Dikes Overtopping Kernel - Technical documentation

Generated by Doxygen 1.8.9.1

Tue Oct 31 2017 14:59:06

Contents

1	Mod	lules Inc	dex												1
	1.1	Module	es List					 	 	 	 	 		 	 1
2	Data	a Type I	ndex												3
	2.1	Class	List					 	 	 	 	 		 	 3
3	File	Index													5
	3.1	File Lis	st					 	 	 	 	 		 	 5
4	Mod	lule Dod	cumentatio	on											7
	4.1	dllover	topping Mo	odule Re	ference			 	 	 	 	 		 	 7
		4.1.1	Detailed	Descripti	on			 	 	 	 	 		 	 8
		4.1.2	Function	/Subrouti	ne Docı	umen	tation	 	 	 	 	 		 	 8
			4.1.2.1	calculat	eqo			 	 	 	 	 		 	 8
			4.1.2.2	calculat	eqof .			 	 	 	 	 		 	 g
			4.1.2.3	calculat	eqoj .			 	 	 	 	 		 	 10
			4.1.2.4	calczva	lue			 	 	 	 	 		 	 11
			4.1.2.5	geomet	ry_c_f			 	 	 	 	 		 	 12
			4.1.2.6	getlang	uage .			 	 	 	 	 		 	 13
			4.1.2.7	input_j_	f			 	 	 	 	 		 	 13
			4.1.2.8	omkeer	variantf			 	 	 	 	 		 	 14
			4.1.2.9	omkeer	variantj			 	 	 	 	 		 	 15
			4.1.2.10	setlang	uage .			 	 	 	 	 		 	 16
			4.1.2.11	validate	inputc			 	 	 	 	 		 	 16
			4.1.2.12	validate	inputf .			 	 	 	 	 		 	 17
			4.1.2.13	validate	inputj .			 	 	 	 	 		 	 18
			4.1.2.14	version	number			 	 	 	 	 		 	 18
		4.1.3	Variable	Documer	ntation			 	 	 	 	 		 	 19
			4.1.3.1	separat	ionchar	٠		 	 	 	 	 		 	 19
	4.2	factorr	noduleove	rtopping	Module	Refer	rence	 	 	 	 	 		 	 19
		4.2.1	Detailed	Descripti	on			 	 	 	 	 		 	 19
		400	Eupotion	(Cubrouti	no Dooi	ıman	tation								10

iv CONTENTS

		4.2.2.1	calculategammab	19
		4.2.2.2	calculategammabeta	20
		4.2.2.3	calculategammaf	20
		4.2.2.4	calculatetanalpha	21
4.3	formula	amoduleov	vertopping Module Reference	22
	4.3.1	Detailed	Description	23
	4.3.2	Function/	Subroutine Documentation	23
		4.3.2.1	adjustinfluencefactors	23
		4.3.2.2	calculateanglewaveattack	24
		4.3.2.3	calculatebreakerlimit	24
		4.3.2.4	calculatebreakerparameter	24
		4.3.2.5	calculatewavelength	25
		4.3.2.6	calculatewaveovertoppingdischarge	25
		4.3.2.7	calculatewaverunup	26
		4.3.2.8	calculatewavesteepness	27
		4.3.2.9	cubicroots	28
		4.3.2.10	isequalreal	28
		4.3.2.11	isequalzero	28
		4.3.2.12	realrootscubicfunction	29
		4.3.2.13	rootsdepressedcubic	29
		4.3.2.14	rootsgeneralcubic	30
4.4	geome	trymodule	overtopping Module Reference	31
	4.4.1	Detailed	Description	31
	4.4.2	Function/	Subroutine Documentation	31
		4.4.2.1	adjustnonhorizontalberms	31
		4.4.2.2	allocatevectorsgeometry	32
		4.4.2.3	basicgeometrytest	33
		4.4.2.4	calculatehorzdistance	33
		4.4.2.5	calculatehorzlengths	34
		4.4.2.6	calculatesegmentslopes	35
		4.4.2.7	checkcrosssection	35
		4.4.2.8	copygeometry	36
		4.4.2.9	deallocategeometry	37
		4.4.2.10	determinesegmenttypes	38
		4.4.2.11	initializegeometry	39
		4.4.2.12	mergesequentialberms	40
		4.4.2.13	removeberms	41
		4.4.2.14	removedikesegments	42
		4.4.2.15	splitcrosssection	42
4.5	mainm	oduleovert	topping Module Reference	43

CONTENTS

	4.5.1	Detailed	Description	44
	4.5.2	Function	Subroutine Documentation	44
		4.5.2.1	calculateovertopping	44
		4.5.2.2	calculateovertoppingnegativefreeboard	45
		4.5.2.3	calculateovertoppingsection	45
		4.5.2.4	calculatewaveovertopping	46
		4.5.2.5	checkinputdata	47
		4.5.2.6	checkmodelfactors	48
		4.5.2.7	interpolateresultssections	49
4.6	module	elogging M	lodule Reference	49
	4.6.1	Detailed	Description	50
	4.6.2	Variable I	Documentation	50
		4.6.2.1	maxfilenamelength	50
4.7	omkee	rvariantmo	odule Module Reference	50
	4.7.1	Detailed	Description	50
	4.7.2	Function	Subroutine Documentation	51
		4.7.2.1	checkifonberm	51
		4.7.2.2	iteratetogivendischarge	53
		4.7.2.3	iteratetogivendischargevalidprofile	54
	4.7.3	Variable I	Documentation	55
		4.7.3.1	minzberm	55
		4.7.3.2	toldikeheight	55
		4.7.3.3	toldischarge	55
4.8	overto	opinginterf	ace Module Reference	56
	4.8.1	Detailed	Description	56
	4.8.2	Variable I	Documentation	56
		4.8.2.1	varmodelfactorcriticalovertopping	56
4.9	overto	opingmess	ages Module Reference	56
	4.9.1	Detailed	Description	57
	4.9.2	Function/	Subroutine Documentation	57
		4.9.2.1	getformattoomanyberms	57
		4.9.2.2	getlanguage	57
		4.9.2.3	getmsg_calc_influence_roughness	57
		4.9.2.4	getmsg_calculategammaftohigh	57
		4.9.2.5	getmsg_calculategammaftolow	58
		4.9.2.6	getmsg_first_segment_berm	58
		4.9.2.7	getmsg_last_segment_berm	58
		4.9.2.8	getmsgremovenonhorizontalberm	58
		4.9.2.9	getovertoppingformat	58
		4.9.2.10	getovertoppingmessage	59

vi CONTENTS

		4.9.2.11	getovertoppingparameter	. 59
		4.9.2.12	setlanguage	. 60
	4.9.3	Variable [Documentation	. 60
		4.9.3.1	language	. 60
		4.9.3.2	maxmsg	. 60
		4.9.3.3	maxpar	. 60
4.10	typedef	initionsove	ertopping Module Reference	. 60
	4.10.1	Detailed [Description	. 61
	4.10.2	Variable [Documentation	. 61
		4.10.2.1	berm_max	. 61
		4.10.2.2	berm_min	. 62
		4.10.2.3	fb_max	. 62
		4.10.2.4	fb_min	. 62
		4.10.2.5	fn_max	. 62
		4.10.2.6	$fn_min \ \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots$. 62
		4.10.2.7	foreshore_max	. 62
		4.10.2.8	$foreshore_min \ \ldots \ \ldots$. 62
		4.10.2.9	$frunup1 \dots \dots$. 62
		4.10.2.10	frunup2	. 62
		4.10.2.11	frunup3	. 62
		4.10.2.12	fs_max	. 63
		4.10.2.13	$fs_min \ \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots$. 63
		4.10.2.14	margindiff	. 63
		4.10.2.15	margingrad	. 63
		4.10.2.16	mz2_max	. 63
		4.10.2.17	mz2_min	. 63
		4.10.2.18	rfactor_max	. 63
		4.10.2.19	rfactor_min	. 63
		4.10.2.20	slope_max	. 63
		4.10.2.21	slope_min	. 64
		4.10.2.22	xdiff_min	. 64
		4.10.2.23	z2_iter_max1	. 64
		4.10.2.24	z2_iter_max2	. 64
		4.10.2.25	z2_margin	. 64
4.11	waveru	nup Modul	le Reference	. 64
	4.11.1	Detailed [Description	. 64
	4.11.2	Function/	Subroutine Documentation	. 65
		4.11.2.1	convergedwithresidu	. 65
		4.11.2.2	determinestartingvalue	. 65
		4.11.2.3	findsmallestresidu	. 66

CONTENTS vii

			4.11.2.4	innercalculation	66
			4.11.2.5	iterationwaverunup	67
	4.12	zfunctio	onsovertop	pping Module Reference	68
		4.12.1	Detailed I	Description	69
		4.12.2	Function/	Subroutine Documentation	69
			4.12.2.1	adjustprofile	69
			4.12.2.2	calculateqorto	70
			4.12.2.3	profileinstructure	71
			4.12.2.4	zfunclogratios	72
5	Data	Type D	ocumenta	ation	75
•	5.1			ace::overtoppinggeometrytype Type Reference	7 5
	J. I	5.1.1		Description	75 76
		5.1.2		Data Documentation	76
		5.1.2	5.1.2.1	normal	76 76
			5.1.2.1	npoints	76 76
			5.1.2.3	roughness	76 76
			5.1.2.4	xcoords	76 76
			5.1.2.5	ycoords	76 76
	5.2	overtor		ace::overtoppinggeometrytypef Type Reference	76 77
	5.2	5.2.1		Description	77
		5.2.1		Data Documentation	77
		5.2.2	5.2.2.1	normal	77
			5.2.2.1	npoints	77
			5.2.2.3	roughness	77 78
				xcoords	78 78
			5.2.2.5	ycoords	78 78
	5.3	module		ogging Type Reference	78 78
	0.0	5.3.1		Description	78 78
		5.3.2		Data Documentation	70 79
		0.0.2	5.3.2.1	filename	79 79
			5.3.2.2	verbosity	79 79
	5.4	omkeei		dule::tomkeer Type Reference	79
	0.1	5.4.1		Description	80
		5.4.2		Data Documentation	80
		0.1.2	5.4.2.1	dischargeprofile	80
			5.4.2.2	isberm	80
			5.4.2.3	isvalidz	80
			5.4.2.4	zprofile	80
	5.5	typedet		ertopping::tpgeometry Type Reference	80
	0.0	.,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,			

viii CONTENTS

	5.5.1	Detailed	Description	82
	5.5.2	Member	Data Documentation	82
		5.5.2.1	nbermsegments	82
		5.5.2.2	ncoordinates	82
		5.5.2.3	psi	82
		5.5.2.4	roughnessfactors	82
		5.5.2.5	segmentslopes	82
		5.5.2.6	segmenttypes	82
		5.5.2.7	xcoorddiff	82
		5.5.2.8	xcoordinates	83
		5.5.2.9	ycoorddiff	83
		5.5.2.10	ycoordinates	83
5.6	typede	finitionsov	ertopping::tpload Type Reference	83
	5.6.1	Detailed	Description	84
	5.6.2	Member	Data Documentation	84
		5.6.2.1	$h \ldots \ldots \ldots \ldots \ldots \ldots$	84
		5.6.2.2	hm0	84
		5.6.2.3	phi	84
		5.6.2.4	tm_10	84
5.7	typede	finitionsov	ertopping::tpovertopping Type Reference	84
	5.7.1	Detailed	Description	85
	5.7.2	Member	Data Documentation	85
		5.7.2.1	qo	85
		5.7.2.2	z2	85
5.8	typede	finitionsov	ertopping::tpovertoppinginput Type Reference	86
	5.8.1	Detailed	Description	87
	5.8.2	Member	Data Documentation	87
		5.8.2.1	computedovertopping	87
		5.8.2.2	criticalovertopping	87
		5.8.2.3	factordeterminationq_b_f_b	87
		5.8.2.4	factordeterminationq_b_f_n	87
		5.8.2.5	fshallow	87
		5.8.2.6	m_z2	87
		5.8.2.7	reductionfactorforeshore	87
		5.8.2.8	relaxationfactor	88
5.9	overtop	opinginterf	ace::tpprofilecoordinate Type Reference	88
	5.9.1	Detailed	Description	88
	5.9.2	Member	Data Documentation	88
		5.9.2.1	roughness	89
		5.9.2.2	xcoordinate	89

CONTENTS

		5.9.2.3 zcoordinate	89
6	File I	ocumentation	91
	6.1	dllOvertopping.f90 File Reference	91
		6.1.1 Detailed Description	92
	6.2	factorModuleOvertopping.f90 File Reference	92
		6.2.1 Detailed Description	93
	6.3	formulaModuleOvertopping.f90 File Reference	93
		6.3.1 Detailed Description	94
	6.4	geometryModuleOvertopping.f90 File Reference	94
		6.4.1 Detailed Description	95
	6.5	mainModuleOvertopping.f90 File Reference	95
		6.5.1 Detailed Description	96
	6.6	ModuleLogging.f90 File Reference	96
		6.6.1 Detailed Description	96
	6.7	omkeerVariantModule.f90 File Reference	96
		6.7.1 Detailed Description	97
	6.8	overtoppingInterface.f90 File Reference	97
		6.8.1 Detailed Description	97
	6.9	OvertoppingMessages.f90 File Reference	97
		6.9.1 Detailed Description	98
	6.10	typeDefinitionsOvertopping.f90 File Reference	98
		6.10.1 Detailed Description	100
	6.11	waveRunup.f90 File Reference	100
		6.11.1 Detailed Description	100
	6.12	zFunctionsOvertopping.f90 File Reference	100
		6.12.1 Detailed Description	101
Ind	lex		103

Chapter 1

Modules Index

1.1 Modules List

Here is a list of all modules with brief descriptions:

dilovertopping	
Main entry for the dll DikesOvertopping	7
factormoduleovertopping	
Functions for the slope angle and influence factors	19
formulamoduleovertopping	
Core computations for Dikes Overtopping	22
geometrymoduleovertopping	
Core computations related to the geometry	31
mainmoduleovertopping	
Core computations for Dikes Overtopping	43
modulelogging	
Steering the extra logging	49
omkeervariantmodule	
Module for the 'omkeerVariant'	50
overtoppinginterface	
Module for the interface of dllOvertopping	56
overtoppingmessages	
Module for the messages in the overtopping dll, in Dutch or English	56
typedefinitionsovertopping	
Type definitions for Dikes Overtopping	30
waverunup	
Iteration procedure for 2% wave runup	34
zfunctionsovertopping	
Module for the Limit State Functions (Z-functions) for wave overtopping	36

2 **Modules Index**

Chapter 2

Data Type Index

2.1 Class List

Here are the data types with brief descriptions:

overtoppinginterface::overtoppinggeometrytype	75
overtoppinginterface::overtoppinggeometrytypef	77
modulelogging::tlogging	
TLogging: structure for steering the logging	78
omkeervariantmodule::tomkeer	79
typedefinitionsovertopping::tpgeometry	
TpGeometry: structure with geometry data	80
typedefinitionsovertopping::tpload	
TpLoad: structure with load parameters	83
typedefinitionsovertopping::tpovertopping	
TpOvertopping: structure with overtopping results	84
typedefinitionsovertopping::tpovertoppinginput	
OvertoppingModelFactors: C-structure with model factors	86
overtoppinginterface::tpprofilecoordinate	88

Data Type Index

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

dllOvertopping.t90	
Main entry for the dll DikesOvertopping FUNCTIONS/SUBROUTINES exported from dll← Overtopping.dll:	91
factorModuleOvertopping.f90	31
This file contains a module with functions for the slope angle and influence factors	92
formulaModuleOvertopping.f90	02
This file contains a module with the core computations for Dikes Overtopping	93
geometryModuleOvertopping.f90	
This file contains a module with the core computations for Dikes Overtopping related to the	
geometry	94
mainModuleOvertopping.f90	
This file contains a module with the core computations for Dikes Overtopping	95
ModuleLogging.f90	
Module for steering the extra logging	96
omkeerVariantModule.f90	
This file contains the omkeerVariant	96
overtoppingInterface.f90	
This file contains the parameters and types (structs) as part of the interface to and from dll←	
Overtopping	97
OvertoppingMessages.f90	
This file contains the messages in the overtopping dll, in Dutch or English	97
typeDefinitionsOvertopping.f90	
This file contains a module with the type definitions for Dikes Overtopping	98
waveRunup.f90	
This file contains a module with the iteration procedure for 2% wave runup	100
zFunctionsOvertopping.f90	
This file contains the limit state functions for wave overtopping within VTV	100

6 File Index

Chapter 4

Module Documentation

4.1 dllovertopping Module Reference

Main entry for the dll DikesOvertopping.

Functions/Subroutines

subroutine, public calculateqo (load, geometryInput, dikeHeight, modelFactors, overtopping, success, error
 —
 Text, verbosity, logFile)

Subroutine that calculates the discharge needed for the Z-function DikesOvertopping Wrapper for calculateQoF←: convert C-like input structures to Fortran input structures.

subroutine, public calculateqoj (load, xcoords, ycoords, roughness, normal, npoints, dikeHeight, model
 —
 Factors, output, succes, errorMessage)

Subroutine that calculates the discharge Wrapper for calculateQoF: convert basic types to Fortran input structures.

subroutine, public calculateqof (load, geometryF, dikeHeight, modelFactors, overtopping, success, errorText, logging)

Subroutine that calculates the discharge needed for the Z-function DikesOvertopping.

- subroutine, public calczvalue (criticalOvertoppingRate, modelFactors, Qo, z, success, errorMessage)
 - Subroutine that calculates the Z-function DikesOvertopping based on the discharge calculated with calculateQoF.
- · subroutine, public validateinputc (geometryInput, dikeHeight, modelFactors, success, errorText)
 - Subroutine that validates the geometry Wrapper for ValidateInputF: convert C-like input structures to Fortran input structures
- subroutine, public validateinputj (x, y, roughness, normal, nPoints, dikeHeight, modelFactorsArray, success, errorText)

Subroutine that validates the geometry Wrapper for ValidateInputF: convert basic types to Fortran input structures.

• subroutine, public validateinputf (geometryF, dikeHeight, modelFactors, errorStruct)

Subroutine that validates the geometry.

• subroutine, public omkeervariantj (load, xcoords, ycoords, roughness, normal, npoints, givenDischarge, dikeHeight, modelFactors, output, succes, errorMessage)

Wrapper for omkeerVariantF: get dikeHeight by given discharge.

 subroutine, public omkeervariantf (load, geometryF, givenDischarge, dikeHeight, modelFactors, overtopping, success, errorText, logging)

Subroutine with omkeerVariant (get dikeHeight by given discharge)

• subroutine, public setlanguage (lang)

Subroutine that sets the language for error and validation messages.

· subroutine, public getlanguage (lang)

Subroutine that gets the language for error and validation messages.

• subroutine, public versionnumber (version)

Subroutine that delivers the version number.

• type(overtoppinggeometrytypef) function geometry_c_f (geometryInput)

Private subroutine that converts geometry from c-pointer to fortran struct.

• subroutine input_j_f (x, y, roughness, normal, geometryF, modelFactorsJ, modelFactorsF, loadJ, loadF)

convert java input to Fortran input

Variables

• character, parameter separationchar = char(9)

use horizontal tab for separation

4.1.1 Detailed Description

Main entry for the dll DikesOvertopping.

4.1.2 Function/Subroutine Documentation

4.1.2.1 subroutine, public dllovertopping::calculateqo (type(tpload), intent(in) *load,* type(overtoppinggeometrytype), intent(in) *geometryInput,* real(kind=wp), intent(in) *dikeHeight,* type(tpovertoppinginput), intent(inout) *modelFactors,* type (tpovertopping), intent(out) *overtopping,* logical, intent(out) *success,* character(len=*), intent(out) *errorText,* integer, intent(in) *verbosity,* character(len=*), intent(in) *logFile*)

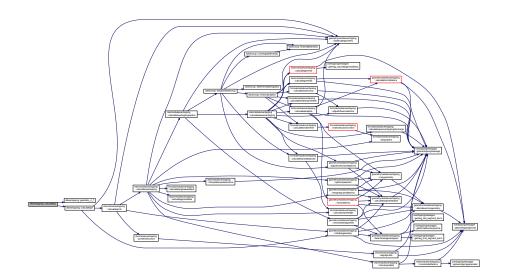
Subroutine that calculates the discharge needed for the Z-function DikesOvertopping Wrapper for calculateQoF: convert C-like input structures to Fortran input structures.

