

# Package: ggmapcn (via r-universe)

January 15, 2025

**Title** Customizable China Map Visualizations

**Version** 0.0.2

**Description** ggmapcn is a ggplot2 extension package for visualizing China's map with customizable projections and styling.

**License** GPL-3

**Encoding** UTF-8

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.3.2

**Depends** R (>= 3.1.0), ggplot2 (>= 3.4.0)

**LazyData** true

**Imports** sf (>= 1.0.0), dplyr (>= 1.1.0), rlang (>= 1.1.3), ggspatial (>= 1.1.8), terra (>= 1.7), tidyterra (>= 0.6.0)

**URL** <https://r imagination.github.io/ggmapcn>

**BugReports** <https://github.com/Rimagination/ggmapcn/issues>

**Suggests** knitr, rmarkdown

**VignetteBuilder** knitr

**News** NEWS.md

**Config/pak/sysreqs** libgdal-dev gdal-bin libgeos-dev libicu-dev libjpeg-dev libpng-dev libssl-dev libproj-dev libsqlite3-dev libudunits2-dev

**Repository** <https://r imagination.r-universe.dev>

**RemoteUrl** <https://github.com/r imagination/ggmapcn>

**RemoteRef** HEAD

**RemoteSha** e4ce773cb9d6558eb52b087837b05da87bf63422

## Contents

basemap_dem . . . . .	2
basemap_vege . . . . .	3
coord_proj . . . . .	5
geom_boundary_cn . . . . .	6
geom_buffer_cn . . . . .	9
geom_loc . . . . .	10
geom_mapcn . . . . .	11
geom_world . . . . .	13

<b>Index</b>	<b>15</b>
--------------	-----------

---

basemap_dem	<i>Elevation Map of China Layer for ggplot2</i>
-------------	---

---

### Description

basemap\_dem adds a digital elevation model (DEM) raster map of China as a layer to ggplot2. The function ensures the output map remains rectangular, regardless of the chosen projection. It supports displaying the DEM either within China's boundary or in a larger rectangular area around China. Users can provide their own DEM data using the data parameter, or the default built-in DEM data will be used.

### Usage

```
basemap_dem(
  data = NULL,
  crs = NULL,
  within_china = FALSE,
  maxcell = 1e+06,
  na.rm = FALSE,
  ...
)
```

### Arguments

data	Optional. A terra raster object for custom DEM data. If NULL (default), the function uses the built-in DEM data (gebco_2024.tif).
crs	Coordinate reference system (CRS) for the projection. Defaults to the CRS of the DEM data. Users can specify other CRS strings (e.g., "EPSG:4326" or custom projections).
within_china	Logical. If TRUE, displays only the DEM within China's boundary. If FALSE, displays the DEM for a larger rectangular area around China. Default is FALSE.
maxcell	Maximum number of cells for rendering (to improve performance). Defaults to 1e6.
na.rm	Logical. If TRUE, removes missing values. Default is FALSE.
...	Additional parameters passed to geom_spatraster.

**See Also**[geom\\_boundary\\_cn](#)**Examples**

```
# Define Azimuthal Equidistant projection centered on China
china_proj <- "+proj=aeqd +lat_0=35 +lon_0=105 +ellps=WGS84 +units=m +no_defs"

# Example 1: Display full rectangular area around China using built-in DEM data
ggplot() +
  basemap_dem(within_china = FALSE) +
  tidyterra::scale_fill_hypso_tint_c(
    palette = "gmt_globe",
    breaks = c(-10000, -5000, 0, 2000, 5000, 8000)
  ) +
  theme_minimal()

# Example 2: Display only China's DEM and boundaries using built-in DEM data
ggplot() +
  basemap_dem(crs = china_proj, within_china = TRUE) +
  geom_boundary_cn(crs = china_proj) +
  tidyterra::scale_fill_hypso_c(
    palette = "dem_print",
    breaks = c(0, 2000, 4000, 6000),
    limits = c(0, 7000)
  ) +
  labs(fill = "Elevation (m)") +
  theme_minimal()

# Example 3: Use custom DEM data by specifying the path to your DEM file
# dem_path <- "path/to/your/dem_file.tif" # Specify the path to your DEM file
dem_path <- system.file("extdata", "gebco_2024.tif", package = "ggmapcn") # Use the built-in DEM data
ggplot() +
  basemap_dem(data = terra::rast(dem_path), within_china = FALSE) +
  theme_minimal()
```

