

Package: globalhex (via r-universe)

October 15, 2026

Title H3-Style Access to Hierarchical DGGRID Grids

Version 0.0.0.9000

Description An experimental package exposing a small, H3-shaped R API over modern DGGRID hierarchical indexes, using a bundled copy of DGGRID. The default grid is the equal-area IGEO7 grid with Z7 digit-string indexes. ISEA3H with Z3 indexes and a partial compatibility path through dggridR are also supported. Large cell datasets can be aggregated through the hierarchy and plotted with viewport-aware level of detail before polygon geometry is constructed, with automatic raster rendering for large display layers. Optional raster helpers provide exact area-weighted extraction and fast cell-center sampling from modern terra rasters.

License AGPL (>= 3)

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Imports cli, jsonlite, rlang, tibble, vctrs

Suggests bench, dggridR, dplyr, exactextractr, knitr, maps, rmarkdown, sf, terra, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

SystemRequirements GNU make, C++17

URL <https://github.com/flmnh-ai/globalhex>,
<https://flmnh-ai.github.io/globalhex/>

BugReports <https://github.com/flmnh-ai/globalhex/issues>

Config/pak/sysreqs make

Repository <https://flmnh-ai.r-universe.dev>

Date/Publication 2026-10-15 21:08:09 UTC

RemoteUrl <https://github.com/flmnh-ai/globalhex>

RemoteRef HEAD
RemoteSha 4c35a6f3422936bb6ca97efc50b16a7d48731df5

Contents

as_dggridR_dggs	2
as_globalhex_cell	3
as_seqnum	4
cell_aggregate	4
cell_boundary	5
cell_center	5
cell_children	6
cell_cover	6
cell_disk	7
cell_distance	8
cell_extract	8
cell_grid	9
cell_is_valid	9
cell_neighbors	10
cell_parent	10
cell_resolution	11
cell_ring	11
cell_sample	12
globalhex_info	12
hex_data	13
hex_grid	13
legacy_grid	14
plot.globalhex_data	15
plot_resolution	16
point_to_cell	17
raster_to_cell	18
st_as_sf.globalhex_cell	19
Index	21

as_dggridR_dggs	<i>Convert to a low-level dggridR grid</i>
-----------------	--

Description

Convert to a low-level dggridR grid

Usage

as_dggridR_dggs(grid, resolution = NULL)

Arguments

grid An ISEA3H `hex_grid()` or a `legacy_grid()`.

resolution Resolution for a modern grid; ignored for a legacy grid.

Value

A dggridR grid list.

as_globalhex_cell	<i>Construct a globalhex cell vector</i>
-------------------	--

Description

Use this after character-only storage such as CSV or a database. Cell vectors may contain multiple resolutions, but all cells must share one grid system and orientation.

Usage

```
as_globalhex_cell(x = character(), grid = hex_grid())
```

Arguments

x A character vector of DGGRID cell IDs.

grid A grid from `hex_grid()` or `legacy_grid()`.

Value

A globalhex_cell vector.

Examples

```
grid <- hex_grid()
as_globalhex_cell(c("0203124", "02031240"), grid)
```

as_seqnum	<i>Convert hierarchical cells to sequence numbers</i>
-----------	---

Description

Values are returned as character strings to avoid losing integer precision.

Usage

```
as_seqnum(cell)
```

Arguments

cell A globalhex_cell vector.

Value

A character vector with one sequence number per cell.

cell_aggregate	<i>Aggregate values to hierarchical ancestors</i>
----------------	---

Description

Parent assignment uses the Z7 or Z3 digit-string hierarchy directly; it does not perform polygon operations or invoke DGGRID. With fun = NULL, numeric values use the mean, logical values use a majority rule, and categorical values use the mode. Summation is always explicit (fun = sum).

Usage

```
cell_aggregate(x, resolution, fun = NULL, na.rm = TRUE, value = NULL)
```

Arguments

x A `hex_data()` object.

resolution Target ancestor resolution.

fun A summary function or function name. NULL selects a conservative type-aware default.

na.rm Whether to remove missing values before aggregation.

value Attribute columns to aggregate. NULL aggregates all columns.

