Package 'weirs'

July 22, 2025

Title A Hydraulics Package to Compute Open-Channel Flow over Weirs

Version 0.25

Depends R (>= 2.7.0), utils

Date 2015-08-20

Author William Asquith

Description Provides computational support for flow over weirs, such as sharp-crested, broad-crested, and embankments. Initially, the package supports broad- and sharp-crested weirs.

Maintainer William Asquith <william.asquith@ttu.edu>

License GPL-2

NeedsCompilation no

Repository CRAN

Date/Publication 2015-08-20 19:24:10

Contents

`weir.broadcrest` *Compute Open-Channel Flow over Broad-Crested Weir*

Description

Parent function for broad-crested weir functions of the **weirs** package. This simply named function is set to dispatch to subordinate functions according to the supported procedure types for broad-crested weir flow. The supported procedure types are:

1. TWRI3A5, which represents the implementation in `weir3a5.broadcrest()` the procedures of Hulsing, Harry, 1967, Measurement of peak discharge at dams by indirect methods: U.S. Geological Survey Techniques of Water-Resources Investigations, Book 3, Chapter A5, 29 p., http://pubs.usgs.gov/twri/twri3-a5/
2. BOS, which represents the implementation in `weirbos.broadcrest()` the procedures of Bos, M.G., 1989, Discharge measurement structures: International Institute for Land Reclamation and Improvement Publication 20, Wageningen, The Netherlands, 401 p. http://content.alterra.wur.nl/Internet/webdocs/ilri-publicaties/publicaties/Pub20/pub20.pdf
3. NA is the empty set and will error out.

Usage

```
weir.broadcrest(..., type=c("TWRI3A5", "BOS", NA))
```

Arguments

...	Arguments for subordinate broad-crested weir functions; and
type	Identify the broad-crested weir function to dispatch too.

Value

An R `data.frame()` is returned.

Author(s)

W. Asquith

See Also

`weir3a5.broadcrest`, `weirbos.broadcrest`

`weir.sharpcrest` *Compute Open-Channel Flow over Sharp-Crested Weir*

Description

Parent function for sharp-crested weir functions of the **weirs** package. This simply named function is set to dispatch to subordinate functions according to the supported procedure types for sharp-crested weir flow. The supported procedural types are:

1. TWRI3A5, which represents the implementation in `weir3a5.sharpcrest()` the procedures of Hulsing, Harry, 1967, Measurement of peak discharge at dams by indirect methods: U.S. Geological Survey Techniques of Water-Resources Investigations, Book 3, Chapter A5, 29 p., http://pubs.usgs.gov/twri/twri3-a5/
2. `NA` is the empty set and will error out.

Usage

```
weir.sharpcrest(..., type=c("TWRI3A5", NA))
```

Arguments

`...`	Arguments for subordinate sharp-crested weir functions; and
`type`	Identify the sharp-crested weir function to dispatch too.

Value

An R `data.frame()` is returned.

Author(s)

W. Asquith

See Also

`weir3a5.sharpcrest`

`weir3a5.broadcrest` *Compute Open-Channel Flow over Broad-Crested Weir by TWRI3A5*

Description

Compute open-channel flow (discharge) over a broad-crested weir in accordance with Hulsing (1967) [TWRI3A5]. The weir crest of opening (width) b in feet is P feet above the channel bottom and L feet long in the flow direction. A rectangular approach channel is specified by width B, but the area of the channel (and hence rectangular assumption) can be bypassed by function arguments, although B is used in the contraction ratio b/B unless this ratio is superceded. For the `weir3a5.broadcrest()` function, the computations are exclusively based on the foot-second unit system and careful adherance by the user is required as not all "coefficients" are dimensionless.

The discharge equation for an acceptable tail-water condition h_t is

$$Q = k_c k_R k_s C b H^{1.5}$$

where Q is discharge in cubic feet per second, k_c is the contraction coefficient that also is a function of the abutment rounding r, k_R is the approach rounding coefficient that is a function of the approach crest rounding R, k_s is the downstream embankment slope coefficient, C is the discharge

coefficient, b is the width in feet of the weir crest, and H is total head in feet on the weir, which is computed by

$$H = h + v_o = h + \alpha v^2/2g$$

where h is static head in feet on the weir, v_o is velocity head in feet in the approach section, v is mean velocity in feet per second in the section computed by $v = Q/A$ for cross section area A in square feet, which by default is computed by $A = (h + P)B$, but can be superceded. The quantity g is the acceleration of gravity and is hardwired to 32.2 feet per square second. The dimensionless quantity α permits accommodation of a velocity head correction that is often attributable to cross section subdivision. The α is outside the scope of this documentation, is almost always $\alpha = 1$, and is made available as an argument for advanced users.

The `weir3a5.broadcrest()` function is vectorized meaning that optional vectors of h can be specified along with an optional and equal length vector h_t. The function assumes rectangular approach conditions to compute approach area A if not superceded by the optional `A` argument, which also can be a vector.

The `weir3a5.broadcrest()` function also permits optional vectors of L and b/B (by the argument `contractratio`) so that tuning of the weir-computed discharge to a measured discharge potentially can be made. The crest length L can be used to increase discharge slightly by shortening in say the circumstances of a slightly downward sloping crest. (Such potential "crest" sloping is distinct from the downstream embankment slope—do not confuse the two.) The b/B can be used to decrease discharge by decreasing k_c in say the circumstance of an inlet that is rougher or has asperities that slightly increase the expected contraction and reduce flow efficiency. To clarify, the fact that L and b/B can be vectorized as optional arguments shows a mechanism by which tuning of the computational results to measured Q values can occur without replacing the fundamental nomographs and lookup tables of TWRI3A5 for k_c, k_R, k_s, and C. In all cases, these coefficients can be superceded by user-specified scalars or vectors in various combinations.

