Package ‘kitagawa’

January 29, 2024

Type     Package
Title    Spectral Response of Water Wells to Harmonic Strain and Pressure Signals
Version  3.1.2
Date     2024-01-26

Description Provides tools to calculate the theoretical hydrodynamic response of an aquifer undergoing harmonic straining or pressurization, or analyze measured responses. There are two classes of models here, designed for use with confined aquifers: (1) for sealed wells, based on the model of Kitagawa et al (2011, <doi:10.1029/2010JB007794>), and (2) for open wells, based on the models of Cooper et al (1965, <doi:10.1029/JZ070i016p03915>), Hsieh et al (1987, <doi:10.1029/WR023i010p01824>), Rojstaczer (1988, <doi:10.1029/JB093iB11p13619>), Liu et al (1989, <doi:10.1029/JB094iB07p09453>), and Wang et al (2018, <doi:10.1029/2018WR022793>). Wang’s solution is a special exception which allows for leakage out of the aquifer (semi-confined); it is equivalent to Hsieh’s model when there is no leakage (the confined case). These models treat strain (or aquifer head) as an input to the physical system, and fluid-pressure (or water height) as the output. The applicable frequency band of these models is characteristic of seismic waves, atmospheric pressure fluctuations, and solid earth tides.

License  GPL (>= 2)

URL     https://github.com/abarbour/kitagawa

BugReports https://github.com/abarbour/kitagawa/issues

Depends  R (>= 2.10.1), stats
Imports   Bessel, kelvin (>= 1.2.0), psd (>= 2.0.0)
Suggests dplyr, tibble, RColorBrewer, signal, testthat, knitr, rmarkdown, formatR, covr
alpha_constants

Calculate any constants depending on effective stress coefficient $\alpha$

Description

This function accesses the appropriate method to calculate the $\alpha$-dependent constant associated with the choice of c.type. There are currently four such constants, which correspond to Equations 10, 11, 18, 19 in Kitagawa et al (2011).

This function is not likely to be needed by the user.

Usage

```r
alpha_constants(alpha = 0, c.type = c("Phi", "Psi", "A", "Kel"))
```

## Default S3 method:
```r
alpha_constants(alpha = 0, c.type = c("Phi", "Psi", "A", "Kel"))
```
Arguments

alpha     the constant alpha (see `omega_constants`)
c.type    the constant to calculate

Details

What is "alpha"?: The constant $\alpha$ is a function of frequency $\omega$ as well as aquifer and well parameters; it is formally defined as

$$\alpha \equiv R_S \sqrt{\omega S/T}$$

where $S$ is the storativity, $T$ is the aquifer’s effective transmissivity, and $R_S$ is the radius of the screened portion of the well.

What is calculated?: The various constants which may be calculated with this function are

- Phi  Given as $\Phi$ in Eqn. 10
- Psi  Given as $\Psi$ in Eqn. 11
- A   Given as $A_i, i = 1, 2$ in Eqns. 18, 19
- Kel The complex Kelvin functions (see Abramowitz and Stegun, 1972)

Value

Complex matrix having values representing the constant represented by c.type, as well as any other $\alpha$-dependent constants which are needed in the computation.

Author(s)

A. J. Barbour <andy.barbour@gmail.com>

See Also

`omega_constants, well_response`

Other ConstantsCalculators: `kitagawa-constants, omega_constants()`

Examples

```r
alpha_constants()  # kelvin::Keir gives warning
alpha_constants(1)  # defaults to constant 'Phi' (note output also has Kel)
alpha_constants(1:10, c.type="A")  # constant 'A' (again, note output)
```
cross_spectrum

Calculate the cross-spectrum of two timeseries

Description

Calculate the cross-spectrum of two timeseries

Usage

cross_spectrum(x, ...)

