# Package ‘savR’

April 15, 2017

**Type** Package

**Title** Parse and analyze Illumina SAV files

**Version** 1.12.0

**Date** 2015-07-28

**Author** R. Brent Calder

**Maintainer** R. Brent Calder <brent.calder@einstein.yu.edu>

**Description** Parse Illumina Sequence Analysis Viewer (SAV) files,
access data, and generate QC plots.

**License** AGPL-3

**URL** https://github.com/bcalder/savR

**BugReports** https://github.com/bcalder/savR/issues

**Depends** ggplot2

**Imports** methods, reshape2, scales, gridExtra, XML

**Suggests** Cairo, testthat

**biocViews** Sequencing

**NeedsCompilation** no

## R topics documented:

## savR-package *Parse and analyze Illumina SAV files*

### Description

Parse Illumina Sequence Analysis Viewer files

### Details

| | |
|---|---|
| Package: | savR |
| Type: | Package |
| Version: | 1.7.5 |
| Date: | 2015-07-28 |
| License: | AGPL-3 |
| LazyLoad: | yes |

Parse Illumina Sequence Analysis Viewer (SAV) files, access data, and generate QC plots.

### Author(s)

R. Brent Calder <brent.calder@einstein.yu.edu>

### References

For information about Illumina SAV, please refer to
http://supportres.illumina.com/documents/documentation/software_documentation/sav/sequencinganalysisviewer_userguide_15020619c.pdf
For other implementations (and inspiration) please see
http://search.cpan.org/dist/Bio-IlluminaSAV/Bio/IlluminaSAV.pm
https://bitbucket.org/invitae/illuminate

## buildReports *Generate Illumina reports folder*

### Description

Generate a folder of images that approximates the format of the folder that was superceded by InterOp. Requires the Cairo package.

### Usage

```
buildReports(project, destination)

## S4 method for signature 'savProject,character'
buildReports(project,
  destination = "./savR-reports")

## S4 method for signature 'savProject,missing'
buildReports(project)
```

### Arguments

| | |
|---|---|
| project | SAV project |
| destination | path to save reports folder |

### Examples

```
## Not run:
example(savR)
buildReports(fc, "reports")

## End(Not run)
```

## clusterQualityGtN *Get the proportion of clusters over a specified quality threshold*

### Description

Return the ratio of clusters with a quality score less than or equal to a specified value (n) for the requested lanes and cycles.

### Usage

```
clusterQualityGtN(project, lane, cycle, n)

## S4 method for signature 'savProject,integer,integer,integer'
clusterQualityGtN(project, lane,
  cycle, n = 30L)
```

### Arguments

| | |
|---|---|
| `project` | SAV project |
| `lane` | lane(s) number |
| `cycle` | cycle(s) number |
| `n` | quality threshold |

### Examples

```
## Not run:
example(savR)
clusterQualityGtN(fc, 1L, 25L, 30L)

## End(Not run)
```

## `clusters` *Get number of clusters per lane*

### Description

Sum the total number of clusters for all tiles in the lane.

### Usage

```
clusters(project, lane)

## S4 method for signature 'savProject,integer'
clusters(project, lane = 1L)
```

### Arguments

| | |
|---|---|
| `project` | SAV project |
| `lane` | lane(s) number |

### Examples

```
## Not run:
example(savR)
clusters(fc, 1L)

## End(Not run)
```

## `correctedIntensities` *Get Corrected Intensity data*

### Description

Returns a data frame of corrected intensity data.

### Usage

```
correctedIntensities(project)

## S4 method for signature 'savProject'
correctedIntensities(project)
```

### Arguments

| | |
|---|---|
| `project` | SAV project |

### Details

`lane`: Lane number

`tile`: Tile ID

`cycle`: Cycle number

`avg_intensity`: Average intensity

`avg_cor_[ACGT]`: Average corrected intensity of channel A, C, G, or T

`avg_cor_called_[ACGT]`: Average corrected intensity for called clusters in channel A, C, G, or T

`num_{none|[ACGT]}`: Number of called bases for no-call, A, C, G, or T

`sig_noise`: Signal to noise ratio

### Value

sorted data.frame of CI data.

### Examples

```
example(savR)
colnames(correctedIntensities(fc))
```

## cycles *Get the total number of cycles*

### Description

Accessor to obtain the total number of cycles sequenced in an Illumina sequencing run.

### Usage

```
cycles(project)

## S4 method for signature 'savProject'
cycles(project)
```

### Arguments

| | |
|---|---|
| `project` | SAV project |

### Value

total number of cycles in run, including all sequencing and index reads.

