# Package 'gsw'

August 9, 2017

**Version** 1.0-5

**Date** 2017-08-08

**Title** Gibbs Sea Water Functions



**Maintainer** Dan Kelley <dan.kelley@dal.ca>

**Depends** R (>= 2.15), testthat

**Suggests** knitr,

**BugReports** https://github.com/TEOS-10/GSW-R/issues

**Description** Provides an interface to the Gibbs 'SeaWater' ('TEOS-10') C library, version 3.05-4 (commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec', dated 2017-08-07,
available at <https://github.com/TEOS-10/GSW-C>, which stems from 'Matlab' and other code written by members of Working Group 127 of 'SCOR'/'IAPSO' (Scientific Committee on Oceanic Research / International Association for the Physical Sciences of the Oceans).

**URL** http://teos-10.github.io/GSW-R/index.html

**License** GPL (>= 2) | file LICENSE

**LazyLoad** yes

**LazyData** no

**Encoding** UTF-8

**RoxygenNote** 6.0.1

**VignetteBuilder** knitr

**NeedsCompilation** yes

**Author** Dan Kelley [aut, cre, cph] (C wrapper plus R code, tests, and documentation),
Clark Richards [aut, cph] (C wrapper plus R code, tests, and

documentation),
WG127 SCOR/IAPSO [aut, cph] (Original 'Matlab' and derived code)

**Repository** CRAN

**Date/Publication** 2017-08-09 21:05:24 UTC

# R topics documented:

---

`argfix` *Reshape list elements to match that of the first element*

---

## Description

This is mainly used within gsw, to ensure that arguments sent to the C functions are of equal length. This is a convenience, for processing data that often have this condition. For example, a CTD profile is likely to have many values for SP, t, and p, but just a single value for each of longitude and latitude. It is important to call argfix() to handle such cases, because otherwise the underlying C code will be looking past the end of the vectors storing longitude and latitude, which can yield odd results or even segmentation faults.

## Usage

```
argfix(list)
```

## Arguments

| | |
|---|---|
| `list` | A list of elements, typically arguments that will be used in GSW functions. |

## Value

A list with all elements of same shape (length or dimension).

gsw — *R implementation of Thermodynamic Equation Of Seawater - 2010 (TEOS-10)*

## Description

Provides an R interface to the TEOS-10 / GSW (Gibbs Sea Water) library, partly for use by the oce package (see http://dankelley.github.io/oce) and partly for general use. It is assumed that users are familiar with the science and methodology of GSW, and that the package vignette (obtained by typing `vignette("gsw")` in an R window) provides enough orientation to get users started with the `gsw` functions.

## Details

`gsw` was developed using open-source methodologies, on the GitHub site (https://github.com/TEOS-10/GSW-R), which is part of a set of sites dedicated to GSW formulations in various languages.

The `gsw` system is to link R functions with the C version of the TEOS-10 library. The R function names are chosen to match those of the Matlab version of GSW, and the function arguments also match with one exception: in `gsw`, longitude and latitude are indicated with their full names, whereas in Matlab they are indicated with `long` and `lat`; since R permits abbreviated function arguments, the shortened names can be used in `gsw` as well.

The documentation for the `gsw` functions focuses mainly on the arguments and return values, relying on links to the TEOS-10 webpages for details.

See http://www.teos-10.org/pubs/gsw/html/gsw_contents.html for a list of the TEOS-10 functions and http://teos-10.github.io/GSW-R/documentation.html for a list of the functions implemented in the present package.

Each function is tested during the building of the package, which means that results are guaranteed to match those of the equivalent Matlab functions to at least 8 digits.

A significant difference from the Matlab case is in the inspection of the dimensions of arguments. The Matlab library has rules for expanding some arguments to match others. For example, if Practical Salinity is a matrix and pressure is a single value, then that single pressure is used throughout a calculation of Absolute Salinity. This convenience is only partly mimicked in the present package. Since the underlying C code works on vectors, the R functions in `gsw` start by transforming the arguments accordingly. This involves using `rep` on each argument to get something with length matching the first argument, and, after the computation is complete, converting the return value into a matrix, if the first argument was a matrix. There are some exceptions to this, however. For example, `gsw_SA_from_SP` and similar functions can handle the case in which the `SA` argument is a matrix and `longitude` and `latitude` are vectors sized to match. This can be handy with gridded datasets. However, the careful analyst will probably prefer to avoid this and other conveniences, supplying properly-matched arguments from the outset.

gsw_adiabatic_lapse_rate_from_CT
*Adiabatic Lapse Rate*

## Description

Note that the unit is K/Pa; multiply by 1e4 to get the more useful K/dbar.

## Usage

```
gsw_adiabatic_lapse_rate_from_CT(SA, CT, p)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

adiabatic lapse rate (note unconventional unit) [ K/Pa ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_adiabatic_lapse_rate_from_CT.html

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.7856, 28.4329, 22.8103, 10.2600,  6.8863,  4.4036)
p <- c(      10,      50,     125,     250,     600,    1000)
lr <- gsw_adiabatic_lapse_rate_from_CT(SA, CT, p)
expect_equal(lr*1e7, c(0.240199646230069, 0.238457486976761, 0.203635157319712,
                       0.119829566859790, 0.100052760967308, 0.087773070307283))
```

gsw_adiabatic_lapse_rate_ice
*Adiabatic Lapse Rate of Ice*

### Description

Note that the unit is K/Pa; multiply by 1e4 to get the more useful K/dbar.

### Usage

```
gsw_adiabatic_lapse_rate_ice(t, p)
```

### Arguments

| | |
|---|---|
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

adiabatic lapse rate (note unconventional unit) [ K/Pa ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_adiabatic_lapse_rate_ice.html

### Examples

```
t  <- c(-10.7856, -13.4329, -12.8103, -12.2600, -10.8863, -8.4036)
p <- c(       10,       50,      125,      250,      600,    1000)
lr <- gsw_adiabatic_lapse_rate_ice(t, p)
expect_equal(lr*1e7, c(0.218777853913651, 0.216559115188599, 0.216867659957613,
                     0.216988337914416, 0.217182707402780, 0.218100558740840))
```

gsw_alpha *Thermal expansion coefficient with respect to Conservative Temperature*

## Description

Thermal expansion coefficient with respect to Conservative Temperature, using the 75-term equation for specific volume.

## Usage

```
gsw_alpha(SA, CT, p)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

thermal expansion coefficient with respect to Conservative Temperature [ 1/K ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_alpha.html

## See Also

Other things related to density: gsw_CT_from_rho, gsw_CT_maxdensity, gsw_SA_from_rho, gsw_alpha_on_beta, gsw_alpha_wrt_t_exact, gsw_alpha_wrt_t_ice, gsw_beta_const_t_exact, gsw_beta, gsw_pot_rho_t_exact, gsw_rho_alpha_beta, gsw_rho_first_derivatives_wrt_enthalpy, gsw_rho_first_derivatives, gsw_rho_ice, gsw_rho_t_exact, gsw_rho, gsw_sigma0, gsw_sigma1, gsw_sigma2, gsw_sigma3, gsw_sigma4, gsw_specvol_alpha_beta, gsw_specvol_anom_standard, gsw_specvol_ice, gsw_specvol_t_exact, gsw_specvol

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.7856, 28.4329, 22.8103, 10.2600,  6.8863,  4.4036)
p <-  c(     10,      50,     125,     250,     600,    1000)
alpha <- gsw_alpha(SA,CT,p)
expect_equal(alpha*1e3, c( 0.324464211877393, 0.322610094680523, 0.281335030247435,
                        0.173529986885424, 0.146898108553385, 0.130265123640082))
```

---

`gsw_alpha_on_beta` *Thermal expansion coefficient over haline contraction coefficient*

---

## Description

Thermal expansion coefficient over haline contraction coefficient, using the 75-term equation for specific volume.

## Usage

```
gsw_alpha_on_beta(SA, CT, p)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

ratio of thermal expansion coefficient to haline contraction coefficient [ (g/kg)/K ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_alpha_on_beta.html

### See Also

Other things related to density: gsw_CT_from_rho, gsw_CT_maxdensity, gsw_SA_from_rho, gsw_alpha_wrt_t_exact, gsw_alpha_wrt_t_ice, gsw_alpha, gsw_beta_const_t_exact, gsw_beta, gsw_pot_rho_t_exact, gsw_rho_alpha_beta, gsw_rho_first_derivatives_wrt_enthalpy, gsw_rho_first_derivatives, gsw_rho_ice, gsw_rho_t_exact, gsw_rho, gsw_sigma0, gsw_sigma1, gsw_sigma2, gsw_sigma3, gsw_sigma4, gsw_specvol_alpha_beta, gsw_specvol_anom_standard, gsw_specvol_ice, gsw_specvol_t_exact, gsw_specvol

### Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
p <-  c(     10,      50,     125,     250,     600,    1000)
alpha_on_beta <- gsw_alpha_on_beta(SA,CT,p)
expect_equal(alpha_on_beta, c(0.452468543022009, 0.449601695030057, 0.387140203094424,
                              0.230778871228268, 0.193747796234162, 0.170946048860385))
```

---

## gsw_alpha_wrt_t_exact *Thermal expansion coefficient with respect to in-situ temperature*

---

### Description

Thermal expansion coefficient with respect to in-situ temperature.

### Usage

```
gsw_alpha_wrt_t_exact(SA, t, p)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

thermal expansion coefficient with respect to in-situ temperature [ 1/K ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_alpha_wrt_t_exact.html

### See Also

Other things related to density: gsw_CT_from_rho, gsw_CT_maxdensity, gsw_SA_from_rho, gsw_alpha_on_beta, gsw_alpha_wrt_t_ice, gsw_alpha, gsw_beta_const_t_exact, gsw_beta, gsw_pot_rho_t_exact, gsw_rho_alpha_beta, gsw_rho_first_derivatives_wrt_enthalpy, gsw_rho_first_derivatives, gsw_rho_ice, gsw_rho_t_exact, gsw_rho, gsw_sigma0, gsw_sigma1, gsw_sigma2, gsw_sigma3, gsw_sigma4, gsw_specvol_alpha_beta, gsw_specvol_anom_standard, gsw_specvol_ice, gsw_specvol_t_exact, gsw_specvol

### Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
t <-  c(28.7856, 28.4329, 22.8103, 10.2600,  6.8863,  4.4036)
p <-  c(     10,      50,     125,     250,     600,    1000)
alpha_wrt_t_exact <- gsw_alpha_wrt_t_exact(SA,t,p)
expect_equal(alpha_wrt_t_exact*1e3, c(0.325601747227247, 0.323448083851267, 0.281413883319329,
                               0.172825692975230, 0.145569941503599, 0.128362986933288))
```

---

## gsw_alpha_wrt_t_ice    *Ice Thermal Expansion Coefficient with Respect to in-situ Temperature*

---

### Description

Thermal expansion coefficient of ice, with respect to in-situ temperature.

### Usage

```
gsw_alpha_wrt_t_ice(t, p)
```

### Arguments

| | |
|---|---|
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

thermal expansion coefficient with respect to in-situ temperature [ 1/K ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_alpha_wrt_t_ice.html

## See Also

Other things related to density: gsw_CT_from_rho, gsw_CT_maxdensity, gsw_SA_from_rho, gsw_alpha_on_beta, gsw_alpha_wrt_t_exact, gsw_alpha, gsw_beta_const_t_exact, gsw_beta, gsw_pot_rho_t_exact, gsw_rho_alpha_beta, gsw_rho_first_derivatives_wrt_enthalpy, gsw_rho_first_derivatives, gsw_rho_ice, gsw_rho_t_exact, gsw_rho, gsw_sigma0, gsw_sigma1, gsw_sigma2, gsw_sigma3, gsw_sigma4, gsw_specvol_alpha_beta, gsw_specvol_anom_standard, gsw_specvol_ice, gsw_specvol_t_exact, gsw_specvol

## Examples

```
t <- c(-10.7856, -13.4329, -12.8103, -12.2600, -10.8863, -8.4036)
p <-  c(     10,       50,      125,      250,      600,    1000)
alpha <- gsw_alpha_wrt_t_ice(t, p)
expect_equal(alpha*1e3, c(0.154472408751279, 0.153041866100900, 0.153232698269327,
                        0.153297634665747, 0.153387461617896, 0.153938395452558))
```

---

gsw_beta *Haline contraction coefficient at constant Conservative Temperature*

---

## Description

Haline contraction coefficient with respect to Conservative Temperature, using the 75-term equation for specific volume.

## Usage

```
gsw_beta(SA, CT, p)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

Haline contraction coefficient at constant Conservative Temperature [ kg/g ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_beta.html

## See Also

Other things related to density: gsw_CT_from_rho, gsw_CT_maxdensity, gsw_SA_from_rho, gsw_alpha_on_beta, gsw_alpha_wrt_t_exact, gsw_alpha_wrt_t_ice, gsw_alpha, gsw_beta_const_t_exact, gsw_pot_rho_t_exact, gsw_rho_alpha_beta, gsw_rho_first_derivatives_wrt_enthalpy, gsw_rho_first_derivatives, gsw_rho_ice, gsw_rho_t_exact, gsw_rho, gsw_sigma0, gsw_sigma1, gsw_sigma2, gsw_sigma3, gsw_sigma4, gsw_specvol_alpha_beta, gsw_specvol_anom_standard, gsw_specvol_ice, gsw_specvol_t_exact, gsw_specvol

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.7856, 28.4329, 22.8103, 10.2600,  6.8863,  4.4036)
p <-  c(     10,      50,     125,     250,     600,    1000)
beta <- gsw_beta(SA,CT,p)
expect_equal(beta, 1e-3*c(0.717521909550091, 0.717657376442386, 0.726169785748549,
                          0.750420924314564, 0.754903052075032, 0.756841573481865))
```

---

gsw_beta_const_t_exact    *Haline contraction coefficient at constant in-situ temperature*

---

## Description

Haline contraction coefficient at constant in-situ temperature.

## Usage

```
gsw_beta_const_t_exact(SA, t, p)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

Haline contraction coefficient at constant in-situ temperature [ kg/g ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_beta_const_t_exact.html

## See Also

Other things related to density: gsw_CT_from_rho, gsw_CT_maxdensity, gsw_SA_from_rho, gsw_alpha_on_beta, gsw_alpha_wrt_t_exact, gsw_alpha_wrt_t_ice, gsw_alpha, gsw_beta, gsw_pot_rho_t_exact, gsw_rho_alpha_beta, gsw_rho_first_derivatives_wrt_enthalpy, gsw_rho_first_derivatives, gsw_rho_ice, gsw_rho_t_exact, gsw_rho, gsw_sigma0, gsw_sigma1, gsw_sigma2, gsw_sigma3, gsw_sigma4, gsw_specvol_alpha_beta, gsw_specvol_anom_standard, gsw_specvol_ice, gsw_specvol_t_exact, gsw_specvol

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
t <- c( 28.7856, 28.4329, 22.8103, 10.2600,  6.8863,  4.4036)
p <- c(      10,      50,     125,     250,     600,    1000)
b <- gsw_beta_const_t_exact(SA, t, p)
expect_equal(b*1e3, c(0.731120837010429, 0.731071779078011, 0.736019128913071,
                    0.753810501711847, 0.757259405338257, 0.758649268096996))
```

gsw_cabbeling *Cabbeling coefficient*

## Description

Cabbeling coefficient (75-term equation)

## Usage

```
gsw_cabbeling(SA, CT, p)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

Cabbeling coefficient with respect to Conservative Temperature [ 1/(K^2) ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_cabbeling.html

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
p <-  c(     10,      50,     125,     250,     600,    1000)
cabbeling <- gsw_cabbeling(SA,CT,p)
expect_equal(cabbeling*1e4, c(0.086645721047423, 0.086837829466794, 0.092525582052438,
                              0.108884336975401, 0.112971197222338, 0.115483896148927))
```

gsw_chem_potential_water_ice *Chemical Potential of Ice*

## Description

Chemical Potential of Ice

## Usage

```
gsw_chem_potential_water_ice(t, p)
```

## Arguments

| | |
|---|---|
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

chemical potential [ J/kg ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_chem_potential_water_ice.html

## See Also

Other things related to chemical potential: gsw_chem_potential_water_t_exact

## Examples

```
t <- c(-10.7856, -13.4329, -12.8103, -12.2600,  -10.8863,  -8.4036)
p <- c(      10,       50,      125,      250,      600,      1000)
pot <- gsw_chem_potential_water_ice(t, p)
expect_equal(pot/1e4, c(-1.340648365149857, -1.644921413491445, -1.480991678890353,
                      -1.272436055728805, -0.711509477199393, 0.045575390357792))
```

## gsw_chem_potential_water_t_exact

*Chemical Potential of Water in Seawater*

### Description

Chemical Potential of Water in Seawater

### Usage

```
gsw_chem_potential_water_t_exact(SA, t, p)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

chemical potential [ J/kg ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_chem_potential_water_t_exact.html

### See Also

Other things related to chemical potential: gsw_chem_potential_water_ice

### Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
t <- c(28.7856, 28.4329, 22.8103, 10.2600,  6.8863,  4.4036)
p <- c(      10,       50,      125,      250,      600,      1000)
pot <- gsw_chem_potential_water_t_exact(SA, t, p)
expect_equal(pot, c(-8.545561146284534, -8.008085548342105, -5.103980139874876,
                   -0.634067782745442, 3.335566803473286, 7.555434445971858))
```

gsw_cp_ice *Specific heat to ice*

## Description

Specific heat of ice

## Usage

```
gsw_cp_ice(t, p)
```

## Arguments

| | |
|---|---|
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

specific heat [ J/(K*kg) ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_cp_ice.html

## Examples

```
t <- c(-10.7856, -13.4329, -12.8103, -12.2600, -10.8863, -8.4036)
p <-  c(     10,      50,     125,     250,     600,    1000)
cp <- gsw_cp_ice(t, p)
expect_equal(cp, c(2017.314262094657, 1997.830122682709, 2002.281331375396,
                 2006.127319545421, 2015.676303959609, 2033.308170371559))
```

## gsw_cp_t_exact *Isobaric heat capacity*

### Description

Isobaric heat capacity

### Usage

```
gsw_cp_t_exact(SA, t, p)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

heat capacity [ J/(kg*K) ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_cp_t_exact.html

### Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
t <-  c(28.7856, 28.4329, 22.8103, 10.2600,  6.8863,  4.4036)
p <-  c(     10,      50,     125,     250,     600,    1000)
cp_t_exact <- gsw_cp_t_exact(SA, t, p)
expect_equal(cp_t_exact/1e3, c(4.002888003958537, 4.000980283927373, 3.995546468894633,
                               3.985076769021370, 3.973593843482723, 3.960184084786622))
```

gsw_CT_first_derivatives
*First Derivatives of Conservative Temperature*

## Description

First Derivatives of Conservative Temperature

## Usage

```
gsw_CT_first_derivatives(SA, pt)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| pt | potential temperature (ITS-90) [ degC ] |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

A list containing CT_SA [ K/(g/kg) ], the derivative of Conservative Temperature with respect to Absolute Salinity, and CT_pt [ unitless ], the derivative of Conservative Temperature with respect to potential temperature.

## References

http://www.teos-10.org/pubs/gsw/html/gsw_CT_first_derivatives.html

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
pt <- c(28.7832, 28.4209, 22.7850, 10.2305,  6.8292,  4.3245)
r <- gsw_CT_first_derivatives(SA, pt)
expect_equal(r$CT_SA, c(-0.041981092877806, -0.041558140199508, -0.034739209004865,
                        -0.018711103772892, -0.014075941811725, -0.010571716552295))
expect_equal(r$CT_pt, c(1.002814937296636, 1.002554817053239, 1.001645140295163,
                        1.000003771100520, 0.999716359504731, 0.999474326580093))
```

## gsw_CT_first_derivatives_wrt_t_exact

*Derivatives of Conservative Temperature with Respect to or at Constant in-situ Temperature*

### Description

Derivatives of Conservative Temperature with Respect to or at Constant in-situ Temperature

### Usage

```
gsw_CT_first_derivatives_wrt_t_exact(SA, t, p)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

A list containing CT_SA_wrt_t [ K/(g/kg) ], the derivative of Conservative Temperature with respect to Absolute Salinity at constant temperature and pressure, CT_t_wrt_t [ unitless], the derivative of Conservative Temperature with respect to temperature at constant Absolute Salinity and pressure, and CT_p_wrt_t, the derivative of Conservative Temperature with respect to pressure at constant Absolute Salinity and temperature.

### References

http://www.teos-10.org/pubs/gsw/html/gsw_CT_first_derivatives_wrt_t_exact.html

### Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
t <- c( 28.7856, 28.4329, 22.8103, 10.2600,  6.8863,  4.4036)
p <- c(      10,      50,     125,     250,     600,    1000)
r <- gsw_CT_first_derivatives_wrt_t_exact(SA, t, p)
expect_equal(r$CT_SA_wrt_t, c(-0.041988694538987, -0.041596549088952, -0.034853545749326,
                              -0.019067140454607, -0.015016439826591, -0.012233725491373))
```

```
expect_equal(r$CT_t_wrt_t, c(1.002752642867571, 1.002243118597902, 1.000835702767227,
                             0.998194915250648, 0.995219303532390, 0.991780205482695))
expect_equal(r$CT_p_wrt_t/1e-7, c(-0.241011880838437, -0.239031676279078, -0.203649928441505,
                              -0.119370679226136, -0.099140832825342, -0.086458168643579))
```

---

`gsw_CT_freezing` *Conservative Temperature of Freezing Seawater*

---

### Description

Conservative Temperature of Freezing Seawater

### Usage

```
gsw_CT_freezing(SA, p, saturation_fraction = 1)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| saturation_fraction | saturation fraction of dissolved air in seawater |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

Conservative Temperature at freezing of seawater [ degC ].

### References

http://www.teos-10.org/pubs/gsw/html/gsw_CT_freezing.html

### Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
p <-  c(     10,      50,     125,     250,     600,    1000)
saturation_fraction <- 1
CT <- gsw_CT_freezing(SA, p, saturation_fraction)
expect_equal(CT, c(-1.899683776424096, -1.940791867869104, -2.006240664432488,
                   -2.092357761318778, -2.359300831770506, -2.677162675412748))
```

## gsw_CT_freezing_first_derivatives *First Derivatives of Conservative Temperature for Freezing Water*

### Description

First Derivatives of Conservative Temperature for Freezing Water

### Usage

```
gsw_CT_freezing_first_derivatives(SA, p, saturation_fraction = 1)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| saturation_fraction | fraction of air in water [unitless] |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

A list containing CTfreezing_SA [ K/(g/kg) ], the derivative of Conservative Temperature with respect to Absolute Salinity at constant potential temperature, and CTfreezing_p [ unitless], the derivative of Conservative Temperature with respect to pressure at constant Absolute Salinity.

### References

http://www.teos-10.org/pubs/gsw/html/gsw_CT_freezing_first_derivatives.html

### Examples

```
SA <- c(                  34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
p <- c(                        10,      50,     125,     250,     600,    1000)
saturation_fraction <- c(      1,     0.8,     0.6,     0.5,     0.4,       0)
r <- gsw_CT_freezing_first_derivatives(SA, p, saturation_fraction)
expect_equal(r$CTfreezing_SA, c(-0.058193253897272, -0.058265158334170, -0.058345661671901,
                              -0.058373842446463, -0.058534544740846, -0.058730846361252))
expect_equal(r$CTfreezing_p/1e-7, c(-0.765300390432684, -0.766942996466485, -0.769892679988284,
                              -0.774561011527902, -0.787769143040504, -0.802771548245855))
```

gsw_CT_freezing_first_derivatives_poly
*First Derivatives of Conservative Temperature for Freezing Water (Polynomial version)*

## Description

First Derivatives of Conservative Temperature for Freezing Water (Polynomial version)

## Usage

```
gsw_CT_freezing_first_derivatives_poly(SA, p, saturation_fraction = 1)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| saturation_fraction | fraction of air in water [unitless] |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

A list containing CTfreezing_SA [ K/(g/kg) ], the derivative of Conservative Temperature with respect to Absolute Salinity at constant potential temperature, and CTfreezing_p [ unitless], the derivative of Conservative Temperature with respect to pressure at constant Absolute Salinity.