## S3 method for class 'mts'
cross_spectrum(x, ...)

## Default S3 method:
cross_spectrum(
  x,
  y,
  k = 10,
  samp = 1,
  q,
  adaptive = FALSE,
  verbose = FALSE,
  ...
)

Arguments

x numeric; timeseries

... additional arguments to \texttt{pspectrum}

y numeric; timeseries. if missing, assumed to be column no. 2 in \texttt{x}

k integer; the number of sine tapers, unless this is \texttt{NULL}; in the latter case a Welch-based spectrum is calculated rather than a multitaper spectrum. There are distinct advantages and disadvantages to either of these.

samp numeric; the sampling rate (e.g., \texttt{deltat}) of the data; must be the same for \texttt{x} and \texttt{y}

q numeric; the probability quantile [0,1] to calculate coherence significance levels; if missing, a pre-specified sequence is included. This is will be ignored for Welch-based spectra (see \texttt{k}).

adaptive logical; should adaptive multitaper estimation be used?

verbose logical; should messages be printed?
Examples

```r
require(stats)
require(psd)

n <- 1000
ramp <- seq_len(n)
parab <- ramp^2

set.seed(1255)
X <- ts(rnorm(n) + ramp/2)
Y <- ts(rnorm(n) + ramp/10 + parab/100)

# Calculate the multitaper cross spectrum
csd <- cross_spectrum(X, Y, k=20)
```

Description

The response of an aquifer depends on its mechanical and hydrological properties; if these are not known or specified, these constants are used.

Usage

```r
constants(do.str = TRUE)
```

Arguments

do.str logical; should the structure be printed?

Details

The function `constants` shows the structure of (optionally), and returns the assumed constants, which do not reside in the namespace.

Values:

- **For water:** Density and bulk modulus
- **Gravity:** Standard gravitational acceleration at 6371km radius (Earth)
- **Conversions:** Degrees to radians

Value

The constants, invisibly.
See Also

well_response and open_well_response

kitagawa-package

Other ConstantsCalculators: alpha_constants(), omega_constants()

Examples

counts() counts(FALSE) # returns invisibly

kitagawa-utilities General utility functions

Description

General utility functions

Usage

.nullchk(X)

.in0to1(X)

is.wrsp(X)

is.owrsp(X)

Arguments

X something to be checked (vector, scalar, wrsp object, ...)

Details

.nullchk quickly checks for NULL and NA, and raises an error if TRUE; This function is not likely to be needed by the user.

.in0to1 checks if values are numeric and in [0,1] (inclusive).

is.wrsp and is.owrsp report whether an object has S3 class 'wrsp' or 'owrsp', respectively. Such an object would be returned by, for example, well_response.

Author(s)

A. J. Barbour <andy.barbour@gmail.com>

See Also

kitagawa-package
Examples

```r
## Not run:
.nullchk(1:10) # OK
.nullchk(NULL) # error
.nullchk(c(1:10,NULL)) # error
.nullchk(NA) # error
.nullchk(c(1:10,NA)) # error

.in0to1(1:10) # error
.in0to1(NULL) # error
.in0to1(c(1:10,NULL)) # error
.in0to1(NA) # error
.in0to1(c(1:10,NA)) # error
.in0to1(c(1,NA)) # error

is.wrsp(1) # FALSE

## End(Not run)
```

---

**Description**

Logarithmic smoothing with loess

**Usage**

```r
logsmoo(x, y, x.is.log = FALSE, ...)
```

**Arguments**

- `x` numeric; the index series (cannot contain NA)
- `y` numeric; the series of values associated with `x`
- `x.is.log` logical; determines whether the series in `x` has been log-transformed already. If FALSE then log10 is used.
- `...` additional parameters (e.g., `span`) passed to `loess.smooth`

**Value**

The result of `loess.smooth`

**References**

See Also

`loess.smooth` and `approxfun`

Examples

```r
set.seed(11133)
n <- 101
lx <- seq(-1,1,length.out=n)
y <- rnorm(n) + cumsum(rnorm(n))
plot(lx, y, col='grey')
lines(logsmoo(lx, y, x.is.log=TRUE))
```

---

**logticks**

*Add proper logarithm ticks to a plot axis.*

**Description**

Add proper logarithm ticks to a plot axis.

