

# Package ‘SensusR’

January 20, 2025

**Type** Package

**Title** Sensus Analytics

**Version** 2.3.1

**Date** 2019-02-01

**Author** Matthew S. Gerber

**Maintainer** Matthew S. Gerber <gerber.matthew@gmail.com>

**Description** Provides access and analytic functions for Sensus data.

**License** GPL-3

**Copyright** The Rector and Visitors of the University of Virginia

**URL** <https://predictive-technology-laboratory.github.io/sensus/>

**Imports** jsonlite (>= 0.9.16), lubridate (>= 1.3.3), plyr (>= 1.8.3),  
ggmap (>= 2.6.1), ggplot2 (>= 2.2.1), R.utils (>= 2.3.0),  
openssl (>= 0.9.6)

**RoxygenNote** 6.1.1

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2019-02-01 18:03:27 UTC

## Contents

plot.AccelerometerDatum . . . . .	2
plot.AltitudeDatum . . . . .	3
plot.BatteryDatum . . . . .	3
plot.CellTowerDatum . . . . .	4
plot.CompassDatum . . . . .	4
plot.LightDatum . . . . .	5
plot.LocationDatum . . . . .	5
plot.ScreenDatum . . . . .	6
plot.SoundDatum . . . . .	6
plot.SpeedDatum . . . . .	7
plot.TelephonyDatum . . . . .	7

plot.WlanDatum . . . . .	8
sensus.decompress.gz.files . . . . .	8
sensus.decrypt.bin.files . . . . .	9
sensus.get.all.timestamp.lags . . . . .	9
sensus.get.timestamp.lags . . . . .	10
sensus.get.unique.device.ids . . . . .	10
sensus.list.activities . . . . .	11
sensus.list.aws.s3.buckets . . . . .	11
sensus.plot.lag.cdf . . . . .	12
sensus.read.json.files . . . . .	12
sensus.remove.device.id . . . . .	13
sensus.sync.from.aws.s3 . . . . .	14
sensus.write.csv.files . . . . .	14
sensus.write.rdata.files . . . . .	15
SensusR . . . . .	15
trim . . . . .	16
trim.leading . . . . .	16
trim.trailing . . . . .	17
<b>Index</b>	<b>18</b>

---

plot.AccelerometerDatum

*Plot accelerometer data.*

---

## Description

Plot accelerometer data.

## Usage

```
## S3 method for class 'AccelerometerDatum'
plot(x, pch = ".", type = "l", ...)
```

## Arguments

x	Accelerometer data.
pch	Plotting character.
type	Line type.
...	Other plotting parameters.

## Value

None

---

plot.AltitudeDatum     *Plot altitude data.*

---

**Description**

Plot altitude data.

**Usage**

```
## S3 method for class 'AltitudeDatum'  
plot(x, pch = ".", type = "l", ...)
```

**Arguments**

x	Altitude data.
pch	Plotting character.
type	Line type.
...	Other plotting parameters.

**Value**

None

---

plot.BatteryDatum     *Plot battery data.*

---

**Description**

Plot battery data.

**Usage**

```
## S3 method for class 'BatteryDatum'  
plot(x, pch = ".", type = "l",  
      main = "Battery", ...)
```

**Arguments**

x	Battery data.
pch	Plotting character.
type	Line type.
main	Main title.
...	Other plotting parameters.

**Value**

None

plot.CellTowerDatum *Plot cell tower data.*

---

**Description**

Plot cell tower data.

**Usage**

```
## S3 method for class 'CellTowerDatum'  
plot(x, ...)
```

**Arguments**

x	Cell tower data.
...	Other plotting arguments.

**Value**

None

---

plot.CompassDatum *Plot compass data.*

---

**Description**

Plot compass data.

**Usage**

```
## S3 method for class 'CompassDatum'  
plot(x, pch = ".", type = "l", ...)
```

**Arguments**

x	Compass data.
pch	Plotting character.
type	Line type.
...	Other plotting parameters.

**Value**

None

---

plot.LightDatum      *Plot light data.*

---

**Description**

Plot light data.

