Package 'hydReng'

January 9, 2025

Type Package

Title Hydraulic Engineering Tools

Version 0.1.0

Description The 'hydReng' package provides a set of functions for hydraulic engineering tasks and natural hazard assessments. It includes basic hydraulics (wetted area, wetted perimeter, flow, flow velocity, flow depth, and maximum flow) for open channels with arbitrary geometry under uniform flow conditions. For structures such as circular pipes, weirs, and gates, the package includes calculations for pressure flow, backwater depth, and overflow over a weir crest. Additionally, it provides formulas for calculating bedload transport. The formulas used can be found in standard literature on hydraulics, such as Bollrich (2019, ISBN:978-3-410-29169-5) or Hager (2011, ISBN:978-3-642-77430-0).

License GPL-3

Encoding UTF-8

RoxygenNote 7.3.2

URL https://github.com/NiccoloGalatioto/hydReng

BugReports https://github.com/NiccoloGalatioto/hydReng/issues

Suggests testthat (>= 3.0.0)

Imports methods

NeedsCompilation no

Author Galatioto Niccolo [cre, aut],
Bühlmann Marius [aut],
HOLINGER AG [cph, fnd]

Maintainer Galatioto Niccolo <niccolo.galatioto@gmail.com>

Repository CRAN

Date/Publication 2025-01-09 17:40:02 UTC

Contents

`bedload_MPM` *Bedload Transport Capacity (Meyer-Peter Müller)*

Description

Calculates the bedload transport capacity using the formula by Meyer-Peter Müller. The formula is valid for bed slopes less than 0.005.

Usage

```
bedload_MPM(dm, J, Rs, B, f_kSt = 0.85, t_crit = 0.047, rho_s = 2650, s = 2.65)
```

Arguments

`dm`	Median grain size [m].
`J`	Bottom slope [-].
`Rs`	Hydraulic radius [m].
`B`	Bottom width [m].
`f_kSt`	Friction factor = (k_StS / k_Str)^(3/2) (default: 0.85).
`t_crit`	Critical shear stress [-] (default: 0.047).
`rho_s`	Density of bedload material [kg/m3] (default: 2650).
`s`	Relative solid density [-] (default: 2.65).

Value

Returns the bedload transport rate [kg/s].

References

Bezzola, G.R. (2012). Vorlesungsmanuskript Flussbau. ETH Zürich, Versuchsanstalt für Wasserbau, Hydrologie und Glaziologie VAW.

Examples

```
bedload_MPM(dm = 0.1, J = 0.01, Rs = 1.5, B = 20)
bedload_MPM(dm = 0.1, J = 0.01, Rs = 1.5, B = 20, t_crit = 0.06)
```

---

bedload_SJ *Bedload Transport Capacity (Smart and Jaeggi)*

Description

Calculates the bedload transport capacity based on the formula by Smart and Jaeggi (1983). This formula is recommended for slopes between 0.005 and 0.2.

Usage

```
bedload_SJ(d30, dm, d90, J, Rs, um, B, t_crit = 0.05, rho_s = 2650,
s_value = 2.65)
```

Arguments

d30	Grain size distribution parameter [m].
dm	Median grain size [m].
d90	Grain size distribution parameter [m].
J	Bottom slope [-].
Rs	Hydraulic radius [m].
um	Mean flow velocity [m/s].
B	Bottom width [m].
t_crit	Critical shear stress [-] (default: 0.05).
rho_s	Density of bedload material [kg/m3] (default: 2650).
s_value	Relative solid density [-] (default: 2.65).

Value

bedload_SJ returns the bedload transport rate [kg/s]

References

Smart, G. M., & Jäggi, M. N. R. (1983). Sediment transport in steilen Gerinnen. Mitteilungen der Versuchsanstalt für Wasserbau, Hydrologie und Glaziologie der ETH Zürich, 64, Zürich.

Examples

```
d30 <- 0.05
dm <- 0.1
d90 <- 0.2
J <- 0.03
Rs <- 1
um <- 2
B <- 3

bedload_SJ(d30 = 0.05, dm = 0.10, d90 = 0.2, J = 0.03, Rs = 1, um = 2, B = 5)
```

`CSarbitrary-class` *CSarbitrary Class*

Description

Defines a cross-section class with arbitrary geometry for hydraulic calculations. For single open channels only, avoid geometries with multiple channels.

Slots

`x` A numeric vector of x-coordinates [m].

