

# SWASHES

1.01.00

Generated by Doxygen 1.6.3

Mon Feb 6 10:27:17 2012



# Contents

<b>1</b>	<b>Class Index</b>	<b>1</b>
1.1	Class Hierarchy . . . . .	1
<b>2</b>	<b>Class Index</b>	<b>3</b>
2.1	Class List . . . . .	3
<b>3</b>	<b>File Index</b>	<b>5</b>
3.1	File List . . . . .	5
<b>4</b>	<b>Class Documentation</b>	<b>7</b>
4.1	bedload Class Reference . . . . .	7
4.1.1	Detailed Description . . . . .	7
4.1.2	Constructor & Destructor Documentation . . . . .	8
4.1.2.1	bedload . . . . .	8
4.1.2.2	~bedload . . . . .	8
4.1.3	Member Function Documentation . . . . .	8
4.1.3.1	compute . . . . .	8
4.1.3.2	param . . . . .	8
4.1.3.3	paramwarning . . . . .	8
4.2	bump Class Reference . . . . .	9
4.2.1	Detailed Description . . . . .	9
4.2.2	Constructor & Destructor Documentation . . . . .	10
4.2.2.1	bump . . . . .	10
4.2.2.2	~bump . . . . .	10
4.2.3	Member Function Documentation . . . . .	10
4.2.3.1	abcd . . . . .	10
4.2.3.2	compute . . . . .	10
4.2.3.3	determinant . . . . .	10
4.2.3.4	function . . . . .	11

4.2.3.5	height	11
4.2.3.6	p	11
4.2.3.7	param	11
4.2.3.8	q	11
4.2.3.9	RHJump	12
4.3	choice_solution Class Reference	13
4.3.1	Detailed Description	13
4.3.2	Constructor & Destructor Documentation	13
4.3.2.1	choice_solution	13
4.3.2.2	~choice_solution	13
4.3.3	Member Function Documentation	13
4.3.3.1	compute	13
4.4	dam_break Class Reference	14
4.4.1	Detailed Description	14
4.4.2	Constructor & Destructor Documentation	14
4.4.2.1	dam_break	14
4.4.2.2	~dam_break	14
4.4.3	Member Function Documentation	15
4.4.3.1	compute	15
4.4.3.2	function	15
4.4.3.3	param	15
4.5	Dressler_dam Class Reference	16
4.5.1	Detailed Description	16
4.5.2	Constructor & Destructor Documentation	16
4.5.2.1	Dressler_dam	16
4.5.2.2	~Dressler_dam	16
4.5.3	Member Function Documentation	17
4.5.3.1	compute	17
4.5.3.2	param	17
4.6	MacDonald_like Class Reference	18
4.6.1	Detailed Description	18
4.6.2	Constructor & Destructor Documentation	18
4.6.2.1	MacDonald_like	18
4.6.2.2	~MacDonald_like	19
4.6.3	Member Function Documentation	19
4.6.3.1	compute	19

---

4.6.3.2	Delta_topo_Darcy_Weisbach	19
4.6.3.3	Delta_topo_Manning	19
4.6.3.4	param	19
4.7	MacDonald_like_diffus Class Reference	20
4.7.1	Detailed Description	20
4.7.2	Constructor & Destructor Documentation	20
4.7.2.1	MacDonald_like_diffus	20
4.7.2.2	~MacDonald_like_diffus	20
4.7.3	Member Function Documentation	21
4.7.3.1	compute	21
4.7.3.2	Delta_topo_diffus	21
4.7.3.3	param	21
4.8	MacDonaldB1 Class Reference	22
4.8.1	Detailed Description	22
4.8.2	Constructor & Destructor Documentation	22
4.8.2.1	MacDonaldB1	22
4.8.2.2	~MacDonaldB1	22
4.8.3	Member Function Documentation	23
4.8.3.1	compute	23
4.8.3.2	Delta_topo	23
4.8.3.3	param	23
4.9	MacDonaldB2 Class Reference	24
4.9.1	Detailed Description	24
4.9.2	Constructor & Destructor Documentation	24
4.9.2.1	MacDonaldB2	24
4.9.2.2	~MacDonaldB2	24
4.9.3	Member Function Documentation	25
4.9.3.1	compute	25
4.9.3.2	Delta_topo	25
4.9.3.3	param	25
4.10	parameters Class Reference	26
4.10.1	Detailed Description	26
4.10.2	Constructor & Destructor Documentation	26
4.10.2.1	parameters	26
4.10.2.2	~parameters	26
4.10.3	Member Function Documentation	26

4.10.3.1	<code>get_choice</code>	26
4.10.3.2	<code>get_choicedim</code>	27
4.10.3.3	<code>get_choicedomain</code>	27
4.10.3.4	<code>get_choicetype</code>	27
4.10.3.5	<code>get_Nxex</code>	27
4.10.3.6	<code>get_Nyex</code>	27
4.10.3.7	<code>help</code>	27
4.10.3.8	<code>setparameters</code>	27
4.10.4	Member Data Documentation	27
4.10.4.1	<code>choice</code>	27
4.10.4.2	<code>choicedim</code>	27
4.10.4.3	<code>choicedomain</code>	27
4.10.4.4	<code>choicetype</code>	27
4.10.4.5	<code>Nx_ex</code>	27
4.10.4.6	<code>Ny_ex</code>	28
4.11	Sampson Class Reference	29
4.11.1	Detailed Description	29
4.11.2	Constructor & Destructor Documentation	29
4.11.2.1	<code>Sampson</code>	29
4.11.2.2	<code>~Sampson</code>	29
4.11.3	Member Function Documentation	30
4.11.3.1	<code>compute</code>	30
4.11.3.2	<code>param</code>	30
4.12	solution Class Reference	31
4.12.1	Detailed Description	32
4.12.2	Constructor & Destructor Documentation	32
4.12.2.1	<code>solution</code>	32
4.12.2.2	<code>~solution</code>	33
4.12.3	Member Function Documentation	33
4.12.3.1	<code>allocation</code>	33
4.12.3.2	<code>compute</code>	33
4.12.3.3	<code>desallocation</code>	33
4.12.3.4	<code>head</code>	33
4.12.3.5	<code>savefinal2D</code>	33
4.12.3.6	<code>savefinalcritical</code>	33
4.12.3.7	<code>savefinalcriticalinit</code>	33

---

4.12.3.8	savefinalmu	34
4.12.4	Member Data Documentation	34
4.12.4.1	dx_ex	34
4.12.4.2	dy_ex	34
4.12.4.3	hex	34
4.12.4.4	l	34
4.12.4.5	L	34
4.12.4.6	Nx_ex	34
4.12.4.7	Ny_ex	34
4.12.4.8	qex	34
4.12.4.9	T	34
4.12.4.10	uex	34
4.12.4.11	xex	35
4.12.4.12	yex	35
4.12.4.13	zex	35
4.13	Thacker Class Reference	36
4.13.1	Detailed Description	36
4.13.2	Constructor & Destructor Documentation	36
4.13.2.1	Thacker	36
4.13.2.2	~Thacker	36
4.13.3	Member Function Documentation	37
4.13.3.1	compute	37
4.13.3.2	param	37
4.14	Thacker2D Class Reference	38
4.14.1	Detailed Description	38
4.14.2	Constructor & Destructor Documentation	38
4.14.2.1	Thacker2D	38
4.14.2.2	~Thacker2D	38
4.14.3	Member Function Documentation	39
4.14.3.1	compute	39
4.14.3.2	param	39
<b>5</b>	<b>File Documentation</b>	<b>41</b>
5.1	Headers/bedload.hpp File Reference	41
5.1.1	Detailed Description	41
5.1.2	Define Documentation	42
5.1.2.1	Class_bedload	42

5.2	Headers/bump.hpp File Reference	43
5.2.1	Detailed Description	43
5.3	Headers/choice_solution.hpp File Reference	44
5.3.1	Detailed Description	44
5.3.2	Define Documentation	44
5.3.2.1	Class_choice_solution	44
5.4	Headers/dam_break.hpp File Reference	45
5.4.1	Detailed Description	45
5.5	Headers/Dressler_dam.hpp File Reference	46
5.5.1	Detailed Description	46
5.6	Headers/MacDonald_like.hpp File Reference	47
5.6.1	Detailed Description	47
5.7	Headers/MacDonald_like_diffus.hpp File Reference	48
5.7.1	Detailed Description	48
5.8	Headers/MacDonaldB1.hpp File Reference	49
5.8.1	Detailed Description	49
5.9	Headers/MacDonaldB2.hpp File Reference	50
5.9.1	Detailed Description	50
5.10	Headers/misc.hpp File Reference	51
5.10.1	Define Documentation	51
5.10.1.1	grav	51
5.10.1.2	grav_dem	51
5.10.1.3	max	51
5.10.1.4	min	51
5.10.1.5	PI	52
5.10.1.6	version	52
5.10.1.7	zero	52
5.10.2	Typedef Documentation	52
5.10.2.1	SCALAR	52
5.10.2.2	TAB	52
5.11	Headers/parameters.hpp File Reference	53
5.11.1	Detailed Description	53
5.12	Headers/Sampson.hpp File Reference	54
5.12.1	Detailed Description	54
5.12.2	Define Documentation	54
5.12.2.1	Class_sampson	54