Value

A globalhex_data object with one row per represented ancestor.

Examples

```
cells <- point_to_cell(rbind(c(0, 0), c(0.01, 0.01)), 5)
cell_aggregate(hex_data(cells, value = c(1, 3)), 4)
```

cell_boundary	<i>Return cell boundary rings</i>
---------------	-----------------------------------

Description

Return cell boundary rings

Usage

```
cell_boundary(cell)
```

Arguments

cell A globalhex_cell vector.

Value

A tibble with cell and boundary; boundary is a list-column of closed longitude/latitude matrices.

cell_center	<i>Return cell centers</i>
-------------	----------------------------

Description

Return cell centers

Usage

```
cell_center(cell)
```

Arguments

cell A globalhex_cell vector.

Value

A tibble with cell, lon, and lat columns.

Examples

```
x <- point_to_cell(c(-73.9857, 40.7484), 5)
cell_center(x)
```

cell_children	<i>Return cell descendants</i>
---------------	--------------------------------

Description

Return cell descendants

Usage

```
cell_children(cell, resolution = NULL, max_cells = 100000L, progress = NULL)
```

Arguments

cell	A globalhex_cell vector.
resolution	Target finer resolution. NULL selects immediate children for each input cell.
max_cells	Safety cap per input cell.
progress	Show progress for multi-level expansion. NULL shows a delayed progress bar in interactive sessions, while TRUE or FALSE always enables or disables it.

Value

A tibble with the original parent and each descendant cell.

cell_cover	<i>Cover polygons with cells</i>
------------	----------------------------------

Description

DGGRID selects cells whose nuclei fall inside the supplied polygon region. Large polygons at fine resolutions can be expensive.

Usage

```
cell_cover(
  polygon,
  resolution,
  grid = hex_grid(),
  include_holes = TRUE,
  max_cells = 100000L,
  progress = NULL,
  densify = 0.01
)
```

Arguments

polygon	An sf/sfc POLYGON or MULTIPOLYGON object.
resolution	Target grid resolution.
grid	A modern grid from hex_grid() .
include_holes	Whether interior holes exclude cell nuclei.
max_cells	Safety cap applied to the returned result.
progress	Show progress while DGGRID covers the polygon. NULL shows a delayed progress bar in interactive sessions, while TRUE or FALSE always enables or disables it.
densify	Maximum angular spacing in degrees when DGGRID densifies polygon edges. The conservative default is intended for arbitrary input polygons; rectangular raster footprints can safely use a coarser value.

Value

A globalhex_cell vector.

cell_disk	<i>Return cells within grid distance k</i>
-----------	--

Description

Return cells within grid distance k

Usage

```
cell_disk(cell, k = 1L, progress = NULL)
```

Arguments

cell	One or more origin cells.
k	Non-negative graph radius.
progress	Show progress for multi-step traversal. NULL shows a delayed progress bar in interactive sessions, while TRUE or FALSE always enables or disables it.

Value

A tibble with origin, cell, and exact distance columns.

cell_distance	<i>Return shortest neighbor-graph distance</i>
---------------	--

Description

The current implementation uses bounded breadth-first search.

Usage

```
cell_distance(origin, destination, max_k = 100L, progress = NULL)
```

Arguments

origin, destination	Cell vectors. Size-one inputs are recycled.
max_k	Maximum search radius before returning NA_integer_.
progress	Show progress for the bounded search. NULL shows a delayed progress bar in interactive sessions, while TRUE or FALSE always enables or disables it.

Value

An integer vector.

cell_extract	<i>Extract raster summaries into cells</i>
--------------	--

Description

Cell polygons are generated only for the supplied cells. Extraction uses exact raster-pixel coverage through `exactextractr::exact_extract()`. With `fun = NULL`, continuous layers use an area-weighted mean and categorical layers use a coverage-weighted mode. This makes longitude/latitude rasters safe by default, because their pixels do not have equal physical area.

