

1. Overview

Running the **StreamLight** model requires a parameter file that describes various site characteristics and a driver file that contains inputs into the model. For convenience, a series of functions have been included in the companion package **StreamLightUtils** to derive some of the values required for parameter files and to create standardized driver files for use with **StreamLight**. Remotely sensed data products with good broadscale coverage are used to derive these inputs within **StreamLightUtils**, but there are of course many potential sources for similar data that could be used to create driver files and users are welcome to create their own workflow for creating driver or parameter files. This tutorial uses two sites from North Carolina to work through using **StreamLightUtils** to create standardized model driver files, creating parameter files, and generating estimates using **StreamLight**. An outline of topics covered in this tutorial are presented below:

Usage: If you use **StreamLight** in your research please cite this package and the associated manuscript in *Freshwater Science* (Savoy et al. 2020, *in press*). To see the suggested citation for this package, run `citation("StreamLight")` in the R prompt.

A fully indexed .pdf version of this README file is also located in the root directory for the package.

Outline

1. Overview and getting started

- 1.1 Overview of the **stream_light** function
- 1.2 Getting started

2. Preparing a driver file

- 2.1 Downloading and processing total incoming shortwave radiation
- 2.2 Downloading and processing MODIS LAI data
- 2.3 Creating a standardized driver file

3. Preparing a parameter file

- 3.1 Parameter descriptions and values

4. Running the StreamLight model

- 4.1 Running the model
- 4.2 Aggregating hourly estimates to daily

1.1 Overview of the **stream_light** function

First, let's take a look at the **stream_light** function which has the following structure:

stream_light(*driver_file*, *Lat*, *Lon*, *channel_azimuth*, *bottom_width*, *BH*, *BS*, *WL*, *TH*, *overhang_height*, *x_LAD*)

- *driver_file* The model driver file
- *Lat* The site latitude
- *Lon* The site longitude
- *channel_azimuth* Channel azimuth

- *bottom_width* Bottom width (m)
- *BH* Bank height (m)
- *BS* Bank slope
- *WL* Water level (m)
- *TH* Tree height (m)
- *overhang* Maximum canopy overhang (m)
- *overhang_height* Height of the maximum canopy overhang (m). If *overhang_height* = NA, then the model defaults to a value of 75% of tree height.
- *x_LAD* Leaf angle distribution, default = 1

The first argument for the function (*driver_file*) is a standardized model driver file that contains total

incoming irradiance (W m^{-2}) and leaf area index (LAI) ($\text{m}^2 \text{m}^{-2}$) which are used as model inputs. The remaining arguments in the function are parameters that describe site characteristics. On the surface this seems like a large number of parameters; however, section 3 of this tutorial provides more indepth information on each of these parameters and some simplifying assumptions that can be used to reduce the number of necessary parameters.

1.2 Getting started

For first time installation run the following code:

```
#Install the devtools package if you do not already have it
install.packages("devtools")

#Use the devtools package to install StreamLightUtils
devtools::install_github("psavoy/StreamLightUtils")
devtools::install_github("psavoy/StreamLight")
```

Before beginning, Load the **StreamLightUtils** and **StreamLight** libraries.

```
library("StreamLightUtils")
library("StreamLight")
```

2. Preparing a driver file

There are two necessary components to drive the **StreamLight** model. First, incoming above canopy total irradiance (W m^{-2}) is needed as an input into the radiative transfer model. Second, daily estimates of LAI are needed to determine the attenuation of light by canopies within the radiative transfer model. This section walks through downloading and processing this data into a standardized model driver file.

2.1 Downloading and processing total incoming shortwave radiation (W m^{-2})

Total incoming shortwave radiation (W m^{-2}) is provided by the National Land Data Assimilation System (NLDAS) at hourly timesteps at 0.125 degree spatial resolution. There are two potential workflows available: i.) download data for a single site, and ii.) bulk download of data for many sites.