---

5.13	Headers/solution.hpp File Reference . . . . .	55
5.13.1	Detailed Description . . . . .	55
5.14	Headers/Thacker.hpp File Reference . . . . .	56
5.14.1	Detailed Description . . . . .	56
5.15	Headers/Thacker2D.hpp File Reference . . . . .	57
5.15.1	Detailed Description . . . . .	57
5.16	Sources/bedload.cpp File Reference . . . . .	58
5.17	Sources/bump.cpp File Reference . . . . .	59
5.18	Sources/choice_solution.cpp File Reference . . . . .	60
5.19	Sources/dam_break.cpp File Reference . . . . .	61
5.20	Sources/Dressler_dam.cpp File Reference . . . . .	62
5.21	Sources/MacDonald_like.cpp File Reference . . . . .	63
5.22	Sources/MacDonald_like_diffus.cpp File Reference . . . . .	64
5.23	Sources/MacDonaldB1.cpp File Reference . . . . .	65
5.24	Sources/MacDonaldB2.cpp File Reference . . . . .	66
5.25	Sources/parameters.cpp File Reference . . . . .	67
5.26	Sources/Sampson.cpp File Reference . . . . .	68
5.27	Sources/solution.cpp File Reference . . . . .	69
5.28	Sources/swashes.cpp File Reference . . . . .	70
5.28.1	Detailed Description . . . . .	70
5.28.2	Function Documentation . . . . .	70
5.28.2.1	main . . . . .	70
5.29	Sources/Thacker.cpp File Reference . . . . .	71
5.30	Sources/Thacker2D.cpp File Reference . . . . .	72



# Chapter 1

## Class Index

### 1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

choice_solution . . . . .	13
parameters . . . . .	26
solution . . . . .	31
bedload . . . . .	7
bump . . . . .	9
dam_break . . . . .	14
Dressler_dam . . . . .	16
MacDonald_like . . . . .	18
MacDonald_like_diffus . . . . .	20
MacDonaldB1 . . . . .	22
MacDonaldB2 . . . . .	24
Sampson . . . . .	29
Thacker . . . . .	36
Thacker2D . . . . .	38



# Chapter 2

## Class Index

### 2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

<a href="#">bedload</a> (Class which performs bedload (Exner) analytic solution with Grass or MPM equations )	7
<a href="#">bump</a> (Class which allows to perform the bump analytic solutions ) . . . . .	9
<a href="#">choice_solution</a> (Class which allows to choose the analytic solution ) . . . . .	13
<a href="#">dam_break</a> (Class which allows to perform the dam break analytic solution with wet and dry soil )	14
<a href="#">Dressler_dam</a> (Class which allows to perform the Dressler dam break analytic solution ) . . . .	16
<a href="#">MacDonald_like</a> (Class which allows to perform the MacDonald like analytic solution ) . . . . .	18
<a href="#">MacDonald_like_diffus</a> (Class which allows to perform the MacDonald like analytic solution with diffusion ) . . . . .	20
<a href="#">MacDonaldB1</a> (Class which allows to perform the MacDonald PSEUDO 2D analytic solution ) .	22
<a href="#">MacDonaldB2</a> (Class which allows to perform the MacDonald PSEUDO 2D analytic solution ) .	24
<a href="#">parameters</a> (Class that defines the common parameters ) . . . . .	26
<a href="#">Sampson</a> (Class which allows to perform the <a href="#">Sampson</a> analytic solution ) . . . . .	29
<a href="#">solution</a> (Class which allows to perform the analytic solutions ) . . . . .	31
<a href="#">Thacker</a> (Class which allows to perform the <a href="#">Thacker</a> analytic solution ) . . . . .	36
<a href="#">Thacker2D</a> (Class which allows to perform the <a href="#">Thacker</a> 2D analytic solution ) . . . . .	38



# Chapter 3

## File Index

### 3.1 File List

Here is a list of all files with brief descriptions:

Headers/bedload.hpp (Performs bedload (Exner) analytic solutions ) . . . . .	41
Headers/bump.hpp (Performs the bump analytic solutions ) . . . . .	43
Headers/choice_solution.hpp (Allows to choose the analytic solution ) . . . . .	44
Headers/dam_break.hpp (Performs the dam break analytic solution ) . . . . .	45
Headers/Dressler_dam.hpp (Performs classical and modified (personnal communication with Valerio Caleffi, illustration in Valiani et al. 1999) Dressler analytic solution ) . . . . .	46
Headers/MacDonald_like.hpp (Performs the MacDonald like analytic solutions ) . . . . .	47
Headers/MacDonald_like_diffus.hpp (Performs the MacDonald like analytic solutions with diffusion ) . . . . .	48
Headers/MacDonaldB1.hpp (Performs the MacDonald PSEUDO 2D analytic solutions ) . . . . .	49
Headers/MacDonaldB2.hpp (Performs the MacDonald PSEUDO 2D analytic solutions ) . . . . .	50
Headers/misc.hpp . . . . .	51
Headers/parameters.hpp (Defines the common parameters ) . . . . .	53
Headers/Sampson.hpp (Performs the Sampson analytic solution ) . . . . .	54
Headers/solution.hpp (Performs the analytic solutions ) . . . . .	55
Headers/Thacker.hpp (Performs the Thacker analytic solution ) . . . . .	56
Headers/Thacker2D.hpp (Performs the Thacker 2D analytic solutions ) . . . . .	57
Sources/bedload.cpp . . . . .	58
Sources/bump.cpp . . . . .	59
Sources/choice_solution.cpp . . . . .	60
Sources/dam_break.cpp . . . . .	61
Sources/Dressler_dam.cpp . . . . .	62
Sources/MacDonald_like.cpp . . . . .	63
Sources/MacDonald_like_diffus.cpp . . . . .	64
Sources/MacDonaldB1.cpp . . . . .	65
Sources/MacDonaldB2.cpp . . . . .	66
Sources/parameters.cpp . . . . .	67
Sources/Sampson.cpp . . . . .	68
Sources/solution.cpp . . . . .	69
Sources/swashes.cpp (Main function. Declares the solution and calculates the chosen analytic solution for 1D Shallow Water equations ) . . . . .	70
Sources/Thacker.cpp . . . . .	71
Sources/Thacker2D.cpp . . . . .	72





# Chapter 4

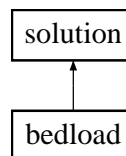
## Class Documentation

### 4.1 bedload Class Reference

class which performs bedload (Exner) analytic solution with Grass or MPM equations

```
#include <bedload.hpp>
```

Inheritance diagram for bedload:



#### Public Member Functions

- [bedload](#) ([parameters](#) &)  
*Constructor.*
- virtual [~bedload](#) ()  
*Destructor.*
- void [compute](#) ()  
*Virtual method which is specific to each analytic solution.*
- void [param](#) ([SCALAR](#), [SCALAR](#), int, [SCALAR](#), [SCALAR](#), [SCALAR](#), [SCALAR](#), [SCALAR](#), [SCALAR](#), [SCALAR](#), [SCALAR](#), [SCALAR](#), [SCALAR](#), [SCALAR](#), [SCALAR](#))
- void [paramwarning](#) ()

#### 4.1.1 Detailed Description

class which performs bedload (Exner) analytic solution with Grass or MPM equations

Definition at line 61 of file bedload.hpp.

## 4.1.2 Constructor & Destructor Documentation

### 4.1.2.1 `bedload::bedload` (parameters & *par*)

Constructor.

Definition at line 48 of file `bedload.cpp`.

### 4.1.2.2 `bedload::~~bedload` () [`virtual`]

Destructor.

Definition at line 130 of file `bedload.cpp`.

## 4.1.3 Member Function Documentation

### 4.1.3.1 `void bedload::compute` () [`virtual`]

Virtual method which is specific to each analytic solution.

Implements [solution](#).

Definition at line 135 of file `bedload.cpp`.

### 4.1.3.2 `void bedload::param` (SCALAR *L*, SCALAR *dx\_ex*, int *Nx\_ex*, SCALAR *T*, SCALAR *uexl*, SCALAR *hexl*, SCALAR *z0l*, SCALAR *zexl*, SCALAR *uexr*, SCALAR *hexr*, SCALAR *z0r*, SCALAR *zexr*, SCALAR *alpha*, SCALAR *beta*, SCALAR *A*, SCALAR *q*, SCALAR *C*, SCALAR *p*)

Definition at line 148 of file `bedload.cpp`.

### 4.1.3.3 `void bedload::paramwarning` ()

Definition at line 164 of file `bedload.cpp`.