Usage

```
weir3a5.broadcrest(h, ht=NULL, b=NULL, B=NULL, P=NULL, L=NULL,
                   R=0, r=0, A=NULL, alpha=1,
                   slopeus="vertical", slopeds="vertical",
                   kc=NULL, kr=NULL, ks=NULL, C=NULL,
                   low.head.class=c("paved", "gravel"),
                   contractratio=NULL,
                   extended=TRUE,
                   header="",
                   flowdigits=2, coedigits=3,
                   verbose=FALSE, eps=0.001, maxit=20)
```

Arguments

`h`	Mandatory scalar or vector of static heads h in feet on the weir;
`ht`	Optional scalar or vector of tail water heads h_t in feet on the weir;
`b`	Mandatory scalar width of weir crest b in feet normal to flow;
`B`	Mandatory scalar width (or top width) of approach channel B in feet. Technically, it is possible with arguments `contractratio` and `A` to bypass any computations normally involving B. This would be the mechanism to bypass the B restriction as a scalar requirement;

P	Mandatory scalar height of weir crest P in feet above channel bed;
L	Optional scalar or vector of lengths L in feet of broad-crested weir in direction of flow;
R	Optional scalar radius of curvature R in feet of vertical upstream face;
r	Optional scalar radius of curvature r in feet on the vertical abutments at inlet of weir crest;
A	Optional scalar or vector of approach cross-section area A in square feet for each h that supersedes the rectangular channel computation $A = (h + P)B$;
alpha	Optional scalar or vector of velocity head correction term α dimensionless. The default is unity ($\alpha = 1$), which is most certainly appropriate for the vast majority of weir computations;
slopeus	String signifying the approach embankment slope in the format "*hz*:*vt*", thus, slope is defined as the ratio of the horizontal *hz* to vertical distance *vt*. (This is opposite of the more common convention for the trigometric function `tan()`.) The string "vertical" must be provided as the value for `slopeus` for vertical slopes;
slopeds	String signifying the downstream embankment slope in the format "*hz*:*vt*", thus, slope is defined as the ratio of the horizontal *hz* to vertical distance *vt*. (This is opposite of the more common convention for the trigometric function `tan()`.) The string "vertical" must be provided as the value for `slopeds` for vertical slopes;
kc	Contraction coefficient k_c, if provided, supercedes nomograph lookup and interpolation by h/P and b/B. Optionally, this coefficient may be a vector;
kr	Rounding coefficient k_R, if provided, supercedes tabular lookup and interpolation by R/h. Optionally, this coefficient may be a vector;
ks	Downstream embankment slope coefficient k_s, if provided supercedes tabular lookup and interpolation by h/L and downstream slope `slopeds`. Optionally, this coefficient may be a vector;
C	Discharge coefficient, if provided, supercedes nomograph lookup and interpolation by h/L and `slopeus`. Optionally, this coefficient may be a vector;
low.head.class	For $h/L < 0.1$, low head on the weir is concluded and alternative C nomograph and interpolation is made based on figure 23 of TWRI3A5. Use of the alternative C requires a "paved" and "gravel" classification in which total head H is used and not h as in the primary C nomographs. How well the paved classification applies to concrete, wood, and metal broad-crested weirs is not discussed in TWRI3A5. Finally, it is expected that most users can use (should use) the paved classification. More formal procedures for embankment flow are provided in TWRI3A5;
contractratio	Optional vector of user specified contraction ratios, if provided, supercedes use of b/B. For example, `b.over.B[i] <- contractratio[i]`;
extended	A logical that controls the contents of the data frame on return;
header	A string (usually) or any other content to add to the `attributes()` of the returned data frame under the non-original label name of `header`;
flowdigits	The number of digits to report on flow, velocity head, and total head;

coedigits	The number of digits to report on weir coefficients;
verbose	A logical controlling intermediate messages. This might be reserved for development work and no verbose output in a released version of **weirs** could occur;
eps	An absolute error of discharge for convergence in cubic feet per second; and
maxit	Maximum number of iterations for the computation of the total head from summation of static and velocity head $H = h + \alpha v^2/2g$ for the final Q_H in item flow of the returned data frame.

Value

An R data.frame() is returned and the extended=TRUE version is described below:

head	Echoed h on the input in feet;
flow	Flow Q_H in cubic feet per second based on total head H;
delta	First order difference of Q_H;
flowo	Flow Q_h in cubic feet per second based on static head h;
error	Absolute convergence error ϵ of Q_H in cubic feet per second;
velhead	Velocity head $v_o = v^2/2g = (Q_H/A)^2/2g$ in feet;
ht	Echoed h_t on the input in feet;
H	Total head $H = h + v_o$;
L	Echoed L in feet;
b.over.B	Echoed b/B;
h.over.L	Echoed h/L;
h.over.P	Echoed h/P;
C	Discharge coefficient C;
kc	Contraction coefficient k_c;
kr	Rounding coefficient k_R;
ks	Downstream slope coefficient k_s;
message	Messages concerning the computation of Q for each value of h; and
source	weir3a5.broadcrest.

The extended=FALSE version is restricted to the most salient items including Q_H, Q_h, v_o, C, k_c, k_R, and k_s.

Note

The weir3a5.broadcrest() function will stop() under conditions of unspecified or implausible L, B, and P as well as incompatibility of b and B, such as $B < b$. This function will also stop() if the length of the vector arguments or optional vector arguments do not match the length of h. The only exception is that if h_t is not specified, then internally it is treated a vector of length h having values of zero. There are other conditions that will cause the function to stop and consultation of the if() statements at the beginning of the function is recommended.