---

**basemap\_vege***Vegetation Map of China Layer for ggplot2*

---

**Description**

Adds a vegetation raster map of China to a ggplot2 plot, with color-coded vegetation types.

**Usage**

```
basemap_vege(
  color_table = NULL,
  crs = NULL,
```

```

    maxcell = 1e+06,
    use_coltab = TRUE,
    na.rm = FALSE,
    ...
  )

```

### Arguments

<code>color_table</code>	A data frame containing vegetation types and their corresponding colors. It should have columns "code" (raster values), "type" (vegetation names), and "col" (hex color codes). If NULL, a default color table based on standard vegetation classifications for China is used.
<code>crs</code>	A character string specifying the coordinate reference system for the projection. If NULL, the default projection "+proj=aeqd +lat_0=35 +lon_0=105 +ellps=WGS84 +units=m +no_defs" is applied.
<code>maxcell</code>	An integer indicating the maximum number of cells for rendering to improve performance. Defaults to 1e6.
<code>use_coltab</code>	A logical value indicating whether to use the color table for raster values. Default is TRUE.
<code>na.rm</code>	A logical value indicating whether to remove missing values. Default is FALSE.
<code>...</code>	Additional parameters passed to <code>geom_spatraster</code> .

### Value

A `ggplot2` layer object representing the vegetation map of China.

### References

Zhang X, Sun S, Yong S, et al. (2007). *Vegetation map of the People's Republic of China (1:1000000)*. Geology Publishing House, Beijing.

### Examples

```

## Not run:
# Add vegetation raster of China to a ggplot
ggplot() +
  basemap_vege() +
  theme_minimal()

# Customize color table
custom_colors <- data.frame(
  code = 0:11,
  type = c(
    "Non-vegetated", "Needleleaf forest", "Needleleaf and broadleaf mixed forest",
    "Broadleaf forest", "Scrub", "Desert", "Steppe", "Grassland",
    "Meadow", "Swamp", "Alpine vegetation", "Cultivated vegetation"
  ),
  col = c(
    "#8D99B3", "#97B555", "#34BF36", "#9ACE30", "#2EC6C9", "#E5CE0E",

```

```

      "#5BB1ED", "#6494EF", "#7AB9CB", "#D97A80", "#B87701", "#FEB780"
    )
  )
  ggplot() +
    basemap_vege(color_table = custom_colors) +
    labs(fill = "Vegetation type group") +
    theme_minimal()

## End(Not run)

```

---

 coord\_proj

*Coordinate System with Transformed Limits for Custom Projections*


---

## Description

coord\_proj is a wrapper around `ggplot2::coord_sf()`. It simplifies specifying map limits (xlim, ylim) in longitude and latitude (WGS84 CRS) and automatically transforms them into the specified CRS for accurate projections.

This function extends the functionality of `coord_sf()` to seamlessly handle user-specified geographic boundaries in any projection, ensuring accurate mapping.

## Usage

```

coord_proj(
  crs = NULL,
  xlim = NULL,
  ylim = NULL,
  expand = TRUE,
  default_crs = "EPSG:4326",
  ...
)

```

## Arguments

crs	A character string specifying the coordinate reference system (CRS) for the projection (e.g., "EPSG:4326" or custom projections like "+proj=merc").
xlim	Longitude range (in degrees) to display, as a numeric vector of length 2.
ylim	Latitude range (in degrees) to display, as a numeric vector of length 2.
expand	Logical, whether to expand the plot limits. Default is TRUE.
default_crs	A character string specifying the CRS of the input xlim and ylim. Default is "EPSG:4326".
...	Additional arguments passed to <code>ggplot2::coord_sf()</code> .