The documentation for this class was generated from the following files:

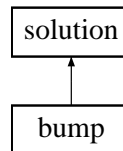
- [Headers/bedload.hpp](#)
- [Sources/bedload.cpp](#)

## 4.2 bump Class Reference

class which allows to perform the bump analytic solutions

```
#include <bump.hpp>
```

Inheritance diagram for bump:



### Public Member Functions

- `bump` (parameters &)  
*Constructor.*
- virtual `~bump` ()  
*Destructor.*
- void `compute` ()  
*Virtual method which is specific to each analytic solution.*
- `SCALAR` function (`SCALAR`, `SCALAR`, `SCALAR`)
- `SCALAR` p (`SCALAR` a, `SCALAR` b, `SCALAR` c)  
*coefficient Cardano method*
- `SCALAR` q (`SCALAR` a, `SCALAR` b, `SCALAR` c, `SCALAR` d)  
*coefficient Cardano method*
- `SCALAR` determinant (`SCALAR` p, `SCALAR` q)  
*Determinant in the Cardano method/related to number of roots.*
- `SCALAR` height (`SCALAR` p, `SCALAR` q, `SCALAR` a, `SCALAR` b, `SCALAR` hnear)  
*Computation of the 3rd order polynomia roots.*
- void `abcd` (`SCALAR` q0, `SCALAR` hfin, `SCALAR` zbx, `SCALAR` zbfm, `SCALAR` &a, `SCALAR` &b, `SCALAR` &c, `SCALAR` &d)  
*Enters the coefficients of the 3rd order polynomia we want to solve.*
- `SCALAR` RHJump (`SCALAR` hplus, `SCALAR` hmois, `SCALAR` q)  
*steady state RH relation (to check if zero at the shock)*
- void `param` (`SCALAR`, `SCALAR`, int)

### 4.2.1 Detailed Description

class which allows to perform the bump analytic solutions

Definition at line 61 of file bump.hpp.

## 4.2.2 Constructor & Destructor Documentation

### 4.2.2.1 `bump::bump` (parameters & *par*)

Constructor.

Definition at line 48 of file bump.cpp.

### 4.2.2.2 `bump::~bump` () [`virtual`]

Destructor.

Definition at line 147 of file bump.cpp.

## 4.2.3 Member Function Documentation

### 4.2.3.1 `void bump::abcd` (SCALAR *q0*, SCALAR *hfin*, SCALAR *zbx*, SCALAR *zbfm*, SCALAR & *a*, SCALAR & *b*, SCALAR & *c*, SCALAR & *d*)

Enters the coefficients of the 3rd order polynomia we want to solve.

#### Parameters

*q0* : Inflow

*hfin* : exit height

*zbx* : bottom of the current cell

*zbfm* : exit bottom

*a* : coeff x3

*b* : coeff x2

*c* : coeff x

*d* : coeff 1

Definition at line 300 of file bump.cpp.

### 4.2.3.2 `void bump::compute` () [`virtual`]

Virtual method which is specific to each analytic solution.

Research of the limit x

PB !!

Computation of the height

Implements [solution](#).

Definition at line 150 of file bump.cpp.

### 4.2.3.3 `SCALAR bump::determinant` (SCALAR *p*, SCALAR *q*)

Determinant in the Cardano method/related to number of roots.

**Parameters**

$p$  : computed in function p

$q$  : computed in function q

Definition at line 246 of file bump.cpp.

**4.2.3.4 SCALAR bump::function (SCALAR, SCALAR, SCALAR)****4.2.3.5 SCALAR bump::height (SCALAR  $p$ , SCALAR  $q$ , SCALAR  $a$ , SCALAR  $b$ , SCALAR  $hnear$ )**

Computation of the 3rd order polynomia roots.

**Parameters**

$p$  : cardano coeff

$q$  : cardano coeff

$a$  :

$b$  :

$hnear$  : height of the previous or following cell (depending on the height computation direction)

Definition at line 252 of file bump.cpp.

**4.2.3.6 SCALAR bump::p (SCALAR  $a$ , SCALAR  $b$ , SCALAR  $c$ )**

coefficient Cardano method

**Parameters**

$a,b,c$  : coeff of the 3rd order polynomia

Definition at line 234 of file bump.cpp.

**4.2.3.7 void bump::param (SCALAR  $L$ , SCALAR  $dx_{ex}$ , int  $Nx_{ex}$ )**

Definition at line 313 of file bump.cpp.

**4.2.3.8 SCALAR bump::q (SCALAR  $a$ , SCALAR  $b$ , SCALAR  $c$ , SCALAR  $d$ )**

coefficient Cardano method

**Parameters**

$a,b,c,d$  : coeff of the 3rd order polynomia

Definition at line 240 of file bump.cpp.

#### 4.2.3.9 SCALAR bump::RHJump (SCALAR *hplus*, SCALAR *hminus*, SCALAR *q*)

steady state RH relation (to check if zero at the shock)

##### Parameters

- hplus* : height right side
- hminus* : height left side
- q* : flow

Definition at line 308 of file bump.cpp.

The documentation for this class was generated from the following files:

- [Headers/bump.hpp](#)
- [Sources/bump.cpp](#)

## 4.3 choice\_solution Class Reference

class which allows to choose the analytic solution.

```
#include <choice_solution.hpp>
```

### Public Member Functions

- [choice\\_solution](#) (parameters &)

*Constructor.*

- void [compute](#) ()

*Performs the solution.*

- virtual [~choice\\_solution](#) ()

*Destructor.*

### 4.3.1 Detailed Description

class which allows to choose the analytic solution.

Definition at line 107 of file choice\_solution.hpp.

### 4.3.2 Constructor & Destructor Documentation

#### 4.3.2.1 choice\_solution::choice\_solution (parameters & par)

Constructor.

Definition at line 45 of file choice\_solution.cpp.

#### 4.3.2.2 choice\_solution::~~choice\_solution () [virtual]

Destructor.

Definition at line 516 of file choice\_solution.cpp.

### 4.3.3 Member Function Documentation

#### 4.3.3.1 void choice\_solution::compute ()

Performs the solution.

Definition at line 512 of file choice\_solution.cpp.

The documentation for this class was generated from the following files:

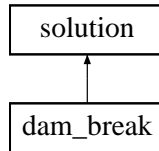
- Headers/[choice\\_solution.hpp](#)
- Sources/[choice\\_solution.cpp](#)

## 4.4 dam\_break Class Reference

class which allows to perform the dam break analytic solution with wet and dry soil

```
#include <dam_break.hpp>
```

Inheritance diagram for dam\_break:



### Public Member Functions

- [dam\\_break](#) (parameters &)

*Constructor.*

- virtual [~dam\\_break](#) ()

*Destructor.*

- void [compute](#) ()

*Virtual method which is specific to each analytic solution.*

- [SCALAR function](#) (SCALAR, SCALAR, SCALAR)

*Function  $x^6 - 9*cr^2*x^4 + 16*cl*cr^2*x^3 - cr^2*(cr^2 + 8*cl^2)*x^2 + cr^6$  to get the roots by dichotomy.*

- void [param](#) (SCALAR, SCALAR, SCALAR, int, SCALAR)

### 4.4.1 Detailed Description

class which allows to perform the dam break analytic solution with wet and dry soil

Definition at line 61 of file dam\_break.hpp.

### 4.4.2 Constructor & Destructor Documentation

#### 4.4.2.1 dam\_break::dam\_break (parameters & par)

Constructor.

Definition at line 48 of file dam\_break.cpp.

#### 4.4.2.2 dam\_break::~dam\_break () [virtual]

Destructor.

Definition at line 90 of file dam\_break.cpp.



### 4.4.3 Member Function Documentation

#### 4.4.3.1 void dam\_break::compute () [virtual]

Virtual method which is specific to each analytic solution.

Implements [solution](#).

Definition at line 95 of file dam\_break.cpp.

#### 4.4.3.2 SCALAR dam\_break::function (SCALAR $x$ , SCALAR $v_{left}$ , SCALAR $v_{right}$ )

Function  $x^6 - 9cr^2x^4 + 16clcr^2x^3 - cr^2(cr^2 + 8cl^2)x^2 + cr^6$  to get the roots by dichotomy.

Definition at line 163 of file dam\_break.cpp.

#### 4.4.3.3 void dam\_break::param (SCALAR $L$ , SCALAR $x_{dam}$ , SCALAR $dx_{ex}$ , int $Nx_{ex}$ , SCALAR $T$ )

Definition at line 171 of file dam\_break.cpp.

The documentation for this class was generated from the following files:

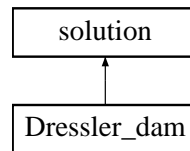
- Headers/[dam\\_break.hpp](#)
- Sources/[dam\\_break.cpp](#)

## 4.5 Dressler\_dam Class Reference

class which allows to perform the Dressler dam break analytic solution

```
#include <Dressler_dam.hpp>
```

Inheritance diagram for Dressler\_dam:



### Public Member Functions

- [Dressler\\_dam](#) (parameters &)

*Constructor.*

- virtual [~Dressler\\_dam](#) ()

*Destructor.*

- void [compute](#) ()

*Virtual method which is specific to each analytic solution.*

- void [param](#) (SCALAR, SCALAR, SCALAR, SCALAR, int, SCALAR)

### 4.5.1 Detailed Description

class which allows to perform the Dressler dam break analytic solution

Definition at line 61 of file `Dressler_dam.hpp`.

### 4.5.2 Constructor & Destructor Documentation

#### 4.5.2.1 Dressler\_dam::Dressler\_dam (parameters & par)

Constructor.

Definition at line 48 of file `Dressler_dam.cpp`.

#### 4.5.2.2 Dressler\_dam::~~Dressler\_dam () [virtual]

Destructor.

Definition at line 78 of file `Dressler_dam.cpp`.

### 4.5.3 Member Function Documentation

#### 4.5.3.1 void Dressler\_dam::compute () [virtual]

Virtual method which is specific to each analytic solution.

Implements [solution](#).

Definition at line 82 of file Dressler\_dam.cpp.