### Examples

```
example(savR)
cycles(fc)
```

## directions *Get the number of sequence reads*

### Description

Returns the number of sequencing reads (excluding index reads).

### Usage

```
directions(project)

## S4 method for signature 'savProject'
directions(project)
```

### Arguments

| | |
|---|---|
| `project` | SAV project |

### Value

number of reads

### Examples

```
example(savR)
directions(fc)
```

## errorMetrics *Get Error Metrics*

### Description

Error metrics for lane, tile, and cycle.

### Usage

```
errorMetrics(project)

## S4 method for signature 'savProject'
errorMetrics(project)
```

### Arguments

| | |
|---|---|
| `project` | SAV project |

### Details

`lane`: Lane number

`tile`: Tile ID

`cycle`: Cycle number

`errorrate`: Error rate

`nPerfect`: number of perfect reads

`n[1-4]Error`: Number of reads with 1, 2, 3 and 4 errors

### Value

sorted data.frame of Error metrics

### Examples

```
example(savR)
colnames(extractionMetrics(fc))
```

## extractionMetrics *Get Extraction Metrics*

### Description

Extraction (intensity and FWHM) metrics for lane, tile, and cycle.

### Usage

```
extractionMetrics(project)

## S4 method for signature 'savProject'
extractionMetrics(project)
```

### Arguments

| | |
|---|---|
| `project` | SAV project |

### Details

`lane`**:** Lane number

`tile`**:** Tile ID

`cycle`**:** Cycle number

`FWHM_[ACGT]`**:** Full width at half maximum for A, C, G, or T

`int_[ACGT]`**:** Intensity of channel A, C, G, or T

`datestamp`**:** Time/date stamp

### Value

sorted data.frame of Extraction metrics

### Examples

```
example(savR)
colnames(extractionMetrics(fc))
```

## `flowcellLayout` *Get flowcell layout*

### Description

Accessor to obtain information about the characteristics of the flowcell from an Illumina sequencing run.

### Usage

```
flowcellLayout(project)

## S4 method for signature 'savProject'
flowcellLayout(project)
```

### Arguments

| | |
|---|---|
| `project` | SAV project |

### Value

illuminaFlowCellLayout-class object

### Examples

```
example(savR)
flowcellLayout(fc)
```

## `illuminaFlowCellLayout-class` *Layout of an Illumina flowcell*

### Description

Class representation of the features of an Illumina flow cell.

### Slots

`lanecount`: Number of lanes on the flowcell

`surfacecount`: Number of surfaces

`swathcount`: Number of imaging swaths

`tilecount`: Number of tiles per swath

`sectionperlane`: Number of sections per lane (NextSeq)

`lanepersection`: Number of lanes per section (NextSeq)

`tilenamingconvention`: Description of deviation from original formatting layout

## `illuminaRead-class` *Illumina read*

### Description

Class representation of the features of an Illumina sequencing read.

### Slots

`number`: the index of this read in sequencing

`cycles`: number of cycles in this read

`index`: logical representing whether or not this read is an index read

## `location` *Get Flowcell folder location*

### Description

Accessor to obtain the path to data for a particular SAV project.

### Usage

```
location(project)

## S4 method for signature 'savProject'
location(project)
```

### Arguments

| | |
|---|---|
| `project` | SAV project |

### Value

normalized path to Illumina run data.

### Examples

```
example(savR)
location(fc)
```

## `pfBoxplot` *PF Boxplot*

### Description

Generate a boxplot of the numbers of clusters and the number of Illumina pass-filter clusters per tile and lane

### Usage

```
pfBoxplot(project)

## S4 method for signature 'savProject'
pfBoxplot(project)
```

### Arguments

| | |
|---|---|
| `project` | SAV project |

## `pfClusters` *Get number of PF clusters per lane*

### Description

Sum the total pass filter number of clusters for all tiles in the lane.

### Usage

```
pfClusters(project, lane)

## S4 method for signature 'savProject,integer'
pfClusters(project, lane = 1L)
```

### Arguments

| | |
|---|---|
| `project` | SAV project |
| `lane` | lane(s) number |

### Examples

```
## Not run:
example(savR)
pfClusters(fc, 1L)

## End(Not run)
```

## plotFWHM *Generate FWHM plots*

### Description

Plots the average full width of clusters at half maximum (FWHM) of each tile for a given cycle and base.