## Value

A ggplot2 coord\_sf object with the transformed limits.

**See Also**

[ggplot2::coord\\_sf, geom\\_world](#)

**Examples**

```
# World map with default projection and limits
ggplot() +
  geom_world() +
  coord_proj(
    crs = "+proj=longlat +datum=WGS84",
    xlim = c(-180, 180),
    ylim = c(-90, 90),
    expand=FALSE
  ) +
  theme_minimal()

# Focused view with Azimuthal Equidistant projection
china_proj <- "+proj=aeqd +lat_0=35 +lon_0=105 +ellps=WGS84 +units=m +no_defs"
ggplot() +
  geom_world(fill = "lightblue") +
  coord_proj(
    crs = china_proj,
    xlim = c(60, 140),
    ylim = c(-10, 50)
  ) +
  theme_minimal()

# Display a small map of the South China Sea Islands with a custom projection
ggplot() +
  geom_boundary_cn() +
  theme_bw() +
  coord_proj(
    crs = china_proj,
    expand = FALSE,
    xlim = c(105, 123),
    ylim = c(2, 23)
  )
)
```

---

geom\_boundary\_cn

*Plot Boundaries of China*

---

**Description**

Draws various types of boundaries for China, including mainland boundaries, coastlines, ten-segment line, special administrative region (SAR) boundaries, and undefined boundaries. Each boundary type can be customized in terms of color, line width, and line type. This function also allows optional addition of a compass and a scale bar.

**Usage**

```
geom_boundary_cn(
  crs = "+proj=aeqd +lat_0=35 +lon_0=105 +ellps=WGS84 +units=m +no_defs",
  compass = FALSE,
  scale = FALSE,
  mainland_color = "black",
  mainland_size = 0.5,
  mainland_linetype = "solid",
  coastline_color = "blue",
  coastline_size = 0.3,
  coastline_linetype = "solid",
  ten_segment_line_color = "black",
  ten_segment_line_size = 0.5,
  ten_segment_line_linetype = "solid",
  SAR_boundary_color = "grey",
  SAR_boundary_size = 0.5,
  SAR_boundary_linetype = "dashed",
  undefined_boundary_color = "black",
  undefined_boundary_size = 0.5,
  undefined_boundary_linetype = "longdash",
  ...
)
```

**Arguments**

<code>crs</code>	Character. Coordinate reference system (CRS) for the projection. Defaults to "+proj=aeqd +lat_0=35 +lon_0=105 +ellps=WGS84 +units=m +no_defs". Users can specify other CRS strings to customize the projection (e.g., "+proj=merc" for Mercator).
<code>compass</code>	Logical. Whether to display a compass (north arrow). Default is FALSE. If set to TRUE, a default compass (north arrow) with <code>ggspatial::north_arrow_fancy_orienteering()</code> will be added to the top-left corner. To customize the compass, use <code>ggspatial::annotation_north_arrow()</code> directly.
<code>scale</code>	Logical. Whether to display a scale bar. Default is FALSE. If set to TRUE, a default scale bar with <code>ggspatial::annotation_scale()</code> will be added to the bottom-left corner. To customize the scale bar, use <code>ggspatial::annotation_scale()</code> directly.
<code>mainland_color</code>	Character. Color for the mainland boundary. Default is "black".
<code>mainland_size</code>	Numeric. Line width for the mainland boundary. Default is 0.5.
<code>mainland_linetype</code>	Character. Line type for the mainland boundary. Default is "solid".
<code>coastline_color</code>	Character. Color for the coastline. Default is "blue".
<code>coastline_size</code>	Numeric. Line width for the coastline. Default is 0.3.
<code>coastline_linetype</code>	Character. Line type for the coastline. Default is "solid".

```

ten_segment_line_color
    Character. Color for the ten-segment line. Default is "black".
ten_segment_line_size
    Numeric. Line width for the ten-segment line. Default is 0.5.
ten_segment_line_linetype
    Character. Line type for the ten-segment line. Default is "solid".
SAR_boundary_color
    Character. Color for the SAR boundary. Default is "grey".
SAR_boundary_size
    Numeric. Line width for the SAR boundary. Default is 0.5.
SAR_boundary_linetype
    Character. Line type for the SAR boundary. Default is "dashed".
undefined_boundary_color
    Character. Color for the undefined boundary. Default is "lightgrey".
undefined_boundary_size
    Numeric. Line width for the undefined boundary. Default is 0.5.
undefined_boundary_linetype
    Character. Line type for the undefined boundary. Default is "longdash".
...
    Additional parameters passed to geom_sf.

