Package: rGEDIsimulator (via r-universe)

August 27, 2024

Type Package

Title NASA's Global Ecosystem Dynamics Investigation (GEDI) Simulator for ALS Data

Version 0.2.1

Description Simulates the fullwaveform GEDI data and calculates metrics based on aerial lidar systems data.

License GPL-3

Encoding UTF-8

LazyData true

Depends methods, rGEDI

Imports fs, hdf5r, utils

Suggests lidR, plot3D

SystemRequirements GNU Scientific Library (>= 2.1), HDF5 (>= 1.8.13), libgeotiff (>= 1.4.0), szip (>= 2.1), zlib (>= 1.2)

NeedsCompilation yes

RoxygenNote 7.2.3

Roxygen list(markdown = TRUE)

BugReports https://github.com/caiohamamura/rGEDIsimulator/issues

URL https://github.com/caiohamamura/rGEDIsimulator

Copyright inst/COPYRIGHTS

Repository https://caiohamamura.r-universe.dev

RemoteUrl https://github.com/caiohamamura/rGEDIsimulator

RemoteRef HEAD

RemoteSha 086e354b00afd098861290878991ad12d00805b7

Contents

`rGEDIsimulator-package` *rGEDIsimulator: A NASA's Global Ecosystem Dynamics Investigation (GEDI) simulator using ALS dataset.*

Description

The rGEDI package provides functions for i) downloading, ii) visualizing, iii) clipping, iv) gridding, iv) simulating and v) exporting GEDI data.

Note

See more details about GEDI data in `https://gedi.umd.edu/data/products/`.

Author(s)

Carlos A. Silva, Caio Hamamura, Ruben Valbuena, Steve Hancock, Adrian Cardil, Eben N. Broadbent, Danilo R. A. de Almeida, Celso H. L. Silva Junior and Carine Klauberg

See Also

For comprehensive examples refer to `https://github.com/carlos-alberto-silva/rGEDI/blob/master/README.md`

`gediWFMetrics` *GEDI full waveform data processing*

Description

GEDI full waveform data processing and metrics extraction

Usage

```
gediWFMetrics(
  input,
  outRoot,
  writeFit = FALSE,
  writeGauss = FALSE,
  bounds = NULL,
  ground = FALSE,
  useInt = FALSE,
  useFrac = FALSE,
  rhRes = 5,
  laiRes = 10,
  laiH = 30,
```

```
noRHgauss = FALSE,
gTol = 0,
fhdHistRes = 0.001,
forcePsigma = FALSE,
bayesGround = FALSE,
dontTrustGround = FALSE,
noRoundCoord = FALSE,
noCanopy = FALSE,
dcBias = 0,
nSig = 0,
hNoise = 0,
linkNoise = NULL,
linkFsig = NULL,
linkPsig = NULL,
trueSig = NULL,
bitRate = NULL,
maxDN = NULL,
renoise = FALSE,
newPsig = -1,
oldPsig = 0.764331,
addDrift = NULL,
missGround = FALSE,
minGap = NULL,
photonCount = FALSE,
pcl = FALSE,
nPhotons = 2.1,
photonWind = 200,
noiseMult = 0.1,
rhoVrhoG = NULL,
nPhotC = 2.1,
nPhotG = -1,
photHDF = FALSE,
meanN = 0,
thresh = 1e-14,
varNoise = FALSE,
varScale = NULL,
statsLen = NULL,
noiseTrack = FALSE,
sWidth = NULL,
psWidth = 0,
msWidth = NULL,
preMatchF = FALSE,
postMatchF = FALSE,
pFile = NULL,
gWidth = 1.2,
minGsig = 0.764331,
minWidth = 0,
medNoise = FALSE,
```

```
  varDrift = NULL,
  driftFac = NULL,
  rhoG = 0.4,
  rhoC = 0.57,
  pSigma = NULL,
  gold = FALSE,
  deconTol = NULL,
  readBinLVIS = FALSE,
  readHDFlvis = FALSE,
  readHDFgedi = TRUE,
  level2 = NULL,
  beamList = NULL,
  skipBeams = NULL,
  readBeams = NULL
)
```