## References

http://www.teos-10.org/pubs/gsw/html/gsw_CT_freezing_first_derivatives_poly.html

## Examples

```
SA <- c(                 34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
p <- c(                       10,      50,     125,     250,     600,    1000)
saturation_fraction <- c(      1,     0.8,     0.6,     0.5,     0.4,       0)
r <- gsw_CT_freezing_first_derivatives_poly(SA, p, saturation_fraction)
expect_equal(r$CTfreezing_SA, c(-0.058191181082769, -0.058263310660779, -0.058343573188907,
                                -0.058370514075271, -0.058528023214462, -0.058722959729433))
expect_equal(r$CTfreezing_p/1e-7, c(-0.765690732336706, -0.767310677213890, -0.770224214219328,
                                -0.774843488962665, -0.787930403016584, -0.802821704643775))
```

gsw_CT_freezing_poly *Conservative Temperature Freezing Point (Polynomial version)*

## Description

Conservative Temperature Freezing Point (Polynomial version)

## Usage

```
gsw_CT_freezing_poly(SA, p, saturation_fraction = 1)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| saturation_fraction | saturation fraction of dissolved air in seawater |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

Conservative Temperature at freezing of seawater [ degC ].

## References

http://www.teos-10.org/pubs/gsw/html/gsw_CT_freezing_poly.html

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
p <-  c(     10,      50,     125,     250,     600,    1000)
saturation_fraction <- 1
CT_freezing <- gsw_CT_freezing(SA, p, saturation_fraction)
expect_equal(CT_freezing, c(-1.899683776424096, -1.940791867869104, -2.006240664432488,
                            -2.092357761318778, -2.359300831770506, -2.677162675412748))
```

`gsw_CT_from_enthalpy` *Conservative Temperature from Enthalpy*

### Description

Conservative Temperature from Enthalpy

### Usage

```
gsw_CT_from_enthalpy(SA, h, p)
```

### Arguments

| | |
|---|---|
| `SA` | Absolute Salinity [ g/kg ] |
| `h` | specific enthalpy [ J/kg ] |
| `p` | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

Conservative Temperature [ degC ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_CT_from_enthalpy.html

### See Also

Other things related to enthalpy: `gsw_dynamic_enthalpy`, `gsw_enthalpy_CT_exact`, `gsw_enthalpy_diff`, `gsw_enthalpy_first_derivatives_CT_exact`, `gsw_enthalpy_first_derivatives`, `gsw_enthalpy_ice`, `gsw_enthalpy_t_exact`, `gsw_enthalpy`, `gsw_frazil_properties_potential_poly`, `gsw_frazil_properties_potent` `gsw_pot_enthalpy_from_pt_ice_poly`, `gsw_pot_enthalpy_from_pt_ice`, `gsw_pot_enthalpy_ice_freezing_poly`, `gsw_pot_enthalpy_ice_freezing`, `gsw_pt_from_pot_enthalpy_ice_poly`, `gsw_pt_from_pot_enthalpy_ice`, `gsw_specvol_first_derivatives_wrt_enthalpy`, `gsw_specvol_first_derivatives`

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
h <- c(1.15103e5, 1.14014e5, 0.92180e5, 0.43255e5, 0.33087e5, 0.26970e5)
p <- c(       10,        50,       125,       250,       600,      1000)
pt <- c(28.7832, 28.4209, 22.7850, 10.2305,  6.8292,  4.3245)
CT <- gsw_CT_from_enthalpy(SA, h, p)
expect_equal(CT, c(28.809854569021972, 28.439026483379287, 22.786196534098817,
  10.226106994920777, 6.827159682675204, 4.323428660306681))
```

---

# gsw_CT_from_entropy *Conservative Temperature from Entropy*

---

## Description

Conservative Temperature from Entropy

## Usage

```
gsw_CT_from_entropy(SA, entropy)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| entropy | specific entropy [ J/(degC*kg) ] |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

Conservative Temperature [ degC ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_CT_from_entropy.html

## See Also

Other things related to entropy: gsw_entropy_first_derivatives, gsw_entropy_from_pt, gsw_entropy_from_t, gsw_entropy_ice, gsw_pt_from_entropy

## Examples

```
SA <- c(      34.7118,  34.8915,  35.0256,  34.8472, 34.7366, 34.7324)
entropy <- c(400.3892, 395.4378, 319.8668, 146.7910, 98.6471, 62.7919)
CT <- gsw_CT_from_entropy(SA, entropy)
expect_equal(CT, c(28.809902787278070, 28.439199226786918, 22.786199266954270,
                 10.226197672488652, 6.827196739780282, 4.323602945446461))
```

---

gsw_CT_from_pt    *Conservative Temperature from Potential Temperature*

---

## Description

Conservative Temperature from Potential Temperature

## Usage

```
gsw_CT_from_pt(SA, pt)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| pt | potential temperature (ITS-90) [ degC ] |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

Conservative Temperature [ degC ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_CT_from_pt.html

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
pt <- c(28.7832, 28.4209, 22.7850, 10.2305,  6.8292,  4.3245)
CT <- gsw_CT_from_pt(SA, pt)
expect_equal(CT, c(28.809923015982083, 28.439144260767169, 22.786246608464264,
                   10.226165605435785, 6.827183417643142,  4.323565182322069))
```

`gsw_CT_from_rho` *Conservative Temperature from Density, Absolute Salinity and Pressure*

## Description

Conservative Temperature from Density, Absolute Salinity and Pressure

## Usage

```
gsw_CT_from_rho(rho, SA, p)
```

## Arguments

| | |
|---|---|
| rho | seawater density [ kg/m^3 ] |
| SA | Absolute Salinity [ g/kg ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

A list containing two estimates of Conservative Temperature: `CT` and `CT_multiple`, each in [ degC ].

## References

http://www.teos-10.org/pubs/gsw/html/gsw_CT_from_rho.html

## See Also

Other things related to density: `gsw_CT_maxdensity`, `gsw_SA_from_rho`, `gsw_alpha_on_beta`, `gsw_alpha_wrt_t_exact`, `gsw_alpha_wrt_t_ice`, `gsw_alpha`, `gsw_beta_const_t_exact`, `gsw_beta`, `gsw_pot_rho_t_exact`, `gsw_rho_alpha_beta`, `gsw_rho_first_derivatives_wrt_enthalpy`, `gsw_rho_first_derivat`, `gsw_rho_ice`, `gsw_rho_t_exact`, `gsw_rho`, `gsw_sigma0`, `gsw_sigma1`, `gsw_sigma2`, `gsw_sigma3`, `gsw_sigma4`, `gsw_specvol_alpha_beta`, `gsw_specvol_anom_standard`, `gsw_specvol_ice`, `gsw_specvol_t_exact`, `gsw_specvol`

### Examples

```
rho <- c(1021.8484, 1022.2647, 1024.4207, 1027.7841, 1029.8287, 1031.9916)
SA <- c(   34.7118,   34.8915,   35.0256,   34.8472,   34.7366,   34.7324)
p <- c(         10,        50,       125,       250,       600,      1000)
r <- gsw_CT_from_rho(rho, SA, p)
expect_equal(r$CT, c(28.784377302226968, 28.432402127485858, 22.808745445250068,
                     10.260169334807866, 6.887336649146716, 4.404594162282834))
```

---

## gsw_CT_from_t — *Convert from temperature to conservative temperature*

---

### Description

Convert from temperature to conservative temperature

### Usage

```
gsw_CT_from_t(SA, t, p)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

Conservative Temperature [ degC ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_CT_from_t.html

### Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
t <-  c(28.7856, 28.4329, 22.8103, 10.2600,  6.8863,  4.4036)
p <-  c(     10,      50,     125,     250,     600,    1000)
CT <- gsw_CT_from_t(SA, t, p)
expect_equal(CT, c(28.809919826700281, 28.439227816091140, 22.786176893078498,
                   10.226189266620782, 6.827213633479988, 4.323575748610455))
```

---

## gsw_CT_maxdensity    *Conservative Temperature at Maximum Density*

---

### Description

Conservative Temperature at Maximum Density

### Usage

```
gsw_CT_maxdensity(SA, p)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

Conservative Temperature [ degC ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_CT_maxdensity.html

### See Also

Other things related to density: gsw_CT_from_rho, gsw_SA_from_rho, gsw_alpha_on_beta, gsw_alpha_wrt_t_exact, gsw_alpha_wrt_t_ice, gsw_alpha, gsw_beta_const_t_exact, gsw_beta, gsw_pot_rho_t_exact, gsw_rho_alpha_beta, gsw_rho_first_derivatives_wrt_enthalpy, gsw_rho_first_derivatives, gsw_rho_ice, gsw_rho_t_exact, gsw_rho, gsw_sigma0, gsw_sigma1, gsw_sigma2, gsw_sigma3, gsw_sigma4, gsw_specvol_alpha_beta, gsw_specvol_anom_standard, gsw_specvol_ice, gsw_specvol_t_exact, gsw_specvol

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
p <-  c(     10,      50,     125,     250,     600,    1000)
CT <- gsw_CT_maxdensity(SA, p)
expect_equal(CT, c(-3.731407240089855, -3.861137427731664, -4.060390602245942,
                   -4.306222571955388, -5.089240667106197, -6.028034316992341))
```

---

gsw_CT_second_derivatives

*Second Derivatives of Conservative Temperature*

---

## Description

Second Derivatives of Conservative Temperature

## Usage

```
gsw_CT_second_derivatives(SA, pt)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| pt | potential temperature (ITS-90) [ degC ] |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

A list containing CT_SA_SA [ K/(g/kg)^2 ], the second derivative of Conservative Temperature with respect to Absolute Salinity at constant potential temperature, and CT_SA_pt [ 1/(g/kg) ], the derivative of Conservative Temperature with respect to potential temperature and Absolute Salinity, and CT_pt_pt [ 1/degC ], the second derivative of Conservative Temperature with respect to potential temperature.

## References

http://www.teos-10.org/pubs/gsw/html/gsw_CT_second_derivatives.html

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
pt <- c(28.7832, 28.4209, 22.7850, 10.2305,  6.8292,  4.3245)
r <- gsw_CT_second_derivatives(SA, pt)
expect_equal(r$CT_SA_SA/1e-3, c(-0.060718502077064, -0.062065324400873, -0.084017055354742,
                              -0.148436050120131, -0.171270386500246, -0.189920754900116))
expect_equal(r$CT_SA_pt, c(-0.001197415000869, -0.001198309530139, -0.001226523296082,
                           -0.001335896286481, -0.001380492698572, -0.001417751669135))
expect_equal(r$CT_pt_pt/1e-3, c(0.123012754427146, 0.124662008871271, 0.140829458783443,
                                0.140646803448166, 0.113684095615077, 0.082286843477998))
```

---

gsw_C_from_SP *Electrical Conductivity from Practical Salinity*

---

## Description

Electrical conductivity (in mS/cm) from Practical Salinity. To convert the return value to conductivity ratio, divide by 42.9140 (the value of conductivity at S=35, T68=15, and p=0).

## Usage

```
gsw_C_from_SP(SP, t, p)
```

## Arguments

| | |
|---|---|
| SP | Practical Salinity (PSS-78) [ unitless ] |
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

electrical conductivity [ mS/cm ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_C_from_SP.html

## See Also

Other things related to salinity: gsw_SA_from_SP_Baltic, gsw_SA_from_SP, gsw_SA_from_Sstar, gsw_SP_from_C, gsw_SP_from_SA, gsw_SP_from_SK, gsw_SP_from_SR, gsw_SP_from_Sstar, gsw_SR_from_SP, gsw_Sstar_from_SA, gsw_Sstar_from_SP, gsw_deltaSA_from_SP

Other things related to conductivity: gsw_SP_from_C

## Examples

```
SP <- c(34.5487, 34.7275, 34.8605, 34.6810, 34.5680, 34.5600)
t <- c( 28.7856, 28.4329, 22.8103, 10.2600,  6.8863,  4.4036)
p <- c(      10,      50,     125,     250,     600,    1000)
C <- gsw_C_from_SP(SP, t, p)
expect_equal(C, c(56.412599581571186, 56.316185602699953, 50.670369333973944,
                  38.134518936104350, 35.056577637635257, 32.986550607990118))
```

---

`gsw_deltaSA_from_SP` *Absolute Salinity Anomaly from Practical Salinity*

---

## Description

Absolute Salinity Anomaly from Practical Salinity

## Usage

```
gsw_deltaSA_from_SP(SP, p, longitude, latitude)
```

## Arguments

| | |
|---|---|
| SP | Practical Salinity (PSS-78) [ unitless ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| longitude | longitude in decimal degrees, positive to the east of Greenwich. (This is called `long` in the TEOS-10 Matlab code.) |
| latitude | latitude in decimal degrees, positive to the north of the equator. (This is called `lat` in the TEOS-10 Matlab code.) |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

deltaSA Absolute Salinity Anomaly [ g/kg ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_deltaSA_from_SP.html

## See Also

Other things related to salinity: gsw_C_from_SP, gsw_SA_from_SP_Baltic, gsw_SA_from_SP, gsw_SA_from_Sstar, gsw_SP_from_C, gsw_SP_from_SA, gsw_SP_from_SK, gsw_SP_from_SR, gsw_SP_from_Sstar, gsw_SR_from_SP, gsw_Sstar_from_SA, gsw_Sstar_from_SP

## Examples

```
SP =   c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
p =    c(     10,      50,     125,     250,     600,    1000)
lat =  c(      4,       4,       4,       4,       4,       4)
long = c(    188,     188,     188,     188,     188,     188)
deltaSA = gsw_deltaSA_from_SP(SP,p,long,lat)
expect_equal(deltaSA, c(0.000167203365230, 0.000268836122231, 0.000665803155705,
                        0.002706154619403, 0.005652977406832,  0.009444734661606))
```

---

# gsw_dilution_coefficient_t_exact

*Dilution coefficient*

---

## Description

Dilution coefficient

## Usage

```
gsw_dilution_coefficient_t_exact(SA, t, p)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

dilution coefficient [ (J/kg)(kg/g) ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_dilution_coefficient_t_exact.html

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
t <- c( 28.7856, 28.4329, 22.8103, 10.2600,  6.8863,  4.4036)
p <- c(      10,      50,     125,     250,     600,    1000)
dc <- gsw_dilution_coefficient_t_exact(SA, t, p)
expect_equal(dc, c(79.140034211532040, 79.104983526833820, 77.503312016847389,
                 73.535062653715272, 72.483378545466564, 71.760667498673087))
```

---

gsw_dynamic_enthalpy *Dynamic enthalpy of seawater (75-term equation)*

---

## Description

Dynamic enthalpy of seawater (75-term equation)

## Usage

```
gsw_dynamic_enthalpy(SA, CT, p)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

dynamic enthalpy [ J/kg ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_enthalpy.html

## See Also

Other things related to enthalpy: gsw_CT_from_enthalpy, gsw_enthalpy_CT_exact, gsw_enthalpy_diff, gsw_enthalpy_first_derivatives_CT_exact, gsw_enthalpy_first_derivatives, gsw_enthalpy_ice, gsw_enthalpy_t_exact, gsw_enthalpy, gsw_frazil_properties_potential_poly, gsw_frazil_properties_potent: gsw_pot_enthalpy_from_pt_ice_poly, gsw_pot_enthalpy_from_pt_ice, gsw_pot_enthalpy_ice_freezing_poly, gsw_pot_enthalpy_ice_freezing, gsw_pt_from_pot_enthalpy_ice_poly, gsw_pt_from_pot_enthalpy_ice, gsw_specvol_first_derivatives_wrt_enthalpy, gsw_specvol_first_derivatives

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <-c( 28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
p <- c(      10,      50,     125,     250,     600,    1000)
de <- gsw_dynamic_enthalpy(SA, CT, p)
expect_equal(de/1000, c(0.097864698087770, 0.489161476686235, 1.220512192086506,
                        2.433731199531144, 5.833880057399701, 9.711443860944032))
```

---

# gsw_enthalpy *Specific enthalpy of seawater (75-term equation)*

---

## Description

Specific enthalpy of seawater (75-term equation)

## Usage

```
gsw_enthalpy(SA, CT, p)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

specific enthalpy [ J/kg ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_enthalpy.html

## See Also

Other things related to enthalpy: gsw_CT_from_enthalpy, gsw_dynamic_enthalpy, gsw_enthalpy_CT_exact, gsw_enthalpy_diff, gsw_enthalpy_first_derivatives_CT_exact, gsw_enthalpy_first_derivatives, gsw_enthalpy_ice, gsw_enthalpy_t_exact, gsw_frazil_properties_potential_poly, gsw_frazil_properties_pot gsw_pot_enthalpy_from_pt_ice_poly, gsw_pot_enthalpy_from_pt_ice, gsw_pot_enthalpy_ice_freezing_poly, gsw_pot_enthalpy_ice_freezing, gsw_pt_from_pot_enthalpy_ice_poly, gsw_pt_from_pot_enthalpy_ice, gsw_specvol_first_derivatives_wrt_enthalpy, gsw_specvol_first_derivatives

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <-c( 28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
p <- c(      10,      50,     125,     250,     600,    1000)
e <- gsw_enthalpy(SA, CT, p)
expect_equal(e/1e5, c(1.151031813559086, 1.140146926828028, 0.921800138366058,
                    0.432553713026279, 0.330871609742468, 0.269706841603465))
```

---

gsw_enthalpy_CT_exact *Seawater Specific Enthalpy in terms of Conservative Temperature*

---

## Description

Seawater Specific Enthalpy in terms of Conservative Temperature

## Usage

```
gsw_enthalpy_CT_exact(SA, CT, p)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

specific enthalpy [ J/kg ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_enthalpy_ct_exact.html

## See Also

Other things related to enthalpy: gsw_CT_from_enthalpy, gsw_dynamic_enthalpy, gsw_enthalpy_diff, gsw_enthalpy_first_derivatives_CT_exact, gsw_enthalpy_first_derivatives, gsw_enthalpy_ice, gsw_enthalpy_t_exact, gsw_enthalpy, gsw_frazil_properties_potential_poly, gsw_frazil_properties_potent gsw_pot_enthalpy_from_pt_ice_poly, gsw_pot_enthalpy_from_pt_ice, gsw_pot_enthalpy_ice_freezing_poly, gsw_pot_enthalpy_ice_freezing, gsw_pt_from_pot_enthalpy_ice_poly, gsw_pt_from_pot_enthalpy_ice, gsw_specvol_first_derivatives_wrt_enthalpy, gsw_specvol_first_derivatives

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
p <- c(      10,      50,     125,     250,     600,    1000)
e <- gsw_enthalpy_CT_exact(SA, CT, p)
expect_equal(e/1e5, c(1.151031813321767, 1.140146925586514, 0.921800131787836,
                    0.432553712315790, 0.330871615358722, 0.269706848807403))
```

---

gsw_enthalpy_diff *Specific Enthalpy Difference with Pressure*

---

## Description

Specific enthalpy difference [ J/kg ].

## Usage

```
gsw_enthalpy_diff(SA, CT, p_shallow, p_deep)
```

## Arguments

SA Absolute Salinity [ g/kg ]

CT Conservative Temperature [ degC ]

p_shallow pressure at a shallower depth [ dbar ]

p_deep pressure at a deeper depth [ dbar ]

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

specific enthalpy difference [ J/kg ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_enthalpy_diff.html

## See Also

Other things related to enthalpy: gsw_CT_from_enthalpy, gsw_dynamic_enthalpy, gsw_enthalpy_CT_exact, gsw_enthalpy_first_derivatives_CT_exact, gsw_enthalpy_first_derivatives, gsw_enthalpy_ice, gsw_enthalpy_t_exact, gsw_enthalpy, gsw_frazil_properties_potential_poly, gsw_frazil_properties_potent gsw_pot_enthalpy_from_pt_ice_poly, gsw_pot_enthalpy_from_pt_ice, gsw_pot_enthalpy_ice_freezing_poly, gsw_pot_enthalpy_ice_freezing, gsw_pt_from_pot_enthalpy_ice_poly, gsw_pt_from_pot_enthalpy_ice, gsw_specvol_first_derivatives_wrt_enthalpy, gsw_specvol_first_derivatives

## Examples

```
SA <- c(  34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(  28.7856, 28.4329, 22.8103, 10.2600,  6.8863,  4.4036)
p_shallow <- c(10,      50,     125,     250,     600,    1000)
p_deep <- c(  110,     150,     225,     350,     700,    1100)
ed <- gsw_enthalpy_diff(SA, CT, p_shallow, p_deep)
expect_equal(ed/1e2, c(9.784180644568052, 9.780195056105020, 9.759587700515114,
                       9.727552719534447, 9.708223170174454, 9.687871289079633))
```

## gsw_enthalpy_first_derivatives

*First Derivatives of Enthalpy*

### Description

First Derivatives of Enthalpy

### Usage

```
gsw_enthalpy_first_derivatives(SA, CT, p)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

a list containing `h_SA` [ (J/kg)/(g/kg) ], the derivative of enthalpy wrt Absolute Salinity, and `h_CT` [ (J/kg)/degC ], the derivative of enthalpy wrt Conservative Temperature.

### References

http://www.teos-10.org/pubs/gsw/html/gsw_enthalpy_first_derivatives.html

### See Also

Other things related to enthalpy: gsw_CT_from_enthalpy, gsw_dynamic_enthalpy, gsw_enthalpy_CT_exact, gsw_enthalpy_diff, gsw_enthalpy_first_derivatives_CT_exact, gsw_enthalpy_ice, gsw_enthalpy_t_exact, gsw_enthalpy, gsw_frazil_properties_potential_poly, gsw_frazil_properties_potential, gsw_pot_enthalpy_from_pt_ice_poly, gsw_pot_enthalpy_from_pt_ice, gsw_pot_enthalpy_ice_freezing_poly, gsw_pot_enthalpy_ice_freezing, gsw_pt_from_pot_enthalpy_ice_poly, gsw_pt_from_pot_enthalpy_ice, gsw_specvol_first_derivatives_wrt_enthalpy, gsw_specvol_first_derivatives

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.7856, 28.4329, 22.8103, 10.2600,  6.8863,  4.4036)
p <-  c(     10,      50,     125,     250,     600,    1000)
d <- gsw_enthalpy_first_derivatives(SA, CT, p)
expect_equal(d$h_SA, c(-0.070223912348929, -0.351159768365102, -0.887025065692568,
                     -1.829602387915694, -4.423463748270238, -7.405100077558673))
expect_equal(d$h_CT/1e3, c(3.991899705530481, 3.992025640520101, 3.992210365030743,
                         3.992284150250490, 3.992685389122658, 3.993014168534175))
```

---

`gsw_enthalpy_first_derivatives_CT_exact`

*First Derivatives of Enthalpy wrt CT*

---

## Description

First Derivatives of Enthalpy wrt CT

## Usage

```
gsw_enthalpy_first_derivatives_CT_exact(SA, CT, p)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

a list containing `h_SA` [ (J/kg)/(g/kg) ], the derivative of enthalpy wrt Absolute Salinity, and `h_CT` [ (J/kg)/degC ], the derivative of enthalpy wrt Conservative Temperature.

## Bugs

The HTML documentation suggests that this function returns 3 values, but there are only 2 returned values in the C code used here (and the matlab code on which that is based). Also, the d/dSA check values given the HTML are not reproduced by the present function. This was reported on Mar 18, 2017 as https://github.com/TEOS-10/GSW-Matlab/issues/7. See https://github.com/TEOS-10/GSW-R/issues/34

## References

http://www.teos-10.org/pubs/gsw/html/gsw_enthalpy_first_derivatives_CT_exact.html

## See Also

Other things related to enthalpy: gsw_CT_from_enthalpy, gsw_dynamic_enthalpy, gsw_enthalpy_CT_exact, gsw_enthalpy_diff, gsw_enthalpy_first_derivatives, gsw_enthalpy_ice, gsw_enthalpy_t_exact, gsw_enthalpy, gsw_frazil_properties_potential_poly, gsw_frazil_properties_potential, gsw_pot_enthalpy_from_pt_ice_poly, gsw_pot_enthalpy_from_pt_ice, gsw_pot_enthalpy_ice_freezing_poly, gsw_pot_enthalpy_ice_freezing, gsw_pt_from_pot_enthalpy_ice_poly, gsw_pt_from_pot_enthalpy_ice, gsw_specvol_first_derivatives_wrt_enthalpy, gsw_specvol_first_derivatives

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.7856, 28.4329, 22.8103, 10.2600,  6.8863,  4.4036)
p <-  c(     10,      50,     125,     250,     600,    1000)
d <- gsw_enthalpy_first_derivatives_CT_exact(SA, CT, p)
expect_equal(d$h_SA, c(-0.070224183838619, -0.351159869043798, -0.887036550157504,
                     -1.829626251448858, -4.423522691827955, -7.405211691293971))
expect_equal(d$h_CT/1e3, c(3.991899712269790, 3.992025674159605, 3.992210402650973,
                         3.992283991748418, 3.992685275917238, 3.993014370250710))
```

---

gsw_enthalpy_ice    *Ice Specific Enthalpy*

---

## Description

Specific enthalpy of ice [ J/kg ]. Note that this is a negative quantity.