**Usage**

```
## S3 method for class 'LightDatum'  
plot(x, pch = ".", type = "l", ...)
```

**Arguments**

x	Light data.
pch	Plotting character.
type	Line type.
...	Other plotting parameters.

**Value**

None

---

plot.LocationDatum      *Plot location data.*

---

**Description**

Plot location data.

**Usage**

```
## S3 method for class 'LocationDatum'  
plot(x, ...)
```

**Arguments**

x	Location data.
...	Arguments to pass to plotting routines. This can include two special arguments: <code>qmap.args</code> (passed to <code>qmap</code> ) and <code>geom.point.args</code> (passed to <code>geom_point</code> ).

**Value**

None

plot.ScreenDatum      *Plot screen data.*

---

**Description**

Plot screen data.

**Usage**

```
## S3 method for class 'ScreenDatum'  
plot(x, ...)
```

**Arguments**

x                      Screen data.  
...                     Other plotting parameters.

**Value**

None

---

plot.SoundDatum      *Plot sound data.*

---

**Description**

Plot sound data.

**Usage**

```
## S3 method for class 'SoundDatum'  
plot(x, pch = ".", type = "l", ...)
```

**Arguments**

x                      Sound data.  
pch                    Plotting character.  
type                   Line type.  
...                    Other plotting parameters.

**Value**

None

---

plot.SpeedDatum      *Plot speed data.*

---

**Description**

Plot speed data.

**Usage**

```
## S3 method for class 'SpeedDatum'  
plot(x, pch = ".", type = "l", ...)
```

**Arguments**

x	Speed data.
pch	Plotting character.
type	Line type.
...	Other plotting parameters.

**Value**

None

---

plot.TelephonyDatum      *Plot telephony data.*

---

**Description**

Plot telephony data.

**Usage**

```
## S3 method for class 'TelephonyDatum'  
plot(x, ...)
```

**Arguments**

x	Telephony data.
...	Other plotting parameters.

**Value**

None

---

plot.WlanDatum	<i>Plot WLAN data.</i>
----------------	------------------------

---

**Description**

Plot WLAN data.

**Usage**

```
## S3 method for class 'WlanDatum'
plot(x, ...)
```

**Arguments**

x	WLAN data.
...	Other plotting parameters.

**Value**

None

---

sensus.decompress.gz.files	<i>Decompresses JSON files downloaded from AWS S3.</i>
----------------------------	--

---

**Description**

Decompresses JSON files downloaded from AWS S3.

**Usage**

```
sensus.decompress.gz.files(local.path, skip = TRUE, overwrite = FALSE,
  remove = FALSE)
```

**Arguments**

local.path	Path to location on local machine.
skip	If TRUE and the output file already exists, the output file is returned as is.
overwrite	If TRUE and the output file already exists, the file is silently overwritten; otherwise an exception is thrown (unless skip is TRUE).
remove	If TRUE, the input file is removed afterward, otherwise not.

**Value**

None



---

sensus.decrypt.bin.files

*Decrypts Sensus .bin files that were encrypted using asymmetric public/private key encryption.*

---

### Description

Decrypts Sensus .bin files that were encrypted using asymmetric public/private key encryption.

### Usage

```
sensus.decrypt.bin.files(data.path, is.directory = TRUE,  
  recursive = TRUE, rsa.private.key.path,  
  rsa.private.key.password = askpass, replace.files = FALSE)
```

### Arguments

`data.path` Path to Sensus .bin data (either a file or a directory).  
`is.directory` Whether or not the path is a directory.  
`recursive` Whether or not to read files recursively from directory indicated by path.  
`rsa.private.key.path` Path to RSA private key generated using OpenSSL.  
`rsa.private.key.password` Password used to decrypt the RSA private key.  
`replace.files` Whether or not to delete .bin files after they have been decrypted.

### Value

None

---

sensus.get.all.timestamp.lags

*Get timestamp lags for a Sensus data frame.*

---

### Description

Get timestamp lags for a Sensus data frame.

### Usage

```
sensus.get.all.timestamp.lags(data)
```

### Arguments

`data` Data to plot lags for (e.g., the result of `sensus.read.json.files`).