Arguments

`input`	`rGEDI::gedi.level1b` (may be a list of objects). Simulated waveform input object(s).
`outRoot`	name. output filename root
`writeFit`	write fitted waveform
`writeGauss`	write Gaussian parameters
`bounds`	minX minY maxX maxY. only analyse data within bounds
`ground`	read true ground from file
`useInt`	use discrete intensity instead of count
`useFrac`	use fractional hits rather than counts
`rhRes`	r. percentage energy resolution of RH metrics
`laiRes`	res. lai profile resolution in metres
`laiH`	h. height to calculate LAI to
`noRHgauss`	do not fit Gaussians
`gTol`	tol. ALS ground tolerance. Used to calculate slope.
`fhdHistRes`	res. waveform intensity resolution to use when calculating FHD from histograms
`forcePsigma`	do not read pulse sigma from file
`bayesGround`	use Bayseian ground finding
`dontTrustGround`	don't trust ground in waveforms, if included
`noRoundCoord`	do not round up coords when outputting
`noCanopy`	do not calculate FHD histograms and LAI profiles
`dcBias`	n. mean noise level
`nSig`	sig. noise sigma

hNoise	n. hard threshold noise as a fraction of integral
linkNoise	linkM cov. apply Gaussian noise based on link margin at a cover
linkFsig	sig. footprint width to use when calculating and applying signal noise
linkPsig	sig. pulse width to use when calculating and applying signal noise
trueSig	sig. true sigma of background noise
bitRate	n. digitisation bit rate
maxDN	max. maximum DN
renoise	remove noise from truth before applying new noise level
newPsig	sig. new value for pulse width, when lengthening pulse
oldPsig	sig. old value for pulse width if not defined in waveform file, when lengthening pulse
addDrift	xi. apply detector background drift
missGround	assume ground is missed to assess RH metrics
minGap	gap. delete signal beneath min detectable gap fraction
photonCount	output point cloud from photon counting
pcl	convert to photon counting pulsecompressed
nPhotons	n. mean number of photons
photonWind	x. window length for photon counting search, metres
noiseMult	x. noise multiplier for photoncounting
rhoVrhoG	x. ratio of canopy to ground reflectance at this wavelength. Not different from rhoV and rhoG
nPhotC	n. mean number of canopy photons (replaces nPhotons and rhoVrhoG)
nPhotG	n. mean number of ground photons (replaces nPhotons and rhoVrhoG)
photHDF	write photoncounting
meanN	n. mean noise level, if using a predefined mean level
thresh	n. noise threshold, if using a predefined noise threshold
varNoise	use a variable noise threshold
varScale	x. variable noise threshold scale (multiple of stdev above mean to set threshold)
statsLen	len. length to calculate noise stats over for varNoise
noiseTrack	use noise tracking
sWidth	sig. smoothing width, after denoising
psWidth	sigma. smoothing width, before denoising
msWidth	sig. smoothing width, after noise stats, before denoising
preMatchF	matched filter before denoising
postMatchF	matched filter after denoising
pFile	file. read pulse file, for deconvolution and matched filters
gWidth	sig. Gaussian parameter selection smoothing width
minGsig	sig. minimum Gaussian sigma to fit

`minWidth`	n. minimum feature width in bins
`medNoise`	use median stats rather than mean
`varDrift`	correct detector drift with variable factor
`driftFac`	xi. fix drift with constant drift factor
`rhoG`	rho. ground reflectance
`rhoC`	rho. canopy reflectance
`pSigma`	sig. pulse width to smooth by if using Gaussian pulse
`gold`	deconvolve with Gold's method
`deconTol`	deconvolution tolerance
`readBinLVIS`	input is an LVIS binary file. Default FALSE.
`readHDFlvis`	read LVIS HDF5 input. Default FALSE.
`readHDFgedi`	read GEDI simulator HDF5 input. Default TRUE.
`level2`	name. level2 filename for LVIS ZG. Default NULL.
`beamList`	character. 0/1 for whether or not to use beams 18, default "11111111"
`skipBeams`	character. list of beam numbers to skip. No spaces between (eg "123")
`readBeams`	character. list of beam numbers to read. No spaces between (eg "123")