**Usage**

```r
logticks(
  ax = 1,
  n.minor = 9,
  t.lims,
  t.ratio = 0.5,
  major.ticks = NULL,
  base = c("ten", "ln", "two"),
  ticks.only = FALSE,
  ...
)
```

**Arguments**

- `ax` numeric; the axis number to add tick-marks to
- `n.minor` numeric; the number of minor ticks to display
- `t.lims` numeric; the upper and lower tick limits (in log space)
- `t.ratio` numeric; the ratio of minor to major tick lengths.
- `major.ticks` numeric; the axis limits.
base numeric; the base of the logarithm (somewhat experimental)
ticks.only logical; on the axis
... additional parameters passed to the axis call for the major ticks.

Details

This uses pretty with n=5, and assumes that the data along the axis ax has already been transformed into its logarithm. Only integer exponents are labeled.
The functions log_ticks, log2_ticks, and log10_ticks are wrapper functions.
Set the axt parameter (e.g. xaxt) to 'n' in the original plot command to prevent adding default tick marks.

Author(s)

A. J. Barbour <andy.barbour@gmail.com>

References

This was modified from a post on StackOverflow: https://stackoverflow.com/questions/6955440/displaying-minor-logarithmic-ticks-in-x-axis-in-r

See Also

Other PlotUtilities: wrsp-methods

Examples

x <- 10^(0:8)
y <- 1:9
plot(log10(x),y,xaxt="n",xlab="x",xlim=c(0,9))
logticks()
logticks(ax=3, ticks.only=TRUE)
par(tcl=0.5) # have tick marks show up on inside instead
plot(log10(x),y,xaxt="n",xlab="x",xlim=c(0,9))
logticks()
logticks(ax=3, ticks.only=TRUE)

omega_constants Calculate any constants that depend on angular frequency ω

Description

This function accesses the appropriate method to calculate the ω-dependent constant associated with the choice of c.type.

This function is not likely to be needed by the user.
Usage

omega_constants(omega = 0, c.type = c("alpha", "diffusivity_time"), ...)

## Default S3 method:
omega_constants(omega = 0, c.type = c("alpha", "diffusivity_time"), ...)

Arguments

omega frequency, [rad/sec]
c.type the constant to calculate...

... additional params passed to calculator. In the case of c.type="alpha", set S., T., Rs.; and, in the case of c.type="diffusivity_time", set D. or S., T..

Details

What is "omega"?: The name is in reference to radial frequency $\omega$, which is defined as

$$\omega \equiv \frac{2\pi}{\tau}$$

where $\tau$ is the period of oscillation.

What is the "alpha" calculation?:
The parameter $\alpha$ is defined as

$$\alpha \equiv r_w \sqrt{\omega S/T}$$

where $r_w$ is the radius of the well, where $S$ is the storativity, and $T$ is transmissivity. See the parameter ... for details on how to pass these values.

This definition is common to many papers on the topic. For example, it corresponds to Equation 12 in Kitagawa et al (2011). Because the computation of $\alpha$ depends also on physical properties, additional parameters can be passed through (e.g. the transmissivity).

What is the "diffusivity_time" calculation?: This is a similar calculation to omega_norm. It uses the effective hydraulic diffusivity $D$, which is defined in this case as the ratio of transmissivity to storativity:

$$D \equiv \frac{T}{S}$$

Value

Values of the constant represented by c.type for the given parameters

Warnings Issued

In the case c.type='alpha', the parameters S., T., and Rs. should be passed; otherwise, values are assumed to ensure the calculation does not fail, and the results are essentially meaningless.

Warnings will be issued if any necessary parameters are missing, indicating default values were used.
omega_norm

Description

Dimensionless frequency from diffusivity and depth

Usage

omega_norm(omega, Diffusiv, z, invert = FALSE)

Arguments

omega numeric; angular frequency
Diffusiv numeric; hydraulic diffusivity
z numeric; depth
invert logical; should omega be taken as normalized frequency?

Details

Dimensionless frequency \( Q \) is defined as

\[
Q = \frac{z^2 \omega}{2D}
\]

where \( z \) is the well depth, \( \omega \) is the angular frequency and \( D \) is the hydraulic diffusivity.
Value

omega_norm returns dimensionless frequency, unless invert=TRUE where it will assume omega is
dimensionless frequency, and return radial frequency.

