- 179,189 ----
  offshoreWaarden= [StatWeibullInv(trialf,ones(lt,1)*designc,Stat,Regio)
ones(lt,1)*designa];

  % transformatie van offshore naar nearshore
! nearshoreWaarden = Offshore2Nearshore ( ...
!     Regio,Locatie.x,Locatie.y, correctieRVB, offshoreWaarden,
StroomCor,faalMechID);

  % alle bijbehorende kruinhoogtes worden uitgerekend
! z = StatBerekenZuitNs (FaalMechanisme, Profiel, nearshoreWaarden, WaarnStormIndex);

  % in z0 is de kruinhoogte die zo dicht mogelijk bij Z=0 ligt
  [zo,jo]= sort(abs(z));
```

Stat/StatBerekenZ

```
function Z = StatBerekenZ (FaalMechanisme,  Regio, Locatie, correctieRVB, ...
   Profiel, WaarnOverschrFactor, WaarnStormSectorIndex,Waarneming,WaarnStormIndex,
...
   Stat,StroomCor,faalMechID)

%STATBEREKENZ     In StatBerekenZ wordt de Z-functie uitgerekend voor een gegeven
%                 faalmechanisme. Deze functie is beschreven in 3.1.1 van het
%                 functioneel ontwerp
%                 Merk op, dat de case "waterstand" bedoeld is om de
%                 controlefunctie te kunnen bepalen in verband met
%                 het Randvoorwaardenboek.
%  AANROEP
%     Z = StatBerekenZ (FaalMechanisme, Locatie,correctieRVB, Profiel, ...
%        WaarnOverschrFactor, WaarnStormSectorIndex,Waarneming,WaarnStormIndex,Stat)
%  INVOER
%     FaalMechanisme:         het faalmechanisme ('golfoploop', 'golfoverslag' of
%                             'blokkeninstabiliteit')
%     Locatie:                de x,y coordinaten van de locatie
%     correctieRVB:           de correctie op de waterstand voor verschillen met het
Randvoorwaardenboek
%     Profiel:                het dijkprofiel
%     WaarnOverschrFactor:    formule: -
log(waarnemingenoverschrijdingsaantallen)/kappa
%                             (waterstand, windsnelheid, piekperiode, sign.
golfhoogte)
%     WaarnStormSectorIndex:  de sectorindex van de storm (windrichting van de storm)
%     Waarneming:             de golfrichting en de windrichting van de waarnemingen
%     WaarnStormIndex:        de index van de storm behorend bij de waarneming
%     Stat:                   de Weibull parameters voor de waterstand, windsnelheid,
%                             piekperiode en sign. golfhoogte
%  UITVOER
%     Z:                      een vector met Z-waarden

%  EXTERNE FUNCTIES
%     Offshore2Nearshore
%     StatBerekenZGolfoploop
%     StatBerekenZGolfoverslag
%     StatBerekenZBlokinstabiliteit

% Copyright © 2000 Rijkswaterstaat/RIKZ All Rights Reserved.

% Auteur: Abe Hoekstra/Ingrid Lammers, HKV lijn in water

% $Header: /PR/313/HYDRA_K/Stat/StatBerekenZ.m 3     3-11-00 10:37 Hoekstra $
% $NoKeywords: $

GlobalConstants

% Allereerst worden de offshore gegevens bepaald met de functie StatWeibullInv uit de
% overschrijdingsfrequenties van de waarnemingen
% Daarna worden de offshoregegevens getransformeerd naar nearshore gegevens met de
functie Offshore2Nearshore
% Vervolgens wordt afhankelijk van het faalmechanisme de betreffende Z functie
uitgerekend

Var_of= [StatWeibullInv(WaarnOverschrFactor,WaarnStormSectorIndex,Stat,Regio)
Waarneming];

% De offshoregegevens bestaan uit:
% 1: waterstand
% 2: windsnelheid
% 3: piekperiode
% 4: significante golfhoogte
% 5: golfrichting
% 6: windrichting
% 7: noodsluiting (Oosterschelde)


% transformatie van offshore naar nearshore
% De nearshoregegevens bestaan uit:
% 1: waterstand
% 2: windsnelheid
```

```
% 3: piekperiode
% 4: significante golfhoogte
% 5: golfrichting
% 6: windrichting


[Var_ne,WaarnStormIndex] = Offshore2Nearshore ( ...

Regio,Locatie.x,Locatie.y,correctieRVB,Var_of,StroomCor,faalMechID,WaarnStormIndex);

for i = 1:length(Profiel)
   switch FaalMechanisme.naam

   case 'golfoploop',
         [alfa_repr, gamma_b, gamma_f, hoogte_2pr, s_op, zeta, z_2pr] ...
            = StatBerekenZGolfoploop (Profiel(i), Var_ne(:,4), Var_ne(:,3),
Var_ne(:,1), Var_ne(:,5));
            Z(:,i) = FaalMechanisme.golfoploop.Hc(i) - Var_ne(:,1)-z_2pr;

   case 'waterstand',
         Z(:,i) = FaalMechanisme.golfoploop.Hc(i) - Var_ne(:,1);

   case 'golfoverslag',
         Z(:,i) = StatBerekenZGolfoverslag ( Profiel(i),
FaalMechanisme.golfoverslag.Qc, ...
            Var_ne(:,4), Var_ne(:,3), Var_ne(:,1), Var_ne(:,5));

   case 'blokkeninstabiliteit',
      if length(FaalMechanisme.blokkeninstabiliteit.Blok) == 1,
         Z(:,i) = StatBerekenZBlokinstabiliteit (Profiel(i), ...
            FaalMechanisme.blokkeninstabiliteit.Blok, ...
            Var_ne(:,4), Var_ne(:,3), Var_ne(:,1), Var_ne(:,5));
      else
         Z(:,i) = StatBerekenZBlokinstabiliteit (Profiel(i), ...
            FaalMechanisme.blokkeninstabiliteit.Blok(i), ...
            Var_ne(:,4), Var_ne(:,3), Var_ne(:,1), Var_ne(:,5));
      end

   case {'instabiliteit van de bekleding','instabiliteit van de bekleding met gras'}
      AantalLagenDijkbekleding = length(FaalMechanisme.dijkopbouw);
      ZWaardeLaagZi=zeros(size(Var_ne,1),AantalLagenDijkbekleding);
      for zi=1:AantalLagenDijkbekleding
         switch FaalMechanisme.dijkopbouw(zi).bekleding
         case 'asfalt'
            ZWaardeLaagZi(:,zi) = StatBerekenZAsfalt (Profiel(i), ...
               FaalMechanisme.dijkopbouw(zi), ...
               Var_ne(:,4), Var_ne(:,3), Var_ne(:,1), Var_ne(:,5));
         case 'steenzetting'
            ZWaardeLaagZi(:,zi) = StatBerekenZSteenzetting ...
               (Profiel(i), FaalMechanisme.dijkopbouw(zi), ...
               Var_ne(:,4), Var_ne(:,3), Var_ne(:,1), Var_ne(:,5));
         case 'gras'
            ZWaardeLaagZi(:,zi) = StatBerekenZGras (Profiel(i), ...
               FaalMechanisme.dijkopbouw(zi), WaarnStormIndex, ...
               Var_ne(:,4), Var_ne(:,3), Var_ne(:,1), Var_ne(:,5));
         case 'blokken'
            ZWaardeLaagZi(:,zi) = StatBerekenZBlokinstabiliteit ...
               (Profiel(i), FaalMechanisme.dijkopbouw(zi), ...
               Var_ne(:,4), Var_ne(:,3), Var_ne(:,1), Var_ne(:,5));
         end
      end
      %
      % Bepaal minimum Z-waarde over alle lagen ...
      %
      Z(:,i)=min(ZWaardeLaagZi,[],2);
   end
end
%
% Bepaal minimum Z-waarde over alle profielen ...
%
Z = min(Z,[],2);

Z = real(Z);
nanind = find(isnan(Z));
Z(nanind) = -Inf;
```

```
% De nearshoregegevens bestaan uit:
% 1: waterstand
% 2: windsnelheid
% 3: piekperiode
% 4: significante golfhoogte
% 5: golfrichting
% 6: windrichting
```

Stat/StatBerekenZBlokinstabiliteit

```
***************
*** 34,43 ****
--- 34,50 ----
  helling_berm = Profiel.helling_berm;
  ruwheid      = Profiel.ruwheid;

+ if isfield(Blok,'blokdikte_laag')
     blokdikte_laag = Blok.blokdikte_laag;
     blokdikte_hoog = Blok.blokdikte_hoog;
     dichtheid_laag = Blok.dichtheid_laag;
     dichtheid_hoog = Blok.dichtheid_hoog;
+ else
+    blokdikte_laag = Blok.bekledingsgegevens.blokdikte_laag;
+    blokdikte_hoog = Blok.bekledingsgegevens.blokdikte_hoog;
+    dichtheid_laag = Blok.bekledingsgegevens.dichtheid_laag;
+    dichtheid_hoog = Blok.bekledingsgegevens.dichtheid_hoog;
+ end

  g = 9.81;

***************
*** 83,100 ****

  % For matrices, MIN(X) is a row vector containing the minimum element from each
column.

! % Voor de verschillende waterstanden wo

- n = size (sl(:),1);
-
- bloksterkte_hoog = ones(n,1) * (blokdikte_hoog * dichtheid_hoog);
- bloksterkte_laag = ones(n,1) * (blokdikte_laag * dichtheid_laag);
- bloksterkte = [bloksterkte_laag bloksterkte_hoog];
-
- index = find ((sl - H_s < hoogte_berm) | (sl - H_s < hoogte_kruin) | ...
-               (sl + H_s > hoogte_berm) | (sl + H_s > hoogte_teen));
-
- bloksterkte(index) = inf .* ones(size(index));
- bloksterkte = min ([bloksterkte ones(n,1)*inf]')';
  q = blokbelasting;
  Z = bloksterkte - blokbelasting;
--- 90,100 ----

  % For matrices, MIN(X) is a row vector containing the minimum element from each
column.

! %
! % Bert Jagers (WL): code vereenvoudigd
! %
! bloksterkte = min (blokdikte_hoog * dichtheid_hoog, blokdikte_laag *
dichtheid_laag);
! bloksterkte = repmat(bloksterkte,size(sl));

  q = blokbelasting;
  Z = bloksterkte - blokbelasting;
```

Stat/StatBerekenZuitNs

```
function [Z, q] = StatBerekenZuitNs (FaalMechanisme, Profiel, Var_ne,
WaarnStormIndex)
```

```
%STATBEREKENZUITNS   In StatBerekenZuitNs wordt de Z-functie uitgerekend voor een
gegeven
%                    faalmechanisme uitgaande van de nearshorewaarden.
%                    Deze nearshorewaarden hoeven dus niet te worden uitgerekend
zoals in
%                    StatBerekenZ, verder is de functie hetzelfde.
%                    Deze functie is beschreven in 3.1.1 van het functioneel ontwerp
%                    Merk op, dat de case "waterstand" bedoeld is om de
%                    controlefunctie te kunnen bepalen in verband met
%                    het Randvoorwaardenboek.
%  AANROEP
%     [Z,q] = StatBerekenZuitNs (FaalMechanisme, Profiel, Var_ne, WaarnStormIndex)
%  INVOER
%     FaalMechanisme:         het faalmechanisme ('golfoploop', 'golfoverslag' of
%                             'blokkeninstabiliteit')
%     Profiel:                het dijkprofiel
%     Var_ne:                 de nearshorewaarden
%                             De nearshoregegevens bestaan uit:
%                             1: waterstand
%                             2: windsnelheid
%                             3: piekperiode
%                             4: significante golfhoogte
%                             5: golfrichting
%                             6: windrichting
%     WaarnStormIndex:        de index van de storm behorend bij de waarneming
%  UITVOER
%     Z:                      een vector met Z-waarden
%     q:                      een vector met belastingen

%  EXTERNE FUNCTIES
%     StatBerekenZGolfoploop
%     StatBerekenZGolfoverslag
%     StatBerekenZBlokinstabiliteit
%     StatBerekenZAsfalt
%     StatBerekenZGras
%     StatBerekenZSteenzetting

% Copyright © 2000 Rijkswaterstaat/RIKZ All Rights Reserved.

