

# Package ‘dggridR’

May 8, 2026

**Type** Package

**Title** Discrete Global Grids

**Version** 3.1.1

**Date** 2026-01-18

**NeedsCompilation** yes

**Depends** R (>= 4.1.0)

**Imports** Rcpp (>= 0.12.12), collapse (>= 1.8.0), sf (>= 1.0), s2 (>= 1.1)

**LinkingTo** Rcpp

**RcppModules** dgfuncs, gridgens, gridstats

**Suggests** ggplot2, knitr, rmarkdown, dplyr, maps, R.rsp, testthat

**License** AGPL (>= 3)

## Description

Spatial analyses involving binning require that every bin have the same area, but this is impossible using a rectangular grid laid over the Earth or over any projection of the Earth. Discrete global grids use hexagons, triangles, and diamonds to overcome this issue, overlaying the Earth with equally-sized bins. This package provides utilities for working with discrete global grids, along with utilities to aid in plotting such data.

**URL** <https://github.com/r-barnes/dggridR/>

**BugReports** <https://github.com/r-barnes/dggridR/issues>

**RoxygenNote** 7.3.2

**Encoding** UTF-8

**VignetteBuilder** knitr, R.rsp

**Language** en-US

**Author** Richard Barnes [aut],  
Kevin Sahr [aut, cph],  
Gerald Evenden [cph],  
Angus Johnson [cph],  
Frank Warmerdam [cph],  
Even Rouault [cph],

Lian Song [ctb],  
 Edzer Pebesma [ctb],  
 Sebastian Krantz [ctb, cre]

**Maintainer** Sebastian Krantz <sebastian.krantz@graduateinstitute.ch>

**Repository** CRAN

**Date/Publication** 2026-01-20 06:10:19 UTC

## Contents

dgcellstogrid . . . . .	3
dgconstruct . . . . .	4
dgearthgrid . . . . .	5
dgGEO_to_GEO . . . . .	6
dgGEO_to_PLANE . . . . .	7
dgGEO_to_PROJTRI . . . . .	8
dgGEO_to_Q2DD . . . . .	8
dgGEO_to_Q2DI . . . . .	9
dgGEO_to_SEQNUM . . . . .	10
dggetres . . . . .	11
dginfo . . . . .	11
dgmaxcell . . . . .	12
dgPROJTRI_to_GEO . . . . .	13
dgPROJTRI_to_PLANE . . . . .	13
dgPROJTRI_to_PROJTRI . . . . .	14
dgPROJTRI_to_Q2DD . . . . .	15
dgPROJTRI_to_Q2DI . . . . .	16
dgPROJTRI_to_SEQNUM . . . . .	16
dgQ2DD_to_GEO . . . . .	17
dgQ2DD_to_PLANE . . . . .	18
dgQ2DD_to_PROJTRI . . . . .	19
dgQ2DD_to_Q2DD . . . . .	19
dgQ2DD_to_Q2DI . . . . .	20
dgQ2DD_to_SEQNUM . . . . .	21
dgQ2DI_to_GEO . . . . .	22
dgQ2DI_to_PLANE . . . . .	22
dgQ2DI_to_PROJTRI . . . . .	23
dgQ2DI_to_Q2DD . . . . .	24
dgQ2DI_to_Q2DI . . . . .	25
dgQ2DI_to_SEQNUM . . . . .	25
dgquakes . . . . .	26
dgrectgrid . . . . .	27
dgsavegrid . . . . .	28
dgSEQNUM_to_GEO . . . . .	28
dgSEQNUM_to_PLANE . . . . .	29
dgSEQNUM_to_PROJTRI . . . . .	30
dgSEQNUM_to_Q2DD . . . . .	30
dgSEQNUM_to_Q2DI . . . . .	31

*dgcellstogrid* 3

dgSEQNUM_to_SEQNUM . . . . .	32
dgsetres . . . . .	32
dgshptogrid . . . . .	33
dgverify . . . . .	34
dg_closest_res . . . . .	35
dg_closest_res_to_area . . . . .	36
dg_closest_res_to_cls . . . . .	37
dg_closest_res_to_spacing . . . . .	38
dg_env . . . . .	39
dg_process_polydata . . . . .	39
dg_shpfname_south_africa . . . . .	40

**Index** 41

---

*dgcellstogrid*      *Return boundary coordinates for specified cells*

---

### Description

Returns the coordinates constituting the boundary of a specified set of cells. Duplicates are eliminated to reduce processing and storage requirements.

### Usage

```
dgcellstogrid(dggs, cells, savegrid = NA, return_sf = TRUE)
```

### Arguments

<code>dggs</code>	A <code>dggs</code> object from <code>dgconstruct()</code>
<code>cells</code>	The cells to get the boundaries of
<code>savegrid</code>	If <code>savegrid</code> is set to a file path, then a shapefile containing the grid is written to that path and the filename is returned. No other manipulations are done. Default: <code>NA</code> (do not save grid, return it)
<code>return_sf</code>	logical. If <code>FALSE</code> , a long-format data frame giving the coordinates of the vertices of each cell is returned. This is considerably faster and more memory efficient than creating an <code>sf</code> data frame.

### Value

Returns an `sf` object. If `!is.na(savegrid)`, returns a filename.