Parameters

in	geometryinput	struct with geometry and roughness as c-pointers
in	load	struct with waterlevel and wave parameters
in	dikeheight	dike height
in,out	modelfactors	struct with modelfactors
out	overtopping	structure with overtopping results
out	success	flag for success
out	errortext	error message (only set if not successful)
in	verbosity	level of verbosity
in	logfile	filename of logfile

Definition at line 71 of file dllOvertopping.f90.

Here is the call graph for this function:



4.1.2.2 subroutine, public dllovertopping::calculateqof (type(tpload), intent(in) *load*, type(overtoppinggeometrytypef), intent(in) *geometryF*, real(kind=wp), intent(in) *dikeHeight*, type(tpovertoppinginput), intent(inout) *modelFactors*, type (tpovertopping), intent(out) *overtopping*, logical, intent(out) *success*, character(len=*), intent(out) *errorText*, type(tlogging), intent(in) *logging*)

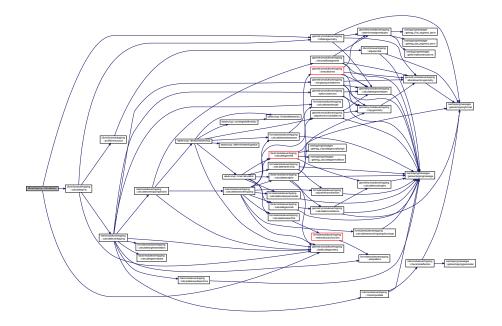
Subroutine that calculates the discharge needed for the Z-function DikesOvertopping.

Parameters

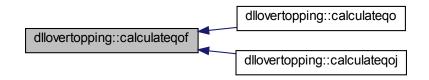
in	geometryf	struct with geometry and roughness
in	load	struct with waterlevel and wave parameters
in	dikeheight	dike height
in,out	modelfactors	struct with modelFactors
out	overtopping	structure with overtopping results
out	success	flag for success
out	errortext	error message (only set if not successful)
in	logging	logging struct

Definition at line 144 of file dllOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.1.2.3 subroutine, public dllovertopping::calculateqoj (real(kind=wp), dimension(4), intent(in) *load,* real(kind=wp), dimension(npoints), intent(in) *xcoords,* real(kind=wp), dimension(npoints-1), intent(in) *roughness,* real(kind=wp), intent(in) *normal,* integer, intent(in) *npoints,* real(kind=wp), intent(in) *dikeHeight,* real(kind=wp), dimension(8), intent(in) *modelFactors,* real(kind=wp), dimension(2), intent(out) *output,* logical, intent(out) *succes,* character(len=256), intent(out) *errorMessage*)

Subroutine that calculates the discharge Wrapper for calculateQoF: convert basic types to Fortran input structures.

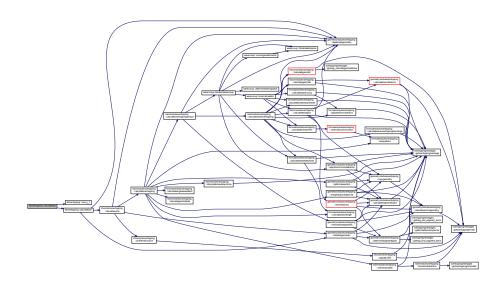
Parameters

in	npoints	the number of coordinates	
in	dikeheight	dike height	
in	normal	dike normal	
out	succes	flag for success	
out	errormessage	error message (only set if not successful)	

in	load	input load (wl and 3 wave numbers)
in	xcoords	the x coordinates
in	ycoords	the y coordinates
in	roughness	the roughness at sections
in	modelfactors	the overtopping modelfactors
out	output	output array: 1: Qo; 2: Z2

Definition at line 103 of file dllOvertopping.f90.

Here is the call graph for this function:



4.1.2.4 subroutine, public dllovertopping::calczvalue (real(kind=wp), intent(in) *criticalOvertoppingRate*, type(tpovertoppinginput), intent(inout) *modelFactors*, real(kind=wp), intent(in) *Qo*, real(kind=wp), intent(out) *z*, logical, intent(out) *success*, character(len=*), intent(out) *errorMessage*)

Subroutine that calculates the Z-function DikesOvertopping based on the discharge calculated with calculateQoF.

Parameters

in	criticalovertop-	critical overtoppingrate	
	pingrate		
in,out	modelfactors	struct with modelfactors	
in	qo	calculated discharge	
out	Z	z value	
out	errormessage	error message (only if not successful)	
out	success	flag for success	

Definition at line 175 of file dllOvertopping.f90.

Here is the call graph for this function:



4.1.2.5 type(overtoppinggeometrytypef) function dllovertopping::geometry_c_f (type(overtoppinggeometrytype), intent(in) geometryInput) [private]

Private subroutine that converts geometry from c-pointer to fortran struct.

Parameters

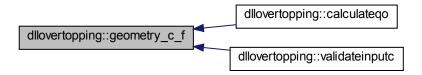
in	geometryinput	struct with geometry and roughness as c-pointers
----	---------------	--------------------------------------------------

Returns

fortran struct with geometry and roughness

Definition at line 501 of file dllOvertopping.f90.

Here is the caller graph for this function:



4.1.2.6 subroutine, public dllovertopping::getlanguage (character(len=*), intent(out) lang)

Subroutine that gets the language for error and validation messages.

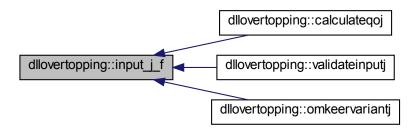
Definition at line 467 of file dllOvertopping.f90.

4.1.2.7 subroutine dllovertopping::input_j_f (real(kind=wp), dimension(:), intent(in) x, real(kind=wp), dimension(:), intent(in) y, real(kind=wp), dimension(:), intent(in) roughness, real(kind=wp), intent(in) normal, type(overtoppinggeometrytypef), intent(out) geometryF, real(kind=wp), dimension(:), intent(in) modelFactorsJ, type(tpovertoppinginput), intent(out) modelFactorsF, real(kind=wp), dimension(:), intent(in), optional loadJ, type(tpload), intent(out), optional loadF)

convert java input to Fortran input

Definition at line 524 of file dllOvertopping.f90.

Here is the caller graph for this function:



4.1.2.8 subroutine, public dllovertopping::omkeervariantf (type(tpload), intent(in) *load*, type(overtoppinggeometrytypef), intent(in) *geometryF*, real(kind=wp), intent(in) *givenDischarge*, real(kind=wp), intent(out) *dikeHeight*, type(tpovertoppinginput), intent(inout) *modelFactors*, type (tpovertopping), intent(inout) *overtopping*, logical, intent(out) *success*, character(len=*), intent(out) *errorText*, type(tlogging), intent(in) *logging*)

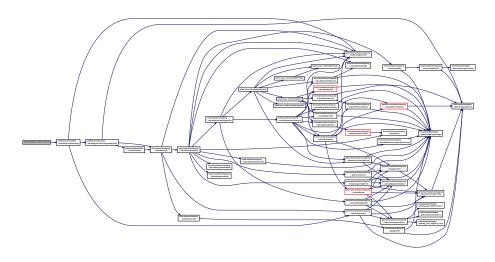
Subroutine with omkeerVariant (get dikeHeight by given discharge)

Parameters

in	geometryf	struct with geometry and roughness
in	load	struct with waterlevel and wave parameters
in	givendischarge	discharge to iterate to
out	dikeheight	dike height
in,out	modelfactors	struct with modelFactors
in,out	overtopping	structure with overtopping results
out	success	flag for success
out	errortext	error message (only set if not successful)
in	logging	logging struct

Definition at line 431 of file dllOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.1.2.9 subroutine, public dllovertopping::omkeervariantj (real(kind=wp), dimension(4), intent(in) *load,* real(kind=wp), dimension(npoints), intent(in) *xcoords,* real(kind=wp), dimension(npoints), intent(in) *ycoords,* real(kind=wp), dimension(npoints-1), intent(in) *roughness,* real(kind=wp), intent(in) *normal,* integer, intent(in) *nopints,* real(kind=wp), intent(in) *givenDischarge,* real(kind=wp), intent(out) *dikeHeight,* real(kind=wp), dimension(8), intent(in) *modelFactors,* real(kind=wp), dimension(2), intent(out) *output,* logical, intent(out) *succes,* character(len=256), intent(out) *errorMessage*

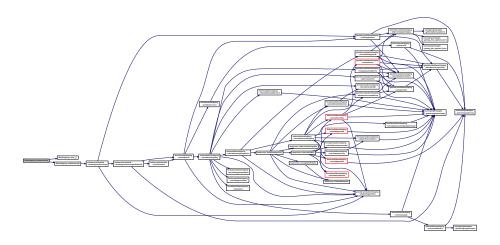
Nrapper for	omkoor\/	/ariantF∙ c	at dikaHa	iaht hy i	aivan die	charge

Parameters

in	npoints	number of coordinates	
in	givendischarge	input discharge	
out	dikeheight	dike height	
in	normal	dike normal	
out	succes	flag for success	
out	errormessage	error message (only set if not successful)	
in	load	input load (wl, and 3 wave parameters)	
in	xcoords	the x-coordinates	
in	ycoords	the y-coordinates	
in	roughness	the roughness at sections	
in	modelfactors	the overtopping modelfactors	
out	output	output array: 1: Z2; 2: Qo	

Definition at line 388 of file dllOvertopping.f90.

Here is the call graph for this function:



4.1.2.10 subroutine, public dllovertopping::setlanguage (character(len=*), intent(in) lang)

Subroutine that sets the language for error and validation messages.

Definition at line 454 of file dllOvertopping.f90.

4.1.2.11 subroutine, public dllovertopping::validateinputc (type(overtoppinggeometrytype), intent(in) *geometrylnput*, real(kind=wp), intent(in) *dikeHeight*, type(tpovertoppinginput), intent(inout) *modelFactors*, logical, intent(out) *success*, character(len=*), intent(out) *errorText*)

Subroutine that validates the geometry Wrapper for ValidateInputF: convert C-like input structures to Fortran input structures.

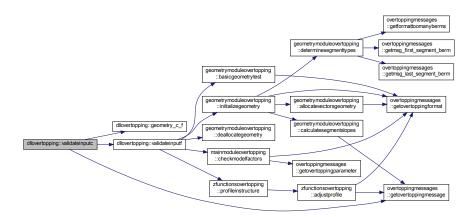
Parameters

in	geometryinput	struct with geometry and roughness as c-pointers
in	dikeheight	dike height
in,out	modelfactors	struct with modelfactors
out	success	flag for success

t-		away managan (anh) ant if not average (i)
out	errortext	error message (only set if not successful)

Definition at line 195 of file dllOvertopping.f90.

Here is the call graph for this function:



4.1.2.12 subroutine, public dllovertopping::validateinputf (type(overtoppinggeometrytypef), intent(in) *geometryF*, real(kind=wp), intent(in) *dikeHeight*, type(tpovertoppinginput), intent(inout) *modelFactors*, type(terrormessages), intent(inout) *errorStruct*)

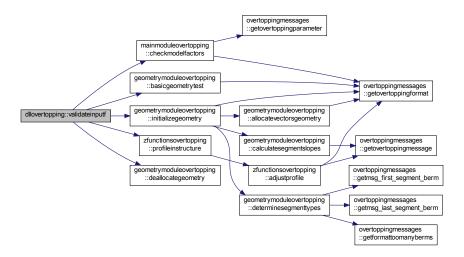
Subroutine that validates the geometry.

Parameters

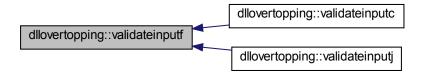
in	geometryf	struct with geometry and roughness
in	dikeheight	dike height
in,out	modelfactors	struct with modelFactors
in,out	errorstruct	error message (only set if not successful)

Definition at line 308 of file dllOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.1.2.13 subroutine, public dllovertopping::validateinputj (real(kind=wp), dimension(npoints), intent(in) x, real(kind=wp), dimension(npoints), intent(in) y, real(kind=wp), dimension(npoints-1), intent(in) roughness, real(kind=wp), intent(in) normal, integer, intent(in) nPoints, real(kind=wp), intent(in) dikeHeight, real(kind=wp), dimension(8), intent(in) modelFactorsArray, logical, intent(out) success, character(len=256), intent(out) errorText)

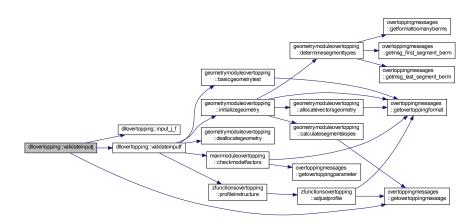
Subroutine that validates the geometry Wrapper for ValidateInputF: convert basic types to Fortran input structures.

Parameters

in	npoints	number of profile points
in	X	x coordinates of profile
in	У	y coordinates of profile
in	roughness	roughness of profile sections
in	normal	dike normal
in	dikeheight	dike height
in	modelfactorsar-	array with modelfactors
	ray	
out	success	flag for success
out	errortext	error message (only set if not successful)

Definition at line 246 of file dllOvertopping.f90.

Here is the call graph for this function:



4.1.2.14 subroutine, public dllovertopping::versionnumber (character(len=*), intent(out) version)

Subroutine that delivers the version number.

Parameters

out	version	version number
-----	---------	----------------

Definition at line 479 of file dllOvertopping.f90.

4.1.3 Variable Documentation

4.1.3.1 character, parameter dllovertopping::separationchar = char(9)

use horizontal tab for separation

Definition at line 61 of file dllOvertopping.f90.

4.2 factormoduleovertopping Module Reference

functions for the slope angle and influence factors

Functions/Subroutines

- subroutine, public calculatetanalpha (h, Hm0, z2, geometry, tanAlpha, succes, errorMessage)
 calculateTanAlpha representative slope angle
- subroutine, public calculategammabeta (Hm0, Tm_10, beta, gammaBeta_z, gammaBeta_o) calculateGammaBeta influence factor angle of wave attack
- subroutine, public calculategammaf (h, ksi0, ksi0Limit, gammaB, z2, geometry, gammaF, succes, error
 — Message)

calculateGammaF influence factor roughness

subroutine, public calculategammab (h, Hm0, z2, geometry, gammaB, succes, errorMessage)
 calculateGammaB influence factor berms

4.2.1 Detailed Description

functions for the slope angle and influence factors

4.2.2 Function/Subroutine Documentation

4.2.2.1 subroutine, public factormoduleovertopping::calculategammab (real(kind=wp), intent(in) *h*, real(kind=wp), intent(in) *Hm0*, real(kind=wp), intent(in) *z2*, type(tpgeometry), intent(in) *geometry*, real(kind=wp), intent(out) *gammaB*, logical, intent(out) *succes*, character(len=*), intent(out) *errorMessage*)

calculateGammaB influence factor berms

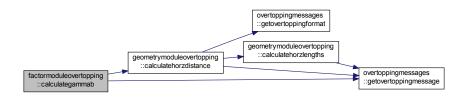
Parameters

in	h	local water level (m+NAP)
in	hm0	significant wave height (m)
in	z2	2% wave run-up (m)
in	geometry	structure with geometry data
out	gammab	influence factor berms

out	succes	flag for succes
out	errormessage	error message

Definition at line 317 of file factorModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.2.2.2 subroutine, public factormoduleovertopping::calculategammabeta (real(kind=wp), intent(inout) *Hm0*, real(kind=wp), intent(inout) *Tm_10*, real(kind=wp), intent(in) *beta*, real(kind=wp), intent(out) *gammaBeta_z*, real(kind=wp), intent(out) *gammaBeta_o*)

calculateGammaBeta influence factor angle of wave attack

Parameters

in,out	hm0	significant wave height (m)
in,out	tm_10	spectral wave period (s)
in	beta	angle of wave attack (degree)
out	gammabeta_z	influence factor angle of wave attack 2% wave run-up
out	gammabeta_o	influence factor angle of wave attack overtopping

Definition at line 139 of file factorModuleOvertopping.f90.

Here is the caller graph for this function:



4.2.2.3 subroutine, public factormoduleovertopping::calculategammaf (real(kind=wp), intent(in) h, real(kind=wp), intent(in) gammaB, real(kind=wp), intent(in) gammaB, real(kind=wp), intent(in) gammaB, real(kind=wp), intent(in) succes, character(len=*), intent(out) errorMessage)

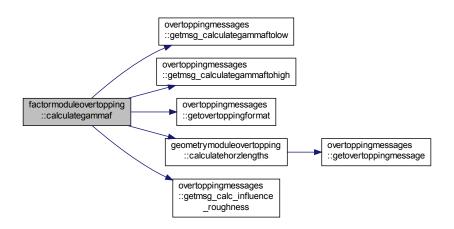
calculateGammaF influence factor roughness

Parameters

in	h	local water level (m+NAP)
in	ksi0	breaker parameter
in	ksi0limit	limit value breaker parameter
in	gammab	influence factor berms
in	z2	2% wave run-up (m)
in	geometry	structure with geometry data
out	gammaf	influence factor roughness
out	succes	flag for succes
out	errormessage	error message

Definition at line 179 of file factorModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.2.2.4 subroutine, public factormoduleovertopping::calculatetanalpha (real(kind=wp), intent(in) *h,* real(kind=wp), intent(in) *Hm0,* real(kind=wp), intent(in) *z2,* type(typeometry), intent(in) *geometry,* real(kind=wp), intent(out) *tanAlpha,* logical, intent(out) *succes,* character(len=*), intent(out) *errorMessage*)

calculateTanAlpha representative slope angle

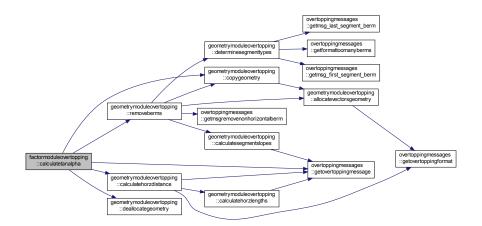
Parameters

in	h	local water level (m+NAP)
in	hm0	significant wave height (m)
in	z2	2% wave run-up (m)
in	geometry	structure with geometry data

out	tanalpha	representative slope angle
out	succes	flag for succes
out	errormessage	error message

Definition at line 60 of file factorModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.3 formulamoduleovertopping Module Reference

the core computations for Dikes Overtopping

Functions/Subroutines

• subroutine, public calculatewaverunup (Hm0, ksi0, ksi0Limit, gammaB, gammaF, gammaBeta, modelFactors, z2, succes, errorMessage)

calculateWaveRunup: calculate wave runup

• subroutine, public calculatewaveovertoppingdischarge (h, Hm0, tanAlpha, gammaB, gammaB, gammaBeta, ksi0, hCrest, modelFactors, Qo, succes, errorMessage)

calculateWaveOvertoppingDischarge: calculate the wave overtopping discharge

• subroutine, public calculatewavelength (Tm_10, L0)

calculateWaveLength: calculate the wave length

• subroutine, public calculatewavesteepness (Hm0, Tm_10, s0, succes, errorMessage)

calculateWaveSteepness: calculate the wave steepness

• subroutine, public calculatebreakerparameter (tanAlpha, s0, ksi0, succes, errorMessage)

calculateBreakerParameter: calculate the breaker parameter

• subroutine, public calculateanglewaveattack (phi, psi, beta)

calculateAngleWaveAttack: calculate the angle of wave attack

• subroutine, public calculatebreakerlimit (gammaB, ksi0Limit, succes, errorMessage)

calculateBreakerLimit: calculate the breaker limit

• subroutine, public adjustinfluencefactors (gammaB, gammaF, gammaBeta, gammaBetaType, ksi0, ksi0Limit, succes, errorMessage)

adjustInfluenceFactors: adjust the influence factors

• subroutine realrootscubicfunction (a, b, c, d, N, x, succes, errorMessage)

realRootsCubicFunction: calculate the roots of a cubic function

• subroutine rootsgeneralcubic (a, b, c, d, z, succes, errorMessage)

rootsGeneralCubic: calculate the roots of a generic cubic function

• subroutine rootsdepressedcubic (p, q, z)

rootsDepressedCubic: calculate the roots of a depressed cubic function

• subroutine cubicroots (z, roots)

cubicRoots: calculate the roots of a cubic function

• logical function, public isequalreal (x1, x2)

isEqualReal: are two reals (almost) equal

• logical function, public isequalzero (x)

isEqualZero: is a real (almost) zero

4.3.1 Detailed Description

the core computations for Dikes Overtopping

4.3.2 Function/Subroutine Documentation

4.3.2.1 subroutine, public formulamoduleovertopping::adjustinfluencefactors (real(kind=wp), intent(inout) gammaB, real(kind=wp), intent(inout) gammaF, real(kind=wp), intent(inout) gammaBeta, integer, intent(in) gammaBetaType, real(kind=wp), intent(in) ksi0, real(kind=wp), intent(in) ksi0Limit, logical, intent(out) succes, character(len=*), intent(out) errorMessage)

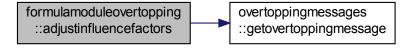
adjustInfluenceFactors: adjust the influence factors

Parameters

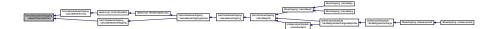
in,out	gammab	influence factor berms
in,out	gammaf	influence factor roughness
in,out	gammabeta	influence factor angle of wave attack
in	gammabetatype	type influence factor angle of wave attack: 1 = wave run-up, 2 = overtopping
in	ksi0	breaker parameter
in	ksi0limit	limit value breaker parameter
out	succes	flag for succes
out	errormessage	error message

Definition at line 403 of file formulaModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.3.2.2 subroutine, public formulamoduleovertopping::calculateanglewaveattack (real(kind=wp), intent(in) *phi*, real(kind=wp), intent(out) *beta*)

calculateAngleWaveAttack: calculate the angle of wave attack

Parameters

	in	phi	wave direction (degree)
ſ	in	psi	dike normal (degree)
ſ	out	beta	angle of wave attack (degree)

Definition at line 309 of file formulaModuleOvertopping.f90.