% Auteur: Abe Hoekstra/Ingrid Lammers, HKV lijn in water

% $Header: /PR/313/HYDRA_K/Stat/StatBerekenZuitNs.m 3     3-11-00 10:37 Hoekstra $
% $NoKeywords: $

for i = 1:length(Profiel)
   switch FaalMechanisme.naam

   case 'golfoploop'
      [alfa_repr, gamma_b, gamma_f, q, s_op, zeta, z_2pr] ...
         = StatBerekenZGolfoploop (Profiel(i), Var_ne(:,4), Var_ne(:,3), Var_ne(:,1),
Var_ne(:,5));
      Z(:,i) = FaalMechanisme.golfoploop.Hc(i) - Var_ne(:,1)-z_2pr;

   case 'waterstand'
      Z(:,i) = FaalMechanisme.golfoploop.Hc(i) - Var_ne(:,1);

   case 'golfoverslag'
      [Z(:,i), q(:,i)]  = StatBerekenZGolfoverslag ( Profiel(i),
FaalMechanisme.golfoverslag.Qc, ...
         Var_ne(:,4), Var_ne(:,3), Var_ne(:,1), Var_ne(:,5));

   case 'blokkeninstabiliteit'
      [Z(:,i),q] = StatBerekenZBlokinstabiliteit (Profiel(i), ...
         FaalMechanisme.blokkeninstabiliteit.Blok, ...
         Var_ne(:,4), Var_ne(:,3), Var_ne(:,1), Var_ne(:,5));

   case {'instabiliteit van de bekleding','instabiliteit van de bekleding met gras'}
      AantalWaarn = size(Var_ne,1);
      AantalLagenDijkbekleding = length(FaalMechanisme.dijkopbouw);
      ZWaardeLaagZi = zeros(AantalWaarn,AantalLagenDijkbekleding);
      q             = zeros(AantalWaarn,AantalLagenDijkbekleding);
      for zi=1:AantalLagenDijkbekleding
         switch FaalMechanisme.dijkopbouw(zi).bekleding
         case 'asfalt'
```

```
            [ZWaardeLaagZi(:,zi),q(:,zi)] = StatBerekenZAsfalt ...
               (Profiel(i), FaalMechanisme.dijkopbouw(zi), ...
               Var_ne(:,4), Var_ne(:,3), Var_ne(:,1), Var_ne(:,5));
         case 'steenzetting'
            [ZWaardeLaagZi(:,zi),q(:,zi)] = StatBerekenZSteenzetting ...
               (Profiel(i), FaalMechanisme.dijkopbouw(zi), ...
               Var_ne(:,4), Var_ne(:,3), Var_ne(:,1), Var_ne(:,5));
         case 'gras'
            [ZWaardeLaagZi(:,zi),q(:,zi)] = StatBerekenZGras (Profiel(i), ...
               FaalMechanisme.dijkopbouw(zi), WaarnStormIndex, ...
               Var_ne(:,4), Var_ne(:,3), Var_ne(:,1), Var_ne(:,5));
         case 'blokken'
            [ZWaardeLaagZi(:,zi),q(:,zi)] = StatBerekenZBlokinstabiliteit ...
               (Profiel(i), FaalMechanisme.dijkopbouw(zi), ...
               Var_ne(:,4), Var_ne(:,3), Var_ne(:,1), Var_ne(:,5));
         end
      end
      %
      % Bepaal minimum Z-waarde over alle lagen ...
      %
      [Z(:,i),LaagI]=min(ZWaardeLaagZi,[],2);
      %
      % ... en bepaal bijbehorende q waarden.
      %
      q=q((LaagI-1)*AantalWaarn+(1:AantalWaarn)');
   end
end
%
% Bepaal minimum Z-waarde over alle profielen ...
%
Z = min(Z,[],2);

Z = real(Z);
nanind = find(isnan(Z));
Z(nanind) = -Inf;
```

Stat/StatDataHerhalingstijdFalen

```
***************
*** 76,82 ****
        k = k+1;
        % bereken Z-functie voor gegeven faalmechanisme en verschuivingsvector
             z = StatBerekenZ (FaalMechanisme, Regio, Locatie, correctieRVB,
Profiel, ...
!
WaarnOverschrFactor+lambda,WaarnStormSectorIndex,Waarneming,Stat,StroomCor,faalMechID
);
        % bereken aantal Z-waarden klienr dan nul, m.a.w. bereken aantal verschoven
waarnemingen in het faalgebied
        [aantalZkleinerNul, Jm] = ...
          SelecteerMaxZ ( z, Kappa, WaarnStormIndex, SelectieWaarnemingen);
--- 76,83 ----
        k = k+1;
        % bereken Z-functie voor gegeven faalmechanisme en verschuivingsvector
             z = StatBerekenZ (FaalMechanisme, Regio, Locatie, correctieRVB,
Profiel, ...
!
WaarnOverschrFactor+lambda,WaarnStormSectorIndex,Waarneming,WaarnStormIndex, ...
!          Stat,StroomCor,faalMechID);
        % bereken aantal Z-waarden klienr dan nul, m.a.w. bereken aantal verschoven
waarnemingen in het faalgebied
        [aantalZkleinerNul, Jm] = ...
          SelecteerMaxZ ( z, Kappa, WaarnStormIndex, SelectieWaarnemingen);
```

Stat/StatFaalFreqMinimizeFunc

```
***************
*** 1,5 ****
  function ff= StatFaalFreqMinimizeFunc (lambda, FaalMechanisme, Regio, Locatie,
correctieRVB, ...
```

```
!    WaarnOverschrFactor, Waarneming, WaarnStormSectorIndex, Profiel,
Stat,StroomCor,faalMechID)

  %STATFAALFREQMINIMIZEFUNC  In StatFaalFreqMinimizeFunc wordt de Z-waarde
gekwadrateerd.
  %                          ff wordt later gebruikt om die waarneming te vinden die
het
--- 1,6 ----
  function ff= StatFaalFreqMinimizeFunc (lambda, FaalMechanisme, Regio, Locatie,
correctieRVB, ...
!    WaarnOverschrFactor, Waarneming, WaarnStormSectorIndex, WaarnStormIndex, ...
!    Profiel, Stat,StroomCor,faalMechID)

  %STATFAALFREQMINIMIZEFUNC  In StatFaalFreqMinimizeFunc wordt de Z-waarde
gekwadrateerd.
  %                          ff wordt later gebruikt om die waarneming te vinden die
het
***************
*** 36,41 ****

  z= StatBerekenZ(FaalMechanisme, Regio, Locatie, correctieRVB, Profiel, ...
     WaarnOverschrFactor+ones(size(WaarnOverschrFactor)).*lambda,...
!    WaarnStormSectorIndex,Waarneming, Stat, StroomCor, faalMechID);
!
! ff= z'*z;
--- 37,55 ----

  z= StatBerekenZ(FaalMechanisme, Regio, Locatie, correctieRVB, Profiel, ...
     WaarnOverschrFactor+ones(size(WaarnOverschrFactor)).*lambda,...
!    WaarnStormSectorIndex,Waarneming,WaarnStormIndex, Stat, StroomCor, faalMechID);
! %
! % Code aanpassing door Bert Jagers, WL | Delft Hydraulics
! %
! % De volgende code
! %
! % ff= z'*z;
! %
! % is vervangen door de onderstaande code op basis van de volgende overwegingen:
! %    (1) meerdere profielen zijn reeds weggemiddeld in StatBerekenZ wat dat
! %        betreft zou z een reele scalar zijn.
! %    (2) in het geval van grasbekleding bekijken we een hele storm in welk
! %        geval de verkregen z een vectorgrootheid is waarvan we nog het minimum
! %        moeten nemen (dit wordt bereikt aan het eind van de storm).
! %
! ff= min(z)^2;
```

Stat/statGrensTContour

```
***************
*** 133,141 ****
     Var(:,Xindex)= V1(:);
     Var(:,Yindex)= V2(:);

     % bereken Z-functie voor gegeven faalmechanisme
     zMatrixNormOffshore(:,k)=  StatBerekenZ (FaalMechanisme, ...
!       Regio,Locatie, correctieRVB, Profiel, Var(:,1:4), ones(l,1)*k,Var(:,5:7),
Stat, StroomCor, faalMechID);

     W1= ones(size(w2))*w1';
     W2= w2*ones(size(w1'));
--- 133,146 ----
     Var(:,Xindex)= V1(:);
     Var(:,Yindex)= V2(:);

+    %
+    % Bert Jagers (WL): dummy WaarnStormIndex om aanroep StatBerekenZ goed te
krijgen
+    %
+    WaarnStormIndex=[];
     % bereken Z-functie voor gegeven faalmechanisme
     zMatrixNormOffshore(:,k)=  StatBerekenZ (FaalMechanisme, ...
!       Regio,Locatie, correctieRVB, Profiel, Var(:,1:4), ones(l,1)*k,Var(:,5:7), ...
!       WaarnStormIndex, Stat, StroomCor, faalMechID);
```

```
     W1= ones(size(w2))*w1';
     W2= w2*ones(size(w1'));
```

Stat/StatKruinhBloksMinimizeFunc

```
function ff= StatKruinhBloksMinimizeFunc ...
   (ParamWaarde, FaalMechanisme,Profiel, NearshoreWaarden, WaarnStormIndex)

%STATKRUINHBLOKSMINIMIZEFUNC  In StatKruinhBloksMinimizeFunc worden een Z-waarde
%                             gekwadrateerd . ff wordt later gebruikt om die
waarneming
%                             te vinden die het dichtst bij Z=0 ofwel ff=0 ligt
%  AANROEP
%     ff= StatKruinhBloksMinimizeFunc ( ...
%        ParamWaarde, FaalMechanisme,Profiel, NearshoreWaarden, WaarnStormIndex)
%  INVOER
%     ParamWaarde:            de waarde van de te minimaliseren parameter, nl.
kruinhoogte,
%                             bloksterkte, of andere bekledingsgrootheid
%     FaalMechanisme:         het faalmechanisme ('golfoploop', 'golfoverslag' of
%                             'blokkeninstabiliteit')
%     Profiel:                het dijkprofiel
%     NearshoreWaarden:       een rijtje met een waterstand, windsnelheid,
piekperiode, sign.
%                             golfhoogte, golfrichting en windrichting
%     WaarnStormIndex:        de index van de storm behorend bij de waarneming

%  EXTERNE FUNCTIES
%     StatBerekenZuitNs

% Copyright © 2000 Rijkswaterstaat/RIKZ All Rights Reserved.
%
% Auteur: Abe Hoekstra/Ingrid Lammers, HKV lijn in water

% $Header: /PR/313/HYDRA_K/Stat/StatKruinhBloksMinimizeFunc.m 3     3-11-00 10:37
Hoekstra $
% $NoKeywords: $

switch FaalMechanisme.naam

case 'golfoploop',
   for i = 1:length(Profiel)
      Profiel(i).hoogte_kruin = ParamWaarde;
      FaalMechanisme.golfoploop.Hc(i) = ParamWaarde;
   end

case 'waterstand',
   for i = 1:length(Profiel)
      Profiel(i).hoogte_kruin = ParamWaarde;
      FaalMechanisme.golfoploop.Hc(i) = ParamWaarde;
   end

case 'golfoverslag',
   for i = 1:length(Profiel)
      Profiel(i).hoogte_kruin = ParamWaarde;
   end

case 'blokkeninstabiliteit'
   FaalMechanisme.blokkeninstabiliteit.Blok.dichtheid_laag = ParamWaarde;
   FaalMechanisme.blokkeninstabiliteit.Blok.dichtheid_hoog = ParamWaarde;

case 'instabiliteit van de bekleding'
   parameter = FaalMechanisme.ontwerpparameter;
   FaalMechanisme.dijkopbouw.bekledingsgegevens = ...
      setfield(FaalMechanisme.dijkopbouw.bekledingsgegevens,parameter,ParamWaarde);
   if strcmp(parameter,'dichtheid_hoog')
      FaalMechanisme.dijkopbouw.bekledingsgegevens.dichtheid_laag = ParamWaarde;
   end

end

z = StatBerekenZuitNs (FaalMechanisme, Profiel, NearshoreWaarden, WaarnStormIndex);
```

```
ff =z*z';
```

Ui/CBfaalmechanisme

```
***************
*** 7,12 ****
--- 7,16 ----
  %  INVOER
  %     RadioButtonHandle:   handle naar betreffende radio-button

+ %  EXTERNE FUNCTIES
+ %     CBRadio
+ %     CBafhankelijkheid
+
  % Copyright © 2000 Rijkswaterstaat/RIKZ All Rights Reserved.