2.1.1. Downloading and processing NLDAS data for a single site

Incoming shortwave radiation data at a single site can be downloaded using the function **NLDAS_DL** which has the following structure:

NLDAS_DL(*save_dir*, *Site_ID*, *Lat*, *Lon*, *startDate*)

- *save_dir* The save directory for the downloaded data to be placed in. For example, "C:/"
- *Site_ID* The site identifier ("Site_ID")
- *Lat* The site latitude
- *Lon* The site longitude
- *startDate* The start date for the downloaded data ("YYYY-MM-DD")

Once the data has been downloaded it requires some processing to format date and time information and extract the relevant data. The downloaded NLDAS data can be processed using the function **NLDAS_proc** which has the following structure:

NLDAS_proc(*read_dir*, *Site_IDs*)

- *read_dir* The directory containing the downloaded NLDAS data
- *Site_IDs* The Site ID(s) ("Site_ID")
write_output A logical value indicating whether the output should be written to disk (write_output = TRUE) or returned to the R environment (write_output = FALSE). The default value is FALSE since for most datasets this is a suitable approach; however, for very large datasets (thousands of sites) it may be easier to write files to disk instead of storing them in the workspace.
- *save_dir* An optional parameter to use only when write_output = TRUE that indicates the save directory for files to be placed in. For example, "C:/"

```
#Set the download location (add your own directory)
working_dir <- "C:/"

#Download NLDAS data at NC_NHC
NLDAS_DL(
  save_dir = working_dir,
  Site_ID = "NC_NHC",
  Lat = 35.9925,
  Lon = -79.0460,
  startDate = "2017-01-01"
)

#Process the downloaded data
NLDAS_processed <- NLDAS_proc(
  read_dir = working_dir,
  Site_IDs = "NC_NHC"
)
```

2.1.2 Downloading and processing NLDAS data for multiple sites

Incoming shortwave radiation data at multiple sites can be downloaded using the function **NLDAS_DL_bulk** which has the following structure:

NLDAS_DL_bulk(*save_dir*, *site_locs*, *startDate*)

- `save_dir` The save directory for the downloaded data to be placed in. For example, "C:/"
- `site_locs` A table with Site_ID, Lat, and Lon, and startDate
- `startDate` An optional parameter. By default, if nothing is provided the function assumes that site_locs has a column that contains startDate. Alternatively, a single startDate can be provided as an argument for the download (YYYY-MM-DD)

Recall from earlier that our table of site information has a column called "startDate", so here the optional parameter is not used. If data fails to download for a site, the **NLDAS_DL_bulk** function will automatically check and retry downloading data for all sites with missing data. However, it is possible that data does not exist for a site. Consequently, it is prudent to confirm the successfully downloaded sites.

```
#Read in a table with initial site information
sites <- data(NC_site_basic)

#Download NLDAS data at NC_NHC
NLDAS_DL_bulk(
  save_dir = working_dir,
  site_locs = sites
)

#List of successfully downloaded sites
NLDAS_list <- stringr::str_sub(list.files(working_dir), 1, -11)

#Processing the downloaded NLDAS data
NLDAS_processed <- StreamLightUtils::NLDAS_proc(read_dir = working_dir,
NLDAS_list)
```

2.2 Downloading and processing MODIS LAI data

There are a variety of sources for LAI data, but for convenience a function is included to process two MODIS LAI products: 1.) MCD15A3H.006 which has 4 day temporal resolution and a pixel size of 500m, and 2.) MCD15A2H.006 which has 8 day temporal resolution and a pixel size of 500m. This example uses MCD15A3H but once the data has been downloaded the process is the same for either product. These products are downloaded through the [AppEEARS website](#).

2.2.1 Export a table of site information

A .csv of Site_ID, Lat and Lon can be used to submit a request that extracts point samples from multiple locations.

```
#Make a table for the MODIS request
request_sites <- sites[, c("Site_ID", "Lat", "Lon")]

#Export your sites as a .csv for the AppEEARS request
write.table(
  request_sites,
  paste0(working_dir, "/NC_sites.csv"),
  sep = ",",
  row.names = FALSE,
  quote = FALSE,
  col.names = FALSE
)
```

2.2.2 Submitting a request

The functions in **StreamLightUtils** are based on accessing LAI data through the [AppEEARS website](#). This requires a NASA EARTHDATA account to access so if you do not already have an account you may [register here](#). Once registered, you can begin by making a request:

1. Click on Extract > Point Sample from the top menu bar
2. On the next page select Start a new request
3. The NC_sites.csv that was exported can now simply be drag-and-dropped to populate a list of sites. You may notice that special characters like "_" are removed from the ID field, don't worry this is ok.

