Package 'skymodelr'

September 14, 2026

Type Package

Title Generates and Samples Realistic Terrestrial Atmospheres

Version 0.6.4

Description Generates physically based sky environment maps and radiance samples using the spectral Hosek-Wilkie and Prague atmosphere models. Functions write high-dynamic-range 'OpenEXR' domes in latitude-longitude projections, compute per-direction RGB or 55-channel values, and optionally composite time-accurate star fields and moon phases. Features include automatic sun and moon positioning from date, time and location, support for sea-level and high-altitude observers, wide-spectrum coefficients, and multithreaded C++ acceleration for fast, high-resolution output. For model details, see Hosek and Wilkie (2012) <doi:10.1145/2185520.2185591>, Hosek and Wilkie (2013) <doi:10.1109/MCG.2013.18>, Wilkie et al. (2021) <doi:10.1145/3450626.3459758>, and Vevoda et al. (2022) <doi:10.1111/cgf.14677>.

License GPL-3

Copyright file inst/COPYRIGHTS

Depends R (>= 4.3.0)

Imports Rcpp, swephR, rayvertex (>= 0.14.0), rayimage (>= 0.27.0)

LinkingTo Rcpp, RcppThread, libimath, libopenexr (>= 3.4.12-6)

Encoding UTF-8

LazyData true

Config/build/compilation-database true

Suggests testthat (>= 3.0.0), libopenexr (>= 3.4.12-6), rayrender (>= 0.40.1)

Config/testthat/edition 3

URL https://www.skymodelr.com

BugReports https://github.com/tylermorganwall/skymodelr/issues

Config/roxygen2/version 8.0.0

NeedsCompilation yes

Author Tyler Morgan-Wall [aut, cre, cph],
Petr Vevoda [ctb],
Charles University [cph],
Eric Bruneton [ctb, cph],
Lukas Hosek [ctb, cph],
Alexander Wilkie [ctb, cph]

Maintainer Tyler Morgan-Wall <tylermw@gmail.com>

Repository CRAN

Date/Publication 2026-09-14 21:50:02 UTC

Contents

`calculate_sky_radiance`
Sample Prague sky radiance at a chosen wavelength.

Description

Evaluate the Prague spectral sky model at arbitrary spherical directions and return radiance at a user-specified wavelength. Use `render_mode = "all"` for atmosphere + solar disk, `"atmosphere"` for atmospheric radiance only, or `"sun"` for the solar disk only.

Usage

```
calculate_sky_radiance(
  phi,
  theta,
  lambda_nm,
  altitude = 0,
```

```
  elevation = 10,
  visibility = 50,
  albedo = 0.5,
  azimuth = 90,
  number_cores = 1,
  wide_spectrum = FALSE,
  render_mode = "all"
)
```

Arguments

phi	Azimuthal angle of the sample, degrees. Vectorized. Range 0 to 360.
theta	Vertical angle of the sample, degrees. Vectorized. Range -90 to 90.
lambda_nm	Wavelength in nanometers. Vectorized. Must lie within the loaded Prague dataset range.
altitude	Default 0, vectorized. Altitude of the viewer in meters. Range 0 to 15000.
elevation	Default 10, vectorized. Solar elevation angle above/below the horizon (degrees). Range -4.2 to 90.
visibility	Default 50, vectorized. Range 20 to 131.8. Meteorological range in kilometers for Prague model.
albedo	Default 0.5, vectorized. Range 0 to 1. Ground albedo.
azimuth	Default 90, single value. Solar azimuth (degrees). Defaults South.
number_cores	Default 1. Number of threads to use in computation.
wide_spectrum	Default FALSE. Whether to use the wide-spectrum (55-channel, polarised) coefficients.
render_mode	Default "all". One of "all", "atmosphere", or "sun". Use "all" for atmosphere + solar disk, "atmosphere" for atmospheric radiance only, or "sun" for the solar disk only.

Value

Numeric vector of radiance values at the requested wavelength(s).

Examples

```
lambda_vals = calculate_sky_radiance(
  phi = c(90, 90),
  theta = c(45, 45),
  lambda_nm = 550,
  altitude = c(0, 10000),
  elevation = 20,
  visibility = 80,
  albedo = 0.1
)
cbind(altitude = c(0, 10000), radiance = lambda_vals)
```

`calculate_sky_values` *Sample a direction from the Prague model.*

Description

Evaluate the Prague spectral sky model at arbitrary spherical directions without writing an image, returning radiance-only samples.

Usage

```
calculate_sky_values(
  phi,
  theta,
  altitude = 0,
  elevation = 10,
  visibility = 50,
  albedo = 0.5,
  azimuth = 90,
  number_cores = 1,
  wide_spectrum = FALSE,
  render_mode = "all",
  prague_rgb_correction = TRUE,
  prague_rgb_correction_strength = 1,
  prague_rgb_correction_gain = "auto"
)
```

Arguments

`phi`	Horizontal angle of the sample, degrees. Vectorized. Range 0 to 360.
`theta`	Vertical angle of the sample, degrees. Vectorized. Range -90 to 90.
`altitude`	Default `0`, vectorized. Altitude of the viewer in meters. Range 0 to 15000.
`elevation`	Default `10`, vectorized. Solar elevation angle above/below the horizon (degrees). Range -4.2 to 90.
`visibility`	Default `50`, vectorized. Range 20 to 131.8. Meteorological range in kilometers for Prague model.
`albedo`	Default `0.5`, vectorized. Range 0 to 1. Ground albedo.
`azimuth`	Default `90`, single value. Solar azimuth (degrees). Defaults South.
`number_cores`	Default `1`. Number of threads to use in computation.
`wide_spectrum`	Default `FALSE`. Whether to use the wide-spectrum (55-channel, polarised) coefficients.
`render_mode`	Default `"all"`. One of `"all"`, `"atmosphere"`, or `"sun"`. Use `"all"` for atmosphere + solar disk, `"atmosphere"` for atmospheric radiance only, or `"sun"` for the solar disk only.