**Examples**

```

library(dggridR)
data(dgquakes)

#Construct a grid with cells about ~1000 miles wide
dggs      <- dgconstruct(spacing=1000,metric=FALSE)
dgquakes$cell <- dgGEO_to_SEQNUM(dggs,dgquakes$lat,dgquakes$lon)$seqnum

#Get grid cells for the earthquakes identified
grid      <- dgcellstogrid(dggs, dgquakes$cell)

```

---

<code>dgconstruct</code>	<i>Construct a discrete global grid system (dggs) object</i>
--------------------------	--

---

**Description**

Construct a discrete global grid system (dggs) object

**Usage**

```

dgconstruct(
  projection = "ISEA",
  aperture = 3,
  topology = "HEXAGON",
  res = NA,
  precision = 7,
  area = NA,
  spacing = NA,
  cls = NA,
  resround = "nearest",
  metric = TRUE,
  show_info = TRUE,
  azimuth_deg = 0,
  pole_lat_deg = 58.28252559,
  pole_lon_deg = 11.25
)

```

**Arguments**

<code>projection</code>	Type of grid to use. Options are: ISEA and FULLER. Default: ISEA3H
<code>aperture</code>	How finely subsequent resolution levels divide the grid. Options are: 3, 4. Not all options work with all projections and topologies. Default: 3
<code>topology</code>	Shape of cell. Options are: HEXAGON, DIAMOND, TRIANGLE. Default: HEXAGON

res	Resolution. Must be in the range [0,30]. Larger values represent finer resolutions. Appropriate resolutions can be found with <code>dg_closest_res_to_area()</code> , <code>dg_closest_res_to_spacing()</code> , and <code>dg_closest_res_to_cls()</code> . Default is 9, which corresponds to a cell area of ~2600 sq km and a cell spacing of ~50 km. Only one of res, area, length, or cls should be used.
precision	Round output to this number of decimal places. Must be in the range [0,30]. Default: 7.
area	The desired area of the grid's cells. Only one of res, area, length, or cls should be used.
spacing	The desired spacing between the center of adjacent cells. Only one of res, area, length, or cls should be used.
cls	The desired CLS of the cells. Only one of res, area, length, or cls should be used.
resround	What direction to search in. Must be nearest, up, or down.
metric	Whether input and output should be in metric (TRUE) or imperial (FALSE)
show_info	Print the area, spacing, and CLS of the chosen resolution.
azimuth_deg	Rotation in degrees of grid about its pole, value in [0,360]. Default=0.
pole_lat_deg	Latitude in degrees of the pole, value in [-90,90]. Default=58.28252559.
pole_lon_deg	Longitude in degrees of the pole, value in [-180,180]. Default=11.25.

**Value**

Returns a `dggs` object which can be passed to other `dggridR` functions

**Examples**

```
library(dggridR)
dggs <- dgconstruct(res=20)

dggs <- dgconstruct(area=5,metric=FALSE)
```

---

<code>dgearthgrid</code>	<i>Return the coordinates constituting the boundary of cells for the entire Earth</i>
--------------------------	---

---

**Description**

Note: If you have a high-resolution grid this may take a very long time to execute.

**Usage**

```
dgearthgrid(dggs, savegrid = NA, return_sf = TRUE)
```

**Arguments**

dggs	A dggs object from dgconstruct().
savegrid	If savegrid is set to a file path, then a shapefile containing the grid is written to that path and the filename is returned. No other manipulations are done. Default: NA (do not save grid, return it)
return_sf	logical. If FALSE, a long-format data frame giving the coordinates of the vertices of each cell is returned. This is considerably faster and more memory efficient than creating an sf data frame.

**Value**

Returns an sf object. If `!is.na(savegrid)`, returns a filename.

**Examples**

```
library(dggridR)
dggs <- dgconstruct(res=20)
res <- dg_closest_res_to_spacing(dggs, spacing=1000, round='down', metric=FALSE)
dggs <- dgsetres(dggs, res)
gridfilename <- dgearthgrid(dggs, savegrid=tempfile(fileext=".shp")) #Save directly to a file
```

---

 dgGEO\_to\_GEO

*Convert from GEO to GEO*


---

**Description**

Uses a discrete global grid system to convert between GEO and GEO (see vignette for details)

**Usage**

```
dgGEO_to_GEO(dggs, in_lon_deg, in_lat_deg)
```

**Arguments**

dggs	A dggs object from dgconstruct()
in_lon_deg	Vector of longitude, in degrees
in_lat_deg	Vector of latitude, in degrees

**Value**

Returns a dggs object which can be passed to other dggridR functions

**Examples**

```
## Not run:  
library(dggridR)  
dggs <- dgconstruct(res=20)  
  
dgGEO_to_GEO(dggs, in_lon_deg, in_lat_deg)  
  
## End(Not run)
```

---

dgGEO_to_PLANE	<i>Convert from GEO to PLANE</i>
----------------	----------------------------------

---

**Description**

Uses a discrete global grid system to convert between GEO and PLANE (see vignette for details)

**Usage**

```
dgGEO_to_PLANE(dggs, in_lon_deg, in_lat_deg)
```

**Arguments**

dggs	A dggs object from dgconstruct()
in_lon_deg	Vector of longitude, in degrees
in_lat_deg	Vector of latitude, in degrees

**Value**

Returns a dggs object which can be passed to other dggridR functions

**Examples**

```
## Not run:  
library(dggridR)  
dggs <- dgconstruct(res=20)  
  
dgGEO_to_PLANE(dggs, in_lon_deg, in_lat_deg)  
  
## End(Not run)
```

---

dgGEO\_to\_PROJTRI      *Convert from GEO to PROJTRI*

---

### Description

Uses a discrete global grid system to convert between GEO and PROJTRI (see vignette for details)

### Usage

```
dgGEO_to_PROJTRI(dggs, in_lon_deg, in_lat_deg)
```

### Arguments

dggs	A dggs object from dgconstruct()
in_lon_deg	Vector of longitude, in degrees
in_lat_deg	Vector of latitude, in degrees

### Value

Returns a dggs object which can be passed to other dggridR functions

### Examples

```
## Not run:
library(dggridR)
dggs <- dgconstruct(res=20)

dgGEO_to_PROJTRI(dggs, in_lon_deg, in_lat_deg)

## End(Not run)
```