When the weir3a5.broadcrest() function encounters non-stopping errors or warnings, it silently continues with error reporting in the message item in the returned data frame. This behavior is considered a feature and necessary to support the return of the data frame. The message states are:

1. If h_t is too large, then submergence is assumed and NA is returned for all items. The evaluation of submergence is made if $h_t/h \geq 0.85$;
2. If h is zero, then zero is returned for Q_H, Q_h, ϵ, and v_o and NA is returned for others;
3. If a given h tests as too high for broad-crested weir flow and hence the weir is functioning as sharp-crested, then NA is returned for all items; however, for very shallow approach embankment slopes (> 1), then critical $h/L = 2.4$ is used for all h/P and such weirs with $h/L < 2.4$ are treated as broad-crested;
4. If the contraction ratio b/B is too small ($b/B < 0.20$), then too much contraction is concluded and NA is returned for all items;
5. If the upstream embankment slope is too shallow (> 2), then C is indeterminant and NA is returned for all items;
6. If the downstream embankment slope is too shallow (> 5, then k_s is indeterminant and only the values for C, k_c, and k_R are returned;
7. If nonconvergence occurs or estimated Q goes to infinity (supercritical approach or choking), then NA is returned for all Q, ϵ, and v_o, but the estimated C, k_c, k_R, and k_s are returned; and
8. If no problems were detected, then ok is the message.

The influence of abutment rounding by the ratio $r/b > 0$ on k_c is accommodated by prorating between (1) k_c from h/P and b/B or user-specified k_c and (2) $k_c = 1$ unless $r/b > 0.12$ for which $k_c = 1$.

Nomograph and tabular lookup and interpolation is made throughout the computations. The linear interpolating `approx()` function is used for all interpolation. Most commonly, a form of bilinear interpolation is made. First, the two bounding curves for a given condition are interpolated in the horizontal direction and then the resulting two values are interpolated in the vertical. The horizontal interpolation by `approx()` explicitly uses the `rule=2`, which means that extrapolation to the left and right using the respective end point is made. In other words, the nomographs (and tables) are flat lined when extrapolation is needed. Within the code, the horizontal interpolations can be identified by `rule=2` and the vertical interpolations lack the `rule` argument. Finally, the nomographs are in the hashed environment `.weir.nomographs`, which sources from the file '`sysdata.rda`' of the package. The file '`./inst/Nomographs4R/nomographs.R`' is used to create the '`sysdata.rda`' file.

Author(s)

W. Asquith with digitizing of nomograph contributions by W. Miller

References

Hulsing, Harry, 1967, Measurement of peak discharge at dams by indirect methods: U.S. Geological Survey Techniques of Water-Resources Investigations, Book 3, Chapter A5, 29 p., http://pubs.usgs.gov/twri/twri3-a5/

See Also

weir.broadcrest

Examples

```
# Simple, scalar inputs and results
weir.broadcrest(0.5, b=8, B=11, P=6, L=3.25);

# Vector of heads
weir.broadcrest(c(0.5,0.4,0.3), b=8, B=11, P=6, L=3.25);

# Vector of heads and "effective weir lengths"
# Nomograph TWRI3A5-fig7 is being tested here.
weir.broadcrest(  c(0.51, 0.53, 0.70),
                L=c(1,    1.1,  1.20), b=6.18, B=7.84, P=1.13);

# User specified contraction--superceds 8/11
# Nomograph TWRI3A5-fig3 is being tested here.
weir.broadcrest(0.5, b=8, B=11, contractratio=9/11, P=6, L=3.25);
weir.broadcrest(0.5, b=8, B=11, P=6, L=3.25); # compare to previous

# Randomly pick contraction ratios to span the nomograph
# Nomograph TWRI3A5-fig3 is being tested here.
n <- 30; bB <- sort(runif(n));
weir.broadcrest(rep(1,n), b=8, B=11, contractratio=bB, P=6, L=3.25);

# Randomly pick lengths and likely kick-off some sharp-crested messages
# Nomograph TWRI3A5-fig6 is being tested here.
L <- sort(runif(n, min=1, max=30));
weir.broadcrest(rep(10,n), b=8, B=11, P=6, L=L);

# Flow estimate in the non-low.head.classification
# Nomograph TWRI3A5-fig23 is being tested here.
weir.broadcrest(.1, b=6, B=6, P=4, L=1);
# Slighly lower are a realistic step change in h. See how the C
# changes dramatically by  (TWRI3A5-fig23), but at 0.01 feet there
#is some smoothness in the result.
weir.broadcrest(.09, b=6, B=6, P=4, L=1);
# Now try extremely close to 0.10 feet
weir.broadcrest(.0999999, b=6, B=6, P=4, L=1);
# Now try switching from "paved" to "gravel" nomograph (TWRI3A5-fig23)
weir.broadcrest(.09, b=6, B=6, P=4, L=1, low.head.class="gravel");

# Nomograph TWRI3A5-fig7 is being tested here for upstream slope
weir.broadcrest(0.5, slopeus="3:1",     b=8, B=11, P=6, L=3.25);
weir.broadcrest(0.5, slopeus="2:1",     b=8, B=11, P=6, L=3.25);
weir.broadcrest(0.5, slopeus="1.999:1", b=8, B=11, P=6, L=3.25);
weir.broadcrest(0.5, slopeus="0.51:1",  b=8, B=11, P=6, L=3.25);
weir.broadcrest(0.5, slopeus="0.50:1",  b=8, B=11, P=6, L=3.25);
weir.broadcrest(0.5, slopeus="0.49:1",  b=8, B=11, P=6, L=3.25);
try(weir.broadcrest(0.5, slopeus="force-error", b=8, B=11, P=6, L=3.25));

# Nomograph TWRI3A5-p9 (table) is being tested here for
# various downstream slopes
weir.broadcrest(3.5, slopeds="1:1",    b=6, B=11, P=6, L=3.25);
weir.broadcrest(3.5, slopeds="2:1",    b=6, B=11, P=6, L=3.25);
```

```
weir.broadcrest(3.5, slopeds="2.5:1",  b=6, B=11, P=6, L=3.25);
weir.broadcrest(3.5, slopeds="3:1",    b=6, B=11, P=6, L=3.25);
weir.broadcrest(3.5, slopeds="3.5:1",  b=6, B=11, P=6, L=3.25);
weir.broadcrest(3.5, slopeds="4.99:1", b=6, B=11, P=6, L=3.25);
weir.broadcrest(3.5, slopeds="5:1",    b=6, B=11, P=6, L=3.25);
weir.broadcrest(3.5, slopeds="5.1:1",  b=6, B=11, P=6, L=3.25);


# Now configure some weirs for which kc, kr, ks, and C are all
# looked up starting with some (R)ounding and then some more
# (r)ounding and note the changes to kr
weir.broadcrest(3.5, slopeds="3.5:1",  R=0, b=6, B=11, P=6, L=3.25);
# change kr by crest rounding
weir.broadcrest(3.5, slopeds="3.5:1",  R=1, b=6, B=11, P=6, L=3.25);
# change kc by abutment rounding
weir.broadcrest(3.5, slopeds="3.5:1",  R=1, r=.5, b=6, B=11, P=6, L=3.25);
# Now force override of all coefficients
weir.broadcrest(3.5, slopeds="3.5:1",  R=1, r=.5, b=6, B=11, P=6, L=3.25,
                kc=0.95, kr=1.01, ks=.94, C=3.1);


# Now vectorize the coefficients
weir.broadcrest(rep(0.5,3), b=8, B=11, P=6, L=3.25, C=c(3, 3.1, 3.2),
    kc=c(0.95, 1, 1.05), kr=c(1, 1.03, 1.08), ks=c(0.99, 0.95, 0.90));


# Now create a rating curve
h <- seq(0.01,2,by=.01);
Q <- weir.broadcrest(h, b=8, B=11, P=6, L=3.25);
plot(Q$flow, Q$head, type="l", col=8, lwd=7,
     xlab="DISCHARGE, IN CUBIC FEET PER SECOND",
     ylab="HEAD ON WEIR, IN FEET");
lines(sort(Q$flow), Q$head, col=2, lty=2, lwd=3);

# Now take the Q, repeat the sort and then recompute the delta
ii <- order(Q$flow);
Q <- Q[ii,]; # sort the entire data frame
Q$delta <- c(NA, diff(Q$flow)); # recompute deltas
# This insures a monotonic increasing relations between h and Q
# even though it might not be as smooth as hand-guided
# interpolation would provide.
```