Here is the caller graph for this function:

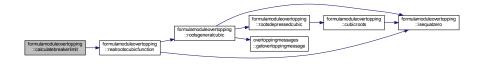


4.3.2.3 subroutine, public formulamoduleovertopping::calculatebreakerlimit (real(kind=wp), intent(in) *gammaB*, real(kind=wp), intent(out) *ksi0Limit*, logical, intent(out) *succes*, character(len=*), intent(out) *errorMessage*)

calculateBreakerLimit: calculate the breaker limit

Definition at line 332 of file formulaModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.3.2.4 subroutine, public formulamoduleovertopping::calculatebreakerparameter (real(kind=wp), intent(in) tanAlpha, real(kind=wp), intent(in) s0, real(kind=wp), intent(out) ksi0, logical, intent(out) succes, character(len=*), intent(out) errorMessage)

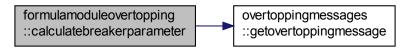
calculateBreakerParameter: calculate the breaker parameter

Parameters

in	tanalpha	representative slope angle
in	s0	wave steepness
out	ksi0	breaker parameter
out	succes	flag for succes
out	errormessage	error message

Definition at line 268 of file formulaModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.3.2.5 subroutine, public formulamoduleovertopping::calculatewavelength (real(kind=wp), intent(in) *Tm_10*, real(kind=wp), intent(out) *L0*)

calculateWaveLength: calculate the wave length

Parameters

in	tm_10	spectral wave period (s)
out	10	wave length (m)

Definition at line 205 of file formulaModuleOvertopping.f90.

Here is the caller graph for this function:



4.3.2.6 subroutine, public formulamoduleovertopping::calculatewaveovertoppingdischarge (real(kind=wp), intent(in) h, real(kind=wp), intent(in) hm0, real(kind=wp), intent(in) tanAlpha, real(kind=wp), intent(in) gammaB, real(kind=wp), intent(in) gammaB, real(kind=wp), intent(in) hcrest, type(tpovertoppinginput), intent(in) modelFactors, real(kind=wp), intent(out) Qo, logical, intent(out) succes, character(len=*), intent(out) errorMessage)

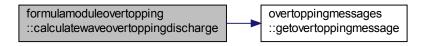
calculateWaveOvertoppingDischarge: calculate the wave overtopping discharge

Parameters

in	h	local water level (m+NAP)
in	hm0	significant wave height (m)
in	tanalpha	representative slope angle
in	gammab	influence factor berms
in	gammaf	influence factor roughness
in	gammabeta	influence factor angle of wave attack
in	ksi0	breaker parameter
in	hcrest	crest level (m+NAP)
in	modelfactors	structure with model factors
out	qo	wave overtopping discharge (I/m per s)
out	succes	flag for succes
out	errormessage	error message

Definition at line 108 of file formulaModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.3.2.7 subroutine, public formulamoduleovertopping::calculatewaverunup (real(kind=wp), intent(in) *Hm0*, real(kind=wp), intent(in) *ksi0*, real(kind=wp), intent(in) *ksi0Limit*, real(kind=wp), intent(inout) *gammaB*, real(kind=wp), intent(inout) *gammaF*, real(kind=wp), intent(inout) *gammaBeta*, type (tpovertoppinginput), intent(in) *modelFactors*, real(kind=wp), intent(out) *z2*, logical, intent(out) *succes*, character(len=*), intent(out) *errorMessage*)

calculateWaveRunup: calculate wave runup

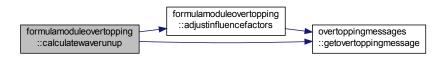
Parameters

in	hm0	significant wave height (m)
in	ksi0	breaker parameter
in	ksi0limit	limit value breaker parameter
in,out	gammab	influence factor berms
in,out	gammaf	influence factor roughness
in,out	gammabeta	influence factor angle of wave attack
in	modelfactors	structure with model factors
out	z2	2% wave run-up (m)

out	succes	flag for succes
out	errormessage	error message

Definition at line 59 of file formulaModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.3.2.8 subroutine, public formulamoduleovertopping::calculatewavesteepness (real(kind=wp), intent(in) *Hm0*, real(kind=wp), intent(out) *s0*, logical, intent(out) *succes*, character(len=*), intent(out) *errorMessage*)

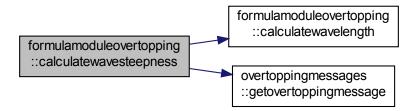
calculateWaveSteepness: calculate the wave steepness

Parameters

in	hm0	significant wave height (m)
in	tm_10	spectral wave period (s)
out	s0	wave steepness
out	succes	flag for succes
out	errormessage	error message

Definition at line 226 of file formulaModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.3.2.9 subroutine formulamoduleovertopping::cubicroots (double complex, intent(in) z, double complex, dimension(3), intent(out) roots) [private]

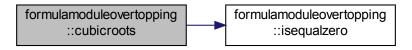
cubicRoots: calculate the roots of a cubic function

Parameters

in	Z	complex number
out	roots	cubic roots

Definition at line 651 of file formulaModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.3.2.10 logical function, public formulamoduleovertopping::isequalreal (real(kind=wp), intent(in) x1, real(kind=wp), intent(in) x2)

isEqualReal: are two reals (almost) equal

Parameters

[in	v1	firet real
L	T11	λ1	ilist real
	in	x2	second real

Definition at line 692 of file formula Module Overtopping. f90.

4.3.2.11 logical function, public formulamoduleovertopping::isequalzero (real(kind=wp), intent(in) x)

isEqualZero: is a real (almost) zero

Parameters

in	X	real number

Definition at line 716 of file formulaModuleOvertopping.f90.

Here is the caller graph for this function:



4.3.2.12 subroutine formulamoduleovertopping::realrootscubicfunction (real(kind=wp), intent(in) a, real(kind=wp), intent(in) b, real(kind=wp), intent(in) c, real(kind=wp), intent(in) d, integer, intent(out) N, real(kind=wp), dimension(3), intent(out) x, logical, intent(out) succes, character(len=*), intent(out) errorMessage) [private]

realRootsCubicFunction: calculate the roots of a cubic function

Parameters

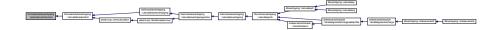
in	а	coefficient a cubic function
in	b	coefficient b cubic function
in	С	coefficient c cubic function
in	d	coefficient d cubic function
out	n	number of real roots cubic function
out	X	real roots cubic function
out	succes	flag for succes
out	errormessage	error message

Definition at line 501 of file formulaModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.3.2.13 subroutine formulamoduleovertopping::rootsdepressedcubic (real(kind=wp), intent(in) p, real(kind=wp), intent(in) q, double complex, dimension(3), intent(out) z) [private]

rootsDepressedCubic: calculate the roots of a depressed cubic function

Parameters

in	р	coefficient p depressed cubic
in	q	coefficient q depressed cubic
out	Z	roots depressed cubic

Definition at line 610 of file formulaModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.3.2.14 subroutine formulamoduleovertopping::rootsgeneralcubic (real(kind=wp), intent(in) *a,* real(kind=wp), intent(in) *b,* real(kind=wp), intent(in) *c,* real(kind=wp), intent(in) *d,* double complex, dimension(3), intent(out) *z,* logical, intent(out) *succes,* character(len=*), intent(out) *errorMessage*) [private]

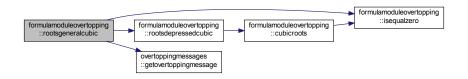
rootsGeneralCubic: calculate the roots of a generic cubic function

Parameters

in	а	coefficients a cubic function
in	b	coefficients b cubic function
in	С	coefficients c cubic function
in	d	coefficients d cubic function
out	Z	roots cubic function
out	succes	flag for succes
out	errormessage	error message

Definition at line 557 of file formulaModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.4 geometrymoduleovertopping Module Reference

core computations related to the geometry

Functions/Subroutines

• subroutine, public checkcrosssection (psi, nCoordinates, xCoordinates, yCoordinates, roughnessFactors, succes, errorMessage)

checkCrossSection: check cross section

• subroutine, public initializegeometry (psi, nCoordinates, xCoordinates, yCoordinates, roughnessFactors, geometry, succes, errorMessage)

initializeGeometry: initialize the geometry

• subroutine, public allocatevectorsgeometry (nCoordinates, geometry, succes, errorMessage)

allocateVectorsGeometry: allocate the geometry vectors

• subroutine, public deallocategeometry (geometry)

deallocateGeometry: deallocate the geometry vectors

subroutine, public calculatesegmentslopes (geometry, succes, errorMessage)

calculateSegmentSlopes: calculate the segment slopes

subroutine, public determinesegmenttypes (geometry, succes, errorMessage)

determineSegmentTypes: determine the segment types

subroutine, public copygeometry (geometry, geometryCopy, succes, errorMessage)

copyGeometry: copy a geometry structure

• subroutine, public mergesequentialberms (geometry, geometryMergedBerms, succes, errorMessage)

mergeSequentialBerms: merge sequential berms

• subroutine, public adjustnonhorizontalberms (geometry, geometryFlatBerms, succes, errorMessage)

adjustNonHorizontalBerms: adjust non-horizontal berms

subroutine, public removeberms (geometry, geometryNoBerms, succes, errorMessage)

removeBerms: remove berms

• subroutine, public removedikesegments (geometry, index, geometryAdjusted, succes, errorMessage)

removeDikeSegments: remove dike segments

• subroutine, public splitcrosssection (geometry, L0, NwideBerms, geometrysectionB, geometrysectionF, succes, errorMessage)

splitCrossSection: split a cross section

• subroutine, public calculatehorzlengths (geometry, yLower, yUpper, horzLengths, succes, errorMessage) calculateHorzLengths: calculate horizontal lengths

• subroutine, public calculatehorzdistance (geometry, yLower, yUpper, dx, succes, errorMessage)

calculateHorzDistance: calculate horizontal distance

• subroutine, public basicgeometrytest (geometryF, success, errorStruct)

basicGeometryTest: test the input geometry (the adjusted geometry is checked elsewhere)

4.4.1 Detailed Description

core computations related to the geometry

4.4.2 Function/Subroutine Documentation

4.4.2.1 subroutine, public geometrymoduleovertopping::adjustnonhorizontalberms (type (tpgeometry), intent(in) *geometry*, type (tpgeometry), intent(out) *geometryFlatBerms*, logical, intent(out) *succes*, character(len=*), intent(out) *errorMessage*)

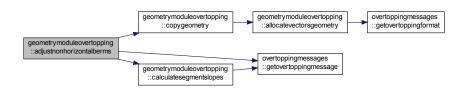
adjustNonHorizontalBerms: adjust non-horizontal berms

Parameters

in	geometry	structure with geometry data
out	geometryflat-	geometry data with horizontal berms
	berms	
out	succes	flag for succes
out	errormessage	error message

Definition at line 571 of file geometryModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.4.2.2 subroutine, public geometrymoduleovertopping::allocatevectorsgeometry (integer, intent(in) *nCoordinates*, type (tpgeometry), intent(inout) *geometry*, logical, intent(out) *succes*, character(len=*), intent(inout) *errorMessage*)

allocateVectorsGeometry: allocate the geometry vectors

Parameters

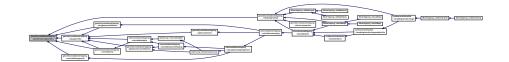
in	ncoordinates	number of coordinates
in,out	geometry	structure with geometry data
out	succes	succes flag
in,out	errormessage	error message (only set in case of error)

Definition at line 246 of file geometryModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.4.2.3 subroutine, public geometrymoduleovertopping::basicgeometrytest (type(overtoppinggeometrytypef), intent(in) *geometryF*, logical, intent(out) *success*, type(terrormessages), intent(inout) *errorStruct*)

basicGeometryTest: test the input geometry (the adjusted geometry is checked elsewhere)

Parameters

in	geometryf	struct with geometry and roughness
in,out	errorstruct	error message (only set if not successful)
out	success	success flag

Definition at line 1074 of file geometryModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.4.2.4 subroutine, public geometrymoduleovertopping::calculatehorzdistance (type (typeometry), intent(in) *geometry*, real(kind=wp), intent(in) *yLower*, real(kind=wp), intent(out) *dx*, logical, intent(out) *succes*, character(len=*), intent(out) *errorMessage*)

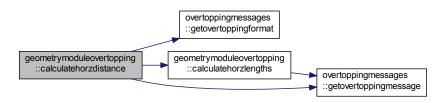
calculateHorzDistance: calculate horizontal distance

in	geometry	structure with geometry data

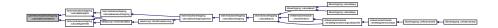
in	ylower	y-coordinate lower bound (m+NAP)
in	yupper	y-coordinate upper bound (m+NAP)
out	dx	horizontal distance between bounds (m)
out	succes	flag for succes
out	errormessage	error message

Definition at line 1025 of file geometryModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.4.2.5 subroutine, public geometrymoduleovertopping::calculatehorzlengths (type (tpgeometry), intent(in) *geometry*, real(kind=wp), intent(in) *yLower*, real(kind=wp), intent(in) *yUpper*, real(kind=wp), dimension(geometry%ncoordinates-1), intent(out) *horzLengths*, logical, intent(out) *succes*, character(len=*), intent(out) *errorMessage*)

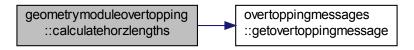
calculateHorzLengths: calculate horizontal lengths

Parameters

in	geometry	structure with geometry data
in	ylower	y-coord. lower bound (m+NAP)
in	yupper	y-coord. upper bound (m+NAP)
out	horzlengths	horizontal lengths segments (m)
out	succes	flag for succes
out	errormessage	error message

Definition at line 929 of file geometryModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.4.2.6 subroutine, public geometrymoduleovertopping::calculatesegmentslopes (type (type (type ometry), intent(inout) *geometry*, logical, intent(out) *succes*, character(len=*), intent(out) *errorMessage*)

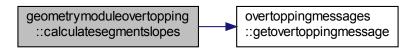
calculateSegmentSlopes: calculate the segment slopes

Parameters

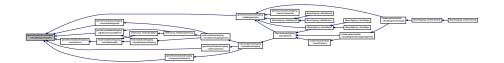
in,out	geometry	structure with geometry data
out	succes	flag for succes
out	errormessage	error message

Definition at line 308 of file geometryModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.4.2.7 subroutine, public geometrymoduleovertopping::checkcrosssection (real(kind=wp), intent(in) *psi*, integer, intent(in) *nCoordinates*, real(kind=wp), dimension (ncoordinates), intent(in) *xCoordinates*, real(kind=wp), dimension (ncoordinates-1), intent(in) *yCoordinates*, real(kind=wp), dimension(ncoordinates-1), intent(in) *roughnessFactors*, logical, intent(out) *succes*, character(len=*), intent(out) *errorMessage*)

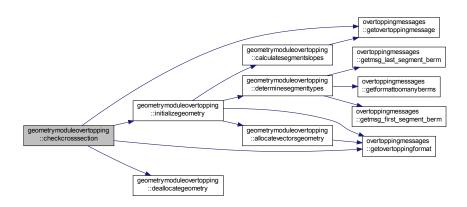
checkCrossSection: check cross section

in	psi	dike normal (degrees)
in	ncoordinates	number of coordinates
in	xcoordinates	x-coordinates (m)

ir	1	ycoordinates	y-coordinates (m+NAP)
ir	1	roughnessfac-	roughness factors
		tors	
ou	t	succes	flag for succes
ou	t	errormessage	error message

Definition at line 59 of file geometryModuleOvertopping.f90.

Here is the call graph for this function:



4.4.2.8 subroutine, public geometrymoduleovertopping::copygeometry (type (tpgeometry), intent(in) *geometry*, type (tpgeometry), intent(inout) *geometryCopy*, logical, intent(out) *succes*, character(len=*), intent(inout) *errorMessage*)

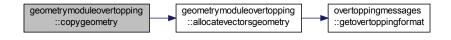
copyGeometry: copy a geometry structure

Parameters

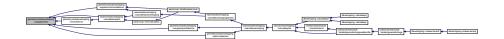
in	geometry	structure with geometry data
in,out	geometrycopy	structure with geometry data copy
out	succes	succes flag
in,out	errormessage	error message, only set in case of error

Definition at line 411 of file geometryModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



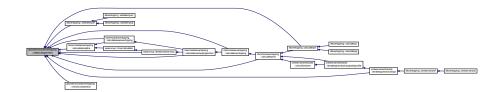
4.4.2.9 subroutine, public geometrymoduleovertopping::deallocategeometry (type (tpgeometry), intent(inout) *geometry*) deallocateGeometry: deallocate the geometry vectors

Parameters

in,out	geometry	structure with geometry data

Definition at line 285 of file geometryModuleOvertopping.f90.