```

### Value

A ggplot2 layer (or list of layers) displaying China's multi-segment boundaries with the specified styles, optionally including a compass (north arrow) and a scale bar.

### Examples

```

## Not run:
# Plot China's boundaries with default settings
ggplot() +
  geom_boundary_cn() +
  theme_minimal()

# Plot China's boundaries with a compass and scale bar
ggplot() +
  geom_boundary_cn(compass = TRUE, scale = TRUE) +
  theme_minimal()

# For customized compass or scale bar, use ggspatial directly:
ggplot() +
  geom_boundary_cn() +
  ggspatial::annotation_north_arrow(
    location = "br", style = ggspatial::north_arrow_minimal()
  ) +
  ggspatial::annotation_scale(
    location = "tr", width_hint = 0.3
  ) +
  theme_minimal()

## End(Not run)

```



---

geom_buffer_cn	<i>Plot Buffered Layers for China's Boundary</i>
----------------	--

---

## Description

This function creates a ggplot2 layer for displaying buffered areas around China's boundaries, including both the mainland boundary and the ten-segment line. Buffers with user-defined distances are generated around each boundary, providing flexibility in projection and appearance.

## Usage

```
geom_buffer_cn(  
  mainland_dist = 20000,  
  ten_line_dist = NULL,  
  crs = "+proj=aeqd +lat_0=35 +lon_0=105 +ellps=WGS84 +units=m +no_defs",  
  color = NA,  
  fill = "#D2D5EB",  
  ...  
)
```

## Arguments

mainland_dist	Numeric. The buffer distance (in meters) for the mainland boundary.
ten_line_dist	Numeric. The buffer distance (in meters) for each segment of the ten-segment line. If not specified, it defaults to the same value as mainland_dist.
crs	Character. The coordinate reference system (CRS) for the projection. Defaults to "+proj=aeqd +lat_0=35 +lon_0=105 +ellps=WGS84 +units=m +no_defs". Users can specify other CRS strings to customize the projection (e.g., "+proj=merc" for Mercator).
color	Character. The border color for the buffer area. Default is NA (transparent).
fill	Character. The fill color for the buffer area. Default is "#D2D5EB".
...	Additional parameters passed to geom_sf.

## Value

A ggplot2 layer displaying buffered areas around China's boundaries, with customizable buffer distances for the mainland boundary and the ten-segment line, using the specified projection.

## Examples

```
## Not run:  
# Plot buffers with specified distances for mainland and ten-segment line  
ggplot() +  
  geom_buffer_cn(  
    mainland_dist = 10000,  
    ten_line_dist = 5000
```

```

) +
  theme_minimal()

## End(Not run)

```

---

geom\_loc

*Visualize Spatial Point Data*


---

## Description

geom\_loc is a wrapper around `ggplot2::geom_sf()` designed for visualizing spatial point data. It supports both `sf` objects and tabular data frames with longitude and latitude columns, automatically transforming them into the specified coordinate reference system (CRS).