`prague_rgb_correction`
: Default `TRUE`. Whether to apply the Prague RGB tint correction to returned RGB values. This correction is only applied to Prague RGB output.

`prague_rgb_correction_strength`
: Default `1`. Strength of the Prague RGB tint correction. Use `0` for no correction and `1` for the calibrated correction.

`prague_rgb_correction_gain`
: Default `"auto"`. Multiplicative linear RGB gain used by the Prague RGB tint correction. `"auto"` uses the calibrated default `c(R = 0.94438727, G = 1.02157200, B = 0.95012063)`.

Value

3-column RGB matrix.

Examples

```
# Generate a basic atmosphere with the Prague model
value_grid = expand.grid(
  phi = seq(0, 360, by = 30),
  theta = seq(0, 90, by = 10),
  altitude = c(0, 10000)
)
vals = calculate_sky_values(
  phi = value_grid$phi,
  theta = value_grid$theta,
  altitude = value_grid$altitude,
  elevation = 45,
  visibility = 120,
  albedo = 0
)
cbind(value_grid, vals)
```

`calculate_sun_brightness` *Sample sun luminance at the disk center.*

Description

Evaluate the Prague or Hosek sky model at the sun center and integrate against the CIE Y curve to return a luminance value. This is useful for relative attenuation (e.g., comparing zenith sun to a low sun).

Usage

```
calculate_sun_brightness(
  elevation = 10,
  azimuth = 90,
  albedo = 0.5,
  turbidity = 3,
  altitude = 0,
  visibility = 50,
  hosek = TRUE,
  wide_spectrum = FALSE,
  lambda_nm = NULL
)
```

Arguments

`elevation`	Default `10`. Solar elevation angle above/below the horizon (degrees). Range `[-4.2, 90]` for Prague, `[0, 90]` for Hosek.
`azimuth`	Default `90`. Solar azimuth (degrees).
`albedo`	Default `0.5`. Ground albedo, range 0 to 1.
`turbidity`	Default `3`. Atmospheric turbidity, range 1.7 to 10 (*Hosek only*).
`altitude`	Default `0`. Observer altitude (m), range 0 to 15000 (*Prague only*).
`visibility`	Default `50`. Meteorological range (km); *Prague only*.
`hosek`	Default `TRUE`. `FALSE` selects the Prague model.
`wide_spectrum`	Default `FALSE`. Use wide-spectrum (55-channel) coefficients for Prague at sea level only.
`lambda_nm`	Optional vector of wavelengths for Hosek sampling.

Value

Numeric scalar of sun luminance (CIE Y, relative scale).

Examples

```
calculate_sun_brightness(elevation = 45, hosek = TRUE)
```

`clear_sky_data` *Clear cached sky data files*

Description

Removes files cached by `download_sky_data()` under `tools::R_user_dir("skymodelr", "data")`.

Usage

```
clear_sky_data(files = NULL, ask = interactive())
```

Arguments

files	Default NULL. Character vector of cached file basenames to remove.
ask	Default interactive(). Whether to ask for confirmation before deleting files.

Value

Invisibly, the paths successfully removed.

Examples

```
clear_sky_data()
```

---

download_sky_data *Download Prague Sky Model Coefficient Data*

Description

This model which allows for sun angles below the horizon, wide spectral ranges including infrared, polarization, as well rendering at altitude) need to download a coefficient file. There are three versions of this dataset, listed here in order of increasing size:

Usage

```
download_sky_data(sea_level = TRUE, wide_spectrum = FALSE)
```

Arguments

sea_level	Default TRUE. Download the sea-level-only data. Set to FALSE to download the full-altitude dataset.
wide_spectrum	Default FALSE. If TRUE, downloads the wide-spectrum (55-channel, polarised) version. Valid only when sea_level = TRUE.

Details

Argument combination	File	Size
sea_level = TRUE, wide_spectrum = FALSE	SkyModelDatasetGround.dat	107MB
sea_level = TRUE, wide_spectrum = TRUE	PragueSkyModelDatasetGroundInfra.dat	574MB
sea_level = FALSE	SkyModelDataset.dat	2.4GB

Value

Invisibly, the full path to the data file.