---

dgGEO\_to\_Q2DD      *Convert from GEO to Q2DD*

---

### Description

Uses a discrete global grid system to convert between GEO and Q2DD (see vignette for details)

### Usage

```
dgGEO_to_Q2DD(dggs, in_lon_deg, in_lat_deg)
```

**Arguments**

dggs	A dggs object from dgconstruct()
in_lon_deg	Vector of longitude, in degrees
in_lat_deg	Vector of latitude, in degrees

**Value**

Returns a dggs object which can be passed to other dggridR functions

**Examples**

```
## Not run:
library(dggridR)
dggs <- dgconstruct(res=20)

dgGEO_to_Q2DD(dggs, in_lon_deg, in_lat_deg)

## End(Not run)
```

---

dgGEO_to_Q2DI	<i>Convert from GEO to Q2DI</i>
---------------	---------------------------------

---

**Description**

Uses a discrete global grid system to convert between GEO and Q2DI (see vignette for details)

**Usage**

```
dgGEO_to_Q2DI(dggs, in_lon_deg, in_lat_deg)
```

**Arguments**

dggs	A dggs object from dgconstruct()
in_lon_deg	Vector of longitude, in degrees
in_lat_deg	Vector of latitude, in degrees

**Value**

Returns a dggs object which can be passed to other dggridR functions

**Examples**

```
## Not run:  
library(dggridR)  
dggs <- dgconstruct(res=20)  
  
dgGEO_to_Q2DI(dggs, in_lon_deg, in_lat_deg)  
  
## End(Not run)
```

---

dgGEO_to_SEQNUM	<i>Convert from GEO to SEQNUM</i>
-----------------	-----------------------------------

---

**Description**

Uses a discrete global grid system to convert between GEO and SEQNUM (see vignette for details)

**Usage**

```
dgGEO_to_SEQNUM(dggs, in_lon_deg, in_lat_deg)
```

**Arguments**

dggs	A dggs object from dgconstruct()
in_lon_deg	Vector of longitude, in degrees
in_lat_deg	Vector of latitude, in degrees

**Value**

Returns a dggs object which can be passed to other dggridR functions

**Examples**

```
## Not run:  
library(dggridR)  
dggs <- dgconstruct(res=20)  
  
dgGEO_to_SEQNUM(dggs, in_lon_deg, in_lat_deg)  
  
## End(Not run)
```

---

dggetres	<i>Get table of grid resolution information</i>
----------	---

---

**Description**

Gets a grid's resolution and cell property info as a data frame.

**Usage**

```
dggetres(dggs)
```

**Arguments**

dggs            A dggs object from dgconstruct()

**Value**

A data frame containing the resolution levels, number of cells, area of those cells, intercell spacing, and characteristic length scale of the cells. All values are in kilometres.

**Examples**

```
library(dggridR)
dggs <- dgconstruct(res=20)
dggetres(dggs)
```

---

dginfo	<i>Print info about a dggs object to the screen</i>
--------	---

---

**Description**

dggs objects have many settings. This returns all of them, along with info about the grid being specified.

**Usage**

```
dginfo(dggs)
```

**Arguments**

dggs            A dggs object from dgconstruct()

**Value**

No return. All info is printed to the screen.

## Examples

```
library(dggridR)
dggs <- dgconstruct(res=20)
dginfo(dggs)
```

---

dgmaxcell

*Get largest cell id for a dggs*

---

## Description

Cells are labeled 1-N. This function returns N. This is useful if you want to choose cells from the dggs randomly.

## Usage

```
dgmaxcell(dggs, res = NA)
```

## Arguments

dggs	A dggs object from dgconstruct()
res	If NA, use the resolution specified by the dggs. Otherwise, override the resolution.

## Value

The maximum cell id.

## Examples

```
#Choose a set of cells randomly distributed over the Earth
library(dggridR)
dggs <- dgconstruct(spacing=1000, metric=FALSE, resound='down')
N <- 100 #Number of cells
maxcell <- dgmaxcell(dggs) #Get maximum cell id
cells <- sample(1:maxcell, N, replace=FALSE) #Choose random cells
grid <- dgcellstogrid(dggs,cells) #Get grid
```

---

dgPROJTRI\_to\_GEO      *Convert from PROJTRI to GEO*

---

**Description**

Uses a discrete global grid system to convert between PROJTRI and GEO (see vignette for details)

**Usage**

```
dgPROJTRI_to_GEO(dggs, in_tnum, in_tx, in_ty)
```

**Arguments**

dggs	A dggs object from dgconstruct()
in_tnum	Vector of triangle numbers
in_tx	Vector of triangle x values
in_ty	Vector of triangle y values

**Value**

Returns a dggs object which can be passed to other dggridR functions

**Examples**

```
## Not run:  
library(dggridR)  
dggs <- dgconstruct(res=20)  
  
dgPROJTRI_to_GEO(dggs, in_tnum, in_tx, in_ty)  
  
## End(Not run)
```

---

dgPROJTRI\_to\_PLANE      *Convert from PROJTRI to PLANE*

---

**Description**

Uses a discrete global grid system to convert between PROJTRI and PLANE (see vignette for details)

**Usage**

```
dgPROJTRI_to_PLANE(dggs, in_tnum, in_tx, in_ty)
```

**Arguments**

dggs	A dggs object from dgconstruct()
in_tnum	Vector of triangle numbers
in_tx	Vector of triangle x values
in_ty	Vector of triangle y values

**Value**

Returns a dggs object which can be passed to other dggridR functions

**Examples**

```
## Not run:
library(dggridR)
dggs <- dgconstruct(res=20)

dgPROJTRI_to_PLANE(dggs, in_tnum, in_tx, in_ty)

## End(Not run)
```