`weir3a5.sharpcrest` *Compute Open-Channel Flow over Broad-Crested Weir by TWRI3A5*

Description

Compute open-channel flow (discharge) over a sharp-crested weir in general accordance with Hulsing (1967) [TWRI3A5]. The weir crest of opening (width) b in feet is P feet above the channel bottom and L feet long in the flow direction. A rectangular approach channel is specified by width

B, but the area of the channel (and hence rectangular assumption) can be bypassed by function arguments, although B is used in the contraction ratio b/B unless this ratio is superceded. For the `weir3a5.sharpcrest()` function, the computations are exclusively based on the foot-second unit system and careful adherance by the user is required as not all "coefficients" are dimensionless.

The discharge equation for an acceptable tail-water condition h_t is

$$Q = k_c k_t C b H^{1.5}$$

where Q is discharge in cubic feet per second, k_c is the contraction coefficient that also is a function of the abutment rounding r, k_t is the submergence adjustment coefficient, C is the discharge coefficient, b is the width in feet of the weir crest, and H is total free-flow head in feet on the weir assuming $h_t = 0$, which is computed by

$$H = h + v_o = h + \alpha v^2/2g$$

where h is static head in feet on the weir, v_o is velocity head in feet in the approach section, v is mean velocity in feet per second in the section computed by $v = Q/A$ for cross section area A in square feet, which by default is computed by $A = (h + P)B$, but can be superceded. The quantity g is the acceleration of gravity and is hardwired to 32.2 feet per square second. The dimensionless quantity α permits accommodation of a velocity head correction that is often attributable to cross section subdivision. The α is outside the scope of this documentation, is almost always $\alpha = 1$, and is made available as an argument for advanced users.

The `weir3a5.sharpcrest()` function is vectorized meaning that optional vectors of h can be specified along with an optional and equal length vector h_t. The function assumes rectangular approach conditions to compute approach area A if not superceded by the optional `A` argument, which also can be a vector.

The `weir3a5.sharpcrest()` function also permits optional vectors of L and b/B (by the argument `contractratio`) so that tuning of the weir-computed discharge to a measured discharge potentially can be made. The crest length L can be used to increase discharge slightly by shortening in say the circumstances of a slightly downward sloping crest. The b/B can be used to decrease discharge by decreasing k_c in say the circumstance of an inlet that is rougher or has asperities that slightly increase the expected contraction and reduce flow efficiency. To clarify, the fact that L and b/B can be vectorized as optional arguments shows a mechanism by which tuning of the computational results to measured Q values can occur without replacing the fundamental nomographs and lookup tables of TWRI3A5 for k_c, k_t, and C. In all cases, these coefficients can be superceded by user-specified scalars or vectors in various combinations.