Examples

```
# Standard (11-channel, sea-level) coefficients
download_sky_data()

# Wide-spectrum sea-level coefficients
download_sky_data(wide_spectrum = TRUE)

# Full altitude-range coefficients
download_sky_data(sea_level = FALSE)
```

---

generate_moon_latlong *Generate the atmosphere with the moon*

Description

Note that this is just a scaled version of `generate_sky()`, scaled down by the luminance of the moon as compared to the sun. This function takes the phase of the moon into account, along with the increase in luminosity around a full moon (known as opposition surge). Moonlight attenuation uses the Rozenberg/Krisciunas-Schaefer airmass approximation with configurable `moon_extinction_kV`.

Usage

```
generate_moon_latlong(
  datetime,
  lat,
  lon,
  filename = NA,
  albedo = 0.5,
  turbidity = 3,
  altitude = 0,
  resolution = 2048,
  number_cores = 1,
  moon_atmosphere = FALSE,
  earthshine = TRUE,
  earthshine_albedo = 0.19,
  solar_irradiance_w_m2 = 1300,
  moon_extinction_kV = 0.172,
  hosek = TRUE,
  wide_spectrum = FALSE,
  visibility = 50,
  moon_texture_width = 801,
  moon_texture_height = 801,
  verbose = FALSE
)
```

Arguments

`datetime`	POSIX-compatible date-time.
`lat`	Observer latitude (degrees N).
`lon`	Observer longitude (degrees E; west < 0).
`filename`	Default `NA`. Path to the image file to write.
`albedo`	Default `0.5`. Ground albedo, range 0 to 1.
`turbidity`	Default `3`. Atmospheric turbidity, range 1.7 to 10 (*Hosek only*).
`altitude`	Default `0`. Observer altitude (m), range 0 to 15000 (*Prague only*).
`resolution`	Default `2048`. Image height in pixels (width = 2 * height).
`number_cores`	Default `1`. CPU threads to use.
`moon_atmosphere`	Default `FALSE`. If `TRUE`, this generates atmospheric scattering from light from the moon.
`earthshine`	Default `TRUE`. If `FALSE`, disable earthshine contribution on the moon.
`earthshine_albedo`	Default `0.19`. Effective Earth albedo term used in the earthshine irradiance approximation.
`solar_irradiance_w_m2`	Default `1300`. Reference solar irradiance at 1 AU (W/m^2) used to normalize earthshine emission intensity.
`moon_extinction_kV`	Default `0.172`. V-band atmospheric extinction coefficient for direct moonlight attenuation.
`hosek`	Default `TRUE`. `FALSE` selects the Prague model.
`wide_spectrum`	Default `FALSE`. 55-channel Prague coefficients (altitude = 0m only).
`visibility`	Default `50`. Meteorological range (km); *Prague only*.
`moon_texture_width`	Default `801`. Internal moon-texture render width.
`moon_texture_height`	Default `801`. Internal moon-texture render height.
`verbose`	Default `FALSE`. Whether to print progress bars/diagnostic info.

Details

The complete lunar disk is calibrated before directions below the horizontal horizon are clipped. The upper limb remains visible after the center sets, with the finite horizon attenuation and tint retained below zero center elevation. The visible segment is not renormalized to the full Moon's brightness. Finite sky-map resolution limits how finely this edge is resolved.

Disk visibility is independent of the atmospheric model's elevation domain: the optional atmospheric component requires center elevation at least zero for Hosek or -4.2 degrees for Prague. Below that domain it is omitted, while any visible lunar limb is still drawn. Horizon depression and refraction are not modeled.

Value

Either the image array, or the array is invisibly returned if a file is written. The array has dimensions `(resolution, 2 * resolution, 4)`.

Note

Writing to non-EXR formats will introduce precision loss because HDR data are quantised to the destination format, and low dynamic range outputs like PNG and JPEG files will not represent the true luminosity values encoded in the array.

Examples

```
# Moonlit sky (Hosek), mid-evening in DC
generate_moon_latlong(
  datetime = as.POSIXct("2025-09-05 19:30:00", tz = "America/New_York"),
  lat = 38.9072,
  lon = -77.0369,
  resolution = 400,
  turbidity = 3,
  verbose = TRUE
) |>
  rayimage::render_exposure(15) |>
  rayimage::plot_image()
```

---

generate_planets *Generate a bright-planet image array*

Description

Build a planetary luminance map aligned with the sky dome for compositing within `generate_sky_latlong()`.

Usage

```
generate_planets(
  datetime,
  lon,
  lat,
  filename = NA,
  resolution = 2048,
  turbidity = 3,
  ozone_du = 300,
  altitude = 0,
  color = FALSE,
  planet_width = 1,
  upper_hemisphere_only = TRUE,
  atmosphere_effects = TRUE,
  number_cores = 1,
```

```
  verbose = FALSE
)
```

Arguments

datetime	POSIXct timestamp used for ephemerides.
lon	Observer longitude in degrees (east positive).
lat	Observer latitude in degrees.
filename	Default NA. Destination image path to write. When NA, the image array is returned without writing.
resolution	Default 2048. Map half-width (image is 2 * resolution by resolution).
turbidity	Atmospheric turbidity for extinction modelling.
ozone_du	Column ozone (Dobson Units) for colour shifts.
altitude	Observer altitude in metres.
color	Render RGB (TRUE) stars or monochrome (FALSE).
planet_width	Approximate point-spread size for planets in pixels.
upper_hemisphere_only	If TRUE, mask pixels below the horizon.
atmosphere_effects	If TRUE, apply atmospheric extinction.
number_cores	CPU threads used for rendering.
verbose	Emit diagnostic output when TRUE.