Here is the caller graph for this function:



4.4.2.10 subroutine, public geometrymoduleovertopping::determinesegmenttypes (type (type (type ometry), intent(inout) *geometry*, logical, intent(out) *succes*, character(len=*), intent(out) *errorMessage*)

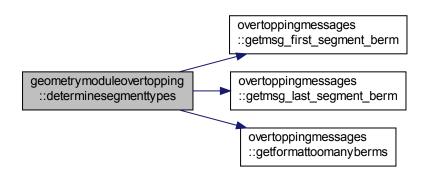
determineSegmentTypes: determine the segment types

Parameters

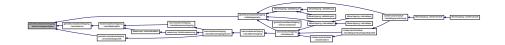
in, ou	ıt	geometry	structure with geometry data
out		succes	success flag
out		errormessage	error message; only set in case of error

Definition at line 347 of file geometryModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.4.2.11 subroutine, public geometrymoduleovertopping::initializegeometry (real(kind=wp), intent(in) *psi*, integer, intent(in) *nCoordinates*, real(kind=wp), dimension (ncoordinates), intent(in) *xCoordinates*, real(kind=wp), dimension (ncoordinates-1), intent(in) *roughnessFactors*, type (tpgeometry), intent(out) *geometry*, logical, intent(out) *succes*, character(len=*), intent(out) *errorMessage*)

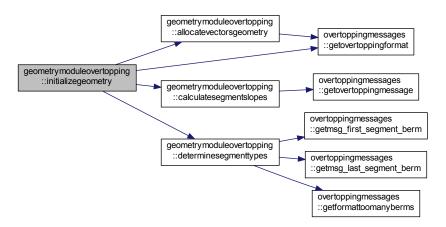
initializeGeometry: initialize the geometry

Parameters

in	psi	dike normal (degree)
in	ncoordinates	number of coordinates
in	xcoordinates	x-coordinates (m)
in	ycoordinates	y-coordinates (m+NAP)
in	roughnessfac-	roughness factors
	tors	
out	geometry	structure with geometry data
out	succes	flag for succes
out	errormessage	error message

Definition at line 170 of file geometryModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.4.2.12 subroutine, public geometrymoduleovertopping::mergesequentialberms (type (tpgeometry), intent(in) *geometry,* type (tpgeometry), intent(inout) *geometryMergedBerms*, logical, intent(out) *succes*, character(len=*), intent(out) *errorMessage*)

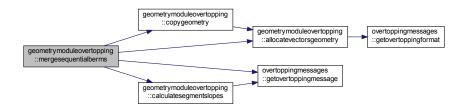
mergeSequentialBerms: merge sequential berms

in	geometry	structure with geometry data

in,out	geome-	geometry data with merged sequential berms
	trymergedberms	
out	succes	flag for succes
out	errormessage	error message

Definition at line 462 of file geometryModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.4.2.13 subroutine, public geometrymoduleovertopping::removeberms (type (tpgeometry), intent(in) *geometry*, type (tpgeometry), intent(out) *geometryNoBerms*, logical, intent(out) *succes*, character(len=*), intent(out) *errorMessage*)

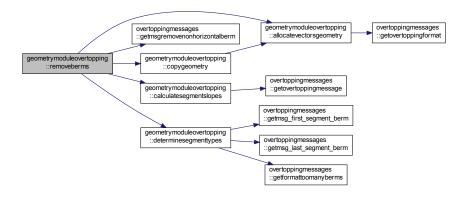
removeBerms: remove berms

Parameters

in	geometry	structure with geometry data
out	geome-	geometry data withouth berms
	trynoberms	
out	succes	flag for succes
out	errormessage	error message

Definition at line 660 of file geometryModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.4.2.14 subroutine, public geometrymoduleovertopping::removedikesegments (type (typeometry), intent(in) *geometry*, integer, intent(in) *index*, type (typeometry), intent(out) *geometryAdjusted*, logical, intent(out) *succes*, character(len=*), intent(out) *errorMessage*)

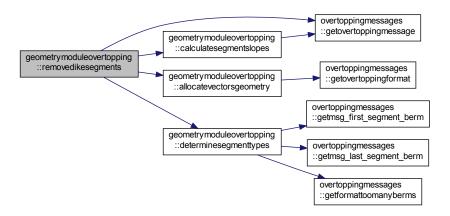
removeDikeSegments: remove dike segments

Parameters

in	geometry	structure with geometry data
in	index	index starting point new cross section
out	geometryad-	geometry data with removed dike segments
	justed	
out	succes	flag for succes
out	errormessage	error message

Definition at line 760 of file geometryModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.4.2.15 subroutine, public geometrymoduleovertopping::splitcrosssection (type (typeometry), intent(in) *geometry*, real(kind=wp), intent(in) *L0*, integer, intent(out) *NwideBerms*, type (typeometry), intent(out) *geometrysectionB*, type (typeometry), intent(out) *geometrysectionF*, logical, intent(out) *succes*, character(len=*), intent(out) *errorMessage*)

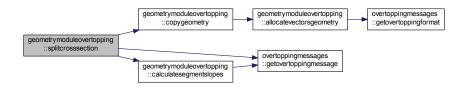
splitCrossSection: split a cross section

Parameters

in	geometry	structure with geometry data
in	10	wave length (m)
out	nwideberms	number of wide berms
out	geometrysec-	geometry data with wide berms to ordinary berms
	tionb	
out	geometrysec-	geometry data with wide berms to foreshores
	tionf	
out	succes	flag for succes
out	errormessage	error message

Definition at line 824 of file geometryModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.5 mainmoduleovertopping Module Reference

core computations for Dikes Overtopping

Functions/Subroutines

- subroutine, public calculateovertopping (geometry, load, modelFactors, overtopping, logging, succes, error
 — Message)
 - calculateOvertopping: calculate the overtopping
- subroutine, public calculateovertoppingsection (geometry, h, Hm0, Tm_10, L0, gammaBeta_z, gammaBeta
 _o, modelFactors, overtopping, logging, succes, errorMessage)
 - $calculate Over topping Section:\ calculate\ the\ over topping\ for\ a\ section$
- subroutine, public calculatewaveovertopping (geometry, h, Hm0, Tm_10, z2, gammaBeta_o, modelFactors, Qo, succes, errorMessage)
 - calculateWaveOvertopping: calculate wave overtopping
- subroutine calculateovertoppingnegativefreeboard (load, geometry, overtopping, succes, errorMessage) calculateOvertoppingNegativeFreeboard: calculate overtopping in case of negative freeboard
- subroutine, public interpolateresultssections (geometry, L0, NwideBerms, overtoppingB, overtoppingF, overtopping, succes, errorMessage)
 - interpolateResultsSections: interpolate results for split cross sections
- subroutine, public checkinputdata (geometry, load, modelFactors, succes, errorMessage)

checkInputdata: check the input data

• subroutine, public checkmodelfactors (modelFactors, dimErrMessage, errorMessages, ierr)

checkModelFactors: check the input data

4.5.1 Detailed Description

core computations for Dikes Overtopping

4.5.2 Function/Subroutine Documentation

4.5.2.1 subroutine, public mainmoduleovertopping::calculateovertopping (type (typeometry), intent(in) *geometry*, type (typoad), intent(in) *load*, type (typovertoppinginput), intent(in) *modelFactors*, type (typovertopping), intent(out) *overtopping*, type(tlogging), intent(in) *logging*, logical, intent(out) *succes*, character(len=*), intent(out) *errorMessage*)

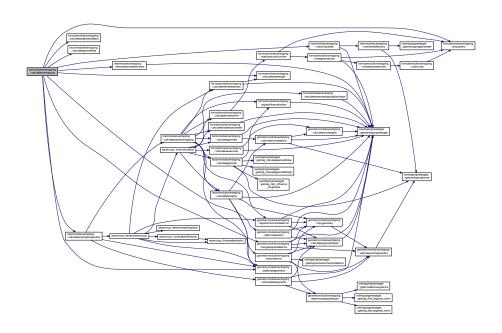
calculateOvertopping: calculate the overtopping

Parameters

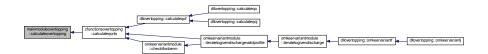
in	geometry	structure with geometry data
in	load	structure with load parameters
in	modelfactors	structure with model factors
out	overtopping	structure with overtopping results
in	logging	logging struct
out	succes	flag for succes
out	errormessage	error message

Definition at line 61 of file mainModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.5.2.2 subroutine mainmoduleovertopping::calculateovertoppingnegativefreeboard (type (tpload), intent(in) *load*, type (tpgeometry), intent(in) *geometry*, type (tpovertopping), intent(inout) *overtopping*, logical, intent(out) *succes*, character(len=*), intent(out) *errorMessage*) [private]

calculateOvertoppingNegativeFreeboard: calculate overtopping in case of negative freeboard

Parameters

in	geometry	structure with geometry data
in	load	structure with load parameters
in,out	overtopping	structure with overtopping results
out	succes	flag for succes
out	errormessage	error message

Definition at line 505 of file mainModuleOvertopping.f90.

Here is the call graph for this function:



4.5.2.3 subroutine, public mainmoduleovertopping::calculateovertoppingsection (type (typeometry), intent(in) *geometry*, real(kind=wp), intent(in) *h*, real(kind=wp), intent(in) *Hm0*, real(kind=wp), intent(in) *Tm_10*, real(kind=wp), intent(in) *L0*, real(kind=wp), intent(inout) *gammaBeta_z*, real(kind=wp), intent(inout) *gammaBeta_o*, type (typovertoppinginput), intent(in) *modelFactors*, type (typovertopping), intent(out) *overtopping*, type(tlogging), intent(in) *logging*, logical, intent(out) *succes*, character(len=*), intent(out) *errorMessage*)

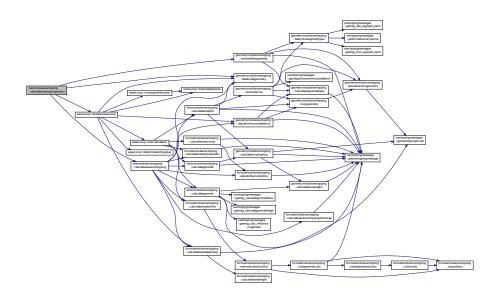
calculateOvertoppingSection: calculate the overtopping for a section

in	geometry	structure with geometry data
in	h	local water level (m+NAP)
in	hm0	significant wave height (m)
in	tm_10	spectral wave period (s)
in	10	wave length (m)
in,out	gammabeta_z	influence angle wave attack wave run-up

in,out	gammabeta_o	influence angle wave attack overtopping
in	modelfactors	structure with model factors
out	overtopping	structure with overtopping results
in	logging	logging struct
out	succes	flag for succes
out	errormessage	error message

Definition at line 191 of file mainModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.5.2.4 subroutine, public mainmoduleovertopping::calculatewaveovertopping (type (typeometry), intent(in) geometry, real(kind=wp), intent(in) h, real(kind=wp), intent(in) Hm0, real(kind=wp), intent(in) Tm_10, real(kind=wp), intent(in) z2, real(kind=wp), intent(inout) gammaBeta_o, type (typovertoppinginput), intent(in) modelFactors, real(kind=wp), intent(out) Qo, logical, intent(out) succes, character(len=*), intent(out) errorMessage)

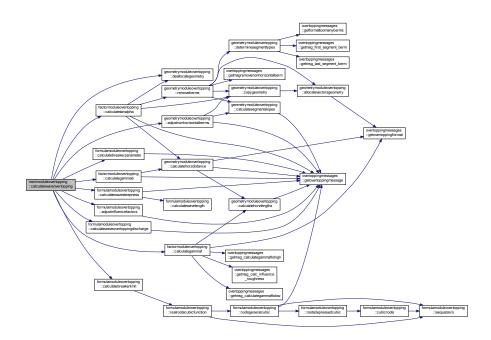
calculateWaveOvertopping: calculate wave overtopping

in	geometry	structure with geometry data
in	h	local water level (m+NAP)
in	hm0	significant wave height (m)
in	tm_10	spectral wave period (s)
in	z2	2% wave run-up (m)

in,out	gammabeta_o	influence angle wave attack overtopping
in	modelfactors	structure with model factors
out	qo	wave overtopping discharge (m3/m per s)
out	succes	flag for succes
out	errormessage	error message

Definition at line 418 of file mainModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.5.2.5 subroutine, public mainmoduleovertopping::checkinputdata (type (typeometry), intent(in) *geometry*, type (typoometry, intent(in) *load*, type (typovertoppinginput), intent(in) *modelFactors*, logical, intent(out) *succes*, character(len=*), intent(out) *errorMessage*)

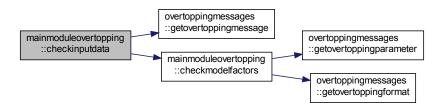
checkInputdata: check the input data

in	geometry	structure with geometry data
in	load	structure with load parameters
in	modelfactors	structure with model factors
out	succes	flag for succes

out	errormessage	error message

Definition at line 622 of file mainModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.5.2.6 subroutine, public mainmoduleovertopping::checkmodelfactors (type (tpovertoppinginput), intent(in) *modelFactors*, integer, intent(in) *dimErrMessage*, character(len=*), dimension(dimerrmessage), intent(out) *errorMessages*, integer, intent(out) *ierr*)

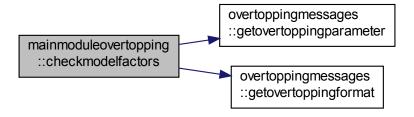
checkModelFactors: check the input data

Parameters

in	modelfactors	structure with model factors
in	dimerrmessage	max. number of error messages
out	ierr	number of errors found
out	errormessages	error message

Definition at line 684 of file mainModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.5.2.7 subroutine, public mainmoduleovertopping::interpolateresultssections (type (tpgeometry), intent(in) *geometry*, real(kind=wp), intent(in) *L0*, integer, intent(in) *NwideBerms*, type (tpovertopping), intent(in) *overtoppingB*, type (tpovertopping), intent(in) *overtoppingF*, type (tpovertopping), intent(out) *overtopping*, logical, intent(out) *succes*, character(len=*), intent(out) *errorMessage*)

interpolateResultsSections: interpolate results for split cross sections

Parameters

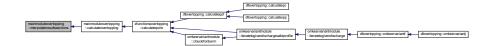
in	geometry	structure with geometry data
in	10	wave length (m)
in	nwideberms	number of wide berms
in	overtoppingb	structure with overtopping results ordinary berms
in	overtoppingf	structure with overtopping results foreshores
out	overtopping	structure with combined overtopping results
out	succes	flag for succes
out	errormessage	error message

Definition at line 541 of file mainModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.6 modulelogging Module Reference

steering the extra logging

Data Types

· type tlogging

TLogging: structure for steering the logging.

Variables

• integer, parameter maxfilenamelength = 256

maximum length of filename

4.6.1 Detailed Description

steering the extra logging

4.6.2 Variable Documentation

4.6.2.1 integer, parameter modulelogging::maxfilenamelength = 256

maximum length of filename

Definition at line 35 of file ModuleLogging.f90.

4.7 omkeervariantmodule Module Reference

Module for the 'omkeerVariant'.

Data Types

· type tomkeer

Functions/Subroutines

• subroutine, public iteratetogivendischarge (load, geometryF, givenDischarge, dikeHeight, modelFactors, overtopping, success, errorText, logging)

Subroutine with omkeerVariant.

• subroutine iteratetogivendischargevalidprofile (load, geometry, givenDischarge, dikeHeight, modelFactors, overtopping, logging, success, errorText)

Subroutine with iterateToGivenDischarge, with already checked profile.

subroutine checkifonberm (geometry, load, modelfactors, overtopping, givenDischarge, dikeHeight, omkeer
 —
 Props, logging, iUp, iLow, dis1, dis2, minDikeHeight, maxDikeHeight, foundValue, success, errorText)

Variables

- real(kind=wp), parameter toldischarge = 1d-3
- real(kind=wp), parameter toldikeheight = 1d-3
- real(kind=wp), parameter minzberm = 0.1_wp

4.7.1 Detailed Description

Module for the 'omkeerVariant'.

4.7.2 Function/Subroutine Documentation

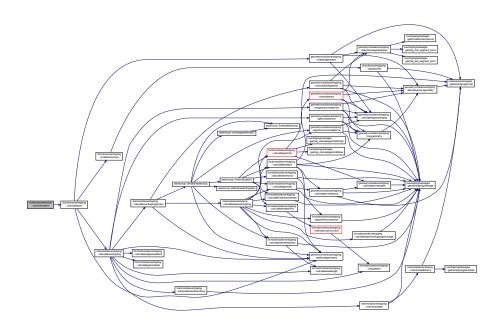
4.7.2.1 subroutine omkeervariantmodule::checkifonberm (type(tpgeometry), intent(in) *geometry*, type(tpload), intent(in) *load*, type(tpovertoppinginput), intent(inout) *modelfactors*, type(tpovertopping), intent(inout) *overtopping*, real(kind=wp), intent(in) *givenDischarge*, real(kind=wp), intent(inout) *dikeHeight*, type(tomkeer), intent(inout) *omkeerProps*, type(tlogging), intent(in) *logging*, integer, intent(in) *iUp*, integer, intent(in) *iLow*, real(kind=wp), intent(inout) *dis1*, real(kind=wp), intent(inout) *dis2*, real(kind=wp), intent(inout) *minDikeHeight*, real(kind=wp), intent(inout) *maxDikeHeight*, logical, intent(inout) *foundValue*, logical, intent(inout) *success*, character(len=*), intent(inout) *errorText*)

Parameters

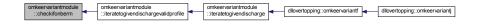
in	geometry	internal structure with geometry data
in	load	struct with waterlevel and wave parameters
in,out	modelfactors	struct with modelFactors
in,out	overtopping	structure with overtopping results
in,out	omkeerprops	struct with derived properties
in	logging	logging struct
in	iup	upper bound on profile
in	ilow	lower bound on profile
in	givendischarge	discharge to iterate to
in,out	dis1	discharge at minDikeHeight
in,out	dis2	discharge at maxDikeHeight
in,out	mindikeheight	lower bound dike heigth
in,out	maxdikeheight	upper bound dike heigth
in,out	dikeheight	dike height
in,out	foundvalue	flag for early succesfull return
in,out	success	flag for success
in,out	errortext	error message (only set if not successful)

Definition at line 248 of file omkeerVariantModule.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.7.2.2 subroutine, public omkeervariantmodule::iteratetogivendischarge (type(tpload), intent(in) *load*, type(overtoppinggeometrytypef), intent(in) *geometryF*, real(kind=wp), intent(in) *givenDischarge*, real(kind=wp), intent(out) *dikeHeight*, type(tpovertoppinginput), intent(inout) *modelFactors*, type (tpovertopping), intent(inout) *overtopping*, logical, intent(out) *success*, character(len=*), intent(out) *errorText*, type(tlogging), intent(in) *logging*)

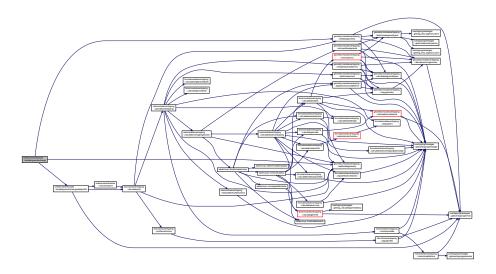
Subroutine with omkeerVariant.

Parameters

in	geometryf	struct with geometry and roughness
in	load	struct with waterlevel and wave parameters
in	givendischarge	discharge to iterate to
out	dikeheight	dike height
in,out	modelfactors	struct with modelFactors
in,out	overtopping	structure with overtopping results
out	success	flag for success
out	errortext	error message (only set if not successful)
in	logging	logging struct

Definition at line 63 of file omkeerVariantModule.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.7.2.3 subroutine omkeervariantmodule::iteratetogivendischargevalidprofile (type(tpload), intent(in) *load*, type(tpgeometry), intent(in) *geometry*, real(kind=wp), intent(in) *givenDischarge*, real(kind=wp), intent(out) *dikeHeight*, type(tpovertoppinginput), intent(inout) *modelFactors*, type (tpovertopping), intent(inout) *overtopping*, type(tlogging), intent(in) *logging*, logical, intent(out) *success*, character(len=*), intent(out) *errorText*) [private]

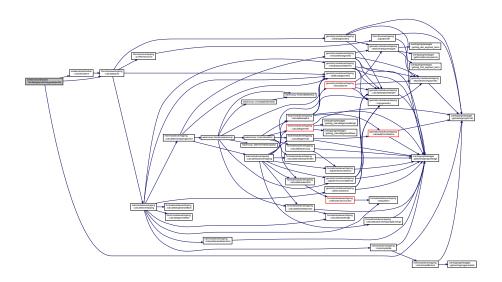
Subroutine with iterateToGivenDischarge, with already checked profile.

in	geometry	internal structure with geometry data

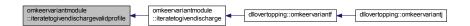
in	load	struct with waterlevel and wave parameters
in	givendischarge	discharge to iterate to
out	dikeheight	dike height
in,out	modelfactors	struct with modelFactors
in,out	overtopping	structure with overtopping results
in	logging	logging struct
out	success	flag for success
out	errortext	error message (only set if not successful)

Definition at line 102 of file omkeerVariantModule.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.7.3 Variable Documentation

4.7.3.1 real(kind=wp), parameter omkeervariantmodule::minzberm = 0.1_wp

Definition at line 46 of file omkeerVariantModule.f90.

4.7.3.2 real(kind=wp), parameter omkeervariantmodule::toldikeheight = 1d-3

Definition at line 45 of file omkeerVariantModule.f90.

4.7.3.3 real(kind=wp), parameter omkeervariantmodule::toldischarge = 1d-3

Definition at line 44 of file omkeerVariantModule.f90.

4.8 overtoppinginterface Module Reference

Module for the interface of dllOvertopping.

Data Types

- · type overtoppinggeometrytype
- · type overtoppinggeometrytypef
- · type tpprofilecoordinate

Variables

integer, parameter, public varmodelfactorcriticalovertopping = 8
 Model factor critical overtopping.

4.8.1 Detailed Description

Module for the interface of dllOvertopping.

4.8.2 Variable Documentation

4.8.2.1 integer, parameter, public overtoppinginterface::varmodelfactorcriticalovertopping = 8

Model factor critical overtopping.

Definition at line 38 of file overtoppingInterface.f90.

4.9 overtoppingmessages Module Reference

Module for the messages in the overtopping dll, in Dutch or English.

Functions/Subroutines

• subroutine setlanguage (lang)

IDs for the strings in this module:

• subroutine getlanguage (lang)

Subroutine that gets the language for error and validation messages.

character(len=maxmsg) function getovertoppingmessage (ID)

Subroutine that returns a message with the corresponding ID in the current language.

character(len=maxmsg) function getovertoppingformat (ID)

Subroutine that returns a Fortran format string with the corresponding ID in the current language.

character(len=maxpar) function getovertoppingparameter (ID)

Subroutine that returns the name of an input parameter with the corresponding ID in the current language.