Enter a name to identify your sample

Point Sample name

Upload coordinates from a file

Drop a CSV file containing the coordinates or [click here](#) to select the file. Coordinates can also be entered manually in the uploaded coordinates box.

The CSV file can contain up to 4 columns separated by commas with each coordinate on a separate line.

1. ID (optional) - uniquely identifies the coordinate
2. Category (optional) - label to group common coordinates
3. Latitude - latitude in decimal degrees (-90 to 90)
4. Longitude - longitude in decimal degrees (-180 to 180)

+ Copy

Uploaded coordinates (ID, Category, Lat, Long):

Upload or enter the coordinates to include in the sample (e.g. US-Ha1, DBF, 42.5378, -72.1715)

Enter a name to identify your sample

Point Sample name

Upload coordinates from a file

Drop a CSV file containing the coordinates or [click here](#) to select the file. Coordinates can also be entered manually in the uploaded coordinates box.

The CSV file can contain up to 4 columns separated by commas with each coordinate on a separate line.

1. ID (optional) - uniquely identifies the coordinate
2. Category (optional) - label to group common coordinates
3. Latitude - latitude in decimal degrees (-90 to 90)
4. Longitude - longitude in decimal degrees (-180 to 180)

Uploaded coordinates (ID, Category, Lat, Long): 2

NCNHC: 35.9925, -79.046
NCUENG: 36.1359, -79.1588

4. Next, begin filling in the remaining information to submit a request including a name, Start Date, End Date, and the Selected layers.
5. If possible, **it is advisable to select dates with some time on either side of the desired period to estimate light**. This is to help constrain some of the LAI processing steps such as interpolating to daily values. For example, if I wanted to run the model for 2015 I might download LAI data from 2014-2016.
6. Type MCD15A3H into the layers search box and select the product. Add all of the layers to the selected layers.

Enter a name to identify your sample

NC_sites

Upload coordinates from a file

Drop a CSV file containing the coordinates or click here to select the file. Coordinates can also be entered manually in the uploaded coordinates box.

The CSV file can contain up to 4 columns separated by commas with each coordinate on a separate line.

1. ID (optional) - uniquely identifies the coordinate
2. Category (optional) - label to group common coordinates
3. Latitude - latitude in decimal degrees (-90 to 90)
4. Longitude - longitude in decimal degrees (-180 to 180)

Start Date: 01-01-2018

End Date: 11-14-2018


☐ Is Date Recurring?


Uploaded coordinates (ID, Category, Lat, Long): 2

NCNHC, 35.9925, -79.046

NCUEng, 36.1359, -79.1588

Selected coordinates



Add coordinates using the  tool. View coordinate details by clicking the markers on the map.

Select the layers to include in the sample

Combined MODIS Leaf Area Index (LAI) and Fraction of Photosynthetically Active Radiation (FPAR)
MCD15A3H.006, 500m, 4 day, (2002-07-04 to Present)

There are no layers available for this product.

Selected layers

FparExtra_QC	500m, 4 day	—
FparLai_QC	500m, 4 day	—
FparStdDev_500m	500m, 4 day	—
Fpar_500m	500m, 4 day	—
LaiStdDev_500m	500m, 4 day	—
Lai_500m	500m, 4 day	—

Once everything is filled out submit the request. You will receive an email notification of the request and then a second notification when the download is ready. Downloads are typically ready the same day or within a day or two depending on the size of the request. For reference, the request in this example only took 15 minutes to complete.