`z` A numeric vector of z-coordinates [m].

`xb_l` x-coordinate of the left bank bottom [m].

`xb_r` x-coordinate of the right bank bottom [m].

`kSt_B` Roughness of the channel bed [m^(1/3)/s].

`kSt_l` Roughness of the left bank [m^(1/3)/s].

`kSt_r` Roughness of the right bank [m^(1/3)/s].

Examples

```
# Define sample cross-section data
x <- c(0, 4, 9, 13)
z <- c(2, 0, 0, 2)
cs <- new("CSarbitrary", x = x, z = z, xb_l = 4, xb_r = 9,
          kSt_B = 35, kSt_l = 45, kSt_r = 45)
```

CScircle-class *CScircle Class*

Description

Defines a cross-section class with circular geometry for hydraulic calculations.

Slots

`Di` Diameter of the pipe [m].

`kSt` Roughness of the pipe according to Strickler [m^(1/3)/s].

`ks` Roughness of the pipe according to Prandtl-Coolebrook-White [mm] (SIA 190)

Examples

```
csC <- CScircle(Di = 1, kSt = 75)
csC <- CScircle(Di = 1, ks = 1.5)
```

d_aequiv *Equivalent Hydraulic Diameter*

Description

Calculates the equivalent hydraulic diameter of a rectangular cross-section given its width and height.

Usage

```
d_aequiv(b, h)
```

Arguments

`b`	Width of the rectangle [m].
`h`	Height of the rectangle [m].

Value

The equivalent hydraulic diameter [m].

Examples

```
d_aequiv(b = 2, h = 1)
```

flow *Flow*

Description

Calculates the discharge of a CSarbitrary or CScircle object for a given flow depth and bottom slope under uniform flow conditions.

Usage

```
flow(object, h, J, method = "Strickler", ret = "all", plot = FALSE)
```

Arguments

object	A CSarbitrary or CScircle object.
h	Flow depth [m].
J	Bottom slope [-].
method	Method to calculate the roughness. Allowed are "Strickler" (equal roughness) "Einstein" (mean roughness) and "Prandtl-Coolebrook-White".
ret	Defines the result returned by the function.
plot	Logical; if 'TRUE', plots the results.

Value

A list containing the following hydraulic variables:

Q Discharge [m3/s].

v Flow velocity [m/s].

kSt_m Mean roughness [m^(1/3)/s] (if method = "Einstein").

A Wetted area [m^2].

Examples

```
# Example for CSarbitrary object
x <- c(0, 4, 9, 13)
z <- c(2, 0, 0, 2)
cs <- CSarbitrary(
  x = x, z = z, xb_l = 4, xb_r = 9,
  kSt_B = 35, kSt_l = 45, kSt_r = 45
)
flow(cs, h = 2, J = 0.0001, method = "Einstein", ret = "Q")
flow(cs, h = 2, J = 0.0001, method = "Einstein", plot = TRUE)

# Example for CScircle object
csC <- CScircle(Di = 0.7, ks = 1.5, kSt = 75)
flow(csC, h = 0.46, J = 0.004)
flow(csC, h = 0.46, J = 0.004, method = "Prandtl-Coolebrook-White", plot = TRUE)
```

flow_depth *Flow Depth*

Description

Calculates the flow depth of a CSarbitrary or CScircle object for a given discharge and bottom slope under uniform flow conditions.

Usage

```
flow_depth(object, Q, J, method = "Strickler", ret = "all", plot = FALSE)
```

Arguments

object	A CSarbitrary or CScircle object.
Q	Discharge [m3/s].
J	Bottom slope [-].
method	Method to calculate the roughness. Allowed are "Strickler" (equal roughness) "Einstein" (mean roughness) and "Prandtl-Coolebrook-White".
ret	Defines the result returned by the function.
plot	Logical; if 'TRUE', plots the results.

Value

A list containing the following hydraulic variables:

h Flow depth [m].

v Flow velocity [m/s].

Fr Froude number [-].

kSt_m Mean roughness [m^(1/3)/s] (if method = "Einstein").

A Wetted area [m^2].

P Wetted perimeter [m].