## Usage

```
gsw_enthalpy_ice(t, p)
```

## Arguments

| | |
|---|---|
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

specific enthalpy [ J/kg ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_enthalpy_ice.html

## See Also

Other things related to enthalpy: gsw_CT_from_enthalpy, gsw_dynamic_enthalpy, gsw_enthalpy_CT_exact, gsw_enthalpy_diff, gsw_enthalpy_first_derivatives_CT_exact, gsw_enthalpy_first_derivatives, gsw_enthalpy_t_exact, gsw_enthalpy, gsw_frazil_properties_potential_poly, gsw_frazil_properties_potent: gsw_pot_enthalpy_from_pt_ice_poly, gsw_pot_enthalpy_from_pt_ice, gsw_pot_enthalpy_ice_freezing_poly, gsw_pot_enthalpy_ice_freezing, gsw_pt_from_pot_enthalpy_ice_poly, gsw_pt_from_pot_enthalpy_ice, gsw_specvol_first_derivatives_wrt_enthalpy, gsw_specvol_first_derivatives

## Examples

```
t <- c(-10.7856, -13.4329, -12.8103, -12.2600, -10.8863, -8.4036)
p <-  c(     10,       50,      125,      250,      600,     1000)
se <- gsw_enthalpy_ice(t, p)
expect_equal(se/1e5, c(-3.554414597446597, -3.603380857687490, -3.583089884253586,
                     -3.558998379233944, -3.494811024956881, -3.402784319238127))
```

---

`gsw_enthalpy_second_derivatives`
*Second Derivatives of Enthalpy*

---

## Description

Second Derivatives of Enthalpy

## Usage

```
gsw_enthalpy_second_derivatives(SA, CT, p)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

A list containing h_SA_SA [ (J/kg)/(g/kg)^2 ], the second derivative of enthalpy with respect to Absolute Salinity, h_SA_CT [ (J/kg)/(K*g/kg) ], the derivative of enthalpy with respect to Absolute Salinity and Conservative Temperature, and h_CT_CT [ (J/kg)/degC^2 ], the second derivative of enthalpy with respect to Conservative Temperature.

### References

http://www.teos-10.org/pubs/gsw/html/gsw_enthalpy_second_derivatives.html

### Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.7856, 28.4329, 22.8103, 10.2600,  6.8863,  4.4036)
p <- c(      10,      50,     125,     250,     600,    1000)
r <- gsw_enthalpy_second_derivatives(SA, CT, p)
expect_equal(r$h_SA_SA, c(0.000080922482023, 0.000404963500641, 0.001059800046742,
                          0.002431088963823, 0.006019611828423, 0.010225411250217))
expect_equal(r$h_SA_CT, c(0.000130004715129, 0.000653614489248, 0.001877220817849,
                          0.005470392103793, 0.014314756132297, 0.025195603327700))
expect_equal(r$h_CT_CT, c(0.000714303909834, 0.003584401249266, 0.009718730753139,
                          0.024064471995224, 0.061547884081343, 0.107493969308119))
```

---

## gsw_enthalpy_second_derivatives_CT_exact

*Second Derivatives of Enthalpy (exact)*

---

### Description

Second Derivatives of Enthalpy (exact)

### Usage

```
gsw_enthalpy_second_derivatives_CT_exact(SA, CT, p)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

A list containing h_SA_SA [ (J/kg)/(g/kg)^2 ], the second derivative of enthalpy with respect to Absolute Salinity, h_SA_CT [ (J/kg)/(K*g/kg) ], the derivative of enthalpy with respect to Absolute Salinity and Conservative Temperature, and h_CT_CT [ (J/kg)/degC^2 ], the second derivative of enthalpy with respect to Conservative Temperature.

### References

http://www.teos-10.org/pubs/gsw/html/gsw_enthalpy_second_derivatives_CT_exact.html

### Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.7856, 28.4329, 22.8103, 10.2600,  6.8863,  4.4036)
p <- c(      10,      50,     125,     250,     600,    1000)
r <- gsw_enthalpy_second_derivatives_CT_exact(SA, CT, p)
expect_equal(r$h_SA_SA, c(0.000082767011576, 0.000414469343141, 0.001089580017293,
                          0.002472193425998, 0.006103171596320, 0.010377465312463))
expect_equal(r$h_SA_CT, c(0.000130320164426, 0.000655016236924, 0.001879127443985,
                          0.005468695168037, 0.014315709000526, 0.025192691262061))
expect_equal(r$h_CT_CT, c(0.000714365642428, 0.003584965089168, 0.009733337653703,
                          0.024044402143825, 0.061449390733344, 0.107333638394904))
```

---

## gsw_enthalpy_t_exact *Seawater Specific Enthalpy in terms of in-situ Temperature*

---

### Description

Seawater Specific Enthalpy in terms of in-situ Temperature

### Usage

```
gsw_enthalpy_t_exact(SA, t, p)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

specific enthalpy [ J/kg ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_enthalpy_t_exact.html

### See Also

Other things related to enthalpy: gsw_CT_from_enthalpy, gsw_dynamic_enthalpy, gsw_enthalpy_CT_exact, gsw_enthalpy_diff, gsw_enthalpy_first_derivatives_CT_exact, gsw_enthalpy_first_derivatives, gsw_enthalpy_ice, gsw_enthalpy, gsw_frazil_properties_potential_poly, gsw_frazil_properties_potential, gsw_pot_enthalpy_from_pt_ice_poly, gsw_pot_enthalpy_from_pt_ice, gsw_pot_enthalpy_ice_freezing_poly, gsw_pot_enthalpy_ice_freezing, gsw_pt_from_pot_enthalpy_ice_poly, gsw_pt_from_pot_enthalpy_ice, gsw_specvol_first_derivatives_wrt_enthalpy, gsw_specvol_first_derivatives

### Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
t <- c( 28.7856, 28.4329, 22.8103, 10.2600,  6.8863,  4.4036)
p <- c(      10,      50,     125,     250,     600,    1000)
e <- gsw_enthalpy_t_exact(SA, t, p)
expect_equal(e/1e5, c(1.151032604783763, 1.140148036012021, 0.921799209310966,
                    0.432553283808897, 0.330872159700175, 0.269705880448018))
```

---

`gsw_entropy_first_derivatives`

*First Derivatives of Entropy*

---

### Description

First Derivatives of Entropy

### Usage

```
gsw_entropy_first_derivatives(SA, CT)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

a list containing `eta_SA` [ (J/(kg*degC) / (g/kg) ], the derivative of entropy wrt Absolute Salinity, and `eta_CT` [ (J/(kg*degC^2) ], the derivative of entropy wrt Conservative Temperature.

### References

http://www.teos-10.org/pubs/gsw/html/gsw_entropy_first_derivatives.html

### See Also

Other things related to entropy: gsw_CT_from_entropy, gsw_entropy_from_pt, gsw_entropy_from_t, gsw_entropy_ice, gsw_pt_from_entropy

### Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
d <- gsw_entropy_first_derivatives(SA, CT)
expect_equal(d$eta_SA, c(-0.263286800711655, -0.263977276574528, -0.255367497912925,
                         -0.238066586439561, -0.234438260606436, -0.232820684341694))
expect_equal(d$eta_CT, c(13.221031210083824, 13.236911191313675, 13.489004628681361,
                         14.086599016583795, 14.257729576432077, 14.386429945649411))
```

---

## gsw_entropy_from_pt    *Specific Entropy ito Absolute Salinity and Potential Temperature*

---

### Description

Calculates specific entropy in terms of Absolute Salinity and Potential Temperature.

### Usage

```
gsw_entropy_from_pt(SA, pt)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| pt | potential temperature (ITS-90) [ degC ] |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

specific entropy [ J/(kg*degC) ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_entropy_from_pt.html

### See Also

Other things related to entropy: gsw_CT_from_entropy, gsw_entropy_first_derivatives, gsw_entropy_from_t, gsw_entropy_ice, gsw_pt_from_entropy

### Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
pt <- c(28.7832, 28.4210, 22.7850, 10.2305,  6.8292,  4.3245)
e <- gsw_entropy_from_pt(SA, pt)
expect_equal(e/1e2, c(4.003894674443156, 3.954383994925507, 3.198674385897981,
                      1.467905482842553, 0.986469100565646, 0.627913567234252))
```

---

## gsw_entropy_from_t    *Specific Entropy i.t.o. Absolute Salinity, Temperature, and Pressure*

---

### Description

Calculates specific entropy in terms of Absolute Salinity, in-situ temperature and pressure.

### Usage

```
gsw_entropy_from_t(SA, t, p)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

specific entropy [ J/(kg*K) ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_entropy_from_t.html

### See Also

Other things related to entropy: gsw_CT_from_entropy, gsw_entropy_first_derivatives, gsw_entropy_from_pt, gsw_entropy_ice, gsw_pt_from_entropy

### Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
t <- c( 28.7856, 28.4329, 22.8103, 10.2600,  6.8863,  4.4036)
p <- c(      10,      50,     125,     250,     600,    1000)
e <- gsw_entropy_from_t(SA, t, p)
expect_equal(e/1e2, c(4.003894252787245, 3.954381784340642, 3.198664981986740,
                      1.467908815899072, 0.986473408657975, 0.627915087346090))
```

---

## gsw_entropy_ice *Entropy of ice*

### Description

Entropy of ice

### Usage

```
gsw_entropy_ice(t, p)
```

### Arguments

| | |
|---|---|
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

entropy [ J/(kg*degC) ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_entropy_ice.html

### See Also

Other things related to entropy: gsw_CT_from_entropy, gsw_entropy_first_derivatives, gsw_entropy_from_pt, gsw_entropy_from_t, gsw_pt_from_entropy

### Examples

```
t <- c(-10.7856, -13.4329, -12.8103, -12.2600,  -10.8863,  -8.4036)
p <- c(      10,       50,      125,      250,      600,      1000)
e <- gsw_entropy_ice(t, p)
expect_equal(e/1e3, c(-1.303663820598987, -1.324090218294577, -1.319426394193644,
                    -1.315402956671801, -1.305426590579231, -1.287021035328113))
```

---

## gsw_entropy_second_derivatives

*Second Derivatives of Entropy*

---

### Description

Second Derivatives of Entropy

### Usage

```
gsw_entropy_second_derivatives(SA, CT)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

A list containing eta_SA_SA [ (J/(K*kg))/(g/kg)^2 ], the second derivative of entropy with respect to Absolute Salinity, eta_SA_CT [ (J/(K*kg))/(K*g/kg) ], the derivative of entropy with respect to Absolute Salinity and Conservative Temperature, and eta_CT_CT [ (J/(K*kg))/K^2 ], the second derivative of entropy with respect to Conservative Temperature.

### Bugs

As of March 27, 2017, the test values listed in "Examples" do not match values provided at the TEOS-10 website listed in "References", but they match with values given by the Matlab code that is provided on the TEOS-10 website. It is expected that the TEOS-10 website will be updated by May 2017. As those updates to the TEOS-10 website become available, the present comment will be revised or deleted.

### References

http://www.teos-10.org/pubs/gsw/html/gsw_entropy_second_derivatives.html

### See Also

Other functions with suspicious test values on the TEOS-10 website: gsw_rho_second_derivatives_wrt_enthalpy, gsw_specvol_second_derivatives_wrt_enthalpy, gsw_t_freezing_first_derivatives_poly

### Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
r <- gsw_entropy_second_derivatives(SA, CT)
expect_equal(r$eta_SA_SA, c(-0.007627718929669, -0.007591969960708, -0.007528186784540,
                            -0.007455177590576, -0.007441108287466, -0.007414368396280))
expect_equal(r$eta_SA_CT, c(-0.001833104216751, -0.001819473824306, -0.001580843823414,
                            -0.000930111408561, -0.000717011215195, -0.000548410546830))
expect_equal(r$eta_CT_CT, c(-0.043665023731109, -0.043781336189326, -0.045506114440888,
                            -0.049708939454018, -0.050938690879443, -0.051875017843472))
```

`gsw_Fdelta` *Ratio of Absolute to Preformed Salinity, minus 1*

## Description

Ratio of Absolute to Preformed Salinity, minus 1

## Usage

```
gsw_Fdelta(p, longitude, latitude)
```

## Arguments

| | |
|---|---|
| `p` | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| `longitude` | longitude in decimal degrees, positive to the east of Greenwich. (This is called `long` in the TEOS-10 Matlab code.) |
| `latitude` | latitude in decimal degrees, positive to the north of the equator. (This is called `lat` in the TEOS-10 Matlab code.) |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

(S/SStar)-1 [ unitless ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_Fdelta.html

## Examples

```
p <- c(         10,    50,   125,   250,   600, 1000)
latitude <- c(   4,     4,     4,     4,     4,    4)
longitude <- c(188,   188,   188,   188,   188,  188)
r <- gsw_Fdelta(p, longitude, latitude)
expect_equal(r/1e-3, c(0.006472309923452, 0.010352848168433, 0.025541937543450,
                      0.104348729347986, 0.218678084205081, 0.365415366571266))
```

## gsw_frazil_properties *Properties of Frazil ice*

### Description

Calculation of Absolute Salinity, Conservative Temperature, and ice mass fraction based on bulk Absolute Salinity, bulk enthalpy, and pressure

### Usage

```
gsw_frazil_properties(SA_bulk, h_bulk, p)
```

### Arguments

| | |
|---|---|
| SA_bulk | Absolute Salinity of a combination of seawater and ice [ g/kg ] |
| h_bulk | enthalpy of a mixture of seawater and ice [ J/kg ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

a list containing `SA_final`, `h_final` and `w_Ih_final`.

### References

http://www.teos-10.org/pubs/gsw/html/gsw_frazil_properties.html

### Examples

```
SA_bulk <- c(  34.7118,   34.8915,   35.0256,   34.8472,   34.7366,   34.7324)
h_bulk <- c( -4.5544e4, -4.6033e4, -4.5830e4, -4.5589e4, -4.4948e4, -4.4027e4)
p <- c(             10,        50,       125,       250,       600,      1000)
r <- gsw_frazil_properties(SA_bulk, h_bulk, p)
expect_equal(r$SA_final, c(39.111030663000442, 39.407625769681573, 39.595789974885108,
                         39.481230045372889, 39.591177095552503, 39.826467709177123))
expect_equal(r$CT_final, c(-2.156311126114311, -2.204672298963783, -2.273689262333450,
                         -2.363714136353600, -2.644541000680772, -2.977651291726651))
expect_equal(r$w_Ih_final, c(0.112480560814322, 0.114600300867556, 0.115421108602301,
                           0.117372990660305, 0.122617649983886, 0.127906590822347))
```

`gsw_frazil_properties_potential`
*Properties of Frazil ice i.t.o. potential enthalpy*

## Description

Calculation of Absolute Salinity, Conservative Temperature, and ice mass fraction based on bulk Absolute Salinity, bulk potential enthalpy, and pressure

## Usage

```
gsw_frazil_properties_potential(SA_bulk, h_pot_bulk, p)
```

## Arguments

| | |
|---|---|
| `SA_bulk` | Absolute Salinity of a combination of seawater and ice [ g/kg ] |
| `h_pot_bulk` | potential enthalpy of a mixture of seawater and ice [ J/kg ] |
| `p` | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

a list containing `SA_final`, `h_final` and `w_Ih_final`.

## References

http://www.teos-10.org/pubs/gsw/html/gsw_frazil_properties_potential.html

## See Also

Other things related to enthalpy: `gsw_CT_from_enthalpy`, `gsw_dynamic_enthalpy`, `gsw_enthalpy_CT_exact`, `gsw_enthalpy_diff`, `gsw_enthalpy_first_derivatives_CT_exact`, `gsw_enthalpy_first_derivatives`, `gsw_enthalpy_ice`, `gsw_enthalpy_t_exact`, `gsw_enthalpy`, `gsw_frazil_properties_potential_poly`, `gsw_pot_enthalpy_from_pt_ice_poly`, `gsw_pot_enthalpy_from_pt_ice`, `gsw_pot_enthalpy_ice_freezing_poly`, `gsw_pot_enthalpy_ice_freezing`, `gsw_pt_from_pot_enthalpy_ice_poly`, `gsw_pt_from_pot_enthalpy_ice`, `gsw_specvol_first_derivatives_wrt_enthalpy`, `gsw_specvol_first_derivatives`

## Examples

```
SA_bulk <- c(     34.7118,   34.8915,   35.0256,   34.8472,   34.7366,   34.7324)
h_pot_bulk <- c(-4.5544e4, -4.6033e4, -4.5830e4, -4.5589e4, -4.4948e4, -4.4027e4)
p <- c(                10,        50,       125,       250,       600,      1000)
r <- gsw_frazil_properties_potential(SA_bulk, h_pot_bulk, p)
expect_equal(r$SA_final, c(39.098258701462051, 39.343217598625756, 39.434254585716296,
                         39.159536295126657, 38.820511558004590, 38.542322667924459))
expect_equal(r$CT_final, c(-2.155553336670014, -2.200844802695826, -2.264077329325076,
                         -2.344567015865174, -2.598559540430464, -2.900814843304696))
expect_equal(r$w_Ih_final, c(0.112190640891586, 0.113150826758543, 0.111797588975174,
                           0.110122251260246, 0.105199838799201, 0.098850365110330))
```

---

# gsw_frazil_properties_potential_poly

*Properties of Frazil ice i.t.o. potential enthalpy (polynomial version)*

---

## Description

Calculation of Absolute Salinity, Conservative Temperature, and ice mass fraction based on bulk Absolute Salinity, bulk potential enthalpy, and pressure

## Usage

```
gsw_frazil_properties_potential_poly(SA_bulk, h_pot_bulk, p)
```

## Arguments

| | |
|---|---|
| SA_bulk | Absolute Salinity of a combination of seawater and ice [ g/kg ] |
| h_pot_bulk | potential enthalpy of a mixture of seawater and ice [ J/kg ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

a list containing `SA_final`, `h_final` and `w_Ih_final`.

## References

http://www.teos-10.org/pubs/gsw/html/gsw_frazil_properties_potential_poly.html

### See Also

Other things related to enthalpy: gsw_CT_from_enthalpy, gsw_dynamic_enthalpy, gsw_enthalpy_CT_exact, gsw_enthalpy_diff, gsw_enthalpy_first_derivatives_CT_exact, gsw_enthalpy_first_derivatives, gsw_enthalpy_ice, gsw_enthalpy_t_exact, gsw_enthalpy, gsw_frazil_properties_potential, gsw_pot_enthalpy_from_pt_ice_poly, gsw_pot_enthalpy_from_pt_ice, gsw_pot_enthalpy_ice_freezing_poly, gsw_pot_enthalpy_ice_freezing, gsw_pt_from_pot_enthalpy_ice_poly, gsw_pt_from_pot_enthalpy_ice, gsw_specvol_first_derivatives_wrt_enthalpy, gsw_specvol_first_derivatives

### Examples

```
SA_bulk <- c(     34.7118,    34.8915,    35.0256,    34.8472,    34.7366,    34.7324)
h_pot_bulk <- c(-4.5544e4, -4.6033e4, -4.5830e4, -4.5589e4, -4.4948e4, -4.4027e4)
p <- c(               10,         50,        125,        250,        600,       1000)
r <- gsw_frazil_properties_potential_poly(SA_bulk, h_pot_bulk, p)
expect_equal(r$SA_final, c(39.098264696022831, 39.343217436835218, 39.434244243586633,
                         39.159511498029801, 38.820458704205542, 38.542256756176229))
expect_equal(r$CT_final, c(-2.155537691991377, -2.200841508940901, -2.264094318382661,
                         -2.344613208230164, -2.598663953454472, -2.900948531145453))
expect_equal(r$w_Ih_final, c(0.112190777010854, 0.113150823111566, 0.111797356032850,
                           0.110121687760246, 0.105198620534670, 0.098848824039493))
```

---

## gsw_frazil_ratios_adiabatic

*Ratios of SA, CT and p changes when Frazil Ice Forms*

---

### Description

Ratios of changes in SA, CT and p that occur when frazil ice forms due to changes in pressure upon the mixture of seawater and ice.

### Usage

```
gsw_frazil_ratios_adiabatic(SA, p, w_Ih)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| w_Ih | initial mass fraction (ice) / (water + ice) |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

a list containing dSA_dCT_frazil, dSA_dP_frazil and dCT_dP_frazil.

### References

http://www.teos-10.org/pubs/gsw/html/gsw_frazil_ratios_adiabatic.html

### Examples

```
SA <- c(  34.7118,   34.8915,   35.0256,   34.8472,   34.7366,   34.7324)
p <- c(        10,        50,       125,       250,       600,      1000)
w_Ih <- c(    0.9,      0.84,       0.4,      0.25,      0.05,      0.01)
r <- gsw_frazil_ratios_adiabatic(SA, p, w_Ih)
expect_equal(r$dSA_dCT_frazil, c(3.035152370800401, 1.932548405396193, 0.613212115809003,
                               0.516103092738565, 0.436656742034200, 0.425827266533876))
expect_equal(r$dSA_dP_frazil/1e-6, c(-0.197406834470366, -0.133213926580032, -0.045580136143659,
                             -0.038806356507548, -0.033541272953744, -0.033350141194082))
expect_equal(r$dCT_dP_frazil/1e-7, c(-0.650401727338347, -0.689317412221414, -0.743301297684333,
                             -0.751910946738026, -0.768138213038669, -0.783184728059898))
```

---

gsw_frazil_ratios_adiabatic_poly

*Ratios of SA, CT and p changes when Frazil Ice Forms (polynomial form)*

---

### Description

Ratios of changes in SA, CT and p that occur when frazil ice forms due to changes in pressure upon the mixture of seawater and ice.

### Usage

```
gsw_frazil_ratios_adiabatic_poly(SA, p, w_Ih)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| w_Ih | initial mass fraction (ice) / (water + ice) |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

a list containing dSA_dCT_frazil, dSA_dP_frazil and dCT_dP_frazil.

### References

http://www.teos-10.org/pubs/gsw/html/gsw_frazil_ratios_adiabatic_poly.html

### Examples

```
SA <- c(  34.7118,   34.8915,   35.0256,   34.8472,   34.7366,   34.7324)
p <- c(        10,        50,       125,       250,       600,      1000)
w_Ih <- c(    0.9,      0.84,       0.4,      0.25,      0.05,      0.01)
r <- gsw_frazil_ratios_adiabatic_poly(SA, p, w_Ih)
expect_equal(r$dSA_dCT_frazil, c(3.035308957896530, 1.932631198810934, 0.613220785586734,
                                 0.516106221687200, 0.436657158542033, 0.425827675768018))
expect_equal(r$dSA_dP_frazil/1e-6, c(-0.197512213108610, -0.133280971893621, -0.045599951957139,
                               -0.038820466574251, -0.033548047632788, -0.033352365425407))
expect_equal(r$dCT_dP_frazil/1e-7, c(-0.650715350062703, -0.689634794137768, -0.743613932027895,
                               -0.752179782823459, -0.768292629045686, -0.783236208526200))
```

---

## gsw_geo_strf_dyn_height

*Geostrophic Dynamic Height Anomaly*

---

### Description

This calculates a geopotential anomaly, called either the dynamic height anomaly or the geostrophic streamfunction in the TEOS-10 document listed as [1] below; users should read that and the references therein for more details on the definition and its calculation here.

To get the column-integrated value in meters, take the first value of the returned vector and divide by $9.7963 m/s^2$. Note that this yields an integral with the top measured pressure (not zero) as an upper limit.

### Usage

```
gsw_geo_strf_dyn_height(SA, CT, p, p_ref = 0)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| p_ref | reference pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

Because of the scheme used in the underlying C code, the pressures must be in order, and must not have any repeats. Also, there must be at least 4 pressure values. Violating any of these three restrictions yields an error.

If `p_ref` exceeds the largest `p` value, a vector of zeros is returned, in accordance with the underlying C code.

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

A vector containing geopotential anomaly in $m^2/s^2$ for each level. For more on the units, see [2].

### References


1. http://www.teos-10.org/pubs/gsw/html/gsw_geo_strf_dyn_height.html
2. Talley et al., 2011. Descriptive Physical Oceanography, 6th edition, Elsevier.


### Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
p <- c(      10,      50,     125,     250,     600,    1000)
p_ref <- 1000
dh <- gsw_geo_strf_dyn_height(SA, CT, p, p_ref)
expect_equal(dh, c(17.039204557769487, 14.665853784722286, 10.912861136923812,
                  7.567928838774945, 3.393524055565328, 0))
```

---

## gsw_geo_strf_dyn_height_pc

*Geostrophic Dynamic Height Anomaly (Piecewise-Constant Profile)*

---

### Description

Geostrophic Dynamic Height Anomaly (Piecewise-Constant Profile)

### Usage

```
gsw_geo_strf_dyn_height_pc(SA, CT, delta_p)
```

## Arguments

- `SA` Absolute Salinity [ g/kg ]
- `CT` Conservative Temperature [ degC ]
- `delta_p` difference in sea pressure between the deep and shallow limits of layers within which `SA` and `CT` are assumed to be constant. Note that `delta_p` must be positive.

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

A list containing `dyn_height`, the dynamic height anomaly [ m^2/s^2 ], and `p_mid` [ dbar ], the pressures at the layer centres. Note that the dynamic height anomaly unit, also known as a "dynamic meter", corresponds to approximately 1.02 metres of sealevel height (see e.g. Talley et al., 2011. Descriptive Physical Oceanography, 6th edition. Elsevier).