Author(s)

A. J. Barbour <andy.barbour@gmail.com>

See Also

open_well_response, kitagawa-package

Other utilities: sensing_volume()

open_well_response

Spectral response for an open well

Description

This is the primary function to calculate the response for an open (exposed to air) well.

Usage

open_well_response(omega, T., S., ...)

## Default S3 method:
open_well_response(
  omega,
  T.,
  S.,
  Rs. = (8/12) * (1200/3937),
  rho,
  grav,
  z,
  Hw,
  Ta,
  leak,
  freq.units = c("rad_per_sec", "Hz"),
  model = c("rojstaczer", "liu", "cooper", "hsieh", "wang"),
  as.pressure = TRUE,
  ...
)
Arguments

- **omega**: numeric; frequency, (see `freq.units`)
- **T**: numeric; effective aquifer transmissivity \([m^2/s]\)
- **S**: numeric; well storativity, \([\text{unitless}]\)
- **Rs**: numeric; the radius of the open (screened) section
- **rho**: numeric; fluid density (assumed if missing)
- **grav**: numeric; the local gravitational acceleration (assumed if missing)
- **z**: numeric; From Rojstaczer (1988): the depth from the water table (assumed if missing and if needed)
- **Hw**: numeric; height of water column above confined surface (assumed if missing and if needed)
- **Ta**: numeric; thickness of aquifer (assumed if missing and if needed)
- **leak**: numeric; specific leakage \([K'/b'][1/s]\)
- **freq.units**: character; setup the units of `omega`
- **as.pressure**: logical; should the response be relative to aquifer pressure? (default is aquifer head)

Details

As opposed to `well_response`, this calculates the theoretical, complex well response for an unsealed (open) well.

The response depends strongly on the physical properties given. Default values are assumed where reasonable—for instance, the pore-fluid is assumed to be water—but considerable care should be invested in the choice of parameters, especially in the case of starting parameters in an optimization scheme.

The responses returned here are, effectively, the amplification of water levels in a well, relative to the pressure head in the aquifer; or

\[
Z = \frac{z}{h} \equiv \frac{\rho g z}{p}
\]

If `as.pressure=TRUE`, then the responses are scaled by `rho*grav` so that they represent water levels relative to aquifer pressure:

\[
Z = \frac{z}{p}
\]

Not all parameters need to be given, but should be. For example, if either `rho` or `grav` are not specified, they are taken from `constants`. Parameters which do not end in \(\_\) do not need to be specified (they may be excluded); if they are missing, assumptions may be made and warnings will be thrown.

Value

An object with class `owrsp`
Models

"rojstaczer": Rojstaczer (1988) is based on measurements of water level and strain from volumetric or areal strainmeters.

"cooper", "hsieh", and "liu": Cooper et al (1965), Hsieh et al (1987) and Liu et al (1989) are based on measurements of water level and displacements from seismometers or strainmeters; these models are expressed succinctly in Roeloffs (1996).

The sense of the phase shift for the Liu and Rojstaczer models are reversed from their original presentation, in order to account for differences in sign convention.

"wang": Wang et al (2018) allows for specific leakage – vertical conductivity across a semi-permeable aquitard – but the perfectly confined case (i.e., Hsieh, et al 1987) is recovered when leakage is zero.

Author(s)

A. J. Barbour and J. Kennel

References

See kitagawa-package for references and more background.

See Also

well_response for the sealed-well equivalents, and owrsp-methods for a description of the class 'owrsp' and its methods.

