

# Package ‘CytoPipelineGUI’

September 11, 2024

**Title** GUI's for visualization of flow cytometry data analysis pipelines

**Version** 1.2.0

**Description** This package is the companion of the `CytoPipeline` package.

It provides GUI's (shiny apps) for the visualization of flow cytometry data analysis pipelines that are run with `CytoPipeline`.

Two shiny applications are provided, i.e.

an interactive flow frame assessment and comparison tool and

an interactive scale transformations visualization and adjustment tool.

**License** GPL-3

**Encoding** UTF-8

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.3.1

**BugReports** <https://github.com/UCLouvain-CBIO/CytoPipelineGUI/issues>

**URL** <https://uclouvain-cbio.github.io/CytoPipelineGUI>

**biocViews** FlowCytometry, Preprocessing, QualityControl, WorkflowStep, ImmunoOncology, Software, Visualization, GUI, ShinyApps

**Collate** plots.R shiny-functions.R shiny-scaleTransform-module.R shiny.R CytoPipelineGUI-package.R

**Depends** R (>= 4.3), CytoPipeline

**Imports** shiny, plotly, ggplot2, flowCore

**Suggests** testthat (>= 3.0.0), vdiff, diffviewer, knitr, rmarkdown, BiocStyle, patchwork

**VignetteBuilder** knitr

**Config/testthat/edition** 3

**git\_url** <https://git.bioconductor.org/packages/CytoPipelineGUI>

**git\_branch** RELEASE\_3\_19

**git\_last\_commit** d9711f4

**git\_last\_commit\_date** 2024-04-30

**Repository** Bioconductor 3.19

**Date/Publication** 2024-09-11

**Author** Philippe Hauchamps [aut, cre] (<<https://orcid.org/0000-0003-2865-1852>>),  
 Laurent Gatto [aut] (<<https://orcid.org/0000-0002-1520-2268>>),  
 Dan Lin [ctb]

**Maintainer** Philippe Hauchamps <[philippe.hauchamps@uclouvain.be](mailto:philippe.hauchamps@uclouvain.be)>

## Contents

CytoPipelineCheckApp . . . . .	2
CytoPipelineGUI . . . . .	3
plotDiffFlowFrame . . . . .	4
plotScaleTransformedChannel . . . . .	7
plotSelectedFlowFrame . . . . .	9
plotSelectedWorkflow . . . . .	12
ScaleTransformApp . . . . .	13
<b>Index</b>	<b>15</b>

---

CytoPipelineCheckApp *interactive visualization of flow cytometry data analysis pipeline objects stored in cache*

---

## Description

interactive visualization of flow cytometry data analysis pipeline objects stored in cache

## Usage

```
CytoPipelineCheckApp(dir = ".", debug = FALSE)
```

## Arguments

dir	the root directory into which the engine will look for existing CytoPipeline experiments
debug	if TRUE, will output messages on the console tracking the shiny events, for debugging purposes

## Value

no return value

## Examples

```
# run CytoPipeline object first

outputDir <- base::tempdir()

rawDataDir <-
  system.file("extdata", package = "CytoPipeline")
experimentName <- "OMIP021_PeacoQC"
sampleFiles <-
  file.path(
    rawDataDir,
    list.files(
      rawDataDir,
      pattern = "Donor"))
jsonDir <- system.file("extdata", package = "CytoPipeline")
jsonPath <- file.path(jsonDir, "pipelineParams.json")

pipL2 <- CytoPipeline(
  jsonPath,
  experimentName = experimentName,
  sampleFiles = sampleFiles)

suppressWarnings(execute(
  pipL2,
  rmCache = TRUE,
  path = outputDir))

# run shiny app

if (interactive())
  CytoPipelineCheckApp(dir = outputDir)
```

---

CytoPipelineGUI

*CytoPipelineGUI package*

---

## Description

CytoPipelineGUI is the companion package of CytoPipeline, and is used for interactive visualization. It implements two shiny applications :

- a shiny app for interactive comparison of flow frames that are the results of CytoProcessingSteps of the same or different CytoPipeline experiments. It is launched using the following statement: `CytoPipelineCheckApp()`
- a shiny app for interactive visualization and manual adjustments of scale transformation objects. It is launched using the following statement: `ScaleTransformApp()`

**Author(s)**

**Maintainer:** Philippe Hauchamps <philippe.hauchamps@uclouvain.be> ([ORCID](#))

Authors:

- Laurent Gatto <laurent.gatto@uclouvain.be> ([ORCID](#))

Other contributors:

- Dan Lin <dan.8.lin@gsk.com> [contributor]

**See Also**

[CytoPipeline](#), [CytoPipelineCheckApp](#), [ScaleTransformApp](#)

---

plotDiffFlowFrame	<i>Plot the difference plot between two flow frames from a CytoPipeline run</i>
-------------------	---

---

**Description**

Based on an experiment name, this function will gather the required flowFrames from the CytoPipeline disk cache and display a difference plot using the user chosen 1D or 2D view.