Examples

```
# Example for CSarbitrary object
x <- c(0, 4, 9, 13)
z <- c(2, 0, 0, 2)
cs <- CSarbitrary(
  x = x, z = z, xb_l = 4, xb_r = 9,
  kSt_B = 35, kSt_l = 45, kSt_r = 45
)
flow_depth(cs, Q = 8.677, J = 0.0001, method = "Einstein", ret = "h")
flow_depth(cs, Q = 8.677, J = 0.0001, method = "Einstein", plot = TRUE)
```

```
# Example for CScircle object
csC <- CScircle(Di = 0.7, ks = 1.5, kSt = 75)
flow_depth(csC, Q = 0.46, J = 0.004)
flow_depth(csC, Q = 0.46, J = 0.004, method = "Prandtl-Coolebrook-White", plot = TRUE)
```

`flow_depth_gate` *Water Depth Upstream Of Gate*

Description

Calculates the upstream water depth for a gate based on given discharge and gate parameters.

Usage

```
flow_depth_gate(a, Q, B, alpha, h2 = NULL, ret = "h0")
```

Arguments

`a`	Gate opening height [m].
`Q`	Discharge [m3/s].
`B`	Gate width [m].
`alpha`	Gate angle from horizontal [degrees].
`h2`	Optional. Downstream water depth [m]. Default is NULL (free flow).
`ret`	Specifies the return value. "h0" for depth only or "all" for all intermediate results.

Value

A list containing the following hydraulic variables:

h0 Upstream water depth [m].

psi Contraction coefficient [-].

mu Discharge coefficient [-].

v Flow velocity [m/s].

Examples

```
flow_depth_gate(a = 0.5, Q = 2.5, B = 2.0, alpha = 90)
flow_depth_gate(a = 0.5, Q = 2.5, B = 2.0, alpha = 90, h2 = 0.8)
flow_depth_gate(a = 0.5, Q = 2.5, B = 2.0, alpha = 90, h2 = 0.8, ret = "all")
```

flow_depth_weir *Flow Depth At Weir Crest*

Description

Calculates the height difference between the upstream water level and the weir crest.

Usage

```
flow_depth_weir(B, Q, w = Inf, mu = 0.73)
```

Arguments

B	Width of the weir [m].
Q	Flow rate [m3/s].
w	Height of the weir crest (upstream) [m]. If w = Inf, the upstream velocity is considered 0.
mu	Discharge coefficient [-]. Default is 0.73.

Value

A list with the following components:

h Flow depth over the weir [m].

v Flow velocity [m/s].

Examples

```
flow_depth_weir(B = 3, Q = 5)
flow_depth_weir(B = 3, Q = 5, w = 1)
```

flow_gate *Discharge At Underflow Gate*

Description

Calculates the discharge through a gate under free or submerged conditions.

Usage

```
flow_gate(a, h0, B, alpha, h2 = NULL, ret = "Q")
```

Arguments

a	Gate opening height [m].
h0	Upstream water depth [m].
B	Gate width [m].
alpha	Gate angle from horizontal [degrees].
h2	Optional. Downstream water depth [m]. Default is NULL (free flow).
ret	Specifies the return value. "Q" for discharge only or "all" for all intermediate results.

Value

A list containing the following hydraulic variables:

Q Flow [m3/s].

psi Contraction coefficient [-].

mu Discharge coefficient [-].

v Flow velocity [m/s].

chi Coefficient for submerged flow [-].

Examples

```
flow_gate(a = 0.5, h0 = 1.0, B = 2.0, alpha = 90)
flow_gate(a = 0.5, h0 = 1.0, B = 2.0, alpha = 90, h2 = 0.8)
flow_gate(a = 0.5, h0 = 1.0, B = 2.0, alpha = 90, h2 = 0.8, ret = "all")
```

---

flow_max *Maximum Flow*

Description

Calculates the maximum discharge of a CSarbitrary or CScircle object for a given bottom slope under uniform flow conditions.

Usage

```
flow_max(object, J, method = "Strickler", ret = "all", plot = FALSE)
```

Arguments

object	A CSarbitrary or CScircle object.
J	Bottom slope [-].
method	Method to calculate the roughness. Allowed are "Strickler" (equal roughness) "Einstein" (mean roughness) and "Prandtl-Coolebrook-White".
ret	Defines the result returned by the function.
plot	Logical; if TRUE, plots the results.

Value

A list containing the following hydraulic variables:

Qmax Maximum discharge [m3/s].

hmax Maximum flow depth [m].

v Flow velocity [m/s].

kSt_m Mean roughness [m^(1/3)/s] (if method = "Einstein").

A Wetted area [m2].