Value

Either the image array, or the array is invisibly returned if a file is written. The array has dimensions (resolution, 2 * resolution, 4).

Note

Writing to non-EXR formats will introduce precision loss because HDR data are quantised to the destination format, and low dynamic range outputs like PNG and JPEG files will not represent the true luminosity values encoded in the array.

Examples

```
# Basic star field over Washington, DC at a fixed time
generate_planets(
  datetime = as.POSIXct("2025-03-21 02:20:00", tz = "America/New_York"),
  lon = -77.0369,
  lat = 38.9072,
  resolution = 400,
  color = TRUE,
  planet_width = 1,
  atmosphere_effects = TRUE,
  upper_hemisphere_only = TRUE,
```

```
  number_cores = 2
) |>
  rayimage::plot_image()
```

`generate_sky` *Generate a Hosek-Wilkie sky dome array*

Description

Evaluate either the Hosek-Wilkie or Prague analytic sky models and return a high-dynamic-range image array for the given solar configuration. An image file is written only when `filename` is supplied.

Usage

```
generate_sky(
  filename = NA,
  albedo = 0.1,
  turbidity = 3,
  elevation = 10,
  azimuth = 90,
  altitude = 0,
  resolution = 2048,
  number_cores = 1,
  hosek = TRUE,
  wide_spectrum = FALSE,
  visibility = 50,
  verbose = FALSE,
  render_mode = "all",
  below_horizon = TRUE,
  prague_rgb_correction = TRUE,
  prague_rgb_correction_strength = 1,
  prague_rgb_correction_gain = "auto",
  exr_adopted_white = "D60",
  exr_metadata = TRUE
)
```

Arguments

`filename`	Default `NA`. Path to an image file to write. If not given, the array is returned instead.
`albedo`	Default `0.1`. 0.0-1.0 ground albedo. Grass has an albedo of about 0.09, while a landscape covered in snow will have an albedo of 1.0.
`turbidity`	Default `3`. 1.7-10 atmospheric turbidity. Only valid for Hosek model.
`elevation`	Default `10`. Solar elevation above the horizon (degrees).

`azimuth` Default `90`, sun directly east. Solar azimuth (degrees). The left edge of the image faces north and the middle faces south.

`altitude` Default `0`. Altitude of the viewer in meters. Valid range: 0 to 15000. Only valid for the Prague model.

`resolution` Default `2048`. Height of the image. Width is twice this number.

`number_cores` Default `1`. Number of threads to use in computation.

`hosek` Default `TRUE`. Set to `FALSE` to enable the Prague 2021-22 spectral sky model.

`wide_spectrum` Default `FALSE`. Use the 55-channel Prague coefficients (sea level only).

`visibility` Default `50`. Meteorological range in kilometres for Prague model.

`verbose` Default `FALSE`. Whether to print progress bars/diagnostic info.

`render_mode` Default `"all"`. One of `"all"`, `"atmosphere"`, or `"sun"`. Use `"all"` for atmosphere + solar disk, `"atmosphere"` for atmospheric radiance only, or `"sun"` for the solar disk only.

`below_horizon` Default `TRUE`. Whether to sample atmospheric scattering below the horizon, which is non-zero when altitude > 0.

`prague_rgb_correction`
Default `TRUE`. Whether to apply the Prague RGB tint correction. This correction is only applied when `hosek = FALSE`. Use `FALSE` for raw Prague RGB output.

`prague_rgb_correction_strength`
Default `1`. Strength of the Prague RGB tint correction. Use `0` for no correction and `1` for the calibrated correction.

`prague_rgb_correction_gain`
Default `"auto"`. Multiplicative linear RGB gain used by the Prague RGB tint correction. `"auto"` uses the calibrated default `c(R = 0.94438727, G = 1.02157200, B = 0.95012063)`.

`exr_adopted_white`
Default `"D60"`. Adopted neutral white to write to EXR metadata for generated sky maps. Currently supports `"D60"`, `"D65"`, or a numeric XYZ white with Y = 1. This tags the EXR `adoptedNeutral` metadata and does not change pixel values.

`exr_metadata` Default `TRUE`. Whether to attach skymodelr EXR metadata before writing EXR output. Metadata includes sRGB/Rec.709 chromaticities and the selected adopted neutral white.

Details

For Prague RGB output, `prague_rgb_correction = TRUE` applies a fixed linear RGB gain to reduce the small magenta / negative-green tint observed in the RGB projection of Prague sky maps. Set `prague_rgb_correction = FALSE` to recover raw Prague RGB output. The correction is not a white balance and is not applied to wavelength-specific spectral radiance.

EXR metadata: Generated sky maps are linear RGB. When `exr_metadata = TRUE`, skymodelr tags the image with sRGB/Rec.709 chromaticities and an EXR `adoptedNeutral` white. The default adopted neutral is D60. This metadata does not convert pixel values; it informs downstream readers how to treat the scene neutral. Use `exr_adopted_white = "D65"` for D65 adopted neutral, or `exr_metadata = FALSE` to omit skymodelr EXR metadata. The RGB encoding remains linear sRGB / Rec.709 even when the adopted neutral is D60.