Details

a) Metrics descriptions

a.1) Metrics available to GEDI

- *gHeight* Ground elevation (m) from Gaussian fitting
- *maxGround* Ground elevation (m) from lowest maximum
- *inflGround* Ground elevation (m) from inflection points.
- *signal top* Elevation of first point above noise (may include noise tracking).
- *signal bottom* Elevation of last return above noise (may include noise tracking).
- *cover* Canopy cover (fraction) from area of Gaussian fitted ground. Uses rho_v=0.57 and rho_g=0.4.
- *leading edge ext* Leading edge extent (m), from Lefksy et al (2007).
- *trailing edge extent* Trailing edge extent (m), from Lefksy et al (2007).
- *rhGauss 0-100* RH metrics, 0%-100%, using ground from Gaussian fitting (m).
- *rhMax 0-100* RH metrics, 0%-100%, using ground from lowest maximum (m).
- *rhInfl 0-100* RH metrics, 0%-100%, using ground from inflection points (m).
- *gaussHalfCov* Canopy cover (fraction) from double the energy beneath the Gaussian ground. Uses rho_v=0.57 and rho_g=0.4.
- *maxHalfCov* Canopy cover (fraction) from double the energy beneath the lowest maximum ground. Uses rho_v=0.57 and rho_g=0.4.
- *infHalfCov* Canopy cover (fraction) from double the energy beneath the inflection point ground. Uses rho_v=0.57 and rho_g=0.4.

- *bayHalfCov* Canopy cover (fraction) from double the energy beneath the experimental "Bayesian" ground. Uses rho_v=0.57 and rho_g=0.4.
- *lon* Footprint centre longitude in projection of ALS data (m).
- *lat* Footprint centre latitude in projection of ALS data (m).
- *waveEnergy* Total energy within waveform (will be 1 scaled by noise for simulations).
- *blairSense* Blair's sensitivity metric. Canopy cover at which this SNR would have90% chance of detecting ground (does not account for rho_v/rho_g).
- *FHD* Foliage height diversity
- *niM2* Wenge Ni's biomass metric, equal to the sum of the RH metrics to the power of 2 (unpublished)
- *niM2.1* Wenge Ni's biomass metric, equal to the sum of the RH metrics to the power of 2.1 (unpublished)

a.2) Metrics unavailable to GEDI

- *wave ID* Waveform label, relates to plot name and footprint number.
- *true ground* Ground elevation (m) from ALS. Centre of gravity of ground points within footprint
- *true top* Levation of highest point of waveform (m), without noise. Includes pulse blurring.
- *ground slope* Effective ground slope (degrees), from width of ground return. Includes roughness.
- *ALS cover* Canopy cover (fraction) from ALS data. Uses rho_v=0.57 and rho_g=0.4.
- *rhReal 0-100* RH metrics, 0%-100%, using "true" ground from ALS data (m).
- *groundOverlap* Fraction of ground return overlapping with canopy return. A measure of understorey.
- *groundMin* Depth of minimum between ground and canopy return. A measure of understorey.
- *groundInfl* d2y/dx2 of inflection point between ground and canopy return. A measure of understorey.
- *pointDense* Average ALS point density within GEDI footprint.
- *beamDense* Average ALS beam density within GEDI footprint.

a.3) System settings

- *pSigma* GEDI system pulse width, sigma (m).
- *fSigma* GEDI footprint width, sigma (m).
- *linkM* Link margin if noise is added (db).
- *linkCov* Canopy cover at which the above link margin is true (fraction).
- *filename* Name of input waveform file.

b) Signal processing description

- *Gaussian fitting* Used for "gHeight", "rhGauss" and "gaussHalfCov". The waveform is denoised (mean+5*sigma, noise tracking to avoid truncation), smoothed (pSigma*0.75) and Gaussians fitted with Levenberg-Marquardt optimisation. The center of the lowest Gaussian containing at least 0.5% of the waveform energy is selected as the ground.