Examples

```
# Example for CSarbitrary object
x <- c(0, 4, 9, 13)
z <- c(2, 0, 0, 2)
cs <- CSarbitrary(
  x = x, z = z, xb_l = 4, xb_r = 9,
  kSt_B = 35, kSt_l = 45, kSt_r = 45
)
flow_max(cs, J=0.0001, method="Einstein",ret="Qmax")
flow_max(cs, J=0.0001, method="Einstein",plot=TRUE)

# Example for CScircle object
csC <- CScircle(Di = 0.7, ks = 1.5, kSt = 75)
flow_max(csC, J=0.004)
flow_max(csC, J = 0.004, method = "Prandtl-Coolebrook-White", plot = TRUE)
```

---

flow_max_freeboard *Maximum Flow Including Freeboard*

Description

Calculates the maximum discharge of a CSarbitrary object including a freebord for a given bottom slope under uniform flow conditions.

Usage

```
flow_max_freeboard(object, J, type = "KOHS", sigma_wz = 0, fw = TRUE, fv = FALSE, ft = 0,
fe = NULL, fe_min = 0, fe_max = Inf, method = "Strickler",
ret = "all", plot = FALSE)
```

Arguments

object	A CSarbitrary object.
J	Bottom slope [-].
type	Type of freeboard calculation. Defaults to "KOHS".
sigma_wz	Uncertainty in bed elevation (morphodynamics) [m].

fw	Logical; considers freeboard due to uncertainty in water elevation. If TRUE, calculates according to KOHS; if FALSE, sets fw = 0.
fv	Logical; considers freeboard due to waves. If 'TRUE', calculates according to KOHS; if FALSE, sets fv = 0.
ft	Freeboard due to driftwood based on KOHS (2013) [m].
fe	Fixed freeboard value to override calculations [m].
fe_min	Minimum freeboard [m].
fe_max	Maximum freeboard [m].
method	Method to calculate the roughness. Allowed are "Strickler" (equal roughness) and "Einstein" (mean roughness).
ret	Definition of the result returned by the function ("all", "Qmax", "hmax", "fe", or "v").
plot	Logical; whether to plot the results.

Value

Depending on ret, returns flow, water level, velocity, or all details.

References

KOHS (2013). Freibord bei Hochwasserschutzprojekten und Gefahrenbeurteilungen - Empfehlungen der Kommission Hochwasserschutz KOHS. Wasser Energie Luft 105(1): 43-53.

Examples

```
# Cross section
x <- c(-0.85, 3, 15, 18.85)
z <- c(3.85, 0, 0, 3.85)
cs<- CSarbitrary(x = x, z = z, xb_l = 3, xb_r = 15,
                                     kSt_B = 45)

# Channel
flow_max_freeboard(cs, sigma_wz = 0.3, fv = FALSE, J = 2.2 * 10^-2)
# Dam
flow_max_freeboard(cs, sigma_wz = 0.3, fv = TRUE, J = 2.2 * 10^-2)
# Bridge
flow_max_freeboard(cs, sigma_wz = 0.3, fv = TRUE, ft = 0.5,
           J = 2.2 * 10^-2)

# Sensitivity analysis for slope
J <- seq(1, 3, 0.1) * 10^-2
Q <- sapply(J, function(J) {
  flow_max_freeboard(cs, sigma_wz = 0.3, fv = TRUE, ft = 0.5,
             J = J)$Qmax
})
plot(J, Q, type = "l")
```

flow_velocity *Flow Velocity*

Description

Calculates the flow velocity of a CSarbitrary or CScircle object for a given water level and bottom slope under uniform flow conditions.

Usage

```
flow_velocity(object, h, J, method = "Strickler",nu = 1.14e-6,...)
```

Arguments

object	A CSarbitrary or CScircle object.
h	Flow depth [m].
J	Bottom slope [-].
method	Method to calculate the roughness. Allowed are "Strickler" (equal roughness) "Einstein" (mean roughness) and "Prandtl-Coolebrook-White".
nu	Kinematic viscosity [m2/s]. Only for CScircle objects
...	Additional arguments.