Value

Either the image array, or the array is invisibly returned if a file is written. The array has dimensions `(resolution, 2 * resolution, 4)`.

Note

Writing to non-EXR formats will introduce precision loss because HDR data are quantised to the destination format, and low dynamic range outputs like PNG and JPEG files will not represent the true luminosity values encoded in the array.

Examples

```
sky = generate_sky(
  resolution = 8,
  elevation = 30,
  azimuth = 135,
  render_mode = "atmosphere"
)
dim(sky)


# Hosek model (default): clear morning, Sun SE, with solar disk
generate_sky(
  resolution = 400,
  elevation  = 15,
  azimuth    = 135,
  turbidity  = 3,
  render_mode = "all"
) |>
  rayimage::plot_image()

# Same view but hazier and without the solar disk
generate_sky(
  resolution = 400,
  elevation  = 15,
  azimuth    = 135,
  turbidity  = 6,
  render_mode = "atmosphere"
) |>
  rayimage::plot_image()

# Prague model (requires downloaded coefficients)
generate_sky(
  resolution = 400,
  hosek      = FALSE,
  altitude   = 0,
  visibility = 80,
  albedo     = 0.2,
  elevation  = 5,
  azimuth    = 220,
  number_cores = 2
) |>
```

```
  rayimage::plot_image()

generate_sky(
  "raw_prague.exr",
  elevation = 60,
  azimuth = 315,
  hosek = FALSE,
  prague_rgb_correction = FALSE
)
```

`generate_sky_latlong` *Generate a location and time-specific sky dome (optionally with stars)*

Description

Convenience wrapper around `generate_sky()` that:

1. Computes the Sun's apparent position for `datetime`, `lat`, and `lon` (via Swiss Ephemeris / **swephR**).
2. Renders the corresponding sky model.
3. Optionally overlays a star field using `generate_stars()` or moon with `generate_moon_latlong()`.

Usage

```
generate_sky_latlong(
  datetime,
  lat,
  lon,
  filename = NA,
  albedo = 0.5,
  turbidity = 3,
  altitude = 0,
  resolution = 2048,
  number_cores = 1,
  hosek = TRUE,
  wide_spectrum = FALSE,
  visibility = 50,
  stars = FALSE,
  star_width = 1,
  planets = FALSE,
  moon = FALSE,
  moon_atmosphere = FALSE,
  moon_hosek = TRUE,
  render_mode = "all",
  below_horizon = TRUE,
  prague_rgb_correction = TRUE,
```

```
  prague_rgb_correction_strength = 1,
  prague_rgb_correction_gain = "auto",
  exr_adopted_white = "D60",
  exr_metadata = TRUE,
  verbose = FALSE,
  stars_exposure = 0,
  ...
)
```

Arguments

datetime	POSIX-compatible date-time.
lat	Observer latitude (degrees N).
lon	Observer longitude (degrees E; west < 0).
filename	Default NA. Path to the .exr file to write.
albedo	Default 0.5. Ground albedo, range 0 to 1.
turbidity	Default 3. Atmospheric turbidity, range 1.7 to 10 (*Hosek only*).
altitude	Default 0. Observer altitude (m), range 0 to 15000 (*Prague only*).
resolution	Default 2048. Image height in pixels (width = 2 * height).
number_cores	Default 1. CPU threads to use.
hosek	Default TRUE. FALSE selects the Prague model.
wide_spectrum	Default FALSE. 55-channel Prague coefficients (altitude = 0m only).
visibility	Default 50. Meteorological range (km); *Prague only*.
stars	Default FALSE. If TRUE, composite a star field using generate_stars(). Automatically falls back to a pure star render when the black-sky condition is met (see *Details*).
star_width	Default 1. Passed to generate_stars() to control stellar point-spread size.
planets	Default FALSE. If TRUE, composite bright planets via generate_planets().
moon	Default FALSE. If TRUE, overlay a moon render from generate_moon_latlong().
moon_atmosphere	Default FALSE. If TRUE, this generates atmospheric scattering from light from the moon.
moon_hosek	Default TRUE. Whether to use the faster (but less accurate) Hosek model for atmospheric scattering from the moon. Note that the light scattered from the moon is much less intense than the sun, and thus small inaccuracies are much less likely to be noticable.
render_mode	Default "all". One of "all", "atmosphere", or "sun". Use "all" for atmosphere + solar disk, "atmosphere" for atmospheric radiance only, or "sun" for the solar disk only.
below_horizon	Default TRUE. Whether to sample atmospheric scattering below the horizon, which is non-zero when altitude > 0.
prague_rgb_correction	Default TRUE. Whether to apply the Prague RGB tint correction. This correction is only applied when hosek = FALSE. Use FALSE for raw Prague RGB output.

`prague_rgb_correction_strength`
: Default `1`. Strength of the Prague RGB tint correction. Use `0` for no correction and `1` for the calibrated correction.

`prague_rgb_correction_gain`
: Default `"auto"`. Multiplicative linear RGB gain used by the Prague RGB tint correction. `"auto"` uses the calibrated default `c(R = 0.94438727, G = 1.02157200, B = 0.95012063)`.