## Usage

```

geom_loc(
  data,
  lon = NULL,
  lat = NULL,
  crs = "+proj=aeqd +lat_0=35 +lon_0=105 +ellps=WGS84 +units=m +no_defs",
  mapping = aes(),
  ...
)

```

## Arguments

data	A data frame, tibble, or <code>sf</code> object containing spatial point data.
lon	A character string. The name of the longitude column in data (required if data is tabular).
lat	A character string. The name of the latitude column in data (required if data is tabular).
crs	A character string. The target coordinate reference system (CRS) for the data. Defaults to "+proj=aeqd +lat_0=35 +lon_0=105 +ellps=WGS84 +units=m +no_defs".
mapping	Aesthetic mappings created by <code>ggplot2::aes()</code> , such as color or size.
...	Additional parameters passed to <code>ggplot2::geom_sf()</code> , such as size, alpha, or color.

## Details

This function simplifies the process of visualizing spatial data in `ggplot2` by automatically handling CRS transformations and providing an interface for both `sf` and tabular data. If the input is a tabular data frame, it will be converted to an `sf` object using the specified longitude and latitude columns.

See `ggplot2::geom_sf()` for details on additional parameters and aesthetics.

**Value**

A **ggplot2** layer for visualizing spatial point data.

**See Also**

[geom\\_boundary\\_cn](#)

**Examples**

```
# Generate a random dataset with latitude and longitude
set.seed(123)
data_sim <- data.frame(
  Longitude = runif(100, 80, 120),
  Latitude = runif(100, 28, 40),
  Category = sample(c("Type A", "Type B", "Type C"), 100, replace = TRUE)
)

# Visualize the data with China's boundaries
ggplot() +
  geom_boundary_cn() +
  geom_loc(
    data = data_sim, lon = "Longitude", lat = "Latitude",
    mapping = aes(color = Category), size = 1, alpha = 0.7
  ) +
  theme_minimal()
```

---

geom\_mapcn

*Plot China Map with Customizable Options*

---

**Description**

geom\_mapcn provides a flexible interface for visualizing China's administrative boundaries. Users can select administrative levels (province, city, or county), apply custom projections, and filter specific regions.

**Usage**

```
geom_mapcn(
  data = NULL,
  admin_level = "province",
  crs = "+proj=aeqd +lat_0=35 +lon_0=105 +ellps=WGS84 +units=m +no_defs",
  color = "black",
  fill = "white",
  linewidth = 0.5,
  filter_attribute = NULL,
  filter = NULL,
  ...
)
```

**Arguments**

<code>data</code>	An sf object containing China's map data. If NULL, the function loads the package's default map. Users can select provincial, municipal, or county-level maps using the <code>admin_level</code> parameter.
<code>admin_level</code>	A character string specifying the administrative level of the map. Options are "province" (default), "city", or "county". The corresponding GeoJSON files ( <code>China_sheng.geojson</code> , <code>China_shi.geojson</code> , <code>China_xian.geojson</code> ) must be located in the package's <code>extdata</code> folder.
<code>crs</code>	Coordinate Reference System (CRS). Defaults to "+proj=aeqd +lat_0=35 +lon_0=105 +ellps=WGS84 +units=m +no_defs". Users can specify other CRS strings (e.g., "EPSG:4326").
<code>color</code>	Border color. Default is "black".
<code>fill</code>	Fill color. Default is "white".
<code>linewidth</code>	Line width for borders. Default is 0.5.
<code>filter_attribute</code>	Column name for filtering regions (e.g., "name_en").
<code>filter</code>	Character vector of values to filter specific regions (e.g., <code>c("Beijing", "Shanghai")</code> ).
<code>...</code>	Additional parameters passed to <code>geom_sf</code> .

**Value**

A ggplot2 layer for visualizing China's administrative boundaries.

**Examples**

```
# Plot provincial map
ggplot() +
  geom_mapcn() +
  theme_minimal()

# Filter specific provinces
ggplot() +
  geom_mapcn(filter_attribute = "name_en", filter = c("Beijing", "Shanghai"), fill = "red") +
  theme_minimal()

# Use a Mercator projection
ggplot() +
  geom_mapcn(crs = "+proj=merc", linewidth = 0.7) +
  theme_minimal()
```

## Description

`geom_world` is a wrapper around `ggplot2::geom_sf()` designed for visualizing world maps with added flexibility. It allows custom projections, filtering specific countries or regions, and detailed aesthetic customizations for borders and fills.