2.2.3 Unpacking the downloaded LAI data

Once the download is ready it can be processed using two built-in functions to **StreamLightUtils**. The downloaded .zip file can be unpacked using **AppEEARS_unpack_QC** which has the following structure:

AppEEARS_unpack_QC(*zip_file*, *zip_dir*, *request_sites*)

- *zip_file* The name of the zip file. For example, "myzip.zip"
- *zip_dir* The directory the zip file is located in. For example, "C:/"
- *request_sites* A string of site IDs

This function returns the unpacked data as a list, with each element in the list representing the data for a given site.

```
MOD_unpack <- AppEEARS_unpack_QC(
  zip_file = "nc-sites.zip",
  zip_dir = working_dir,
  request_sites[, "Site_ID"]
)
```

2.2.4 Processing the downloaded LAI data

The **StreamLightUtils** package leverages the [phenofit package](#) to help handling the processing of LAI data. There are a variety of curve fitting methods and this tutorial uses the approach from [Gu et al. \(2009\)](#). The unpacked data can then be processed using **AppEEARS_proc** which has the following structure:

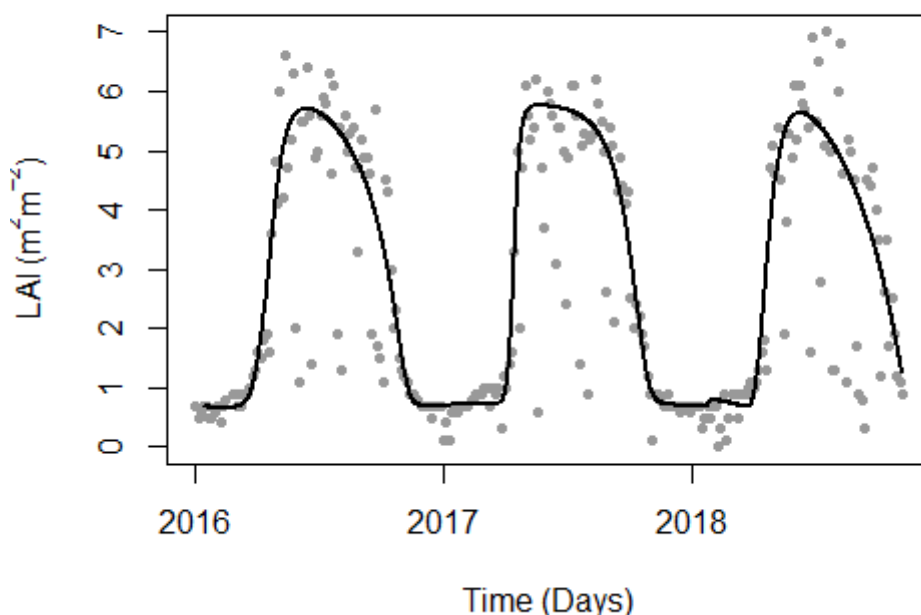
AppEEARS_proc(*Site*, *proc_type*)

- *unpacked_LAI* Output from the AppEEARS_unpack_QC function
- *fit_method* There are several options available from the phenofit package including "AG", "Beck", "Elmore", "Gu", "Klos", "Zhang".
- *write_output* Logical indicating whether to write each individual driver file to disk. Default value is FALSE.
- *save_dir* Optional parameter when write_output = TRUE. The save directory for files to be placed in. For example, "C:/
- *plot* Logical, where plot = TRUE generates a plot and plot = FALSE does not