Usage

```
weir3a5.sharpcrest(h, ht=NULL, b=NULL, B=NULL, P=NULL, L=NULL,
                   r=0, A=NULL, alpha=1,
                   slopeus="vertical",
                   kc=NULL, kt=NULL, C=NULL,
                   contractratio=NULL,
                   extended=TRUE,
                   header="", resetkts=TRUE,
                   flowdigits=2, coedigits=3,
                   verbose=FALSE, eps=0.001, maxit=20)
```

Arguments

h	Mandatory scalar or vector of static heads h in feet on the weir;
ht	Optional scalar or vector of tail water heads h_t in feet on the weir;
b	Mandatory scalar width of weir crest b in feet normal to flow;
B	Mandatory scalar width (or top width) of approach channel B in feet. Technically, it is possible with arguments `contractratio` and `A` to bypass any computations normally involving B. This would be the mechanism to bypass the B as a scalar requirement;
P	Mandatory scalar height of weir crest P in feet above channel bed;
L	Optional scalar or vector of lengths L in feet of broad-crested weir in direction of flow;
r	Optional scalar radius of curvature r in feet on the vertical abutments at inlet of weir crest;
A	Optional scalar or vector of approach cross-section area A in square feet for each h that supersedes the rectangular channel computation $A = (h + P)B$;
alpha	Optional scalar or vector of velocity head correction term α dimensionless. The default is unity ($\alpha = 1$), which is most certainly appropriate for the vast majority of weir computations;
slopeus	String signifying the approach embankment slope in the format "*hz*:*vt*", thus, slope is defined as the ratio of the horizontal *hz* to vertical distance *vt*. (This is opposite of the more common convention for the trigometric function `tan()`.) The string "vertical" must be provided as the value for `slopeus` for vertical slopes;
kc	Contraction coefficient k_c, if provided, supercedes nomograph lookup and interpolation by h/P and b/B. Optionally, this coefficient may be a vector;
kt	Coefficient for submergence adjustment, if provided, supercedes nomograph lookup and interpolation by H/P and h_t/P. Optionally, this coefficient may be a vector;
C	Discharge coefficient, if provided, supercedes nomograph lookup and interpolation by h/L and `slopeus`. Optionally, this coefficient may be a vector;
contractratio	Optional vector of user specified contraction ratios, if provided, supercedes use of b/B. For example, `b.over.B[i] <- contractratio[i]`;
extended	A logical that controls the contents of the data frame on return;
header	A string (usually) or any other content to add to the `attributes()` of the returned data frame under the non-original label name of `header`;
resetkts	A logical controlling whether interpolated $k_t > 1$ values are reset to $k_t = 1$ and so diverges slightly from TWRI3A5 (fig.4);
flowdigits	The number of digits to report on flow, velocity head, total head, computed h_t/H, and computed H/P;
coedigits	The number of digits to report on weir coefficients;
verbose	A logical controlling intermediate messages. This might be reserved for development work and no verbose output in a released version of **weirs** could occur;

eps	An absolute error of discharge for convergence in cubic feet per second; and
maxit	Maximum number of iterations for the computation of the total head from summation of static and velocity head $H = h + \alpha v^2/2g$ for the final Q_H in item flow of the returned data frame.

Value

An R data.frame() is returned and the extended=TRUE version is described below:

head	Echoed h on the input in feet;
flow	Flow $k_t Q_H$ in cubic feet per second based on total head H;
delta	First order difference of $k_t Q_H$;
flowfree	Flow Q_H in cubic feet per second using free flow conditions, $h_t = 0$;
flowo	Flow Q_h in cubic feet per second using free flow conditions, $h_t = 0$ based on static head h;
error	Absolute convergence error ϵ of Q_H in cubic feet per second ;
velhead	Velocity head $v_o = v^2/2g = (Q_H/A)^2/2g$ in feet;
Hfree	Total head $H = h + v_o$ in feet;
ht	Echoed h_t on the input in feet;
L	Echoed L in feet;
b.over.B	Echoed b/B;
h.over.L	Echoed h/L;
h.over.P	Echoed h/P;
ht.over.H	Computed h_t/H;
H.over.P	Computed H/P;
C	Discharge coefficient C;
kc	Contraction coefficient k_c;
kt	Coefficient to adjust for submergence k_t. Note that interpolated values $k_t > 1$ are set to $k_t = 1$, if resetkts=TRUE, because weir3a5.sharpcrest() uses iteration to determine H. This practice diverges slightly from TWRI3A5 (fig.4);
message	Messages concerning the computation of Q for each value of h; and
source	weir3a5.sharpcrest.

The extended=FALSE version is restricted to the most salient items including $k_t Q_H$, Q_H, Q_h, v_o, C, k_c, and k_t.

Note

The weir3a5.sharpcrest() function will stop() under conditions of unspecified or implausible L, B, and P as well as incompatibility of b and B, such as $B < b$. This function will also stop() if the length of the vector arguments or optional vector arguments do not match the length of h. The only exception is that if h_t is not specified, then internally it is treated a vector of length h having values of zero. There are other conditions that will cause the function to stop and consultation of the if() statements at the beginning of the function is recommended.

When the `weir3a5.sharpcrest()` function encounters non-stopping errors or warnings, it silently continues with error reporting in the `message` item in the returned data frame. This behavior is considered a feature and necessary to support the return of the data frame. The message states are:

1. If $h/P > 5$, then C has much uncertainty and `NA` is returned for all items;
2. If h is zero, then zero is returned for all Q, ϵ, and v_o and `NA` is returned for others;
3. If a given h tests as too low for sharp-crested weir flow and hence the weir is functioning as broad-crested, then `NA` is returned for all items; however, for very shallow approach embankment slopes (> 1), then critical $h/L = 2.4$ is used for all h/P and such weirs with $h/L < 2.4$ are treated as broad-crested;
4. If the contraction ratio b/B is too small ($b/B < 0.20$), then too much contraction is concluded and `NA` is returned for all items;
5. If the upstream embankment slope is too shallow (> 1), then C is indeterminant and `NA` is returned for all items;
6. If nonconvergence occurs or estimated Q_H goes to infinity (supercritical approach or choking), then `NA` is returned for all Q, ϵ, and v_o, but the estimated C, k_c, and k_t are returned;
7. If $h_t/H > 0.95$ by H from free flow conditions, then too much submergence for k_t computation, and;
8. If no problems were detected, then `ok` is the message.