Usage

```
cell_extract(cell, raster, fun = NULL, weights = "area", ..., progress = NULL)
```

Arguments

cell	A <code>globalhex_cell</code> vector.
raster	A <code>terra::SpatRaster</code> .
fun	A named <code>exactextractr</code> summary, a vector of summary names, or a custom summary function. NULL chooses a type-aware default.
weights	Weights for continuous defaults and explicit weighted summaries. Use "area" (the default), NULL for coverage fractions alone, or a compatible <code>terra::SpatRaster</code> .
...	Additional arguments passed to <code>exactextractr::exact_extract()</code> .
progress	Whether <code>exactextractr</code> displays progress. NULL follows <code>interactive()</code> .

Value

A `hex_data()` object with one row per input cell.

cell_grid	<i>Return the grid attached to cells</i>
-----------	--

Description

Return the grid attached to cells

Usage

```
cell_grid(cell)
```

Arguments

cell A globalhex_cell vector.

Value

The `hex_grid()` or `legacy_grid()` shared by every element.

cell_is_valid	<i>Check cell identifiers</i>
---------------	-------------------------------

Description

The default check is local and fast. It checks the two-digit base cell, resolution, hierarchy digits, and deleted pentagon subsequences. Strict validation additionally asks DGGRID (or dggridR for legacy cells) to resolve each syntactically valid identifier.

Usage

```
cell_is_valid(x, grid = NULL, strict = FALSE)
```

Arguments

x A globalhex_cell or character vector.

grid Grid definition for character input. NULL uses `hex_grid()`. The attached grid is always used for globalhex_cell input.

strict Whether to confirm syntactically valid identifiers with the relevant backend.

Value

A logical vector with one value per identifier.

Examples

```
cell_is_valid(c("0203124", "99"))
cell_is_valid(point_to_cell(c(0, 0), 5))
```

cell_neighbors	<i>Return immediate cell neighbors</i>
----------------	--

Description

The return type is stable for one or many origins. Normal hexagons have six rows per origin; pentagons have five.

Usage

```
cell_neighbors(cell)
```

Arguments

cell A globalhex_cell vector.

Value

A tibble with origin and neighboring cell columns.

Examples

```
x <- point_to_cell(c(-73.9857, 40.7484), 5)
cell_neighbors(x)
```

cell_parent	<i>Return cell ancestors</i>
-------------	------------------------------

Description

Return cell ancestors

Usage

```
cell_parent(cell, resolution = NULL)
```

Arguments

cell A globalhex_cell vector.

resolution Target coarser resolution. NULL selects each cell's immediate parent, including for mixed-resolution vectors.

Value

A globalhex_cell vector with one parent per input cell.

cell_resolution	<i>Return cell resolution</i>
-----------------	-------------------------------

Description

Return cell resolution

Usage

```
cell_resolution(cell)
```

Arguments

cell A globalhex_cell vector.

Value

An integer vector, one value per cell.

Examples

```
cell_resolution(as_globalhex_cell(c("0203124", "02031240")))
```

cell_ring	<i>Return cells at exactly grid distance k</i>
-----------	--

Description

Return cells at exactly grid distance k

Usage

```
cell_ring(cell, k = 1L, progress = NULL)
```

Arguments

cell One or more origin cells.
k Non-negative graph distance.
progress Show progress for multi-step traversal. See [cell_disk\(\)](#).

Value

A tibble with origin, cell, and distance columns.

cell_sample	<i>Sample raster values at cell centers</i>
-------------	---

Description

cell_sample() is the fast, geometry-free raster path. Cell centers are transformed into the raster coordinate reference system and sampled with `terra::extract()`. Use `cell_extract()` when partial raster-pixel coverage must be represented exactly.

Usage

```
cell_sample(cell, raster, method = c("simple", "bilinear"))
```

Arguments

cell	A globalhex_cell vector.
raster	A <code>terra::SpatRaster</code> .
method	Either "simple" for nearest-cell lookup or "bilinear".