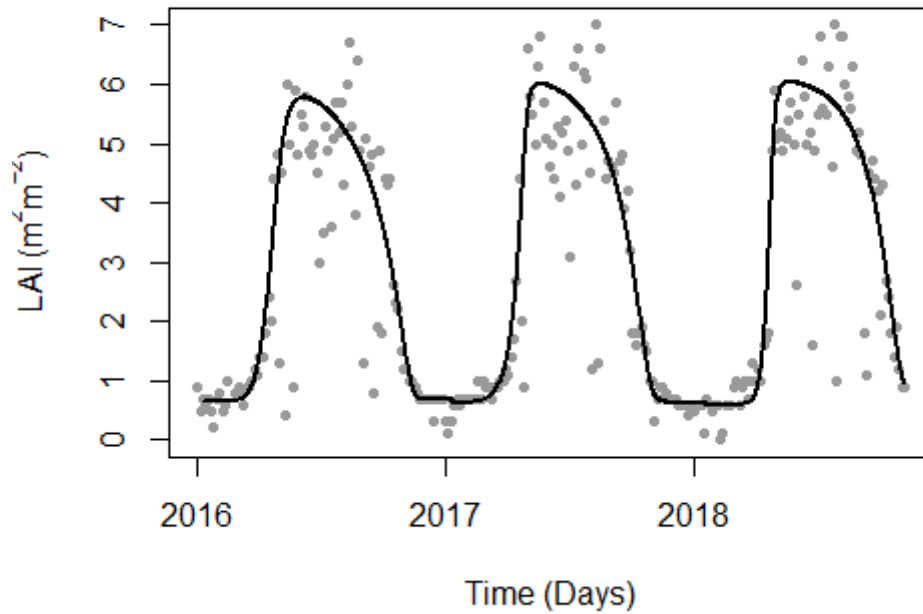
Let's process the LAI data and visualize the results. The black line is the new fitted, interpolated, daily LAI.

```
MOD_processed <- AppEEARS_proc(
  unpacked_LAI = MOD_unpack,
  fit_method = "Gu",
  plot = TRUE
)
```

Site= NC_NHC r2 = 0.7 ;RMSE = 1.23 ;MAE = 0.69



Site= NC_UEno r2 = 0.77 ;RMSE = 1.11 ;MAE = 0.65



2.3 Creating a standardized driver file

Finally, a driver file can be made using the **make_driver** function which has the following structure:

make_driver(*site_locs*, *NLDAS_processed*, *MOD_processed*, *write_output*, *save_dir*)

- *site_locs* A table with Site_ID, Lat, and Lon, and the coordinate reference system designated as an EPSG code.
For example, the most common geographic system is WGS84 and its EPSG code is 4326
- *NLDAS_processed* Output from the **NLDAS_proc** function
- *MOD_processed* Output from the **AppEEARS_proc** function
- *write_output* Logical value to indicate whether to write each individual driver file to disk. Default value is FALSE.
- *save_dir* Optional parameter when *write_output* = TRUE. The save directory for files to be placed in. For example, "C:/

```
#Make the driver files
make_driver(
  site_locs = sites,
  NLDAS_processed = NLDAS_processed,
  MOD_processed = MOD_processed
)
```

Let's take a moment to examine the final structure of the driver file

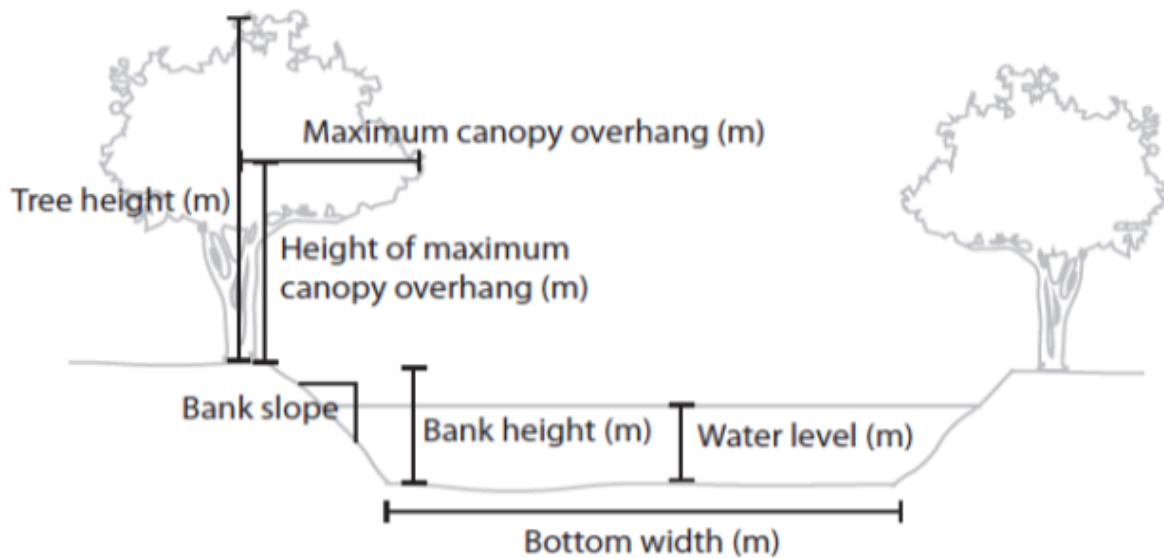
```
#Read in a driver file
data(NC_NHC_driver)
head(NC_NHC_driver)
```