The conditions important for k_t computation are:

1. If $h_t/H > 0.95$ for total H for free-flow ($h_t = 0$) conditions, then too much submergence is concluded and k_t is `NA` and hence `flow` is `NA`;
2. If $h_t = 0$, then $k_t = 1$ and `flow` is equal to `flowfree`;
3. If $H/P < 0.20$, then k_t can not be computed and is `NA` and hence `flow` is equal to `NA`;
4. If $H/P > 2$, then k_t can not be computed and is `NA` and hence `flow` is equal to `NA`;
5. If $k_t > 1$, then $k_t = 1$ by resetting dependent on the `resetkts` logical argument. This practice differs from TWRI3A5, but prevents submergence from producing more Q than free-flow conditions. The difference is that this function uses iteration to solve for the total head for the free-flow conditions and not a single computation step as seemingly implied in TWRI3A5.

The influence of abutment rounding by the ratio $r/b > 0$ on k_c is accommodated by prorating between (1) k_c from h/P and b/B or user-specified k_c and (2) $k_c = 1$ unless $r/b > 0.12$ for which $k_c = 1$.

Nomograph lookup and interpolation is made throughout the computations. The linear interpolating `approx()` function is used for all interpolation. Most commonly, a form of bilinear interpolation is made. First, the two bounding curves for a given condition are interpolated in the horizontal direction and then the resulting two values are interpolated in the vertical. The horizontal interpolation by `approx()` explicitly uses the `rule=2`, which means that extrapolation to the left and right using the respective end point is made. In other words, the nomographs (and tables) are flat lined when extrapolation is needed. Within the code, the horizontal interpolations can be identified by `rule=2` and the vertical interpolations lack the `rule` argument. Finally, the nomographs are in the hashed environment `.weir.nomographs`, which sources from the file '`sysdata.rda`' of the package. The file '`./inst/Nomographs4R/nomographs.R`' is used to create the '`sysdata.rda`' file.

Author(s)

W. Asquith with digitizing of nomograph contributions by W. Miller

References

Hulsing, Harry, 1967, Measurement of peak discharge at dams by indirect methods: U.S. Geological Survey Techniques of Water-Resources Investigations, Book 3, Chapter A5, 29 p., http://pubs.usgs.gov/twri/twri3-a5/

See Also

weir.sharpcrest

Examples

```
weir3a5.sharpcrest(0.45, L=0.125, P=0.32, b=5.81, B=5.81)


h <- seq(0.15,0.64,by=.01)
Qo <- weir3a5.sharpcrest(h, L=0.125, P=0.32, b=5.81, B=5.81)
print(Qo)

ht <- seq(0.15,0.64,by=.01)/2
weir3a5.sharpcrest(h, ht=ht, L=0.125, P=0.32, b=5.81, B=5.81)

plot(Qo$flow, Qo$head, type="l", log="xy")
Q <- weir3a5.sharpcrest(h, ht=0.21*h, L=0.125, P=0.32, b=5.81, B=5.81)
lines(Q$flow, Q$head, lty=2)
Q <- weir3a5.sharpcrest(h, ht=0.4*h, L=0.125, P=0.32, b=5.81, B=5.81)
lines(Q$flow, Q$head, lty=2)
Q <- weir3a5.sharpcrest(h, ht=0.6*h, L=0.125, P=0.32, b=5.81, B=5.81)
lines(Q$flow, Q$head, lty=2)
Q <- weir3a5.sharpcrest(h, ht=0.8*h, L=0.125, P=0.32, b=5.81, B=5.81)
lines(Q$flow, Q$head, lty=2)
```

---

weirbos.broadcrest *Compute Open-Channel Flow over Broad-Crested Weir by Bos (1989)*

Description

Compute open-channel flow (discharge) over a broad-crested weir in accordance with Bos (1989) [BOS] with extension into the coefficients k_c, k_R, and k_s from Hulsing (1967). The weir crest of opening (width) b in feet is P feet above the channel bottom and L feet long in the flow direction. A rectangular approach channel is specified by width B, but the area of the channel (and hence rectangular assumption) can be bypassed by function arguments, although B is used in the contraction ratio b/B unless this ratio is superceded. For the weirbos.broadcrest() function, the

computations are exclusively based on the foot-second unit system and careful adherance by the user is required.

The discharge equation for an acceptable tail-water condition h_t is

$$Q = k_c k_R k_s C_v C b \frac{2}{3}\sqrt{\frac{2}{3}g}H^{1.5}$$

where Q is discharge in cubic feet per second, k_c is the contraction coefficient that also is a function of the abutment rounding r, k_R is the approach rounding coefficient that is a function of the approach crest rounding R, k_s is the downstream embankment slope coefficient, C_v is the approach velocity coefficient, C is the discharge coefficient, b is the width in feet of the weir crest, and H is total head in feet on the weir, which is computed by

$$H = h + v_o = h + \alpha v^2/2g$$

where h is static head in feet on the weir, v_o is velocity head in feet in the approach section, v is mean velocity in feet per second in the section computed by $v = Q/A$ for cross section area A in square feet, which by default is computed by $A = (h + P)B$, but can be superceded. The quantity g is the acceleration of gravity and is hardwired to 32.2 feet per square second. The dimensionless quantity α permits accommodation of a velocity head correction that is often attributable to cross section subdivision. The α is outside the scope of this documentation, is almost always $\alpha = 1$, and is made available as an argument for advanced users.

