

# Package ‘LightFitR’

May 7, 2026

**Type** Package

**Title** Design Complex Light Regimes

**Version** 1.0.0

**Description** A system for accurately designing complex light regimes using LEDs.  
Takes calibration data and user-defined target irradiances and it tells you what intensities to use.  
For more details see Vong et al. (2025) <[doi:10.1101/2025.06.06.658293](https://doi.org/10.1101/2025.06.06.658293)>.

**License** GPL (>= 3)

**Encoding** UTF-8

**URL** <https://github.com/ginavong/LightFitR/>

**BugReports** <https://github.com/ginavong/LightFitR/issues>

**LazyData** true

**Depends** R (>= 3.5)

**Imports** utils, graphics, lubridate, nns, stringr

**RoxygenNote** 7.3.2

**Suggests** knitr, rmarkdown, testthat (>= 3.0.0)

**Config/testthat/edition** 3

**VignetteBuilder** knitr

**NeedsCompilation** no

**Author** Gina Vong [aut, cre, cph] (ORCID:  
<<https://orcid.org/0000-0002-3913-7667>>)

**Maintainer** Gina Vong <[gywv500@york.ac.uk](mailto:gywv500@york.ac.uk)>

**Repository** CRAN

**Date/Publication** 2025-07-25 16:30:14 UTC

## Contents

calibration . . . . .	2
checkRange . . . . .	3
example_closest . . . . .	4
example_intensities . . . . .	4
example_regime . . . . .	5
helio.checkFormat . . . . .	5
helio.csv_schedule . . . . .	6
helio.disco . . . . .	6
helio.dyna.leds . . . . .	7
helio.eventLimit . . . . .	8
helio.json_schedule . . . . .	8
internal.closestIntensities . . . . .	9
makeRegime . . . . .	10
nls_intensities . . . . .	11
read.helio_json . . . . .	12
sle_intensities . . . . .	13
target_irradiance . . . . .	14
time_vector . . . . .	14
write.helioSchedule . . . . .	15
<b>Index</b>	<b>16</b>

---

calibration	<i>calibration data</i>
-------------	-------------------------

---

### Description

Example calibration data

### Usage

calibration

### Format

A data frame with 12 columns:

**filename** File that the raw data came from

**time** The time when a given measurement was taken

**led** LED channel being calibrated at that timepoint

**intensity** Intensity the light is set to

**wavelength** The wavelength this row describes

**irradiance** The irradiance measured at that wavelength by the spectrometer

### Source

<[https://github.com/ginavong/2024\\_LightFitR\\_MethodsPaper/blob/master/data/heliospectra\\_measurements/calibration/Apo](https://github.com/ginavong/2024_LightFitR_MethodsPaper/blob/master/data/heliospectra_measurements/calibration/Apo)

---

`checkRange`*Check that the intended irradiances are achievable by the lights*

---

**Description**

Check that the intended irradiances are achievable by the lights

**Usage**

```
checkRange(  
  intended_irradiance,  
  calibration_leds,  
  calibration_wavelengths,  
  calibration_intensities,  
  calibration_irradiances  
)
```

**Arguments**

`intended_irradiance`

Matrix of intended irradiances to be checked. Each row corresponds to an LED channel.

`calibration_leds`

A numeric vector of LED values from calibration, mapping to intensities and irradiances (i.e. the same length)

`calibration_wavelengths`

A numeric vector of wavelengths from calibration, corresponding to intensities and irradiances

`calibration_intensities`

A numeric vector of intensities (heliospectra units) from calibration

`calibration_irradiances`

A numeric vector of measured irradiances (any units, as long as it is consistently used) from calibration

**Value**

Boolean: TRUE = pass, FALSE = fail

**Examples**

```
calib <- LightFitR::calibration  
irradiances <- LightFitR::target_irradiance  
checkRange(irradiances, calib$led, calib$wavelength, calib$intensity, calib$irradiance)
```

---

example_closest	<i>closest intensities</i>
-----------------	----------------------------

---

**Description**

Matrix of closest intensities for example purposes. Generated from 'target\_irradiance'

**Usage**

```
example_closest
```

**Format**

A matrix with 9 rows and 10 columns: each row represents an LED channel and each column represents an event