- *Maximum* Used for "maxGround", "rhMax" and "maxHalfCov". The waveform is denoised (mean+5*sigma, noise tracking to avoid truncation), smoothed (pSigma*0.75). The lowest maximum is taken as the ground.
- *Inflection points* Used for "inflGround", "rhInfl" and "inflHalfCov". The waveform is denoised (mean+5*sigma, noise tracking to avoid truncation), smoothed (pSigma*0.75). The centre of gravity between the lowest two inflection points is taken as the ground.
- *Half covers* Used for "gaussHalfCov", "maxHalfCov" and "inflHalfCov". Sum energy beneath estimated ground position. Double that is the ground energy. Calculate canopy cover, correcting for rho_v and rho_g.
  $cover = \frac{E_{can}}{E_{can} + E_g * \frac{rho_v}{rho_g}}$
  Where Ecan is the canopy energy, Eg is the ground energy, rho_v is the vegetation reflectance and rho_g is the ground reflectance.
- *Edge extents* These are described in: Lefsky, Michael A., Michael Keller, Yong Pang, Plinio B. De Camargo, and Maria O. Hunter. "Revised method for forest canopy height estimation from Geoscience Laser Altimeter System waveforms." Journal of Applied Remote Sensing 1, no. 1 (2007): 013537-013537.

Value

Returns a list of metrics derived from the simulated full waveform. A text file (txt) containing the metrics will be saved in the output folder (outRoot). Please see the details section for checking the definition of the metrics.

See Also

i) Hancock, S., Armston, J., Hofton, M., Sun, X., Tang, H., Duncanson, L.I., Kellner, J.R. and Dubayah, R., 2019. The GEDI simulator: A large-footprint waveform lidar simulator for calibration and validation of spaceborne missions. Earth and Space Science. doi:10.1029/2018EA000506

ii) gediSimulator: https://bitbucket.org/StevenHancock/gedisimulator/src/master/

Examples

```
libsAvailable <- require(lidR) && require(plot3D)
if (libsAvailable) {
  outdir <- tempdir()

  # Specifying the path to ALS data (zip)
  alsfile_Amazon_zip <- system.file("extdata", "Amazon.zip", package = "rGEDIsimulator")
 alsfile_Savanna_zip <- system.file("extdata", "Savanna.zip", package = "rGEDIsimulator")

  # Unzipping ALS data
  alsfile_Amazon_filepath <- unzip(alsfile_Amazon_zip, exdir = outdir)
  alsfile_Savanna_filepath <- unzip(alsfile_Savanna_zip, exdir = outdir)

  # Reading and plot ALS file (las file)
  als_Amazon <- readLAS(alsfile_Amazon_filepath)
  als_Savanna <- readLAS(alsfile_Savanna_filepath)
```

```
# Extracting plot center geolocations
xcenter_Amazon <- mean(bbox(als_Amazon)[1, ])
ycenter_Amazon <- mean(bbox(als_Amazon)[2, ])
xcenter_Savanna <- mean(bbox(als_Savanna)[1, ])
ycenter_Savanna <- mean(bbox(als_Savanna)[2, ])

# Simulating GEDI full waveform
wf_Amazon <- gediWFSimulator(
  input = alsfile_Amazon_filepath,
  output = file.path(outdir, "gediWF_amazon_simulation.h5"),
  coords = c(xcenter_Amazon, ycenter_Amazon)
)

wf_Savanna <- gediWFSimulator(
  input = alsfile_Savanna_filepath,
  output = file.path(outdir, "gediWF_Savanna_simulation.h5"),
  coords = c(xcenter_Savanna, ycenter_Savanna)
)

# Extracting GEDI full waveform derived metrics without adding noise to the full waveform
wf_amazon_metrics <- gediWFMetrics(input = wf_Amazon, outRoot = file.path(outdir, "amazon"))
wf_savanna_metrics <- gediWFMetrics(input = wf_Savanna, outRoot = file.path(outdir, "savanna"))

metrics <- rbind(wf_amazon_metrics, wf_savanna_metrics)
rownames(metrics) <- c("Amazon", "Savanna")
head(metrics)

# Extracting GEDI full waveform derived metrics after adding noise to the waveform
wf_amazon_metrics_noise <- gediWFMetrics(
  input = wf_Amazon,
  outRoot = file.path(outdir, "amazon"),
  linkNoise = c(3.0103, 0.95),
  maxDN = 4096,
  sWidth = 0.5,
  varScale = 3
)

wf_savanna_metrics_noise <- gediWFMetrics(
  input = wf_Savanna,
  outRoot = file.path(outdir, "savanna"),
  linkNoise = c(3.0103, 0.95),
  maxDN = 4096,
  sWidth = 0.5,
  varScale = 3
)

close(wf_Amazon)
close(wf_Savanna)

metrics_noise <- rbind(wf_amazon_metrics_noise, wf_savanna_metrics_noise)
rownames(metrics_noise) <- c("Amazon", "Savanna")
head(metrics_noise)
}
```