The discharge equation is solved for two conditions that consider the approach velocity head. First, the equation is solved as shown above for H through iteration and $C_v = 1$. Second, the equation is solved using h and C_v, which is determined by nomograph lookup.

The `weirbos.broadcrest()` function is vectorized meaning that optional vectors of h can be specified along with an optional and equal length vector h_t. The function assumes rectangular approach conditions to compute approach area A if not superceded by the optional `A` argument, which also can be a vector.

The `weirbos.broadcrest()` function also permits optional vectors of L and b/B (by the argument `contractratio`) so that tuning of the weir-computed discharge to a measured discharge potentially can be made. The crest length L can be used to increase discharge slightly by shortening in say the circumstances of a slightly downward sloping crest. (Such potential "crest" sloping is distinct from the downstream embankment slope—do not confuse the two.) The b/B can be used to decrease discharge by decreasing k_R in say the circumstance of an inlet that is rougher or has asperities that slightly increase the expected contraction and reduce flow efficiency. To clarify, the fact that L and b/B can be vectorized as optional arguments shows a mechanism by which tuning of the computational results to measured Q values can occur without replacing the fundamental C and C_v nomographs of BOS and nomograph and lookup tables for k_c, k_R, and k_s of TWRI3A5 (Hulsing, 1967). In all cases, these coefficients can be superceded by user-specified scalars or vectors in various combinations.

Usage

```
weirbos.broadcrest(h, ht=NULL, b=NULL, B=NULL, P=NULL, L=NULL,
                   R=0, r=0, A=NULL, alpha=1,
                   slopeus="vertical", slopeds="vertical",
                   kc=NULL, kr=NULL, ks=NULL, C=NULL,
```

```
    contractratio=NULL,
    hhptest=TRUE, extended=TRUE,
    header="",
    flowdigits=2, coedigits=3,
    verbose=FALSE, eps=0.001, maxit=20)
```

Arguments

h	Mandatory scalar or vector of static heads h in feet on the weir;
ht	Optional scalar or vector of tail water heads h_t in feet on the weir;
b	Mandatory scalar width of weir crest b in feet normal to flow;
B	Mandatory scalar width (or top width) of approach channel B in feet. Technically, it is possible with arguments contractratio and A to bypass any computations normally involving B. This would be the mechanism to bypass the B as a scalar restriction;
P	Mandatory scalar height of weir crest P in feet above channel bed;
L	Optional scalar or vector of lengths L in feet of broad-crested weir in direction of flow;
R	Optional scalar radius of curvature R in feet of vertical upstream face;
r	Optional scalar radius of curvature r in feet on the vertical abutments at inlet of weir crest;
A	Optional scalar or vector of approach cross-section area A in square feet for each h that supersedes the rectangular channel computation $A = (h + P)B$;
alpha	Optional scalar or vector of velocity head correction term α dimensionless. The default is unity ($\alpha = 1$), which is most certainly appropriate for the vast majority of weir computations;
slopeus	String signifying the approach embankment slope in the format "*hz*:*vt*", thus, slope is defined as the ratio of the horizontal *hz* to vertical distance *vt*. (This is opposite of the more common convention for the trigometric function tan().) The string "vertical" must be provided as the value for slopeus for vertical slopes;
slopeds	String signifying the downstream embankment slope in the format "*hz*:*vt*", thus, slope is defined as the ratio of the horizontal *hz* to vertical distance *vt*. (This is opposite of the more common convention for the trigometric function tan().) The string "vertical" must be provided as the value for slopeds for vertical slopes;
kc	Contraction coefficient k_c, if provided, supercedes nomograph lookup and interpolation by h/P and b/B. Optionally, this coefficient may be a vector;
kr	Rounding coefficient k_R, if provided, supercedes tabular lookup and interpolation by R/h. Optionally, this coefficient may be a vector;
ks	Downstream embankment slope coefficient k_s, if provided supercedes tabular lookup and interpolation by h/L and downstream slope slopeds. Optionally, this coefficient may be a vector;
C	Discharge coefficient, if provided, supercedes nomograph lookup and interpolation by h/L and slopeus. Optionally, this coefficient may be a vector;

`hhptest`	Logical for test of $h/(h+P) \leq 0.35$ to follow Bos (1989) to determine C, but the test can be ignored with this logical argument;
`contractratio`	Optional vector of user specified contraction ratios, if provided, supercedes use of b/B. For example, `b.over.B[i] <- contractratio[i]`;
`extended`	A logical that controls the contents of the data frame on return;
`header`	A string (usually) or any other content to add to the `attributes()` of the returned data frame under the non-original label name of `header`;
`flowdigits`	The number of digits to report on flow, velocity head, and total head;
`coedigits`	The number of digits to report on weir coefficients;
`verbose`	A logical controlling intermediate messages. This might be reserved for development work and no verbose output in a released version of **weirs** could occur;
`eps`	An absolute error of discharge for convergence in cubic feet per second; and
`maxit`	Maximum number of iterations for the computation of the total head from summation of static and velocity head $H = h + \alpha v^2/2g$ for the final Q_H in item `flow` of the returned data frame.