Value

A `hex_data()` object with one row per input cell and one column per raster layer.

globalhex_info	<i>Report the local globalhex environment</i>
----------------	---

Description

globalhex runs a DGGRID executable compiled when the package is installed. To use a different DGGRID build (version 9.0 or later), set `options(globalhex.dggrid = "/path/to/dggrid")` or the DGGRID_BIN (or DGGRID) environment variable. These take precedence over a dggrid on the PATH, which in turn takes precedence over the bundled copy.

Usage

```
globalhex_info()
```

Value

A named list describing R, optional packages, and the DGGRID executable in use.

hex_data	<i>Construct lazily geometric global-hex data</i>
----------	---

Description

hex_data() keeps indexed cells and ordinary attribute columns together. It deliberately stores no polygon geometry; geometry is generated only for the cells ultimately selected for display.

Usage

```
hex_data(cell, ...)
```

Arguments

cell	A globalhex_cell vector.
...	One or more vectors of cell attributes. Size-one values are recycled by tibble::tibble() .

Value

A globalhex_data tibble.

Examples

```
cells <- point_to_cell(rbind(c(0, 0), c(10, 10)), 5)
x <- hex_data(cells, value = c(1, 2))
x
```

hex_grid	<i>Define a global hexagonal grid</i>
----------	---------------------------------------

Description

hex_grid() describes spatial identity only: the DGGS family, hierarchy, and orientation. Resolution belongs to cells and is supplied to [point_to_cell\(\)](#) or [cell_cover\(\)](#). Backend discovery is deliberately separate from the grid definition.

Usage

```
hex_grid(
  system = c("igeo7", "isea3h"),
  pole_lon = 11.25,
  pole_lat = 58.28252559,
  azimuth = 0
)
```

Arguments

system Either "igeo7" (the equal-area Z7 default) or "isea3h" (equal-area Z3).
 pole_lon, pole_lat, azimuth
 Orientation in degrees. IGEO7 has a canonical orientation; custom values are supported for ISEA3H.

Value

A globalhex_grid object.

Examples

```
hex_grid()  
hex_grid("isea3h")
```

 legacy_grid

Define the explicit legacy dggridR fallback

Description

This opt-in constructor quarantines the older sequence-number behavior. Legacy cells support point, center, boundary, and sf conversion, but not the modern hierarchy or neighbor operations.

Usage

```
legacy_grid(resolution, pole_lon = 11.25, pole_lat = 58.28252559, azimuth = 0)
```

Arguments

resolution ISEA3H resolution.
 pole_lon, pole_lat, azimuth
 Orientation in degrees.

Value

A globalhex_legacy_grid object.

plot.globalhex_data *Plot global-hex values with hierarchical level of detail*

Description

Source polygons are never constructed. The method selects a legitimate Z7 or Z3 ancestor level and aggregates source values before making any display geometry. `render = "auto"` draws smaller results as vector polygons and rasterizes larger results for a compact, device-friendly display.

Usage

```
## S3 method for class 'globalhex_data'
plot(
  x,
  y = NULL,
  value = NULL,
  resolution = NULL,
  max_cells = 100000L,
  fun = NULL,
  na.rm = TRUE,
  lod = TRUE,
  extent = NULL,
  xlim = NULL,
  ylim = NULL,
  device_size = NULL,
  cell_pixels = 3,
  crs = 4326,
  border = NA,
  pal = NULL,
  render = c("auto", "polygon", "raster"),
  raster_threshold = 20000L,
  ...
)
```

Arguments

<code>x</code>	A <code>hex_data()</code> object.
<code>y</code>	Unused; present for the base <code>plot()</code> generic.
<code>value</code>	Attribute column to plot. NULL uses the first attribute.
<code>resolution</code>	Explicit display resolution. NULL selects one automatically.
<code>max_cells</code>	Hard maximum number of display cells.
<code>fun</code>	Aggregation function. NULL uses type-aware defaults; use <code>sum</code> explicitly for counts.
<code>na.rm</code>	Whether aggregation removes missing values.
<code>lod</code>	Whether to use automatic hierarchical level of detail.