- subroutine getmsg_calculategammaftolow (message)
- subroutine getmsg_calculategammaftohigh (message)
- subroutine getmsg_calc_influence_roughness (message)
- subroutine getmsg_first_segment_berm (message)
- subroutine getmsg_last_segment_berm (message)
- subroutine getformattoomanyberms (cfmt)
- subroutine getmsgremovenonhorizontalberm (message)

Variables

- integer, parameter, private maxmsg = 128
- integer, parameter, private maxpar =32
- character(len=2), private language = 'NL'

default : Dutch

4.9.1 Detailed Description

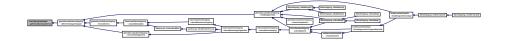
Module for the messages in the overtopping dll, in Dutch or English.

4.9.2 Function/Subroutine Documentation

4.9.2.1 subroutine overtoppingmessages::getformattoomanyberms (character(len=*), intent(out) cfmt)

Definition at line 413 of file OvertoppingMessages.f90.

Here is the caller graph for this function:



4.9.2.2 subroutine overtoppingmessages::getlanguage (character(len=*), intent(out) lang)

Subroutine that gets the language for error and validation messages.

Parameters

out	lang	filled with current language ID

Definition at line 119 of file OvertoppingMessages.f90.

4.9.2.3 subroutine overtoppingmessages::getmsg_calc_influence_roughness (character(len=*), intent(out) message)

Definition at line 386 of file OvertoppingMessages.f90.

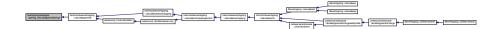
Here is the caller graph for this function:



4.9.2.4 subroutine overtoppingmessages::getmsg_calculategammaftohigh (character(len=*), intent(out) message)

Definition at line 377 of file OvertoppingMessages.f90.

Here is the caller graph for this function:



4.9.2.5 subroutine overtoppingmessages::getmsg_calculategammaftolow (character(len=*), intent(out) message)

Definition at line 368 of file OvertoppingMessages.f90.

Here is the caller graph for this function:



4.9.2.6 subroutine overtoppingmessages::getmsg_first_segment_berm (character(len=*), intent(out) message)

Definition at line 395 of file OvertoppingMessages.f90.

Here is the caller graph for this function:



4.9.2.7 subroutine overtoppingmessages::getmsg_last_segment_berm (character(len=*), intent(out) message)

Definition at line 404 of file OvertoppingMessages.f90.

Here is the caller graph for this function:



4.9.2.8 subroutine overtoppingmessages::getmsgremovenonhorizontalberm (character(len=*), intent(out) message)

Definition at line 422 of file OvertoppingMessages.f90.

Here is the caller graph for this function:



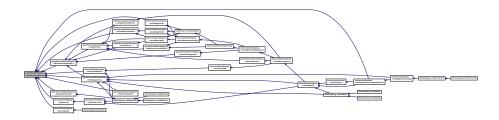
4.9.2.9 character(len=maxmsg) function overtoppingmessages::getovertoppingformat (integer, intent(in) ID)

Subroutine that returns a Fortran format string with the corresponding ID in the current language.

in	id	identification number of string

Definition at line 271 of file OvertoppingMessages.f90.

Here is the caller graph for this function:



4.9.2.10 character(len=maxmsg) function overtoppingmessages::getovertoppingmessage (integer, intent(in) ID)

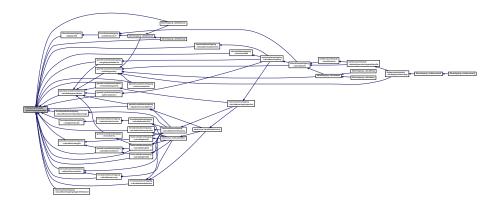
Subroutine that returns a message with the corresponding ID in the current language.

Parameters

in	id	identification number of string
----	----	---------------------------------

Definition at line 129 of file OvertoppingMessages.f90.

Here is the caller graph for this function:



4.9.2.11 character(len=maxpar) function overtoppingmessages::getovertoppingparameter (integer, intent(in) ID)

Subroutine that returns the name of an input parameter with the corresponding ID in the current language.

Parameters

	1	
in	id	identification number of string

Definition at line 329 of file OvertoppingMessages.f90.

Here is the caller graph for this function:



4.9.2.12 subroutine overtoppingmessages::setlanguage (character(len=*), intent(in) lang)

IDs for the strings in this module:

Subroutine that sets the language for error and validation messages only strings 'NL' and 'UK' are recognized (lower and upper case)

Parameters

in	lang	new language ID to be used

Definition at line 101 of file OvertoppingMessages.f90.

4.9.3 Variable Documentation

4.9.3.1 character(len=2), private overtoppingmessages::language = 'NL'

default : Dutch

Definition at line 38 of file OvertoppingMessages.f90.

4.9.3.2 integer, parameter, private overtoppingmessages::maxmsg = 128

Definition at line 36 of file OvertoppingMessages.f90.

4.9.3.3 integer, parameter, private overtoppingmessages::maxpar =32

Definition at line 36 of file OvertoppingMessages.f90.

4.10 typedefinitions overtopping Module Reference

type definitions for Dikes Overtopping

Data Types

· type tpgeometry

tpGeometry: structure with geometry data

type tpload

tpLoad: structure with load parameters

• type tpovertopping

tpOvertopping: structure with overtopping results

type tpovertoppinginput

Over topping ModelFactors: C-structure with model factors.

Variables

- real(kind=wp), parameter frunup1 = 1.65_wp
- real(kind=wp), parameter frunup2 = 4.00_wp
- real(kind=wp), parameter frunup3 = 1.50_wp
- real(kind=wp), parameter xdiff_min = 2.0d-2

minimal value distance between x-coordinates (m)

• real(kind=wp), parameter margindiff = 1.0d-14

margin for minimal distance (m)

- real(kind=wp), parameter berm_min = 0.0d0
 minimal value gradient berm segment
- real(kind=wp), parameter berm_max = 1.0d0/15
 maximal value gradient berm segment
- real(kind=wp), parameter slope_min = 1.0d0/8
 minimal value gradient slope segment
- real(kind=wp), parameter slope_max = 1.0d0
 maximal value gradient slope segment
- real(kind=wp), parameter margingrad = 0.0025d0
 margin for minimal and maximal gradients
- real(kind=wp), parameter rfactor_min = 0.5d0
 minimal value roughness factor dike segments
- real(kind=wp), parameter rfactor_max = 1.0d0
 maximal value roughness factor dike segments
- real(kind=wp), parameter mz2_min = 0.0d0
 minimal value model factor of 2% runup height
- real(kind=wp), parameter mz2_max = huge(mz2_max)
 maximal value model factor of 2% runup height
- real(kind=wp), parameter fb_min = 0.0d0
 minimal value model factor for breaking waves
- real(kind=wp), parameter fb_max = huge(fB_max)
 maximal value model factor for breaking waves
- real(kind=wp), parameter fn_min = 0.0d0
 minimal value model factor for non-breaking waves
- real(kind=wp), parameter fn_max = huge(fN_max)
 maximal value model factor for non-breaking waves
- real(kind=wp), parameter fs_min = 0.0d0
 minimal value model factor for shallow waves
- real(kind=wp), parameter fs_max = huge(fS_max)
 maximal value model factor for shallow waves
- real(kind=wp), parameter foreshore_min = 0.3d0

 minimal value reduction factor foreshore
- real(kind=wp), parameter foreshore_max = 1.0d0
 maximal value reduction factor foreshore
- integer, parameter z2_iter_max1 = 49

 maximal number of iterations for calculation z2 part 1
- integer, parameter z2_iter_max2 = 70

maximal number of iterations for calculation z2 part 1 & 2

• real(kind=wp), parameter z2_margin = 0.001d0

margin for convergence criterium calculation z2

4.10.1 Detailed Description

type definitions for Dikes Overtopping

4.10.2 Variable Documentation

4.10.2.1 real(kind=wp), parameter typedefinitionsovertopping::berm_max = 1.0d0/15

maximal value gradient berm segment

Definition at line 91 of file typeDefinitionsOvertopping.f90.

4.10.2.2 real(kind=wp), parameter typedefinitionsovertopping::berm_min = 0.0d0 minimal value gradient berm segment

4.10.2.3 real(kind=wp), parameter typedefinitionsovertopping::fb_max = huge(fB_max)

maximal value model factor for breaking waves

Definition at line 100 of file typeDefinitionsOvertopping.f90.

Definition at line 90 of file typeDefinitionsOvertopping.f90.

4.10.2.4 real(kind=wp), parameter typedefinitionsovertopping::fb_min = 0.0d0

minimal value model factor for breaking waves

Definition at line 99 of file typeDefinitionsOvertopping.f90.

4.10.2.5 real(kind=wp), parameter typedefinitionsovertopping::fn_max = huge(fN_max)

maximal value model factor for non-breaking waves

Definition at line 102 of file typeDefinitionsOvertopping.f90.

4.10.2.6 real(kind=wp), parameter typedefinitionsovertopping::fn_min = 0.0d0

minimal value model factor for non-breaking waves

Definition at line 101 of file typeDefinitionsOvertopping.f90.

4.10.2.7 real(kind=wp), parameter typedefinitionsovertopping::foreshore_max = 1.0d0

maximal value reduction factor foreshore

Definition at line 106 of file typeDefinitionsOvertopping.f90.

4.10.2.8 real(kind=wp), parameter typedefinitionsovertopping::foreshore_min = 0.3d0

minimal value reduction factor foreshore

Definition at line 105 of file typeDefinitionsOvertopping.f90.

4.10.2.9 real(kind=wp), parameter typedefinitionsovertopping::frunup1 = 1.65_wp

Definition at line 63 of file typeDefinitionsOvertopping.f90.

4.10.2.10 real(kind=wp), parameter typedefinitionsovertopping::frunup2 = 4.00_wp

Definition at line 64 of file typeDefinitionsOvertopping.f90.

4.10.2.11 real(kind=wp), parameter typedefinitionsovertopping::frunup3 = 1.50_wp

Definition at line 65 of file typeDefinitionsOvertopping.f90.

4.10.2.12 real(kind=wp), parameter typedefinitionsovertopping::fs_max = huge(fS_max)

maximal value model factor for shallow waves

Definition at line 104 of file typeDefinitionsOvertopping.f90.

4.10.2.13 real(kind=wp), parameter typedefinitionsovertopping::fs_min = 0.0d0

minimal value model factor for shallow waves

Definition at line 103 of file typeDefinitionsOvertopping.f90.

4.10.2.14 real(kind=wp), parameter typedefinitionsovertopping::margindiff = 1.0d-14

margin for minimal distance (m)

Definition at line 89 of file typeDefinitionsOvertopping.f90.

4.10.2.15 real(kind=wp), parameter typedefinitionsovertopping::margingrad = 0.0025d0

margin for minimal and maximal gradients

Definition at line 94 of file typeDefinitionsOvertopping.f90.

4.10.2.16 real(kind=wp), parameter typedefinitionsovertopping::mz2_max = huge(mz2_max)

maximal value model factor of 2% runup height

Definition at line 98 of file typeDefinitionsOvertopping.f90.

4.10.2.17 real(kind=wp), parameter typedefinitionsovertopping::mz2_min = 0.0d0

minimal value model factor of 2% runup height

Definition at line 97 of file typeDefinitionsOvertopping.f90.

4.10.2.18 real(kind=wp), parameter typedefinitionsovertopping::rfactor_max = 1.0d0

maximal value roughness factor dike segments

Definition at line 96 of file typeDefinitionsOvertopping.f90.

4.10.2.19 real(kind=wp), parameter typedefinitionsovertopping::rfactor_min = 0.5d0

minimal value roughness factor dike segments

Definition at line 95 of file typeDefinitionsOvertopping.f90.

4.10.2.20 real(kind=wp), parameter typedefinitionsovertopping::slope_max = 1.0d0

maximal value gradient slope segment

Definition at line 93 of file typeDefinitionsOvertopping.f90.

64 Module Documentation

4.10.2.21 real(kind=wp), parameter typedefinitionsovertopping::slope_min = 1.0d0/8

minimal value gradient slope segment

Definition at line 92 of file typeDefinitionsOvertopping.f90.

4.10.2.22 real(kind=wp), parameter typedefinitionsovertopping::xdiff_min = 2.0d-2

minimal value distance between x-coordinates (m)

Definition at line 88 of file typeDefinitionsOvertopping.f90.

4.10.2.23 integer, parameter typedefinitionsovertopping::z2_iter_max1 = 49

maximal number of iterations for calculation z2 part 1

Definition at line 107 of file typeDefinitionsOvertopping.f90.

4.10.2.24 integer, parameter typedefinitionsovertopping::z2 iter max2 = 70

maximal number of iterations for calculation z2 part 1 & 2

Definition at line 108 of file typeDefinitionsOvertopping.f90.

4.10.2.25 real(kind=wp), parameter typedefinitionsovertopping::z2_margin = 0.001d0

margin for convergence criterium calculation z2

Definition at line 109 of file typeDefinitionsOvertopping.f90.

4.11 waverunup Module Reference

Iteration procedure for 2% wave runup.

Functions/Subroutines

subroutine, public iterationwaverunup (geometry, h, Hm0, Tm_10, gammaBeta_z, modelFactors, z2, logging, succes, errorMessage)

iterationWaveRunup: iteration for the wave runup

real(kind=wp) function innercalculation (geometry, h, Hm0, gammaBeta_z, modelFactors, z2, s0, geometry ←
FlatBerms, succes, errorMessage)

innerCalculation: inner calculation for the wave runup

• real(kind=wp) function determinestartingvalue (i, relaxationFactor, z2_start, z2_end, Hm0)

determineStartingValue: helper function to find a start value for z2

• integer function findsmallestresidu (z2_start, z2_end, n)

findSmallestResidu: helper function to find the smallest residu

• subroutine convergedwithresidu (z2_start, z2_end, logging)

convergedWithResidu: helper function to handle convergence with a higher residu than expected if logging is enabled, it writes a small message to the logfile

4.11.1 Detailed Description

Iteration procedure for 2% wave runup.

4.11.2 Function/Subroutine Documentation

4.11.2.1 subroutine waverunup::convergedwithresidu (real(kind=wp), dimension(:), intent(in) z2_start, real(kind=wp), dimension(:), intent(inout) z2_end, type(tlogging), intent(in) logging) [private]

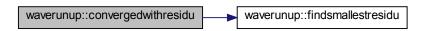
convergedWithResidu: helper function to handle convergence with a higher residu than expected if logging is enabled, it writes a small message to the logfile

Parameters

	in	z2_start	_start array with z2 values at the start of the iteration i	
ſ	in,out	z2_end	z2_end array with z2 values at the end of iteration i	
in logging logging struct		logging struct		

Definition at line 373 of file waveRunup.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.11.2.2 real(kind=wp) function waverunup::determinestartingvalue (integer, intent(in) *i*, real(kind=wp), intent(in) relaxationFactor, real(kind=wp), dimension(:), intent(in) z2_start, real(kind=wp), dimension(:), intent(in) z2_end, real(kind=wp), intent(in) Hm0) [private]

determineStartingValue: helper function to find a start value for z2

Parameters

in	i	i current iteration number	
in	relaxationfactor	relaxation factor as given by user	
in	z2_start	array with z2 values at the start of the iteration i	
in	z2_end	array with z2 values at the end of iteration i	
in	hm0	significant wave height	

Returns

return value: start value for z2 in current iteration

Definition at line 292 of file waveRunup.f90.

Here is the caller graph for this function:



66 Module Documentation

4.11.2.3 integer function waverunup::findsmallestresidu (real(kind=wp), dimension(:), intent(in) z2_start, real(kind=wp), dimension(:), intent(in) z2_end, integer, intent(in), optional n) [private]

findSmallestResidu: helper function to find the smallest residu

Parameters

in	z2_start	array with z2 values at the start of the iteration i	
in	z2_end	array with z2 values at the end of iteration i	
in	n	number of iterations already done	

Definition at line 330 of file waveRunup.f90.

Here is the caller graph for this function:



4.11.2.4 real(kind=wp) function waverunup::innercalculation (type (tpgeometry), intent(in) geometry, real(kind=wp), intent(in) h, real(kind=wp), intent(in) Hm0, real(kind=wp), intent(inout) gammaBeta_z, type (tpovertoppinginput), intent(in) modelFactors, real(kind=wp), intent(in) z2, real(kind=wp), intent(in) s0, type (tpgeometry), intent(in) geometryFlatBerms, logical, intent(out) succes, character(len=*), intent(out) errorMessage) [private]

innerCalculation: inner calculation for the wave runup

Parameters

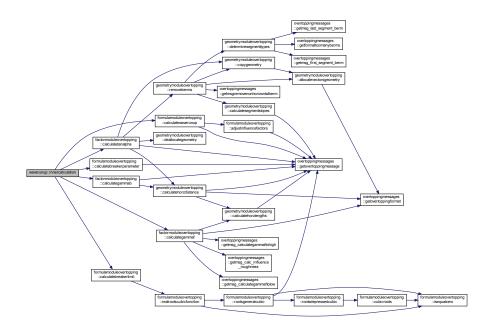
in	geometry	structure with geometry data	
in	h	local water level (m+NAP)	
in	hm0	significant wave height (m)	
in,out	gammabeta_z	influence factor angle wave attack 2% run-up	
in	modelfactors	structure with model factors	
in	z2	2% wave run-up (m)	
in	s0	wave steepness	
in	geometryflat-	t- structure with geometry data with horizontal berms	
	berms		
out	succes	flag for succes	
out	errormessage	error message	

Returns

2% wave run-up at end of inner calculation

Definition at line 181 of file waveRunup.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.11.2.5 subroutine, public waverunup::iterationwaverunup (type (tpgeometry), intent(in) *geometry*, real(kind=wp), intent(in) *h*, real(kind=wp), intent(in) *Hm0*, real(kind=wp), intent(in) *Tm_10*, real(kind=wp), intent(inout) *gammaBeta_z*, type (tpovertoppinginput), intent(in) *modelFactors*, real(kind=wp), intent(out) *z2*, type(tlogging), intent(in) *logging*, logical, intent(out) *succes*, character(len=*), intent(out) *errorMessage*)

iterationWaveRunup: iteration for the wave runup

Parameters

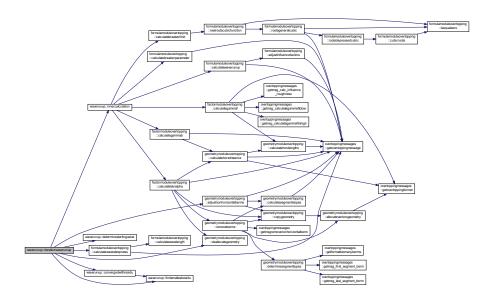
in	geometry	structure with geometry data	
in	h	local water level (m+NAP)	
in	hm0	nm0 significant wave height (m)	
in	tm_10	_10 spectral wave period (s)	
in,out	gammabeta_z	eta_z influence factor angle wave attack 2% run-up	
in	in modelfactors structure with model factors		

68 Module Documentation

out	z2	2% wave run-up (m)
in	logging	logging struct
out	succes	flag for succes
out	errormessage	error message

Definition at line 58 of file waveRunup.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.12 zfunctionsovertopping Module Reference

Module for the Limit State Functions (Z-functions) for wave overtopping.

Functions/Subroutines

• subroutine, public calculateqorto (dikeHeight, modelFactors, overtopping, load, geometry, logging, succes, errorMessage)

Subroutine to calculate the overtopping discharge with the Overtopping dll.

• subroutine, public profileinstructure (nrCoordinates, xcoordinates, ycoordinates, dikeHeight, nrCoords

Adjusted, xCoords Adjusted, zCoords Adjusted, succes, error Message)

Subroutine to fill the profile in a structure and call the adjustment function of the profile due to a desired dike height.

subroutine adjustprofile (nrCoordinates, coordinates, dikeHeight, nrCoordsAdjusted, xCoordsAdjusted, z
 — CoordsAdjusted, succes, errorMessage)

Subroutine adjust the profile due to a desired dike height.

• real(kind=wp) function, public zfunclogratios (qo, qc, mqo, mqc, success, errorMessage)

Routine to compute the limit state value by using the logs of the overtopping discharges (computed and desired)

4.12.1 Detailed Description

Module for the Limit State Functions (Z-functions) for wave overtopping.