**Value**

List of lags organized by datum type.

---

```
sensus.get.timestamp.lags
```

*Get timestamp lags for a Sensus datum.*

---

**Description**

Get timestamp lags for a Sensus datum.

**Usage**

```
sensus.get.timestamp.lags(datum)
```

**Arguments**

datum                    Data to plot lags for (e.g., the result of `sensus.read.json.files`).

**Value**

List of lags.

---

```
sensus.get.unique.device.ids
```

*Gets unique device IDs within a dataset.*

---

**Description**

Gets unique device IDs within a dataset.

**Usage**

```
sensus.get.unique.device.ids(data)
```

**Arguments**

data                    Data to write, as read using [sensus.read.json.files](#).

**Value**

Unique device IDs within the data.

---

`sensus.list.activities`*Lists activities in a given phase and state.*

---

**Description**

Lists activities in a given phase and state.

**Usage**

```
sensus.list.activities(data, phase = "Starting", state = "Active")
```

**Arguments**

<code>data</code>	Data, as returned by <a href="#">sensus.read.json.files</a> .
<code>phase</code>	Phase of activity (Starting, During, Stopping)
<code>state</code>	State of phase (Active, Inactive, Unknown)

**Value**

None

---

`sensus.list.aws.s3.buckets`*Lists S3 buckets.*

---

**Description**

Lists S3 buckets.

**Usage**

```
sensus.list.aws.s3.buckets(profile = "default",  
aws.path = "/usr/local/bin/aws")
```

**Arguments**

<code>profile</code>	AWS credentials profile to use for authentication.
<code>aws.path</code>	Path to AWS client.

**Value**

None

`sensus.plot.lag.cdf` *Plot the CDF of inter-reading time lags.*

---

**Description**

Plot the CDF of inter-reading time lags.

**Usage**

```
sensus.plot.lag.cdf(datum, xlim = c(0, 1),  
  xlab = "Inter-reading time (seconds)", ylab = "Percentile",  
  main = paste("Inter-reading times (n=", nrow(datum), ")", sep = ""))
```

**Arguments**

<code>datum</code>	Data frame for a single datum.
<code>xlim</code>	Limits for the x-axis.
<code>xlab</code>	Label for x-axis.
<code>ylab</code>	Label for y-axis.
<code>main</code>	Label for plot.

**Value**

None.

---

`sensus.read.json.files`

*Read JSON-formatted Sensus data.*

---

**Description**

Read JSON-formatted Sensus data.

**Usage**

```
sensus.read.json.files(data.path, is.directory = TRUE,  
  recursive = TRUE, local.timezone = Sys.timezone(),  
  data.types = NULL)
```

**Arguments**

<code>data.path</code>	Path to Sensus JSON data (either a file or a directory).
<code>is.directory</code>	Whether or not the path is a directory.
<code>recursive</code>	Whether or not to read files recursively from directory indicated by path.
<code>local.timezone</code>	The local timezone to convert datum timestamps to, or NULL to leave the timestamps unconverted.
<code>data.types</code>	Specific data types to read. A full list of data types can be found here: <a href="https://predictive-technology-laboratory.github.io/sensus/api/Sensus.Datum.html">https://predictive-technology-laboratory.github.io/sensus/api/Sensus.Datum.html</a> . For example <code>c("AccelerometerDatum", "HeightDatum")</code> will only read accelerometer and height data. Pass NULL to read all data types.

**Value**

All data, listed by type.

**Examples**

```
# data.path = system.file("extdata", "example-data", package="SensusR")
# data = sensus.read.json.files(data.path)
```

---

```
sensus.remove.device.id
```

*Removes all data associated with a device ID from a data collection.*

---

**Description**

Removes all data associated with a device ID from a data collection.

**Usage**

```
sensus.remove.device.id(datum, device.id)
```

**Arguments**

<code>datum</code>	Data collection to process.
<code>device.id</code>	Device ID to remove.

**Value**

Data without a particular device ID.