---

dgPROJTRI\_to\_PROJTRI *Convert from PROJTRI to PROJTRI*

---

**Description**

Uses a discrete global grid system to convert between PROJTRI and PROJTRI (see vignette for details)

**Usage**

```
dgPROJTRI_to_PROJTRI(dggs, in_tnum, in_tx, in_ty)
```

**Arguments**

dggs	A dggs object from dgconstruct()
in_tnum	Vector of triangle numbers
in_tx	Vector of triangle x values
in_ty	Vector of triangle y values

**Value**

Returns a dggs object which can be passed to other dggridR functions

**Examples**

```
## Not run:
library(dggridR)
dggs <- dgconstruct(res=20)

dgPROJTRI_to_PROJTRI(dggs, in_tnum, in_tx, in_ty)

## End(Not run)
```

---

dgPROJTRI\_to\_Q2DD      *Convert from PROJTRI to Q2DD*

---

**Description**

Uses a discrete global grid system to convert between PROJTRI and Q2DD (see vignette for details)

**Usage**

```
dgPROJTRI_to_Q2DD(dggs, in_tnum, in_tx, in_ty)
```

**Arguments**

dggs	A dggs object from dgconstruct()
in_tnum	Vector of triangle numbers
in_tx	Vector of triangle x values
in_ty	Vector of triangle y values

**Value**

Returns a dggs object which can be passed to other dggridR functions

**Examples**

```
## Not run:
library(dggridR)
dggs <- dgconstruct(res=20)

dgPROJTRI_to_Q2DD(dggs, in_tnum, in_tx, in_ty)

## End(Not run)
```

---

dgPROJTRI\_to\_Q2DI      *Convert from PROJTRI to Q2DI*

---

### Description

Uses a discrete global grid system to convert between PROJTRI and Q2DI (see vignette for details)

### Usage

```
dgPROJTRI_to_Q2DI(dggs, in_tnum, in_tx, in_ty)
```

### Arguments

dggs	A dggs object from dgconstruct()
in_tnum	Vector of triangle numbers
in_tx	Vector of triangle x values
in_ty	Vector of triangle y values

### Value

Returns a dggs object which can be passed to other dggridR functions

### Examples

```
## Not run:
library(dggridR)
dggs <- dgconstruct(res=20)

dgPROJTRI_to_Q2DI(dggs, in_tnum, in_tx, in_ty)

## End(Not run)
```

---

dgPROJTRI\_to\_SEQNUM      *Convert from PROJTRI to SEQNUM*

---

### Description

Uses a discrete global grid system to convert between PROJTRI and SEQNUM (see vignette for details)

### Usage

```
dgPROJTRI_to_SEQNUM(dggs, in_tnum, in_tx, in_ty)
```

**Arguments**

dggs	A dggs object from dgconstruct()
in_tnum	Vector of triangle numbers
in_tx	Vector of triangle x values
in_ty	Vector of triangle y values

**Value**

Returns a dggs object which can be passed to other dggridR functions

**Examples**

```
## Not run:
library(dggridR)
dggs <- dgconstruct(res=20)

dgPROJTRI_to_SEQNUM(dggs, in_tnum, in_tx, in_ty)

## End(Not run)
```

---

dgQ2DD_to_GEO	<i>Convert from Q2DD to GEO</i>
---------------	---------------------------------

---

**Description**

Uses a discrete global grid system to convert between Q2DD and GEO (see vignette for details)

**Usage**

```
dgQ2DD_to_GEO(dggs, in_quad, in_qx, in_qy)
```

**Arguments**

dggs	A dggs object from dgconstruct()
in_quad	Vector of quad numbers
in_qx	Vector of quadrant x values
in_qy	Vector of quadrant y values

**Value**

Returns a dggs object which can be passed to other dggridR functions

**Examples**

```
## Not run:  
library(dggridR)  
dggs <- dgconstruct(res=20)  
  
dgQ2DD_to_GEO(dggs, in_quad, in_qx, in_qy)  
  
## End(Not run)
```

---

dgQ2DD\_to\_PLANE      *Convert from Q2DD to PLANE*

---

**Description**

Uses a discrete global grid system to convert between Q2DD and PLANE (see vignette for details)

**Usage**

```
dgQ2DD_to_PLANE(dggs, in_quad, in_qx, in_qy)
```

**Arguments**

dggs	A dggs object from dgconstruct()
in_quad	Vector of quad numbers
in_qx	Vector of quadrant x values
in_qy	Vector of quadrant y values

**Value**

Returns a dggs object which can be passed to other dggridR functions

**Examples**

```
## Not run:  
library(dggridR)  
dggs <- dgconstruct(res=20)  
  
dgQ2DD_to_PLANE(dggs, in_quad, in_qx, in_qy)  
  
## End(Not run)
```

---

dgQ2DD\_to\_PROJTRI      *Convert from Q2DD to PROJTRI*

---

**Description**

Uses a discrete global grid system to convert between Q2DD and PROJTRI (see vignette for details)

**Usage**

```
dgQ2DD_to_PROJTRI(dggs, in_quad, in_qx, in_qy)
```

**Arguments**

dggs	A dggs object from dgconstruct()
in_quad	Vector of quad numbers
in_qx	Vector of quadrant x values
in_qy	Vector of quadrant y values

**Value**

Returns a dggs object which can be passed to other dggridR functions

**Examples**

```
## Not run:  
library(dggridR)  
dggs <- dgconstruct(res=20)  
  
dgQ2DD_to_PROJTRI(dggs, in_quad, in_qx, in_qy)  
  
## End(Not run)
```

---

dgQ2DD\_to\_Q2DD      *Convert from Q2DD to Q2DD*

---

**Description**

Uses a discrete global grid system to convert between Q2DD and Q2DD (see vignette for details)