4.12.2 Function/Subroutine Documentation

4.12.2.1 subroutine zfunctionsovertopping::adjustprofile (integer, intent(in) nrCoordinates, type(typrofilecoordinate), dimension(nrcoordinates), intent(in) coordinates, real(kind=wp), intent(in) dikeHeight, integer, intent(out) nrCoordsAdjusted, real(kind=wp), dimension(:), pointer xCoordsAdjusted, real(kind=wp), dimension(:), pointer zCoordsAdjusted, logical, intent(out) succes, character(len=*), intent(out) errorMessage) [private]

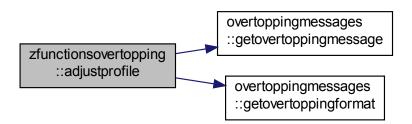
Subroutine adjust the profile due to a desired dike height.

Parameters

in	nrcoordinates	number of coordinates of the profile	
in	coordinates	structure for the profile	
in	dikeheight	dike height	
out	nrcoordsad-	number of coordinates in the adjusted profile	
	justed	d	
	xcoordsadjusted	sted vector with x-coordinates of the adjusted profile	
zcoordsadjusted vector with y-coordinates of the adjusted profile		vector with y-coordinates of the adjusted profile	
out	out succes flag for succes		
out	errormessage	error message	

Definition at line 131 of file zFunctionsOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



70 Module Documentation

4.12.2.2 subroutine, public zfunctionsovertopping::calculateqorto (real(kind=wp), intent(in) dikeHeight, type(tpovertoppinginput), intent(inout) modelFactors, type (tpovertopping), intent(out) overtopping, type (tpload), intent(in) load, type (tpgeometry), intent(in) geometry, type(tlogging), intent(in) logging, logical, intent(out) succes, character(len=*), intent(out) errorMessage)

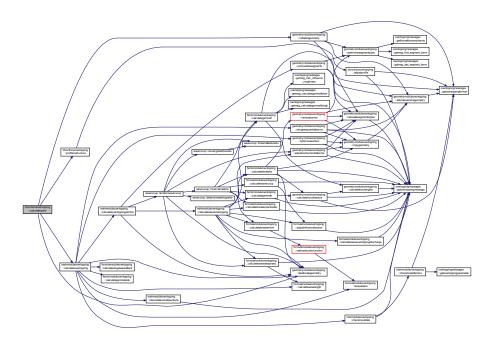
Subroutine to calculate the overtopping discharge with the Overtopping dll.

Parameters

in	dikeheight	dike height	
in,out	modelfactors	struct with model factors	
out	overtopping	structure with overtopping results	
in	geometry	structure with geometry data	
in	load	structure with load parameters	
in	logging	logging struct	
out	succes	flag for succes	
out	errormessage	error message	

Definition at line 56 of file zFunctionsOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.12.2.3 subroutine, public zfunctionsovertopping::profileinstructure (integer, intent(in) nrCoordinates, real(kind=wp), dimension(nrcoordinates), intent(in) xcoordinates, real(kind=wp), dimension(nrcoordinates), intent(in) ycoordinates, real(kind=wp), intent(in) dikeHeight, integer, intent(out) nrCoordsAdjusted, real(kind=wp), dimension(:), pointer xCoordsAdjusted, real(kind=wp), dimension(:), pointer zCoordsAdjusted, logical, intent(out) succes, character(len=*), intent(out) errorMessage)

Subroutine to fill the profile in a structure and call the adjustment function of the profile due to a desired dike height.

72 Module Documentation

Parameters

in	nrcoordinates	number of coordinates of the profile	
in	xcoordinates	vector with x-coordinates of the profile	
in	ycoordinates	vector with y-coordinates of the profile	
in	dikeheight	dike height	
out	nrcoordsad-	number of coordinates in the adjusted profile	
	justed	d	
	xcoordsadjusted	vector with x-coordinates of the adjusted profile	
	zcoordsadjusted vector with y-coordinates of the adjusted profile		
out	succes	flag for succes	
out	errormessage	error message	

Definition at line 106 of file zFunctionsOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.12.2.4 real (kind=wp) function, public zfunctionsovertopping::zfunclogratios (real (kind=wp), intent(in) *qo*, real (kind=wp), intent(in) *qc*, real (kind=wp), intent(in) *mqc*, logical, intent(out) *success*, character(len=*), intent(out) *errorMessage*)

Routine to compute the limit state value by using the logs of the overtopping discharges (computed and desired)

Parameters

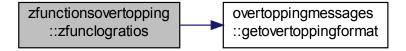
in	qo	computed overtopping discharge	
in	qc	Critical overtopping discharge	
in	mqo	Model factor computed overtopping discharge	
in	mqc	Model factor Critical overtopping discharge	
out	success	Flag for succes	
out	errormessage	error message, only set if not successful	

Returns

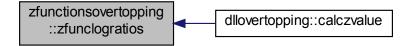
Value z-function

Definition at line 241 of file zFunctionsOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



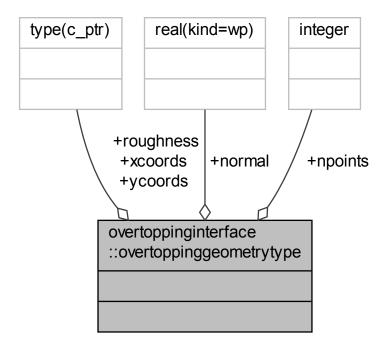
74 **Module Documentation**

Chapter 5

Data Type Documentation

5.1 overtoppinginterface::overtoppinggeometrytype Type Reference

Collaboration diagram for overtoppinginterface::overtoppinggeometrytype:



Public Attributes

- real(kind=wp) normal
- integer npoints
- type(c_ptr) xcoords
- type(c_ptr) ycoords
- type(c_ptr) roughness

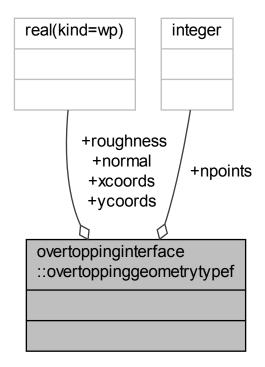
5.1.1 Detailed Description

Definition at line 50 of file overtoppingInterface.f90.

o.iii Botanoa	- Decemption
Definition at line	46 of file overtoppingInterface.f90.
5.1.2 Member	Data Documentation
5.1.2.1 real(kind=	-wp) overtoppinginterface::overtoppinggeometrytype::normal
Definition at line	47 of file overtoppingInterface.f90.
5.1.2.2 integer ov	vertoppinginterface::overtoppinggeometrytype::npoints
Definition at line	48 of file overtoppingInterface.f90.
5.1.2.3 type(c_pt	r) overtoppinginterface::overtoppinggeometrytype::roughness
Definition at line	51 of file overtoppingInterface.f90.
5404 tuma(a mt	
5.1.2.4 type(c_pti	r) overtoppinginterface::overtoppinggeometrytype::xcoords
Definition at line	49 of file overtoppingInterface.f90.
5.1.2.5 type(c_pt	r) overtoppinginterface::overtoppinggeometrytype::ycoords

5.2 overtoppinginterface::overtoppinggeometrytypef Type Reference

Collaboration diagram for overtoppinginterface::overtoppinggeometrytypef:



Public Attributes

- real(kind=wp) normal
- · integer npoints
- real(kind=wp), dimension(:), pointer xcoords
- real(kind=wp), dimension(:), pointer ycoords
- real(kind=wp), dimension(:), pointer roughness

5.2.1 Detailed Description

Definition at line 54 of file overtoppingInterface.f90.

5.2.2 Member Data Documentation

5.2.2.1 real(kind=wp) overtoppinginterface::overtoppinggeometrytypef::normal

Definition at line 55 of file overtoppingInterface.f90.

5.2.2.2 integer overtoppinginterface::overtoppinggeometrytypef::npoints

Definition at line 56 of file overtoppingInterface.f90.

5.2.2.3 real(kind=wp), dimension(:), pointer overtoppinginterface::overtoppinggeometrytypef::roughness

Definition at line 59 of file overtoppingInterface.f90.

5.2.2.4 real(kind=wp), dimension(:), pointer overtoppinginterface::overtoppinggeometrytypef::xcoords

Definition at line 57 of file overtoppingInterface.f90.

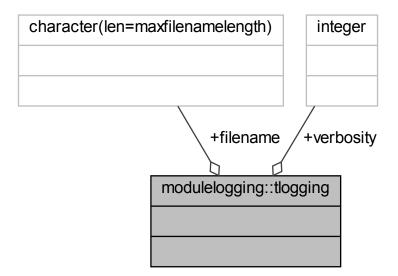
5.2.2.5 real(kind=wp), dimension(:), pointer overtoppinginterface::overtoppinggeometrytypef::ycoords

Definition at line 58 of file overtoppingInterface.f90.

5.3 modulelogging::tlogging Type Reference

TLogging: structure for steering the logging.

Collaboration diagram for modulelogging::tlogging:



Public Attributes

- integer verbosity = verboseNone

 level of verbosity: one of verboseNone, verboseBasic, verboseDetailed, verboseDebugging
- character(len=maxfilenamelength) filename = ''
 filename of logging

5.3.1 Detailed Description

TLogging: structure for steering the logging.

Definition at line 38 of file ModuleLogging.f90.

5.3.2 Member Data Documentation

5.3.2.1 character(len=maxfilenamelength) modulelogging::tlogging::filename = ' '

filename of logging

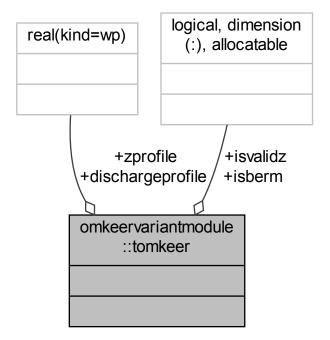
Definition at line 40 of file ModuleLogging.f90.

5.3.2.2 integer modulelogging::tlogging::verbosity = verboseNone

level of verbosity: one of verboseNone, verboseBasic, verboseDetailed, verboseDebugging Definition at line 39 of file ModuleLogging.f90.

5.4 omkeervariantmodule::tomkeer Type Reference

Collaboration diagram for omkeervariantmodule::tomkeer:



Public Attributes

- logical, dimension(:), allocatable isberm
- logical, dimension(:), allocatable isvalidz
- real(kind=wp), dimension(:), allocatable dischargeprofile
- real(kind=wp), dimension(:), allocatable zprofile

5.4.1	Detailed	Description
-------	----------	-------------

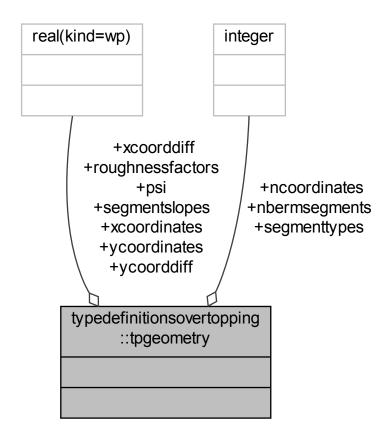
5.4.1	Detailed Description				
Definiti	Definition at line 48 of file omkeerVariantModule.f90.				
5.4.2	Member Data Documentation				
5.4.2.1	real(kind=wp), dimension(:), allocatable omkeervariantmodule::tomkeer::dischargeprofile				
Definition at line 51 of file omkeerVariantModule.f90.					
5422	logical dimension(s) allocatable embroayyayiantmoduloutembooyyiiahaym				
5.4.2.2	logical, dimension(:), allocatable omkeervariantmodule::tomkeer::isberm				
Definition at line 49 of file omkeerVariantModule.f90.					
5.4.2.3	logical, dimension(:), allocatable omkeervariantmodule::tomkeer::isvalidz				
Definition at line 50 of file omkeerVariantModule.f90.					
5.4.2.4	real(kind=wp), dimension(:), allocatable omkeervariantmodule::tomkeer::zprofile				

5.5 typedefinitionsovertopping::tpgeometry Type Reference

tpGeometry: structure with geometry data

Definition at line 52 of file omkeerVariantModule.f90.

Collaboration diagram for typedefinitionsovertopping::tpgeometry:



Public Attributes

- real(kind=wp) psi
 - dike normal (degrees)
- integer ncoordinates
 - number of coordinates cross section
- real(kind=wp), dimension(:), pointer xcoordinates => null()
 - vector with x-coordinates cross section (m)
- real(kind=wp), dimension(:), pointer ycoordinates => null()
- vector with y-coordinates cross section (m+NAP)
 real(kind=wp), dimension(:), pointer roughnessfactors => null()
 - vector with roughness factors cross section
- real(kind=wp), dimension(:), pointer xcoorddiff => null()
 - vector with differences in x-coordinates (m)
- real(kind=wp), dimension(:), pointer ycoorddiff => null()
 - vector with differences in y-coordinates (m)
- real(kind=wp), dimension(:), pointer segmentslopes => null()
 - vector with slopes dike segments
- integer, dimension(:), pointer segmenttypes

vector with segment types (1=slope,2=berm,3=other)

• integer nbermsegments

number of berm segments

5.5.1 Detailed Description

tpGeometry: structure with geometry data

Definition at line 42 of file typeDefinitionsOvertopping.f90.

5.5.2 Member Data Documentation

5.5.2.1 integer typedefinitionsovertopping::tpgeometry::nbermsegments

number of berm segments

Definition at line 52 of file typeDefinitionsOvertopping.f90.

5.5.2.2 integer typedefinitionsovertopping::tpgeometry::ncoordinates

number of coordinates cross section

Definition at line 44 of file typeDefinitionsOvertopping.f90.

5.5.2.3 real(kind=wp) typedefinitionsovertopping::tpgeometry::psi

dike normal (degrees)

Definition at line 43 of file typeDefinitionsOvertopping.f90.

5.5.2.4 real(kind=wp), dimension(:), pointer typedefinitionsovertopping::tpgeometry::roughnessfactors => null()

vector with roughness factors cross section

Definition at line 47 of file typeDefinitionsOvertopping.f90.

5.5.2.5 real(kind=wp), dimension(:), pointer typedefinitionsovertopping::tpgeometry::segmentslopes => null()

vector with slopes dike segments

Definition at line 50 of file typeDefinitionsOvertopping.f90.

 $5.5.2.6 \quad integer, dimension (:), pointer type definitions over topping:: type ometry:: segment types$

vector with segment types (1=slope,2=berm,3=other)

Definition at line 51 of file typeDefinitionsOvertopping.f90.

5.5.2.7 real(kind=wp), dimension(:), pointer typedefinitionsovertopping::tpgeometry::xcoorddiff => null()

vector with differences in x-coordinates (m)

Definition at line 48 of file typeDefinitionsOvertopping.f90.

5.5.2.8 real(kind=wp), dimension(:), pointer typedefinitionsovertopping::tpgeometry::xcoordinates => null()

vector with x-coordinates cross section (m)

Definition at line 45 of file typeDefinitionsOvertopping.f90.

5.5.2.9 real(kind=wp), dimension(:), pointer typedefinitionsovertopping::tpgeometry::ycoorddiff => null()

vector with differences in y-coordinates (m)

Definition at line 49 of file typeDefinitionsOvertopping.f90.

5.5.2.10 real(kind=wp), dimension(:), pointer typedefinitionsovertopping::tpgeometry::ycoordinates => null()

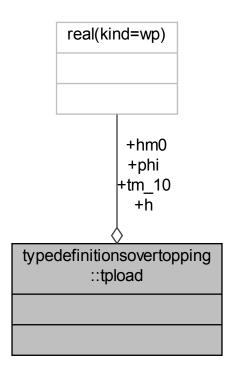
vector with y-coordinates cross section (m+NAP)

Definition at line 46 of file typeDefinitionsOvertopping.f90.

5.6 typedefinitionsovertopping::tpload Type Reference

tpLoad: structure with load parameters

Collaboration diagram for typedefinitionsovertopping::tpload:



Public Attributes

• real(kind=wp) h

local water level (m+NAP)

• real(kind=wp) hm0

significant wave height (m)

• real(kind=wp) tm_10

spectral wave period (s)

• real(kind=wp) phi

wave direction (degrees)

5.6.1 Detailed Description

tpLoad: structure with load parameters

Definition at line 56 of file typeDefinitionsOvertopping.f90.

5.6.2 Member Data Documentation

5.6.2.1 real(kind=wp) typedefinitionsovertopping::tpload::h

local water level (m+NAP)

Definition at line 57 of file typeDefinitionsOvertopping.f90.

5.6.2.2 real(kind=wp) typedefinitionsovertopping::tpload::hm0

significant wave height (m)

Definition at line 58 of file typeDefinitionsOvertopping.f90.

5.6.2.3 real(kind=wp) typedefinitionsovertopping::tpload::phi

wave direction (degrees)

Definition at line 60 of file typeDefinitionsOvertopping.f90.

5.6.2.4 real(kind=wp) typedefinitionsovertopping::tpload::tm_10

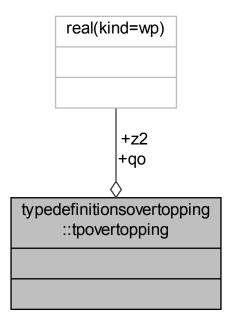
spectral wave period (s)

Definition at line 59 of file typeDefinitionsOvertopping.f90.

5.7 typedefinitionsovertopping::tpovertopping Type Reference

tpOvertopping: structure with overtopping results

Collaboration diagram for typedefinitionsovertopping::tpovertopping:



Public Attributes

- real(kind=wp) z2
 - 2% wave run-up (m)
- real(kind=wp) qo

wave overtopping discharge (m3/m per s)

5.7.1 Detailed Description

tpOvertopping: structure with overtopping results

Definition at line 80 of file typeDefinitionsOvertopping.f90.

5.7.2 Member Data Documentation

5.7.2.1 real(kind=wp) typedefinitionsovertopping::tpovertopping::qo

wave overtopping discharge (m3/m per s)

Definition at line 82 of file typeDefinitionsOvertopping.f90.

5.7.2.2 real(kind=wp) typedefinitionsovertopping::tpovertopping::z2

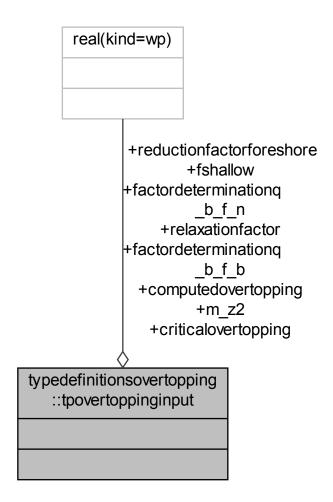
2% wave run-up (m)

Definition at line 81 of file typeDefinitionsOvertopping.f90.

5.8 typedefinitionsovertopping::tpovertoppinginput Type Reference

OvertoppingModelFactors: C-structure with model factors.

Collaboration diagram for typedefinitionsovertopping::tpovertoppinginput:



Public Attributes

```
    real(kind=wp) factordeterminationq_b_f_n
        model factor for non-breaking waves
    real(kind=wp) factordeterminationq_b_f_b
        model factor for breaking waves
    real(kind=wp) m_z2
        model factor describing the uncertainty of 2% runup height
    real(kind=wp) fshallow
        model factor for shallow waves
    real(kind=wp) computedovertopping
        model factor computed overtopping
```

real(kind=wp) criticalovertopping

model factor critical overtopping

real(kind=wp) relaxationfactor

relaxation factor iteration procedure wave runup

real(kind=wp) reductionfactorforeshore = 0.5_wp

reduction factor foreshore

5.8.1 Detailed Description

OvertoppingModelFactors: C-structure with model factors.

Definition at line 68 of file typeDefinitionsOvertopping.f90.

5.8.2 Member Data Documentation

5.8.2.1 real(kind=wp) typedefinitionsovertopping::tpovertoppinginput::computedovertopping

model factor computed overtopping

Definition at line 73 of file typeDefinitionsOvertopping.f90.

5.8.2.2 real(kind=wp) typedefinitionsovertopping::tpovertoppinginput::criticalovertopping

model factor critical overtopping

Definition at line 74 of file typeDefinitionsOvertopping.f90.

5.8.2.3 real(kind=wp) typedefinitionsovertopping::tpovertoppinginput::factordeterminationq_b_f_b

model factor for breaking waves

Definition at line 70 of file typeDefinitionsOvertopping.f90.

 $5.8.2.4 \quad real(kind=wp) \ type definitions over topping:: tpover topping input:: factor determination q_b_f_n$

model factor for non-breaking waves

Definition at line 69 of file typeDefinitionsOvertopping.f90.