  % Auteur: Abe Hoekstra, HKV lijn in water
***************
*** 21,26 ****
--- 25,31 ----
  rh = controls.faal.rh;
  eh = controls.faal.eh;
  th = controls.faal.th;
+ ph = controls.faal.ph;

  % alle radiobuttons uit en gekozen aan zetten
  set (rh,'Value',0);
***************
*** 45,48 ****
--- 50,65 ----
     else
        set (blokhandles,'Visible','off');
     end
  end
+
+ %
+ % Bert Jagers, WL | Delft Hydraulics
+ %
+ % Als gekozen faalmechanisme=4 (instabiliteit van de bekleding) is, dan
+ % de 'wijzig bekleding' control zichtbaar maken, anders onzichtbaar.
+ %
+ if RadioButtonHandle == rh(4),
+    set (ph,'Visible','on');
+ else
+    set (ph,'Visible','off');
+ end
```

Ui/CBmenu

```
***************
*** 68,75 ****
--- 68,79 ----
  %%%try

     figuur = get(callbackObject,'Parent');
+    while ~isempty(figuur) & ~strcmp(get(figuur,'type'),'figure')
+       figuur=get(figuur,'parent');
+    end

     tag = upper(get(callbackObject,'Tag'));
+    label = strrep(get(callbackObject,'label'),'&','');

     switch (tag)

***************
*** 91,112 ****
        FGhydra_k ('CONTROLE', figuur);

     case 'MNUREKENENFAALKANSEN'
!       % berkenen faalkansen
```

```
        StatusbalkTekst ('Berekenen herhalingstijd falen...');
        berekenFaalKansen ( get(get(callbackObject,'Parent'), 'Parent'));
        StatusbalkTekst ([]);

     case 'MNUREKENENKRUINHOOGTEBLOKDIKTE'
        % haal additonele invoer
        fig=FGkruinhoogteBlokdikte('CREATE');
        [ actie,herh, stilwater, ontwerp, Quantielen] = ...
           FGkruinhoogteBlokdikte('ACTIVATE',fig);

        switch actie,

        case 'REKENEN'
!          % bereken kruinhoogte/bloksterkte
!          StatusbalkTekst ('Berekenen kruinhoogte/bloksterkte...');
           berekenKruinhBloksterkte (get(get(callbackObject,'Parent'), 'Parent'), ...
              herh, stilwater, ontwerp, Quantielen);
           StatusbalkTekst ([]);
--- 95,174 ----
        FGhydra_k ('CONTROLE', figuur);

     case 'MNUREKENENFAALKANSEN'
!       %
!       % berekenen faalkansen
!       %
!       uiWaarden = FGhydra_k ('WAARDEN', figuur);
!       %
!       % controleer op de aanwezigheid van bekledingen waarvoor de belasting
!       % afhankelijk is van destormduur.
!       %
!       if StormDuurAfhankelijk(uiWaarden) & (uiWaarden.selectie.afhankelijkheid == 0
| uiWaarden.selectie.belastingvar == 0)
!          uiwait(msgbox('Het geselecteerde bekledingstype wordt gekenmerkt door
stormduurafhankelijke belasting. Selecteer waarnemingen, ongunstigste belasting voor
deze berekening.',label,'error','modal'));
!          return
!       end
!       %
!       % geef foutmelding als bekledingdefinitie niet klopt.
!       %
!       [Fout,Foutmelding] = controleerBekleding(uiWaarden);
!       if Fout
!          uiwait(msgbox(Foutmelding))
!          return
!       end
!       %
!       % geef waarschuwing in geval van meerdere bekledingen.
!       %
!       if MeerdereBekledingen(uiWaarden)
!          doorgaan=questdlg('Waarschuwing: Toetsing van samengestelde bekleding niet
volgens Leidraad Toetsen. Toch doorgaan?', ...
!              label, 'Ja','Nee','Nee');
!          if strcmp(doorgaan,'Nee'), return; end
!       end
!       %
        StatusbalkTekst ('Berekenen herhalingstijd falen...');
        berekenFaalKansen ( get(get(callbackObject,'Parent'), 'Parent'));
        StatusbalkTekst ([]);

     case 'MNUREKENENKRUINHOOGTEBLOKDIKTE'
+       %
+       % Deze optie is niet beschikbaar indien meerdere bekledingstypen
+       % geselecteerd zijn.
+       %
+       uiWaarden = FGhydra_k ('WAARDEN', figuur);
+       if MeerdereBekledingen(uiWaarden)
+          uiwait(msgbox('Deze optie is niet beschikbaar voor samengestelde
bekledingen.',label,'error','modal'))
+          return
+       end
+       %
+       % controleer op de aanwezigheid van bekledingen waarvoor de belasting
+       % afhankelijk is van destormduur.
+       %
```

```
+       if StormDuurAfhankelijk(uiWaarden) & (uiWaarden.selectie.afhankelijkheid == 0
| uiWaarden.selectie.belastingvar == 0)
+          uiwait(msgbox('Het geselecteerde bekledingstype wordt gekenmerkt door
stormduurafhankelijke belasting. Selecteer waarnemingen, ongunstigste belasting voor
een ontwerp.',label,'error','modal'));
+          return
+       end
+       %
+       % geef foutmelding als bekledingdefinitie niet klopt.
+       %
+       [Fout,Foutmelding] = controleerBekleding(uiWaarden);
+       if Fout
+          uiwait(msgbox(Foutmelding))
+          return
+       end
+       %
        % haal additonele invoer
        fig=FGkruinhoogteBlokdikte('CREATE');
        [ actie,herh, stilwater, ontwerp, Quantielen] = ...
           FGkruinhoogteBlokdikte('ACTIVATE',fig);
+       if StormDuurAfhankelijk(uiWaarden) & ontwerp
+          uiwait(msgbox('Het ontwerppunt is niet gedefinieerd voor bekleding
gekenmerkt door stormduurafhankelijke belasting. Ontwerppunt berekening
uitgeschakeld.','Tweedimensionale grafieken','warn','modal'));
+          ontwerp = 0;
+       end

        switch actie,

        case 'REKENEN'
!          % ontwerpberekening (voorheen: bereken kruinhoogte/bloksterkte)
!          StatusbalkTekst ('Ontwerpberekening...');
           berekenKruinhBloksterkte (get(get(callbackObject,'Parent'), 'Parent'), ...
              herh, stilwater, ontwerp, Quantielen);
           StatusbalkTekst ([]);
***************
*** 128,133 ****
--- 190,213 ----
        drawnow;

     case 'MNUGRAFIEKENFAALKANS'
+       %
+       % geef foutmelding als bekledingdefinitie niet klopt.
+       %
+       uiWaarden = FGhydra_k ('WAARDEN', figuur);
+       [Fout,Foutmelding] = controleerBekleding(uiWaarden);
+       if Fout
+          uiwait(msgbox(Foutmelding))
+          return
+       end
+       %
+       % geef waarschuwing in geval van meerdere bekledingen.
+       %
+       if MeerdereBekledingen(uiWaarden)
+          doorgaan=questdlg('Waarschuwing: Toetsing van samengestelde bekleding niet
volgens Leidraad Toetsen. Toch doorgaan?', ...
+             label, 'Ja','Nee','Nee');
+          if strcmp(doorgaan,'Nee'), return; end
+       end
+       %
        StatusbalkTekst ('Maken grafiek herhalingstijd falen...');
        grafiekfaalkans ( get(get(callbackObject,'Parent'), 'Parent'));
        StatusbalkTekst ([]);
***************
*** 138,149 ****
        StatusbalkTekst ([]);

     case 'MNUGRAFIEKEN2D'
        [ actie,parameters2D] = ...
           Fg2dParameters('SHOW', defParameters2D );
        drawnow;

        switch actie,
-
        case 'REKENEN'
```

```
           defParameters2D = parameters2D;
           StatusbalkTekst ('Maken tweedimensionale grafieken...');
--- 218,254 ----
        StatusbalkTekst ([]);

     case 'MNUGRAFIEKEN2D'
+       %
+       % GRAFIEK2D kan niet gemaakt worden in het geval van belastingen die
+       % afhankelijk zijn van de stormduur zoals in het geval van gras.
+       %
+       uiWaarden = FGhydra_k ('WAARDEN', figuur);
+       if StormDuurAfhankelijk(uiWaarden)
+          uiwait(msgbox('Deze optie is niet beschikbaar voor stormduurafhankelijke
belastingen.',label,'error','modal'));
+          return
+       end
+       %
+       % geef foutmelding als bekledingdefinitie niet klopt.
+       %
+       [Fout,Foutmelding] = controleerBekleding(uiWaarden);
+       if Fout
+          uiwait(msgbox(Foutmelding))
+          return
+       end
+       %
+       % geef waarschuwing in geval van meerdere bekledingen.
+       %
+       if MeerdereBekledingen(uiWaarden)
+          doorgaan=questdlg('Waarschuwing: Toetsing van samengestelde bekleding niet
volgens Leidraad Toetsen. Toch doorgaan?', ...
+             label, 'Ja','Nee','Nee');
+          if strcmp(doorgaan,'Nee'), return; end
+       end
+       %
        [ actie,parameters2D] = ...
           Fg2dParameters('SHOW', defParameters2D );
        drawnow;

        switch actie,
        case 'REKENEN'
           defParameters2D = parameters2D;
           StatusbalkTekst ('Maken tweedimensionale grafieken...');
***************
*** 152,159 ****
        end

     case 'MNUHELPINFORMATIE'
!       helpdlg({'' '       Hydra_k versie 2.04' '' '         oktober 2002' ...
!             '' '       Copyright © 2002 Rijkswaterstaat/RIKZ All Rights Reserved'
''}, ...
           'Informatie')
     end

--- 257,269 ----
        end

     case 'MNUHELPINFORMATIE'
!       helpdlg({''
!          '       Hydra_k versie 2.05 met aanpassingen volgens Fase 2 t/m 5'
!          '       van "Haalbaarheidsstudie HYDRA-K", WL | Delft Hydraulics,'
!          '       2003/2004'
!          ''
!          '       Copyright © 2002 Rijkswaterstaat/RIKZ All Rights Reserved'
!          ''}, ...
           'Informatie')
     end

***************
*** 295,301 ****
--- 405,422 ----
  case 3, %blokkeninstabiliteit
     titel = [titel {['Faalmechanisme: blokkeninstabiliteit, bloksterkte='
num2str(bloksterkte,'%.2f') ]}];

+ case 4, %instabiliteit van de bekleding
```

```
+    %
+    % bloksterkte zou moeten zijn: {parameter waarde}
+    %
+    if ischar(bloksterkte{2})
+       titel = [titel {sprintf('Faalmechanisme: instabiliteit van de bekleding,
%s=%s',bloksterkte{:})}];
+    elseif bloksterkte{2}<0
+       titel = [titel {sprintf('Faalmechanisme: instabiliteit van de bekleding, %s
onbepaald',bloksterkte{1})}];
+    else
+       titel = [titel {sprintf('Faalmechanisme: instabiliteit van de bekleding,
%s=%.2f',bloksterkte{:})}];
     end
+ end

  grensTSchaal = m.PossibleOffshoreParameters(Parameters2D.hor).Range;
  grensTschaal = [grensTSchaal m.PossibleOffshoreParameters(Parameters2D.ver).Range];
***************
*** 335,340 ****
--- 456,468 ----