#### 4.5.3.2 void Dressler\_dam::param (SCALAR *L*, SCALAR *x<sub>dam</sub>*, SCALAR *C*, SCALAR *dx<sub>ex</sub>*, int *Nx<sub>ex</sub>*, SCALAR *T*)

Definition at line 176 of file Dressler\_dam.cpp.

The documentation for this class was generated from the following files:

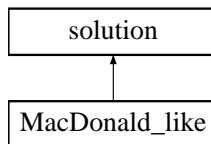
- Headers/[Dressler\\_dam.hpp](#)
- Sources/[Dressler\\_dam.cpp](#)

## 4.6 MacDonald\_like Class Reference

class which allows to perform the MacDonald like analytic solution

```
#include <MacDonald_like.hpp>
```

Inheritance diagram for MacDonald\_like:



### Public Member Functions

- [MacDonald\\_like](#) (parameters &)

*Constructor.*

- virtual [~MacDonald\\_like](#) ()

*Destructor.*

- void [compute](#) ()

*Performs the [MacDonald\\_like](#) solutions.*

- [SCALAR Delta\\_topo\\_Manning](#) (SCALAR q, SCALAR h, SCALAR dh, SCALAR Rain, SCALAR n)

*Evaluation of the slope variation for Manning friction law.*

- [SCALAR Delta\\_topo\\_Darcy\\_Weisbach](#) (SCALAR q, SCALAR h, SCALAR dh, SCALAR Rain, SCALAR n)

*Evaluation of the slope variation for Darcy-Weisbach friction law.*

- void [param](#) (SCALAR, SCALAR, int)

### 4.6.1 Detailed Description

class which allows to perform the MacDonald like analytic solution

Definition at line 61 of file MacDonald\_like.hpp.

### 4.6.2 Constructor & Destructor Documentation

#### 4.6.2.1 MacDonald\_like::MacDonald\_like (parameters & par)

Constructor.

Definition at line 47 of file MacDonald\_like.cpp.

#### 4.6.2.2 MacDonald\_like::~~MacDonald\_like () [virtual]

Destructor.

Definition at line 410 of file MacDonald\_like.cpp.

### 4.6.3 Member Function Documentation

#### 4.6.3.1 void MacDonald\_like::compute () [virtual]

Performs the [MacDonald\\_like](#) solutions.

Implements [solution](#).

Definition at line 415 of file MacDonald\_like.cpp.

#### 4.6.3.2 SCALAR MacDonald\_like::Delta\_topo\_Darcy\_Weisbach (SCALAR $q$ , SCALAR $h$ , SCALAR $dh$ , SCALAR $Rain$ , SCALAR $n$ )

Evaluation of the slope variation for Darcy-Weisbach friction law.

Definition at line 454 of file MacDonald\_like.cpp.

#### 4.6.3.3 SCALAR MacDonald\_like::Delta\_topo\_Manning (SCALAR $q$ , SCALAR $h$ , SCALAR $dh$ , SCALAR $Rain$ , SCALAR $n$ )

Evaluation of the slope variation for Manning friction law.

Definition at line 450 of file MacDonald\_like.cpp.

#### 4.6.3.4 void MacDonald\_like::param (SCALAR $L$ , SCALAR $dx_{ex}$ , int $Nx_{ex}$ )

Definition at line 459 of file MacDonald\_like.cpp.

The documentation for this class was generated from the following files:

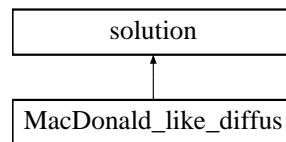
- Headers/[MacDonald\\_like.hpp](#)
- Sources/[MacDonald\\_like.cpp](#)

## 4.7 MacDonald\_like\_diffus Class Reference

class which allows to perform the MacDonald like analytic solution with diffusion

```
#include <MacDonald_like_diffus.hpp>
```

Inheritance diagram for MacDonald\_like\_diffus:



### Public Member Functions

- [MacDonald\\_like\\_diffus](#) (parameters &)  
*Constructor.*
- virtual [~MacDonald\\_like\\_diffus](#) ()  
*Destructor.*
- void [compute](#) ()  
*Performs the *MacDonald\_like* solutions with diffusion.*
- [SCALAR](#) [Delta\\_topo\\_diffus](#) ([SCALAR](#) q, [SCALAR](#) h, [SCALAR](#) dh, [SCALAR](#) ddh, [SCALAR](#) kt, [SCALAR](#) kl, [SCALAR](#) muv, [SCALAR](#) muh)  
*Evaluation of the slope variation.*
- void [param](#) ([SCALAR](#), [SCALAR](#), int)

#### 4.7.1 Detailed Description

class which allows to perform the MacDonald like analytic solution with diffusion

Definition at line 61 of file MacDonald\_like\_diffus.hpp.

#### 4.7.2 Constructor & Destructor Documentation

##### 4.7.2.1 MacDonald\_like\_diffus::MacDonald\_like\_diffus (parameters & par)

Constructor.

Definition at line 47 of file MacDonald\_like\_diffus.cpp.

##### 4.7.2.2 MacDonald\_like\_diffus::~~MacDonald\_like\_diffus () [virtual]

Destructor.

Definition at line 129 of file MacDonald\_like\_diffus.cpp.

### 4.7.3 Member Function Documentation

#### 4.7.3.1 void MacDonald\_like\_diffus::compute () [virtual]

Performs the [MacDonald\\_like](#) solutions with diffusion.

Implements [solution](#).

Definition at line 135 of file MacDonald\_like\_diffus.cpp.

#### 4.7.3.2 SCALAR MacDonald\_like\_diffus::Delta\_topo\_diffus (SCALAR $q$ , SCALAR $h$ , SCALAR $dh$ , SCALAR $ddh$ , SCALAR $kt$ , SCALAR $kl$ , SCALAR $mu\nu$ , SCALAR $mu\eta$ )

Evaluation of the slope variation.

Definition at line 163 of file MacDonald\_like\_diffus.cpp.

#### 4.7.3.3 void MacDonald\_like\_diffus::param (SCALAR $L$ , SCALAR $dx_{ex}$ , int $Nx_{ex}$ )

Definition at line 167 of file MacDonald\_like\_diffus.cpp.

The documentation for this class was generated from the following files:

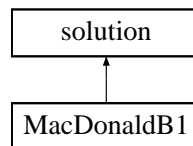
- [Headers/MacDonald\\_like\\_diffus.hpp](#)
- [Sources/MacDonald\\_like\\_diffus.cpp](#)

## 4.8 MacDonaldB1 Class Reference

class which allows to perform the MacDonald PSEUDO 2D analytic solution

```
#include <MacDonaldB1.hpp>
```

Inheritance diagram for MacDonaldB1:



### Public Member Functions

- [MacDonaldB1](#) ([parameters](#) &)  
*Constructor.*
- virtual [~MacDonaldB1](#) ()  
*Destructor.*
- void [compute](#) ()  
*Virtual method which is specific to each analytic solution.*
- void [param](#) ([SCALAR](#), [SCALAR](#), [SCALAR](#), int)
- [SCALAR](#) [Delta\\_topo](#) ([SCALAR](#), [SCALAR](#), [SCALAR](#), [SCALAR](#), [SCALAR](#), [SCALAR](#), [SCALAR](#), [SCALAR](#), [SCALAR](#), [SCALAR](#))

### 4.8.1 Detailed Description

class which allows to perform the MacDonald PSEUDO 2D analytic solution

Definition at line 61 of file MacDonaldB1.hpp.

### 4.8.2 Constructor & Destructor Documentation

#### 4.8.2.1 MacDonaldB1::MacDonaldB1 ([parameters](#) & *par*)

Constructor.

Definition at line 47 of file MacDonaldB1.cpp.

#### 4.8.2.2 MacDonaldB1::~~MacDonaldB1 () [[virtual](#)]

Destructor.

Definition at line 177 of file MacDonaldB1.cpp.



### 4.8.3 Member Function Documentation

#### 4.8.3.1 void MacDonaldB1::compute () [virtual]

Virtual method which is specific to each analytic solution.

Implements [solution](#).

Definition at line 142 of file MacDonaldB1.cpp.

#### 4.8.3.2 SCALAR MacDonaldB1::Delta\_topo (SCALAR *h*, SCALAR *hp*, SCALAR *b*, SCALAR *bp*, SCALAR *Q*, SCALAR *n*, SCALAR *Z*, SCALAR *exp1*, SCALAR *exp2*)

Definition at line 159 of file MacDonaldB1.cpp.

#### 4.8.3.3 void MacDonaldB1::param (SCALAR *L*, SCALAR *dx\_ex*, SCALAR *n*, int *Nx\_ex*)

Definition at line 163 of file MacDonaldB1.cpp.