5.8.2.5 real(kind=wp) typedefinitionsovertopping::tpovertoppinginput::fshallow

model factor for shallow waves

Definition at line 72 of file typeDefinitionsOvertopping.f90.

 $5.8.2.6 \quad real (kind=wp) \ type definitions over topping:: tpover topping in put:: m_z 2$

model factor describing the uncertainty of 2% runup height

Definition at line 71 of file typeDefinitionsOvertopping.f90.

5.8.2.7 real(kind=wp) typedefinitionsovertopping::tpovertoppinginput::reductionfactorforeshore = 0.5_wp

reduction factor foreshore

Definition at line 76 of file typeDefinitionsOvertopping.f90.

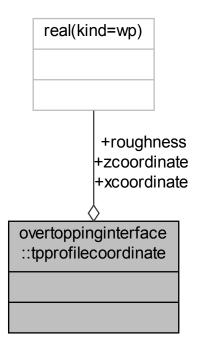
5.8.2.8 real(kind=wp) typedefinitionsovertopping::tpovertoppinginput::relaxationfactor

relaxation factor iteration procedure wave runup

Definition at line 75 of file typeDefinitionsOvertopping.f90.

5.9 overtoppinginterface::tpprofilecoordinate Type Reference

Collaboration diagram for overtoppinginterface::tpprofilecoordinate:



Public Attributes

- real(kind=wp) xcoordinate
 - X-coordinate foreland profile.
- real(kind=wp) zcoordinate
 - Z-coordinate foreland profile.
- real(kind=wp) roughness

Roughness of the area between two points.

5.9.1 Detailed Description

Definition at line 40 of file overtoppingInterface.f90.

5.9.2 Member Data Documentation

5.9.2.1 real(kind=wp) overtoppinginterface::tpprofilecoordinate::roughness

Roughness of the area between two points.

Definition at line 43 of file overtoppingInterface.f90.

5.9.2.2 real(kind=wp) overtoppinginterface::tpprofilecoordinate::xcoordinate

X-coordinate foreland profile.

Definition at line 41 of file overtoppingInterface.f90.

5.9.2.3 real(kind=wp) overtoppinginterface::tpprofilecoordinate::zcoordinate

Z-coordinate foreland profile.

Definition at line 42 of file overtoppingInterface.f90.

Chapter 6

File Documentation

6.1 dllOvertopping.f90 File Reference

Main entry for the dll DikesOvertopping FUNCTIONS/SUBROUTINES exported from dllOvertopping.dll:

Modules

· module dllovertopping

Main entry for the dll DikesOvertopping

Functions/Subroutines

 subroutine, public dllovertopping::calculateqo (load, geometryInput, dikeHeight, modelFactors, overtopping, success, errorText, verbosity, logFile)

Subroutine that calculates the discharge needed for the Z-function DikesOvertopping Wrapper for calculateQoF : convert C-like input structures to Fortran input structures.

subroutine, public dllovertopping::calculateqoj (load, xcoords, ycoords, roughness, normal, npoints, dike
 Height, modelFactors, output, succes, errorMessage)

Subroutine that calculates the discharge Wrapper for calculateQoF: convert basic types to Fortran input structures.

subroutine, public dllovertopping::calculateqof (load, geometryF, dikeHeight, modelFactors, overtopping, success, errorText, logging)

Subroutine that calculates the discharge needed for the Z-function DikesOvertopping.

Subroutine that calculates the Z-function DikesOvertopping based on the discharge calculated with calculateQoF.

subroutine, public dllovertopping::validateinputc (geometryInput, dikeHeight, modelFactors, success, error
 — Text)

Subroutine that validates the geometry Wrapper for ValidateInputF: convert C-like input structures to Fortran input structures.

Subroutine that validates the geometry Wrapper for ValidateInputF: convert basic types to Fortran input structures.

- $\bullet \ \ \text{subroutine, public $dllover topping::} \\ \text{validate input f (geometry F, dike Height, model Factors, error Struct)}$
 - Subroutine that validates the geometry.
- subroutine, public dllovertopping::omkeervariantj (load, xcoords, ycoords, roughness, normal, npoints, givenDischarge, dikeHeight, modelFactors, output, succes, errorMessage)

Wrapper for omkeerVariantF: get dikeHeight by given discharge.

92 File Documentation

subroutine, public dllovertopping::omkeervariantf (load, geometryF, givenDischarge, dikeHeight, model
 —
 Factors, overtopping, success, errorText, logging)

Subroutine with omkeerVariant (get dikeHeight by given discharge)

subroutine, public dllovertopping::setlanguage (lang)

Subroutine that sets the language for error and validation messages.

• subroutine, public dllovertopping::getlanguage (lang)

Subroutine that gets the language for error and validation messages.

• subroutine, public dllovertopping::versionnumber (version)

Subroutine that delivers the version number.

• type(overtoppinggeometrytypef) function dllovertopping::geometry_c_f (geometryInput)

Private subroutine that converts geometry from c-pointer to fortran struct.

 subroutine dllovertopping::input_j_f (x, y, roughness, normal, geometryF, modelFactorsJ, modelFactors→ F, loadJ, loadF)

convert java input to Fortran input

Variables

character, parameter dllovertopping::separationchar = char(9)
 use horizontal tab for separation

6.1.1 Detailed Description

Main entry for the dll DikesOvertopping FUNCTIONS/SUBROUTINES exported from dllOvertopping.dll:

- · calcZValue
- · calculateQo
- · calculateQoF
- · calculateQoJ
- ValidateInputC
- ValidateInputF
- ValidateInputJ
- · omkeerVariantF
- · omkeerVariantJ
- SetLanguage
- · GetLanguage
- versionNumber

6.2 factorModuleOvertopping.f90 File Reference

This file contains a module with functions for the slope angle and influence factors.

Modules

· module factormoduleovertopping

functions for the slope angle and influence factors

Functions/Subroutines

• subroutine, public factormoduleovertopping::calculatetanalpha (h, Hm0, z2, geometry, tanAlpha, succes, errorMessage)

calculateTanAlpha representative slope angle

• subroutine, public factormoduleovertopping::calculategammabeta (Hm0, Tm_10, beta, gammaBeta_z, gammaBeta_o)

calculateGammaBeta influence factor angle of wave attack

• subroutine, public factormoduleovertopping::calculategammaf (h, ksi0, ksi0Limit, gammaB, z2, geometry, gammaF, succes, errorMessage)

calculateGammaF influence factor roughness

subroutine, public factormoduleovertopping::calculategammab (h, Hm0, z2, geometry, gammaB, succes, errorMessage)

calculateGammaB influence factor berms

6.2.1 Detailed Description

This file contains a module with functions for the slope angle and influence factors.

6.3 formulaModuleOvertopping.f90 File Reference

This file contains a module with the core computations for Dikes Overtopping.

Modules

· module formulamoduleovertopping

the core computations for Dikes Overtopping

Functions/Subroutines

• subroutine, public formulamoduleovertopping::calculatewaverunup (Hm0, ksi0, ksi0Limit, gammaB, gammaF, gammaBeta, modelFactors, z2, succes, errorMessage)

calculateWaveRunup: calculate wave runup

• subroutine, public formulamoduleovertopping::calculatewaveovertoppingdischarge (h, Hm0, tanAlpha, gammaB, gammaF, gammaBeta, ksi0, hCrest, modelFactors, Qo, succes, errorMessage)

calculateWaveOvertoppingDischarge: calculate the wave overtopping discharge

• subroutine, public formulamoduleovertopping::calculatewavelength (Tm_10, L0)

calculateWaveLength: calculate the wave length

subroutine, public formulamoduleovertopping::calculatewavesteepness (Hm0, Tm_10, s0, succes, error
 — Message)

calculateWaveSteepness: calculate the wave steepness

subroutine, public formulamoduleovertopping::calculatebreakerparameter (tanAlpha, s0, ksi0, succes, error
 — Message)

calculateBreakerParameter: calculate the breaker parameter

• subroutine, public formulamoduleovertopping::calculateanglewaveattack (phi, psi, beta)

calculateAngleWaveAttack: calculate the angle of wave attack

subroutine, public formulamoduleovertopping::calculatebreakerlimit (gammaB, ksi0Limit, succes, error
 — Message)

calculateBreakerLimit: calculate the breaker limit

• subroutine, public formulamoduleovertopping::adjustinfluencefactors (gammaB, gammaBeta, gammaBetaType, ksi0, ksi0Limit, succes, errorMessage)

94 File Documentation

adjustInfluenceFactors: adjust the influence factors

subroutine formulamoduleovertopping::realrootscubicfunction (a, b, c, d, N, x, succes, errorMessage)

realRootsCubicFunction: calculate the roots of a cubic function

subroutine formulamoduleovertopping::rootsgeneralcubic (a, b, c, d, z, succes, errorMessage)

rootsGeneralCubic: calculate the roots of a generic cubic function

subroutine formulamoduleovertopping::rootsdepressedcubic (p, q, z)

rootsDepressedCubic: calculate the roots of a depressed cubic function

• subroutine formulamoduleovertopping::cubicroots (z, roots)

cubicRoots: calculate the roots of a cubic function

• logical function, public formulamoduleovertopping::isequalreal (x1, x2)

isEqualReal: are two reals (almost) equal

• logical function, public formulamoduleovertopping::isequalzero (x)

isEqualZero: is a real (almost) zero

6.3.1 Detailed Description

This file contains a module with the core computations for Dikes Overtopping.

6.4 geometryModuleOvertopping.f90 File Reference

This file contains a module with the core computations for Dikes Overtopping related to the geometry.

Modules

 module geometrymoduleovertopping core computations related to the geometry

Functions/Subroutines

subroutine, public geometrymoduleovertopping::checkcrosssection (psi, nCoordinates, xCoordinates, y
 —
 Coordinates, roughnessFactors, succes, errorMessage)

checkCrossSection: check cross section

subroutine, public geometrymoduleovertopping::initializegeometry (psi, nCoordinates, xCoordinates, y
 —
 Coordinates, roughnessFactors, geometry, succes, errorMessage)

initializeGeometry: initialize the geometry

 subroutine, public geometrymoduleovertopping::allocatevectorsgeometry (nCoordinates, geometry, succes, errorMessage)

allocateVectorsGeometry: allocate the geometry vectors

• subroutine, public geometrymoduleovertopping::deallocategeometry (geometry)

deallocateGeometry: deallocate the geometry vectors

- subroutine, public geometrymoduleovertopping::calculatesegmentslopes (geometry, succes, errorMessage) calculateSegmentSlopes: calculate the segment slopes
- subroutine, public geometrymoduleovertopping::determinesegmenttypes (geometry, succes, errorMessage) determineSegmentTypes: determine the segment types
- subroutine, public geometrymoduleovertopping::copygeometry (geometry, geometryCopy, succes, error
 — Message)

copyGeometry: copy a geometry structure

• subroutine, public geometrymoduleovertopping::mergesequentialberms (geometry, geometryMergedBerms, succes, errorMessage)

mergeSequentialBerms: merge sequential berms

• subroutine, public geometrymoduleovertopping::adjustnonhorizontalberms (geometry, geometryFlatBerms, succes, errorMessage)

adjustNonHorizontalBerms: adjust non-horizontal berms

 subroutine, public geometrymoduleovertopping::removeberms (geometry, geometryNoBerms, succes, errorMessage)

removeBerms: remove berms

• subroutine, public geometrymoduleovertopping::removedikesegments (geometry, index, geometryAdjusted, succes, errorMessage)

removeDikeSegments: remove dike segments

subroutine, public geometrymoduleovertopping::splitcrosssection (geometry, L0, NwideBerms, geometrysection←
 B, geometrysectionF, succes, errorMessage)

splitCrossSection: split a cross section

subroutine, public geometrymoduleovertopping::calculatehorzlengths (geometry, yLower, yUpper, horz
 — Lengths, succes, errorMessage)

calculateHorzLengths: calculate horizontal lengths

subroutine, public geometrymoduleovertopping::calculatehorzdistance (geometry, yLower, yUpper, dx, succes, errorMessage)

calculateHorzDistance: calculate horizontal distance

• subroutine, public geometrymoduleovertopping::basicgeometrytest (geometryF, success, errorStruct)

basicGeometryTest: test the input geometry (the adjusted geometry is checked elsewhere)

6.4.1 Detailed Description

This file contains a module with the core computations for Dikes Overtopping related to the geometry.

6.5 mainModuleOvertopping.f90 File Reference

This file contains a module with the core computations for Dikes Overtopping.

Modules

· module mainmoduleovertopping

core computations for Dikes Overtopping

Functions/Subroutines

• subroutine, public mainmoduleovertopping::calculateovertopping (geometry, load, modelFactors, overtopping, logging, succes, errorMessage)

calculateOvertopping: calculate the overtopping

• subroutine, public mainmoduleovertopping::calculateovertoppingsection (geometry, h, Hm0, Tm_10, L0, gammaBeta_z, gammaBeta_o, modelFactors, overtopping, logging, succes, errorMessage)

 $calculate Over topping Section: \ calculate \ the \ over topping \ for \ a \ section$

• subroutine, public mainmoduleovertopping::calculatewaveovertopping (geometry, h, Hm0, Tm_10, z2, gammaBeta o, modelFactors, Qo, succes, errorMessage)

calculateWaveOvertopping: calculate wave overtopping

• subroutine mainmoduleovertopping::calculateovertoppingnegativefreeboard (load, geometry, overtopping, succes, errorMessage)

calculateOvertoppingNegativeFreeboard: calculate overtopping in case of negative freeboard

• subroutine, public mainmoduleovertopping::interpolateresultssections (geometry, L0, NwideBerms, overtoppingB, overtoppingF, overtopping, succes, errorMessage)

96 File Documentation

interpolateResultsSections: interpolate results for split cross sections

subroutine, public mainmoduleovertopping::checkinputdata (geometry, load, modelFactors, succes, error
 — Message)

checkInputdata: check the input data

subroutine, public mainmoduleovertopping::checkmodelfactors (modelFactors, dimErrMessage, error
 — Messages, ierr)

checkModelFactors: check the input data

6.5.1 Detailed Description

This file contains a module with the core computations for Dikes Overtopping.

6.6 ModuleLogging.f90 File Reference

Module for steering the extra logging.

Data Types

· type modulelogging::tlogging

TLogging: structure for steering the logging.

Modules

 module modulelogging steering the extra logging

Variables

• integer, parameter modulelogging::maxfilenamelength = 256 maximum length of filename

6.6.1 Detailed Description

Module for steering the extra logging.

6.7 omkeerVariantModule.f90 File Reference

This file contains the omkeerVariant.

Data Types

· type omkeervariantmodule::tomkeer

Modules

· module omkeervariantmodule

Module for the 'omkeerVariant'.

Functions/Subroutines

Subroutine with omkeerVariant.

subroutine omkeervariantmodule::iteratetogivendischargevalidprofile (load, geometry, givenDischarge, dike
 Height, modelFactors, overtopping, logging, success, errorText)

Subroutine with iterateToGivenDischarge, with already checked profile.

subroutine omkeervariantmodule::checkifonberm (geometry, load, modelfactors, overtopping, given
 — Discharge, dikeHeight, omkeerProps, logging, iUp, iLow, dis1, dis2, minDikeHeight, maxDikeHeight, found
 — Value, success, errorText)

Variables

- real(kind=wp), parameter omkeervariantmodule::toldischarge = 1d-3
- real(kind=wp), parameter omkeervariantmodule::toldikeheight = 1d-3
- real(kind=wp), parameter omkeervariantmodule::minzberm = 0.1_wp

6.7.1 Detailed Description

This file contains the omkeerVariant.

6.8 overtoppingInterface.f90 File Reference

This file contains the parameters and types (structs) as part of the interface to and from dllOvertopping.

Data Types

- · type overtoppinginterface::tpprofilecoordinate
- · type overtoppinginterface::overtoppinggeometrytype
- · type overtoppinginterface::overtoppinggeometrytypef

Modules

· module overtoppinginterface

Module for the interface of dllOvertopping.

Variables

• integer, parameter, public overtoppinginterface::varmodelfactorcriticalovertopping = 8

Model factor critical overtopping.

6.8.1 Detailed Description

This file contains the parameters and types (structs) as part of the interface to and from dllOvertopping.

6.9 OvertoppingMessages.f90 File Reference

This file contains the messages in the overtopping dll, in Dutch or English.

98 File Documentation

Modules

· module overtoppingmessages

Module for the messages in the overtopping dll, in Dutch or English.

Functions/Subroutines

subroutine overtoppingmessages::setlanguage (lang)

IDs for the strings in this module:

subroutine overtoppingmessages::getlanguage (lang)

Subroutine that gets the language for error and validation messages.

• character(len=maxmsg) function overtoppingmessages::getovertoppingmessage (ID)

Subroutine that returns a message with the corresponding ID in the current language.

• character(len=maxmsg) function overtoppingmessages::getovertoppingformat (ID)

Subroutine that returns a Fortran format string with the corresponding ID in the current language.

• character(len=maxpar) function overtoppingmessages::getovertoppingparameter (ID)

Subroutine that returns the name of an input parameter with the corresponding ID in the current language.

- subroutine overtoppingmessages::getmsg_calculategammaftolow (message)
- subroutine overtoppingmessages::getmsg_calculategammaftohigh (message)
- subroutine overtoppingmessages::getmsg_calc_influence_roughness (message)
- subroutine overtoppingmessages::getmsg_first_segment_berm (message)
- subroutine overtoppingmessages::getmsg_last_segment_berm (message)
- subroutine overtoppingmessages::getformattoomanyberms (cfmt)
- subroutine overtoppingmessages::getmsgremovenonhorizontalberm (message)

Variables

- integer, parameter, private overtoppingmessages::maxmsg = 128
- integer, parameter, private overtoppingmessages::maxpar =32
- character(len=2), private overtoppingmessages::language = 'NL'

default : Dutch

6.9.1 Detailed Description

This file contains the messages in the overtopping dll, in Dutch or English.

6.10 typeDefinitionsOvertopping.f90 File Reference

This file contains a module with the type definitions for Dikes Overtopping.

Data Types

type typedefinitionsovertopping::tpgeometry

tpGeometry: structure with geometry data

· type typedefinitionsovertopping::tpload

tpLoad: structure with load parameters

· type typedefinitionsovertopping::tpovertoppinginput

OvertoppingModelFactors: C-structure with model factors.

· type typedefinitionsovertopping::tpovertopping

tpOvertopping: structure with overtopping results

Modules

 module typedefinitionsovertopping type definitions for Dikes Overtopping

Variables

- real(kind=wp), parameter typedefinitionsovertopping::frunup1 = 1.65_wp
- real(kind=wp), parameter typedefinitionsovertopping::frunup2 = 4.00 wp
- real(kind=wp), parameter typedefinitionsovertopping::frunup3 = 1.50 wp
- real(kind=wp), parameter typedefinitionsovertopping::xdiff_min = 2.0d-2 minimal value distance between x-coordinates (m)
- real(kind=wp), parameter typedefinitionsovertopping::margindiff = 1.0d-14
 margin for minimal distance (m)
- real(kind=wp), parameter typedefinitionsovertopping::berm_min = 0.0d0
 minimal value gradient berm segment
- real(kind=wp), parameter typedefinitionsovertopping::berm_max = 1.0d0/15
 maximal value gradient berm segment
- real(kind=wp), parameter typedefinitionsovertopping::slope_min = 1.0d0/8
 minimal value gradient slope segment
- real(kind=wp), parameter typedefinitionsovertopping::slope_max = 1.0d0
 maximal value gradient slope segment
- real(kind=wp), parameter typedefinitionsovertopping::margingrad = 0.0025d0
 margin for minimal and maximal gradients
- real(kind=wp), parameter typedefinitionsovertopping::rfactor_min = 0.5d0
 minimal value roughness factor dike segments
- real(kind=wp), parameter typedefinitionsovertopping::rfactor_max = 1.0d0
 maximal value roughness factor dike segments
- real(kind=wp), parameter typedefinitionsovertopping::mz2_min = 0.0d0
 minimal value model factor of 2% runup height
- real(kind=wp), parameter typedefinitionsovertopping::mz2_max = huge(mz2_max)
 maximal value model factor of 2% runup height
- real(kind=wp), parameter typedefinitionsovertopping::fb_min = 0.0d0
 minimal value model factor for breaking waves
- real(kind=wp), parameter typedefinitionsovertopping::fb_max = huge(fB_max)
 maximal value model factor for breaking waves
- real(kind=wp), parameter typedefinitionsovertopping::fn_min = 0.0d0
 minimal value model factor for non-breaking waves
- real(kind=wp), parameter typedefinitionsovertopping::fn_max = huge(fN_max)
 maximal value model factor for non-breaking waves
- real(kind=wp), parameter typedefinitionsovertopping::fs_min = 0.0d0 minimal value model factor for shallow waves
- real(kind=wp), parameter typedefinitionsovertopping::fs_max = huge(fS_max)
 maximal value model factor for shallow waves
- real(kind=wp), parameter typedefinitionsovertopping::foreshore_min = 0.3d0
 minimal value reduction factor foreshore
- real(kind=wp), parameter typedefinitionsovertopping::foreshore_max = 1.0d0
 maximal value reduction factor foreshore
- integer, parameter typedefinitionsovertopping::z2_iter_max1 = 49

 maximal number of iterations for calculation z2 part 1
- integer, parameter typedefinitionsovertopping::z2_iter_max2 = 70

 maximal number of iterations for calculation z2 part 1 & 2
- real(kind=wp), parameter typedefinitionsovertopping::z2_margin = 0.001d0
 margin for convergence criterium calculation z2

100 File Documentation

6.10.1 Detailed Description

This file contains a module with the type definitions for Dikes Overtopping.