---

example_intensities	<i>example intensities</i>
---------------------	----------------------------

---

**Description**

Matrix of random intensities for example purposes

**Usage**

```
example_intensities
```

**Format**

A matrix with 9 rows and 10 columns: each row represents an LED channel and each column represents an event

---

example_regime	<i>regime matrix</i>
----------------	----------------------

---

**Description**

Example regime matrix

**Usage**

```
example_regime
```

**Format**

A matrix with 13 rows and 10 columns:

**time** time in HH:MM:SS format

**hour** The hour of the event

**minute** The minute of the event

**second** The second of the event

**380nm** Intensity at 380nm LED channel

**400nm** Intensity at 400nm LED channel

**etc.**

---

helio.checkFormat	<i>Check formatting of the heliospectra matrices</i>
-------------------	--

---

**Description**

Heliospectra DYNA matrices should have 9 rows (1 for each LED channel) and up to 150 columns (max number of events that Heliospectra DYNA can store)

**Usage**

```
helio.checkFormat(check_matrix)
```

**Arguments**

**check\_matrix** Matrix to be checked. Rows correspond to LEDs and columns are events / time-points.

**Value**

Vector of booleans: TRUE = pass, FALSE = fail

## Examples

```
matrix_to_check <- LightFitR::target_irradiance
helio.checkFormat(target_irradiance)
```

---

helio.csv_schedule	<i>Format regime_matrix for csv output that Heliospectra lights can parse</i>
--------------------	---

---

## Description

Format regime\_matrix for csv output that Heliospectra lights can parse

## Usage

```
helio.csv_schedule(regime_matrix, filename)
```

## Arguments

regime_matrix	Matrix containing light regime, as generated by makeRegime
filename	Character. Filename to export to

## Value

Matrix formatted for Heliospectra lights csv

## Examples

```
tempfile_name = tempfile(fileext='.csv')
helio.csv_schedule(LightFitR::example_regime, tempfile_name)
```

---

helio.disco	<i>Turn Heliospectra into disco lights</i>
-------------	--

---

## Description

Congratulations! You've found the easter egg function!

## Usage

```
helio.disco(filename, format = c("csv", "json"))
```

## Arguments

**filename** Character. Filename to export to  
**format** Character. Which format to export to? csv or json. Use extensions '.csv' or '.txt'

## Details

This writes a schedule for Heliospectra DYNA lights which randomly changes colour every second for a fun disco effect. The disco will last until the Heliospectra runs out of memory (150 events), so you can get 2 min 30s of disco out of your expensive lights... Enjoy!

## Value

Disco schedule file for the heliospectra

## Examples

```
tempfile_name = tempfile(fileext='.txt')
helio.disco(tempfile_name, format='json')
```

---

helio.dyna.leds      *heliospectra DYNA LEDs*

---

## Description

Data about the heliospectra DYNA LED channels

## Usage

```
helio.dyna.leds
```

## Format

A data frame with 9 rows and 3 columns:

**name** Name of the LED channel

**wavelength** Wavelength of the LED channel

**colour** Colour of the LED channel

## Source

<<https://heliospectra.com/led-grow-lights/dyna/>>

---

helio.eventLimit	<i>Maximum number of events</i>
------------------	---------------------------------

---

**Description**

Maximum number of events programmable onto heliospectra

**Usage**

```
helio.eventLimit
```

**Format**

Integer representing maximum allowable events

---

helio.json_schedule	<i>Format regime_matrix for json output that Heliospectra lights can parse</i>
---------------------	--

---

**Description**

Format regime\_matrix for json output that Heliospectra lights can parse

**Usage**

```
helio.json_schedule(regime_matrix, filename)
```

**Arguments**

regime_matrix	Matrix containing light regime, as generated by makeRegime
filename	Character. Filename to export to

**Value**

Character in json format that Heliospectra can parse

**Examples**

```
tempfile_name = tempfile(fileext='.txt')  
helio.csv_schedule(LightFitR::example_regime, tempfile_name)
```

---

`internal.closestIntensities`

*Internal function. Find the intensities corresponding to the closest irradiance match between intended and calibration.*

---

## Description

Internal function. Find the intensities corresponding to the closest irradiance match between intended and calibration.