**Usage**

```
dgQ2DD_to_Q2DD(dggs, in_quad, in_qx, in_qy)
```

**Arguments**

dggs	A dggs object from dgconstruct()
in_quad	Vector of quad numbers
in_qx	Vector of quadrant x values
in_qy	Vector of quadrant y values

**Value**

Returns a dggs object which can be passed to other dggridR functions

**Examples**

```
## Not run:
library(dggridR)
dggs <- dgconstruct(res=20)

dgQ2DD_to_Q2DD(dggs, in_quad, in_qx, in_qy)

## End(Not run)
```

---

dgQ2DD\_to\_Q2DI      *Convert from Q2DD to Q2DI*

---

**Description**

Uses a discrete global grid system to convert between Q2DD and Q2DI (see vignette for details)

**Usage**

```
dgQ2DD_to_Q2DI(dggs, in_quad, in_qx, in_qy)
```

**Arguments**

dggs	A dggs object from dgconstruct()
in_quad	Vector of quad numbers
in_qx	Vector of quadrant x values
in_qy	Vector of quadrant y values

**Value**

Returns a dggs object which can be passed to other dggridR functions

**Examples**

```
## Not run:  
library(dggridR)  
dggs <- dgconstruct(res=20)  
  
dgQ2DD_to_Q2DI(dggs, in_quad, in_qx, in_qy)  
  
## End(Not run)
```

---

dgQ2DD_to_SEQNUM	<i>Convert from Q2DD to SEQNUM</i>
------------------	------------------------------------

---

**Description**

Uses a discrete global grid system to convert between Q2DD and SEQNUM (see vignette for details)

**Usage**

```
dgQ2DD_to_SEQNUM(dggs, in_quad, in_qx, in_qy)
```

**Arguments**

dggs	A dggs object from dgconstruct()
in_quad	Vector of quad numbers
in_qx	Vector of quadrant x values
in_qy	Vector of quadrant y values

**Value**

Returns a dggs object which can be passed to other dggridR functions

**Examples**

```
## Not run:  
library(dggridR)  
dggs <- dgconstruct(res=20)  
  
dgQ2DD_to_SEQNUM(dggs, in_quad, in_qx, in_qy)  
  
## End(Not run)
```

---

dgQ2DI\_to\_GEO                      *Convert from Q2DI to GEO*

---

### Description

Uses a discrete global grid system to convert between Q2DI and GEO (see vignette for details)

### Usage

```
dgQ2DI_to_GEO(dggs, in_quad, in_i, in_j)
```

### Arguments

dggs	A dggs object from dgconstruct()
in_quad	Vector of quad numbers
in_i	Vector of quadrant i values
in_j	Vector of quadrant j values

### Value

Returns a dggs object which can be passed to other dggridR functions

### Examples

```
## Not run:
library(dggridR)
dggs <- dgconstruct(res=20)

dgQ2DI_to_GEO(dggs, in_quad, in_i, in_j)

## End(Not run)
```

---

dgQ2DI\_to\_PLANE                      *Convert from Q2DI to PLANE*

---

### Description

Uses a discrete global grid system to convert between Q2DI and PLANE (see vignette for details)

### Usage

```
dgQ2DI_to_PLANE(dggs, in_quad, in_i, in_j)
```

**Arguments**

dggs	A dggs object from dgconstruct()
in_quad	Vector of quad numbers
in_i	Vector of quadrant i values
in_j	Vector of quadrant j values

**Value**

Returns a dggs object which can be passed to other dggridR functions

**Examples**

```
## Not run:
library(dggridR)
dggs <- dgconstruct(res=20)

dgQ2DI_to_PLANE(dggs, in_quad, in_i, in_j)

## End(Not run)
```

---

dgQ2DI\_to\_PROJTRI      *Convert from Q2DI to PROJTRI*

---

**Description**

Uses a discrete global grid system to convert between Q2DI and PROJTRI (see vignette for details)

**Usage**

```
dgQ2DI_to_PROJTRI(dggs, in_quad, in_i, in_j)
```

**Arguments**

dggs	A dggs object from dgconstruct()
in_quad	Vector of quad numbers
in_i	Vector of quadrant i values
in_j	Vector of quadrant j values

**Value**

Returns a dggs object which can be passed to other dggridR functions

**Examples**

```
## Not run:
library(dggridR)
dggs <- dgconstruct(res=20)

dgQ2DI_to_PROJTRI(dggs, in_quad, in_i, in_j)

## End(Not run)
```

---

dgQ2DI_to_Q2DD	<i>Convert from Q2DI to Q2DD</i>
----------------	----------------------------------

---

**Description**

Uses a discrete global grid system to convert between Q2DI and Q2DD (see vignette for details)

**Usage**

```
dgQ2DI_to_Q2DD(dggs, in_quad, in_i, in_j)
```

**Arguments**

dggs	A dggs object from dgconstruct()
in_quad	Vector of quad numbers
in_i	Vector of quadrant i values
in_j	Vector of quadrant j values

**Value**

Returns a dggs object which can be passed to other dggridR functions

**Examples**

```
## Not run:
library(dggridR)
dggs <- dgconstruct(res=20)

dgQ2DI_to_Q2DD(dggs, in_quad, in_i, in_j)

## End(Not run)
```

---

dgQ2DI\_to\_Q2DI      *Convert from Q2DI to Q2DI*

---

**Description**

Uses a discrete global grid system to convert between Q2DI and Q2DI (see vignette for details)

**Usage**

```
dgQ2DI_to_Q2DI(dggs, in_quad, in_i, in_j)
```

**Arguments**

dggs	A dggs object from dgconstruct()
in_quad	Vector of quad numbers
in_i	Vector of quadrant i values
in_j	Vector of quadrant j values