  FG2d ('SHOW', figNorm );

+ %
+ % Infien geen ontwerppuntberekend is, hoeven we niet verder te gaan.
+ %
+ if isempty(ontwerppuntOffshore)
+    return
+ end
+
  % TEKEN HET OFFSHORE-PLAATJE
  % de schaal voor het plaatje bepalen
  if exist('dataOffshore','var'),
***************
*** 498,503 ****
--- 626,636 ----
        faalMechanisme.blokkeninstabiliteit.Blok(i).blokdikte_laag =
uiWaarden.profiel(i).blokdikte_laag;
        faalMechanisme.blokkeninstabiliteit.Blok(i).blokdikte_hoog =
uiWaarden.profiel(i).blokdikte_hoog;
     end
+
+ case 4, %instabiliteit van de bekleding
+    titel = [titel {['Faalmechanisme: instabiliteit van de bekleding' ]}];
+    faalMechanisme.naam = 'instabiliteit van de bekleding';
+    faalMechanisme.dijkopbouw = uiWaarden.dijkopbouw;
  end

  correctieRVB = BepaalCorrectie(1);
***************
*** 545,550 ****
--- 678,689 ----

  uiWaarden = FGhydra_k ('WAARDEN', Figuur);

+ if StormDuurAfhankelijk(uiWaarden)
+    if ~uiWaarden.selectie.afhankelijkheid | ~uiWaarden.selectie.belastingvar
+       uiwait(msgbox('Deze bekleding is alleen te toetsen op basis van waarnemingen
en ongunstige belasting.','Herhalingstijd falen','error','modal'));
+       return
+    end
+ end

  [kappa, waarnOverschrFreq,  waarneming , waarnStormSectorIndex, WaarnStormIndex]
...
     = HaalWaarnemingen ( ...
***************
*** 578,583 ****
--- 717,726 ----
        faalMechanisme.blokkeninstabiliteit.Blok(i).blokdikte_hoog =
uiWaarden.profiel(i).blokdikte_hoog;
     end

+ case 4, %instabiliteit van de bekleding
+    Titel = [Titel {['Faalmechanisme: instabiliteit van de bekleding' ]}];
```

```
+    faalMechanisme.naam = 'instabiliteit van de bekleding';
+    faalMechanisme.dijkopbouw = uiWaarden.dijkopbouw;
  end

  correctieRVB = BepaalCorrectie(1);
***************
*** 625,643 ****
     UiWaarden.regiodata.richtingSector, ...
     UiWaarden.kappa);

- switch UiWaarden.faalmechanisme
- case 1, % golfoploop
-    faalMechanisme.naam = 'golfoploop';
-
- case 2, % golfoverslag
-    faalMechanisme.naam = 'golfoverslag';
-    faalMechanisme.golfoverslag.Qc = UiWaarden.overslagdebiet;
-
- case 3, %blokkeninstabiliteit
-    faalMechanisme.naam = 'blokkeninstabiliteit';
-
- end
-
  correctieRVB = BepaalCorrectie(1);

  switch UiWaarden.faalmechanisme
--- 768,773 ----
***************
*** 660,665 ****
--- 790,799 ----
        faalMechanisme.blokkeninstabiliteit.Blok(i).blokdikte_hoog =
UiWaarden.profiel(i).blokdikte_hoog;
     end

+ case 4, %instabiliteit van de bekleding
+    faalMechanisme=[];
+    faalMechanisme.naam = 'instabiliteit van de bekleding';
+    faalMechanisme.dijkopbouw = UiWaarden.dijkopbouw;
  end

  [indexFalen, freq] = StatBerekenFaalfrequentie (...
***************
*** 667,673 ****
     WaarnStormIndex, 1e-8, UiWaarden.regiodata.stat, UiWaarden.profiel,
UiWaarden.kappa, ...

UiWaarden.selectie.belastingvar,UiWaarden.selectie.stroming,UiWaarden.faalmechanisme)
;

-
  Uitvoer('p','\n%40s','BEREKENEN HERHALINGSTIJD FALEN')
  Printwaarden (UiWaarden);

--- 801,806 ----
***************
*** 730,736 ****
  %                              faalmechanisme=golfoverslag)
  %  UITVOER
  %     kruinhoogte:            de berekende kruinhoogte of [];
! %     bloksterkte:            de berekende bloksterkte of [];
  %     q:                      het berekende overslagdebiet of [];
  %     ontwerppuntOffshore:    het berekende ontwerppunt offshore of [];
  %     ontwerppuntNearshore:   het berekende ontwerppunt nearshore of [];
--- 863,871 ----
  %                              faalmechanisme=golfoverslag)
  %  UITVOER
  %     kruinhoogte:            de berekende kruinhoogte of [];
! %     bloksterkte:            de berekende bloksterkte of [], of {parameternaam en
! %                             parameterwaarde} in het geval van instabiliteit van
de
! %                             bekleding;
  %     q:                      het berekende overslagdebiet of [];
  %     ontwerppuntOffshore:    het berekende ontwerppunt offshore of [];
  %     ontwerppuntNearshore:   het berekende ontwerppunt nearshore of [];
***************
```

```
*** 774,782 ****
  case 3, %blokkeninstabiliteit
     faalMechanisme.naam = 'blokkeninstabiliteit';

  end

! Uitvoer('p','\n%30s','BEREKENEN KRUINHOOGTE/BLOKSTERKTE')
  Printwaarden (UiWaarden);
  Uitvoer('p','\n%50s%s%s','Gem. herhalingstijd voor falen: ', num2str(Herh), '
jaar')

--- 909,920 ----
  case 3, %blokkeninstabiliteit
     faalMechanisme.naam = 'blokkeninstabiliteit';

+ case 4, %instabiliteit van de bekleding
+    faalMechanisme.naam = 'instabiliteit van de bekleding';
+
  end

! Uitvoer('p','\n%30s','ONTWERPBEREKENING')
  Printwaarden (UiWaarden);
  Uitvoer('p','\n%50s%s%s','Gem. herhalingstijd voor falen: ', num2str(Herh), '
jaar')

***************
*** 848,855 ****
--- 986,1046 ----
     % Aarninkhof (WL): Also return indexFalen, for plotting purposes
     faalMechanisme.indexFalen = indexFalen;

+ case 4,
+    [ ontwerpwaarde, indexFalen] = ...
+       StatBerekenBekleding (UiWaarden.regiodata.naam, UiWaarden.locatie,
correctieRVB, waarnOverschrFreq, waarneming, ...
+       waarnStormSectorIndex, WaarnStormIndex, UiWaarden.regiodata.stat,
UiWaarden.profiel, ...
+       Herh, UiWaarden.kappa,
UiWaarden.selectie.belastingvar,UiWaarden.selectie.stroming,UiWaarden.faalmechanisme,
...
+       UiWaarden.dijkopbouw);
+
+    %
+    % Onderstaande gaat alleen goed als er maar 1 bekledingssegment is ....
+    %
+    faalMechanisme.dijkopbouw = UiWaarden.dijkopbouw;
+    switch UiWaarden.dijkopbouw.bekleding
+    case 'asfalt'
+       parameter = 'laagdikte asfalt';
+       faalMechanisme.dijkopbouw.bekledingsgegevens.laagdikte = ontwerpwaarde;
+    case 'blokken'
+       parameter = 'bloksterkte';
+       faalMechanisme.dijkopbouw.bekledingsgegevens.dichtheid_laag = ontwerpwaarde;
+       faalMechanisme.dijkopbouw.bekledingsgegevens.dichtheid_hoog = ontwerpwaarde;
+       faalMechanisme.dijkopbouw.bekledingsgegevens.blokdikte_laag = 1;
+       faalMechanisme.dijkopbouw.bekledingsgegevens.blokdikte_hoog = 1;
+    case 'gras'
+       parameter = 'kwaliteit grasmat';
+       faalMechanisme.dijkopbouw.bekledingsgegevens.kwaliteit = ontwerpwaarde;
+    case 'steenzetting'
+       parameter = 'sterkte toplaag';
+       faalMechanisme.dijkopbouw.bekledingsgegevens.relatievedichtheid =
ontwerpwaarde;
+       faalMechanisme.dijkopbouw.bekledingsgegevens.toplaagdikte = 1;
     end

+    string = sprintf('Ontwerp%s gebaseerd op %s: ', parameter, faalMechanisme.naam);
+    if isnumeric(ontwerpwaarde) & ~isfinite(ontwerpwaarde)
+       if ontwerpwaarde<0
+          Uitvoer('p','\n\n%50s%s\n', string ,'geen of onvoldoende faalwaarnemingen
gevonden');
+       else
+          Uitvoer('p','\n\n%50s%s\n', string ,'nooit voldoende, faalt altijd');
+       end
+    elseif ischar(ontwerpwaarde)
```

```
+       Uitvoer('p','\n\n%50s%s\n', string ,ontwerpwaarde);
+    else
+       Uitvoer('p','\n\n%50s%10.2f\n', string ,ontwerpwaarde);
+    end
+    % Aarninkhof (WL): Also return indexFalen, for plotting purposes
+    faalMechanisme.indexFalen = indexFalen;
+    %
+    % Voer parameter en ontwerpwaarde uit als bloksterkte
+    %
+    bloksterkte = {parameter ontwerpwaarde};
+ end
+
+ if Ontwerp & stormFlanken
+    Ontwerp=0;
+    Uitvoer('p','Geen ontwerp i.v.m. gebruik stormflanken.');
+ end
+
  if Ontwerp & ~isempty(indexFalen),
     % ontwerppunt berekenen
     [ ontwerppuntOffshore, ontwerppuntNearshore] = ...
```

Ui/FGhydra_k

```
***************
*** 68,74 ****

     h = findobj(h_hs, 'Tag', 'mnuRekenen') ;set (h,'Label','&Rekenen');
     h = findobj(h_hs, 'Tag', 'mnuRekenenFaalkansen') ;set
(h,'Label','&Herhalingstijd falen');
!    h = findobj(h_hs, 'Tag', 'mnuRekenenKruinhoogteBlokdikte') ;set
(h,'Label','&Kruinhoogte/bloksterkte...');
     h = findobj(h_hs, 'Tag', 'mnuRekenenOffshoreNearshore') ;set
(h,'Label','&Offshore-nearshore...');
     h = findobj(h_hs, 'Tag', 'mnuRekenenDeterministischRekenen') ;set
(h,'Label','&Deterministisch...');
     h = findobj(h_hs, 'Tag', 'mnuRekenenStekker') ;set (h,'Label','&Stekker');
--- 68,74 ----

     h = findobj(h_hs, 'Tag', 'mnuRekenen') ;set (h,'Label','&Rekenen');
     h = findobj(h_hs, 'Tag', 'mnuRekenenFaalkansen') ;set
(h,'Label','&Herhalingstijd falen');
!    h = findobj(h_hs, 'Tag', 'mnuRekenenKruinhoogteBlokdikte') ;set
(h,'Label','Ontwerpbere&kening...');
     h = findobj(h_hs, 'Tag', 'mnuRekenenOffshoreNearshore') ;set
(h,'Label','&Offshore-nearshore...');
     h = findobj(h_hs, 'Tag', 'mnuRekenenDeterministischRekenen') ;set
(h,'Label','&Deterministisch...');
     h = findobj(h_hs, 'Tag', 'mnuRekenenStekker') ;set (h,'Label','&Stekker');
***************
*** 116,127 ****
--- 116,130 ----

     th = handles.faal.th;
     rh = handles.faal.rh;
+    ph = handles.faal.ph;
     set (th(1),'String', 'Faalmechanisme:');
     set (rh(1),'String', 'Golfoploop');
     set (rh(2),'String', 'Golfoverslag');
     set (rh(3),'String', 'Blokkeninstabiliteit');
+    set (rh(4),'String', 'Instabiliteit van de bekleding');
     set (th(2),'String', 'Overslagdebiet:');
     set (th(3),'String', 'l/s/m');
+    set (ph(1),'String', 'Wijzig bekleding');

     th = handles.blokken.th;
     set (th(1),'String', 'van teen tot berm');
***************
*** 178,184 ****
  % Copyright © 2000 Rijkswaterstaat/RIKZ All Rights Reserved.