The columns are as follows:

- "local_time" A POSIXct object in local time
- "offset" The offset of local time from UTC, used in the [solar_c function](#)
- "yday" A unique identifier for each day in the format YYYY + day of year
- "DOY" The day of year (1-365 or 366 for leap years)
- "Hour" Hour (0-23)
- "SW_inc" Total incoming shortwave radiation (W m^{-2}) from NLDAS
- "LAI" MODIS leaf area index ($\text{m}^2 \text{m}^{-2}$) interpolated to daily values by [AppEEARS_proc](#)

3. Preparing a parameter file

There are several site parameters required to run **StreamLight**; however, not all of these parameters have built in functions within **StreamLightUtils**. Similarly, not all parameters are easily obtained nor will they all have equal importance for model performance. Here, we detail the same process used to extract parameter values from Savoy et al. (in review). To begin with, let's revisit the parameters used:



- *Lat* The site latitude
- *Lon* The site longitude
- *channel_azimuth* Channel azimuth
- *bottom_width* Bottom width (m)
- *BH* Bank height (m)
- *BS* Bank slope
- *WL* Water level (m)
- *TH* Tree height (m)
- *overhang* Maximum canopy overhang (m)
- *overhang_height* Height of the maximum canopy overhang (m)
- *x_LAD* Leaf angle distribution

To run the model on multiple sites it is easiest to construct a table of parameters for each site such as the following example.

```
#Load the example parameter file
data(NC_params)
head(NC_params, n = 2)
```