**Value**

Returns a dggs object which can be passed to other dggridR functions

**Examples**

```
## Not run:  
library(dggridR)  
dggs <- dgconstruct(res=20)  
  
dgQ2DI_to_Q2DI(dggs, in_quad, in_i, in_j)  
  
## End(Not run)
```

---

dgQ2DI\_to\_SEQNUM      *Convert from Q2DI to SEQNUM*

---

**Description**

Uses a discrete global grid system to convert between Q2DI and SEQNUM (see vignette for details)

**Usage**

```
dgQ2DI_to_SEQNUM(dggs, in_quad, in_i, in_j)
```

**Arguments**

dggs	A dggs object from dgconstruct()
in_quad	Vector of quad numbers
in_i	Vector of quadrant i values
in_j	Vector of quadrant j values

**Value**

Returns a dggs object which can be passed to other dggridR functions

**Examples**

```
## Not run:  
library(dggridR)  
dggs <- dgconstruct(res=20)  
  
dgQ2DI_to_SEQNUM(dggs, in_quad, in_i, in_j)  
  
## End(Not run)
```

---

dgquakes

*All earthquakes with magnitude  $\geq 3.0$  earthquakes for 2015*

---

**Description**

A data frame with 19914 observations on the following 4 variables.

time Time of the quake. Example: 2015-12-31T23:39:28.940Z

lat Latitude of the epicenter. Example: -7.0711

lon Longitude of the epicenter. Example: -173.5178

mag Magnitude of the quake. Example: 3.2

**Usage**

```
data(dgquakes)
```

**Format**

data frame

**Source**

The USGS Earthquake Hazards Program (<https://earthquake.usgs.gov/earthquakes/search/>).

---

dgrectgrid	<i>Return the coordinates constituting the boundary of cells within a specified region</i>
------------	--

---

### Description

Note: This may generate odd results for very large rectangles, because putting rectangles on spheres is weird... as you should know, if you're using this package.

### Usage

```
dgrectgrid(
  dggs,
  minlat = -1,
  minlon = -1,
  maxlat = 1,
  maxlon = 1,
  cellsize = 0.1,
  ...
)
```

### Arguments

dggs	A dggs object from dgconstruct()
minlat	Minimum latitude of region of interest
minlon	Minimum longitude of region of interest
maxlat	Maximum latitude of region of interest
maxlon	Maximum longitude of region of interest
cellsize	Distance, in degrees, between the sample points used to generate the grid. Small values yield long generation times while large values may omit cells.
...	Further arguments passed to <a href="#">dgcellstogrid</a> .

### Value

Returns an sf object. If `!is.na(savegrid)`, returns a filename.

### Examples

```
library(dggridR)
dggs <- dgconstruct(spacing=1000,metric=FALSE,resound='down')

#Get grid cells for the conterminous United States
grid <- dgrectgrid(dggs,
  minlat=24.7433195, minlon=-124.7844079,
  maxlat=49.3457868, maxlon=-66.9513812)
```

---

dgsavegrid	<i>Saves a generated grid to a shapefile</i>
------------	--

---

**Description**

Saves a generated grid to a shapefile

**Usage**

```
dgsavegrid(grid, shpfname)
```

**Arguments**

grid	Grid to be saved
shpfname	File to save the grid to

**Value**

The filename the grid was saved to

---

dgSEQNUM_to_GEO	<i>Convert from SEQNUM to GEO</i>
-----------------	-----------------------------------

---

**Description**

Uses a discrete global grid system to convert between SEQNUM and GEO (see vignette for details)

**Usage**

```
dgSEQNUM_to_GEO(dggs, in_seqnum)
```

**Arguments**

dggs	A dggs object from dgconstruct()
in_seqnum	Globally unique number identifying the surface polygon

**Value**

Returns a dggs object which can be passed to other dggridR functions

**Examples**

```
## Not run:  
library(dggridR)  
dggs <- dgconstruct(res=20)  
  
dgSEQNUM_to_GEO(dggs, in_seqnum)  
  
## End(Not run)
```

---

dgSEQNUM\_to\_PLANE      *Convert from SEQNUM to PLANE*

---

**Description**

Uses a discrete global grid system to convert between SEQNUM and PLANE (see vignette for details)

**Usage**

```
dgSEQNUM_to_PLANE(dggs, in_seqnum)
```

**Arguments**

dggs	A dggs object from dgconstruct()
in_seqnum	Globally unique number identifying the surface polygon

**Value**

Returns a dggs object which can be passed to other dggridR functions

**Examples**

```
## Not run:  
library(dggridR)  
dggs <- dgconstruct(res=20)  
  
dgSEQNUM_to_PLANE(dggs, in_seqnum)  
  
## End(Not run)
```

---

dgSEQNUM\_to\_PROJTRI     *Convert from SEQNUM to PROJTRI*

---

### Description

Uses a discrete global grid system to convert between SEQNUM and PROJTRI (see vignette for details)

### Usage

```
dgSEQNUM_to_PROJTRI(dggs, in_seqnum)
```

### Arguments

dggs	A dggs object from dgconstruct()
in_seqnum	Globally unique number identifying the surface polygon

### Value

Returns a dggs object which can be passed to other dggridR functions

### Examples

```
## Not run:
library(dggridR)
dggs <- dgconstruct(res=20)

dgSEQNUM_to_PROJTRI(dggs, in_seqnum)

## End(Not run)
```

---

dgSEQNUM\_to\_Q2DD     *Convert from SEQNUM to Q2DD*

---

### Description

Uses a discrete global grid system to convert between SEQNUM and Q2DD (see vignette for details)

### Usage

```
dgSEQNUM_to_Q2DD(dggs, in_seqnum)
```

**Arguments**

dggs            A dggs object from dgconstruct()  
in\_seqnum       Globally unique number identifying the surface polygon