### Usage

```
plotFWHM(project, cycle, base)

## S4 method for signature 'savProject,integer,character'
plotFWHM(project, cycle = 1L,
  base = c("A", "C", "G", "T"))

## S4 method for signature 'savProject,missing,missing'
plotFWHM(project)

## S4 method for signature 'savProject,integer,missing'
plotFWHM(project, cycle)

## S4 method for signature 'savProject,missing,character'
plotFWHM(project, base)
```

### Arguments

| | |
|---|---|
| `project` | SAV project |
| `cycle` | sequence cycle |
| `base` | nucleotide base (ACGT) |

## plotIntensity *Plot flowcell intensity by base and cycle*

### Description

Draws a representation of a flowcell, showing the average corrected called intensity values.

## Usage

```
plotIntensity(project, cycle, base)

## S4 method for signature 'savProject,integer,character'
plotIntensity(project, cycle = 1L,
  base = c("A", "C", "G", "T"))

## S4 method for signature 'savProject,missing,missing'
plotIntensity(project)

## S4 method for signature 'savProject,integer,missing'
plotIntensity(project, cycle)

## S4 method for signature 'savProject,missing,character'
plotIntensity(project, base)
```

## Arguments

| | |
|---|---|
| `project` | A savProject-class object |
| `cycle` | integer cycle number |
| `base` | character for nucleotide |

---

# `plotQGT30` *Plot Quality > 30 for a flowcell*

---

## Description

Generate a plot for a given cycle of the percentage of clusters in each tile that are >= Q30.

## Usage

```
plotQGT30(project, cycle)

## S4 method for signature 'savProject,integer'
plotQGT30(project, cycle = 1L)

## S4 method for signature 'savProject,missing'
plotQGT30(project)
```

## Arguments

| | |
|---|---|
| `project` | SAV project |
| `cycle` | sequence cycle |

## qualityHeatmap *Generate a heatmap of qualities*

### Description

Plots a heatmap of quality vs cycle for a given lane for 1 or more sequence reads. Read qualities include sequence + index.

### Usage

```
qualityHeatmap(project, lane, read, collapse)

## S4 method for signature 'savProject,integer,integer,logical'
qualityHeatmap(project, lane,
  read, collapse = T)

## S4 method for signature 'savProject,numeric,numeric,missing'
qualityHeatmap(project, lane,
  read)
```

### Arguments

| | |
|---|---|
| `project` | SAV project |
| `lane` | integer lane specification |
| `read` | integer vector of sequence reads to include (not including index reads) |
| `collapse` | whether or not to collapse index reads into the preceeding read (# reads = directions), default TRUE |

## qualityMetrics *Get Quality Metrics data*

### Description

Quality metric by lane, tile and cycle.

### Usage

```
qualityMetrics(project)

## S4 method for signature 'savProject'
qualityMetrics(project)
```

### Arguments

| | |
|---|---|
| `project` | SAV project |

## Details

`lane`: Lane number

`tile`: Tile ID

`cycle`: Cycle number

`Q1-Q50`: Number of clusters with quality of indicated column

## Value

sorted data.frame of quality data

## Examples

```
example(savR)
colnames(qualityMetrics(fc))
```

---

# `reads` *Get reads*

---

## Description

Accessor to obtain information about the reads of a particular Illumina sequencing run.

## Usage

```
reads(project)

## S4 method for signature 'savProject'
reads(project)
```

## Arguments

| | |
|---|---|
| `project` | SAV project |

## Value

List of illuminaRead-class objects

## Examples

```
example(savR)
reads(fc)
```

## `run` *Get the Run ID*

### Description

Accessor to obtain the string identifier of an Illumina sequencing run.

### Usage

```
run(project)

## S4 method for signature 'savProject'
run(project)
```

### Arguments

| | |
|---|---|
| `project` | SAV project |

### Value

parsed Illumina run id

### Examples

```
example(savR)
run(fc)
```

## `savCorrectedIntensityFormat-class` *Corrected Intensity formatter*

### Description

Lane, tile, cycle, average intensity, corrected intensities (ACGT), average corrected called intensities (ACGT), number of no-calls, number of (ACGT) calls, and signal to noise ratio.

### Slots

`name`: vector of column names

`type`: vector of data types of elements

`lengths`: vector of byte lengths for each element

`order`: vector of column names for sorting

`version`: integer version number

## `savData-class` *Structure for holding parsed InterOp headers and data*

### Description

Structure for holding parsed InterOp headers and data

### Slots

`header`: list of parsed header values

`data`: data.frame of parsed values

## `savErrorFormat-class` *Error Metrics formatter*

### Description

Lane, tile, cycle, errorrate, nPerfect, n1Error, n2Error, n3Error, n4Error.