**Usage**

```
plotDiffFlowFrame(  
  experimentNameFrom,  
  experimentNameTo,  
  whichQueueFrom,  
  whichQueueTo,  
  sampleFileFrom,  
  sampleFileTo,  
  path,  
  flowFrameNameFrom,  
  flowFrameNameTo,  
  xChannelLabelFrom,  
  xChannelLabelTo,  
  yChannelLabelFrom,  
  yChannelLabelTo,  
  interactive = FALSE,  
  useAllCells,  
  nDisplayCells,  
  useFixedLinearRange,  
  linearRange,  
  transfoListName = " "  
)
```

**Arguments**

experimentNameFrom	the experiment name (representing a pipeline run) from which to extract the flow frame ('from' situation)
experimentNameTo	the experiment name (representing a pipeline run) from which to extract the flow frame ('to' situation)
whichQueueFrom	"pre-processing" or "scale transform" ('from' situation)
whichQueueTo	"pre-processing" or "scale transform" ('to' situation)
sampleFileFrom	in case 'whichQueueFrom' is set to 'pre-processing', which sample file to look at for the 'from' situation. This can be a number or a character. <ul style="list-style-type: none"> <li>• if whichQueueFrom == "scale transform", the sampleFileFrom is ignored</li> <li>• if NULL and whichQueueFrom == "pre-processing", the sampleFileFrom is defaulted to the first one belonging to the experiment</li> </ul>
sampleFileTo	same as sampleFileFrom, but for the 'to' situation
path	the root path to look for the CytoPipeline experiment cache
flowFrameNameFrom	for the 'from' situation, the name of the object to fetch (as referenced in the pipeline workflow)
flowFrameNameTo	for the 'to' situation, the name of the object to fetch (as referenced in the pipeline workflow)
xChannelLabelFrom	the label of the channel to be displayed on the x axis: the conventional syntax is : channelName + " - " + channelMarker
xChannelLabelTo	should be equal to xChannelLabelFrom (otherwise no plot is returned but NULL)
yChannelLabelFrom	the label of the channel to be displayed on the y axis: the conventional syntax is : channelName + " - " + channelMarker
yChannelLabelTo	should be equal to yChannelLabelFrom (otherwise no plot is returned but NULL)
interactive	if TRUE, uses ggplot_shiny
useAllCells	if TRUE, no subsampling will be done
nDisplayCells	if useAllCells == FALSE, the number of subsampled cells
useFixedLinearRange	if TRUE, all channels using a linear scale will use a fixed range set by linearRange
linearRange	set for all channels using a linear scale, if useFixedLinearRange == TRUE
transfoListName	if not set to "", the transformation list (as an object name ending with "_obj", as referenced in the pipeline workflow) to be used for display.