The documentation for this class was generated from the following files:

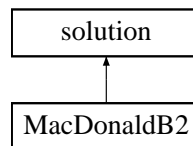
- Headers/[MacDonaldB1.hpp](#)
- Sources/[MacDonaldB1.cpp](#)

## 4.9 MacDonaldB2 Class Reference

class which allows to perform the MacDonald PSEUDO 2D analytic solution

```
#include <MacDonaldB2.hpp>
```

Inheritance diagram for MacDonaldB2:



### Public Member Functions

- [MacDonaldB2](#) ([parameters](#) &)  
*Constructor.*
- virtual [~MacDonaldB2](#) ()  
*Destructor.*
- void [compute](#) ()  
*Virtual method which is specific to each analytic solution.*
- void [param](#) ([SCALAR](#), [SCALAR](#), [SCALAR](#), int)
- [SCALAR](#) [Delta\\_topo](#) ([SCALAR](#), [SCALAR](#), [SCALAR](#), [SCALAR](#), [SCALAR](#), [SCALAR](#), [SCALAR](#), [SCALAR](#), [SCALAR](#), [SCALAR](#))

### 4.9.1 Detailed Description

class which allows to perform the MacDonald PSEUDO 2D analytic solution

Definition at line 61 of file MacDonaldB2.hpp.

### 4.9.2 Constructor & Destructor Documentation

#### 4.9.2.1 MacDonaldB2::MacDonaldB2 ([parameters](#) & *par*)

Constructor.

Definition at line 47 of file MacDonaldB2.cpp.

#### 4.9.2.2 MacDonaldB2::~~MacDonaldB2 () [[virtual](#)]

Destructor.

Definition at line 147 of file MacDonaldB2.cpp.

### 4.9.3 Member Function Documentation

#### 4.9.3.1 void MacDonaldB2::compute () [virtual]

Virtual method which is specific to each analytic solution.

Implements [solution](#).

Definition at line 112 of file MacDonaldB2.cpp.

#### 4.9.3.2 SCALAR MacDonaldB2::Delta\_topo (SCALAR *h*, SCALAR *hp*, SCALAR *b*, SCALAR *bp*, SCALAR *Q*, SCALAR *n*, SCALAR *Z*, SCALAR *exp1*, SCALAR *exp2*)

Definition at line 143 of file MacDonaldB2.cpp.

#### 4.9.3.3 void MacDonaldB2::param (SCALAR *L*, SCALAR *dx\_ex*, SCALAR *n*, int *Nx\_ex*)

Definition at line 129 of file MacDonaldB2.cpp.

The documentation for this class was generated from the following files:

- Headers/[MacDonaldB2.hpp](#)
- Sources/[MacDonaldB2.cpp](#)

## 4.10 parameters Class Reference

class that defines the common parameters.

```
#include <parameters.hpp>
```

### Public Member Functions

- [parameters](#) (int, char \*\*)
- void [setparameters](#) (const char \*)
- virtual [~parameters](#) ()
- void [help](#) ()
- int [get\\_Nxex](#) () const
- int [get\\_Nyex](#) () const
- [SCALAR](#) [get\\_choicedim](#) () const
- int [get\\_choicetype](#) () const
- int [get\\_choice](#) () const
- int [get\\_choicedomain](#) () const

### Protected Attributes

- int [Nx\\_ex](#)
- int [Ny\\_ex](#)
- [SCALAR](#) [choicedim](#)
- int [choicetype](#)
- int [choice](#)
- int [choicedomain](#)

#### 4.10.1 Detailed Description

class that defines the common parameters.

Definition at line 57 of file parameters.hpp.

#### 4.10.2 Constructor & Destructor Documentation

##### 4.10.2.1 parameters::parameters (int *argc*, char \*\* *argv*)

Definition at line 46 of file parameters.cpp.

##### 4.10.2.2 parameters::~parameters () [virtual]

Definition at line 83 of file parameters.cpp.

#### 4.10.3 Member Function Documentation

##### 4.10.3.1 int parameters::get\_choice () const

Definition at line 161 of file parameters.cpp.

#### 4.10.3.2 SCALAR parameters::get\_choicedim () const

Definition at line 165 of file parameters.cpp.

#### 4.10.3.3 int parameters::get\_choicedomain () const

Definition at line 173 of file parameters.cpp.

#### 4.10.3.4 int parameters::get\_choicetype () const

Definition at line 169 of file parameters.cpp.

#### 4.10.3.5 int parameters::get\_Nxex () const

Definition at line 153 of file parameters.cpp.

#### 4.10.3.6 int parameters::get\_Nyex () const

Definition at line 157 of file parameters.cpp.

#### 4.10.3.7 void parameters::help ()

Definition at line 85 of file parameters.cpp.

#### 4.10.3.8 void parameters::setparameters (const char \*)

### 4.10.4 Member Data Documentation

#### 4.10.4.1 int parameters::choice [protected]

Definition at line 62 of file parameters.hpp.

#### 4.10.4.2 SCALAR parameters::choicedim [protected]

Definition at line 60 of file parameters.hpp.

#### 4.10.4.3 int parameters::choicedomain [protected]

Definition at line 63 of file parameters.hpp.

#### 4.10.4.4 int parameters::choicetype [protected]

Definition at line 61 of file parameters.hpp.

#### 4.10.4.5 int parameters::Nx\_ex [protected]

Definition at line 59 of file parameters.hpp.

#### 4.10.4.6 int parameters::Ny\_ex [protected]

Definition at line 59 of file parameters.hpp.

The documentation for this class was generated from the following files:

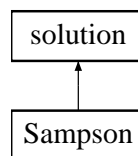
- Headers/[parameters.hpp](#)
- Sources/[parameters.cpp](#)

## 4.11 Sampson Class Reference

class which allows to perform the [Sampson](#) analytic solution

```
#include <Sampson.hpp>
```

Inheritance diagram for Sampson:



### Public Member Functions

- [Sampson](#) (parameters &)

*Constructor.*

- virtual [~Sampson](#) ()

*Destructor.*

- void [compute](#) ()

*Virtual method which is specific to each analytic solution.*

- void [param](#) (SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, int)

#### 4.11.1 Detailed Description

class which allows to perform the [Sampson](#) analytic solution

Definition at line 61 of file Sampson.hpp.

#### 4.11.2 Constructor & Destructor Documentation

##### 4.11.2.1 Sampson::Sampson (parameters & par)

Constructor.

Definition at line 47 of file Sampson.cpp.

##### 4.11.2.2 Sampson::~~Sampson () [virtual]

Destructor.

Definition at line 73 of file Sampson.cpp.

### 4.11.3 Member Function Documentation

#### 4.11.3.1 void Sampson::compute () [virtual]

Virtual method which is specific to each analytic solution.

Implements [solution](#).

Definition at line 78 of file Sampson.cpp.

#### 4.11.3.2 void Sampson::param (SCALAR *L*, SCALAR *h0*, SCALAR *a*, SCALAR *B*, SCALAR *tau*, SCALAR *dx\_ex*, SCALAR *T*, int *Nx\_ex*)

Definition at line 98 of file Sampson.cpp.

The documentation for this class was generated from the following files:

- Headers/[Sampson.hpp](#)
- Sources/[Sampson.cpp](#)

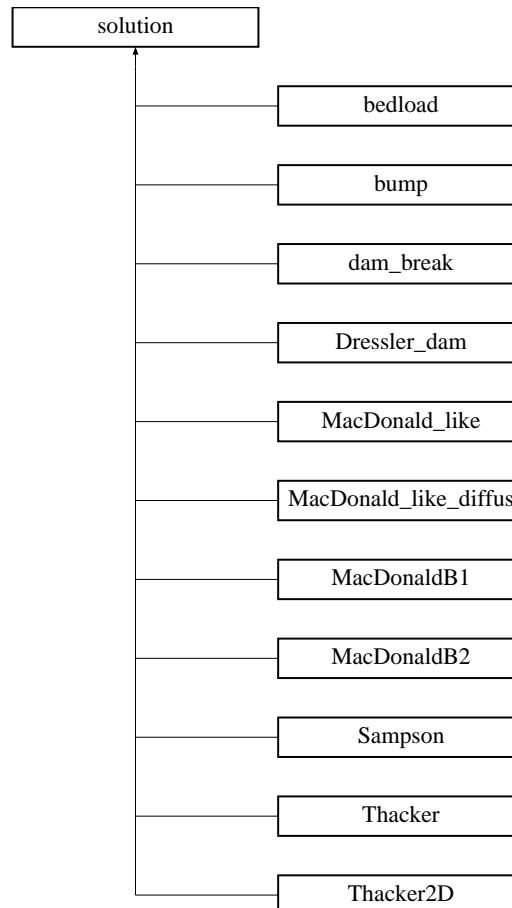


## 4.12 solution Class Reference

class which allows to perform the analytic solutions.

```
#include <solution.hpp>
```

Inheritance diagram for solution:



### Public Member Functions

- `solution (parameters &)`  
*Constructor.*
- `void allocation ()`  
*Tables allocation.*
- `void desallocation ()`  
*Tables desallocation.*
- `virtual void compute ()=0`  
*Virtual method which is specific to each analytic solution.*

- void `savefinalcritical` (`SCALAR *`, `SCALAR *`, `SCALAR *`, `SCALAR *`)  
*Save the analytic solution at the final time with the critical height.*
- void `savefinalcriticalinit` (`SCALAR *`, `SCALAR *`, `SCALAR *`, `SCALAR *`, `SCALAR *`)  
*Save the analytic solution at the final time with the critical height and the initial topography.*
- void `savefinalmu` (`SCALAR *`, `SCALAR *`, `SCALAR *`)  
*Save the analytic solution at the final time without u.*
- void `savefinal2D` (`SCALAR *`, `SCALAR *`, `TAB`, `TAB`, `TAB`, `TAB`)  
*Save the analytic solution at the final time in 2D.*
- void `head` (`parameters &`, `string`, `string`)  
*Write the version of the software and the choice of the solution.*
- virtual `~solution` ()  
*Destructor.*

## Protected Attributes

- int `Nx_ex`
- int `Ny_ex`
- `SCALAR T`
- `SCALAR L`
- `SCALAR l`
- `SCALAR dx_ex`
- `SCALAR dy_ex`
- `SCALAR * xex`
- `SCALAR * yex`
- `SCALAR * hex`
- `SCALAR * uex`
- `SCALAR * qex`
- `SCALAR * zex`

### 4.12.1 Detailed Description

class which allows to perform the analytic solutions.