Other WellResponseFunctions: well_response()

Examples

OWR <- open_well_response(1:10,1,1)
plot(OWR)
OWR <- open_well_response(1/(1:200),1,1,Ta=100,Hw=10,model="liu",freq.units="Hz")
plot(OWR)
Usage

```r
## S3 method for class 'owrsp'
as.data.frame(x, ...)

data.frame.owrsp(x, ...)

## S3 method for class 'owrsp'
print(x, n = 3, ...)

## S3 method for class 'owrsp'
summary(object, ...)

## S3 method for class 'summary.owrsp'
print(x, ...)

## S3 method for class 'owrsp'
lines(x, series = c("amp", "phs"), ...)

## S3 method for class 'owrsp'
points(x, series = c("amp", "phs"), pch = "+", ...)

## S3 method for class 'owrsp'
plot(
x,
xlims = c(-3, 1),
ylims = list(amp = NULL, phs = 185 * c(-1, 1)),
logamp = TRUE,
...
)
```

Arguments

- `x` : 'owrsp' object
- `...` : optional arguments
- `n` : numeric; the number of head and tail to print
- `object` : 'owrsp' object
- `series` : character; the series to plot (amplitude or phase)
- `pch` : point character, as in `par`
- `xlims` : limits for x-axis (applies to both amp and phs frames)
- `ylims` : optional list of limits for y-axis (i.e., list(amp=c(...), phs=c(...)))
- `logamp` : logical; should the amplitude be in log10 units

Details

The response information is a matrix with frequency, complex response \([\omega, Z_\omega(\omega)]\) where the units of \(\omega\) will be as they were input. The amplitude of \(Z\) is in meters per strain, and the phase is in radians.
sensing_volume

Calculate volume of fluids in the sensing region of the borehole.

Description

This function calculates the volume of fluid in the screened section, namely Equation 2 in Kitagawa et al (2011).

Usage

sensing_volume(rad_grout, len_grout, rad_screen, len_screen)
sensing_volume

Arguments

- **rad_grout**
  - radius of the grouting \([m]\)
- **len_grout**
  - length of the grouting \([m]\)
- **rad_screen**
  - radius of the screened interval \([m]\)
- **len_screen**
  - length of the screened interval \([m]\)

Details

Although typical scientific boreholes with water-level sensors are drilled very deeply, pore-fluids are only allowed to flow through a relatively short section, known as the "screened" section. The calculation assumes two pairs of radii and lengths: one for the cemented (grout) section, and another for the screened section.

The volume calculated is

\[
\pi R^2_C(L_C - L_S) + \pi R^2_S L_S
\]

where \(R\) and \(L\) denote radius and length respectively, and subscripts \(C\) and \(S\) denote the cemented and screened sections respectively.

This calculation assumes the measurement is for a sealed well.

Value

scalar, with units of \([m^3]\)

Author(s)

A. J. Barbour <andy.barbour@gmail.com>

See Also

- `well_response`

Other utilities: `omega_norm()`

Examples

```r
# dummy example
sensing_volume(1, 1, 1, 1)
#
# a more physically realistic calculation:
# Physical params applicable for B084 borehole
# (see: http://pbo.unavco.org/station/overview/B084/ for details)
#
Rc <- 0.0508 # m, radius of water-sensing (2in)
Lc <- 146.9 # m, length of grouted region (482ft)
Rs <- 3*Rc # m, radius of screened region (6in)
Ls <- 9.14 # m, length of screened region (30ft)
#
# calculate the sensing volume for the given well parameters
sensing_volume(Rc, Lc, Rs, Ls) # m**3, ~ 1.8
```
well_response  Spectral response for a sealed well

Description

This is the primary function to calculate the response for a sealed well.

Usage

well_response(omega, T., S., Vw., Rs., Ku., B., ...)

## Default S3 method:
well_response(
  omega,
  T.,
  S.,
  Vw.,
  Rs.,
  Ku.,
  B.,
  Avs,
  Aw,
  rho,
  Kf,
  grav,
  freq.units = c("rad_per_sec", "Hz"),
  as.pressure = TRUE,
  ...
)

Arguments

omega  frequency, (see freq.units)
T.  effective aquifer transmissivity \([m^2/s]\)
S.  well storativity, [unitless]
Vw.  well volume, \([m^3]\)
Rs.  radius of screened portion, \([m]\)
Ku.  undrained bulk modulus, [Pa]
B.  Skempton’s coefficient, [unitless, bounded]
...  additional arguments
Avs  amplification factor for volumetric strain \(E_{kk,obs}/E_{kk},{}\)
Aw  amplification factor of well volume change for \(E_{kk},{}\)
rho  fluid density \([kg/m^3]\)
Kf  bulk modulus of fluid, [Pa]
well_response

`grav`  local gravitational acceleration \([m/s^2]\)
`freq.units`  set the units of omega
`as.pressure`  logical; should the response for water pressure? (default is water height)

**Details**

The response depends strongly on the physical properties given. Default values are assumed where reasonable—for instance, the pore-fluid is assumed to be water—but considerable care should be invested in the choice of parameters, unless the function is used in an optimization scheme.