## Usage

```
internal.closestIntensities(  
  irradiance_matrix,  
  calibration_df,  
  peaks = LightFitR::helio.dyna.leds$wavelength  
)
```

## Arguments

`irradiance_matrix` Matrix of intended irradiances. rows = leds and columns = events

`calibration_df` Dataframe of calibration data generated by `internal.calibCombine`

`peaks` Vector of length 8 or 9. Containing wavelengths at which each LED peaks.

## Value

Matrix of closest intensities, in the same format as ‘`irradiance_matrix`’

## Examples

```
# Format calibration data correctly  
calib <- LightFitR::calibration[, c(3, 5, 4, 6)]  
  
# Run function  
internal.closestIntensities(LightFitR::target_irradiance, calib)
```

---

makeRegime	<i>Create a regime (matrix) to program lights to achieve intended irradiances</i>
------------	---

---

### Description

This is a wrapper function that carries out multiple steps:

1. Calculate closest intensities
2. Predict the intensities to use to achieve the target irradiance (via a system of linear equations or non-negative least squares)
3. Tidy the intensities (rounding to integer, keep within the range of intensities that the lights can be set to)
4. Format the intensities and timestamps into a human-readable regime matrix

### Usage

```
makeRegime(
  timeVector_POSIXct,
  irradiance_matrix,
  calibration_leds,
  calibration_wavelengths,
  calibration_intensities,
  calibration_irradiances,
  peaks = LightFitR::helio.dyna.leds$wavelength,
  method = "nnls"
)
```

### Arguments

timeVector_POSIXct	Vector of schedule timepoints in POSIXct format
irradiance_matrix	Matrix of intended irradiances. rows = leds and columns = events
calibration_leds	A numeric vector of LED values from calibration, mapping to intensities and irradiances (i.e. the same length)
calibration_wavelengths	A numeric vector of wavelengths from calibration, corresponding to intensities and irradiances
calibration_intensities	A numeric vector of intensities (heliospectra units) from calibration
calibration_irradiances	A numeric vector of measured irradiances (any units, as long as it is consistently used) from calibration
peaks	Vector of length 8 or 9. Containing wavelengths at which each LED peaks.
method	Use 'nnls' (non-negative least squares) or 'sls' (system of linear equations)

**Value**

Matrix with light regime needed to program the lights

**NNLS vs SLE**

NNLS and SLE largely predict the same intensities, except in outlier cases. The default is NNLS, but if your predicted intensities end up being very far off, try SLE.

**Examples**

```
# Prep variables
calib <- LightFitR::calibration
times <- LightFitR::time_vector
target_irradiance <- LightFitR::target_irradiance

# Run function
makeRegime(times, target_irradiance, calib$led, calib$wavelength, calib$intensity, calib$irradiance)
```

---

npls_intensities	<i>Use non-linear least squares to interpolate intensities</i>
------------------	--

---

**Description**

Use non-linear least squares to interpolate intensities

**Usage**

```
npls_intensities(
  irradiance_matrix,
  closest_intensities,
  calibration_leds,
  calibration_wavelengths,
  calibration_intensities,
  calibration_irradiances,
  peaks = LightFitR::helio.dyna.leds$wavelength
)
```

**Arguments**

```
irradiance_matrix
      Matrix of intended irradiances. rows = leds and columns = events
closest_intensities
      Matrix of closest intensities, generated by 'internal.closestIntensities'. Same
      format as 'irradiance_matrix'
```

calibration\_leds  
A numeric vector of LED values from calibration, mapping to intensities and irradiances (i.e. the same length)

calibration\_wavelengths  
A numeric vector of wavelengths from calibration, corresponding to intensities and irradiances

calibration\_intensities  
A numeric vector of intensities (heliospectra units) from calibration

calibration\_irradiances  
A numeric vector of measured irradiances (any units, as long as it is consistently used) from calibration

peaks  
Vector of length 8 or 9. Containing wavelengths at which each LED peaks.

**Value**

Matrix of intensities to set the lights to, to achieve desired irradiances

**Examples**

```
# Prep variables
target_irradiance = LightFitR::target_irradiance
closest = LightFitR::example_closest
calib = LightFitR::calibration

# Run the function
npls_intensities(target_irradiance, closest,
  calib$led, calib$wavelength, calib$intensity, calib$irradiance)
```

---

read.helio\_json      *Read a heliospectra script (json format) into a matrix.*

---

**Description**

Read a heliospectra script (json format) into a matrix.