Definition at line 60 of file `solution.hpp`.

### 4.12.2 Constructor & Destructor Documentation

#### 4.12.2.1 `solution::solution` (`parameters & par`)

Constructor.

Definition at line 47 of file `solution.cpp`.

#### 4.12.2.2 `solution::~~solution () [virtual]`

Destructor.

Definition at line 113 of file `solution.cpp`.

### 4.12.3 Member Function Documentation

#### 4.12.3.1 `void solution::allocation ()`

Tables allocation.

Definition at line 118 of file `solution.cpp`.

#### 4.12.3.2 `virtual void solution::compute () [pure virtual]`

Virtual method which is specific to each analytic solution.

Implemented in [bedload](#), [bump](#), [dam\\_break](#), [Dressler\\_dam](#), [MacDonald\\_like](#), [MacDonald\\_like\\_diffus](#), [MacDonaldB1](#), [MacDonaldB2](#), [Sampson](#), [Thacker](#), and [Thacker2D](#).

#### 4.12.3.3 `void solution::desallocation ()`

Tables desallocation.

Definition at line 151 of file `solution.cpp`.

#### 4.12.3.4 `void solution::head (parameters & par, string solutiontype, string solutionchoice)`

Write the version of the software and the choice of the solution.

Definition at line 101 of file `solution.cpp`.

#### 4.12.3.5 `void solution::savefinal2D (SCALAR * xex, SCALAR * yex, TAB hex2D, TAB uex2D, TAB vex2D, TAB zex2D)`

Save the analytic solution at the final time in 2D.

Definition at line 86 of file `solution.cpp`.

#### 4.12.3.6 `void solution::savefinalcritical (SCALAR * xex, SCALAR * hex, SCALAR * uex, SCALAR * zex)`

Save the analytic solution at the final time with the critical height.

Definition at line 54 of file `solution.cpp`.

#### 4.12.3.7 `void solution::savefinalcriticalinit (SCALAR * xex, SCALAR * hex, SCALAR * uex, SCALAR * zex, SCALAR * z0)`

Save the analytic solution at the final time with the critical height and the initial topography.

Definition at line 67 of file `solution.cpp`.

#### 4.12.3.8 void solution::savefinalmu (SCALAR \* *xex*, SCALAR \* *hex*, SCALAR \* *zex*)

Save the analytic solution at the final time without u.

Definition at line 79 of file solution.cpp.

### 4.12.4 Member Data Documentation

#### 4.12.4.1 SCALAR solution::dx\_ex [protected]

Definition at line 63 of file solution.hpp.

#### 4.12.4.2 SCALAR solution::dy\_ex [protected]

Definition at line 63 of file solution.hpp.

#### 4.12.4.3 SCALAR\* solution::hex [protected]

Definition at line 67 of file solution.hpp.

#### 4.12.4.4 SCALAR solution::l [protected]

Definition at line 63 of file solution.hpp.

#### 4.12.4.5 SCALAR solution::L [protected]

Definition at line 63 of file solution.hpp.

#### 4.12.4.6 int solution::Nx\_ex [protected]

Definition at line 62 of file solution.hpp.

#### 4.12.4.7 int solution::Ny\_ex [protected]

Definition at line 62 of file solution.hpp.

#### 4.12.4.8 SCALAR\* solution::qex [protected]

Definition at line 69 of file solution.hpp.

#### 4.12.4.9 SCALAR solution::T [protected]

Definition at line 63 of file solution.hpp.

#### 4.12.4.10 SCALAR\* solution::uex [protected]

Definition at line 68 of file solution.hpp.

**4.12.4.11 SCALAR\* solution::xex [protected]**

Definition at line 65 of file solution.hpp.

**4.12.4.12 SCALAR\* solution::yex [protected]**

Definition at line 66 of file solution.hpp.

**4.12.4.13 SCALAR\* solution::zex [protected]**

Definition at line 70 of file solution.hpp.

The documentation for this class was generated from the following files:

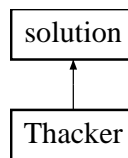
- Headers/[solution.hpp](#)
- Sources/[solution.cpp](#)

## 4.13 Thacker Class Reference

class which allows to perform the [Thacker](#) analytic solution

```
#include <Thacker.hpp>
```

Inheritance diagram for Thacker:



### Public Member Functions

- [Thacker](#) (parameters &)  
*Constructor.*
- virtual [~Thacker](#) ()  
*Destructor.*
- void [compute](#) ()  
*Virtual method which is specific to each analytic solution.*
- void [param](#) (SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, int)

#### 4.13.1 Detailed Description

class which allows to perform the [Thacker](#) analytic solution

Definition at line 61 of file Thacker.hpp.

#### 4.13.2 Constructor & Destructor Documentation

##### 4.13.2.1 Thacker::Thacker (parameters & par)

Constructor.

Definition at line 47 of file Thacker.cpp.

##### 4.13.2.2 Thacker::~~Thacker () [virtual]

Destructor.

Definition at line 72 of file Thacker.cpp.

### 4.13.3 Member Function Documentation

#### 4.13.3.1 `void Thacker::compute () [virtual]`

Virtual method which is specific to each analytic solution.

Implements [solution](#).

Definition at line 77 of file Thacker.cpp.

#### 4.13.3.2 `void Thacker::param (SCALAR L, SCALAR h0, SCALAR a, SCALAR dx_ex, SCALAR T, int Nx_ex)`

Definition at line 98 of file Thacker.cpp.

The documentation for this class was generated from the following files:

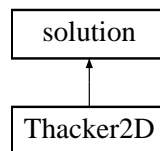
- Headers/[Thacker.hpp](#)
- Sources/[Thacker.cpp](#)

## 4.14 Thacker2D Class Reference

class which allows to perform the [Thacker](#) 2D analytic solution

```
#include <Thacker2D.hpp>
```

Inheritance diagram for Thacker2D:



### Public Member Functions

- [Thacker2D](#) (parameters &)

*Constructor.*

- virtual [~Thacker2D](#) ()

*Destructor.*

- void [compute](#) ()

*Virtual method which is specific to each analytic solution.*

- void [param](#) (SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, SCALAR, int, int)

#### 4.14.1 Detailed Description

class which allows to perform the [Thacker](#) 2D analytic solution

Definition at line 61 of file Thacker2D.hpp.

#### 4.14.2 Constructor & Destructor Documentation

##### 4.14.2.1 Thacker2D::Thacker2D (parameters & par)

Constructor.

Definition at line 47 of file Thacker2D.cpp.

##### 4.14.2.2 Thacker2D::~~Thacker2D () [virtual]

Destructor.

Definition at line 115 of file Thacker2D.cpp.



### 4.14.3 Member Function Documentation

#### 4.14.3.1 void Thacker2D::compute () [virtual]

Virtual method which is specific to each analytic solution.

Implements [solution](#).

Definition at line 131 of file Thacker2D.cpp.

#### 4.14.3.2 void Thacker2D::param (SCALAR *L*, SCALAR *l*, SCALAR *h0*, SCALAR *a*, SCALAR *dx\_ex*, SCALAR *dy\_ex*, SCALAR *T*, int *Nx\_ex*, int *Ny\_ex*)

Definition at line 160 of file Thacker2D.cpp.

The documentation for this class was generated from the following files:

- Headers/[Thacker2D.hpp](#)
- Sources/[Thacker2D.cpp](#)



# Chapter 5

## File Documentation

### 5.1 Headers/bedload.hpp File Reference

Performs bedload (Exner) analytic solutions.

```
#include "solution.hpp"  
#include <vector>  
#include <iomanip>  
#include <iostream>  
#include <cmath>  
#include <stdlib.h>  
#include <fstream>  
#include <complex>  
#include <cstdlib>
```

#### Classes

- class [bedload](#)  
*class which performs bedload (Exner) analytic solution with Grass or MPM equations*

#### Defines

- #define [Class\\_bedload](#)

#### 5.1.1 Detailed Description

Performs bedload (Exner) analytic solutions.

#### Author

Minh Hoang Le and Carine Lucas

Definition in file [bedload.hpp](#).

## 5.1.2 Define Documentation

### 5.1.2.1 #define Class\_bedload

Definition at line 48 of file bedload.hpp.

## 5.2 Headers/bump.hpp File Reference

Performs the bump analytic solutions.

```
#include "solution.hpp"
```

### Classes

- class [bump](#)  
*class which allows to perform the bump analytic solutions*

### 5.2.1 Detailed Description

Performs the bump analytic solutions.

#### Author

Anne-Celine Boulanger, Olivier Delestre

Definition in file [bump.hpp](#).