Assumed values are:

- `Avs`  1  amplification factor for volumetric strain
- `Aw`  1  amplification factor for water well

The responses returned here are, effectively, the amplification of water levels in a well, relative to the aquifer strain; or

\[
Z = \frac{z}{\epsilon} = \frac{p}{\rho g \epsilon}
\]

If `as.pressure=TRUE`, then the responses are scaled by `rho*grav` so that they represent water pressure relative to aquifer strain:

\[
Z = \frac{p}{\epsilon}
\]

Not all parameters need to be given, but should be. For example, if either `rho` or `grav` are not specified, they are taken from `constants`. Parameters which do not end in `.do not need to be specified (they may be excluded); if they are missing, warnings will be thrown.

**Value**

An object with class `'wrsp'`

**Author(s)**

A. J. Barbour

**References**

See `kitagawa-package` for references and more background.

**See Also**

`open_well_response` for the open-well equivalents `wrsp-methods` for a description of the class `'wrsp'` and its methods, `sensing_volume` to easily estimate the volume \(V_w\), and `kitagawa-package` for references and more background.

Other WellResponseFunctions: `open_well_response()`
Examples

#### dummy example
well_response(1:10, T.=1, S.=1, Vw.=1, Rs.=1, Ku.=1, B.=1)

#### a more physically realistic calculation:

# Physical params applicable for B084 borehole
# (see: http://pbo.unavco.org/station/overview/B084/ for details)
#
Rc <- 0.0508 # m, radius of water-sensing (2in)
Lc <- 146.9 # m, length of grouted region (482ft)
Rs <- 3*Rc # m, radius of screened region (6in)
Ls <- 9.14 # m, length of screened region (30ft)
#
# calculate the sensing volume for the given well parameters
Volw <- sensing_volume(Rc, Lc, Rs, Ls) # m**3, ~1.8
#
Frqs <- 10**seq.int(from=-4,to=0,by=0.1) # log10-space
head(Rsp <- well_response(omega=Frqs, T.=1e-6, S.=1e-5,
Vw.=Volw, Rs.=Rs, Ku.=40e9, B.=0.2, freq.units="Hz"))

# Access plot.wrsp:
plot(Rsp)

wrsp-methods

Generic methods for objects with class 'wrsp'.

Description

An object with class 'wrsp' is a list containing the response information, and the mechanical, hydraulic, and material properties used to generate the response for a sealed well.

Usage

```r
## S3 method for class 'wrsp'
as.data.frame(x, ...)

data.frame.wrsp(x, ...)

## S3 method for class 'wrsp'
print(x, n = 3, ...)

## S3 method for class 'wrsp'
summary(object, ...)

## S3 method for class 'summary.wrsp'
print(x, ...)
```
## S3 method for class 'wrsp'
lines(x, series = c("amp", "phs"), ...)

## S3 method for class 'wrsp'
points(x, series = c("amp", "phs"), pch = "+", ...)

## S3 method for class 'wrsp'
plot(
  x,
  xlims = c(-3, 1),
  ylims = list(amp = NULL, phs = 185 * c(-1, 1)),
  logamp = TRUE,
  ...
)
kplot(x, ...)