Value

An R `data.frame()` is returned and the `extended=TRUE` version is described below:

`head`	Echoed h on the input in feet;
`flow`	Flow Q_H in cubic feet per second based on total head H;
`delta`	First order difference of Q_H;
`flowo`	Flow Q_h in cubic feet per second based on static head h;
`flowcv`	Flow Q_{cv} in cubic feet per second based on $C_v \neq 1$ nomograph lookup by h/L and static head h;
`error`	Absolute convergence error ϵ of Q_H in cubic feet per second;
`velhead`	Velocity head $v_o = v^2/2g = (Q_H/A)^2/2g$ in feet;
`H`	Total head $H = h + v_o$;
`ht`	Echoed h_t on the input in feet;
`L`	Echoed L in feet;
`b.over.B`	Echoed b/B;
`h.over.L`	Echoed h/L;
`h.over.P`	Echoed h/P;
`C`	Discharge coefficient C;
`Cv`	Approach velocity coefficient;
`kc`	Contraction coefficient k_c;
`kr`	Rounding coefficient k_R;
`ks`	Downstream slope coefficient k_s;
`message`	Messages concerning the computation of Q for each value of h; and
`source`	`weirbos.broadcrest`.

The `extended=FALSE` version is restricted to the most salient items including Q_H, Q_h, Q_{cv}, v_o, C, C_v, k_c, k_R, and k_s.

Note

The `weirbos.broadcrest()` function will `stop()` under conditions of unspecified or implausible L, B, and P as well as incompatibility of b and B, such as $B < b$. This function will also `stop()` if the length of the vector arguments or optional vector arguments do not match the length of h. The only exception is that if h_t is not specified, then internally it is treated a vector of length h having values of zero. There are other conditions that will cause the function to stop and consultation of the `if()` statements at the beginning of the function is recommended.

When the `weirbos.broadcrest()` function encounters non-stopping errors or warnings, it silently continues with error reporting in the `message` item in the returned data frame. This behavior is considered a feature and necessary to support the return of the data frame. The message states are:

1. If h_t is too large, then submergence is assumed and `NA` is returned for all items. The evaluation of submergence is made if $h_t/h \geq 0.66$;
2. If $h/(h + P) \leq 0.35$ then C can be determined and computation progress, otherwise `NA` is returned for all items;
3. If h is zero, then zero is returned for Q_H, Q_h, Q_{cv}, ϵ, and v_o and `NA` is returned for others;
4. If a given h/L tests as too high ($h/L < 1.5$) for broad-crested weir flow and hence the weir is functioning as sharp-crested, then `NA` is returned for all items. Weirs with $h/L < 0.08$ are treated as low head, then `NA` is returned for all items;
5. If the contraction ratio b/B is too small ($b/B < 0.20$), then too much contraction is concluded and `NA` is returned for all items;
6. If the downstream embankment slope is too shallow (> 5, then k_s is indeterminant and only the values for C, k_c, and k_R are returned;
7. If nonconvergence occurs or estimated Q goes to infinity (supercritical approach or choking), then `NA` is returned for all Q, ϵ, and v_o, but the estimated C, k_c, k_R, and k_s are returned, and;
8. If no problems were detected, then `ok` is the message.

The influence of abutment rounding by the ratio $r/b > 0$ on k_c is accommodated by prorating between (1) k_c from h/P and b/B or user-specified k_c and (2) $k_c = 1$ unless $r/b > 0.10$ for which $k_c = 1$.

Nomograph and tabular lookup and interpolation is made throughout the computations. The linear interpolating `approx()` function is used for all interpolation. Most commonly, a form of bilinear interpolation is made. First, the two bounding curves for a given condition are interpolated in the horizontal direction and then the resulting two values are interpolated in the vertical. The horizontal interpolation by `approx()` explicitly uses the `rule=2`, which means that extrapolation to the left and right using the respective end point is made. In other words, the nomographs (and tables) are flat lined when extrapolation is needed. Within the code, the horizontal interpolations can be identified by `rule=2` and the vertical interpolations lack the `rule` argument. Finally, the nomographs are in the hashed environment `.weir.nomographs`, which sources from the file '`sysdata.rda`' of the package. The file '`./inst/Nomographs4R/nomographs.R`' is used to create the '`sysdata.rda`' file.

Author(s)

W. Asquith with digitizing of nomograph contributions by W. Miller

References

Bos, M.G., 1989, Discharge measurement structures: International Institute for Land Reclamation and Improvement Publication 20, Wageningen, The Netherlands, 401 p. http://content.alterra.wur.nl/Internet/webdocs/ilri-publicaties/publicaties/Pub20/pub20.pdf

Hulsing, Harry, 1967, Measurement of peak discharge at dams by indirect methods: U.S. Geological Survey Techniques of Water-Resources Investigations, Book 3, Chapter A5, 29 p., http://pubs.usgs.gov/twri/twri3-a5/

See Also

weir.broadcrest

Examples

```
# Simple, scalar inputs and results
weir.broadcrest(type="BOS", 0.5, b=8, B=11, P=6, L=3.25);

# Vector of heads
weir.broadcrest(type="BOS", c(0.5,0.4,0.3), b=8, B=11, P=6, L=3.25);

# Now compare two solutions by weirbos.broadcrest() to weir3a5.broadcrest()
h <- seq(0.01,1.5, by=0.001)
Qtank     <- weir3a5.broadcrest(h, b=7.97, B=10.97, P=1.125, L=3.76, flowdigits=4)
QtankbosA <- weirbos.broadcrest(h, b=7.97, B=10.97, P=1.125, L=3.76, flowdigits=4)
QtankbosB <- weirbos.broadcrest(h, b=7.97, B=10.97, P=1.125, L=3.76,
                                   hhptest=FALSE, flowdigits=4)
plot(Qtank$flow, h, type="l", col=8, lwd=6,
     xlab="FLOW, CFS", ylab="HEAD, FEET")
lines(QtankbosA$flow, h, col=2, lwd=4) # see the trunction in the red line
lines(QtankbosB$flow, h, col=3, lwd=2)

# See examples for weir3a5.broadcrest() for additional examples that
# explore higher complexity of argument settings, which test the
# nomographs for k_c, k_R, and k_s
```

Index