gediWFSimulator *GEDI full waveform data simulation*

Description

Simulate GEDI full waveform data from Airborne Laser Scanning (ALS) 3D point cloud

Input and output filenames, and formats

Usage

```
gediWFSimulator(
  input,
  output,
  ground = TRUE,
  ascii = FALSE,
  waveID = NULL,
  coords = NULL,
  listCoord = NULL,
  gridBound = NULL,
  gridStep = 30,
  pSigma = -1,
  pFWHM = 15,
  readPulse = NULL,
  fSigma = 5.5,
  wavefront = NULL,
  res = 0.15,
  topHat = FALSE,
  sideLobe = FALSE,
  lobeAng = 0,
  checkCover = FALSE,
  maxScanAng = 1e+06,
  decimate = 1,
  pBuff = as.integer(2e+08),
  maxBins = as.integer(1024),
  countOnly = FALSE,
  pulseAfter = FALSE,
  pulseBefore = TRUE,
  noNorm = FALSE,
  noOctree = FALSE,
  octLevels = as.integer(0),
  nOctPix = as.integer(40),
  keepOld = FALSE,
  useShadow = FALSE,
  polyGround = FALSE
)
```

Arguments

`input`	character vector. lasfile input filename
`output`	character. output filename
`ground`	record separate ground and canopy waveforms, default TRUE (shouldn't change).
`ascii`	write output as ASCII. Good for quick tests, default FALSE
`waveID`	id. supply a waveID to pass to the output (only for single footprints) Single footprint, list of footprints, or grid of footprints
`coords`	lon lat numeric vector. footprint coordinate in same system as lasfile
`listCoord`	name. Text file with list of coordinates. Pattern: X Y `[waveID]` `[geoCoordsX]` `[geoCoordsY]`. `[]` are optional, separated by spaces.
`gridBound`	minX maxX minY maxY numeric vector. make a grid of waveforms in this box
`gridStep`	res. grid step size Lidar characteristics. Defaults are expected GEDI values.
`pSigma`	pSigmasig. set Gaussian pulse width as 1 sigma
`pFWHM`	fhwm. set Gaussian pulse width as FWHM in ns
`readPulse`	file. read pulse shape and width from a file instead of making Gaussian
`fSigma`	sig. set footprint width
`wavefront`	file. read wavefront shape from file instead of setting Gaussian. Note that footprint width is still set by fSigma
`res`	res. range resolution of waveform digitisation to output, in units of ALS data
`topHat`	use a top hat wavefront
`sideLobe`	use side lobes
`lobeAng`	ang. lobe axis azimuth Input data quality filters
`checkCover`	check that the footprint is covered by ALS data. Do not output if not
`maxScanAng`	ang. maximum scan angle, degrees
`decimate`	x. probability of accepting an ALS beam Computational speed options
`pBuff`	s. point reading buffer size in Gbytes
`maxBins`	for HDF5, limiting number of bins to save trimming.
`countOnly`	only use count method
`pulseAfter`	apply the pulse smoothing after binning for computational speed, at the risk of aliasing (default)
`pulseBefore`	apply the pulse smoothing before binning to avoid the risk of aliasing, at the expense of computational speed
`noNorm`	don't normalise for ALS density Octree
`noOctree`	do not use an octree
`octLevels`	n. number of octree levels to use
`nOctPix`	n. number of octree pixels along a side for the top level Using full waveform input data (not tested)
`keepOld`	do not overwrite old files, if they exist
`useShadow`	account for shadowing in discrete return data through voxelization
`polyGround`	find mean ground elevation and slope through fitting a polynomial #'

Value

Returns an S4 object of class hdf5r::H5File containing the simulated GEDI full-waveform.