## References

http://www.teos-10.org/pubs/gsw/html/gsw_geo_strf_dyn_height.html

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
delta_p <- c(10,      40,      75,     125,     350,     400)
r <- gsw_geo_strf_dyn_height_pc(SA, CT, delta_p)
expect_equal(r$dyn_height, c(-0.300346215853487, -1.755165998114308, -4.423531083131365,
                             -6.816659136254657, -9.453175257818430, -12.721009624991439))
expect_equal(r$p_mid/1e2, c(0.050000000000000, 0.300000000000000, 0.875000000000000,
                            1.875000000000000, 4.250000000000000, 8.000000000000000))
```

---

# gsw_gibbs *Gibbs Energy of Seawater, and its Derivatives*

---

## Description

Gibbs Energy of Seawater, and its Derivatives

## Usage

```
gsw_gibbs(ns, nt, np, SA, t, p = 0)
```

### Arguments

| | |
|---|---|
| ns | An integer, the order of the SA derivative. Must be 0, 1, or 2. |
| nt | An integer, the order of the t derivative. Must be 0, 1, or 2. |
| np | An integer, the order of the p derivative. Must be 0, 1, or 2. |
| SA | Absolute Salinity [ g/kg ] |
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

Gibbs energy [ J/kg ] if ns=nt=np=0. Derivative of energy with respect to SA [ J/kg/(g/kg)^ns ] if ns is nonzero and nt=np=0, etc. Note that derivatives with respect to pressure are in units with Pa, not dbar.

### Caution

The TEOS-10 webpage for gsw_gibbs does not provide test values, so the present R version should be considered untested.

### References

http://www.teos-10.org/pubs/gsw/html/gsw_gibbs.html

### Examples

```
library(gsw)
p <- seq(0, 100, 1)
SA <- rep(35, length(p))
t <- rep(-5, length(p))
## Check the derivative wrt pressure. Note the unit change
E <- gsw_gibbs(0, 0, 0, SA, t, p)
# Estimate derivative from linear fit (try plotting: it is very linear)
m <- lm(E ~ p)
print(summary(m))
plot(p, E)
abline(m)
dEdp1 <- coef(m)[2]
# Calculate derivative ... note we multiply by 1e4 to get from 1/Pa to 1/dbar
dEdp2 <- 1e4 * gsw_gibbs(0, 0, 1, SA[1], t[1], p[1])
## Ratio
dEdp1 / dEdp2
```

`gsw_gibbs_ice` *Gibbs Energy of Ice, and its Derivatives*

## Description

Gibbs Energy of Ice, and its Derivatives

## Usage

```
gsw_gibbs_ice(nt, np, t, p = 0)
```

## Arguments

| | |
|---|---|
| `nt` | An integer, the order of the `t` derivative. Must be 0, 1, or 2. |
| `np` | An integer, the order of the `p` derivative. Must be 0, 1, or 2. |
| `t` | in-situ temperature (ITS-90) [ degC ] |
| `p` | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

Gibbs energy [ J/kg ] if `ns=nt=np=0`. Derivative of energy with respect to `t` [ J/kg/(degC)^nt ] if `nt` is nonzero, etc. Note that derivatives with respect to pressure are in units with Pa, not dbar.

## Caution

The TEOS-10 webpage for `gsw_gibbs_ice` does not provide test values, so the present R version should be considered untested.

## References

http://www.teos-10.org/pubs/gsw/html/gsw_gibbs_ice.html

### Examples

```
library(gsw)
p <- seq(0, 100, 1)
t <- rep(-5, length(p))
## Check the derivative wrt pressure. Note the unit change
E <- gsw_gibbs_ice(0, 0, t, p)
# Estimate derivative from linear fit (try plotting: it is very linear)
m <- lm(E ~ p)
print(summary(m))
plot(p, E)
abline(m)
dEdp1 <- coef(m)[2]
# Calculate derivative ... note we multiply by 1e4 to get from 1/Pa to 1/dbar
dEdp2 <- 1e4 * gsw_gibbs_ice(0, 1, t[1], p[1])
## Ratio
dEdp1 / dEdp2
```

---

## gsw_grav *Gravitational Acceleration*

---

### Description

Gravitational Acceleration

### Usage

```
gsw_grav(latitude, p = 0)
```

### Arguments

| | |
|---|---|
| `latitude` | latitude in decimal degrees, positive to the north of the equator. (This is called `lat` in the TEOS-10 Matlab code.) |
| `p` | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

gravitational acceleration [ m/s^2 ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_grav.html

### Examples

```
lat <- c(-90, -60, -30, 0)
grav <- gsw_grav(lat)
expect_equal(grav, c(9.832186205884799, 9.819178859991149,
                     9.793249257048750, 9.780327000000000))
```

---

gsw_Helmholtz_energy_ice

## *Helmholtz Energy of Ice*

---

### Description

Helmholtz Energy of Ice

### Usage

```
gsw_Helmholtz_energy_ice(t, p)
```

### Arguments

| | |
|---|---|
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

Helmholtz energy if ice [ J/kg ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_Helmholtz_energy_ice.html

## Examples

```
t <- c(-10.7856, -13.4329, -12.8103, -12.2600,  -10.8863,  -8.4036)
p <- c(      10,       50,      125,      250,       600,     1000)
e <- gsw_Helmholtz_energy_ice(t, p)
expect_equal(e/1e4, c(-1.362572315008330, -1.710375005915343, -1.628083272702224,
                      -1.555573047498573, -1.375469831393882, -1.053585607014677))
```

---

gsw_ice_fraction_to_freeze_seawater

*Ice Fraction to Cool Seawater to Freezing*

---

## Description

Ice Fraction to Cool Seawater to Freezing

## Usage

```
gsw_ice_fraction_to_freeze_seawater(SA, CT, p, t_Ih)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| t_Ih | initial temperature of ice [ degC ] |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

a list containing SA_freeze, CT_freeze and w_Ih.

## References

http://www.teos-10.org/pubs/gsw/html/gsw_ice_fraction_to_freeze_seawater.html

## Examples

```
SA <- c(   34.7118,  34.8915,  35.0256,  34.8472,  34.7366, 34.7324)
CT <- c(   28.7856,  28.4329,  22.8103,  10.2600,   6.8863,  4.4036)
p <- c(         10,       50,      125,      250,      600,    1000)
t_Ih <- c(-10.7856, -13.4329, -12.8103, -12.2600, -10.8863, -8.4036)
r <- gsw_ice_fraction_to_freeze_seawater(SA, CT, p, t_Ih)
expect_equal(r$SA_freeze, c(25.823952352620722, 26.120495895535438, 27.460572941868072,
                          30.629978769577168, 31.458222332943784, 32.121170316796444))
expect_equal(r$CT_freeze, c(-1.389936216242376, -1.437013334134283, -1.569815847128818,
                          -1.846419165657020, -2.166786673735941, -2.522730879078756))
expect_equal(r$w_Ih, c(0.256046867272203, 0.251379393389925, 0.215985652155336,
                     0.121020375537284, 0.094378196687535, 0.075181377710828))
```

---

`gsw_internal_energy` *Specific Internal Energy of Seawater (75-term equation)*

---

## Description

Specific Internal Energy of Seawater (75-term equation)

## Usage

```
gsw_internal_energy(SA, CT, p)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

specific internal energy [ J/kg ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_internal_energy.html

### Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.7856, 28.4329, 22.8103, 10.2600,  6.8863,  4.4036)
p <-  c(     10,      50,     125,     250,     600,    1000)
e <- gsw_internal_energy(SA, CT, p)
expect_equal(e/1e5, c(1.148091576956162, 1.134013145527675, 0.909571141498779,
                      0.408593072177020, 0.273985276460357, 0.175019409258405))
```

---

gsw_internal_energy_ice *Specific Internal Energy of Ice (75-term equation)*

---

### Description

Specific Internal Energy of Ice (75-term equation)

### Usage

```
gsw_internal_energy_ice(t, p)
```

### Arguments

| | |
|---|---|
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

specific internal energy [ J/kg ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_internal_energy_ice.html

### Examples

```
t_Ih <- c(-10.7856, -13.4329, -12.8103, -12.2600,  -10.8863,  -8.4036)
p <-  c(     10,      50,     125,     250,     600,    1000)
e <- gsw_internal_energy_ice(t_Ih, p)
expect_equal(e/1e5, c(-3.556606992432442, -3.609926216929878, -3.597799043634774,
                      -3.587312078410920, -3.561207060376329, -3.512700418975375))
```

`gsw_IPV_vs_fNsquared_ratio`

*Ratio of vert. gradient of pot. density to vert grad of locally-referenced pot density*

## Description

Note that the C library had to be patched to get this working; a new version of the library will address the bug directly.

## Usage

```
gsw_IPV_vs_fNsquared_ratio(SA, CT, p, p_ref = 0)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| p_ref | reference pressure [ dbar ] |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

list containing IPV_vs_fNsquared_ratio [ unitless ] and mid-point pressure p_mid [ dbar ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_IPV_vs_fNsquared_ratio.html

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
p <-  c(     10,      50,     125,     250,     600,    1000)
p_ref <- 0
r <- gsw_IPV_vs_fNsquared_ratio(SA, CT, p, p_ref)
expect_equal(r$IPV_vs_fNsquared_ratio, c(0.999742244888022, 0.996939883468178, 0.986141997098021,
                                          0.931595598713477, 0.861224354872028))
expect_equal(r$p_mid, c(30, 87.5, 187.5, 425, 800))
```

gsw_kappa *Isentropic Compressibility of Seawater (75-term equation)*

## Description

Isentropic Compressibility of Seawater (75-term equation)

## Usage

```
gsw_kappa(SA, CT, p)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

isentropic compressibility [ 1/Pa ] (not 1/dbar)

## References

http://www.teos-10.org/pubs/gsw/html/gsw_kappa.html

## See Also

Other things related to compressibility: gsw_kappa_const_t_ice, gsw_kappa_ice, gsw_kappa_t_exact

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <-c( 28.7856, 28.4329, 22.8103, 10.2600,  6.8863,  4.4036)
p <- c(      10,      50,     125,     250,     600,    1000)
kappa <- gsw_kappa(SA, CT, p)
expect_equal(kappa*1e9, c(0.411343648791300, 0.411105416128094, 0.416566236026610,
                          0.435588650838751, 0.438782500588955, 0.439842289994702))
```

## gsw_kappa_const_t_ice *Isothermal Compressibility of Ice*

### Description

Calculate isothermal compressibility of ice, in 1/Pa.

### Usage

```
gsw_kappa_const_t_ice(t, p)
```

### Arguments

| | |
|---|---|
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

isothermal compressibility of ice [ 1/Pa ] (not 1/dbar)

### References

http://www.teos-10.org/pubs/gsw/html/gsw_kappa_const_t_ice.html

### See Also

Other things related to compressibility: gsw_kappa_ice, gsw_kappa_t_exact, gsw_kappa

### Examples

```
t <- c(-10.7856, -13.4329, -12.8103, -12.2600,  -10.8863,  -8.4036)
p <- c(      10,       50,      125,      250,       600,     1000)
kappa <- gsw_kappa_const_t_ice(t, p)
expect_equal(kappa*1e9, c(0.115874753261484, 0.115384948953145, 0.115442212717850,
                          0.115452884634531, 0.115454824232421, 0.115619994536961))
```

## gsw_kappa_ice *Isentropic Compressibility of Ice*

### Description

Calculate isentropic compressibility of ice, in 1/Pa.

### Usage

```
gsw_kappa_ice(t, p)
```

### Arguments

| | |
|---|---|
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

isentropic compressibility of ice [ 1/Pa ] (not 1/dbar)

### References

http://www.teos-10.org/pubs/gsw/html/gsw_kappa_ice.html

### See Also

Other things related to compressibility: gsw_kappa_const_t_ice, gsw_kappa_t_exact, gsw_kappa

### Examples

```
t <- c(-10.7856, -13.4329, -12.8103, -12.2600,  -10.8863,  -8.4036)
p <- c(      10,       50,      125,      250,       600,     1000)
kappa <- gsw_kappa_ice(t, p)
expect_equal(kappa*1e9, c(0.112495239053936, 0.112070687842183, 0.112119091047584,
                          0.112126504739297, 0.112123513812840, 0.112262589530974))
```

gsw_kappa_t_exact *Isentropic compressibility of seawater (exact)*

## Description

Isentropic compressibility of seawater (exact)

## Usage

```
gsw_kappa_t_exact(SA, t, p)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

isentropic compressibility [ 1/Pa ] (not 1/dbar)

## References

http://www.teos-10.org/pubs/gsw/html/gsw_kappa_t_exact.html

## See Also

Other things related to compressibility: gsw_kappa_const_t_ice, gsw_kappa_ice, gsw_kappa

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <-c( 28.7856, 28.4329, 22.8103, 10.2600,  6.8863,  4.4036)
p <- c(      10,      50,     125,     250,     600,    1000)
kappa <- gsw_kappa(SA, CT, p)
expect_equal(kappa*1e9, c(0.411343648791300, 0.411105416128094, 0.416566236026610,
                          0.435588650838751, 0.438782500588955, 0.439842289994702))
```

gsw_latentheat_evap_CT

*Latent heat of evaporation*

## Description

Latent heat of evaporation

## Usage

```
gsw_latentheat_evap_CT(SA, CT)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

latent heat of evaporation [ J/kg ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_latentheat_evap_CT.html

## See Also

Other things related to latent heat: gsw_latentheat_evap_t, gsw_latentheat_melting

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.7856, 28.4329, 22.8103, 10.2600,  6.8863,  4.4036)
lh  <- gsw_latentheat_evap_CT(SA, CT)
expect_equal(lh/1e6, c(2.429947107462561, 2.430774073049213, 2.444220372158452,
                     2.474127109232524, 2.482151446148560, 2.488052297193594))
```

gsw_latentheat_evap_t *Latent heat of evaporation*

## Description

Latent heat of evaporation

## Usage

```
gsw_latentheat_evap_t(SA, t)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| t | in-situ temperature (ITS-90) [ degC ] |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

latent heat of evaporation [ J/kg ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_latentheat_evap_t.html

## See Also

Other things related to latent heat: gsw_latentheat_evap_CT, gsw_latentheat_melting

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
t <- c( 28.7856, 28.4329, 22.8103, 10.2600,  6.8863,  4.4036)
lh = gsw_latentheat_evap_t(SA, t)
expect_equal(lh/1e6, c(2.429882982734836, 2.430730236218543, 2.444217294049004,
                       2.474137411322517, 2.482156276375029, 2.488054617630297))
```

gsw_latentheat_melting

*Latent Heat of Melting*

## Description

Latent Heat of Melting

## Usage

```
gsw_latentheat_melting(SA, p)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

latent heat of freezing [ J/kg ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_latentheat_melting.html

## See Also

Other things related to latent heat: gsw_latentheat_evap_CT, gsw_latentheat_evap_t

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
p <- c(      10,      50,     125,     250,     600,    1000)
lh <- gsw_latentheat_melting(SA, p)
expect_equal(lh/1e5, c(3.299496680271213, 3.298613352397986, 3.297125622834541,
                       3.294973895330757, 3.288480445559747, 3.280715862416388))
```

gsw_melting_ice_equilibrium_SA_CT_ratio
*Calculate d(SA)/d(CT) for Ice Melting in near-freezing Seawater*

## Description

Calculate d(SA)/d(CT) for Ice Melting in near-freezing Seawater

## Usage

```
gsw_melting_ice_equilibrium_SA_CT_ratio(SA, p)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

ratio of change in SA to change in CT [ g/kg/degC ].

## References

http://www.teos-10.org/pubs/gsw/html/gsw_melting_ice_equilibrium_SA_CT_ratio.html

## Examples

```
SA <- c(   34.7118,  34.8915,  35.0256,  34.8472,  34.7366, 34.7324)
p <- c(         10,       50,      125,      250,      600,    1000)
r <- gsw_melting_ice_equilibrium_SA_CT_ratio(SA, p)
expect_equal(r, c(0.420209509196985, 0.422511693121631, 0.424345503216433,
                0.422475836091426, 0.422023427778221, 0.423037622331042))
```

gsw_melting_ice_equilibrium_SA_CT_ratio_poly

*Calculate d(SA)/d(CT) for Ice Melting in near-freezing Seawater (Polynomial version)*

## Description

Calculate d(SA)/d(CT) for Ice Melting in near-freezing Seawater (Polynomial version)

## Usage

```
gsw_melting_ice_equilibrium_SA_CT_ratio_poly(SA, p)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

ratio of change in SA to change in CT [ g/kg/degC ].

## References

http://www.teos-10.org/pubs/gsw/html/gsw_melting_ice_equilibrium_SA_CT_ratio_poly.html

## Examples

```
SA <- c(   34.7118,  34.8915,  35.0256,  34.8472,  34.7366, 34.7324)
p <- c(         10,       50,      125,      250,      600,    1000)
r <- gsw_melting_ice_equilibrium_SA_CT_ratio_poly(SA, p)
expect_equal(r, c(0.420209444587263, 0.422511664682796, 0.424345538275708,
                0.422475965003649, 0.422023755182266, 0.423038080717229))
```

gsw_melting_ice_into_seawater
*Calculate properties related to ice melting in seawater*

## Description

Calculate properties related to ice melting in seawater

## Usage

```
gsw_melting_ice_into_seawater(SA, CT, p, w_Ih, t_Ih)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| w_Ih | initial mass fraction (ice) / (water + ice) |
| t_Ih | initial temperature of ice [ degC ] |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

a list containing `SA_final`, `CT_final` and `w_Ih_final`.

## References

http://www.teos-10.org/pubs/gsw/html/gsw_melting_ice_into_seawater.html

## Examples

```
SA <- c(  34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(   4.7856,  2.4329,  1.8103,  1.2600,  0.6886,  0.4403)
p <- c(        10,      50,     125,     250,     600,    1000)
w_Ih <- c( 0.0560, 0.02513, 0.02159, 0.01210, 0.00943, 0.00751)
t_Ih <- c(-4.7856, -4.4329, -3.8103, -4.2600, -3.8863, -3.4036)
r <- gsw_melting_ice_into_seawater(SA, CT, p, w_Ih, t_Ih)
expect_equal(r$SA_final, c(32.767939199999994, 34.014676604999998, 34.269397295999994,
                           34.425548880000001, 34.409033862000001, 34.471559675999998))
```

```
expect_equal(r$CT_final, c(-0.298448911022612, 0.215263001418312, -0.074341719211557,
                           0.207796293045473, -0.123785388299875, -0.202531182809225))
expect_equal(r$w_Ih_final, rep(0, 6))
```

---

gsw_melting_ice_SA_CT_ratio
*Calculate d(SA)/d(CT) for Ice Melting in Seawater*

---

## Description

Calculate d(SA)/d(CT) for Ice Melting in Seawater

## Usage

```
gsw_melting_ice_SA_CT_ratio(SA, CT, p, t_Ih)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| t_Ih | initial temperature of ice [ degC ] |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

ratio of change in SA to change in CT [ g/kg/degC ].

## References

http://www.teos-10.org/pubs/gsw/html/gsw_melting_ice_SA_CT_ratio.html

## Examples

```
SA <- c(   34.7118,   34.8915,   35.0256,   34.8472,   34.7366, 34.7324)
CT <- c(    3.7856,    3.4329,    2.8103,    1.2600,    0.6886,  0.4403)
p <- c(         10,        50,       125,       250,       600,    1000)
t_Ih <- c(-10.7856, -13.4329, -12.8103, -12.2600, -10.8863, -8.4036)
r <- gsw_melting_ice_SA_CT_ratio(SA, CT, p, t_Ih)
expect_equal(r, c(0.373840909022490, 0.371878514972099, 0.377104664622191,
                0.382777696796156, 0.387133845152000, 0.393947316026914))
```

---

gsw_melting_ice_SA_CT_ratio_poly
*Calculate d(SA)/d(CT) for Ice Melting in Seawater (Polynomial version)*

---

## Description

Calculate d(SA)/d(CT) for Ice Melting in Seawater (Polynomial version)

## Usage

```
gsw_melting_ice_SA_CT_ratio_poly(SA, CT, p, t_Ih)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| t_Ih | initial temperature of ice [ degC ] |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

ratio of change in SA to change in CT [ g/kg/degC ].

## References

http://www.teos-10.org/pubs/gsw/html/gsw_melting_ice_SA_CT_ratio_poly.html

## Examples

```
SA <- c(   34.7118,   34.8915,   35.0256,   34.8472,   34.7366, 34.7324)
CT <- c(    3.7856,    3.4329,    2.8103,    1.2600,    0.6886,  0.4403)
p <- c(         10,        50,       125,       250,       600,    1000)
t_Ih <- c(-10.7856, -13.4329, -12.8103, -12.2600, -10.8863, -8.4036)
r <- gsw_melting_ice_SA_CT_ratio_poly(SA, CT, p, t_Ih)
expect_equal(r, c(0.373840908629278, 0.371878512745054, 0.377104658031030,
                0.382777681212224, 0.387133812279563, 0.393947267481204))
```

gsw_melting_seaice_into_seawater
*Calculate properties related to seaice melting in seawater*

## Description

Calculate properties related to seaice melting in seawater

## Usage

```
gsw_melting_seaice_into_seawater(SA, CT, p, w_seaice, SA_seaice, t_seaice)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| w_seaice | mass fraction (seaice) / (water + seaice) |
| SA_seaice | Absolute Salinity of seaice |
| t_seaice | temperature of seaice |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

a list containing `SA_final` and `CT_final`.

## References

http://www.teos-10.org/pubs/gsw/html/gsw_melting_seaice_into_seawater.html

## Examples

```
SA <- c(      34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(       4.7856,  2.4329,  1.8103,  1.2600,  0.6886,  0.4403)
p <- c(            10,      50,     125,     250,     600,    1000)
w_seaice <- c( 0.0560, 0.02513, 0.02159, 0.01210, 0.00943, 0.00751)
SA_seaice <- c(     5,     4.8,     3.5,     2.5,       1,     0.4)
t_seaice <- c(-4.7856, -4.4329, -3.8103, -4.2600, -3.8863, -3.4036)
r <- gsw_melting_seaice_into_seawater(SA, CT, p, w_seaice, SA_seaice, t_seaice)
```

```
expect_equal(r$SA_final, c(33.047939199999995, 34.135300604999998, 34.344962295999999,
                           34.455798880000003, 34.418463862000003, 34.474563675999995))
expect_equal(r$CT_final, c(-0.018822367305381, 0.345095540241769, 0.020418581143151,
                           0.242672380976922, -0.111078380121959, -0.197363471215418))
```

---

gsw_Nsquared *Calculate Brunt Vaisala Frequency squared*

---

### Description

The result is computed based on first-differencing a computed density with respect pressure, and this can yield noisy results with CTD data that have not been smoothed and decimated. It also yields infinite values, for repeated adjacent pressure (e.g. this occurs twice with the `ctd` dataset provided in the **oce** package).