## S3 method for class 'wrsp'
kplot(
  x,
  xlims = c(-3, 1),
  ylims = list(amp = NULL, phs = 185 * c(-1, 1)),
  logamp = TRUE,
  ...
)

### Arguments

- **x**
  - 'wrsp' object
- **...**
  - optional arguments
- **n**
  - numeric; the number of head and tail to print
- **object**
  - 'wrsp' object
- **series**
  - character; the series to plot (amplitude or phase)
- **pch**
  - point character, as in `par`
- **xlims**
  - limits for x-axis (applies to both amp and phs frames)
- **ylims**
  - optional list of limits for y-axis (i.e., `list(amp=c(..),phs=c(..))`)
- **logamp**
  - logical; should the amplitude be in log10 units

### Details

The response information is a matrix with frequency, complex response \( \omega, Z_\alpha(\omega) \) where the units of \( \omega \) will be as they were input. The amplitude of \( Z \) is in meters per strain, and the phase is in radians.

`kplot` was previously a standalone function, but is now simply a reference to `plot.wrsp`.
Author(s)

A. J. Barbour <andy.barbour@gmail.com>

See Also

well_response

kitagawa-package

Other PlotUtilities: logticks()

Examples

W <- well_response(1:10, T.=1, S.=1, Vw.=1, Rs.=1, Ku.=1, B.=1)
str(W)
print(W)
print(summary(W))
#
# Plot the response
plot(rnorm(10), xlim=c(-1,11), ylim=c(-2,2))
lines(W)
lines(W, "phs", col="red")
points(W)
points(W, "phs")
#
Wdf <- as.data.frame(W)
plot(Mod(wellresp) ~ omega, Wdf) # amplitude
plot(Arg(wellresp) ~ omega, Wdf) # phase
#
# or use the builtin method plot.wrsp
plot(W)
# change limits:
plot(W, xlims=c(-1,1), ylims=list(amp=c(5,8), phs=185*c(-1,1)))
Index

* ConstantsCalculators
  alpha_constants, 2
  kitagawa-constants, 5
  omega_constants, 9

* PlotUtilities
  logticks, 8
  wrsp-methods, 20

* WellResponseFunctions
  open_well_response, 12
  well_response, 18

* utilities
  omega_norm, 11
  sensing_volume, 16
  .in0tol, 6
  .in0tol (kitagawa-utilities), 6
  .nullchk, 6
  .nullchk (kitagawa-utilities), 6

alpha_constants, 2, 6, 11
approxfun, 8
as.data.frame.owrsp (owrsp-methods), 14
as.data.frame.wrsp (wrsp-methods), 20

constants, 5, 13, 19
constants (kitagawa-constants), 5
cross_spectrum, 4

data.frame.owrsp (owrsp-methods), 14
data.frame.wrsp (wrsp-methods), 20
deltat, 4

is.owrsp, 6
is.owrsp (kitagawa-utilities), 6
is.wrsp, 6
is.wrsp (kitagawa-utilities), 6

kitagawa-constants, 5
kitagawa-utilities, 6
kitplot, 21
kitplot (wrsp-methods), 20

lines.owrsp (owrsp-methods), 14
lines.wrsp (wrsp-methods), 20
loess.smooth, 7, 8
log10_ticks, 9
log10_ticks (logticks), 8
log2_ticks, 9
log2_ticks (logticks), 8
log_ticks, 9
log_ticks (logticks), 8
logsmoo, 7
logticks, 8, 22

omega_constants, 3, 6, 9
omega_norm, 10, 11, 12, 17
open_well_response, 6, 12, 12, 16, 19
owrsp (owrsp-methods), 14
owrsp-methods, 14

par, 15, 21
plot.owrsp (owrsp-methods), 14
plot.wrsp, 21
plot.wrsp (wrsp-methods), 20
points.owrsp (owrsp-methods), 14
points.wrsp (wrsp-methods), 20
pretty, 9
print.owrsp (owrsp-methods), 14
print.summary.owrsp (owrsp-methods), 14
print.summary.wrsp (wrsp-methods), 20
print.wrsp (wrsp-methods), 20
pspectrum, 4

sensing_volume, 12, 16, 19
summary.owrsp (owrsp-methods), 14
summary.wrsp (wrsp-methods), 20

well_response, 3, 6, 11, 13, 14, 17, 18, 22
wrsp (wrsp-methods), 20
wrsp-methods, 20