`exr_adopted_white`
: Default `"D60"`. Adopted neutral white to write to EXR metadata for generated sky maps. Currently supports `"D60"`, `"D65"`, or a numeric XYZ white with Y = 1. This tags the EXR `adoptedNeutral` metadata and does not change pixel values.

`exr_metadata`
: Default `TRUE`. Whether to attach skymodelr EXR metadata before writing EXR output. Metadata includes sRGB/Rec.709 chromaticities and the selected adopted neutral white.

`verbose`
: Default `FALSE`. Whether to print progress bars/diagnostic info.

`stars_exposure`
: Default `0`. Increases star exposure by `2^exposure`. Non-physical, this just controls adjustments for artistic effect.

`...`
: Additional **named** arguments forwarded to `generate_stars()`, and when enabled, `generate_planets()` and `generate_moon_latlong()`. Unmatched names are ignored.

Details

Solar angles - altitude (degrees above the horizon) and azimuth (degrees clockwise from east, so 90 degrees = south) - are derived internally; you never have to supply them directly.

Black-sky rule - With the Prague model the sky radiance is defined only down to -4.2 degrees, and with the Hosek model it is defined only about 0 degrees. Below that the function skips the sky render and writes **only stars** when `stars = TRUE`.

Value

Either the raw data, or the data is invisibly returned if filename is given. The EXR is written to `filename`.

Examples

```
sky = generate_sky_latlong(
  datetime = as.POSIXct("2025-03-21 12:00:00", tz = "America/New_York"),
  lat = 38.9072,
  lon = -77.0369,
  resolution = 8
)
dim(sky)


# Morning sunrise on spring solstice over Washington, DC with Prague model
generate_sky_latlong(
  datetime = as.POSIXct("2025-03-21 06:15:00", tz = "America/New_York"),
```

```
  lat = 38.9072,
  lon = -77.0369,
  number_cores = 2,
  hosek = FALSE
) |>
  rayimage::plot_image()

generate_sky_latlong(
  datetime = as.POSIXct("2025-03-21 12:00:00", tz = "America/New_York"),
  lat = 38.9072,
  lon = -77.0369,
  number_cores = 2
) |>
  rayimage::plot_image()

generate_sky_latlong(
  datetime = as.POSIXct("2025-03-21 18:00:00", tz = "America/New_York"),
  lat = 38.9072,
  lon = -77.0369,
  number_cores = 2
) |>
  rayimage::plot_image()

generate_sky_latlong(
  datetime = as.POSIXct("2025-03-21 18:30:00", tz = "America/New_York"),
  lat = 38.9072,
  lon = -77.0369,
  number_cores = 2,
  hosek = FALSE,
  verbose = TRUE
) |>
  rayimage::render_exposure(exposure = 2) |>
  rayimage::plot_image()
```

`generate_stars` *Generate a star-field array aligned with* `generate_sky()`

Description

Render a star map for a given observer location, time, and atmospheric conditions so it can be composited with `generate_sky()`. Returns a `(resolution, 2 * resolution, 4)` array with an opaque alpha channel. An image file is written only when `filename` is supplied.

Usage

```
generate_stars(
  lon,
  lat,
  datetime,
```

```
  filename = NA,
  resolution = 2048,
  turbidity = 3,
  ozone_du = 300,
  altitude = 0,
  color = TRUE,
  star_width = 1,
  upper_hemisphere_only = TRUE,
  atmosphere_effects = TRUE,
  number_cores = 1
)
```

Arguments

lon	Observer longitude in degrees (east positive).
lat	Observer latitude in degrees.
datetime	POSIXct timestamp used to compute local sidereal time.
filename	Default NA. Path to an image file to write. If NA, the image array is returned without writing.
resolution	Default 2048. Map half-width; the output image is 2 * resolution by resolution.
turbidity	Default 3.0. Atmospheric turbidity controlling aerosol optical depth for extinction/reddening.
ozone_du	Default 300.0. Column ozone in Dobson Units used in atmospheric absorption.
altitude	Default 0.0. Observer altitude above mean sea level in meters.
color	Default TRUE. If TRUE, render RGB star colors; if FALSE, render monochrome luminance.
star_width	Default 1. Approximate stellar point-spread size in pixels (controls apparent star sharpness).
upper_hemisphere_only	Default TRUE. If TRUE, pixels below the local horizon are suppressed to match generate_sky()'s visible hemisphere.
atmosphere_effects	Default TRUE. If TRUE, apply atmospheric extinction and color shift using turbidity, ozone_du, and altitude.
number_cores	Default 1. Number of CPU threads to use.

Value

Either the image array, or the array is invisibly returned if a file is written. The array has dimensions (resolution, 2 * resolution, 4).

Note

Writing to non-EXR formats will introduce precision loss because HDR data are quantised to the destination format, and low dynamic range outputs like PNG and JPEG files will not represent the true luminosity values encoded in the array.