### Slots

`name`: vector of column names

`type`: vector of data types of elements

`lengths`: vector of byte lengths for each element

`order`: vector of column names for sorting

`version`: integer version number

## `savExtractionFormat-class` *Extraction Metrics formatter*

### Description

Lane, tile, cycle, FWHM (ACGT), intensity (ACGT), datestamp, timestamp. Datestamp and timestamp are munged at the moment because R does not have native support for 32-bit unsigned integers and I have not implemented a solution.

### Slots

`name`: vector of column names

`type`: vector of data types of elements

`lengths`: vector of byte lengths for each element

`order`: vector of column names for sorting

`version`: integer version number

## `savFormat-class` *Base class for formatters*

### Description

Defines the necessary slots to create parse different binary files using the same generic parser.

### Slots

`name`: vector of column names

`type`: vector of data types of elements

`lengths`: vector of byte lengths for each element

`order`: vector of column names for sorting

`version`: integer version number

`default`: logical default format ()

## `savProject-class` *SAV project class*

### Description

Represents a flowcell, metadata and parsed SAV information

### Slots

`location`: Full path to flowcell directory

`reads`: List of illuminaRead-class

`layout`: illuminaFlowCellLayout-class

`runid`: Run ID

`number`: Run number

`flowcell`: Flowcell ID

`instrument`: Instrument ID

`date`: Run date

`cycles`: Total number of cycles

`directions`: Total number of sequence runs (ends)

`parsedData`: SAV data

## savQualityFormat-class *Quality Metrics formatter*

### Description

Lane, tile, cycle, Q1-Q50 counts

### Slots

`name`: vector of column names

`type`: vector of data types of elements

`lengths`: vector of byte lengths for each element

`order`: vector of column names for sorting

`version`: integer version number

## savQualityFormatV5-class *Quality Metrics formatter version 5*

### Description

Lane, tile, cycle, Q1-Q50 counts

### Slots

`name`: vector of column names

`type`: vector of data types of elements

`lengths`: vector of byte lengths for each element

`order`: vector of column names for sorting

`version`: integer version number

## savR *Build a SAV project*

### Description

Constructor to build a savProject-class object and populate it. A SAV project consists of binary files generated by an Illumina sequencing run and placed in a folder named "InterOp". This folder contains a number of ".bin" files that contain statistics about the run. Creating this object parses all of the files and makes the data available for analysis.

### Usage

```
savR(object)

## S4 method for signature 'character'
savR(object)

## S4 method for signature 'missing'
savR()
```

### Arguments

| | |
|---|---|
| `object` | String Path to Flowcell data |

### Examples

```
fc <- savR(system.file("extdata", "MiSeq", package="savR"))
fc
```

## savTileFormat-class *Tile Metrics formatter*

### Description

Lane, tile, code, value. Codes are:

### Details

| | |
|---|---|
| 100 | Cluster Density |
| 101 | PF Cluster Density |
| 102 | Number of clusters |
| 103 | Number of PF clusters |
| 400 | Control lane |

### Slots

- `name`: vector of column names
- `type`: vector of data types of elements
- `lengths`: vector of byte lengths for each element
- `order`: vector of column names for sorting
- `version`: integer version number (header consists of version (1b), length (1b))

## tileMetrics *Get Tile Metrics*

### Description

Returns the Tile Metrics SAV data.

### Usage

```
tileMetrics(project)

## S4 method for signature 'savProject'
tileMetrics(project)
```

### Arguments

| | |
|---|---|
| `project` | SAV project |

### Details

Metrics for each tile are encoded in the following format:

| | |
|---|---|
| cluster density: | 100 |
| PF cluster density: | 101 |
| number of clusters: | 102 |
| number of PF clusters: | 103 |
| phasing for read N: | (200 + (N - 1) * 2) |
| prephasing for read N: | (201 + (N - 1) * 2) |
| percent aligned for read N: | (300 + N - 1) |
| control lane: | 400 |

`lane`**:** Lane number

`tile`**:** Tile ID

`code`**:** Code described above

`value`**:** Value for code key

### Value

sorted data.frame of tile metrics

### References

Codes for Tile Metrics were obtained from the Python Illuminate package:
https://bitbucket.org/invitae/illuminate

### Examples

```
example(savR)
colnames(tileMetrics(fc))
```

# Index