  schermgrootte = get(0,'ScreenSize');
! figuurpositie(3:4)=[550 600]; % de breedte en hoogte
  figuurpositie(1:2) = (schermgrootte(3:4)-figuurpositie(3:4))/2;
```

```
  h0 = figure(...
--- 181,187 ----
  % Copyright © 2000 Rijkswaterstaat/RIKZ All Rights Reserved.

  schermgrootte = get(0,'ScreenSize');
! figuurpositie(3:4)=[550 620]; % de breedte en hoogte
  figuurpositie(1:2) = (schermgrootte(3:4)-figuurpositie(3:4))/2;

  h0 = figure(...
***************
*** 209,240 ****

  yoffset = positie(4);

  [ handles.regio.th handles.regio.ph] = ...
!    maakRegioControls (h0,0,yoffset-30);

  % zet de controls voor het profiel in het figuur
  [ handles.profiel.th handles.profiel.eh] = ...
!    maakProfielControls (h0,0,yoffset-80);

  % zet de 'Kappa'-controls in het figuur
  [ handles.kappa.th handles.kappa.eh] = ...
!    maakKappaControls (h0,0, yoffset-180);

  % zet de controls voor het faalmechanisme in het figuur
! [ handles.faal.th handles.faal.eh handles.faal.rh] ...
!    = maakFaalmechanismeControls (h0, 0, yoffset-200);

  [handles.blokken.th handles.blokken.eh] = ...
!    maakBlokkenControls (h0, 0, yoffset-260);

  [ handles.afhank.th handles.afhank.rh ] = ...
!    maakAfhankelijkheidControls (h0, 0, yoffset-320);

  [ handles.selectie.th handles.selectie.rh ] = ...
!    maakSelectieControls (h0, 0, yoffset-360);

  [ handles.stroming.th handles.stroming.rh ] = ...
!    maakStromingControls (h0, 0, yoffset-400);

  % maak de status-balk
  [ handles.status.eh] = maakTextControl ( h0, 'left', [5 5 positie(3)-10 13]);
--- 212,251 ----

  yoffset = positie(4);

+ yoffset = yoffset-30;
  [ handles.regio.th handles.regio.ph] = ...
!    maakRegioControls (h0,0,yoffset);

  % zet de controls voor het profiel in het figuur
+ yoffset=yoffset-50;
  [ handles.profiel.th handles.profiel.eh] = ...
!    maakProfielControls (h0,0,yoffset);

  % zet de 'Kappa'-controls in het figuur
+ yoffset=yoffset-100;
  [ handles.kappa.th handles.kappa.eh] = ...
!    maakKappaControls (h0,0, yoffset);

  % zet de controls voor het faalmechanisme in het figuur
! yoffset=yoffset-20;
! [ handles.faal.th handles.faal.eh handles.faal.rh handles.faal.ph] ...
!    = maakFaalmechanismeControls (h0, 0, yoffset);

+ yoffset=yoffset-80;
  [handles.blokken.th handles.blokken.eh] = ...
!    maakBlokkenControls (h0, 0, yoffset);

+ yoffset=yoffset-60;
  [ handles.afhank.th handles.afhank.rh ] = ...
!    maakAfhankelijkheidControls (h0, 0, yoffset);
```

```
+ yoffset=yoffset-40;
  [ handles.selectie.th handles.selectie.rh ] = ...
!    maakSelectieControls (h0, 0, yoffset);

+ yoffset=yoffset-40;
  [ handles.stroming.th handles.stroming.rh ] = ...
!    maakStromingControls (h0, 0, yoffset);

  % maak de status-balk
  [ handles.status.eh] = maakTextControl ( h0, 'left', [5 5 positie(3)-10 13]);
***************
*** 488,494 ****

  % ------------------------------------------------------------------------

! function [th, eh, rh] = maakFaalmechanismeControls (h0, offsetx, offsety)

  %MAAKFAALMECHANISMECONTROLS   Plaats controls voor selectie van faalmechanisme
  %  AANROEP
--- 499,505 ----

  % ------------------------------------------------------------------------

! function [th, eh, rh, ph] = maakFaalmechanismeControls (h0, offsetx, offsety)

  %MAAKFAALMECHANISMECONTROLS   Plaats controls voor selectie van faalmechanisme
  %  AANROEP
***************
*** 501,511 ****
--- 512,524 ----
  %     th:      vector met handles naar text-objecten
  %     eh:      vector met handles naar editcontrol-objecten
  %     rh:      vector met handles naar radiobuttoncontrol-objecten
+ %     ph:      vector met handles naar pushbuttoncontrol-objecten

  %  EXTERNE FUNCTIES
  %     maakTextControl
  %     maakEditControl
  %     maakRadioControl
+ %     maakPushbuttonControl

  % Copyright © 2000 Rijkswaterstaat/RIKZ All Rights Reserved.

***************
*** 522,527 ****
--- 535,542 ----

  rh(3) = maakRadioControl ( h0,[x offsety-36 95 13],'CBfaalmechanisme (gcbo)');

+ rh(4) = maakRadioControl ( h0,[x offsety-54 115 13],'CBfaalmechanisme (gcbo)');
+
  x =offsetx+170;
  th(2) = maakTextControl (h0,'right',[x offsety-18 70 13]);

***************
*** 530,535 ****
--- 545,553 ----
  eh = maakEditControl (h0, 'right',[x offsety-18 w 15],...
     'CBedit(gcbo);','fltwaarde <= 0 | fltwaarde > 100', '0< debiet <= 100');

+ ph = maakPushbuttonControl ( h0,[x offsety-54], [], 'CBbekleding SHOW','wijzig
bekleding');
+ set(ph,'position',get(ph,'position')+[0 0 54 0])
+
  x =offsetx+295;
  th(3) = maakTextControl (h0,'left',[x offsety-18 70 13]);

***************
*** 737,742 ****
--- 755,782 ----

     if get(handles.faal.rh(3),'value'), set ([beh' handles.blokken.th'],{'Visible'},
{'on'});end
  end
+ %
```

```
+ % Zet de default dijkopbouw bij het opstarten (alleen
+ % dan is userdata leeg) ...
+ %
+ if isempty(get(handles.faal.ph,'userdata'))
+    dijkopbouw(1).ondergrens=-inf;
+    dijkopbouw(1).bovengrens=inf;
+    dijkopbouw(1).bekleding='blokken';
+    %
+    % Eventueel toevoegen indien er opgestart wordt met
+    % de optie "Combinatie std. profielen"
+    %
+    %if keuze > size(profielen,2)
+    %   profiel = profielen(1);
+    %end
+    dijkopbouw(1).bekledingsgegevens.dichtheid_laag = profiel.dichtheid_laag;
+    dijkopbouw(1).bekledingsgegevens.dichtheid_hoog = profiel.dichtheid_hoog;
+    dijkopbouw(1).bekledingsgegevens.blokdikte_laag = profiel.blokdikte_laag;
+    dijkopbouw(1).bekledingsgegevens.blokdikte_hoog = profiel.blokdikte_hoog;
+    set(handles.faal.ph,'userdata',dijkopbouw)
+ end
+
  % -----------------------------------------------------------------------
  function zetBeginwaarden (h0)

***************
*** 1094,1104 ****
--- 1134,1161 ----
  waarden.profiel = haalProfiel (handles, handles.profiel.eh, handles.blokken.eh);
  waarden.kappa = haalKappa (handles.kappa.eh);
  waarden.faalmechanisme = haalFaalmechanisme(handles.faal.rh);
+ waarden.dijkopbouw = haalDijkopbouw(handles.faal.ph);
  waarden.overslagdebiet = haalOverslagDebiet(handles.faal.eh);
  waarden.regiodata = haalRegiodata (handles.regio.ph);
  waarden.selectie = haalSelectieCriterium (handles.selectie.rh, handles.afhank.rh,
handles.stroming.rh);

  % -----------------------------------------------------------------------
+ function Dijkopbouw = haalDijkopbouw (ph)
+
+ %HAALDIJKOPBOUW   de dijkopbouw/bekledingsgegevens ophalen
+ %  AANROEP
+ %     Dijkopbouw = haalDijkopbouw (ph)
+ %  INVOER
+ %     ph:         handle van bekledingsknop
+ %  UITVOER
+ %     Dijkopbouw: structure met dijkopbouw/bekledingsgegevens
+
+ % Copyright © 2000 Rijkswaterstaat/RIKZ All Rights Reserved.
+
+ Dijkopbouw = get(ph,'Userdata');
+
+
+ % -----------------------------------------------------------------------
  function Locatie = haalLocatie (ph)

  %HAALLOCATIE   de (x,y)-coordinaten ophalen
```

Ui/FGkruinhoogteBlokdikte

```
***************
*** 114,120 ****
     'Visible'         ,'off'                    , ...
     'Color'           ,[0.8 0.8 0.8]            , ...
     'KeyPressFcn'     ,'eval('''')'             , ...
!    'Name'            ,'Hydra-K (Kruinhoogte/bloksterkte)'  , ...
          'Pointer'         ,'arrow'                    , ...
     'Resize'          ,'off'                    , ...
     'Units'           ,'points'                 , ...
--- 114,120 ----
     'Visible'         ,'off'                    , ...
     'Color'           ,[0.8 0.8 0.8]            , ...
     'KeyPressFcn'     ,'eval('''')'             , ...
!    'Name'            ,'Hydra-K ontwerpberekening'  , ...
```

```
    'Pointer'          ,'arrow'                   , ...
'Resize'           ,'off'                     , ...
'Units'            ,'points'                  , ...
```

A4 Gewijzigde routines t.b.v. de additionele stochast

In deze bijlage wordt de code getoond van de gewijzigde routines t.b.v. het inbouwen van de extra stochast. Zie bijlage A1 voor de context waarin deze routines gebruikt worden. In de onderstaande tekst zijn alleen de gewijzigde regels (met enkele voorafgaande en daarop volgende regels) weergegeven tenzij de gehele nieuwe code ongeveer even lang is in welk geval de nieuwe code wordt getoond. Bij het weergeven van de verschillen is de volgende codering gebruikt: + toegevoegde regels, - verwijderde regels, ! gewijzigde regels.

Stat/StatLaadStatistiek

```
***************
*** 45,55 ****

        wloBestand = IniFile ('LEES', 'statistiek', [sRegio '_wlo'], '' );
        wsoBestand = IniFile ('LEES', 'statistiek', [sRegio '_wso'], '' );
        tm10Bestand = IniFile ('LEES', 'statistiek', [sRegio '_tm10'], '' );

        % lees bestanden
        %hsoStat = load('data\signgolfhoogte.e30');
!       %tpoStat = load('data\piekperiode.e30');
        wloStat = load(['data\' wloBestand]);
        wsoStat = load(['data\' wsoBestand]);

--- 45,56 ----

        wloBestand = IniFile ('LEES', 'statistiek', [sRegio '_wlo'], '' );
        wsoBestand = IniFile ('LEES', 'statistiek', [sRegio '_wso'], '' );
+       tpoBestand = IniFile ('LEES', 'statistiek', [sRegio '_tpo'], '' );
        tm10Bestand = IniFile ('LEES', 'statistiek', [sRegio '_tm10'], '' );

        % lees bestanden
        %hsoStat = load('data\signgolfhoogte.e30');
!       tpoStat = load(['data\' tpoBestand]);
        wloStat = load(['data\' wloBestand]);
        wsoStat = load(['data\' wsoBestand]);

***************
*** 68,74 ****

        lStat(:,1,:) = wloStat(:,[2,3,5,4])';
        lStat(:,2,:) = wsoStat(:,[2,3,5,4])';
!       lStat(:,3,:) = NaN; % tpoStat(:,[2,3,5,4])';
        lStat(:,4,:) = NaN; % hsoStat(:,[2,3,5,4])';
        % alle richtingsectoren zijn hetzelfde, dus kiezen we er maar een