extent	Geographic display extent as <code>c(xmin, xmax, ymin, ymax)</code> .
xlim, ylim	Optional conventional longitude and latitude limits. They override the corresponding part of extent and participate in LOD selection before projection.
device_size	Optional display width and height in pixels.
cell_pixels	Approximate minimum cell diameter in screen pixels.
crs	Display CRS understood by <code>sf::st_crs()</code> . Aggregation always happens before projection.
border	Polygon border passed to <code>plot()</code> . Raster rendering cannot draw individual cell borders.
pal	Optional color palette function or vector.
render	Rendering backend. "auto" rasterizes sufficiently large displays when terra is available, "polygon" always draws vector polygons, and "raster" always rasterizes the displayed polygons.
raster_threshold	Number of displayed cells at which automatic rendering switches to a raster. An explicit non-missing border keeps automatic rendering in polygon mode.
...	Additional arguments passed to the selected sf or terra plot method.

Value

The aggregated globalhex_data used for display, invisibly, with plot_resolution, plot_extent, and plot_render attributes. Polygon rendering also attaches the displayed geometry; raster rendering attaches plot_raster_size instead.

Examples

```
if (requireNamespace("sf", quietly = TRUE)) {
  cells <- point_to_cell(rbind(c(0, 0), c(10, 10)), 5)
  plot(hex_data(cells, value = c(1, 2)), max_cells = 1000)
}
```

plot_resolution	<i>Choose a display resolution for global-hex data</i>
-----------------	--

Description

The selector combines a hard cell cap with a screen-pixel budget, then walks the actual cell hierarchy inside the requested geographic extent. Only cell centers are used during selection; polygon boundaries are not generated.

Usage

```
plot_resolution(
  x,
  extent = NULL,
  device_size = NULL,
  max_cells = 100000L,
  cell_pixels = 3,
  lod = TRUE,
  resolution = NULL
)
```

Arguments

x	A hex_data() object.
extent	Geographic extent as c(xmin, xmax, ymin, ymax) in longitude and latitude. xmin > xmax denotes an antimeridian-crossing extent. NULL infers an extent from a bounded set of hierarchical cells.
device_size	Display width and height in pixels. NULL inspects the current graphics device and falls back to 800 by 600 pixels.
max_cells	Hard maximum number of display cells.
cell_pixels	Approximate minimum cell diameter in screen pixels.
lod	Whether to select a coarser level automatically.
resolution	Optional explicit display resolution.

Value

A single display resolution.

point_to_cell	<i>Convert points to cells</i>
---------------	--------------------------------

Description

Coordinates consistently use longitude before latitude. Named data-frame columns and sf POINT objects are preferred for clarity.

Usage

```
point_to_cell(points, resolution = NULL, grid = hex_grid())
```

Arguments

points	Point coordinates as c(lon, lat), a two-column matrix/data frame, or an sf POINT object.
resolution	Target resolution. Required for modern grids; fixed by a legacy_grid() .
grid	A grid definition from hex_grid() or legacy_grid() .

Value

A `globalhex_cell` vector with one cell per input point.

Examples

```
grid <- hex_grid()
point_to_cell(c(-73.9857, 40.7484), resolution = 5, grid = grid)
```

raster_to_cell	<i>Convert a raster footprint to global-hex data</i>
----------------	--

Description

`raster_to_cell()` covers the raster extent with DGGS cells, includes the neighboring edge cells needed for exact overlap, removes non-intersecting candidates, and extracts raster summaries. It returns no polygon geometry.

Usage

```
raster_to_cell(
  raster,
  resolution,
  grid = hex_grid(),
  fun = NULL,
  weights = "area",
  ...,
  max_cells = 100000L,
  progress = NULL,
  drop_na = TRUE
)
```

Arguments

raster	A terra::SpatRaster .
resolution	Target DGGS resolution.
grid	A modern grid from hex_grid() .
fun, weights, ...	Extraction arguments passed to cell_extract() .
max_cells	Hard maximum number of returned cells.
progress	Show progress where supported. NULL uses interactive defaults.
drop_na	Whether to remove cells for which every extracted raster summary is missing.