**Value**

a ggplot (or plotly if interactive = TRUE) object

**Examples**

```
# run CytoPipeline object first

outputDir <- base::tempdir()

rawDataDir <-
  system.file("extdata", package = "CytoPipeline")
experimentName <- "OMIP021_PeacoQC"
sampleFiles <-
  file.path(
    rawDataDir,
    list.files(rawDataDir, pattern = "Donor"))
jsonDir <- system.file("extdata", package = "CytoPipeline")
jsonPath <- file.path(jsonDir, "pipelineParams.json")

pipL2 <- CytoPipeline(
  jsonPath,
  experimentName = experimentName,
  sampleFiles = sampleFiles)

suppressWarnings(execute(
  pipL2,
  rmCache = TRUE,
  path = outputDir))

plotDiffFlowFrame(
  experimentNameFrom = experimentName,
  whichQueueFrom = "pre-processing",
  sampleFileFrom = 1,
  flowFrameNameFrom = "remove_doublets_obj",
  xChannelLabelFrom = "FSC-A : NA",
  yChannelLabelFrom = "SSC-A : NA",
  path = outputDir,
  experimentNameTo = experimentName,
  whichQueueTo = "pre-processing",
  sampleFileTo = 1,
  flowFrameNameTo = "remove_debris_obj",
  xChannelLabelTo = "FSC-A : NA",
  yChannelLabelTo = "SSC-A : NA",
  useAllCells = TRUE,
  nDisplayCells = 0,
  useFixedLinearRange = TRUE,
  linearRange = c(-100, 262144))

plotDiffFlowFrame(
  experimentNameFrom = experimentName,
```

```

    whichQueueFrom = "pre-processing",
    sampleFileFrom = 1,
    flowFrameNameFrom = "remove_doublets_obj",
    xChannelLabelFrom = "FSC-A : NA",
    yChannelLabelFrom = "SSC-A : NA",
    path = outputDir,
    experimentNameTo = experimentName,
    whichQueueTo = "pre-processing",
    sampleFileTo = 1,
    flowFrameNameTo = "remove_debris_obj",
    xChannelLabelTo = "FSC-A : NA",
    yChannelLabelTo = "SSC-A : NA",
    useAllCells = FALSE,
    nDisplayCells = 100,
    useFixedLinearRange = FALSE,
    linearRange = NULL)

plotDiffFlowFrame(
  experimentNameFrom = experimentName,
  whichQueueFrom = "pre-processing",
  sampleFileFrom = 1,
  flowFrameNameFrom = "remove_debris_obj",
  xChannelLabelFrom = "FSC-A : NA",
  yChannelLabelFrom = "Comp-525/50Violet-A : L/D Aqua - Viability",
  path = outputDir,
  experimentNameTo = experimentName,
  whichQueueTo = "pre-processing",
  sampleFileTo = 1,
  flowFrameNameTo = "remove_dead_cells_obj",
  xChannelLabelTo = "FSC-A : NA",
  yChannelLabelTo = "Comp-525/50Violet-A : L/D Aqua - Viability",
  useAllCells = TRUE,
  nDisplayCells = 0,
  useFixedLinearRange = FALSE,
  linearRange = NULL,
  transfoListName = "scale_transform_estimate_obj")

```

---

plotScaleTransformedChannel

*Plot a flow frame in 1D with explicit user given scale transform*

---

### **Description**

This function plots a 1D view, i.e. the marginal distribution for one specified channel, of the given flow frame, using the specific user-provided scale transformation parameters.

### **Usage**

```
plotScaleTransformedChannel(
```

```

ff,
channel,
applyTransform = c("axis scale only", "data"),
transfoType = c("linear", "logicle"),
linA,
linB,
negDecades,
width,
posDecades
)

```

### Arguments

<code>ff</code>	the flowFrame to be plotted
<code>channel</code>	the name of the channel of which to display the marginal distribution (i.e. the channel name used as column in the ff expression matrix).
<code>applyTransform</code>	if "data", data are explicitly transformed using the user provided scale transformation parameters, before display if "axis scale only" (default), the data are not transformed, i.e. only the x axis scale is defined according to the scale transformation parameters.
<code>transfoType</code>	the transformation type, currently only linear and logicle(bi-exponential) are supported.
<code>linA</code>	the intercept parameter of the linear transformation.
<code>linB</code>	the slope parameter of the linear transformation.
<code>negDecades</code>	the number of additional decades on the negative side for the logicle transformation.
<code>width</code>	the width parameter of the logicle transformation.
<code>posDecades</code>	the number of positive decades of the logicle transformation.

### Value

a ggplot object

### Examples

```

# run CytoPipeline object first

outputDir <- base::tempdir()

rawDataDir <-
  system.file("extdata", package = "CytoPipeline")
experimentName <- "OMIP021_PeacoQC"
sampleFiles <-
  file.path(
    rawDataDir,
    list.files(rawDataDir, pattern = "Donor"))
jsonDir <- system.file("extdata", package = "CytoPipeline")

```



```
jsonPath <- file.path(jsonDir, "pipelineParams.json")

pipL2 <- CytoPipeline(
  jsonPath,
  experimentName = experimentName,
  sampleFiles = sampleFiles)

suppressWarnings(execute(
  pipL2,
  rmCache = TRUE,
  path = outputDir))

ff <- CytoPipeline::getCytoPipelineFlowFrame(
  pipL2,
  path = outputDir,
  whichQueue = "scale transform",
  objectName = "flowframe_aggregate_obj"
)

plotScaleTransformedChannel(
  ff,
  channel = "FSC-A",
  transfoType = "linear",
  linA = 0.0002,
  linB = -0.5)

plotScaleTransformedChannel(
  ff,
  channel = "Comp-670/30Violet-A",
  transfoType = "logicle",
  negDecades = 1,
  width = 0.5,
  posDecades = 4
)

plotScaleTransformedChannel(
  ff,
  channel = "CD3",
  applyTransform = "data",
  transfoType = "logicle",
  negDecades = 1,
  width = 0.5,
  posDecades = 4
)
```

**Description**

Based on an experiment name, this function will gather the required flowFrame from the CytoPipeline disk cache and display it using the user chosen 1D or 2D view.