--- 69,75 ----

        lStat(:,1,:) = wloStat(:,[2,3,5,4])';
        lStat(:,2,:) = wsoStat(:,[2,3,5,4])';
!       lStat(:,3,:) = tpoStat(:,[2,3,5,4])';
        lStat(:,4,:) = NaN; % hsoStat(:,[2,3,5,4])';
        % alle richtingsectoren zijn hetzelfde, dus kiezen we er maar een
```

Stat/StatWeibullFreq

```
***************
*** 65,69 ****
     zeeSpiegelStr(find(isspace(zeeSpiegelStr))) = '_';
     zeesp_corr = str2num(IniFile('LEES', 'general', zeeSpiegelStr,0.05));
  end
!
  frequentie= mu.*exp(-((a-zeesp_corr)./sigma).^alpha+(omega./sigma).^alpha);
--- 65,74 ----
     zeeSpiegelStr(find(isspace(zeeSpiegelStr))) = '_';
     zeesp_corr = str2num(IniFile('LEES', 'general', zeeSpiegelStr,0.05));
```

```
  end
! %
! % Bert Jagers (WL, 2004-02-12)
! % In tegenstelling tot StatWeibullInv wordt deze functie alleen voor
! % waterstanden gebruikt en dus wordt de zeesp_corr hier wel voor alle
! % elementen van a toegepast.
! %
  frequentie= mu.*exp(-((a-zeesp_corr)./sigma).^alpha+(omega./sigma).^alpha);
```

Stat/StatWeibullInv

```
function a = StatWeibullInv ( OverschrFreq,StormSectorIndex,StatWaterstand, Regio )

%STATWEIBULLINV   Deze functie berekent de waterstand bij een gegeven
overschrijdingsfrequentie
%                 uit de inverse Weibullverdeling voor de waterstand
%                 Deze functie is beschreven in het functioneel ontwerp paragraaf
3.1.2 stap 1
%                 van Methode de Haan: Extreme-waarden-statistiek.
%                 De formule voor de frequentie in het functioneel ontwerp kan worden
%                 omgeschreven tot de formule voor a in dit programma.
%  AANROEP
%     [a] = ...
%     StatWeibullInv(OverschrFreq,StormSectorIndex,StatWaterstand))
%  INVOER
%     OverschrFreq:           de overschrijdingsfrequenties van de
waterstandwaarnemingen
%     StormSectorIndex:       de index van de stormsector (wordt bepaald door de
windrichting)
%     StatWaterstand:         de Weibull parameters voor de waterstand voor alle
stormsectoren
%                             De Weibull parameters zijn:
%                                omega als omega, de drempelwaarde
%                                mu als mu(omega), de stormfrequentie
%                                sigma als sigma, de schaalparameter
%                                alpha als alpha, de vorm- krommingsparameter
%     Regio                   naam van de regio (in verband met zeespiegelrijzing)
%  UITVOER
%     a:                      de waterstand, bij de gegeven overschrijdingsfrequentie
en
%                             stormsector
%                             De volgende variabelen worden in het functioneel
ontwerp
%                             weergegeven als:
%                                omega als omega, de drempelwaarde
%                                mu als mu(omega), de stormfrequentie
%                                sigma als sigma, de schaalparameter
%                                alpha als alpha, de vorm- krommingsparameter
%                                de overschrijdingsfrequentie als F[u>a]
%                                de zeespiegelcorrectie 0.05

%  EXTERNE FUNCTIES
%     IniFile

% Copyright © 2000 Rijkswaterstaat/RIKZ All Rights Reserved.

% Auteur: Abe Hoekstra/Ingrid Lammers, HKV lijn in water

% $Header: /PR/313/HYDRA_K/Stat/StatWeibullInv.m 2     14-08-00 8:56 Hoekstra $
% $NoKeywords: $

persistent zeesp_corr
persistent sRegio

%
% De permutatie en "squeeze" zijn nodig omdat de overschrijdingsfrequentie een
% tweedimensionale array is. Dit is echter % een tijdvretende operatie.
% Andere oplossingen?
%
% Bert Jagers (WL): In plaats van omega, mu, sigma en alpha te converteren,
% converteer OverschrFreq en a. Dit geeft 50% reductie ...
%
%omega = squeeze(permute(StatWaterstand(1,:,StormSectorIndex),[3 2 1]));
```

```
%mu    = squeeze(permute(StatWaterstand(2,:,StormSectorIndex),[3 2 1]));
%sigma = squeeze(permute(StatWaterstand(3,:,StormSectorIndex),[3 2 1]));
%alpha = squeeze(permute(StatWaterstand(4,:,StormSectorIndex),[3 2 1]));
omega = StatWaterstand(1,:,StormSectorIndex);
mu    = StatWaterstand(2,:,StormSectorIndex);
sigma = StatWaterstand(3,:,StormSectorIndex);
alpha = StatWaterstand(4,:,StormSectorIndex);

% correctie voor zeespiegelstijging
% Merk op:
% Bewaar het resultaat, omdat anders de INI-file 20000 wordt uitgeroepen!
leesz = 0 ;
if isempty(sRegio),
   leesz  = 1 ;
   sRegio = Regio{:};
end
if ~strcmp(sRegio,Regio{:}),
   leesz = 1 ;
end
if leesz == 1,
   sRegio = Regio{:};
   zeeSpiegelStr = [sRegio '_zeespcorr'];
   zeeSpiegelStr(find(isspace(zeeSpiegelStr))) = '_';
   zeesp_corr = str2num(IniFile('LEES', 'general', zeeSpiegelStr,0.05));
end

szmu=size(mu);
if ~isequal(szmu,[1 1])
   %
   % Bert Jagers (WL): In plaats van omega, mu, sigma en alpha te converteren,
   % converteer nu OverschrFreq (hier) en uiteindelijke matrix a.
   %
   OverschrFreq= reshape(OverschrFreq.',size(mu));

   a= sigma.*max(eps,(omega./sigma).^alpha+log(mu)+OverschrFreq).^(1./alpha);
   %
   % Zeespiegelcorrectie alleen voor waterstand (element 1) !
   %
   a(1,1,:) = a(1,1,:) + zeesp_corr;
   %
   % Bert Jagers (WL): In plaats van omega, mu, sigma en alpha te converteren,
   % converteer nu OverschrFreq (boven) en uiteindelijke matrix a (hier).
   %
   a = reshape(a,[szmu(2:end) 1]).';
else
   a= zeesp_corr +
sigma.*max(eps,(omega./sigma).^alpha+log(mu)+OverschrFreq).^(1./alpha);
end
```

Transformeer/ControleerRand2001Data

```
***************
*** 22,27 ****
--- 22,28 ----

  okee = find(strcmp(MIJNMODEL.Rand2001.Parameters,'WLO'));
  okee = okee & find(strcmp(MIJNMODEL.Rand2001.Parameters,'WSO'));
+ okee = okee & find(strcmp(MIJNMODEL.Rand2001.Parameters,'TPO'));
  okee = okee & find(strcmp(MIJNMODEL.Rand2001.Parameters,'WRO'));
  okee = okee & find(strcmp(MIJNMODEL.Rand2001.Parameters,'WLN'));
  okee = okee & find(strcmp(MIJNMODEL.Rand2001.Parameters,'TPN'));
***************
*** 42,55 ****

  % controleer per windrichting offshore of er wel een volledig gevulde matrix is
  % voor waterstand*windsnelheid. Controleer ook of err geen dubbele zijn

! windrichtingen = unique(MIJNMODEL.Rand2001.tabel(:,3));

  for windrichting =  windrichtingen'
!    ind=find(MIJNMODEL.Rand2001.tabel(:,3)==windrichting );
!    data = MIJNMODEL.Rand2001.tabel(ind,1:2);
     % check op dubbele
```

```
!    okee = prod(size(data)==size(unique(data,'rows')));
     if ~okee,
        error ('Er komen dubbele waarden voor in Rand2001 bestand');
     end
--- 43,57 ----

  % controleer per windrichting offshore of er wel een volledig gevulde matrix is
  % voor waterstand*windsnelheid. Controleer ook of err geen dubbele zijn
+ WRO=find(strcmp(MIJNMODEL.Rand2001.Parameters,'WRO'));

! windrichtingen = unique(MIJNMODEL.Rand2001.tabel(:,WRO));

  for windrichting =  windrichtingen'
!    ind=find(MIJNMODEL.Rand2001.tabel(:,WRO)==windrichting );
!    data = MIJNMODEL.Rand2001.tabel(ind,1:WRO-1);
     % check op dubbele
!    okee = all(size(data)==size(unique(data,'rows')));
     if ~okee,
        error ('Er komen dubbele waarden voor in Rand2001 bestand');
     end
```

Transformeer/LeesDataVanRand2001

```
***************
*** 167,173 ****

     OffShoreRanges =
{MIJNMODEL.PossibleOffshoreParameters([Rand2001.OffshoreIndex]).Range};
     OffShoreRanges = cat(1,OffShoreRanges{:});
!    NearShoreRanges =
{MIJNMODEL.PossibleOffshoreParameters([Rand2001.NearshoreIndex]).Range};
     NearShoreRanges = cat(1,NearShoreRanges{:});

     Rand2001.OffShoreIsRichting =
[MIJNMODEL.PossibleOffshoreParameters(Rand2001.OffshoreIndex).IsRichting];
--- 167,173 ----

     OffShoreRanges =
{MIJNMODEL.PossibleOffshoreParameters([Rand2001.OffshoreIndex]).Range};
     OffShoreRanges = cat(1,OffShoreRanges{:});
!    NearShoreRanges =
{MIJNMODEL.PossibleNearshoreParameters([Rand2001.NearshoreIndex]).Range};
     NearShoreRanges = cat(1,NearShoreRanges{:});

     Rand2001.OffShoreIsRichting =
[MIJNMODEL.PossibleOffshoreParameters(Rand2001.OffshoreIndex).IsRichting];
```

Transformeer/Model

```
function        [Model]= Model ()

%MODEL         Definieer de Model structure.
%  AANROEP
%     [Model]= Model
%  UITVOER
%     Model:   een structure die er als volg uitziet
%                 PossibleOffshoreParameters: [1x6 struct]
%                    Code:          'WLO'  ,'WSO'  ,'TPO'   ,'HSO'   ,'THO'  ,'WRO'
%                    IsRichting:    [ 0       0       0       0       1       1   ]
%                    DbName:        'DZWAT','WINDS','DZTP'  ,'DZHSX' ,''     ,'WINDR'
%                    Description:   beshcrijving van elke parameter
%                    Unit:          eenheid van de parameters
%                    Range:         waarde ranges
%                    Value:         NaN
%                 PossibleNearshoreParameters: [1x6 struct]
%                    Code:          'WLN'  ,'WSN'  ,'TPN'   ,'HSN'   ,'THN'  ,'WRN'
%                    IsRichting:    [ 0       0       0       0       1       1   ]
%                    DbName:        'WATDW','' ,'TPM'   ,'HSIGN' ,'GOLFR','' 
%                                                 of 'TM-10' bij gebruik
NieuweGolfFormules
%                    Description:   beshcrijving van elke parameter
```

```
%                   Unit:          eenheid van de parameters
%                   Range:         waarde ranges
%                   Value:         NaN
%                NrOfPossibleOffshoreParameters: [1x1 struct]
%                   Value:         aantal offshore parameters
%                NrOfPossibleNearshoreParameters: [1x1 struct]
%                   Value:         aantal nearshore parameters

% Copyright © 2000 Rijkswaterstaat/RIKZ All Rights Reserved.