---

```
sensus.sync.from.aws.s3
```

*Synchronizes data from Amazon S3 to a local path.*

---

### Description

Synchronizes data from Amazon S3 to a local path.

### Usage

```
sensus.sync.from.aws.s3(s3.path, profile = "default",
  local.path = tempfile(), aws.path = "/usr/local/bin/aws",
  delete = FALSE, decompress = FALSE)
```

### Arguments

s3.path	Path within S3. This can be a prefix (partial path).
profile	AWS credentials profile to use for authentication.
local.path	Path to location on local machine.
aws.path	Path to AWS client.
delete	Whether or not to delete local files that are not present in the S3 path.
decompress	Whether or not to decompress any gzip files after downloading them.

### Value

Local path to location of downloaded data.

---

```
sensus.write.csv.files
```

*Write data to CSV files.*

---

### Description

Write data to CSV files.

### Usage

```
sensus.write.csv.files(data, directory, file.name.prefix = "")
```

### Arguments

data	Data to write, as read using <a href="#">sensus.read.json.files</a> .
directory	Directory to write CSV files to. Will be created if it does not exist.
file.name.prefix	Prefix to add to the generated file names.

**Value**

None

---

sensus.write.rdata.files

*Write data to rdata files.*

---

**Description**

Write data to rdata files.

**Usage**

```
sensus.write.rdata.files(data, directory, file.name.prefix = "")
```

**Arguments**

`data` Data to write, as read using [sensus.read.json.files](#).  
`directory` Directory to write CSV files to. Will be created if it does not exist.  
`file.name.prefix` Prefix to add to the generated file names.

**Value**

None

---

SensusR

*SensusR: Sensus Analytics*

---

**Description**

Provides access and analytic functions for Sensus data. More information can be found at the following URL:

**Details**

<https://predictive-technology-laboratory.github.io/sensus>

**SensusR functions**

The SensusR functions handle reading, cleaning, plotting, and otherwise analyzing data collected via the Sensus system.

---

trim	<i>Trim leading and trailing white space from a string.</i>
------	---

---

**Description**

Trim leading and trailing white space from a string.

**Usage**

```
trim(x)
```

**Arguments**

x	String to trim.
---	-----------------

**Value**

Result of trimming.

---

---

trim.leading	<i>Trim leading white space from a string.</i>
--------------	--

---

**Description**

Trim leading white space from a string.

**Usage**

```
trim.leading(x)
```

**Arguments**

x	String to trim.
---	-----------------

**Value**

Result of trimming.



---

<code>trim.trailing</code>	<i>Trim trailing white space from a string.</i>
----------------------------	---

---

**Description**

Trim trailing white space from a string.

**Usage**

```
trim.trailing(x)
```

**Arguments**

x	String to trim.
---	-----------------

**Value**

Result of trimming.

# Index

`geom_point`, 5

`plot.AccelerometerDatum`, 2

`plot.AltitudeDatum`, 3

`plot.BatteryDatum`, 3

`plot.CellTowerDatum`, 4

`plot.CompassDatum`, 4

`plot.LightDatum`, 5

`plot.LocationDatum`, 5

`plot.ScreenDatum`, 6

`plot.SoundDatum`, 6

`plot.SpeedDatum`, 7

`plot.TelephonyDatum`, 7

`plot.WlanDatum`, 8

`qmap`, 5

`sensus.decompress.gz.files`, 8

`sensus.decrypt.bin.files`, 9

`sensus.get.all.timestamp.lags`, 9

`sensus.get.timestamp.lags`, 10

`sensus.get.unique.device.ids`, 10

`sensus.list.activities`, 11

`sensus.list.aws.s3.buckets`, 11

`sensus.plot.lag.cdf`, 12

`sensus.read.json.files`, 10, 11, 12, 14, 15

`sensus.remove.device.id`, 13

`sensus.sync.from.aws.s3`, 14

`sensus.write.csv.files`, 14

`sensus.write.rdata.files`, 15

`SensusR`, 15

`SensusR-package (SensusR)`, 15

`trim`, 16

`trim.leading`, 16

`trim.trailing`, 17