Value

Flow velocity [m/s]

Examples

```
# Example for CSarbitrary object
x <- c(0, 4, 9, 13)
z <- c(2, 0, 0, 2)
cs <- CSarbitrary(x = x, z = z, xb_l = 4, xb_r = 9, kSt_B = 35,
                  kSt_l = 45, kSt_r = 45)
flow_velocity(cs, h = 1,J = 0.01, method = "Einstein")

# Example for CScircle object
csC <- CScircle(Di = 0.7,ks = 1.5, kSt = 75)
flow_velocity(csC, h = 0.46, J = 0.004)
flow_velocity(csC, h = 0.46, J = 0.004, method = "Prandtl-Coolebrook-White")
```

flow_weir *Flow Over Weir Crest*

Description

Calculates the flow over a weir crest based on upstream water level.

Usage

```
flow_weir(B, h, w = Inf, mu = 0.73)
```

Arguments

B	Width of the weir [m].
h	Height difference between the upstream water level and the weir crest [m].
w	Height of the weir crest (upstream) [m]. If w = Inf, the upstream velocity is considered 0.
mu	Discharge coefficient [-]. Default is 0.73.

Value

A list with the following components:

Q Flow over the weir [m3/s].

v Flow velocity [m/s].

Examples

```
flow_weir(B = 3, h = 1.2)
flow_weir(B = 3, h = 1.2, w = 1)
```

freeboard *Freeboard Calculation*

Description

Calculates the required freeboard based on the KOHS (2013) recommendations.

Usage

```
freeboard(v, h, sigma_wz = 0, fw = TRUE, fv = FALSE, ft = 0, min = 0,
  max = Inf, fe_fixed = 0)
```

Arguments

v	Flow velocity [m/s].
h	Flow depth [m].
sigma_wz	Uncertainty in bed elevation (morphodynamics) [m].
fw	Logical; considers freeboard due to uncertainty in water elevation. If 'TRUE', calculates according to KOHS; if 'FALSE', sets 'fw = 0'.
fv	Logical; considers freeboard due to waves. If 'TRUE', calculates according to KOHS; if 'FALSE', sets 'fv = 0'.
ft	Freeboard due to driftwood based on KOHS (2013) [m].
min	Minimum allowable freeboard [m].
max	Maximum allowable freeboard [m].
fe_fixed	Fixed freeboard value to override calculations [m].

Value

A numeric value of the calculated freeboard [m].

References

KOHS (2013). Freibord bei Hochwasserschutzprojekten und Gefahrenbeurteilungen - Empfehlungen der Kommission Hochwasserschutz KOHS. Wasser Energie Luft 105(1): 43-53.

Examples

```
freeboard(h = 1.36, sigma_wz = 0.3, fv = FALSE, ft = 0) # Channel example.
freeboard(v = 4.56, h = 1.36, sigma_wz = 0.3, fv = TRUE, ft = 0) # Dam.
freeboard(v = 4.56, h = 1.36, sigma_wz = 0.3, fv = TRUE, ft = 0.5) # Bridge.
```

---

froude_number *Froude Number*

Description

Calculates the froude number of a CSarbitrary or CScircle object for a given water level and velocity under uniform flow conditions.

Usage

```
froude_number(object, v, h)
```

Arguments

object	A CSarbitrary or CScircle object.
v	Flow velocity [m/s].
h	Flow depth [m].

Value

Froude number [-]

Examples

```
# Example for CSarbitrary object
x <- c(0, 4, 9, 13)
z <- c(2, 0, 0, 2)
cs <- CSarbitrary(x = x, z = z, xb_l = 4, xb_r = 9, kSt_B = 35,
                  kSt_l = 45, kSt_r = 45)
froude_number(cs,h=1, v = 2.5)

# Example for CScircle object
csC <- CScircle(Di = 0.7,ks = 1.5, kSt = 75)
froude_number(csC, h = 0.46, v = 2.5)
```

---

mean_roughness *Mean Roughness*

Description

Calculates the mean roughness of a CSarbitrary object for a given set of water levels, based on Einstein (1934).

Usage

```
mean_roughness(object, h)
```

Arguments

object	A CSarbitrary object.
h	A numeric vector of water levels [m].

Value

A numeric vector representing the mean roughness for the given water levels.

Examples

```
# Example usage:
x <- c(0, 4, 9, 13)
z <- c(2, 0, 0, 2)
cs <- CSarbitrary(x = x, z = z, xb_l = 4, xb_r = 9, kSt_B = 35,
                  kSt_l = 45, kSt_r = 45)
h_levels <- c(1, 2)  # water levels
mean_roughness(cs, h_levels)
```

par_fill *Partial Filling Flow Diagram*

Description

Function to generate a plot of partial-filling diagram of circular pipe with discharge and flow velocity

Usage

```
par_fill(object,J,method="Strickler")
```

Arguments

object	A CScircle object.
J	Bottom slope [-].
method	Method to calculate the roughness. Allowed are "Strickler" (equal roughness) and "Prandtl-Coolebrook-White".