### Usage

```
gsw_Nsquared(SA, CT, p, latitude = 0)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| latitude | latitude in decimal degrees, positive to the north of the equator. (This is called `lat` in the TEOS-10 Matlab code.) |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

list containing N2 [ 1/s^ ] and mid-point pressure p_mid [ dbar ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_Nsquared.html

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
p <- c(      10,      50,     125,     250,     600,    1000)
latitude <- 4
r <- gsw_Nsquared(SA, CT, p, latitude=4)
expect_equal(r$N2*1e3, c(0.060843209693499, 0.235723066151305, 0.216599928330380,
                       0.012941204313372, 0.008434782795209))
expect_equal(r$p_mid, c(30, 87.5, 187.5, 425, 800))
```

---

gsw_pot_enthalpy_from_pt_ice

*Potential Enthalpy of Ice*

---

## Description

Potential Enthalpy of Ice

## Usage

```
gsw_pot_enthalpy_from_pt_ice(pt0_ice)
```

## Arguments

| | |
|---|---|
| pt0_ice | potential temperature of ice (ITS-90) [ degC ] |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

potential enthalpy [ J/kg ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_pot_enthalpy_from_pt_ice.html

## See Also

Other things related to enthalpy: gsw_CT_from_enthalpy, gsw_dynamic_enthalpy, gsw_enthalpy_CT_exact, gsw_enthalpy_diff, gsw_enthalpy_first_derivatives_CT_exact, gsw_enthalpy_first_derivatives, gsw_enthalpy_ice, gsw_enthalpy_t_exact, gsw_enthalpy, gsw_frazil_properties_potential_poly, gsw_frazil_properties_potential, gsw_pot_enthalpy_from_pt_ice_poly, gsw_pot_enthalpy_ice_freezing_poly, gsw_pot_enthalpy_ice_freezing, gsw_pt_from_pot_enthalpy_ice_poly, gsw_pt_from_pot_enthalpy_ice, gsw_specvol_first_derivatives_wrt_enthalpy, gsw_specvol_first_derivatives

## Examples

```
pt0_ice <- c(-10.7856, -13.4329, -12.8103, -12.2600,  -10.8863,  -8.4036)
e <- gsw_pot_enthalpy_from_pt_ice(pt0_ice)
expect_equal(e/1e5, c(-3.555459449611868, -3.608607069998877, -3.596153890859193,
                      -3.585123178806596, -3.557490528226009, -3.507198313847837))
```

---

gsw_pot_enthalpy_from_pt_ice_poly *Potential Enthalpy of Ice (Polynomial version)*

---

## Description

Potential Enthalpy of Ice (Polynomial version)

## Usage

```
gsw_pot_enthalpy_from_pt_ice_poly(pt0_ice)
```

## Arguments

| | |
|---|---|
| pt0_ice | potential temperature of ice (ITS-90) [ degC ] |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

potential enthalpy [ J/kg ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_pot_enthalpy_from_pt_ice_poly.html

### See Also

Other things related to enthalpy: gsw_CT_from_enthalpy, gsw_dynamic_enthalpy, gsw_enthalpy_CT_exact, gsw_enthalpy_diff, gsw_enthalpy_first_derivatives_CT_exact, gsw_enthalpy_first_derivatives, gsw_enthalpy_ice, gsw_enthalpy_t_exact, gsw_enthalpy, gsw_frazil_properties_potential_poly, gsw_frazil_properties_potential, gsw_pot_enthalpy_from_pt_ice, gsw_pot_enthalpy_ice_freezing_poly, gsw_pot_enthalpy_ice_freezing, gsw_pt_from_pot_enthalpy_ice_poly, gsw_pt_from_pot_enthalpy_ice, gsw_specvol_first_derivatives_wrt_enthalpy, gsw_specvol_first_derivatives

### Examples

```
pt0_ice <- c(-10.7856, -13.4329, -12.8103, -12.2600,  -10.8863,  -8.4036)
e <- gsw_pot_enthalpy_from_pt_ice_poly(pt0_ice)
expect_equal(e/1e5, c(-3.555459482216265, -3.608607100959428, -3.596153924697033,
                      -3.585123214031169, -3.557490561327994, -3.507198320793373))
```

---

gsw_pot_enthalpy_ice_freezing

## *Potential Enthalpy of Ice at Freezing Point*

---

### Description

Potential Enthalpy of Ice at Freezing Point

### Usage

```
gsw_pot_enthalpy_ice_freezing(SA, p, saturation_fraction = 1)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| saturation_fraction | fraction of air in water [unitless] |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

potential enthalpy [ J/kg ]

### Bugs

1. The C source underlying this function lacks an argument, `saturation_fraction`, which is present in the Matlab source, and so that argument is ignored here.

2. The R code does not reproduce the check values stated at http://www.teos-10.org/pubs/gsw/html/gsw_pot_enthalpy_ice_freezing.html. Those values are incorporated in the test provided in "Examples", so that test is not performed during build tests. See https://github.com/TEOS-10/GSW-R/issues/27.

### References

http://www.teos-10.org/pubs/gsw/html/gsw_pot_enthalpy_ice_freezing.html

### See Also

Other things related to enthalpy: gsw_CT_from_enthalpy, gsw_dynamic_enthalpy, gsw_enthalpy_CT_exact, gsw_enthalpy_diff, gsw_enthalpy_first_derivatives_CT_exact, gsw_enthalpy_first_derivatives, gsw_enthalpy_ice, gsw_enthalpy_t_exact, gsw_enthalpy, gsw_frazil_properties_potential_poly, gsw_frazil_properties_potential, gsw_pot_enthalpy_from_pt_ice_poly, gsw_pot_enthalpy_from_pt_ice, gsw_pot_enthalpy_ice_freezing_poly, gsw_pt_from_pot_enthalpy_ice_poly, gsw_pt_from_pot_enthalpy_ice, gsw_specvol_first_derivatives_wrt_enthalpy, gsw_specvol_first_derivatives

### Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
p <- c(      10,      50,     125,     250,     600,    1000)
saturation_fraction = 1
e <- gsw_pot_enthalpy_ice_freezing(SA, p, saturation_fraction)
## Not run:
expect_equal(e/1e5, c(-3.373409558967978, -3.374434164002012, -3.376117536928847,
                    -3.378453698871986, -3.385497832886802, -3.393768587631489))

## End(Not run)
```

---

## gsw_pot_enthalpy_ice_freezing_first_derivatives

*First Derivatives of Potential Enthalpy*

---

### Description

First Derivatives of Potential Enthalpy

### Usage

```
gsw_pot_enthalpy_ice_freezing_first_derivatives(SA, p)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

A list containing `pot_enthalpy_ice_freezing_SA` [ (J/kg)/(g/kg) ], the derivative of potential enthalpy with respect to Absolute Salinity, and `pot_enthalpy_ice_freezing_p` [ unitless ], the derivative of Conservative Temperature with respect to potential temperature. (Note that the second quantity is denoted `pot_enthalpy_ice_freezing_P` in the documentation for the Matlab function.)

### References

http://www.teos-10.org/pubs/gsw/html/gsw_pot_enthalpy_ice_freezing_first_derivatives.html

### Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
p <- c(      10,      50,     125,     250,     600,    1000)
r <- gsw_pot_enthalpy_ice_freezing_first_derivatives(SA, p)
expect_equal(r$pot_enthalpy_ice_freezing_SA/1e2,
      c(-1.183484968590718, -1.184125268891200, -1.184619267864844,
      -1.184026131143674, -1.183727706650925, -1.183814873741961))
expect_equal(r$pot_enthalpy_ice_freezing_p/1e-3,
      c(-0.202880939983260, -0.203087335312542, -0.203473018454630,
      -0.204112435106666, -0.205889571619502, -0.207895691215823))
```

---

## gsw_pot_enthalpy_ice_freezing_first_derivatives_poly

*First Derivatives of Potential Enthalpy (Polynomial version)*

---

### Description

First Derivatives of Potential Enthalpy (Polynomial version)

## Usage

```
gsw_pot_enthalpy_ice_freezing_first_derivatives_poly(SA, p)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

A list containing `pot_enthalpy_ice_freezing_SA` [ (J/kg)/(g/kg) ], the derivative of potential enthalpy with respect to Absolute Salinity, and `pot_enthalpy_ice_freezing_p` [ unitless ], the derivative of Conservative Temperature with respect to potential temperature. (Note that the second quantity is denoted `pot_enthalpy_ice_freezing_P` in the documentation for the Matlab function.)

## References

http://www.teos-10.org/pubs/gsw/html/gsw_pot_enthalpy_ice_freezing_first_derivatives_poly.html

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
p <- c(      10,      50,     125,     250,     600,    1000)
r <- gsw_pot_enthalpy_ice_freezing_first_derivatives_poly(SA, p)
expect_equal(r$pot_enthalpy_ice_freezing_SA/1e2,
      c(-1.183498006918154, -1.184135169530602, -1.184626138334419,
      -1.184032656542549, -1.183727371435808, -1.183805326863513))
expect_equal(r$pot_enthalpy_ice_freezing_p/1e-3,
      c(-0.202934280214689, -0.203136950111241, -0.203515960539503,
      -0.204145112153220, -0.205898365024147, -0.207885289186464))
```

gsw_pot_enthalpy_ice_freezing_poly
*Potential Enthalpy of Ice at Freezing Point (Polynomial version)*

## Description

Potential Enthalpy of Ice at Freezing Point (Polynomial version)

## Usage

```
gsw_pot_enthalpy_ice_freezing_poly(SA, p, saturation_fraction = 1)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| saturation_fraction | fraction of air in water [unitless] |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

potential enthalpy [ J/kg ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_pot_enthalpy_ice_freezing_poly.html

## See Also

Other things related to enthalpy: gsw_CT_from_enthalpy, gsw_dynamic_enthalpy, gsw_enthalpy_CT_exact, gsw_enthalpy_diff, gsw_enthalpy_first_derivatives_CT_exact, gsw_enthalpy_first_derivatives, gsw_enthalpy_ice, gsw_enthalpy_t_exact, gsw_enthalpy, gsw_frazil_properties_potential_poly, gsw_frazil_properties_potential, gsw_pot_enthalpy_from_pt_ice_poly, gsw_pot_enthalpy_from_pt_ice, gsw_pot_enthalpy_ice_freezing, gsw_pt_from_pot_enthalpy_ice_poly, gsw_pt_from_pot_enthalpy_ice, gsw_specvol_first_derivatives_wrt_enthalpy, gsw_specvol_first_derivatives

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
p <- c(      10,      50,     125,     250,     600,    1000)
saturation_fraction = 1
e <- gsw_pot_enthalpy_ice_freezing_poly(SA, p, saturation_fraction)
expect_equal(e/1e5, c(-3.373370858777002, -3.374395733068549, -3.376079507278181,
                    -3.378416106344322, -3.385460970578123, -3.393731732645173))
```

---

`gsw_pot_rho_t_exact` *Potential density*

---

## Description

Potential density

## Usage

```
gsw_pot_rho_t_exact(SA, t, p, p_ref)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| p_ref | reference pressure [ dbar ] |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

potential density [ kg/m^3 ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_pot_rho_t_exact.html

### See Also

Other things related to density: gsw_CT_from_rho, gsw_CT_maxdensity, gsw_SA_from_rho, gsw_alpha_on_beta, gsw_alpha_wrt_t_exact, gsw_alpha_wrt_t_ice, gsw_alpha, gsw_beta_const_t_exact, gsw_beta, gsw_rho_alpha_beta, gsw_rho_first_derivatives_wrt_enthalpy, gsw_rho_first_derivatives, gsw_rho_ice, gsw_rho_t_exact, gsw_rho, gsw_sigma0, gsw_sigma1, gsw_sigma2, gsw_sigma3, gsw_sigma4, gsw_specvol_alpha_beta, gsw_specvol_anom_standard, gsw_specvol_ice, gsw_specvol_t_exact, gsw_specvol

### Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
t <- c( 28.7856, 28.4329, 22.8103, 10.2600,  6.8863,  4.4036)
p <- c(      10,      50,     125,     250,     600,    1000)
p_ref  <- 0
prho <- gsw_pot_rho_t_exact(SA,t,p,p_ref)
expect_equal(prho/1e3, c(1.021798145811089, 1.022052484416980, 1.023893583651958,
                       1.026667621124443, 1.027107230868492, 1.027409631264134))
```

---

## gsw_pressure_coefficient_ice

*Pressure Coefficient for Ice*

---

### Description

Pressure Coefficient for Ice

### Usage

```
gsw_pressure_coefficient_ice(t, p)
```

### Arguments

| | |
|---|---|
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

specific internal energy [ Pa/degC ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_pressure_coefficient_ice.html

## Examples

```
t <- c(-10.7856, -13.4329, -12.8103, -12.2600,  -10.8863,  -8.4036)
p <-  c(     10,       50,      125,      250,      600,     1000)
pc <- gsw_pressure_coefficient_ice(t, p)
expect_equal(pc/1e6, c(1.333098059787838, 1.326359005133730, 1.327354133828322,
                       1.327793888831923, 1.328549609231685, 1.331416733490227))
```

---

gsw_pressure_freezing_CT

*Pressure at which Seawater Freezes*

---

## Description

Pressure at which Seawater Freezes

## Usage

```
gsw_pressure_freezing_CT(SA, CT, saturation_fraction = 1)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| saturation_fraction | fraction of air in water [unitless] |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

pressure at which freezing will occur [ dbar ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_pressure_freezing_CT.html

### Examples

```
SA <- c(                34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(                -1.8996, -1.9407, -2.0062, -2.0923, -2.3593, -2.6771)
saturation_fraction <- c(      1,     0.8,     0.6,     0.5,     0.4,       0)
p <- gsw_pressure_freezing_CT(SA, CT, saturation_fraction)
expect_equal(p/1e3, c(0.009890530270710, 0.050376026585933, 0.125933117050624,
                    0.251150973076077, 0.601441775836021, 1.002273338145043))
```

---

## gsw_pt0_from_t *Potential temperature referenced to the surface*

---

### Description

Potential temperature referenced to the surface

### Usage

```
gsw_pt0_from_t(SA, t, p)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

potential temperature [ degC ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_pt0_from_t.html

### Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
t <- c( 28.7856, 28.4329, 22.8103, 10.2600,  6.8863,  4.4036)
p <- c(      10,      50,     125,     250,     600,    1000)
pt0 <- gsw_pt0_from_t(SA, t, p)
expect_equal(pt0, c(28.783196819670632, 28.420983342398962, 22.784930399117108,
                    10.230523661095731, 6.829230224409661, 4.324510571845719))
```

---

gsw_pt0_from_t_ice *Potential Temperature of Ice Referenced to the Surface*

---

### Description

Potential Temperature of Ice Referenced to the Surface

### Usage

```
gsw_pt0_from_t_ice(t, p)
```

### Arguments

| | |
|---|---|
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

potential temperature [ degC ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_pt0_from_t_ice.html

### Examples

```
t  <- c(-10.7856, -13.4329, -12.8103, -12.2600, -10.8863, -8.4036)
p <- c(       10,       50,      125,      250,      600,    1000)
pt0 <- gsw_pt0_from_t_ice(t, p)
expect_equal(pt0, c(-10.787787898205298, -13.443730926050607, -12.837427056999708,
                  -12.314321615760905, -11.017040858094250, -8.622907355083088))
```

gsw_pt_first_derivatives
*First Derivatives of Potential Temperature*

## Description

First Derivatives of Potential Temperature

## Usage

```
gsw_pt_first_derivatives(SA, CT)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

A list containing `pt_SA` [ K/(g/kg) ], the derivative of potential temperature with respect to Absolute Salinity, and `pt_CT` [ unitless ], the derivative of potential temperature with respect to Conservative Temperature.

## References

http://www.teos-10.org/pubs/gsw/html/gsw_pt_first_derivatives.html

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
r <- gsw_pt_first_derivatives(SA, CT)
expect_equal(r$pt_SA, c(0.041863223165431, 0.041452303483011, 0.034682095247246,
                        0.018711079068408, 0.014079958329844, 0.010577326129948))
expect_equal(r$pt_CT, c(0.997192967140242, 0.997451686508335, 0.998357568277750,
                        0.999996224076267, 1.000283719083268, 1.000525947028218))
```

gsw_pt_from_CT *Potential temperature from Conservative Temperature*

## Description

Potential temperature from Conservative Temperature

## Usage

```
gsw_pt_from_CT(SA, CT)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

potential temperature [ degC ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_pt_from_CT.html

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
pt <- gsw_pt_from_CT(SA, CT)
expect_equal(pt, c(28.783177048624573, 28.420955597191984, 22.784953468087107,
                   10.230534394434429, 6.829216587061605, 4.324534835990236))
```

## gsw_pt_from_entropy *Potential Temperature from Entropy*

### Description

Potential Temperature from Entropy

### Usage

```
gsw_pt_from_entropy(SA, entropy)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| entropy | specific entropy [ J/(degC*kg) ] |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

potential temperature [ degC ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_pt_from_entropy.html

### See Also

Other things related to entropy: gsw_CT_from_entropy, gsw_entropy_first_derivatives, gsw_entropy_from_pt, gsw_entropy_from_t, gsw_entropy_ice

### Examples

```
SA <- c(      34.7118,  34.8915,  35.0256,  34.8472, 34.7366, 34.7324)
entropy <- c(400.3892, 395.4378, 319.8668, 146.7910, 98.6471, 62.7919)
pt <- gsw_pt_from_entropy(SA, entropy)
expect_equal(pt, c(28.783179828078666, 28.420954825949291, 22.784952736245351,
                   10.230532066931868, 6.829213325916900, 4.324537782985845))
```

gsw_pt_from_pot_enthalpy_ice
*Potential Temperature from Potential Enthalpy of Ice*

### Description

Potential Temperature from Potential Enthalpy of Ice

### Usage

```
gsw_pt_from_pot_enthalpy_ice(pot_enthalpy_ice)
```

### Arguments

pot_enthalpy_ice
potential enthalpy of ice [ J/kg ]

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

potential temperature [ degC ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_pt_from_pot_enthalpy_ice.html

### See Also

Other things related to enthalpy: gsw_CT_from_enthalpy, gsw_dynamic_enthalpy, gsw_enthalpy_CT_exact, gsw_enthalpy_diff, gsw_enthalpy_first_derivatives_CT_exact, gsw_enthalpy_first_derivatives, gsw_enthalpy_ice, gsw_enthalpy_t_exact, gsw_enthalpy, gsw_frazil_properties_potential_poly, gsw_frazil_properties_potential, gsw_pot_enthalpy_from_pt_ice_poly, gsw_pot_enthalpy_from_pt_ice, gsw_pot_enthalpy_ice_freezing_poly, gsw_pot_enthalpy_ice_freezing, gsw_pt_from_pot_enthalpy_ice_poly, gsw_specvol_first_derivatives_wrt_enthalpy, gsw_specvol_first_derivatives

### Examples

```
pot_enthalpy_ice <- c(-3.5544e5, -3.6033e5, -3.5830e5, -3.5589e5, -3.4948e5, -3.4027e5)
pt <- gsw_pt_from_pot_enthalpy_ice(pot_enthalpy_ice)
expect_equal(pt, c(-10.733087588125384, -13.167397822300588, -12.154205899172704,
                   -10.956202704066083, -7.794963180206421, -3.314905214262531))
```

gsw_pt_from_pot_enthalpy_ice_poly
*Potential Temperature from Potential Enthalpy of Ice (Polynomial version)*

## Description

Potential Temperature from Potential Enthalpy of Ice (Polynomial version)

## Usage

```
gsw_pt_from_pot_enthalpy_ice_poly(pot_enthalpy_ice)
```

## Arguments

`pot_enthalpy_ice`
potential enthalpy of ice [ J/kg ]

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

potential temperature [ degC ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_pt_from_pot_enthalpy_ice_poly.html

## See Also

Other things related to enthalpy: gsw_CT_from_enthalpy, gsw_dynamic_enthalpy, gsw_enthalpy_CT_exact, gsw_enthalpy_diff, gsw_enthalpy_first_derivatives_CT_exact, gsw_enthalpy_first_derivatives, gsw_enthalpy_ice, gsw_enthalpy_t_exact, gsw_enthalpy, gsw_frazil_properties_potential_poly, gsw_frazil_properties_potential, gsw_pot_enthalpy_from_pt_ice_poly, gsw_pot_enthalpy_from_pt_ice, gsw_pot_enthalpy_ice_freezing_poly, gsw_pot_enthalpy_ice_freezing, gsw_pt_from_pot_enthalpy_ice, gsw_specvol_first_derivatives_wrt_enthalpy, gsw_specvol_first_derivatives

### Examples

```
pot_enthalpy_ice <- c(-3.5544e5, -3.6033e5, -3.5830e5, -3.5589e5, -3.4948e5, -3.4027e5)
pt <- gsw_pt_from_pot_enthalpy_ice_poly(pot_enthalpy_ice)
expect_equal(pt, c(-10.733085986035007, -13.167396204945987, -12.154204137867396,
                   -10.956201046447006, -7.794963341294590, -3.314907552013722))
```

---

gsw_pt_from_t *Potential Temperature from in-situ Temperature*

---

### Description

Potential Temperature from in-situ Temperature

### Usage

```
gsw_pt_from_t(SA, t, p, p_ref = 0)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| p_ref | reference pressure [ dbar ] |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

potential temperature [ degC ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_pt_from_t.html

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
t <- c( 28.7856, 28.4329, 22.8103, 10.2600,  6.8863,  4.4036)
p <- c(      10,      50,     125,     250,     600,    1000)
p_ref <- 0
pt <- gsw_pt_from_t(SA, t, p, p_ref)
expect_equal(pt, c(28.783196819670632, 28.420983342398962, 22.784930399117108,
                   10.230523661095731, 6.829230224409661, 4.324510571845719))
```

---

`gsw_pt_from_t_ice` *Potential Temperature of Ice from in-situ Temperature*

---

## Description

Potential Temperature of Ice from in-situ Temperature

## Usage

```
gsw_pt_from_t_ice(t, p, p_ref = 0)
```

## Arguments

| | |
|---|---|
| `t` | in-situ temperature (ITS-90) [ degC ] |
| `p` | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| `p_ref` | reference pressure [ dbar ] |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

potential temperature [ degC ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_pt_from_t_ice.html

## Examples

```
t <- c(-10.7856, -13.4329, -12.8103, -12.2600, -10.8863, -8.4036)
p <- c(      10,       50,      125,      250,      600,    1000)
p_ref <- 0 # not actually needed, since 0 is the default
pt <- gsw_pt_from_t_ice(t, p, p_ref)
expect_equal(pt, c(-10.787787898205272, -13.443730926050661, -12.837427056999676,
                 -12.314321615760921, -11.017040858094234, -8.622907355083147))
```

---

gsw_pt_second_derivatives    *Second Derivatives of Potential Temperature*

---

## Description

Second Derivatives of Potential Temperature

## Usage

```
gsw_pt_second_derivatives(SA, CT)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

A list containing `pt_SA_SA` [ K/(g/kg)^2 ], the second derivative of potential temperature with respect to Absolute Salinity at constant potential temperature, and `pt_SA_pt` [ 1/(g/kg) ], the derivative of potential temperature with respect to Conservative Temperature and Absolute Salinity, and `pt_pt_pt` [ 1/degC ], the second derivative of potential temperature with respect to Conservative Temperature.

## References

http://www.teos-10.org/pubs/gsw/html/gsw_pt_second_derivatives.html

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
r <- gsw_pt_second_derivatives(SA, CT)
expect_equal(r$pt_SA_SA/1e-3, c(0.160307058371208, 0.160785497957769, 0.168647220588324,
                                0.198377949876584, 0.210181899321236, 0.220018966513329))
expect_equal(r$pt_SA_CT, c(0.001185581323691, 0.001187068518686, 0.001217629686266,
                           0.001333254154015, 0.001379674342678, 0.001418371539325))
expect_equal(r$pt_CT_CT/1e-3, c(-0.121979811279463, -0.123711264754503, -0.140136818504977,
                               -0.140645384127949, -0.113781055410824, -0.082417269009484))
```

---

`gsw_p_from_z` *Pressure from height (75-term equation)*

---

## Description

Pressure from height (75-term equation)

## Usage

```
gsw_p_from_z(z, latitude)
```

## Arguments

| | |
|---|---|
| `z` | height, zero at surface (but note last 2 args) and positive upwards [ m ] |
| `latitude` | latitude in decimal degrees, positive to the north of the equator. (This is called `lat` in the TEOS-10 Matlab code.) |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

sea pressure [ dbar ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_p_from_z.html

## See Also

Other things related to depth: gsw_z_from_p

### Examples

```
z <- -c(10, 50, 125, 250, 600, 1000)
latitude <- 4
p <- gsw_p_from_z(z, latitude)
expect_equal(p/1e3, c(0.010055726724518, 0.050283543374874, 0.125731858435610,
                      0.251540299593468, 0.604210012340727, 1.007990337692001))
```

---

## gsw_rho *In-situ density*

---

### Description

In-situ density, using the 75-term equation for specific volume.

### Usage

```
gsw_rho(SA, CT, p)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

in-situ density [ kg/m^3 ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_rho.html

### See Also

Other things related to density: gsw_CT_from_rho, gsw_CT_maxdensity, gsw_SA_from_rho, gsw_alpha_on_beta, gsw_alpha_wrt_t_exact, gsw_alpha_wrt_t_ice, gsw_alpha, gsw_beta_const_t_exact, gsw_beta, gsw_pot_rho_t_exact, gsw_rho_alpha_beta, gsw_rho_first_derivatives_wrt_enthalpy, gsw_rho_first_derivat: gsw_rho_ice, gsw_rho_t_exact, gsw_sigma0, gsw_sigma1, gsw_sigma2, gsw_sigma3, gsw_sigma4, gsw_specvol_alpha_beta, gsw_specvol_anom_standard, gsw_specvol_ice, gsw_specvol_t_exact, gsw_specvol

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
p <-  c(     10,      50,     125,     250,     600,    1000)
rho <- gsw_rho(SA,CT,p)
expect_equal(rho/1e3, c(1.021839935738108, 1.022262457966867, 1.024427195413316,
                        1.027790152759127, 1.029837779000189, 1.032002453224572))
```

---

# gsw_rho_alpha_beta

*In-situ density, thermal expansion coefficient and haline contraction coefficient (75-term equation)*

---

## Description

Calculate the in-situ density, the expansion coefficient (with respect to Conservative Temperature) and the haline contraction coefficient (with respect to Absolute Salinity), using the 75-term equation.