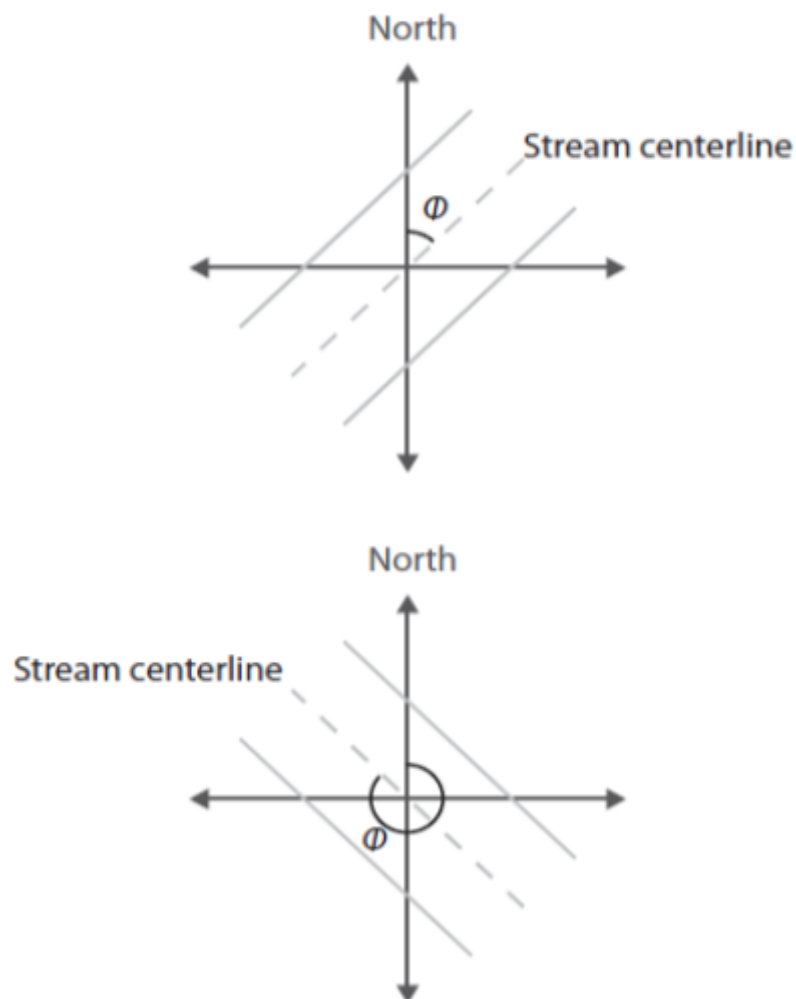
3.1 Parameter descriptions and values

3.1.1 Channel azimuth (*channel_azimuth*)

Currently there is no functionality to derive stream azimuth within **StreamLightUtils**. In the meantime, these can be derived manually using aerial photographs, flowlines, or field derived measurements. Because we have based our model on SHADE2 (Li et al., 2012), we follow their conventions where stream azimuth is measured clockwise from North (see figure below). However, at present both banks are parameterized identically in **StreamLight** (e.g. only a single tree height is put in instead of the heights of trees on either bank) and so in reality a channel azimuth of 45° and 225° will yield the same results. We only mention this point in case future development may allow for parameterizing banks separately, or in case someone wanted to modify the code on their own to add in this functionality.

Example of deriving azimuth, note the first azimuth of the first example is 45° whereas the second example is 315° .

Measuring stream azimuth (ϕ)
(in degrees clockwise from N)



3.1.2 Width (*bottom_width*)

The widths used in this tutorial are from field measurements. However, if field measurements are not available or feasible there are various remotely sensed products such as the [NARWidth dataset](#) from [Allen & Pavelsky](#). There are also empirically-derived estimates, such as those from [McManamay * & DeRolph, 2019](#).

3.1.3 Bank height (*BH*)

Without detailed information of bank heights a default value of 0.1m was used for all sites.

3.1.4 Bank slope (*BS*)

Without detailed information of bank slopes a default value of 100 was used for all sites.

3.1.5 Water level (*WL*)

Without detailed information of water level a default value of 0.1m was used for all sites.

3.1.6 Tree height (*TH*)

StreamLightUtils has a built in function to derive tree height using the LiDAR derived estimates of Simard et al. (2011). The function **extract_height** will retrieve an estimate of tree height (m) based on latitude and longitude and has the following structure:

extract_height(*Site_ID*, *Lat*, *Lon*)

- *Site_ID* The site identifier ("Site_ID")
- *Lat* The site latitude
- *Lon* The site longitude
- *site_crs* The coordinate reference system of the points, preferably designated as an EPSG code. For example, the most common geographic system is WGS84 and its EPSG code is 4326.

Although this parameter file already contains tree height, the following is an example of how to use this function

```
#Extract tree height
extract_height(
  Site_ID = NC_params[, "Site_ID"],
  Lat = NC_params[, "Lat"],
  Lon = NC_params[, "Lon"],
  site_crs = NC_params[, "epsg_crs"]
)
```

3.1.7 Maximum canopy overhang (*overhang*)

Without detailed information on canopy overhang it was assumed that overhang was 10% of tree height at all sites.

3.1.8 Height of maximum canopy overhang (*overhang_height*)

Without detailed information on the height of maximum canopy overhang a value of NA can be used. When *overhang_height* = NA, the model will default to using 75% of tree height.

3.1.9 Leaf angle distribution(*x_LAD*)

Most canopies can be approximated by a spherical distribution of leaf angles ($x = 1$) (Campbell & Norman, 1998), and so *x_LAD* was set to 1 at all sites.