Value

Plots of a partial filling diagram of a circular pipe with discharge and flow velocity

Examples

```
csC <- CScircle(Di = 0.7, ks = 1.5, kSt = 75)
par_fill(csC,J=0.04)
```

pressflow *Flow Under Pressure (Bernoulli)*

Description

Calculates the flow in a pipe or a rectangle under pressure (Bernoulli). The outlet is not submerged, e.g., the exit loss equals 0.

Usage

```
pressflow(z0, z1, h0, Di=NULL, h = NULL, b = NULL, L, ks=NULL, kst,
  xi_e = 0.5, nu = 1.14e-6, calc_lam = "kst")
```

Arguments

z0	Absolute height of upper gate – upstream of the inlet [m.a.s.l].
z1	Absolute height of the pipe/rectangle vertical middle axis at lower gate [m.a.s.l].
h0	Water depth upstream of the gate – upstream of the inlet [m].
Di	Diameter of pipe [m]. If Di is specified, h and b must be NULL.
h	Height of rectangle [m]. If h is specified, Di must be NULL.
b	Width of rectangle [m]. If b is specified, Di must be NULL.
L	Length of pipe [m].
ks	Equivalent sand roughness [m].
kst	Roughness [m^(1/3)/s].
xi_e	Entrance loss [-]. Default = 0.5.
nu	Kinematic viscosity [m2/s]. Default = 1.14e-6.
calc_lam	Defines if lambda should be calculated with ks or kst.

Value

Pressflow returns the flow under pressure:

Q Discharge [m^3/s].

v Flow velocity [m/s].

Examples

```
# Calculate flow in a pipe under pressure with ks value
pressflow(z0 = 415, z1 = 413, h0 = 3, L = 20, Di = 1, ks = 0.01,
  calc_lam = "ks")

# Calculate flow in rectangle under pressure with kst value
pressflow(z0 = 415, z1 = 413, h0 = 3, L = 20, b = 2, h = 1, kst = 60,
  calc_lam = "kst")
```

---

pressflow_depth *Backwater Height Upstream A Inlet Under Pressure (Bernoulli)*

Description

Calculates the backwater height upstream of an inlet (pipe or rectangle) under pressure (Bernoulli). The outlet is not submerged, e.g., the exit loss equals 0.

Usage

```
pressflow_depth(
  z0, z1, Q, Di = NULL, h = NULL, b = NULL, L, ks = NULL, kst,
  xi_e = 0.5, nu = 1.14e-6, calc_lam = "kst"
)
```

Arguments

z0	Absolute height of upper gate – upstream of the inlet [m.a.s.l].
z1	Absolute height of the pipe/rectangle vertical middle axis at lower gate [m.a.s.l].
Q	Flow [m^3/s].
Di	Diameter of pipe [m]. If Di is specified, h and b must be NULL.
h	Height of rectangle [m]. If h is specified, Di must be NULL.
b	Width of rectangle [m]. If b is specified, Di must be NULL.
L	Length of pipe [m].
ks	Equivalent sand roughness [m].
kst	Roughness [m^(1/3)/s].
xi_e	Entrance loss [-]. Default = 0.5.
nu	Kinematic viscosity [m^2/s]. Default = 1.14e-6.
calc_lam	Defines if lambda should be calculated with ks or kst.

Value

Returns the backwater height upstream the inlet:

h0 Water depth upstream the inlet [m].

v Flow velocity [m/s].

Examples

```
# Flow in a pipe under pressure with ks value
pressflow_depth(z0 = 415, z1 = 413, Q = 5.18, L = 20, Di = 1,
                ks = 0.01, calc_lam = "ks")

# Flow in a rectangle under pressure with kst value
pressflow_depth(z0 = 415, z1 = 413, Q = 13.7, L = 20, b = 2, h = 1,
                kst = 60, calc_lam = "kst")
```

---

pressflow_depth_sub *Backwater Height Upstream A Inlet Under Pressure (Bernoulli)*

Description

Calculates the backwater height upstream of an inlet (pipe or rectangle) under pressure (Bernoulli). The outlet is submerged; hence, an exit loss (xi_a) has to be specified.