See Also

i) Hancock, S., Armston, J., Hofton, M., Sun, X., Tang, H., Duncanson, L.I., Kellner, J.R. and Dubayah, R., 2019. The GEDI simulator: A large-footprint waveform lidar simulator for calibration and validation of spaceborne missions. Earth and Space Science. doi:10.1029/2018EA000506

ii) gediSimulator: https://bitbucket.org/StevenHancock/gedisimulator/src/master/

Examples

```
libsAvailable <- require(lidR) && require(plot3D)
if (libsAvailable) {
    outdir <- tempdir()

    # specify the path to ALS data (zip)
   alsfile_Amazon_zip <- system.file("extdata", "Amazon.zip", package = "rGEDIsimulator")
  alsfile_Savanna_zip <- system.file("extdata", "Savanna.zip", package = "rGEDIsimulator")

    # Unzipping ALS data
    alsfile_Amazon_filepath <- unzip(alsfile_Amazon_zip, exdir = outdir)
    alsfile_Savanna_filepath <- unzip(alsfile_Savanna_zip, exdir = outdir)

    # Reading and plot ALS file (las file)
    als_Amazon <- readLAS(alsfile_Amazon_filepath)
    als_Savanna <- readLAS(alsfile_Savanna_filepath)

    # Extracting plot center geolocations
    xcenter_Amazon <- mean(bbox(als_Amazon)[1, ])
    ycenter_Amazon <- mean(bbox(als_Amazon)[2, ])
    xcenter_Savanna <- mean(bbox(als_Savanna)[1, ])
    ycenter_Savanna <- mean(bbox(als_Savanna)[2, ])

    # Simulating GEDI full waveform
    wf_Amazon <- gediWFSimulator(
        input = alsfile_Amazon_filepath,
        output = file.path(outdir, "gediWF_amazon_simulation.h5"),
        coords = c(xcenter_Amazon, ycenter_Amazon)
    )

    wf_Savanna <- gediWFSimulator(
        input = alsfile_Savanna_filepath,
        output = file.path(outdir, "gediWF_Savanna_simulation.h5"),
        coords = c(xcenter_Savanna, ycenter_Savanna)
    )

    # Plot ALS and GEDI simulated full waveform

    oldpar <- par()
    par(mfrow = c(2, 2), mar = c(4, 4, 0, 0), oma = c(0, 0, 1, 1), cex.axis = 1.2)
```

```
    scatter3D(als_Amazon@data$X, als_Amazon@data$Y, als_Amazon@data$Z,
        pch = 16, colkey = FALSE, main = "", cex = 0.5, bty = "u",
        col.panel = "gray90", phi = 30, alpha = 1, theta = 45, col.grid = "gray50",
        xlab = "UTM Easting (m)", ylab = "UTM Northing (m)", zlab = "Elevation (m)"
    )

    # Simulated waveforms shot_number is incremental beggining from 0
    shot_number <- 0
    simulated_waveform_amazon <- getLevel1BWF(wf_Amazon, shot_number)
    plot(simulated_waveform_amazon,
        relative = TRUE, polygon = TRUE, type = "l", lwd = 2, col = "forestgreen",
        xlab = "", ylab = "Elevation (m)", ylim = c(90, 140)
    )
    grid()
    scatter3D(als_Savanna@data$X, als_Savanna@data$Y, als_Savanna@data$Z,
        pch = 16, colkey = FALSE, main = "", cex = 0.5, bty = "u",
        col.panel = "gray90", phi = 30, alpha = 1, theta = 45, col.grid = "gray50",
        xlab = "UTM Easting (m)", ylab = "UTM Northing (m)", zlab = "Elevation (m)"
    )

    shot_number <- 0
    simulated_waveform_savanna <- getLevel1BWF(wf_Savanna, shot_number)
    plot(simulated_waveform_savanna,
        relative = TRUE, polygon = TRUE, type = "l", lwd = 2, col = "green",
        xlab = "Waveform Amplitude (%)", ylab = "Elevation (m)", ylim = c(815, 835)
    )
    grid()

    par(oldpar)

    close(wf_Amazon)
    close(wf_Savanna)
}
```

Index