% Auteur: Abe Hoekstra, HKV lijn in water

OffshoreParms        = char   ('WLO'  ,'WSO'  ,'TPO'  ,'HSO'  ,'THO'  ,'WRO'  );
OffshoreIsRichting   = logical([ 0       0       0       0       1       1   ]);
OffshoreDbFields     = char   ('DZWAT','WINDS','DZTPX','DZHSX',''     ,'WINDR');
NearshoreParms       = char   ('WLN'  ,'WSN'  ,'TPN'  ,'HSN'  ,'THN'  ,'WRN'  ,'TPO'
);
NearshoreIsRichting  = logical([ 0       0       0       0       1       1       0
]);
NearshoreDbFields    = char   ('WATDW',''     ,'TPM'  ,'HSIGN','GOLFR',''
,'DZTPX');

if NieuweGolfFormules,
   NearshoreDbFields = char('WATDW',''     ,'TM-10','HSIGN','GOLFR',''     ,'DZTPX');
end
if PiekPeriode
  i=strmatch('DZTPX',OffshoreDbFields);
  OffshoreDbFields(i,:)='DZTP ';
  i=strmatch('DZTPX',NearshoreDbFields);
  NearshoreDbFields(i,:)='DZTP ';
end

ParmRanges           = [[-2,10];[0,80];[-0.01,40];[-0.01,20];[0,2*pi];[0,2*pi];[-
0.01,40]];
ParmUnits            = char(...
                            '[m]+NAP',...
                            '[m/s]',...
                            '[s]',...
                            '[m]',...
                            '[graden]',...
                            '[graden]',...
                            '[s]');
ParmDescriptions     = char(...
                            'Waterstand',...
                            'Windsnelheid',...
                            'Piekperiode',...
                            'Significante golfhoogte',...
                            'Golfrichting',...
                            'Windrichting',...
                            'Piekperiode2');
NrOfOffshoreParms    = length(cellstr(OffshoreParms));
NrOfNearshoreParms   = length(cellstr(NearshoreParms));


Parm= struct('Code',{''},'IsRichting',{''},
'DbName',{''},'Description',{''},'Unit',{''},'Range',{[NaN,NaN]},'Value',{NaN});

for i=1:NrOfOffshoreParms
        Model.PossibleOffshoreParameters(i)             = Parm;
        Model.PossibleOffshoreParameters(i).Code        = OffshoreParms(i,:);
        Model.PossibleOffshoreParameters(i).IsRichting  = OffshoreIsRichting(i);
        Model.PossibleOffshoreParameters(i).DbName      = OffshoreDbFields(i,:);
        Model.PossibleOffshoreParameters(i).Value       = NaN;
        Model.PossibleOffshoreParameters(i).Range       = ParmRanges(i,:);
        Model.PossibleOffshoreParameters(i).Unit        = ParmUnits(i,:);
        Model.PossibleOffshoreParameters(i).Description =
[deblank(ParmDescriptions(i,:)),' offshore'];
end
for i=1:NrOfNearshoreParms
        Model.PossibleNearshoreParameters(i)            = Parm;
        Model.PossibleNearshoreParameters(i).Code       = NearshoreParms(i,:);
        Model.PossibleNearshoreParameters(i).IsRichting = NearshoreIsRichting(i);
        Model.PossibleNearshoreParameters(i).DbName     = NearshoreDbFields(i,:);
        Model.PossibleNearshoreParameters(i).Value      = NaN;
        Model.PossibleNearshoreParameters(i).Range      = ParmRanges(i,:);
```

```
        Model.PossibleNearshoreParameters(i).Unit       = ParmUnits(i,:);
   Model.PossibleNearshoreParameters(i).Description     =
[deblank(ParmDescriptions(i,:)),' nearshore'];
end

Model.NrOfPossibleOffshoreParameters.Value      = NrOfOffshoreParms;
Model.NrOfPossibleNearshoreParameters.Value     = NrOfNearshoreParms;
```

Transformeer/Offshore2Nearshore

```
***************
*** 1,9 ****
! function [NearshoreData] = ...
!    Offshore2Nearshore ( Regio, X,Y, correctieRVB, OffshoreData, StroomCor,
faalMechID)

  %OFFSHORE2NEARSHORE
  %  AANROEP
! %     [NearshoreData] = Offshore2Nearshore (Regio, X,Y, correctieRVB, OffshoreData,
StroomCor, faalMechID)
  %  INVOER
  %     Regio:         string met naam van de regio
  %     X:             X-coordinaat van de lokatie
--- 1,10 ----
! function [NearshoreData,WaarnStormIndex] = ...
!    Offshore2Nearshore ( Regio, X,Y, correctieRVB, OffshoreData, StroomCor,
faalMechID,WaarnStormIndex)

  %OFFSHORE2NEARSHORE
  %  AANROEP
! %     [NearshoreData] = Offshore2Nearshore (Regio, X,Y, correctieRVB, OffshoreData,
StroomCor,
! %                                           faalMechID,WaarStormIndex)
  %  INVOER
  %     Regio:            string met naam van de regio
  %     X:                X-coordinaat van de lokatie
***************
*** 14,19 ****
--- 15,21 ----
  %                       Significante golfhoogte, Golfrichting(radialen) en
  %                       Windrichting (radialen) en (voor de Oosterschelde)
  %                       een parameter voor de noodsluiting
+ %     WaarnStormIndex:  de index van de storm behorend bij de waarneming
  %  UITVOER
  %     NearshoreData:    De naar nearshore getransformeeerde offshoredata, een 6-
koloms
  %                       matrix met resp. Waterstand, Windsnelheid, Piekperiode,
***************
*** 112,118 ****
     %sWNSfunctie= [sWNSfunctie '(OffshoreData,MIJNMODEL.X,MIJNMODEL.Y)'];

     [SwanInputData] =
WLOffshore2Swan(OffshoreData,MIJNMODEL.X,MIJNMODEL.Y,Regio,correctieRVB);
!
     NearshoreData = transformeerNaarNearshore (MIJNMODEL,SwanInputData);

  %%%catch
--- 114,122 ----
     %sWNSfunctie= [sWNSfunctie '(OffshoreData,MIJNMODEL.X,MIJNMODEL.Y)'];

     [SwanInputData] =
WLOffshore2Swan(OffshoreData,MIJNMODEL.X,MIJNMODEL.Y,Regio,correctieRVB);
!    if nargin>7 & stormFlanken %WaarnStormIndex beschikbaar en stormFlanken
!
[SwanInputData,WaarnStormIndex]=VoegtoeStormFlanken(SwanInputData,WaarnStormIndex);
!    end
     NearshoreData = transformeerNaarNearshore (MIJNMODEL,SwanInputData);

  %%%catch
***************
--- 135,157 ----
+ function
[SwanInputData,WaarnStormIndex]=VoegtoeStormFlanken(SwanInputData,WaarnStormIndex)
```

```
+ flankduur=stormFlanken;
+
+ Start=cat(1,1,find(diff(WaarnStormIndex))+1,length(WaarnStormIndex)+1);
+ nstorms=length(Start)-1;
+ WaarnStormIndex(end+(1:(2*flankduur*nstorms)))=NaN;
+ SwanInputData(end+(1:(2*flankduur*nstorms)),:)=NaN;
+ verloopBegin=ctranspose(0:flankduur)/flankduur;
+ verloopEind=flipud(verloopBegin);
+ for storm = nstorms:-1:1
+    Storm=Start(storm):Start(storm+1)-1;
+    ShiftedStorm=Storm+2*flankduur*(storm-1)+flankduur;
+    NewStorm=(Start(storm):(Start(storm+1)+2*flankduur-1))+2*flankduur*(storm-1);
+    %
+    SwanInputData(ShiftedStorm,:)=SwanInputData(Storm,:);
+    SwanInputData(ShiftedStorm(end):NewStorm(end),1:2) =
verloopEind*SwanInputData(ShiftedStorm(end),1:2);
+    SwanInputData(ShiftedStorm(end)+1:NewStorm(end),3:7) =
repmat(SwanInputData(ShiftedStorm(end),3:7),flankduur,1);
+    SwanInputData(NewStorm(1):ShiftedStorm(1),1:2) =
verloopBegin*SwanInputData(ShiftedStorm(1),1:2);
+    SwanInputData(NewStorm(1):ShiftedStorm(1)-1,3:7) =
repmat(SwanInputData(ShiftedStorm(1),3:7),flankduur,1);
+    %
+    WaarnStormIndex(NewStorm)=WaarnStormIndex(Storm(1));
+ end
```

Transformeer/RichtingTransformArray

```
function T = RichtingTransformArray ( ...
   Rand2001, uniekeWaardenIndex, uniekeWaarden, aantalUniekeWaarden, ind )

%RICHTINGTRANSFORMARRAY    opbouwen van deel transformatiematrix van een
%                          combinatie van golfrichting en windrichting
%  AANROEP
%     T = RichtingTransformArray ( ...
%         Rand2001, uniekeWaardenIndex, uniekeWaarden, aantalUniekeWaarden, ind)
%  INVOER
%     Rand2001:                          gegevens uit Rand2001
%     uniekeWaarden:       matrix met de unieke waarden voor die offshore
%                          parameters die geen richting zijn
%     uniekeWaardenIndex:  bevat de indexen in matrix uniekeWaarden
%                          voor 1 richting (de richting is in de aanroep bepaald)
%                          dus de code:
%                             for j=1:length(uniekeWaardenIndex)
%                                k = uniekeWaardenIndex(j,:)
%                                for l=1:length(k),
%                                   x(j,l) = uniekeWaarden{l}(k(l))
%                                end
%                             end
%                          levert een x op die weer bestaat uit de
%                          offshoreparameterwaarden die geen hoeken zijn
%     aantalUniekeWaarden: aantal unieke waarden
%     ind:                                          index
%  UITVOER
%     T:                                            transformatiematrix

% EXTERNE FUNCTIES
%     ConExtr
%     LinExtr

% Copyright © 2000 Rijkswaterstaat/RIKZ All Rights Reserved.

% Auteur: Abe Hoekstra/Ingrid Lammers, HKV lijn in water

% $Header: /PR/313/HYDRA_K/Transformeer/RichtingTransformArray.m 2     14-08-00 8:57
Hoekstra $
% $NoKeywords: $

% lengte van het aantal unieke combinaties significante golfhoogte, piekperiode,
waterstand en windsnelheid
% bij een gegeven richting
aantalCombinaties = size(uniekeWaardenIndex,1);
```

```
% aantal offshoreparameters waarvoor unieke waarden zijn (dit bepaalt de lengte van
T)
AantalOffshore = length(aantalUniekeWaarden(:));
% aantal nearshoreparameters die een richting zijn (in dit geval 2: de golfrichting
en de windrichting)
AantalNearshore = Rand2001.AantalNearshorePars;

% aantal nearshoreparameters die geen hoek zijn
aantalNsParsGeenHoek = sum(~Rand2001.NearShoreIsRichting);
% aantal nearshoreparameters die wel hoek zijn
aantalNsParsWelHoek = sum(Rand2001.NearShoreIsRichting);
% aantal offshoreparameters die geen hoek zijn
aantalOsParsGeenHoek = sum(~Rand2001.OffShoreIsRichting);
% selecteren van de ondergrenzen van de parameters die geen hoek zijn
% dit wordt later gebruikt voor de logtransformatie voor extrapolatie
LogTransOff=find(Rand2001.OffShoreOndergrens(~Rand2001.OffShoreIsRichting)>=0);
LogTransNear= find(Rand2001.NearShoreOndergrens(~Rand2001.NearShoreIsRichting)>=0);

% voor alle offshoreparameters die geen hoek zijn (dat zijn er 4) worden de
nearshorewaarden geinterpoleerd
% op basis van punt 2 en 7 uit paragraaf 3.2 van het functioneel ontwerp
for j=1:AantalOffshore,
   subUniekeWaardenIndex{j} = unique([1;uniekeWaardenIndex(:,j);
aantalUniekeWaarden(j)]);
   NrOfSub(j)=length(subUniekeWaardenIndex{j});
   b = uniekeWaarden{j};
   SubuniekeWaarden{j} = b(subUniekeWaardenIndex{j});
end

SubI = repmat(1,aantalCombinaties,AantalOffshore);
for j=1:AantalOffshore,
   b = subUniekeWaardenIndex{j};
   if length(b)==3
      SubI(:,j) = 2;
   else
      SubI(:,j) = interp1(b(2:end-1),[2:NrOfSub(j)-
1]',uniekeWaardenIndex(:,j),'nearest');
   end
end