6.11 waveRunup.f90 File Reference

This file contains a module with the iteration procedure for 2% wave runup.

Modules

· module waverunup

Iteration procedure for 2% wave runup.

Functions/Subroutines

• subroutine, public waverunup::iterationwaverunup (geometry, h, Hm0, Tm_10, gammaBeta_z, modelFactors, z2, logging, succes, errorMessage)

iterationWaveRunup: iteration for the wave runup

• real(kind=wp) function waverunup::innercalculation (geometry, h, Hm0, gammaBeta_z, modelFactors, z2, s0, geometryFlatBerms, succes, errorMessage)

innerCalculation: inner calculation for the wave runup

- real(kind=wp) function waverunup::determinestartingvalue (i, relaxationFactor, z2_start, z2_end, Hm0)
 determineStartingValue: helper function to find a start value for z2
- integer function waverunup::findsmallestresidu (z2_start, z2_end, n)

findSmallestResidu: helper function to find the smallest residu

subroutine waverunup::convergedwithresidu (z2_start, z2_end, logging)

convergedWithResidu: helper function to handle convergence with a higher residu than expected if logging is enabled, it writes a small message to the logfile

6.11.1 Detailed Description

This file contains a module with the iteration procedure for 2% wave runup.

6.12 zFunctionsOvertopping.f90 File Reference

This file contains the limit state functions for wave overtopping within VTV.

Modules

· module zfunctionsovertopping

Module for the Limit State Functions (Z-functions) for wave overtopping.

Functions/Subroutines

• subroutine, public zfunctionsovertopping::calculateqorto (dikeHeight, modelFactors, overtopping, load, geometry, logging, succes, errorMessage)

Subroutine to calculate the overtopping discharge with the Overtopping dll.

• subroutine, public <u>zfunctionsovertopping::profileinstructure</u> (nrCoordinates, xcoordinates, ycoordinates, dikeHeight, nrCoordsAdjusted, xCoordsAdjusted, zCoordsAdjusted, succes, errorMessage)

Subroutine to fill the profile in a structure and call the adjustment function of the profile due to a desired dike height.

• subroutine zfunctionsovertopping::adjustprofile (nrCoordinates, coordinates, dikeHeight, nrCoordsAdjusted, xCoordsAdjusted, zCoordsAdjusted, succes, errorMessage)

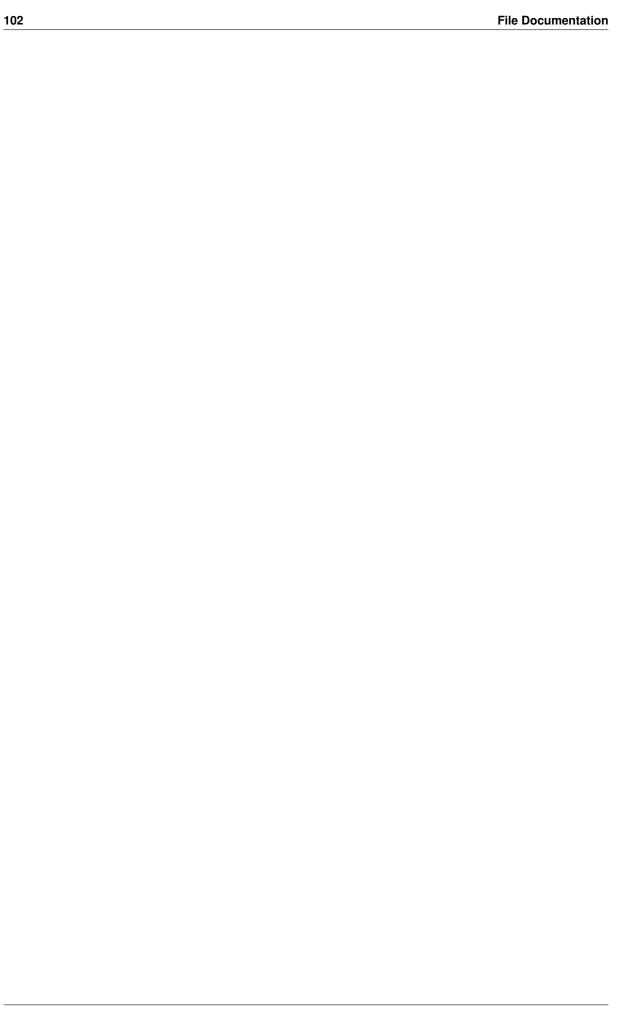
Subroutine adjust the profile due to a desired dike height.

real(kind=wp) function, public zfunctionsovertopping::zfunclogratios (qo, qc, mqo, mqc, success, error
 — Message)

Routine to compute the limit state value by using the logs of the overtopping discharges (computed and desired)

6.12.1 Detailed Description

This file contains the limit state functions for wave overtopping within VTV.



Index

adjustinfluencefactors	calculatewavelength
formulamoduleovertopping, 23	formulamoduleovertopping, 25
adjustnonhorizontalberms	calculatewaveovertopping
geometrymoduleovertopping, 31	mainmoduleovertopping, 46
adjustprofile	calculatewaveovertoppingdischarge
zfunctionsovertopping, 69	formulamoduleovertopping, 25
allocatevectorsgeometry	calculatewaverunup
geometrymoduleovertopping, 32	formulamoduleovertopping, 26
	calculatewavesteepness
basicgeometrytest	formulamoduleovertopping, 27
geometrymoduleovertopping, 33	calczvalue
berm_max	dllovertopping, 11
typedefinitionsovertopping, 61	checkcrosssection
berm_min	geometrymoduleovertopping, 35
typedefinitionsovertopping, 61	checkifonberm
71	omkeervariantmodule, 51
calculateanglewaveattack	checkinputdata
formulamoduleovertopping, 24	mainmoduleovertopping, 47
calculatebreakerlimit	checkmodelfactors
formulamoduleovertopping, 24	mainmoduleovertopping, 48
calculatebreakerparameter	computedovertopping, 46
formulamoduleovertopping, 24	typedefinitionsovertopping::tpovertoppinginput, 87
calculategammab	
factormoduleovertopping, 19	convergedwithresidu
calculategammabeta	waverunup, 65
factormoduleovertopping, 20	copygeometry
	geometrymoduleovertopping, 36
calculategammaf	criticalovertopping
factormoduleovertopping, 20	typedefinitionsovertopping::tpovertoppinginput, 87
calculatehorzdistance	cubicroots
geometrymoduleovertopping, 33	formulamoduleovertopping, 28
calculatehorzlengths	
geometrymoduleovertopping, 34	deallocategeometry
calculateovertopping	geometrymoduleovertopping, 36
mainmoduleovertopping, 44	determinesegmenttypes
calculateovertoppingnegativefreeboard	geometrymoduleovertopping, 38
mainmoduleovertopping, 45	determinestartingvalue
calculateovertoppingsection	waverunup, 65
mainmoduleovertopping, 45	dischargeprofile
calculateqo	omkeervariantmodule::tomkeer, 80
dllovertopping, 8	dllOvertopping.f90, 91
calculateqof	dllovertopping, 7
dllovertopping, 9	calculateqo, 8
calculateqoj	calculateqof, 9
dllovertopping, 10	calculateqoj, 10
calculateqorto	calczvalue, 11
zfunctionsovertopping, 69	geometry_c_f, 11
calculatesegmentslopes	getlanguage, 13
geometrymoduleovertopping, 35	input_j_f, 13
calculatetanalpha	omkeervariantf, 13
factormoduleovertopping, 21	omkeervariantj, 15

104 INDEX

separationchar, 19	fshallow
setlanguage, 16	typedefinitionsovertopping::tpovertoppinginput, 87
validateinputc, 16	
validateinputf, 17	geometry_c_f
validateinputj, 18	dllovertopping, 11
versionnumber, 18	geometryModuleOvertopping.f90, 94
	geometrymoduleovertopping, 31
factorModuleOvertopping.f90, 92	adjustnonhorizontalberms, 31
factordeterminationq_b_f_b	allocatevectorsgeometry, 32
typedefinitionsovertopping::tpovertoppinginput, 87	basicgeometrytest, 33
factordeterminationq_b_f_n	calculatehorzdistance, 33
typedefinitionsovertopping::tpovertoppinginput, 87	calculatehorzlengths, 34
factormoduleovertopping, 19	calculatesegmentslopes, 35
calculategammab, 19	checkcrosssection, 35
calculategammabeta, 20	copygeometry, 36
calculategammaf, 20	deallocategeometry, 36
calculatetanalpha, 21	determinesegmenttypes, 38
fb_max	initializegeometry, 38
typedefinitionsovertopping, 62	mergesequentialberms, 40
fb_min	removeberms, 41
typedefinitionsovertopping, 62	removedikesegments, 42
filename	splitcrosssection, 42
modulelogging::tlogging, 79	getformattoomanyberms
findsmallestresidu	overtoppingmessages, 57
waverunup, 65	getlanguage
fn_max	dllovertopping, 13
typedefinitionsovertopping, 62	overtoppingmessages, 57
fn_min	getmsg_calc_influence_roughness
typedefinitionsovertopping, 62	overtoppingmessages, 57
foreshore_max	getmsg_calculategammaftohigh
typedefinitionsovertopping, 62	overtoppingmessages, 57
foreshore_min	getmsg_calculategammaftolow
typedefinitionsovertopping, 62	overtoppingmessages, 57
formulaModuleOvertopping.f90, 93	getmsg_first_segment_berm
formulamoduleovertopping, 22	overtoppingmessages, 58
adjustinfluencefactors, 23	getmsg last segment berm
calculateanglewaveattack, 24	
calculatebreakerlimit, 24	overtoppingmessages, 58
calculatebreakerparameter, 24	getmsgremovenonhorizontalberm
calculatewavelength, 25	overtoppingmessages, 58
calculatewaveovertoppingdischarge, 25	getovertoppingformat
calculatewaverunup, 26	overtoppingmessages, 58
calculatewavesteepness, 27	getovertoppingmessage
cubicroots, 28	overtoppingmessages, 59
isequalreal, 28	getovertoppingparameter
isequalzero, 28	overtoppingmessages, 59
•	
realrootscubicfunction, 29	h
rootsdepressedcubic, 29	typedefinitionsovertopping::tpload, 84
rootsgeneralcubic, 30	hm0
frunup1	typedefinitionsovertopping::tpload, 84
typedefinitionsovertopping, 62	and the second s
frunup2	initializegeometry
typedefinitionsovertopping, 62	geometrymoduleovertopping, 38
frunup3	innercalculation
typedefinitionsovertopping, 62	waverunup, 66
fs_max	input_j_f
typedefinitionsovertopping, 62	dllovertopping, 13
fs_min	interpolateresultssections
typedefinitionsovertopping, 63	mainmoduleovertopping, 49

INDEX 105

isberm	overtoppinginterface::overtoppinggeometrytype,
omkeervariantmodule::tomkeer, 80	76
isequalreal	overtoppinginterface::overtoppinggeometrytypef,
formulamoduleovertopping, 28	77
isequalzero	npoints
formulamoduleovertopping, 28	overtoppinginterface::overtoppinggeometrytype,
isvalidz	76
omkeervariantmodule::tomkeer, 80	overtoppinginterface::overtoppinggeometrytypef,
iteratetogivendischarge	77
omkeervariantmodule, 52	amkaarVariantMadula f00, 06
iteratetogivendischargevalidprofile	omkeerVariantModule.f90, 96 omkeervariantf
omkeervariantmodule, 54	
iterationwaverunup	dllovertopping, 13
waverunup, 67	omkeervariantj
	dllovertopping, 15
language	omkeervariantmodule, 50
overtoppingmessages, 60	checkifonberm, 51
	iteratetogivendischarge, 52
m_z2	iteratetogivendischargevalidprofile, 54
typedefinitionsovertopping::tpovertoppinginput, 87	minzberm, 55
mainModuleOvertopping.f90, 95	toldikeheight, 55
mainmoduleovertopping, 43	toldischarge, 55
calculateovertopping, 44	omkeervariantmodule::tomkeer, 79
calculateovertoppingnegativefreeboard, 45	dischargeprofile, 80
calculateovertoppingsection, 45	isberm, 80
calculatewaveovertopping, 46	isvalidz, 80
checkinputdata, 47	zprofile, 80
checkmodelfactors, 48	overtoppingInterface.f90, 97
interpolateresultssections, 49	OvertoppingMessages.f90, 97
margindiff	overtoppinginterface, 56
typedefinitionsovertopping, 63	varmodelfactorcriticalovertopping, 56
margingrad	overtoppinginterface::overtoppinggeometrytype, 75
typedefinitionsovertopping, 63	normal, 76
maxfilenamelength	npoints, 76
_	roughness, 76
modulelogging, 50	xcoords, 76
maxmsg	ycoords, 76
overtoppingmessages, 60	
maxpar	overtoppinginterface::overtoppinggeometrytypef, 77
overtoppingmessages, 60	normal, 77
mergesequentialberms	npoints, 77
geometrymoduleovertopping, 40	roughness, 77
minzberm	xcoords, 78
omkeervariantmodule, 55	ycoords, 78
ModuleLogging.f90, 96	overtoppinginterface::tpprofilecoordinate, 88
modulelogging, 49	roughness, 88
maxfilenamelength, 50	xcoordinate, 89
modulelogging::tlogging, 78	zcoordinate, 89
filename, 79	overtoppingmessages, 56
verbosity, 79	getformattoomanyberms, 57
mz2_max	getlanguage, 57
typedefinitionsovertopping, 63	getmsg_calc_influence_roughness, 57
mz2_min	getmsg_calculategammaftohigh, 57
typedefinitionsovertopping, 63	getmsg_calculategammaftolow, 57
у, решениценте не	getmsg_first_segment_berm, 58
nbermsegments	getmsg_last_segment_berm, 58
typedefinitionsovertopping::tpgeometry, 82	getmsgremovenonhorizontalberm, 58
ncoordinates	getovertoppingformat, 58
typedefinitionsovertopping::tpgeometry, 82	getovertoppingmessage, 59
normal	getovertoppingparameter, 59
· · · · · · · · · · · · · · · · · · ·	gotovo toppingparamotor, ov

106 INDEX

	language, 60	typedefinitionsovertopping::tpload, 84
	maxmsg, 60	toldikeheight
	maxpar, 60	omkeervariantmodule, 55
	setlanguage, 59	toldischarge
		omkeervariantmodule, 55
phi		typeDefinitionsOvertopping.f90, 98
	typedefinitionsovertopping::tpload, 84	typedefinitionsovertopping, 60
profi	leinstructure	berm_max, 61
	zfunctionsovertopping, 71	berm_min, 61
psi		fb_max, 62
	typedefinitionsovertopping::tpgeometry, 82	fb_min, 62
		fn_max, 62
qo		fn_min, 62
	typedefinitionsovertopping::tpovertopping, 85	foreshore_max, 62
		foreshore_min, 62
reair	rootscubicfunction	frunup1, 62
	formulamoduleovertopping, 29	frunup2, 62
reau	ctionfactorforeshore	frunup3, 62
	typedefinitionsovertopping::tpovertoppinginput, 87	fs_max, 62
relax	cationfactor	fs_min, 63
	typedefinitionsovertopping::tpovertoppinginput, 87	margindiff, 63
remo	oveberms	margingrad, 63
	geometrymoduleovertopping, 41	mz2_max, 63
remo	ovedikesegments	mz2_min, 63
	geometrymoduleovertopping, 42	rfactor_max, 63
rtact	or_max	rfactor_min, 63
	typedefinitionsovertopping, 63	slope_max, 63
rfact	or_min	slope_min, 63
	typedefinitionsovertopping, 63	xdiff_min, 64
roots	sdepressedcubic	z2_iter_max1, 64
	formulamoduleovertopping, 29	z2_iter_max2, 64
roots	sgeneralcubic	z2_margin, 64
	formulamoduleovertopping, 30	typedefinitionsovertopping::tpgeometry, 80
roug	hness	nbermsegments, 82
	overtoppinginterface::overtoppinggeometrytype,	ncoordinates, 82
	76	psi, 82
	overtoppinginterface::overtoppinggeometrytypef,	roughnessfactors, 82
	77	segmentslopes, 82
	overtoppinginterface::tpprofilecoordinate, 88	segmenttypes, 82
roug	hnessfactors	xcoorddiff, 82
	typedefinitionsovertopping::tpgeometry, 82	xcoordinates, 82
		ycoorddiff, 83
segr	nentslopes	ycoordinates, 83
	typedefinitionsovertopping::tpgeometry, 82	typedefinitionsovertopping::tpload, 83
segr	nenttypes	h, 84
	typedefinitionsovertopping::tpgeometry, 82	hm0, 84
sepa	arationchar	
	dllovertopping, 19	phi, 84 tm 10, 84
setla	inguage	— · ·
	dllovertopping, 16	typedefinitionsovertopping::tpovertopping, 84
	overtoppingmessages, 59	qo, 85
slop	e_max	z2, 85
	typedefinitionsovertopping, 63	typedefinitionsovertopping::tpovertoppinginput, 86
slop	e_min	computedovertopping, 87
	typedefinitionsovertopping, 63	criticalovertopping, 87
splite	crosssection	factordeterminationq_b_f_b, 87
	geometrymoduleovertopping, 42	factordeterminationq_b_f_n, 87
	10	fshallow, 87
tm_1	10	m_z2, 87

```
reductionfactorforeshore, 87
                                                          zfunclogratios
     relaxationfactor, 87
                                                               zfunctionsovertopping, 72
                                                          zfunctionsovertopping, 68
validateinputc
                                                               adjustprofile, 69
     dllovertopping, 16
                                                               calculateqorto, 69
validateinputf
                                                               profileinstructure, 71
     dllovertopping, 17
                                                               zfunclogratios, 72
validateinputj
                                                          zprofile
     dllovertopping, 18
                                                               omkeervariantmodule::tomkeer, 80
varmodelfactorcriticalovertopping
     overtoppinginterface, 56
verbosity
     modulelogging::tlogging, 79
versionnumber
     dllovertopping, 18
waveRunup.f90, 100
waverunup, 64
     convergedwithresidu, 65
     determinestartingvalue, 65
     findsmallestresidu, 65
     innercalculation, 66
     iterationwaverunup, 67
xcoorddiff
     typedefinitionsovertopping::tpgeometry, 82
xcoordinate
     overtoppinginterface::tpprofilecoordinate, 89
xcoordinates
     typedefinitionsovertopping::tpgeometry, 82
xcoords
     overtoppinginterface::overtoppinggeometrytype,
     overtoppinginterface::overtoppinggeometrytypef,
xdiff min
     typedefinitionsovertopping, 64
ycoorddiff
     typedefinitionsovertopping::tpgeometry, 83
ycoordinates
     typedefinitionsovertopping::tpgeometry, 83
ycoords
     overtoppinginterface::overtoppinggeometrytype,
     overtoppinginterface::overtoppinggeometrytypef,
          78
z2
     typedefinitionsovertopping::tpovertopping, 85
z2_iter_max1
     typedefinitionsovertopping, 64
z2 iter max2
     typedefinitionsovertopping, 64
z2 margin
     typedefinitionsovertopping, 64
zFunctionsOvertopping.f90, 100
zcoordinate
     overtoppinginterface::tpprofilecoordinate, 89
```