## 5.3 Headers/choice\_solution.hpp File Reference

Allows to choose the analytic solution.

```
#include "solution.hpp"  
#include "dam_break.hpp"
```

### Classes

- class [choice\\_solution](#)  
*class which allows to choose the analytic solution.*

### Defines

- #define [Class\\_choice\\_solution](#)

#### 5.3.1 Detailed Description

Allows to choose the analytic solution.

#### Author

Olivier Delestre

Definition in file [choice\\_solution.hpp](#).

#### 5.3.2 Define Documentation

##### 5.3.2.1 #define Class\_choice\_solution

Definition at line 94 of file choice\_solution.hpp.

## 5.4 Headers/dam\_break.hpp File Reference

Performs the dam break analytic solution.

```
#include "solution.hpp"
```

### Classes

- class [dam\\_break](#)

*class which allows to perform the dam break analytic solution with wet and dry soil*

### 5.4.1 Detailed Description

Performs the dam break analytic solution.

#### Author

Olivier Delestre

Definition in file [dam\\_break.hpp](#).

## 5.5 Headers/Dressler\_dam.hpp File Reference

Performs classical and modified (personnal communication with Valerio Caleffi, illustration in Valiani et al. 1999) Dressler analytic solution.

```
#include "solution.hpp"
```

### Classes

- class [Dressler\\_dam](#)

*class which allows to perform the Dressler dam break analytic solution*

### 5.5.1 Detailed Description

Performs classical and modified (personnal communication with Valerio Caleffi, illustration in Valiani et al. 1999) Dressler analytic solution.

#### Author

Olivier Delestre

Definition in file [Dressler\\_dam.hpp](#).



## 5.6 Headers/MacDonald\_like.hpp File Reference

Performs the MacDonald like analytic solutions.

```
#include "solution.hpp"
```

### Classes

- class [MacDonald\\_like](#)  
*class which allows to perform the MacDonald like analytic solution*

### 5.6.1 Detailed Description

Performs the MacDonald like analytic solutions.

#### Author

Olivier Delestre

Definition in file [MacDonald\\_like.hpp](#).

## 5.7 Headers/MacDonald\_like\_diffus.hpp File Reference

Performs the MacDonald like analytic solutions with diffusion.

```
#include "solution.hpp"
```

### Classes

- class [MacDonald\\_like\\_diffus](#)  
*class which allows to perform the MacDonald like analytic solution with diffusion*

### 5.7.1 Detailed Description

Performs the MacDonald like analytic solutions with diffusion.

#### Author

Olivier Delestre

Definition in file [MacDonald\\_like\\_diffus.hpp](#).

## 5.8 Headers/MacDonaldB1.hpp File Reference

Performs the MacDonald PSEUDO 2D analytic solutions.

```
#include "solution.hpp"
```

### Classes

- class [MacDonaldB1](#)  
*class which allows to perform the MacDonald PSEUDO 2D analytic solution*

### 5.8.1 Detailed Description

Performs the MacDonald PSEUDO 2D analytic solutions.

#### Author

Pierre-Antoine Ksinant and Carine Lucas

Definition in file [MacDonaldB1.hpp](#).

## 5.9 Headers/MacDonaldB2.hpp File Reference

Performs the MacDonald PSEUDO 2D analytic solutions.

```
#include "solution.hpp"
```

### Classes

- class [MacDonaldB2](#)  
*class which allows to perform the MacDonald PSEUDO 2D analytic solution*

### 5.9.1 Detailed Description

Performs the MacDonald PSEUDO 2D analytic solutions.

#### Author

Pierre-Antoine Ksinant and Carine Lucas

Definition in file [MacDonaldB2.hpp](#).

## 5.10 Headers/misc.hpp File Reference

```
#include <vector>
#include <iomanip>
#include <iostream>
#include <cmath>
#include <stdlib.h>
#include <fstream>
#include <complex>
#include <cstdlib>
```

### Defines

- `#define max(a, b) (a>=b?a:b)`
- `#define min(a, b) (a<=b?a:b)`
- `#define grav 9.81`
- `#define grav_dem 4.905`
- `#define zero 0.`
- `#define PI 3.14159265`
- `#define version " SWASHES version 1.01.00, February 06, 2012"`

### Typedefs

- `typedef double SCALAR`
- `typedef vector< vector< SCALAR > > TAB`

#### 5.10.1 Define Documentation

##### 5.10.1.1 `#define grav 9.81`

Definition at line 55 of file misc.hpp.

##### 5.10.1.2 `#define grav_dem 4.905`

Definition at line 56 of file misc.hpp.

##### 5.10.1.3 `#define max(a, b) (a>=b?a:b)`

Definition at line 52 of file misc.hpp.

##### 5.10.1.4 `#define min(a, b) (a<=b?a:b)`

Definition at line 53 of file misc.hpp.

**5.10.1.5 #define PI 3.14159265**

Definition at line 58 of file misc.hpp.

**5.10.1.6 #define version " SWASHES version 1.01.00, February 06, 2012"**

Definition at line 60 of file misc.hpp.

**5.10.1.7 #define zero 0.**

Definition at line 57 of file misc.hpp.

**5.10.2 Typedef Documentation****5.10.2.1 typedef double SCALAR**

Definition at line 64 of file misc.hpp.

**5.10.2.2 typedef vector< vector< SCALAR > > TAB**

Definition at line 65 of file misc.hpp.

## 5.11 Headers/parameters.hpp File Reference

Defines the common parameters.

### Classes

- class [parameters](#)  
*class that defines the common parameters.*

### 5.11.1 Detailed Description

Defines the common parameters.

#### Author

Olivier Delestre

Definition in file [parameters.hpp](#).

## 5.12 Headers/Sampson.hpp File Reference

Performs the [Sampson](#) analytic solution.

```
#include "solution.hpp"
```

### Classes

- class [Sampson](#)  
*class which allows to perform the [Sampson](#) analytic solution*

### Defines

- #define [Class\\_sampson](#)

#### 5.12.1 Detailed Description

Performs the [Sampson](#) analytic solution.

#### Author

Olivier Delestre

Definition in file [Sampson.hpp](#).

#### 5.12.2 Define Documentation

##### 5.12.2.1 #define Class\_sampson

Definition at line 48 of file Sampson.hpp.



## 5.13 Headers/solution.hpp File Reference

Performs the analytic solutions.

```
#include "misc.hpp"  
#include "parameters.hpp"
```

### Classes

- class [solution](#)  
*class which allows to perform the analytic solutions.*

### 5.13.1 Detailed Description

Performs the analytic solutions.

#### Author

Olivier Delestre

Definition in file [solution.hpp](#).

## 5.14 Headers/Thacker.hpp File Reference

Performs the [Thacker](#) analytic solution.

```
#include "solution.hpp"
```

### Classes

- class [Thacker](#)  
*class which allows to perform the [Thacker](#) analytic solution*

### 5.14.1 Detailed Description

Performs the [Thacker](#) analytic solution.

#### Author

Olivier Delestre

Definition in file [Thacker.hpp](#).

## 5.15 Headers/Thacker2D.hpp File Reference

Performs the [Thacker](#) 2D analytic solutions.

```
#include "solution.hpp"
```

### Classes

- class [Thacker2D](#)  
*class which allows to perform the [Thacker](#) 2D analytic solution*

### 5.15.1 Detailed Description

Performs the [Thacker](#) 2D analytic solutions.

#### Author

Pierre-Antoine Ksinant and Carine Lucas

Definition in file [Thacker2D.hpp](#).

## 5.16 Sources/bedload.cpp File Reference

```
#include "bedload.hpp"
```

## 5.17 Sources/bump.cpp File Reference

```
#include "bump.hpp"
```

## 5.18 Sources/choice\_solution.cpp File Reference

```
#include "choice_solution.hpp"
```

---

## 5.19 Sources/dam\_break.cpp File Reference

```
#include "dam_break.hpp"
```

## 5.20 Sources/Dressler\_dam.cpp File Reference

```
#include "Dressler_dam.hpp"
```



---

## 5.21 Sources/MacDonald\_like.cpp File Reference

```
#include "MacDonald_like.hpp"
```

## 5.22 Sources/MacDonald\_like\_diffus.cpp File Reference

```
#include "MacDonald_like_diffus.hpp"
```

## 5.23 Sources/MacDonaldB1.cpp File Reference

```
#include "MacDonaldB1.hpp"
```

## 5.24 Sources/MacDonaldB2.cpp File Reference

```
#include "MacDonaldB2.hpp"
```

## 5.25 Sources/parameters.cpp File Reference

```
#include "misc.hpp"  
#include "parameters.hpp"
```

## 5.26 Sources/Sampson.cpp File Reference

```
#include "Sampson.hpp"
```

## 5.27 Sources/solution.cpp File Reference

```
#include "solution.hpp"
```

## 5.28 Sources/swashes.cpp File Reference

Main function. Declares the solution and calculates the chosen analytic solution for 1D Shallow Water equations.

```
#include "choice_solution.hpp"  
#include "parameters.hpp"
```

### Functions

- `int main (int argc, char **argv)`

#### 5.28.1 Detailed Description

Main function. Declares the solution and calculates the chosen analytic solution for 1D Shallow Water equations.

#### Author

Olivier Delestre

Definition in file [swashes.cpp](#).

#### 5.28.2 Function Documentation

##### 5.28.2.1 `int main (int argc, char ** argv)`

Definition at line 54 of file [swashes.cpp](#).