Value

A [hex_data\(\)](#) object covering the raster extent.

`st_as_sf.globalhex_cell`*Convert cells to sf polygons*

Description

Cell boundaries are spherical polygons: DGGRID reports their vertices, and the edges between them are great-circle arcs. That is exactly what `sf::st_area()` and other spherical (s2) operations expect, but a flat longitude/latitude map draws straight lines between vertices, so cells that cross the antimeridian (180 degrees longitude) would stretch across the whole map and cells containing a pole would draw as twisted shapes.

Usage

```
## S3 method for class 'globalhex_cell'
st_as_sf(x, wrap = TRUE, ...)
```

Arguments

<code>x</code>	A <code>globalhex_cell</code> vector.
<code>wrap</code>	Whether to split antimeridian cells, close polar cells, and densify long edges for flat longitude/latitude use.
<code>...</code>	Passed to methods; currently unused.

Details

With `wrap = TRUE` (the default) the polygons are prepared for flat longitude/latitude use:

- edges longer than one degree are densified along their great circles;
- cells crossing the antimeridian are split into a MULTIPOLYGON whose parts touch longitude -180 and 180;
- cells containing a pole are closed along the map edge at latitude 90 or -90, so they draw as a polar cap.

Longitudes stay in `[-180, 180]` and spherical areas are unchanged. When any cell is split, every geometry is returned as a MULTIPOLYGON so the column has a single type. For a Pacific-centred map, use `sf::st_shift_longitude()` on the result to move longitudes to `[0, 360]`.

Use `wrap = FALSE` for the raw spherical polygons with one ring of DGGRID vertices per cell, for example before projecting to a polar CRS.

Value

An sf object in EPSG:4326 with one row and geometry per cell.

Examples

```
if (requireNamespace("sf", quietly = TRUE)) {  
  seam <- point_to_cell(c(180, 10), 6)  
  sf::st_geometry(sf::st_as_sf(seam))  
  sf::st_geometry(sf::st_as_sf(seam, wrap = FALSE))  
}
```

Index

as_dggridR_dggs, [2](#)
as_globalhex_cell, [3](#)
as_seqnum, [4](#)

cell_aggregate, [4](#)
cell_boundary, [5](#)
cell_center, [5](#)
cell_children, [6](#)
cell_cover, [6](#)
cell_cover(), [13](#)
cell_disk, [7](#)
cell_disk(), [11](#)
cell_distance, [8](#)
cell_extract, [8](#)
cell_extract(), [12](#), [18](#)
cell_grid, [9](#)
cell_is_valid, [9](#)
cell_neighbors, [10](#)
cell_parent, [10](#)
cell_resolution, [11](#)
cell_ring, [11](#)
cell_sample, [12](#)

exactextractr::exact_extract(), [8](#)

globalhex_info, [12](#)

hex_data, [13](#)
hex_data(), [4](#), [9](#), [12](#), [15](#), [17](#), [18](#)
hex_grid, [13](#)
hex_grid(), [3](#), [7](#), [9](#), [17](#), [18](#)

interactive(), [8](#)

legacy_grid, [14](#)
legacy_grid(), [3](#), [9](#), [17](#)

plot(), [15](#), [16](#)
plot.globalhex_data, [15](#)
plot_resolution, [16](#)
point_to_cell, [17](#)
point_to_cell(), [13](#)
raster_to_cell, [18](#)
sf::st_area(), [19](#)
sf::st_crs(), [16](#)
sf::st_shift_longitude(), [19](#)
st_as_sf.globalhex_cell, [19](#)
terra::extract(), [12](#)
terra::SpatRaster, [8](#), [12](#), [18](#)
tibble::tibble(), [13](#)