## Usage

```
gsw_rho_alpha_beta(SA, CT, p)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

A list containing in-situ density `rho` [ kg/m^3 ], thermal expansion coefficient `alpha` [ 1/degC ], and haline contraction coefficient `beta` [ kg/g ].

## References

http://www.teos-10.org/pubs/gsw/html/gsw_rho_alpha_beta.html

## See Also

Other things related to density: gsw_CT_from_rho, gsw_CT_maxdensity, gsw_SA_from_rho, gsw_alpha_on_beta, gsw_alpha_wrt_t_exact, gsw_alpha_wrt_t_ice, gsw_alpha, gsw_beta_const_t_exact, gsw_beta, gsw_pot_rho_t_exact, gsw_rho_first_derivatives_wrt_enthalpy, gsw_rho_first_derivatives, gsw_rho_ice, gsw_rho_t_exact, gsw_rho, gsw_sigma0, gsw_sigma1, gsw_sigma2, gsw_sigma3, gsw_sigma4, gsw_specvol_alpha_beta, gsw_specvol_anom_standard, gsw_specvol_ice, gsw_specvol_t_exact, gsw_specvol

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
p <-  c(     10,      50,     125,     250,     600,    1000)
r <- gsw_rho_alpha_beta(SA, CT, p)
expect_equal(r$rho/1000, c(1.021839935738108, 1.022262457966867, 1.024427195413316,
                         1.027790152759127, 1.029837779000189, 1.032002453224572))
expect_equal(r$alpha*1000, c(0.324638934509245, 0.322655537959731, 0.281145723210171,
                           0.173199716344780, 0.146289673594824, 0.129414845334599))
expect_equal(r$beta*1000, c(0.717483987596135, 0.717647512290095, 0.726211643644768,
                          0.750500751749777, 0.755052064788492, 0.757050813384370))
```

---

gsw_rho_first_derivatives

*Density First Derivatives wrt SA, CT and p (75-term equation)*

---

## Description

Density First Derivatives wrt SA, CT and p (75-term equation)

## Usage

```
gsw_rho_first_derivatives(SA, CT, p)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

list containing drho_dSA [ kg^2/(g m^3) ], drho_dCT [ kg/(K m^3) ] and drho_dp [ kg/(Pa m^3) ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_rho_first_derivatives.html

### See Also

Other things related to density: gsw_CT_from_rho, gsw_CT_maxdensity, gsw_SA_from_rho, gsw_alpha_on_beta, gsw_alpha_wrt_t_exact, gsw_alpha_wrt_t_ice, gsw_alpha, gsw_beta_const_t_exact, gsw_beta, gsw_pot_rho_t_exact, gsw_rho_alpha_beta, gsw_rho_first_derivatives_wrt_enthalpy, gsw_rho_ice, gsw_rho_t_exact, gsw_rho, gsw_sigma0, gsw_sigma1, gsw_sigma2, gsw_sigma3, gsw_sigma4, gsw_specvol_alpha_beta, gsw_specvol_anom_standard, gsw_specvol_ice, gsw_specvol_t_exact, gsw_specvol

### Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
p <- c(      10,      50,     125,     250,     600,    1000)
r <- gsw_rho_first_derivatives(SA, CT, p)
expect_equal(r$drho_dSA, c(0.733153791778356, 0.733624109867480, 0.743950957375504,
                           0.771357282286743, 0.777581141431288, 0.781278296628328))
expect_equal(r$drho_dCT, c(-0.331729027977015, -0.329838643311336, -0.288013324730644,
                           -0.178012962919839, -0.150654632545556, -0.133556437868984))
expect_equal(r$drho_dp, 1e-6*c(0.420302360738476, 0.420251070273888, 0.426773054953941,
                               0.447763615252861, 0.452011501791479, 0.454118117103094))
```

---

gsw_rho_first_derivatives_wrt_enthalpy
*Density First Derivatives wrt enthalpy (75-term equation)*

---

### Description

Density First Derivatives wrt enthalpy (75-term equation)

### Usage

```
gsw_rho_first_derivatives_wrt_enthalpy(SA, CT, p)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

A list containing `rho_SA_wrt_h` [ (kg/m^3)/(g/kg)/(J/kg) ] and `rho_h` [ (kg/m^3)/(J/kg) ].

### References

http://www.teos-10.org/pubs/gsw/html/gsw_rho_first_derivatives_wrt_enthalpy.html

### See Also

Other things related to density: gsw_CT_from_rho, gsw_CT_maxdensity, gsw_SA_from_rho, gsw_alpha_on_beta, gsw_alpha_wrt_t_exact, gsw_alpha_wrt_t_ice, gsw_alpha, gsw_beta_const_t_exact, gsw_beta, gsw_pot_rho_t_exact, gsw_rho_alpha_beta, gsw_rho_first_derivatives, gsw_rho_ice, gsw_rho_t_exact, gsw_rho, gsw_sigma0, gsw_sigma1, gsw_sigma2, gsw_sigma3, gsw_sigma4, gsw_specvol_alpha_beta, gsw_specvol_anom_standard, gsw_specvol_ice, gsw_specvol_t_exact, gsw_specvol

### Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
p <- c(      10,      50,     125,     250,     600,    1000)
r <- gsw_rho_first_derivatives_wrt_enthalpy(SA, CT, p)
expect_equal(r$rho_SA_wrt_h, c(0.733147960400929, 0.733595114830609, 0.743886977148835,
                               0.771275693831993, 0.777414200397148, 0.781030546357425))
expect_equal(r$rho_h*1e4, c(-0.831005413475887, -0.826243794873652, -0.721438289309903,
                            -0.445892608094272, -0.377326924646647, -0.334475962698187))
```

---

`gsw_rho_ice` *In-situ density of ice*

---

### Description

In-situ density of ice [kg/m^3]

### Usage

```
gsw_rho_ice(t, p)
```

### Arguments

| | |
|---|---|
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

in-situ density [ kg/m^3 ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_rho_ice.html

### See Also

Other things related to density: gsw_CT_from_rho, gsw_CT_maxdensity, gsw_SA_from_rho, gsw_alpha_on_beta, gsw_alpha_wrt_t_exact, gsw_alpha_wrt_t_ice, gsw_alpha, gsw_beta_const_t_exact, gsw_beta, gsw_pot_rho_t_exact, gsw_rho_alpha_beta, gsw_rho_first_derivatives_wrt_enthalpy, gsw_rho_first_derivat gsw_rho_t_exact, gsw_rho, gsw_sigma0, gsw_sigma1, gsw_sigma2, gsw_sigma3, gsw_sigma4, gsw_specvol_alpha_beta, gsw_specvol_anom_standard, gsw_specvol_ice, gsw_specvol_t_exact, gsw_specvol

### Examples

```
t <- c(-10.7856, -13.4329, -12.8103, -12.2600, -10.8863, -8.4036)
p <-  c(      10,       50,      125,      250,      600,     1000)
rho <- gsw_rho_ice(t, p)
expect_equal(rho, c(918.2879969148962, 918.7043487325120, 918.6962796312690,
             918.7513732275766, 918.9291139833307, 919.0032237449378))
```

---

gsw_rho_second_derivatives

*Second Derivatives of Density*

---

### Description

Second Derivatives of Density

## Usage

```
gsw_rho_second_derivatives(SA, CT, p)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

A list containing `rho_SA_SA` [ (kg/m^3)/(g/kg)^2 ], the second derivative of density with respect to Absolute Salinity, `rho_SA_CT` [ (g/kg)/(g/kg)/degC ], the derivative of density with respect to Absolute Salinity and Conservative Temperature, and `rho_CT_CT` [ (kg/m^3)/degC^2 ], the second derivative of density with respect to Conservative Temperature.

## References

http://www.teos-10.org/pubs/gsw/html/gsw_rho_second_derivatives.html

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.7856, 28.4329, 22.8103, 10.2600,  6.8863,  4.4036)
p <- c(     10,      50,     125,     250,     600,    1000)
r <- gsw_rho_second_derivatives(SA, CT, p)
expect_equal(r$rho_SA_SA/1e-3, c(0.207364734477357, 0.207415414547223, 0.192903197286004,
                                 0.135809142211237, 0.122627562106076, 0.114042431905783))
expect_equal(r$rho_SA_CT, c(-0.001832856561477, -0.001837354806146, -0.001988065808078,
                            -0.002560181494807, -0.002708939446458, -0.002798484050141))
expect_equal(r$rho_CT_CT, c(-0.007241243828334, -0.007267807914635, -0.007964270843331,
                            -0.010008164822017, -0.010572200761984, -0.010939294762200))
expect_equal(r$rho_SA_p/1e-8, c(-0.087202931942412, -0.087558612009845, -0.092549696987409,
                              -0.106661486272630, -0.110022261844240, -0.112287954816717))
expect_equal(r$rho_CT_p/1e-8, c(-0.116597992537549, -0.117744271236102, -0.141712549466964,
                              -0.214414626736539, -0.237704139801551, -0.255296606034074))
```

gsw_rho_second_derivatives_wrt_enthalpy
*Second Derivatives of Density wrt Enthalpy*

## Description

Second Derivatives of Density wrt Enthalpy

## Usage

```
gsw_rho_second_derivatives_wrt_enthalpy(SA, CT, p)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

A list containing `rho_SA_SA` [ (kg/m^3)/(g/kg)^2 ], the second derivative of density with respect to Absolute Salinity, `rho_SA_h` [ (g/kg)/(g/kg)/(J/kg)], the derivative of density with respect to Absolute Salinity and enthalpy, and `rho_h_h` [ (kg/m^3)/(J/kg)^2 ], the second derivative of density with respect to enthalpy.

## Bugs

As of March 27, 2017, the test values listed in "Examples" do not match values provided at the TEOS-10 website listed in "References", but they match with values given by the Matlab code that is provided on the TEOS-10 website. It is expected that the TEOS-10 website will be updated by May 2017. As those updates to the TEOS-10 website become available, the present comment will be revised or deleted.

## References

http://www.teos-10.org/pubs/gsw/html/gsw_rho_second_derivatives_wrt_enthalpy.html

### See Also

Other functions with suspicious test values on the TEOS-10 website: gsw_entropy_second_derivatives, gsw_specvol_second_derivatives_wrt_enthalpy, gsw_t_freezing_first_derivatives_poly

### Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
p <- c(      10,      50,     125,     250,     600,    1000)
r <- gsw_rho_second_derivatives_wrt_enthalpy(SA, CT, p)
expect_equal(r$rho_SA_SA/1e-3, c(0.207312267114544, 0.207065033523473, 0.191848346945039,
                                 0.133182862881598, 0.116049034622904, 0.102745309429078))
expect_equal(r$rho_SA_h/1e-6, c(-0.459053080088382, -0.460370569872258, -0.498605615416296,
                              -0.642833108550133, -0.682091962941161, -0.706793055445909))
expect_equal(r$rho_h_h/1e-9, c(-0.454213854637790, -0.455984900239309, -0.499870030989387,
                              -0.628337767293403, -0.664021595759308, -0.687367088752173))
```

---

## gsw_rho_t_exact *In-situ Density of Seawater*

---

### Description

In-situ Density of Seawater

### Usage

```
gsw_rho_t_exact(SA, t, p)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

in-situ density [ kg/m^3 ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_rho_t_exact.html

### See Also

Other things related to density: gsw_CT_from_rho, gsw_CT_maxdensity, gsw_SA_from_rho, gsw_alpha_on_beta, gsw_alpha_wrt_t_exact, gsw_alpha_wrt_t_ice, gsw_alpha, gsw_beta_const_t_exact, gsw_beta, gsw_pot_rho_t_exact, gsw_rho_alpha_beta, gsw_rho_first_derivatives_wrt_enthalpy, gsw_rho_first_derivat: gsw_rho_ice, gsw_rho, gsw_sigma0, gsw_sigma1, gsw_sigma2, gsw_sigma3, gsw_sigma4, gsw_specvol_alpha_beta, gsw_specvol_anom_standard, gsw_specvol_ice, gsw_specvol_t_exact, gsw_specvol

### Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
t <-  c(28.7856, 28.4329, 22.8103, 10.2600,  6.8863,  4.4036)
p <-  c(     10,      50,     125,     250,     600,    1000)
rho <- gsw_rho_t_exact(SA, t, p)
expect_equal(rho/1e3, c(1.021840173185531, 1.022262689926782, 1.024427715941676,
                        1.027790201811623, 1.029837714725961, 1.032002404116447))
```

---

`gsw_SAAR` *Absolute Salinity Anomaly Ratio*

---

### Description

Absolute Salinity Anomaly Ratio

### Usage

```
gsw_SAAR(p, longitude, latitude)
```

### Arguments

| | |
|---|---|
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| longitude | longitude in decimal degrees, positive to the east of Greenwich. (This is called long in the TEOS-10 Matlab code.) |
| latitude | latitude in decimal degrees, positive to the north of the equator. (This is called lat in the TEOS-10 Matlab code.) |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

a list containing SAAR, which is the (unitless) Absolute Salinity Anomality Ratio, and in_ocean is set to 1 if SAAR is nonzero, or to 0 otherwise.

### Bugs

The definition of in_ocean is incorrect, because the C function named gsw_saar, which is called by the present R function, does not calculate in_ocean, as the base Matlab function named gsw_SAAR does. However, examination of the Matlab code shows that in_ocean is set to 0 along with SAAR, whenever the original estimate of the latter is nonfinite. Thus, points that would be siganlled as being on the land by the Matlab code are indicated in the same way with the present R function. However, other points may also be indicated as being on land, if SAAR is simply zero in the first calculation. Whether this poses a problem in practice is an open question, since it seems likely that this function would only be called with oceanic locations, anyway. If problems arise for users, a patch can be written to improve things.

### References

http://www.teos-10.org/pubs/gsw/html/gsw_saar.html

### Examples

```
p <- c(10, 50, 125, 250, 600, 1000)
longitude <- c(188, 188, 188, 188, 188, 188)
latitude <- c(4, 4, 4, 4, 4, 4)
SAAR <- gsw_SAAR(p, longitude, latitude)
expect_equal(1e3*SAAR$SAAR, c(0.004794295602143, 0.007668755837570, 0.018919828449091,
                              0.077293264028981, 0.161974583039298, 0.270652408428964))
expect_equal(SAAR$in_ocean, rep(1, 6))
```

---

## gsw_SA_freezing_from_CT

*Compute Absolute Salinity at Freezing Conservative Temperature*

---

### Description

Compute Absolute Salinity at Freezing Conservative Temperature

### Usage

```
gsw_SA_freezing_from_CT(CT, p, saturation_fraction = 1)
```

### Arguments

| | |
|---|---|
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| saturation_fraction | fraction of air in water [unitless] |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

Absolute Salinity [ g/kg ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_SA_freezing_from_CT.html

## Examples

```
CT <- c(-0.11901, -0.15608, -0.72138, -1.97738, -2.31728, -2.56764)
p <- c(       10,       50,      125,      250,      600,     1000)
saturation_fraction <- 1
SA <- gsw_SA_freezing_from_CT(CT, p, saturation_fraction)
expect_equal(SA, c(2.280500648179144, 2.416867651098550, 11.973503162175106,
                 32.868973869711390, 34.017513292374431, 32.859871943514150))
```

---

gsw_SA_freezing_from_CT_poly

*Compute Absolute Salinity at Freezing Point (Polynomial version)*

---

## Description

Compute Absolute Salinity at Freezing Point (Polynomial version)

## Usage

```
gsw_SA_freezing_from_CT_poly(CT, p, saturation_fraction = 1)
```

## Arguments

| | |
|---|---|
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| saturation_fraction | fraction of air in water [unitless] |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

Absolute Salinity [ g/kg ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_SA_freezing_from_CT_poly.html

### Examples

```
CT <- c(-0.11901, -0.15608, -0.72138, -1.97738, -2.31728, -2.56764)
p <- c(       10,       50,      125,      250,      600,     1000)
saturation_fraction <- 1
SA <- gsw_SA_freezing_from_CT_poly(CT, p, saturation_fraction)
expect_equal(SA, c(2.281810267792954, 2.418134292641376, 11.971996354752958,
                   32.867931280363138, 34.015087798162732, 32.856434894818825))
```

---

`gsw_SA_freezing_from_t` *Compute Absolute Salinity at Freezing in-situ Temperature*

---

### Description

Compute Absolute Salinity at Freezing in-situ Temperature

### Usage

```
gsw_SA_freezing_from_t(t, p, saturation_fraction = 1)
```

### Arguments

| | |
|---|---|
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| saturation_fraction | fraction of air in water [unitless] |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

Absolute Salinity [ g/kg ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_SA_freezing_from_t.html

### Examples

```
t <- c(-0.11901, -0.15608, -0.72138, -1.97738, -2.31728, -2.56764)
p <- c(      10,       50,      125,      250,      600,     1000)
saturation_fraction <- 1
SA <- gsw_SA_freezing_from_t(t, p, saturation_fraction)
expect_equal(SA, c(2.015798440008186, 2.150742019102164, 11.679080083422074,
                   32.844196564019278, 34.138949682974413, 33.100945437175568))
```

---

## gsw_SA_freezing_from_t_poly

*Compute Absolute Salinity at Freezing in-situ Temperature (Polynomial version)*

---

### Description

Compute Absolute Salinity at Freezing in-situ Temperature (Polynomial version)

### Usage

```
gsw_SA_freezing_from_t_poly(t, p, saturation_fraction = 1)
```

### Arguments

| | |
|---|---|
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| saturation_fraction | fraction of air in water [unitless] |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

Absolute Salinity [ g/kg ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_SA_freezing_from_t_poly.html

## Examples

```
t <- c(-0.11901, -0.15608, -0.72138, -1.97738, -2.31728, -2.56764)
p <- c(      10,       50,      125,      250,      600,     1000)
saturation_fraction <- 1
SA <- gsw_SA_freezing_from_t_poly(t, p, saturation_fraction)
expect_equal(SA, c(2.017072489768256, 2.151989342038462, 11.677649626115608,
                 32.843128114999026, 34.136459306273451, 33.097427522625182))
```

---

gsw_SA_from_rho *Compute Absolute Salinity from Density, etc*

---

## Description

Compute Absolute Salinity from Density, etc

## Usage

```
gsw_SA_from_rho(rho, CT, p)
```

## Arguments

| | |
|---|---|
| rho | seawater density [ kg/m^3 ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

Absolute Salinity [ g/kg ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_SA_from_rho.html

### See Also

Other things related to density: gsw_CT_from_rho, gsw_CT_maxdensity, gsw_alpha_on_beta, gsw_alpha_wrt_t_exact, gsw_alpha_wrt_t_ice, gsw_alpha, gsw_beta_const_t_exact, gsw_beta, gsw_pot_rho_t_exact, gsw_rho_alpha_beta, gsw_rho_first_derivatives_wrt_enthalpy, gsw_rho_first_derivat gsw_rho_ice, gsw_rho_t_exact, gsw_rho, gsw_sigma0, gsw_sigma1, gsw_sigma2, gsw_sigma3, gsw_sigma4, gsw_specvol_alpha_beta, gsw_specvol_anom_standard, gsw_specvol_ice, gsw_specvol_t_exact, gsw_specvol

### Examples

```
rho <- c(1021.8482, 1022.2647, 1024.4207, 1027.7841, 1029.8287, 1031.9916)
CT <-c(    28.7856,   28.4329,   22.8103,   10.2600,    6.8863,    4.4036)
p <- c(         10,        50,       125,       250,       600,      1000)
SA <- gsw_SA_from_rho(rho, CT, p)
expect_equal(SA, c(34.712080120418108, 34.891723808488869, 35.026202257609505,
                   34.847160842234572, 34.736398269039945, 34.732228881079742))
```

---

gsw_SA_from_SP *Convert from Practical Salinity to Absolute Salinity*

---

### Description

Calculate Absolute Salinity from Practical Salinity, pressure, longitude, and latitude.

### Usage

```
gsw_SA_from_SP(SP, p, longitude, latitude)
```

## Arguments

| | |
|---|---|
| SP | Practical Salinity (PSS-78) [ unitless ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| longitude | longitude in decimal degrees, positive to the east of Greenwich. (This is called `long` in the TEOS-10 Matlab code.) |
| latitude | latitude in decimal degrees, positive to the north of the equator. (This is called `lat` in the TEOS-10 Matlab code.) |

## Details

If SP is a matrix and if its dimensions correspond to the lengths of longitude and latitude, then the latter are converted to analogous matrices with `expand.grid`.

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at `https://github.com/TEOS-10/GSW-C`, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (`https://github.com/TEOS-10/GSW-Fortran`) which in turn stems from the GSW-Matlab system (`https://github.com/TEOS-10/GSW-Matlab`). Consult `http://www.teos-10.org` to learn more about these software systems, their authorships, and the science behind it all.

## Value

Absolute Salinity [ g/kg ]

## References

`http://www.teos-10.org/pubs/gsw/html/gsw_SA_from_SP.html`

## See Also

Other things related to salinity: `gsw_C_from_SP`, `gsw_SA_from_SP_Baltic`, `gsw_SA_from_Sstar`, `gsw_SP_from_C`, `gsw_SP_from_SA`, `gsw_SP_from_SK`, `gsw_SP_from_SR`, `gsw_SP_from_Sstar`, `gsw_SR_from_SP`, `gsw_Sstar_from_SA`, `gsw_Sstar_from_SP`, `gsw_deltaSA_from_SP`

## Examples

```
SP <- c(34.5487, 34.7275, 34.8605, 34.6810, 34.5680, 34.5600)
p <- c(      10,      50,     125,     250,     600,    1000)
lat <- c(     4,       4,       4,       4,       4,       4)
long <- c(  188,     188,     188,     188,     188,     188)
SA <- gsw_SA_from_SP(SP, p, long, lat)
expect_equal(SA, c(34.711778344814114, 34.891522618230098, 35.025544862476920,
                   34.847229026189588, 34.736628474576051, 34.732363065590846))
```

## gsw_SA_from_SP_Baltic *Convert from Practical Salinity to Absolute Salinity (Baltic)*

### Description

Calculate Absolute Salinity from Practical Salinity, pressure, longitude, and latitude.

### Usage

```
gsw_SA_from_SP_Baltic(SP, longitude, latitude)
```

### Arguments

| | |
|---|---|
| SP | Practical Salinity (PSS-78) [ unitless ] |
| longitude | longitude in decimal degrees, positive to the east of Greenwich. (This is called long in the TEOS-10 Matlab code.) |
| latitude | latitude in decimal degrees, positive to the north of the equator. (This is called lat in the TEOS-10 Matlab code.) |

### Details

If SP is a matrix and if its dimensions correspond to the lengths of longitude and latitude, then the latter are converted to analogous matrices with expand.grid.

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

Absolute Salinity [ g/kg ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_SA_from_SP_Baltic.html

### See Also

Other things related to salinity: gsw_C_from_SP, gsw_SA_from_SP, gsw_SA_from_Sstar, gsw_SP_from_C, gsw_SP_from_SA, gsw_SP_from_SK, gsw_SP_from_SR, gsw_SP_from_Sstar, gsw_SR_from_SP, gsw_Sstar_from_SA, gsw_Sstar_from_SP, gsw_deltaSA_from_SP

### Examples

```
SP <- c( 6.5683, 6.6719, 6.8108, 7.2629, 7.4825, 10.2796)
lon <- c(    20,     20,     20,     20,     20,      20)
lat <- c(    59,     59,     59,     59,     59,      59)
SA <- gsw_SA_from_SP_Baltic(SP, lon, lat)
expect_equal(SA, c(6.669945432342856, 6.773776430742856, 6.912986138057142,
                   7.366094191885713, 7.586183837142856, 10.389520570971428))
```

---

`gsw_SA_from_Sstar` *Absolute Salinity from Preformed Salinity*

---

### Description

Calculate Absolute Salinity from Preformed Salinity, pressure, longitude, and latitude.

### Usage

```
gsw_SA_from_Sstar(Sstar, p, longitude, latitude)
```

### Arguments

| | |
|---|---|
| `Sstar` | Preformed Salinity [ g/kg ] |
| `p` | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| `longitude` | longitude in decimal degrees, positive to the east of Greenwich. (This is called `long` in the TEOS-10 Matlab code.) |
| `latitude` | latitude in decimal degrees, positive to the north of the equator. (This is called `lat` in the TEOS-10 Matlab code.) |

### Details

If Sstar is a matrix and if its dimensions correspond to the lengths of longitude and latitude, then the latter are converted to analogous matrices with `expand.grid`.

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at `https://github.com/TEOS-10/GSW-C`, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (`https://github.com/TEOS-10/GSW-Fortran`) which in turn stems from the GSW-Matlab system (`https://github.com/TEOS-10/GSW-Matlab`). Consult `http://www.teos-10.org` to learn more about these software systems, their authorships, and the science behind it all.