**Value**

Returns a dggs object which can be passed to other dggridR functions

**Examples**

```
## Not run:
library(dggridR)
dggs <- dgconstruct(res=20)

dgSEQNUM_to_Q2DD(dggs, in_seqnum)

## End(Not run)
```

---

dgSEQNUM\_to\_Q2DI            *Convert from SEQNUM to Q2DI*

---

**Description**

Uses a discrete global grid system to convert between SEQNUM and Q2DI (see vignette for details)

**Usage**

```
dgSEQNUM_to_Q2DI(dggs, in_seqnum)
```

**Arguments**

dggs            A dggs object from dgconstruct()  
in\_seqnum       Globally unique number identifying the surface polygon

**Value**

Returns a dggs object which can be passed to other dggridR functions

**Examples**

```
## Not run:
library(dggridR)
dggs <- dgconstruct(res=20)

dgSEQNUM_to_Q2DI(dggs, in_seqnum)

## End(Not run)
```

---

dgSEQNUM\_to\_SEQNUM      *Convert from SEQNUM to SEQNUM*

---

### Description

Uses a discrete global grid system to convert between SEQNUM and SEQNUM (see vignette for details)

### Usage

```
dgSEQNUM_to_SEQNUM(dggs, in_seqnum)
```

### Arguments

dggs	A dggs object from dgconstruct()
in_seqnum	Globally unique number identifying the surface polygon

### Value

Returns a dggs object which can be passed to other dggridR functions

### Examples

```
## Not run:
library(dggridR)
dggs <- dgconstruct(res=20)

dgSEQNUM_to_SEQNUM(dggs, in_seqnum)

## End(Not run)
```

---

dgsetres      *Set the resolution of a dggs object*

---

### Description

Set the resolution of a dggs object

### Usage

```
dgsetres(dggs, res)
```

**Arguments**

dggs	A dggs object from dgconstruct().
res	Resolution. Must be in the range [0,30]. Larger values represent finer resolutions. Appropriate resolutions can be found with dg_closest_res_to_area(), dg_closest_res_to_spacing(), and dg_closest_res_to_cls(). Default is 9, which corresponds to a cell area of ~2600 sq km and a cell spacing of ~50 km. Default: 9.

**Value**

Returns a dggs object which can be passed to other dggridR functions

**Examples**

```
library(dggridR)
dggs <- dgconstruct(res=20)
dggs <- dgsetres(dggs,10)
```

---

dgshptogrid

*Return boundary coordinates for cells intersecting a shapefile*


---

**Description**

Returns the coordinates constituting the boundary of a set of cells which intersect or are contained by a polygon (or polygons) specified in a shapefile. Note that grid cells are also generated for holes in the shapefile's polygon(s).

Note that coordinates in the shapefile must be rounded to check polygon intersections. Currently this round preserves eight decimal digits of precision.

The eighth decimal place is worth up to 1.1 mm of precision: this is good for charting the motions of tectonic plates and the movements of volcanoes. Permanent, corrected, constantly-running GPS base stations might be able to achieve this level of accuracy.

In other words: you should be just fine with this level of precision.

**Usage**

```
dgshptogrid(dggs, shpfname, cellsize = 0.1, ...)
```

**Arguments**

dggs	A dggs object from dgconstruct()
shpfname	Either a sf data frame or the file name of the shapefile. Filename should end with '.shp'.
cellsize	Distance, in degrees, between the sample points used to generate the grid. Small values yield long generation times while large values may omit cells.
...	Further arguments passed to <a href="#">dgcellstogrid</a> .

**Value**

Returns an sf object. If `!is.na(savegrid)`, returns a filename.

**Examples**

```
library(dggridR)

dggs <- dgconstruct(spacing=25, metric=FALSE, resound='nearest')
south_africa_grid <- dgshptogrid(dggs,dg_shpname_south_africa())
```

---

dgverify

*Verify that a dggs object has appropriate values*

---

**Description**

Verify that a dggs object has appropriate values

**Usage**

```
dgverify(dggs)
```

**Arguments**

dggs            The dggs object to be verified

**Value**

The function has no return value. A stop signal is raised if the object is misspecified

**Examples**

```
library(dggridR)
dggs <- dgconstruct(res=20)
dgverify(dggs)
```

---

dg_closest_res	<i>Determine an appropriate grid resolution based on input data.</i>
----------------	--

---

### Description

This is a generic function that is used to determine an appropriate resolution given an area, cell spacing, or correlated length scale. It does so by extracting the appropriate length/area column and searching it for a value close to the input.

### Usage

```
dg_closest_res(  
  dggs,  
  col,  
  val,  
  round = "nearest",  
  show_info = TRUE,  
  metric = TRUE  
)
```

### Arguments

dggs	A dggs object from dgconstruct()
col	Column in which to search for a close value. Should be: area_km, spacing_km, or cls_km.
val	The value to search for
round	What direction to search in. Must be nearest, up, or down.
show_info	Print the area, spacing, and CLS of the chosen resolution.
metric	Whether input and output should be in metric (TRUE) or imperial (FALSE)

### Value

A number representing the grid resolution

### Examples

```
library(dggridR)  
dggs <- dgconstruct(res=20)  
res <- dg_closest_res(dggs, 'area_km', 1)  
dggs <- dgsetres(dggs, res)
```

---

`dg_closest_res_to_area`*Determine resolution based on desired area*

---

### Description

Determine an appropriate grid resolution based on a desired cell area.