## Usage

```
geom_world(  
  data = NULL,  
  crs = "+proj=longlat +datum=WGS84",  
  color = "black",  
  fill = "white",  
  linewidth = 0.5,  
  filter_attribute = "SOC",  
  filter = NULL,  
  ...  
)
```

## Arguments

<code>data</code>	An sf object containing world map data. If NULL, the function loads the package's default <code>world.geojson</code> dataset.
<code>crs</code>	A character string. The target coordinate reference system (CRS) for the map projection. Defaults to <code>"+proj=longlat +datum=WGS84"</code> .
<code>color</code>	A character string specifying the border color for administrative boundaries. Default is <code>"black"</code> .
<code>fill</code>	A character string specifying the fill color for administrative areas. Default is <code>"white"</code> .
<code>linewidth</code>	A numeric value specifying the line width for administrative boundaries. Default is 0.5.
<code>filter_attribute</code>	A character string specifying the column name to use for filtering countries or regions. Default is <code>"SOC"</code> , which refers to the ISO 3166-1 alpha-3 country code in the default dataset.
<code>filter</code>	A character vector specifying the values to filter specific countries or regions. Default is NULL.
<code>...</code>	Additional parameters passed to <code>ggplot2::geom_sf()</code> , such as <code>size</code> , <code>alpha</code> , or <code>lty</code> .

## Details

geom\_world simplifies the process of creating world maps by combining the functionality of geom\_sf with user-friendly options for projections, filtering, and custom styling. Key features include:

- **Custom projections:** Easily apply any CRS to the map.
- **Filtering by attributes:** Quickly focus on specific countries or regions.
- **Flexible aesthetics:** Customize fill, borders, transparency, and other visual properties.

## Value

A ggplot2 layer for world map visualization.

## See Also

[ggplot2::geom\\_sf\(\)](#), [sf::st\\_transform\(\)](#), [sf::st\\_read\(\)](#)

## Examples

```
# Plot the default world map
ggplot() +
  geom_world() +
  theme_minimal()

# Apply Mercator projection
ggplot() +
  geom_world(crs = "+proj=merc") +
  theme_minimal()

# Filter specific countries (e.g., China and its neighbors)
china_neighbors <- c("CHN", "AFG", "BTN", "MMR", "LAO", "NPL", "PRK", "KOR",
                    "KAZ", "KGZ", "MNG", "IND", "BGD", "TJK", "PAK", "LKA", "VNM")

ggplot() +
  geom_world(filter = china_neighbors) +
  theme_minimal()

# Background map + Highlight specific region
ggplot() +
  geom_world(fill = "gray80", color = "gray50", alpha = 0.5) +
  geom_world(filter = c("CHN"), fill = "red", color = "black", linewidth = 1.5) +
  theme_minimal()

# Customize styles with transparency and bold borders
ggplot() +
  geom_world(fill = "lightblue", color = "darkblue", linewidth = 1, alpha = 0.8) +
  theme_void()
```

# Index

## \* **ggplot2.utils**

geom\_loc, [10](#)

basemap\_dem, [2](#)

basemap\_vege, [3](#)

coord\_proj, [5](#)

geom\_boundary\_cn, [3](#), [6](#), [11](#)

geom\_buffer\_cn, [9](#)

geom\_loc, [10](#)

geom\_mapcn, [11](#)

geom\_world, [6](#), [13](#)

ggplot2::aes(), [10](#)

ggplot2::coord\_sf, [6](#)

ggplot2::coord\_sf(), [5](#)

ggplot2::geom\_sf(), [10](#), [13](#), [14](#)

sf::st\_read(), [14](#)

sf::st\_transform(), [14](#)