## 5.29 Sources/Thacker.cpp File Reference

```
#include "Thacker.hpp"
```

### 5.30 Sources/Thacker2D.cpp File Reference

```
#include "Thacker2D.hpp"
```

# Index

- ~Dressler\_dam
  - Dressler\_dam, 16
- ~MacDonaldB1
  - MacDonaldB1, 22
- ~MacDonaldB2
  - MacDonaldB2, 24
- ~MacDonald\_like
  - MacDonald\_like, 18
- ~MacDonald\_like\_diffus
  - MacDonald\_like\_diffus, 20
- ~Sampson
  - Sampson, 29
- ~Thacker
  - Thacker, 36
- ~Thacker2D
  - Thacker2D, 38
- ~bedload
  - bedload, 8
- ~bump
  - bump, 10
- ~choice\_solution
  - choice\_solution, 13
- ~dam\_break
  - dam\_break, 14
- ~parameters
  - parameters, 26
- ~solution
  - solution, 32
- abcd
  - bump, 10
- allocation
  - solution, 33
- bedload, 7
  - ~bedload, 8
  - bedload, 8
  - compute, 8
  - param, 8
  - paramwarning, 8
- bedload.hpp
  - Class\_bedload, 42
- bump, 9
  - ~bump, 10
  - abcd, 10
  - bump, 10
  - compute, 10
  - determinant, 10
  - function, 11
  - height, 11
  - p, 11
  - param, 11
  - q, 11
  - RHJump, 11
- choice
  - parameters, 27
- choice\_solution, 13
  - ~choice\_solution, 13
  - choice\_solution, 13
  - choice\_solution, 13
  - compute, 13
- choice\_solution.hpp
  - Class\_choice\_solution, 44
- choicedim
  - parameters, 27
- choicedomain
  - parameters, 27
- choicetype
  - parameters, 27
- Class\_bedload
  - bedload.hpp, 42
- Class\_choice\_solution
  - choice\_solution.hpp, 44
- Class\_sampson
  - Sampson.hpp, 54
- compute
  - bedload, 8
  - bump, 10
  - choice\_solution, 13
  - dam\_break, 15
  - Dressler\_dam, 17
  - MacDonald\_like, 19
  - MacDonald\_like\_diffus, 21
  - MacDonaldB1, 23
  - MacDonaldB2, 25
  - Sampson, 30
  - solution, 33
  - Thacker, 37
  - Thacker2D, 39

- dam\_break, 14
  - ~dam\_break, 14
  - compute, 15
  - dam\_break, 14
  - dam\_break, 14
  - function, 15
  - param, 15
- Delta\_topo
  - MacDonaldB1, 23
  - MacDonaldB2, 25
- Delta\_topo\_Darcy\_Weisbach
  - MacDonald\_like, 19
- Delta\_topo\_diffus
  - MacDonald\_like\_diffus, 21
- Delta\_topo\_Manning
  - MacDonald\_like, 19
- desallocation
  - solution, 33
- determinant
  - bump, 10
- Dressler\_dam, 16
  - ~Dressler\_dam, 16
  - compute, 17
  - Dressler\_dam, 16
  - Dressler\_dam, 16
  - param, 17
- dx\_ex
  - solution, 34
- dy\_ex
  - solution, 34
- function
  - bump, 11
  - dam\_break, 15
- get\_choice
  - parameters, 26
- get\_choicedim
  - parameters, 26
- get\_choicedomain
  - parameters, 27
- get\_choicetype
  - parameters, 27
- get\_Nxex
  - parameters, 27
- get\_Nyex
  - parameters, 27
- grav
  - misc.hpp, 51
- grav\_dem
  - misc.hpp, 51
- head
  - solution, 33
- Headers/bedload.hpp, 41
- Headers/bump.hpp, 43
- Headers/choice\_solution.hpp, 44
- Headers/dam\_break.hpp, 45
- Headers/Dressler\_dam.hpp, 46
- Headers/MacDonald\_like.hpp, 47
- Headers/MacDonald\_like\_diffus.hpp, 48
- Headers/MacDonaldB1.hpp, 49
- Headers/MacDonaldB2.hpp, 50
- Headers/misc.hpp, 51
- Headers/parameters.hpp, 53
- Headers/Sampson.hpp, 54
- Headers/solution.hpp, 55
- Headers/Thacker.hpp, 56
- Headers/Thacker2D.hpp, 57
- height
  - bump, 11
- help
  - parameters, 27
- hex
  - solution, 34
- L
  - solution, 34
- l
  - solution, 34
- MacDonald\_like, 18
  - ~MacDonald\_like, 18
  - compute, 19
  - Delta\_topo\_Darcy\_Weisbach, 19
  - Delta\_topo\_Manning, 19
  - MacDonald\_like, 18
  - MacDonald\_like, 18
  - param, 19
- MacDonald\_like\_diffus, 20
  - ~MacDonald\_like\_diffus, 20
  - compute, 21
  - Delta\_topo\_diffus, 21
  - MacDonald\_like\_diffus, 20
  - MacDonald\_like\_diffus, 20
  - param, 21
- MacDonaldB1, 22
  - ~MacDonaldB1, 22
  - compute, 23
  - Delta\_topo, 23
  - MacDonaldB1, 22
  - param, 23
- MacDonaldB2, 24
  - ~MacDonaldB2, 24
  - compute, 25
  - Delta\_topo, 25
  - MacDonaldB2, 24
  - param, 25

- main
  - swashes.cpp, 70
- max
  - misc.hpp, 51
- min
  - misc.hpp, 51
- misc.hpp
  - grav, 51
  - grav\_dem, 51
  - max, 51
  - min, 51
  - PI, 51
  - SCALAR, 52
  - TAB, 52
  - version, 52
  - zero, 52
- Nx\_ex
  - parameters, 27
  - solution, 34
- Ny\_ex
  - parameters, 27
  - solution, 34
- p
  - bump, 11
- param
  - bedload, 8
  - bump, 11
  - dam\_break, 15
  - Dressler\_dam, 17
  - MacDonald\_like, 19
  - MacDonald\_like\_diffus, 21
  - MacDonaldB1, 23
  - MacDonaldB2, 25
  - Sampson, 30
  - Thacker, 37
  - Thacker2D, 39
- parameters, 26
  - ~parameters, 26
  - choice, 27
  - choicedim, 27
  - choicedomain, 27
  - choicetype, 27
  - get\_choice, 26
  - get\_choicedim, 26
  - get\_choicedomain, 27
  - get\_choicetype, 27
  - get\_Nxex, 27
  - get\_Nyex, 27
  - help, 27
  - Nx\_ex, 27
  - Ny\_ex, 27
  - parameters, 26
  - setparameters, 27
- paramwarning
  - bedload, 8
- PI
  - misc.hpp, 51
- q
  - bump, 11
- qex
  - solution, 34
- RHJump
  - bump, 11
- Sampson, 29
  - ~Sampson, 29
  - compute, 30
  - param, 30
  - Sampson, 29
- Sampson.hpp
  - Class\_sampson, 54
- savefinal2D
  - solution, 33
- savefinalcritical
  - solution, 33
- savefinalcriticalinit
  - solution, 33
- savefinalmu
  - solution, 33
- SCALAR
  - misc.hpp, 52
- setparameters
  - parameters, 27
- solution, 31
  - ~solution, 32
  - allocation, 33
  - compute, 33
  - deallocation, 33
  - dx\_ex, 34
  - dy\_ex, 34
  - head, 33
  - hex, 34
  - L, 34
  - l, 34
  - Nx\_ex, 34
  - Ny\_ex, 34
  - qex, 34
  - savefinal2D, 33
  - savefinalcritical, 33
  - savefinalcriticalinit, 33
  - savefinalmu, 33
  - solution, 32
  - T, 34
  - uex, 34

- xex, [34](#)
- yex, [35](#)
- zex, [35](#)
- Sources/bedload.cpp, [58](#)
- Sources/bump.cpp, [59](#)
- Sources/choice\_solution.cpp, [60](#)
- Sources/dam\_break.cpp, [61](#)
- Sources/Dressler\_dam.cpp, [62](#)
- Sources/MacDonald\_like.cpp, [63](#)
- Sources/MacDonald\_like\_diffus.cpp, [64](#)
- Sources/MacDonaldB1.cpp, [65](#)
- Sources/MacDonaldB2.cpp, [66](#)
- Sources/parameters.cpp, [67](#)
- Sources/Sampson.cpp, [68](#)
- Sources/solution.cpp, [69](#)
- Sources/swashes.cpp, [70](#)
- Sources/Thacker.cpp, [71](#)
- Sources/Thacker2D.cpp, [72](#)
- swashes.cpp
  - main, [70](#)
- T
  - solution, [34](#)
- TAB
  - misc.hpp, [52](#)
- Thacker, [36](#)
  - ~Thacker, [36](#)
  - compute, [37](#)
  - param, [37](#)
  - Thacker, [36](#)
- Thacker2D, [38](#)
  - ~Thacker2D, [38](#)
  - compute, [39](#)
  - param, [39](#)
  - Thacker2D, [38](#)
- uex
  - solution, [34](#)
- version
  - misc.hpp, [52](#)
- xex
  - solution, [34](#)
- yex
  - solution, [35](#)
- zero
  - misc.hpp, [52](#)
- zex
  - solution, [35](#)