### Usage

```
dg_closest_res_to_area(  
  dggs,  
  area,  
  round = "nearest",  
  show_info = TRUE,  
  metric = TRUE  
)
```

### Arguments

<code>dggs</code>	A dggs object from <code>dgconstruct()</code>
<code>area</code>	The desired area of the grid's cells
<code>round</code>	What direction to search in. Must be nearest, up, or down.
<code>show_info</code>	Print the area, spacing, and CLS of the chosen resolution.
<code>metric</code>	Whether input and output should be in metric (TRUE) or imperial (FALSE)

### Value

A number representing the grid resolution

### Examples

```
library(dggridR)  
dggs <- dgconstruct(res=20)  
res <- dg_closest_res_to_area(dggs,1)  
dggs <- dgsetres(dggs,res)
```

---

dg\_closest\_res\_to\_cls *Determine an appropriate grid resolution based on a desired characteristic length scale of the cells.*

---

### Description

The characteristic length scale (CLS) is the diameter of a spherical cap of the same area as a cell of the specified resolution.

### Usage

```
dg_closest_res_to_cls(  
  dggs,  
  cls,  
  round = "nearest",  
  show_info = TRUE,  
  metric = TRUE  
)
```

### Arguments

dggs	A dggs object from dgconstruct()
cls	The desired CLS of the cells.
round	What direction to search in. Must be nearest, up, or down.
show_info	Print the area, spacing, and CLS of the chosen resolution.
metric	Whether input and output should be in metric (TRUE) or imperial (FALSE)

### Value

A number representing the grid resolution

### Examples

```
library(dggridR)  
dggs <- dgconstruct(res=20)  
res <- dg_closest_res_to_cls(dggs,1)  
dggs <- dgsetres(dggs,res)
```

---

`dg_closest_res_to_spacing`*Determine grid resolution from desired spacing.*

---

**Description**

Determine an appropriate grid resolution based on a desired spacing between the center of adjacent cells.

**Usage**

```
dg_closest_res_to_spacing(  
  dggs,  
  spacing,  
  round = "nearest",  
  show_info = TRUE,  
  metric = TRUE  
)
```

**Arguments**

<code>dggs</code>	A <code>dggs</code> object from <code>dgconstruct()</code>
<code>spacing</code>	The desired spacing between the center of adjacent cells
<code>round</code>	What direction to search in. Must be nearest, up, or down.
<code>show_info</code>	Print the area, spacing, and CLS of the chosen resolution.
<code>metric</code>	Whether input and output should be in metric (TRUE) or imperial (FALSE)

**Value**

A number representing the grid resolution

**Examples**

```
library(dggridR)  
dggs <- dgconstruct(res=20)  
res <- dg_closest_res_to_spacing(dggs,1)  
dggs <- dgsetres(dggs,res)
```

---

`dg_env`*Control global aspects of the dggridR package*

---

**Description**

This environment is used to control global features of the dggridR package. At the moment the only option is 'dg\_debug' which, when set to TRUE provides extensive outputs useful for tracking down bugs.

**Usage**`dg_env`**Format**

An object of class environment of length 1.

---

`dg_process_polydata`*Load a KML file*

---

**Description**

Convert data from internal dggrid functions into something useful: an sp object or a data frame

**Usage**`dg_process_polydata(polydata)`**Arguments**

`polydata` Polygons generated by dggrid. These will be converted.

**Value**

Returns an sf object.

---

dg\_shpname\_south\_africa

*National border of South Africa*

---

**Description**

This variable points to a shapefile containing the national border of South Africa

**Usage**

dg\_shpname\_south\_africa()

**Value**

A filename of a shapefile containing the national border of South Africa

# Index

## \* datasets

- dg\_env, 39
- dgquakes, 26
  
- dg\_closest\_res, 35
- dg\_closest\_res\_to\_area, 36
- dg\_closest\_res\_to\_cls, 37
- dg\_closest\_res\_to\_spacing, 38
- dg\_env, 39
- dg\_process\_polydata, 39
- dg\_shpfname\_south\_africa, 40
- dgcellstogrid, 3, 27, 33
- dgconstruct, 4
- dgearthgrid, 5
- dgGEO\_to\_GEO, 6
- dgGEO\_to\_PLANE, 7
- dgGEO\_to\_PROJTRI, 8
- dgGEO\_to\_Q2DD, 8
- dgGEO\_to\_Q2DI, 9
- dgGEO\_to\_SEQNUM, 10
- dggetres, 11
- dginfo, 11
- dgmaxcell, 12
- dgPROJTRI\_to\_GEO, 13
- dgPROJTRI\_to\_PLANE, 13
- dgPROJTRI\_to\_PROJTRI, 14
- dgPROJTRI\_to\_Q2DD, 15
- dgPROJTRI\_to\_Q2DI, 16
- dgPROJTRI\_to\_SEQNUM, 16
- dgQ2DD\_to\_GEO, 17
- dgQ2DD\_to\_PLANE, 18
- dgQ2DD\_to\_PROJTRI, 19
- dgQ2DD\_to\_Q2DD, 19
- dgQ2DD\_to\_Q2DI, 20
- dgQ2DD\_to\_SEQNUM, 21
- dgQ2DI\_to\_GEO, 22
- dgQ2DI\_to\_PLANE, 22
- dgQ2DI\_to\_PROJTRI, 23
- dgQ2DI\_to\_Q2DD, 24
- dgQ2DI\_to\_Q2DI, 25
  
- dgQ2DI\_to\_SEQNUM, 25
- dgquakes, 26
- dgrectgrid, 27
- dgsavegrid, 28
- dgSEQNUM\_to\_GEO, 28
- dgSEQNUM\_to\_PLANE, 29
- dgSEQNUM\_to\_PROJTRI, 30
- dgSEQNUM\_to\_Q2DD, 30
- dgSEQNUM\_to\_Q2DI, 31
- dgSEQNUM\_to\_SEQNUM, 32
- dgsetres, 32
- dgshptogrid, 33
- dgverify, 34