**Usage**

```
read.helio_json(helio_script)
```

**Arguments**

helio\_script      File (.txt or .json) containing heliospectra regime script

**Value**

Matrix containing the regime encoded by the Heliospectra script

**Examples**

```
example_file <- system.file("extdata", "example_json_schedule.txt",
  package = "LightFitR", mustWork = TRUE)
read.helio_json(example_file)
```

---

sle\_intensities      *Use a system of linear equations to calculate intensities*

---

**Description**

Use a system of linear equations to calculate intensities

**Usage**

```
sle_intensities(
  irradiance_matrix,
  closest_intensities,
  calibration_leds,
  calibration_wavelengths,
  calibration_intensities,
  calibration_irradiances,
  peaks = LightFitR::helio.dyna.leds$wavelength
)
```

**Arguments**

**irradiance\_matrix**  
Matrix of intended irradiances. rows = leds and columns = events

**closest\_intensities**  
Matrix of closest intensities, generated by ‘internal.closestIntensities’. Same format as ‘irradiance\_matrix’

**calibration\_leds**  
A numeric vector of LED values from calibration, mapping to intensities and irradiances (i.e. the same length)

**calibration\_wavelengths**  
A numeric vector of wavelengths from calibration, corresponding to intensities and irradiances

**calibration\_intensities**  
A numeric vector of intensities (heliospectra units) from calibration

**calibration\_irradiances**  
A numeric vector of measured irradiances (any units, as long as it is consistently used) from calibration

**peaks**  
Vector of length 8 or 9. Containing wavelengths at which each LED peaks.

**Value**

Matrix of intensities to set the lights to, to achieve desired irradiances

**Examples**

```
#' # Prep variables
target_irradiance = LightFitR::target_irradiance
closest = LightFitR::example_closest
calib = LightFitR::calibration

# Run the function
sle_intensities(target_irradiance, closest,
  calib$led, calib$wavelength, calib$intensity, calib$irradiance)
```

---

target_irradiance	<i>target irradiances</i>
-------------------	---------------------------

---

**Description**

Matrix of random target irradiances for example purposes

**Usage**

```
target_irradiance
```

**Format**

A matrix with 9 rows and 10 columns: each row represents an LED channel and each column represents an event

---

time_vector	<i>time vector</i>
-------------	--------------------

---

**Description**

Example timepoints for events

**Usage**

```
time_vector
```

**Format**

A vector of length 10 with timepoints in POSIXct format

---

write.helioSchedule    *Write the schedule to file that Heliospectra can parse*

---

**Description**

Writes to json or csv format

**Usage**

```
write.helioSchedule(regime_matrix, filename, format = c("csv", "json"))
```

**Arguments**

regime_matrix	Matrix containing light regime, as generated by makeRegime
filename	Character. Filename to export to
format	Character. Which format to export to? csv or json. Use extensions '.csv' or '.txt'

**Value**

Heliospectra schedule file in either the csv or json format

**Examples**

```
tempcsv_name = tempfile(fileext='.csv')
write.helioSchedule(LightFitR::example_regime, tempcsv_name, format='csv')

temptxt_name = tempfile(fileext='.txt')
write.helioSchedule(LightFitR::example_regime, temptxt_name, format='json')
```

# Index

## \* datasets

- calibration, 2
- example\_closest, 4
- example\_intensities, 4
- example\_regime, 5
- helio.dyna.leds, 7
- helio.eventLimit, 8
- target\_irradiance, 14
- time\_vector, 14

calibration, 2

checkRange, 3

example\_closest, 4

example\_intensities, 4

example\_regime, 5

helio.checkFormat, 5

helio.csv\_schedule, 6

helio.disco, 6

helio.dyna.leds, 7

helio.eventLimit, 8

helio.json\_schedule, 8

internal.closestIntensities, 9

makeRegime, 10

npls\_intensities, 11

read.helio\_json, 12

sle\_intensities, 13

target\_irradiance, 14

time\_vector, 14

write.helioSchedule, 15