4. Running StreamLight

First time installation of the **StreamLight** package from GitHub can be done with the devtools library and once installed, the package can be loaded as normal.

```
devtools::install_github("psavoy/StreamLight")
library("StreamLight")
```

Estimates of average light across a transect can be estimated using the **stream_light** function which has the following structure:

stream_light(*driver_file*, *Lat*, *Lon*, *channel_azimuth*, *bottom_width*, *BH*, *BS*, *WL*, *TH*, *overhang_height*, *x_LAD*)

- *driver_file* The model driver file
- *Lat* The site latitude
- *Lon* The site longitude
- *channel_azimuth* Channel azimuth
- *bottom_width* Bottom width (m)
- *BH* Bank height (m)
- *BS* Bank slope
- *WL* Water level (m)
- *TH* Tree height (m)
- *overhang* Maximum canopy overhang (m)
- *overhang_height* Height of the maximum canopy overhang (m). If overhang_height = NA, then the model defaults to a value of 75% of tree height.
- *x_LAD* Leaf angle distribution, default = 1

As outlined in the previous section on preparing parameter files. In Savoy et al. (in review) we made some simplifying assumptions to facilitate applying this model easily to locations that lacked detailed **in situ** measurements.

4.1 Running the model

4.1.1 Generate estimates for a single site

To run the model for a single site simply add the parameters to the function.

```
#Load the example driver file for NC_NHC
data(NC_NHC_driver)

#Run the model
NC_NHC_modeled <- stream_light(
  NC_NHC_driver,
  Lat = 35.9925,
  Lon = -79.0460,
  channel_azimuth = 330,
  bottom_width = 18.9,
  BH = 0.1,
  BS = 100,
  WL = 0.1,
  TH = 23,
  overhang = 2.3,
  overhang_height = NA,
  x_LAD = 1
)
```

4.1.2 Generate estimates for multiple sites

It is also possible to then loop over multiple sites by wrapping the model in another function and below is an example of this that could be adapted to your own workflow.

```
#Function for batching over multiple sites
batch_model <- function(Site, read_dir, save_dir){
  #Get the model driver
  driver_file <- readRDS(paste(read_dir, "/", Site, "_driver.rds", sep =
  ""))

  #Get model parameters for the site
  site_p <- params[params[, "Site_ID"] == Site, ]

  #Run the model
  modeled <- stream_light(
    driver_file,
    Lat = site_p[, "Lat"],
    Lon = site_p[, "Lon"],
    channel_azimuth = site_p[, "Azimuth"],
    bottom_width = site_p[, "width"],
    BH = site_p[, "BH"],
    BS = site_p[, "BS"],
    WL = site_p[, "WL"],
    TH = site_p[, "TH"],
    overhang = site_p[, "overhang"],
    overhang_height = site_p[, "overhang_height"],
    x_LAD = site_p[, "x"]
  )

  #Save the output
  saveRDS(modeled, paste(save_dir, "/", Site, "_predicted.rds", sep = ""))

} #End batch_model

#Applying the model to all sites
model_rd <- working_dir
model_sd <- working_dir

#Running the model
lapply(
  params[, "Site_ID"],
  FUN = batch_model,
  read_dir = model_rd,
  save_dir = model_sd
)

#Take a look at the output
data(NC_NHC_predicted)
NC_NHC_predicted[1:2, ]
```

The columns are as follows:

- "local_time" A POSIXct object in local time
- "offset" The offset of local time from UTC, used in the solar_c function
- "jday" A unique identifier for each day in the format YYYY + day of year

- "Year" The year
- "DOY" The day of year (1-365 or 366 for leap years)
- "Hour" Hour (0-23)
- "SW_inc" Total incoming shortwave radiation (W m^{-2}) from NLDAS
- "LAI" MODIS leaf area index ($\text{m}^2 \text{m}^{-2}$) interpolated to daily values by [AppEEARS_proc](#)
- "PAR_inc" -- Incoming PAR above the canopy ($\mu\text{mol m}^{-2} \text{s}^{-1}$)
- "PAR_bc" -- Estimated PAR ($\mu\text{mol m}^{-2} \text{s}^{-1}$) directly below the canopy
- "veg_shade" The proportion of the channel crosssection that is shaded by riparian vegetation
- "bank_shade" -- The proportion of the channel crosssection that is shaded by stream banks
- "PAR_stream" The estimated PAR for the channel cross section ($\mu\text{mol m}^{-2} \text{s}^{-1}$)