Usage

```
pressflow_depth_sub(
  z0, z1, Q, h1, v1, Di = NULL, h = NULL, b = NULL, L, ks = NULL, kst, xi_a,
  xi_e = 0.5, nu = 1.14e-6, calc_lam = "kst"
)
```

Arguments

z0	Absolute height of upper gate – upstream of the inlet [m.a.s.l].
z1	Absolute height of the pipe/rectangle vertical middle axis at lower gate [m.a.s.l].
Q	Flow [m^3/s].
h1	Water depth downstream the outlet [m].
v1	Velocity downstream the outlet [m/s].
Di	Diameter of pipe [m]. If Di is specified, h and b must be NULL.
h	Height of rectangle [m]. If h is specified, Di must be NULL.
b	Width of rectangle [m]. If b is specified, Di must be NULL.
L	Length of pipe [m].
ks	Equivalent sand roughness [m].
kst	Roughness [m^(1/3)/s].
xi_a	Exit loss, according to Borda-Carnot formula (1 - A1/A2)^2 [-].
xi_e	Entrance loss [-]. Default = 0.5.
nu	Kinematic viscosity [m^2/s]. Default = 1.14e-6.
calc_lam	Defines if lambda should be calculated with ks or kst.

Value

Returns the backwater height upstream the inlet:

h0 Water depth upstream the inlet [m].

v Flow velocity [m/s].

Examples

```
# Flow in a pipe under pressure with ks value
pressflow_depth_sub(z0=415,z1=413,Q=5.18,h1=2,v1=4,L=20,Di=1,ks=0.01,
calc_lam="ks",xi_a=0.5)

# Flow in a rectangle under pressure with kst value
pressflow_depth_sub(z0=415,z1=413,Q=13.7,h1=2,v1=4,L=20,b=2,h=1,kst=60,
calc_lam="kst",xi_a=0.5)
```

wetted_area *Wetted Area*

Description

Calculates the wetted area of a CSarbitrary or CScircle object for given water levels.

Usage

```
wetted_area(object, h, ret = "A")
```

Arguments

object	An object of class CSarbitrary or CScircle.
h	A numeric vector of water levels (m). For CScircle, only a single numeric value is allowed.
ret	A character string. If 'A', returns total wetted area. If 'Aii', returns wetted area by segment.

Value

A numeric vector or matrix of wetted areas based on the 'ret' argument.

Examples

```
# Example for CSarbitrary object
x <- c(0, 4, 9, 13)
z <- c(2, 0, 0, 2)
cs <- CSarbitrary(x = x, z = z, xb_l = 4, xb_r = 9, kSt_B = 35,
                  kSt_l = 45, kSt_r = 45)

# Calculate total wetted area at water levels 1 m and 2 m
h <- c(1, 2)
wetted_area(cs, h, ret = "A")

# Calculate wetted area for each segment at the same water levels
wetted_area(cs, h, ret = "Aii")

# Example for CScircle object
csC <- CScircle(Di = 1, kSt = 75)

# Calculate total wetted area at water level 1 m
h <- 1
wetted_area(csC, h)
```

wetted_perimeter *Wetted Perimeter*

Description

Calculates the wetted perimeter of a CSarbitrary or CScircle object for given water levels.

Usage

```
wetted_perimeter(object, h, ret = "P")
```

Arguments

object	An object of class CSarbitrary or CScircle.
h	A numeric vector of water levels (m). For CScircle, only a single numeric value is allowed.
ret	A character string. If 'P', returns total wetted perimeter. If 'Pii', returns wetted perimeter by segment.

Value

A numeric vector or matrix of wetted perimeter based on the 'ret' argument.

Examples

```
# Example for CSarbitrary object
x <- c(0, 4, 9, 13)
z <- c(2, 0, 0, 2)
cs <- CSarbitrary(x = x, z = z, xb_l = 4, xb_r = 9, kSt_B = 35,
                  kSt_l = 45, kSt_r = 45)

# Calculate total wetted perimeter at water levels 1 m and 2 m
h <- c(1, 2)
wetted_perimeter(cs, h, ret = "P")

# Calculate wetted perimeter for each segment at the same water levels
wetted_perimeter(cs, h, ret = "Pii")

# Example for CScircle object
csC <- CScircle(Di = 1, kSt = 75)

# Calculate total wetted perimeter at water level 1 m
h <- 1
wetted_perimeter(csC, h)
```

Index