% het definieren van structures
Entries = Rand2001.tabel(ind,Rand2001.AantalOffshorePars+1:end);

%
% NOTE: The first two (non-directional) offshore parameters are treated
% here differently from the other parameters (although there have been only
% two offshore parameters so far).
%
M = repmat(NaN,[NrOfSub(1:2) AantalNearshore]);
for j=1:aantalCombinaties,
   S.type = '()';
   S.subs = num2cell(SubI(j,1:2));
   S.subs{3} = ':';
   M = subsasgn(M,S,Entries(j,:));
end

S.type = '()';
S.subs{1} = ':';
S.subs{2} = ':';
for i=3:AantalOffshore
   N = repmat(NaN,[NrOfSub(1:i) AantalNearshore]);
   for j = 2:NrOfSub(i)-1,
      S.subs{i-1} = ':';
      S.subs{i} = j;
      S.subs{i+1} = ':';
      N = subsasgn(N,S,M);
   end
   M=N;
end

% Extrapolatie naar alle mogelijke waarden over alle richtingen in het
nearshoregebied
% dit is de toepassing van punt 6 uit paragraaf 3.2 van het functioneel ontwerp

M = reshape(M, [prod(NrOfSub), AantalNearshore]);
```

```
MDir =[];
if aantalNsParsWelHoek > 0
   MDir = reshape(M(:,Rand2001.NearShoreIsRichting),[NrOfSub aantalNsParsWelHoek ]);
end

MNoDir = reshape(M(:,~Rand2001.NearShoreIsRichting),[NrOfSub aantalNsParsGeenHoek]);

% extrapolatie hoge waarden
for j=1:aantalOsParsGeenHoek,
   MDir = ConExtr(MDir,j,'up');
   MNoDir=LinExtr(MNoDir,j,SubuniekeWaarden{j},'up');
end

MNoDir = reshape(MNoDir,[prod(NrOfSub),aantalNsParsGeenHoek]);

% logtransformatie voor de lage nearshorehoogwaterstanden en windsnelheden
% zie punt 5 uit paragraaf 3.2 van het functioneel ontwerp
if ~isempty(LogTransNear)
   for j=LogTransNear(:),
      MNoDir(:,j) = log(max(0.1,MNoDir(:,j)));
   end
end
% logtransformatie voor de lage offshorehoogwaterstanden en windsnelheden
% zie punt 5 uit paragraaf 3.2 van het functioneel ontwerp
if ~isempty(LogTransOff)
   for j=1:length(LogTransOff(:)),
      SubuniekeWaarden{LogTransOff(j)} =
log(max(0.1,SubuniekeWaarden{LogTransOff(j)}));
      uniekeWaarden{LogTransOff(j)} = log(max(0.1,uniekeWaarden{LogTransOff(j)}));
   end
end

MNoDir = reshape(MNoDir,[NrOfSub,aantalNsParsGeenHoek]);

% extrapolatie lage waarden
for j=1:aantalOsParsGeenHoek,
   MDir = ConExtr(MDir,j,'down');
   MNoDir = LinExtr(MNoDir,j,SubuniekeWaarden{j},'down');
end
MNoDir = reshape(MNoDir,[prod(NrOfSub),aantalNsParsGeenHoek]);
MDir = reshape(MDir,[prod(NrOfSub),aantalNsParsWelHoek]);
M(:,Rand2001.NearShoreIsRichting) = MDir;
M(:,~Rand2001.NearShoreIsRichting) = MNoDir;
M = reshape(M,[NrOfSub,AantalNearshore]);

% herordening nieuwe waarden
SubuniekeWaarden = SubuniekeWaarden(find(NrOfSub>1));
uniekeWaarden = uniekeWaarden(find(NrOfSub>1));
aantalUniekeWaarden = aantalUniekeWaarden(find(NrOfSub>1));
NrOfSub = NrOfSub(find(NrOfSub>1));
OutGrid = MijnNdgrid(uniekeWaarden(:));

M=squeeze(M);
M = reshape(M,prod(NrOfSub),AantalNearshore);

T = repmat(NaN,prod(aantalUniekeWaarden),AantalNearshore);
P = repmat(NaN,NrOfSub,1);
q = cell(1,1);

% voor alle nearshoreparameters interpolatie
for j=1:AantalNearshore,
   P(:) = M(:,j);
   % voor alle nearshoreparameters die geen hoek zijn interpolatie
   if ~Rand2001.NearShoreIsRichting(j)
      q{1} = P;
      R = mijnInterpn([SubuniekeWaarden(:); q; OutGrid(:)]);
      T(:,j) = R(:);
   else
      % voor alle nearshoreparamters die wel een hoek zijn interpolatie door
uitsplitsing in sinus en cosinus en
      % interpolatie vande arctangens; zie punt 3 uit paragraaf 3.2 vanhet
functioneel ontwerp
      q{1} = cos(P);
      R = mijnInterpn([SubuniekeWaarden(:); q; OutGrid(:)]);
      q{1} = sin(P);
```

```
      S = mijnInterpn([SubuniekeWaarden(:); q; OutGrid(:)]);
      T(:,j) = normaliseerHoek(atan2(S(:),R(:)));
   end
end
% terugtransformeren van de logaritme
T(:,LogTransNear) = exp(T(:,LogTransNear));

Bound = repmat(Rand2001.NearShoreBovengrens,prod(aantalUniekeWaarden),1);
p = find(T>Bound);
T(p) = Bound(p);

Bound = repmat(Rand2001.NearShoreOndergrens,prod(aantalUniekeWaarden),1);
p = find(T<Bound);
T(p) = Bound(p);

% het transformatiearray
T= T(:);
```

Transformeer/TransformeerNaarNearshore

```
***************
*** 82,84 ****
--- 82,112 ----
        NearshoreData(:,mijnModel.Rand2001.NearshoreIndex(j))=
MijnInterpn([waarden(:); q; OffshoreData(:)]);
     end
  end
+
+ %
+ % Bert Jagers (WL):
+ %
+ % Normaal gesproken zouden we nu klaar zijn, maar de correlatie met TPO zit
+ % er nog niet goed in. Binnen de numerieke nauwkeurigheid geldt nu dat
+ % (met sz3=size(mijnModel.Rand2001.Transf.matrix,3); )
+ % repmat(mijnModel.Rand2001.Transf.matrix(:,:,1,:,:),[1 1 sz3 1
1])==mijnModel.Rand2001.Transf.matrix
+ % [[Dit is te kwantificeren door te kijken naar het maximum van de absolute
+ %   waarde van het relatieve verschil: max(abs(verschil./matrix)).       ]]
+ % ofwel de Nearshore data hangt niet af van de derde offshore parameter en
+ % dat is TPO.
+ %
+ % Daarom bepalen we op basis van de correlatie van TPO met WSO nu de
+ % piekperiode die effectief gebruikt is in de transformatie. Uit de
+ % verhouding van deze fictieve TPO waarden (SWAN input) en de
+ % waargenomen TPO waarden, worden de berekende nearshore TPN waarden
+ % gecorrigeerd.
+ %
+
+ if PiekPeriode
+    TPOo = find(strcmp(mijnModel.Rand2001.OffshoreParameters,'TPO'));
+    TPOn = find(strcmp(mijnModel.Rand2001.NearshoreParameters,'TPO'));
+    TPN = find(strcmp(mijnModel.Rand2001.NearshoreParameters,'TPN'));
+    tpo = mijnModel.Rand2001.NearshoreIndex(TPOn);
+    tpn = mijnModel.Rand2001.NearshoreIndex(TPN);
+    NearshoreData(:,tpn) =
NearshoreData(:,tpn).*OffshoreData{TPOo}./NearshoreData(:,tpo);
+ end
```

Transformeer/VulTransFormMatrix

```
***************
*** 42,56 ****


  % converteer hoeken van graden naar radialen voor de betreffende
nearshoreparameters (golfrichting en windrichting)
! for i = find(Rand2001.NearShoreIsRichting)
!    index = length(Rand2001.OffShoreIsRichting)+i;
     Rand2001.tabel(:,index) = Rand2001.tabel(:,index).*(pi/180);
- end
```

```
  % de relatieve significante golfhoogte bepalen door de significante golfhoogte te
delen door de lokale diepte
  % als het te ondiep is neem dan voor de lokale diepte 0.1
! Rand2001.tabel(:,find(strcmp(Rand2001.Parameters,'HSN'))) = ...
!    Rand2001.tabel(:,find(strcmp(Rand2001.Parameters,'HSN'))) ./max(LokaleDiepte
,0.1);

  % bepaal de unieke waarden van elke offshore parameter en vul deze
  % aan met de boven- en ondergrenswaarden
--- 42,54 ----


  % converteer hoeken van graden naar radialen voor de betreffende
nearshoreparameters (golfrichting en windrichting)
! index = aantalOffshorePars+find(Rand2001.NearShoreIsRichting);
  Rand2001.tabel(:,index) = Rand2001.tabel(:,index).*(pi/180);

  % de relatieve significante golfhoogte bepalen door de significante golfhoogte te
delen door de lokale diepte
  % als het te ondiep is neem dan voor de lokale diepte 0.1
! hsn = find(strcmp(Rand2001.Parameters,'HSN'));
! Rand2001.tabel(:,hsn) = Rand2001.tabel(:,hsn) ./max(LokaleDiepte ,0.1);

  % bepaal de unieke waarden van elke offshore parameter en vul deze
  % aan met de boven- en ondergrenswaarden
***************
*** 87,93 ****
  % voor alle offshoreparameters
  for j = 1:aantalOffshorePars,
     % voor de offshoreparameter worden ontbrekende waarden geinterpoleerd op die
indices
!    % die voor overige offshoreparamters wel een unieke waarde hebben
     uniekeWaardenIndex(:,j) = interp1( ...
        uniekeWaarden{j}(2:end-1), ...
        [2:aantalUniekeWaarden(j)-1]',...
--- 85,91 ----
  % voor alle offshoreparameters
  for j = 1:aantalOffshorePars,
     % voor de offshoreparameter worden ontbrekende waarden geinterpoleerd op die
indices
!    % die voor overige offshoreparameters wel een unieke waarde hebben
     uniekeWaardenIndex(:,j) = interp1( ...
        uniekeWaarden{j}(2:end-1), ...
        [2:aantalUniekeWaarden(j)-1]',...
***************
*** 97,115 ****

  end

  % maak matrix met alle mogelijke combinaties van de indexnummers
  % van die parameters die richtingen zijn (golfrichting en windrichting)
! indexCombinaties= Combinaties(indexNummers(find(Rand2001.OffShoreIsRichting)));
  indexCombinaties = [indexCombinaties{:}];
  lGrid = prod(size(indexCombinaties));

! n = prod([aantalUniekeWaarden(find(~Rand2001.OffShoreIsRichting))
aantalNearshorePars]);
  T= repmat(NaN,[n lGrid]);

- isD = find(Rand2001.OffShoreIsRichting);
- isNoD = find(~Rand2001.OffShoreIsRichting);
  j=0;
! % voor elke combinatie van indexnummers van die paramters die richtingen zijn wordt
de functie
  % RichtingTransformarray aangeroepen
  for DirNr = indexCombinaties',
     DirNr = DirNr';
--- 95,114 ----

  end

+ isD = find(Rand2001.OffShoreIsRichting);
+ isNoD = find(~Rand2001.OffShoreIsRichting);
+
  % maak matrix met alle mogelijke combinaties van de indexnummers
```

```
  % van die parameters die richtingen zijn (golfrichting en windrichting)
! indexCombinaties= Combinaties(indexNummers(isD));
  indexCombinaties = [indexCombinaties{:}];
  lGrid = prod(size(indexCombinaties));

! n = prod([aantalUniekeWaarden(isNoD) aantalNearshorePars]);
  T= repmat(NaN,[n lGrid]);

  j=0;
! % voor elke combinatie van indexnummers van die parameters die richtingen zijn
wordt de functie
  % RichtingTransformarray aangeroepen
  for DirNr = indexCombinaties',
     DirNr = DirNr';
```