Examples

```
star_map = generate_stars(
  resolution = 8,
  lon = -77.0369,
  lat = 38.9072,
  datetime = as.POSIXct("2025-03-21 02:20:00", tz = "UTC"),
  atmosphere_effects = FALSE,
  number_cores = 1
)
dim(star_map)


# Note: exposure has been increased for all examples (via white_point) for
# ease of visibility in documentation

# Basic star field over Washington, DC at a fixed time
generate_stars(
  resolution = 400,
  lon = -77.0369,
  lat = 38.9072,
  datetime = as.POSIXct("2025-03-21 02:20:00", tz = "America/New_York"),
  color = TRUE,
  star_width = 1,
  atmosphere_effects = TRUE,
  upper_hemisphere_only = TRUE,
  number_cores = 2
) |>
  rayimage::plot_image()

# Monochrome stars, no atmospheric extinction/reddening, full sphere
generate_stars(
  resolution = 400,
  lon = -122.4194,
  lat = 37.7749,
  datetime = as.POSIXct("2025-06-01 08:00:00", tz = "UTC"),
  color = FALSE,
  star_width = 1,
  upper_hemisphere_only = FALSE,
  atmosphere_effects = FALSE
) |>
  rayimage::plot_image()

# Sharper stars (smaller PSF) with ozone/turbidity and altitude
generate_stars(
  resolution = 400,
  lon = 10,
  lat = 45,
  datetime = as.POSIXct("2025-12-01 22:00:00", tz = "UTC"),
  star_width = 0.5,
  turbidity = 3.5,
  ozone_du = 320,
  altitude = 1000,
```

```
  color = TRUE
) |>
  rayimage::plot_image()
```

---

generate_sun_disk *Generate Sun and Moon Disk Radiance Images*

Description

Generate a celestial disk directly at texture resolution, independently of an environment map. The Sun uses the Prague solar radiance profile. The Moon uses the existing lunar surface texture, topocentric phase orientation, earthshine, photometry, and atmospheric extinction.

Usage

```
generate_sun_disk(
  datetime,
  lat,
  lon,
  altitude = 0,
  resolution = 256,
  visibility = 50,
  albedo = 0.5,
  number_cores = 1,
  wide_spectrum = FALSE,
  prague_rgb_correction = TRUE,
  prague_rgb_correction_strength = 1,
  prague_rgb_correction_gain = "auto",
  atmospheric_attenuation = TRUE
)

generate_moon_disk(
  datetime,
  lat,
  lon,
  altitude = 0,
  resolution = 1024,
  visibility = 50,
  albedo = 0.5,
  number_cores = 1,
  wide_spectrum = FALSE,
  prague_rgb_correction = TRUE,
  prague_rgb_correction_strength = 1,
  prague_rgb_correction_gain = "auto",
  earthshine = TRUE,
  earthshine_albedo = 0.19,
```

```
  solar_irradiance_w_m2 = 1300,
  moon_extinction_kV = 0.172,
  atmospheric_attenuation = TRUE
)
```

Arguments

`datetime`	A single `POSIXct` instant. Its time zone is respected.
`lat`	Observer latitude in degrees, between -90 and 90.
`lon`	Observer longitude in degrees, between -180 and 180.
`altitude`	Default `0`. Observer altitude in meters, from 0 to 15000.
`resolution`	Default `256` for the Sun and `1024` for the Moon. Target disk width and height in pixels, an integer of at least 16. The Moon is cropped from a padded raster without resampling; edge coverage can add a few pixels to the returned dimensions.
`visibility`	Default `50`. Prague visibility in kilometers, from 20 to 131.8.
`albedo`	Default `0.5`. Ground albedo, from 0 to 1.
`number_cores`	Default `1`. Positive integer number of worker threads.
`wide_spectrum`	Default `FALSE`. Use the extended Prague spectral data.
`prague_rgb_correction`	Default `TRUE`. RGB correction passed to `calculate_sky_values()`.
`prague_rgb_correction_strength`	Default `1`. Nonnegative correction strength passed to `calculate_sky_values()`.
`prague_rgb_correction_gain`	Default `"auto"`. Automatic correction or three positive RGB gains, as in `calculate_sky_values()`.
`atmospheric_attenuation`	Default `TRUE`. Include atmospheric extinction and tint for the specified observer. Set `FALSE` to return radiance before it enters Earth's atmosphere; the consuming renderer must apply atmospheric transport and horizon visibility. Lunar phase, surface detail, and earthshine are retained. `moon_extinction_kV` is ignored when this is `FALSE`.
`earthshine`	Default `TRUE`. Include Earth-reflected light on the Moon.
`earthshine_albedo`	Default `0.19`. Nonnegative Earth albedo used for earthshine.
`solar_irradiance_w_m2`	Default `1300`. Positive solar irradiance in watts per square meter used to illuminate the lunar surface.
`moon_extinction_kV`	Default `0.172`. Nonnegative lunar atmospheric extinction coefficient in magnitudes per airmass.

Details

Prague data must be installed with `download_sky_data()`. The atmosphere is evaluated for one observer altitude and time when `atmospheric_attenuation` is `TRUE`. With `FALSE`, solar radiance comes from the model's intrinsic solar spectrum and lunar radiance uses unattenuated photometry without the atmospheric tint. Solar texture values are then independent of altitude and solar elevation; ephemeris placement and lunar phase still use the requested location and time. Neither mode clips the texture to a geometric horizon. Pair these images with an atmosphere-only sky and disable the corresponding rasterized celestial bodies in that sky to avoid counting their light twice.