### Value

Absolute Salinity [ g/kg ]

### References

`http://www.teos-10.org/pubs/gsw/html/gsw_SA_from_Sstar.html`

### See Also

Other things related to salinity: gsw_C_from_SP, gsw_SA_from_SP_Baltic, gsw_SA_from_SP, gsw_SP_from_C, gsw_SP_from_SA, gsw_SP_from_SK, gsw_SP_from_SR, gsw_SP_from_Sstar, gsw_SR_from_SP, gsw_Sstar_from_SA, gsw_Sstar_from_SP, gsw_deltaSA_from_SP

### Examples

```
SP <- c(34.7115, 34.8912, 35.0247, 34.8436, 34.7291, 34.7197)
p <- c(      10,      50,     125,     250,     600,    1000)
lat <- c(     4,       4,       4,       4,       4,       4)
long <- c(  188,     188,     188,     188,     188,     188)
SA <- gsw_SA_from_Sstar(SP, p, long, lat)
expect_equal(SA, c(34.711724663585905, 34.891561223296009, 35.025594598699882,
                   34.847235885385913, 34.736694493054166, 34.732387111902753))
```

---

`gsw_seaice_fraction_to_freeze_seawater`

*Sea ice Fraction to Cool Seawater to Freezing*

---

### Description

Sea ice Fraction to Cool Seawater to Freezing

### Usage

```
gsw_seaice_fraction_to_freeze_seawater(SA, CT, p, SA_seaice, t_seaice)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| SA_seaice | Absolute Salinity of sea ice [ g/kg ] |
| t_seaice | initial temperature of sea ice [ degC ] |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

a list containing SA_freeze, CT_freeze and w_Ih.

### References

http://www.teos-10.org/pubs/gsw/html/gsw_seaice_fraction_to_freeze_seawater.html

### Examples

```
SA <- c(      34.7118, 34.8915, 35.0256, 34.8472, 34.7366,  34.7324)
CT <- c(      -1.7856, -1.4329, -1.8103, -1.2600, -0.6886,   0.4403)
p <- c(            10,      50,     125,     250,     600,     1000)
SA_seaice <- c(     5,     4.8,     3.5,     2.5,       1,      0.4)
t_seaice <- c(-5.7856, -4.4329, -3.8103, -4.2600, -3.8863,  -3.4036)
r <- gsw_seaice_fraction_to_freeze_seawater(SA, CT, p, SA_seaice, t_seaice)
expect_equal(r$SA_freeze, c(34.671271207148074, 34.703449677481224, 34.950192062047861,
                          34.525277379661880, 34.077349518029997, 33.501836583274191))
expect_equal(r$CT_freeze, c(-1.895419711000293, -1.927935638317893, -1.999943183939312,
                          -2.071677444370745, -2.318866154643864, -2.603185031462614))
expect_equal(r$w_seaice, c(0.001364063868629, 0.006249283768465, 0.002391958850970,
                         0.009952101583387, 0.019541106156815, 0.035842627277027))
```

---

## gsw_sigma0 *Potential density anomaly referenced to 0 dbar*

---

### Description

This uses the 75-term density equation, and returns potential density referenced to a pressure of 0 dbar, minus 1000 kg/m^3.

### Usage

```
gsw_sigma0(SA, CT)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

potential density anomaly [ kg/m^3 ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_sigma0.html

### See Also

Other things related to density: gsw_CT_from_rho, gsw_CT_maxdensity, gsw_SA_from_rho, gsw_alpha_on_beta, gsw_alpha_wrt_t_exact, gsw_alpha_wrt_t_ice, gsw_alpha, gsw_beta_const_t_exact, gsw_beta, gsw_pot_rho_t_exact, gsw_rho_alpha_beta, gsw_rho_first_derivatives_wrt_enthalpy, gsw_rho_first_derivat: gsw_rho_ice, gsw_rho_t_exact, gsw_rho, gsw_sigma1, gsw_sigma2, gsw_sigma3, gsw_sigma4, gsw_specvol_alpha_beta, gsw_specvol_anom_standard, gsw_specvol_ice, gsw_specvol_t_exact, gsw_specvol

### Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
sigma0 <- gsw_sigma0(SA,CT)
expect_equal(sigma0, c(21.797900819337656, 22.052215404397316, 23.892985307893923,
                       26.667608665972011, 27.107380455119710, 27.409748977090885))
```

---

## gsw_sigma1 *Potential density anomaly referenced to 1000 dbar*

---

### Description

This uses the 75-term density equation, and returns potential density referenced to a pressure of 1000 dbar, minus 1000 kg/m^3.

### Usage

```
gsw_sigma1(SA, CT)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

potential density anomaly [ kg/m^3 ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_sigma1.html

## See Also

Other things related to density: gsw_CT_from_rho, gsw_CT_maxdensity, gsw_SA_from_rho, gsw_alpha_on_beta, gsw_alpha_wrt_t_exact, gsw_alpha_wrt_t_ice, gsw_alpha, gsw_beta_const_t_exact, gsw_beta, gsw_pot_rho_t_exact, gsw_rho_alpha_beta, gsw_rho_first_derivatives_wrt_enthalpy, gsw_rho_first_derivat gsw_rho_ice, gsw_rho_t_exact, gsw_rho, gsw_sigma0, gsw_sigma2, gsw_sigma3, gsw_sigma4, gsw_specvol_alpha_beta, gsw_specvol_anom_standard, gsw_specvol_ice, gsw_specvol_t_exact, gsw_specvol

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
sigma1 <- gsw_sigma1(SA,CT)
expect_equal(sigma1, c(25.955618850310202, 26.213131422420247, 28.125423775188438,
                       31.120360038882382, 31.637724222733368, 32.002453224572037))
```

---

gsw_sigma2 *Potential density anomaly referenced to 2000 dbar*

---

## Description

This uses the 75-term density equation, and returns potential density referenced to a pressure of 2000 dbar, minus 1000 kg/m^3.

## Usage

```
gsw_sigma2(SA, CT)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

potential density anomaly [ kg/m^3 ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_sigma2.html

### See Also

Other things related to density: gsw_CT_from_rho, gsw_CT_maxdensity, gsw_SA_from_rho, gsw_alpha_on_beta, gsw_alpha_wrt_t_exact, gsw_alpha_wrt_t_ice, gsw_alpha, gsw_beta_const_t_exact, gsw_beta, gsw_pot_rho_t_exact, gsw_rho_alpha_beta, gsw_rho_first_derivatives_wrt_enthalpy, gsw_rho_first_derivat gsw_rho_ice, gsw_rho_t_exact, gsw_rho, gsw_sigma0, gsw_sigma1, gsw_sigma3, gsw_sigma4, gsw_specvol_alpha_beta, gsw_specvol_anom_standard, gsw_specvol_ice, gsw_specvol_t_exact, gsw_specvol

### Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
sigma2 <- gsw_sigma2(SA,CT)
expect_equal(sigma2, c(30.023152223799116, 30.283783336283477, 32.265556840289719,
                       35.474550881051073, 36.067289438047737, 36.492606494879510))
```

---

## gsw_sigma3 *Potential density anomaly referenced to 3000 dbar*

---

### Description

This uses the 75-term density equation, and returns potential density referenced to a pressure of 3000 dbar, minus 1000 kg/m^3.

### Usage

```
gsw_sigma3(SA, CT)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

potential density anomaly with reference pressure 3000 dbar [ kg/m^3 ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_sigma3.html

## See Also

Other things related to density: gsw_CT_from_rho, gsw_CT_maxdensity, gsw_SA_from_rho, gsw_alpha_on_beta, gsw_alpha_wrt_t_exact, gsw_alpha_wrt_t_ice, gsw_alpha, gsw_beta_const_t_exact, gsw_beta, gsw_pot_rho_t_exact, gsw_rho_alpha_beta, gsw_rho_first_derivatives_wrt_enthalpy, gsw_rho_first_derivat: gsw_rho_ice, gsw_rho_t_exact, gsw_rho, gsw_sigma0, gsw_sigma1, gsw_sigma2, gsw_sigma4, gsw_specvol_alpha_beta, gsw_specvol_anom_standard, gsw_specvol_ice, gsw_specvol_t_exact, gsw_specvol

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
sigma3 <- gsw_sigma3(SA,CT)
expect_equal(sigma3, c(34.003747849903675, 34.267409891564057, 36.316415829697917,
                       39.732367693977039, 40.397934186745033, 40.881795690566832))
```

---

gsw_sigma4 *Potential density anomaly referenced to 4000 dbar*

---

## Description

This uses the 75-term density equation, and returns potential density referenced to a pressure of 4000 dbar, minus 1000 kg/m^3.

## Usage

```
gsw_sigma4(SA, CT)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

potential density anomaly with reference pressure 4000 dbar [ kg/m^3 ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_sigma4.html

### See Also

Other things related to density: gsw_CT_from_rho, gsw_CT_maxdensity, gsw_SA_from_rho, gsw_alpha_on_beta, gsw_alpha_wrt_t_exact, gsw_alpha_wrt_t_ice, gsw_alpha, gsw_beta_const_t_exact, gsw_beta, gsw_pot_rho_t_exact, gsw_rho_alpha_beta, gsw_rho_first_derivatives_wrt_enthalpy, gsw_rho_first_derivat gsw_rho_ice, gsw_rho_t_exact, gsw_rho, gsw_sigma0, gsw_sigma1, gsw_sigma2, gsw_sigma3, gsw_specvol_alpha_beta, gsw_specvol_anom_standard, gsw_specvol_ice, gsw_specvol_t_exact, gsw_specvol

### Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
sigma4 <- gsw_sigma4(SA,CT)
expect_equal(sigma4, c(37.900374609834898, 38.166979617032439, 40.280876075282549,
                       43.896091033421953, 44.631677245327637, 45.171817312020039))
```

---

## gsw_sound_speed *Sound speed*

---

### Description

Speed of sound in seawater, using the 75-term equation for specific volume.

### Usage

```
gsw_sound_speed(SA, CT, p)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

sound speed [ m/s ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_sound_speed.html

### See Also

Other things related to sound: gsw_sound_speed_ice, gsw_sound_speed_t_exact

### Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.7856, 28.4329, 22.8103, 10.2600,  6.8863,  4.4036)
p <- c(      10,      50,     125,     250,     600,    1000)
speed <- gsw_sound_speed(SA,CT,p)
expect_equal(speed/1e3, c(1.542426412426373, 1.542558891663385, 1.530801535436184,
                         1.494551099295314, 1.487622786765276, 1.484271672296205))
```

---

## gsw_sound_speed_ice *Sound speed in ice*

---

### Description

Speed of sound in ice.

### Usage

```
gsw_sound_speed_ice(t, p)
```

### Arguments

| | |
|---|---|
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

sound speed [ m/s ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_sound_speed_ice.html

## See Also

Other things related to sound: gsw_sound_speed_t_exact, gsw_sound_speed

## Examples

```
t <- c(-10.7856, -13.4329, -12.8103, -12.2600, -10.8863, -8.4036)
p <- c(      10,       50,      125,      250,      600,    1000)
speed <- gsw_sound_speed_ice(t, p)
expect_equal(speed/1e3, c(3.111311360346254, 3.116492565497544, 3.115833462003452,
                          3.115637032488204, 3.115377253092692, 3.113321384499191))
```

---

gsw_sound_speed_t_exact

*Sound Speed in Seawater*

---

## Description

Sound Speed in Seawater

## Usage

```
gsw_sound_speed_t_exact(SA, t, p)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

sound speed [ m/s ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_sound_speed_t_exact.html

## See Also

Other things related to sound: gsw_sound_speed_ice, gsw_sound_speed

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.7856, 28.4329, 22.8103, 10.2600,  6.8863,  4.4036)
p <- c(      10,      50,     125,     250,     600,    1000)
sound_speed <- gsw_sound_speed_t_exact(SA,CT,p)
expect_equal(sound_speed/1e3, c(1.542615803587414, 1.542703534065789, 1.530844979136360,
                                1.494409996920661, 1.487377102518027, 1.483934609078705))
```

---

gsw_specvol *Specific Volume of Seawater*

---

## Description

Specific Volume of Seawater

## Usage

```
gsw_specvol(SA, CT, p)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

Specific volume (1/density)

## References

http://www.teos-10.org/pubs/gsw/html/gsw_specvol.html

## See Also

Other things related to density: gsw_CT_from_rho, gsw_CT_maxdensity, gsw_SA_from_rho, gsw_alpha_on_beta, gsw_alpha_wrt_t_exact, gsw_alpha_wrt_t_ice, gsw_alpha, gsw_beta_const_t_exact, gsw_beta, gsw_pot_rho_t_exact, gsw_rho_alpha_beta, gsw_rho_first_derivatives_wrt_enthalpy, gsw_rho_first_derivat gsw_rho_ice, gsw_rho_t_exact, gsw_rho, gsw_sigma0, gsw_sigma1, gsw_sigma2, gsw_sigma3, gsw_sigma4, gsw_specvol_alpha_beta, gsw_specvol_anom_standard, gsw_specvol_ice, gsw_specvol_t_exact

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
p <- c(      10,      50,     125,     250,     600,    1000)
specvol <- gsw_specvol(SA, CT, p)
expect_equal(specvol*1e3, c(0.978626852431313, 0.978222365701325, 0.976155264597929,
                            0.972961258011157, 0.971026719344908, 0.968989944622149))
```

---

gsw_specvol_alpha_beta

*Specific Volume, alpha, and beta*

---

## Description

Specific Volume, alpha, and beta

## Usage

```
gsw_specvol_alpha_beta(SA, CT, p)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

a list holding specvol, the specific volume [ m^3/kg ], alpha, the thermal expansion coefficient [ 1/degC ], and beta, the haline contraction coefficient [ kg/g ].

## References

http://www.teos-10.org/pubs/gsw/html/gsw_specvol_alpha_beta.html

## See Also

Other things related to density: gsw_CT_from_rho, gsw_CT_maxdensity, gsw_SA_from_rho, gsw_alpha_on_beta, gsw_alpha_wrt_t_exact, gsw_alpha_wrt_t_ice, gsw_alpha, gsw_beta_const_t_exact, gsw_beta, gsw_pot_rho_t_exact, gsw_rho_alpha_beta, gsw_rho_first_derivatives_wrt_enthalpy, gsw_rho_first_derivat gsw_rho_ice, gsw_rho_t_exact, gsw_rho, gsw_sigma0, gsw_sigma1, gsw_sigma2, gsw_sigma3, gsw_sigma4, gsw_specvol_anom_standard, gsw_specvol_ice, gsw_specvol_t_exact, gsw_specvol

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
p <- c(      10,      50,     125,     250,     600,    1000)
r <- gsw_specvol_alpha_beta(SA, CT, p)
expect_equal(r$specvol/1e-3, c(0.978626852431313, 0.978222365701325, 0.976155264597929,
                              0.972961258011157, 0.971026719344908, 0.968989944622149))
expect_equal(r$alpha/1e-3, c(0.324638934509245, 0.322655537959731, 0.281145723210171,
                            0.173199716344780, 0.146289673594824, 0.129414845334599))
expect_equal(r$beta/1e-3, c(0.717483987596135, 0.717647512290095, 0.726211643644768,
                           0.750500751749777, 0.755052064788492, 0.757050813384370))
```

---

# gsw_specvol_anom_standard

*Specific volume anomaly [standard] (75-term equation)*

---

## Description

Note that the TEOS function named specific_volume_anomaly is not provided in the C library, so it is not provided in R, either.

## Usage

```
gsw_specvol_anom_standard(SA, CT, p)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

Specific volume anomaly [ m^3/kg ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_specvol_anom_standard.html

### See Also

Other things related to density: gsw_CT_from_rho, gsw_CT_maxdensity, gsw_SA_from_rho, gsw_alpha_on_beta, gsw_alpha_wrt_t_exact, gsw_alpha_wrt_t_ice, gsw_alpha, gsw_beta_const_t_exact, gsw_beta, gsw_pot_rho_t_exact, gsw_rho_alpha_beta, gsw_rho_first_derivatives_wrt_enthalpy, gsw_rho_first_derivat: gsw_rho_ice, gsw_rho_t_exact, gsw_rho, gsw_sigma0, gsw_sigma1, gsw_sigma2, gsw_sigma3, gsw_sigma4, gsw_specvol_alpha_beta, gsw_specvol_ice, gsw_specvol_t_exact, gsw_specvol

### Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
p <- c(      10,      50,     125,     250,     600,    1000)
a <- gsw_specvol_anom_standard(SA, CT, p)
expect_equal(a*1e5, c(0.601051894897400, 0.578609769250563, 0.405600538950092,
                      0.142190453761838, 0.104335535578967, 0.076383389577725))
```

---

gsw_specvol_first_derivatives

*First Derivatives of Specific Volume*

---

### Description

First Derivatives of Specific Volume

### Usage

```
gsw_specvol_first_derivatives(SA, CT, p)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

A list containing v_SA [ (m^3/kg)/(g/kg) ], the derivative of specific volume with respect to Absolute Salinity, v_CT [ (m^3/kg)/degC], the derivative of specific volume with respect to Conservative Temperature, and v_p [ (m^3/kg)/dbar ], the derivative of specific volume with respect to pressure. (Note that the last quantity is denoted v_P in the documentation for the Matlab function.)

## References

http://www.teos-10.org/pubs/gsw/html/gsw_specvol_first_derivatives.html

## See Also

Other things related to enthalpy: gsw_CT_from_enthalpy, gsw_dynamic_enthalpy, gsw_enthalpy_CT_exact, gsw_enthalpy_diff, gsw_enthalpy_first_derivatives_CT_exact, gsw_enthalpy_first_derivatives, gsw_enthalpy_ice, gsw_enthalpy_t_exact, gsw_enthalpy, gsw_frazil_properties_potential_poly, gsw_frazil_properties_potential, gsw_pot_enthalpy_from_pt_ice_poly, gsw_pot_enthalpy_from_pt_ice, gsw_pot_enthalpy_ice_freezing_poly, gsw_pot_enthalpy_ice_freezing, gsw_pt_from_pot_enthalpy_ice_poly, gsw_pt_from_pot_enthalpy_ice, gsw_specvol_first_derivatives_wrt_enthalpy

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
p <- c(      10,      50,     125,     250,     600,    1000)
r <- gsw_specvol_first_derivatives(SA, CT, p)
expect_equal(r$v_SA/1e-6, c(-0.702149096451073, -0.702018847212088, -0.708895319156155,
                            -0.730208155560782, -0.733175729406169, -0.733574625737474))
expect_equal(r$v_CT/1e-6, c(0.317700378655437, 0.315628863649601, 0.274441877830800,
                            0.168516613901993, 0.142051181824820, 0.125401683814057))
expect_equal(r$v_p/1e-12, c(-0.402527990904794, -0.402146232553089, -0.406663124765787,
                            -0.423877042622481, -0.426198431093548, -0.426390351853055))
```

gsw_specvol_first_derivatives_wrt_enthalpy
*First Derivatives of Specific Volume wrt Enthalpy*

## Description

First Derivatives of Specific Volume wrt Enthalpy

## Usage

```
gsw_specvol_first_derivatives_wrt_enthalpy(SA, CT, p)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

A list containing v_SA_wrt_h and v_h.

## References

http://www.teos-10.org/pubs/gsw/html/gsw_specvol_first_derivatives_wrt_enthalpy.html

## See Also

Other things related to enthalpy: gsw_CT_from_enthalpy, gsw_dynamic_enthalpy, gsw_enthalpy_CT_exact, gsw_enthalpy_diff, gsw_enthalpy_first_derivatives_CT_exact, gsw_enthalpy_first_derivatives, gsw_enthalpy_ice, gsw_enthalpy_t_exact, gsw_enthalpy, gsw_frazil_properties_potential_poly, gsw_frazil_properties_potential, gsw_pot_enthalpy_from_pt_ice_poly, gsw_pot_enthalpy_from_pt_ice, gsw_pot_enthalpy_ice_freezing_poly, gsw_pot_enthalpy_ice_freezing, gsw_pt_from_pot_enthalpy_ice_poly, gsw_pt_from_pot_enthalpy_ice, gsw_specvol_first_derivatives

### Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
p <- c(      10,      50,     125,     250,     600,    1000)
r <- gsw_specvol_first_derivatives_wrt_enthalpy(SA, CT, p)
expect_equal(r$v_SA_wrt_h/1e-6, c(-0.702143511679586, -0.701991101310494, -0.708834353735310,
                                  -0.730130919555592, -0.733018321892082, -0.733342002723321))
expect_equal(r$v_h/1e-10, c(0.795862623587769, 0.790648383268264, 0.687443468257647,
                            0.422105846942233, 0.355778874334799, 0.314053366403993))
```

---

## gsw_specvol_ice    *Specific Volume of Ice*

---

### Description

Specific Volume of Ice

### Usage

```
gsw_specvol_ice(t, p)
```

### Arguments

| | |
|---|---|
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

Specific volume [ m^3/kg ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_specvol_ice.html

### See Also

Other things related to density: gsw_CT_from_rho, gsw_CT_maxdensity, gsw_SA_from_rho, gsw_alpha_on_beta, gsw_alpha_wrt_t_exact, gsw_alpha_wrt_t_ice, gsw_alpha, gsw_beta_const_t_exact, gsw_beta, gsw_pot_rho_t_exact, gsw_rho_alpha_beta, gsw_rho_first_derivatives_wrt_enthalpy, gsw_rho_first_derivat gsw_rho_ice, gsw_rho_t_exact, gsw_rho, gsw_sigma0, gsw_sigma1, gsw_sigma2, gsw_sigma3, gsw_sigma4, gsw_specvol_alpha_beta, gsw_specvol_anom_standard, gsw_specvol_t_exact, gsw_specvol

### Examples

```
t <- c(-10.7856, -13.4329, -12.8103, -12.2600,  -10.8863,  -8.4036)
p <- c(      10,       50,      125,      250,       600,     1000)
v <- gsw_specvol_ice(t, p)
expect_equal(v, c(0.001088982980677, 0.001088489459509, 0.001088499019939,
                0.001088433747301, 0.001088223220685, 0.001088135464776))
```

---

## gsw_specvol_second_derivatives

*Second Derivatives of Specific Volume*

---

### Description

Second Derivatives of Specific Volume

### Usage

```
gsw_specvol_second_derivatives(SA, CT, p)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

A list containing `specvol_SA_SA` [ (m^3/kg)/(g/kg)^2 ], the second derivative of specific volume with respect to Absolute Salinity, `specvol_SA_CT` [ (m^3/kg)/(g/kg)/degC ], the derivative of specific volume with respect to Absolute Salinity and Conservative Temperature, `specvol_CT_CT` [ (m^3/kg)/degC^2 ], the second derivative of specific volume with respect to Conservative Temperature, `specvol_SA_p` [ (m^3/kg)/(g/kg)/dbar ], the derivative of specific volume with respect to Absolute Salinity and pressure, and `specvol_CT_p` [ (m^3/kg)/K/dbar ], the derivative of specific volume with respect to Conservative Temperature and pressure.

### References

http://www.teos-10.org/pubs/gsw/html/gsw_specvol_second_derivatives.html

### Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.7856, 28.4329, 22.8103, 10.2600,  6.8863,  4.4036)
p <- c(      10,      50,     125,     250,     600,    1000)
r <- gsw_specvol_second_derivatives(SA, CT, p)
expect_equal(r$specvol_SA_SA/1e-8, c(0.080906777599140, 0.080915086639384, 0.084568844270812,
                                     0.096725108896007, 0.099111765836648, 0.100302277946072))
expect_equal(r$specvol_SA_CT/1e-8, c(0.129965332117084, 0.130523053162130, 0.149555815430615,
                                     0.217023290441810, 0.233892039070486, 0.243659989480325))
expect_equal(r$specvol_CT_CT/1e-7, c(0.071409582006642, 0.071582962051991, 0.077436153664104,
                                     0.095329736274850, 0.100105336953738, 0.103044572835472))
expect_equal(r$specvol_SA_p/1e-14, c(0.141281359467752, 0.141507584673426, 0.147247234588907,
                                     0.164580347761218, 0.168069801298412, 0.169948275518754))
expect_equal(r$specvol_CT_p/1e-14, c(0.085542828707964, 0.086723632576213, 0.112156562396990,
                                     0.188269893599500, 0.211615556759369, 0.228609575049911))
```

---

`gsw_specvol_second_derivatives_wrt_enthalpy`

*Second Derivatives of Specific Volume wrt Enthalpy*

---

### Description

Second Derivatives of Specific Volume wrt Enthalpy

### Usage

```
gsw_specvol_second_derivatives_wrt_enthalpy(SA, CT, p)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

A list containing specvol_SA_SA [ (m^3/kg)/(g/kg)^2 ], the second derivative of specific volume with respect to Absolute Salinity, specvol_SA_h [ (m^3/kg)/(g/kg)/(J/kg) ], the derivative of specific volume with respect to Absolute Salinity and enthalpy, and specvol_h_h [ (m^3/kg)/(J/kg)^2 ], the second derivative of specific volume with respect to enthalpy.

## Bugs

As of March 27, 2017, the test values listed in "Examples" do not match values provided at the TEOS-10 website listed in "References", but they match with values given by the Matlab code that is provided on the TEOS-10 website. It is expected that the TEOS-10 website will be updated by May 2017. As those updates to the TEOS-10 website become available, the present comment will be revised or deleted.