The disk is inscribed in the image rectangle. For column `j`, row `i`, width `w`, and height `h`, pixel centers have coordinates `x = 2 * (j - 0.5) / w - 1` and `y = 1 - 2 * (i - 0.5) / h`. Sample only within `x^2 + y^2 <= 1`. The corresponding ray direction is `normalize(forward + tan(radius) * (x * right + y * up))`, where `radius = angular_diameter_deg * pi / 360`. Image-up is the projection of local vertical into the disk plane; image-right points toward increasing azimuth. At the zenith or nadir, north replaces the vertical reference axis. There is no image reprojection or implicit rotation in the return value.

The solar model has a fixed angular profile, mapped here to the apparent ephemeris diameter without changing radiance. Lunar radiance is normalized over disk solid angle to the existing phase-dependent irradiance, with the solar spectral RGB distribution. With `atmospheric_attenuation = TRUE`, lunar attenuation and Prague tint are evaluated at the disk center. At or below zero center elevation, their finite horizon values are retained so the upper limb does not disappear prematurely. This continuation preserves a full disk texture; it is not a model of below-horizon visibility or horizon depression at altitude. Geometric horizon clipping is the consumer's responsibility; sample the full disk and discard directions below the horizon when appropriate.

To save an EXR without changing radiance, tag `image` with `rayimage::ray_read_image(image, normalize = FALSE, source_linear = TRUE, assume_colorspace = rayimage::CS_SRGB)` and write with `clamp = FALSE`.

Value

A list containing:

- `image`: a numeric array with dimensions `c(height, width, 3)` containing linear sRGB radiance, with no exposure, tone mapping, or alpha channel. Antialiased lunar coverage is already applied: do not multiply by alpha. Out-of-gamut solar RGB values may be negative and should be preserved.
- `azimuth_deg`: disk-center azimuth, clockwise from true north, in [0, 360).
- `elevation_deg`: disk-center elevation above the local horizon, in degrees.
- `angular_diameter_deg`: topocentric apparent angular diameter in degrees.
- `atmospheric_attenuation`: whether atmospheric filtering is included.
- `projection`: `"rectilinear"`, using the disk mapping described below.

Examples

```
time = as.POSIXct("2026-01-28 21:00:00", tz = "Pacific/Auckland")
moon = generate_moon_disk(time, lat = -36.87593, lon = 174.7647)
moon[c("azimuth_deg", "elevation_deg", "angular_diameter_deg")]
# Let a renderer evaluate Earth's atmosphere at each interaction instead.
```

```
intrinsic_moon = generate_moon_disk(time, lat = -36.87593, lon = 174.7647,
  atmospheric_attenuation = FALSE)
```

`get_prague_sky_metadata` *Prepare Location and Time Metadata for Native Prague Sky Queries*

Description

Resolve the full-altitude Prague dataset and the Sun's topocentric position without loading coefficients or generating an image. This lets rendering libraries evaluate the model natively while using skymodelr's ephemeris and RGB calibration through a public interface.

Usage

```
get_prague_sky_metadata(
  datetime,
  lat,
  lon,
  altitude = 0,
  visibility = 50,
  albedo = 0.5,
  prague_rgb_correction = TRUE,
  prague_rgb_correction_strength = 1,
  prague_rgb_correction_gain = "auto"
)
```

Arguments

`datetime`	A single `POSIXct` instant. Its time zone is respected.
`lat`	Observer latitude in degrees, between -90 and 90.
`lon`	Observer longitude in degrees, between -180 and 180.
`altitude`	Default `0`. Reference altitude in meters, from 0 to 15000.
`visibility`	Default `50`. Meteorological visibility in kilometers, from 20 to 131.8.
`albedo`	Default `0.5`. Ground albedo, from 0 to 1.
`prague_rgb_correction`	Default `TRUE`. Apply the Prague RGB calibration, as in `calculate_sky_values()`.
`prague_rgb_correction_strength`	Default `1`. Nonnegative calibration strength.
`prague_rgb_correction_gain`	Default `"auto"`. Automatic calibration or three positive RGB gains, as in `calculate_sky_values()`.

Details

Install the data explicitly with `download_sky_data(sea_level = FALSE)`. This function never downloads files or prompts. It supports the visible-spectrum, full-altitude dataset and returns the actual Sun position even below the Prague model's minimum elevation of -4.2 degrees. Native consumers should use zero solar radiance below that limit while retaining transmission for other light sources. This function does not evaluate radiance, attenuation, or atmospheric refraction.

Value

A list with `filename` (the installed full-altitude coefficient file), `elevation_deg`, `azimuth_deg` (clockwise from north), `angular_diameter_deg`, `rgb_gain` (the applied R, G, B gains), and the reference `altitude`, `visibility`, and `albedo`. No native pointers are included; the result can be serialized. The coefficient path is local to this computer and should be resolved again on another machine.

Examples

```
info = get_prague_sky_metadata(
  as.POSIXct("2026-06-21 20:00:00", tz = "America/New_York"),
  lat = 40.7, lon = -74
)
info[c("elevation_deg", "azimuth_deg", "rgb_gain")]
```

---

list_sky_data *List cached sky data files*

Description

Lists files cached by `download_sky_data()` under `tools::R_user_dir("skymodelr", "data")`.

Usage

```
list_sky_data()
```

Value

A data frame with columns `file`, `path`, `size`, and `modified`.

Examples

```
list_sky_data()
```

Index