## References

http://www.teos-10.org/pubs/gsw/html/gsw_specvol_second_derivatives_wrt_enthalpy.html

## See Also

Other functions with suspicious test values on the TEOS-10 website: gsw_entropy_second_derivatives, gsw_rho_second_derivatives_wrt_enthalpy, gsw_t_freezing_first_derivatives_poly

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
p <- c(      10,      50,     125,     250,     600,    1000)
r <- gsw_specvol_second_derivatives_wrt_enthalpy(SA, CT, p)
expect_equal(r$specvol_SA_SA/1e-8, c(0.080900028996264, 0.080937999675000, 0.084663065647101,
                              0.096973364985384, 0.099727453432293, 0.101353037979356))
expect_equal(r$specvol_SA_h/1e-12, c(0.325437133570796, 0.327060462851431, 0.375273569184178,
                              0.545188833073084, 0.589424881889351, 0.616101548209175))
expect_equal(r$specvol_h_h/1e-15, c(0.447949998681476, 0.449121446914278, 0.485998151346315,
                              0.598480711660961, 0.628708349875318, 0.647433212216398))
```

gsw_specvol_t_exact *Specific Volume of Seawater*

## Description

Specific Volume of Seawater

## Usage

```
gsw_specvol_t_exact(SA, t, p)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

Specific volume [ m^3/kg ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_specvol_t_exact.html

## See Also

Other things related to density: gsw_CT_from_rho, gsw_CT_maxdensity, gsw_SA_from_rho, gsw_alpha_on_beta, gsw_alpha_wrt_t_exact, gsw_alpha_wrt_t_ice, gsw_alpha, gsw_beta_const_t_exact, gsw_beta, gsw_pot_rho_t_exact, gsw_rho_alpha_beta, gsw_rho_first_derivatives_wrt_enthalpy, gsw_rho_first_derivat gsw_rho_ice, gsw_rho_t_exact, gsw_rho, gsw_sigma0, gsw_sigma1, gsw_sigma2, gsw_sigma3, gsw_sigma4, gsw_specvol_alpha_beta, gsw_specvol_anom_standard, gsw_specvol_ice, gsw_specvol

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
t <- c( 28.7856, 28.4329, 22.8103, 10.2600,  6.8863,  4.4036)
p <- c(      10,      50,     125,     250,     600,    1000)
v <- gsw_specvol_t_exact(SA, t, p)
expect_equal(v*1e3, c(0.978626625025472, 0.978222143734527, 0.976154768597586,
                      0.972961211575438, 0.971026779948624, 0.968989990731808))
```

---

gsw_spiciness0 *Seawater Spiciness at p=0 dbar*

---

## Description

Calculate seawater spiciness referenced to 0 dbar (i.e. the surface).

## Usage

```
gsw_spiciness0(SA, CT)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

spiciness [ kg/m^3 ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_spiciness0.html

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
spiciness <- gsw_spiciness0(SA, CT)
expect_equal(spiciness, c(5.728998558542941, 5.749940496782486, 4.163547112671111,
                          1.069362556641764, 0.426428274444305, 0.089725188494086))
```

gsw_spiciness1 *Seawater Spiciness at p=1000 dbar*

## Description

Calculate seawater spiciness referenced to 1000 dbar.

## Usage

```
gsw_spiciness1(SA, CT)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

spiciness [ kg/m^3 ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_spiciness1.html

## See Also

Other things related to spiciness: gsw_spiciness2

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
spiciness <- gsw_spiciness1(SA, CT)
expect_equal(spiciness, c(6.311038322123224, 6.326411175472160, 4.667218659743284,
                          1.351722468726905, 0.628494082166029, 0.224779784908478))
```

gsw_spiciness2 *Seawater Spiciness at p=2000 dbar*

## Description

Calculate seawater spiciness referenced to 2000 dbar.

## Usage

```
gsw_spiciness2(SA, CT)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

spiciness [ kg/m^3 ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_spiciness2.html

## See Also

Other things related to spiciness: gsw_spiciness1

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
spiciness <- gsw_spiciness2(SA, CT)
expect_equal(spiciness, c(6.874671751873180, 6.884616399155135, 5.154458892387083,
                          1.624327800598636, 0.823490797424952, 0.355069307641827))
```

gsw_SP_from_C *Convert from Electrical Conductivity to Practical Salinity*

## Description

Convert from Electrical Conductivity to Practical Salinity

## Usage

```
gsw_SP_from_C(C, t, p)
```

## Arguments

| | |
|---|---|
| C | conductivity [ mS/cm ] |
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

Practical Salinity (PSS-78) [ unitless ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_SP_from_C.html

## See Also

Other things related to salinity: gsw_C_from_SP, gsw_SA_from_SP_Baltic, gsw_SA_from_SP, gsw_SA_from_Sstar, gsw_SP_from_SA, gsw_SP_from_SK, gsw_SP_from_SR, gsw_SP_from_Sstar, gsw_SR_from_SP, gsw_Sstar_from_SA, gsw_Sstar_from_SP, gsw_deltaSA_from_SP

Other things related to conductivity: gsw_C_from_SP

## Examples

```
C <- c(34.5487, 34.7275, 34.8605, 34.6810, 34.5680, 34.5600)
t <- c(28.7856, 28.4329, 22.8103, 10.2600,  6.8863,  4.4036)
p <- c(     10,      50,     125,     250,     600,    1000)
SP <- gsw_SP_from_C(C,t,p)
expect_equal(SP,  c(20.009869599086951, 20.265511864874270, 22.981513062527689,
                    31.204503263727982, 34.032315787432829, 36.400308494388170))
```

---

# gsw_SP_from_SA — *Convert from Absolute Salinity to Practical Salinity*

---

## Description

Calculate Practical Salinity from Absolute Salinity, pressure, longitude, and latitude.

## Usage

```
gsw_SP_from_SA(SA, p, longitude, latitude)
```

## Arguments

| | |
|---|---|
| `SA` | Absolute Salinity [ g/kg ] |
| `p` | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| `longitude` | longitude in decimal degrees, positive to the east of Greenwich. (This is called `long` in the TEOS-10 Matlab code.) |
| `latitude` | latitude in decimal degrees, positive to the north of the equator. (This is called `lat` in the TEOS-10 Matlab code.) |

## Details

If SP is a matrix and if its dimensions correspond to the lengths of longitude and latitude, then the latter are converted to analogous matrices with `expand.grid`.

Note: unlike the corresponding Matlab function, this does not return a flag indicating whether the location is in the ocean.

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at `https://github.com/TEOS-10/GSW-C`, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (`https://github.com/TEOS-10/GSW-Fortran`) which in turn stems from the GSW-Matlab system (`https://github.com/TEOS-10/GSW-Matlab`). Consult `http://www.teos-10.org` to learn more about these software systems, their authorships, and the science behind it all.

## Value

Practical Salinity (PSS-78) [ unitless ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_SP_from_SA.html

### See Also

Other things related to salinity: gsw_C_from_SP, gsw_SA_from_SP_Baltic, gsw_SA_from_SP, gsw_SA_from_Sstar, gsw_SP_from_C, gsw_SP_from_SK, gsw_SP_from_SR, gsw_SP_from_Sstar, gsw_SR_from_SP, gsw_Sstar_from_SA, gsw_Sstar_from_SP, gsw_deltaSA_from_SP

### Examples

```
SA <-   c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
p <-    c(     10,      50,     125,     250,     600,    1000)
lat <-  c(      4,       4,       4,       4,       4,       4)
long <- c(    188,     188,     188,     188,     188,     188)
SP <- gsw_SP_from_SA(SA,p,long,lat)
expect_equal(SP, c(34.548721553448317, 34.727477488096639, 34.860554877708005,
                   34.680971112271791, 34.567971663653388, 34.560036751118204))
```

---

## gsw_SP_from_SK *Calculate Practical Salinity from Knudsen Salinity*

---

### Description

Calculate Practical Salinity from Knudsen Salinity

### Usage

```
gsw_SP_from_SK(SK)
```

### Arguments

| | |
|---|---|
| SK | Knudsen Salinity [ parts per thousand, ppt ] |

### Value

Practical Salinity (PSS-78) [ unitless ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_SP_from_SK.html

### See Also

Other things related to salinity: gsw_C_from_SP, gsw_SA_from_SP_Baltic, gsw_SA_from_SP, gsw_SA_from_Sstar, gsw_SP_from_C, gsw_SP_from_SA, gsw_SP_from_SR, gsw_SP_from_Sstar, gsw_SR_from_SP, gsw_Sstar_from_SA, gsw_Sstar_from_SP, gsw_deltaSA_from_SP

### Examples

```
SK <- c(34.5487, 34.7275, 34.8605, 34.6810, 34.5680, 34.5600)
SP <- gsw_SP_from_SK(SK)
expect_equal(SP, c(34.548342096952908, 34.727295637119113, 34.860409847645435,
                   34.680755706371187, 34.567658670360110, 34.559651800554022))
```

---

## gsw_SP_from_SR — *Calculate Practical Salinity from Reference Salinity*

---

### Description

Calculate Practical Salinity from Reference Salinity

### Usage

```
gsw_SP_from_SR(SR)
```

### Arguments

| | |
|---|---|
| SR | Reference Salinity [ g/kg ] |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

Practical Salinity (PSS-78) [ unitless ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_SP_from_SR.html

### See Also

Other things related to salinity: gsw_C_from_SP, gsw_SA_from_SP_Baltic, gsw_SA_from_SP, gsw_SA_from_Sstar, gsw_SP_from_C, gsw_SP_from_SA, gsw_SP_from_SK, gsw_SP_from_Sstar, gsw_SR_from_SP, gsw_Sstar_from_SA, gsw_Sstar_from_SP, gsw_deltaSA_from_SP

### Examples

```
SR <- c(34.5487, 34.7275, 34.8605, 34.6810, 34.5680, 34.5600)
SP <- gsw_SP_from_SR(SR)
expect_equal(SP, c(34.386552667080714, 34.564513505458834, 34.696889296869848,
                   34.518231743800094, 34.405762086435850, 34.397799632817147))
```

---

## gsw_SP_from_Sstar *Practical Salinity from Preformed Salinity*

---

### Description

Practical Salinity from Preformed Salinity

### Usage

```
gsw_SP_from_Sstar(Sstar, p, longitude, latitude)
```

### Arguments

| | |
|---|---|
| `Sstar` | Preformed Salinity [ g/kg ] |
| `p` | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| `longitude` | longitude in decimal degrees, positive to the east of Greenwich. (This is called `long` in the TEOS-10 Matlab code.) |
| `latitude` | latitude in decimal degrees, positive to the north of the equator. (This is called `lat` in the TEOS-10 Matlab code.) |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

Practical Salinity (PSS-78) [ unitless ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_SP_from_Sstar.html

### See Also

Other things related to salinity: gsw_C_from_SP, gsw_SA_from_SP_Baltic, gsw_SA_from_SP, gsw_SA_from_Sstar, gsw_SP_from_C, gsw_SP_from_SA, gsw_SP_from_SK, gsw_SP_from_SR, gsw_SR_from_SP, gsw_Sstar_from_SA, gsw_Sstar_from_SP, gsw_deltaSA_from_SP

### Examples

```
Sstar <- c(34.7115, 34.8912, 35.0247, 34.8436, 34.7291, 34.7197)
p <- c(         10,      50,     125,     250,     600,    1000)
longitude <- 188
latitude <- 4
SP <- gsw_SP_from_Sstar(Sstar, p, longitude, latitude)
expect_equal(SP, c(34.548646570969929, 34.727538423586189, 34.860549501859502,
                   34.681006826476434, 34.568065697992346, 34.560023926979518))
```

---

gsw_SR_from_SP *Calculate Reference Salinity from Practical Salinity*

---

### Description

Calculate Reference Salinity from Practical Salinity

### Usage

```
gsw_SR_from_SP(SP)
```

### Arguments

| | |
|---|---|
| SP | Practical Salinity (PSS-78) [ unitless ] |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

Reference Salinity [ g/kg ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_SR_from_SP.html

### See Also

Other things related to salinity: gsw_C_from_SP, gsw_SA_from_SP_Baltic, gsw_SA_from_SP, gsw_SA_from_Sstar, gsw_SP_from_C, gsw_SP_from_SA, gsw_SP_from_SK, gsw_SP_from_SR, gsw_SP_from_Sstar, gsw_Sstar_from_SA, gsw_Sstar_from_SP, gsw_deltaSA_from_SP

## Examples

```
SP <- c(34.5487, 34.7275, 34.8605, 34.6810, 34.5680, 34.5600)
SR <- gsw_SR_from_SP(SP)
expect_equal(SR, c(34.711611927085727, 34.891255045714303, 35.024882197714305,
                   34.844535778285724, 34.731002934857159, 34.722965211428587))
```

---

gsw_Sstar_from_SA *Convert from Absolute Salinity to Preformed Salinity*

---

## Description

Calculate Preformed Salinity from Absolute Salinity, pressure, longitude, and latitude.

## Usage

```
gsw_Sstar_from_SA(SA, p, longitude, latitude)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| longitude | longitude in decimal degrees, positive to the east of Greenwich. (This is called `long` in the TEOS-10 Matlab code.) |
| latitude | latitude in decimal degrees, positive to the north of the equator. (This is called `lat` in the TEOS-10 Matlab code.) |

## Details

If SA is a matrix and if its dimensions correspond to the lengths of longitude and latitude, then the latter are converted to analogous matrices with `expand.grid`.

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

Preformed Salinity [ g/kg ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_Sstar_from_SA.html

## See Also

Other things related to salinity: gsw_C_from_SP, gsw_SA_from_SP_Baltic, gsw_SA_from_SP, gsw_SA_from_Sstar, gsw_SP_from_C, gsw_SP_from_SA, gsw_SP_from_SK, gsw_SP_from_SR, gsw_SP_from_Sstar, gsw_SR_from_SP, gsw_Sstar_from_SP, gsw_deltaSA_from_SP

## Examples

```
SA <-   c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
p <-    c(     10,      50,     125,     250,     600,    1000)
lat <-  c(      4,       4,       4,       4,       4,       4)
long <- c(    188,     188,     188,     188,     188,     188)
Sstar <- gsw_Sstar_from_SA(SA,p,long,lat)
expect_equal(Sstar, c(34.711575335926490, 34.891138777337822, 35.024705401162166,
                      34.843564118358302, 34.729005527604883, 34.719712883389462))
```

---

# gsw_Sstar_from_SP *Convert from Practical Salinity to Preformed Salinity*

---

## Description

Calculate Preformed Salinity from Practical Salinity, pressure, longitude, and latitude.

## Usage

```
gsw_Sstar_from_SP(SP, p, longitude, latitude)
```

## Arguments

| | |
|---|---|
| SP | Practical Salinity (PSS-78) [ unitless ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| longitude | longitude in decimal degrees, positive to the east of Greenwich. (This is called `long` in the TEOS-10 Matlab code.) |
| latitude | latitude in decimal degrees, positive to the north of the equator. (This is called `lat` in the TEOS-10 Matlab code.) |

## Details

If SP is a matrix and if its dimensions correspond to the lengths of longitude and latitude, then the latter are converted to analogous matrices with `expand.grid`.

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

Preformed Salinity [ g/kg ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_Sstar_from_SP.html

## See Also

Other things related to salinity: gsw_C_from_SP, gsw_SA_from_SP_Baltic, gsw_SA_from_SP, gsw_SA_from_Sstar, gsw_SP_from_C, gsw_SP_from_SA, gsw_SP_from_SK, gsw_SP_from_SR, gsw_SP_from_Sstar, gsw_SR_from_SP, gsw_Sstar_from_SA, gsw_deltaSA_from_SP

## Examples

```
SP <-   c(34.5487, 34.7275, 34.8605, 34.6810, 34.5680, 34.5600)
p <-    c(     10,      50,     125,     250,     600,    1000)
lat <-  c(      4,       4,       4,       4,       4,       4)
long <- c(    188,     188,     188,     188,     188,     188)
Sstar <- gsw_Sstar_from_SP(SP,p,long,lat)
expect_equal(Sstar, c(34.711553680880769, 34.891161395333754, 35.024650265047370,
                      34.843593141519356, 34.729033995955525, 34.719675962471783))
```

---

`gsw_thermobaric` *Thermobaric coefficient (75-term equation)*

---

## Description

Thermobaric coefficient (75-term equation)

## Usage

```
gsw_thermobaric(SA, CT, p)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

thermobaric coefficient wrt Conservative Temperature [ 1/(K Pa) ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_thermobaric.html

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
p <-  c(     10,      50,     125,     250,     600,    1000)
tb <- gsw_thermobaric(SA, CT, p)
expect_equal(tb*1e11, c(0.152618598186650, 0.153662896162852, 0.173429325875738,
                      0.232810160208414, 0.251984724005424, 0.266660342289558))
```

---

`gsw_Turner_Rsubrho` *Turner Angle and Density Ratio*

---

## Description

This uses the 75-term density equation. The values of Turner Angle Tu and density ratio Rrho are calculated at mid-point pressures, `p_mid`.

## Usage

```
gsw_Turner_Rsubrho(SA, CT, p)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

List containing `Tu` [ degrees ], `Rsubrho` [ unitless ], and `p_mid` [ dbar ]

## References

http://www.teos-10.org/pubs/gsw/html/gsw_Turner_Rsubrho.html

## Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
p <-  c(     10,      50,     125,     250,     600,    1000)
r <- gsw_Turner_Rsubrho(SA, CT, p)
expect_equal(r$Tu, c(-2.063858905281147, 41.758435216784427, 47.606966981687535,
                     53.710351151706369, 45.527063858211527))
expect_equal(r$Rsubrho, 100*c(-0.009304335069039, -0.176564834348709, 0.219627771740757,
                              0.065271424662002, 1.087044054679743))
expect_equal(r$p_mid, 100*c(0.300, 0.875, 1.875, 4.250, 8.000))
```

---

gsw_t_deriv_chem_potential_water_t_exact
*Derivative of Chemical Potential of Water in Seawater wrt Temperature*

---

## Description

Derivative of Chemical Potential of Water in Seawater wrt Temperature

## Usage

```
gsw_t_deriv_chem_potential_water_t_exact(SA, t, p)
```

## Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| t | in-situ temperature (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

## Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

## Value

derivative [ J/(g*degC) ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_t_deriv_chem_potential_water_t_exact.html

### Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
t <- c( 28.7856, 28.4329, 22.8103, 10.2600,  6.8863,  4.4036)
p <- c(      10,      50,     125,     250,     600,    1000)
d <- gsw_t_deriv_chem_potential_water_t_exact(SA, t, p)
expect_equal(d, c(-0.428798278908442, -0.423860344327343, -0.345277821010421,
                -0.164446485487145, -0.114228046736087, -0.076990819658255))
```

---

gsw_t_freezing *Freezing Temperature of Seawater*

---

### Description

This uses the C function named gsw_t_freezing_exact, because the C function named gsw_t_freezing does not produce check values that match the Matlab function called gsw_t_freezing (see references for those test values).

### Usage

```
gsw_t_freezing(SA, p, saturation_fraction = 1)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| saturation_fraction | fraction of air in water [unitless] |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

in-situ freezing temperature (ITS-90) [ degC ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_t_freezing.html

### Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
p <- c(      10,      50,     125,     250,     600,    1000)
saturation_fraction <- 1
tf <- gsw_t_freezing(SA, p, saturation_fraction)
expect_equal(tf, c(-1.902730710149803, -1.942908619287183, -2.006861069199743,
                   -2.090985086875259, -2.351293130342102, -2.660498762776720))
```

---

## gsw_t_freezing_first_derivatives

*Derivatives of Freezing Water Properties*

---

### Description

Derivatives of Freezing Water Properties

### Usage

```
gsw_t_freezing_first_derivatives(SA, p, saturation_fraction = 1)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| saturation_fraction | fraction of air in water [unitless] |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

a list containing `tfreezing_SA` [ K/(g/kg) ], the derivative of freezing temperature with Absolute Salinity and `tfreezing_p` [ K/dbar ], the derivative with respect to pressure.

### References

http://www.teos-10.org/pubs/gsw/html/gsw_t_freezing_first_derivatives.html

### Examples

```
SA <- c(               34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
p <- c(                     10,      50,     125,     250,     600,    1000)
saturation_fraction <- c(    1,     0.8,     0.6,     0.5,     0.4,       0)
derivs <- gsw_t_freezing_first_derivatives(SA, p, saturation_fraction)
expect_equal(derivs$tfreezing_SA, c(-0.056811800705787, -0.056856999671114, -0.056903079789292,
                            -0.056904020028541, -0.056974588411844, -0.057082363270642))
expect_equal(derivs$tfreezing_p/1e-7, c(-0.748468312442338, -0.749793159537290, -0.752225023995510,
                              -0.756170965034610, -0.767279572670040, -0.779936552091913))
```

---

## gsw_t_freezing_first_derivatives_poly

*Derivatives of Freezing Water Properties (Polynomial version)*

---

### Description

Derivatives of Freezing Water Properties (Polynomial version)

### Usage

```
gsw_t_freezing_first_derivatives_poly(SA, p, saturation_fraction = 1)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| saturation_fraction | fraction of air in water [unitless] |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

a list containing `tfreezing_SA` [ K/(g/kg) ], the derivative of freezing temperature with Absolute Salinity and `tfreezing_p` [ K/dbar ], the derivative with respect to pressure.

### Bugs

As of March 27, 2017, the test values listed in "Examples" do not match values provided at the TEOS-10 website listed in "References", but they match with values given by the Matlab code that is provided on the TEOS-10 website. It is expected that the TEOS-10 website will be updated by May 2017. As those updates to the TEOS-10 website become available, the present comment will be revised or deleted.

### References

http://www.teos-10.org/pubs/gsw/html/gsw_t_freezing_first_derivatives.html

### See Also

Other functions with suspicious test values on the TEOS-10 website: gsw_entropy_second_derivatives, gsw_rho_second_derivatives_wrt_enthalpy, gsw_specvol_second_derivatives_wrt_enthalpy

### Examples

```
SA <- c(                34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
p <- c(                      10,      50,     125,     250,     600,    1000)
saturation_fraction <- c(     1,     0.8,     0.6,     0.5,     0.4,       0)
derivs <- gsw_t_freezing_first_derivatives_poly(SA, p, saturation_fraction)
expect_equal(derivs$tfreezing_SA, c(-0.056810211094078, -0.056855567524973, -0.056901968693345,
                             -0.056903498206432, -0.056975157476629, -0.057083526206200))
expect_equal(derivs$tfreezing_p/1e-7, c(-0.748987354878138, -0.750288853857513, -0.752676389629787,
                               -0.756549680608529, -0.767482625710990, -0.779985619685683))
```

---

gsw_t_from_CT *In situ temperature from Conservative Temperature*

---

### Description

In situ temperature from Conservative Temperature

### Usage

```
gsw_t_from_CT(SA, CT, p)
```

### Arguments

| | |
|---|---|
| SA | Absolute Salinity [ g/kg ] |
| CT | Conservative Temperature [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

in-situ temperature (ITS-90) [ degC ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_t_from_CT.html

### Examples

```
SA <- c(34.7118, 34.8915, 35.0256, 34.8472, 34.7366, 34.7324)
CT <- c(28.8099, 28.4392, 22.7862, 10.2262,  6.8272,  4.3236)
p <-  c(     10,      50,     125,     250,     600,    1000)
t <- gsw_t_from_CT(SA, CT, p)
expect_equal(t, c(28.785580227725703, 28.432872246163946, 22.810323087627076,
                  10.260010752788906, 6.886286301029376, 4.403624452383043))
```

---

## gsw_t_from_pt0_ice *In situ Temperature from Potential Temperature at 0dbar*

---

### Description

In situ Temperature from Potential Temperature at 0dbar

### Usage

```
gsw_t_from_pt0_ice(pt0_ice, p)
```

### Arguments

| | |
|---|---|
| pt0_ice | potential temperature of ice (ITS-90) [ degC ] |
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

in-situ temperature (ITS-90) [ degC ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_t_from_pt0_ice.html

### Examples

```
pt0_ice  <- c(-10.7856, -13.4329, -12.8103, -12.2600, -10.8863, -8.4036)
p <-  c(     10,      50,     125,     250,     600,    1000)
t <- gsw_t_from_pt0_ice(pt0_ice, p)
expect_equal(t, c(-10.783412084414074, -13.422068638139141, -12.783170223330448,
                -12.205667526492039, -10.755496924674144, -8.184121042593350))
```

---

`gsw_z_from_p` *Height from Pressure*

---

### Description

Computation of height (above sea level) from pressure, using the 75-term equation for specific volume.

### Usage

```
gsw_z_from_p(p, latitude)
```

### Arguments

| | |
|---|---|
| p | sea pressure [dbar], i.e. absolute pressure [dbar] minus 10.1325 dbar |
| latitude | latitude in decimal degrees, positive to the north of the equator. (This is called `lat` in the TEOS-10 Matlab code.) |

### Details

The present R function works with a wrapper to a C function contained within the GSW-C system (Version 3.05-4 dated 2017-08-07, available at https://github.com/TEOS-10/GSW-C, as git commit '5b4d959e54031f9e972f3e863f63e67fa4f5bfec'), which stems from the GSW-Fortran system (https://github.com/TEOS-10/GSW-Fortran) which in turn stems from the GSW-Matlab system (https://github.com/TEOS-10/GSW-Matlab). Consult http://www.teos-10.org to learn more about these software systems, their authorships, and the science behind it all.

### Value

height [ m ]

### References

http://www.teos-10.org/pubs/gsw/html/gsw_z_from_p.html

### See Also

Other things related to depth: gsw_p_from_z

### Examples

```
z <- gsw_z_from_p(c(10, 50, 125, 250, 600,1000), 4)
expect_equal(z/1e2, c(-0.099445834469453, -0.497180897012550, -1.242726219409978,
                      -2.484700576548589, -5.958253480356214, -9.920919060719987))
```

---

saar *Global SA lookup file*

---

### Description

This dataset is not intended for users, but rather for internal use within the gsw package. The dataset stores the 1.4M lookup table defined in the 8.3M file `src/gsw_saar_data.c` in the C library. (The .c file exceeds CRAN limitations on size.)

### Details

The data are designed to replace C elements defined as below in `src/gsw_saar_data.c`:

```
static int gsw_nx=91, gsw_ny=45, gsw_nz=45;
static double longs_ref[91];
static double lats_ref[45];
static double p_ref[45];
static double ndepth_ref[4095];
static double saar_ref[184275];
static double delta_sa_ref[184275];
```

R storage is in a list named `saar`, with elements named as in the C code, i.e. `gsw_nx` etc.

C storage for these variables is allocated as needed, and the data are inserted, when gsw is launched. Thus, the existing C library code "knows" about the data as local storage, which keeps alterations to the C library to a minimum.

The `saar` dataset was created by the following R code. The netcdf file used in this code comes from the GSW-Fortran repository (at commit `baa0c09ffc7ed1f74972a1a2902d8754caa5b4cb`) and its md5 value is `dacb3f981e8e710ac2e83477701b3905`.

```
library(ncdf4)
nc <- nc_open("~/git/GSW-Fortran/test/gsw_data_v3_0.nc")
## Use as.vector() since these will all get handed into C, which does not understand matrices.
p_ref <- as.vector(ncvar_get(nc, "p_ref"))
lats_ref <- as.vector(ncvar_get(nc, "lats_ref"))
```

```
longs_ref <- as.vector(ncvar_get(nc, "longs_ref"))
ndepth_ref <- as.vector(ncvar_get(nc, "ndepth_ref"))
ndepth_ref[!is.finite(ndepth_ref)] <- -9e99
saar_ref <- as.vector(ncvar_get(nc, "SAAR_ref"))
saar_ref[!is.finite(saar_ref)] <- -9e99
delta_sa_ref <- as.vector(ncvar_get(nc, "deltaSA_ref"))
delta_sa_ref[!is.finite(delta_sa_ref)] <- -9e99
saar <- list(gsw_nx=gsw_nx, gsw_ny=gsw_ny, gsw_nz=gsw_nz,
          longs_ref=longs_ref, lats_ref=lats_ref, p_ref=p_ref, ndepth_ref=ndepth_ref,
             saar_ref=saar_ref, delta_sa_ref=delta_sa_ref)
save(saar, file="saar.rda")
tools::resaveRdaFiles("saar.rda")
nc_close(nc)
```

# Index