**Usage**

```
plotSelectedFlowFrame(
  experimentName,
  whichQueue,
  sampleFile,
  flowFrameName,
  path,
  xChannelLabel,
  yChannelLabel,
  useAllCells,
  nDisplayCells,
  useFixedLinearRange,
  linearRange,
  transfoListName = " "
)
```

**Arguments**

experimentName	the experiment name (representing a pipeline run) from which to extract the flow frame
whichQueue	"pre-processing" or "scale transform"
sampleFile	in case 'whichQueue' is set to 'pre-processing', which sample file to look at. This can be a number or a character. <ul style="list-style-type: none"> <li>if whichQueue == "scale transform", the sampleFile is ignored</li> <li>if NULL and whichQueue == "pre-processing", the sampleFile is defaulted to the first one belonging to the experiment</li> </ul>
flowFrameName	the name of the object to fetch (as referenced in the pipeline workflow)
path	the root path to look for the CytoPipeline experiment cache
xChannelLabel	the label of the channel to be displayed on the x axis: the conventional syntax is : channelName + " - " + channelMarker
yChannelLabel	the label of the channel to be displayed on the y axis: the conventional syntax is : channelName + " - " + channelMarker
useAllCells	if TRUE, no subsampling will be done
nDisplayCells	if useAllCells == FALSE, the number of subsampled cells
useFixedLinearRange	if TRUE, all channels using a linear scale will use a fixed range set by linearRange
linearRange	set for all channels using a linear scale, if useFixedLinearRange == TRUE
transfoListName	if not set to " ", the transformation list (as an object name ending with "_obj", as referenced in the pipeline workflow) to be used for display.

**Value**

a ggplot object

**Examples**

```
# run CytoPipeline object first

outputDir <- base::tempdir()

rawDataDir <-
  system.file("extdata", package = "CytoPipeline")
experimentName <- "OMIP021_PeacoQC"
sampleFiles <-
  file.path(
    rawDataDir,
    list.files(rawDataDir, pattern = "Donor"))
jsonDir <- system.file("extdata", package = "CytoPipeline")
jsonPath <- file.path(jsonDir, "pipelineParams.json")

pipL2 <- CytoPipeline(
  jsonPath,
  experimentName = experimentName,
  sampleFiles = sampleFiles)

suppressWarnings(execute(
  pipL2,
  rmCache = TRUE,
  path = outputDir))

plotSelectedFlowFrame(
  experimentName = experimentName,
  whichQueue = "pre-processing",
  sampleFile = 1,
  flowFrameName = "remove_debris_obj",
  path = outputDir,
  xChannelLabel = "FSC-A : NA",
  yChannelLabel = "SSC-A : NA",
  useAllCells = TRUE,
  nDisplayCells = 0,
  useFixedLinearRange = TRUE,
  linearRange = c(-100, 262144))

plotSelectedFlowFrame(
  experimentName = experimentName,
  whichQueue = "pre-processing",
  sampleFile = 1,
  flowFrameName = "remove_debris_obj",
  path = outputDir,
  xChannelLabel = "FSC-A : NA",
  yChannelLabel = "SSC-A : NA",
  useAllCells = FALSE,
```

```

nDisplayCells = 100,
useFixedLinearRange = FALSE,
linearRange = NULL)

plotSelectedFlowFrame(
  experimentName = experimentName,
  whichQueue = "pre-processing",
  sampleFile = 1,
  flowFrameName = "remove_debris_obj",
  path = outputDir,
  xChannellabel = "Comp-670/30Violet-A : BV785 - CD3",
  yChannellabel = "Comp-780/60Red-A : APCCy7 - CD4",
  useAllCells = TRUE,
  nDisplayCells = 0,
  useFixedLinearRange = FALSE,
  linearRange = NULL,
  transfoListName = "scale_transform_estimate_obj")

```

---

plotSelectedWorkflow *Plot a pipeline workflow from a CytoPipeline run*

---

### Description

Plot a pipeline workflow from a CytoPipeline run

### Usage

```
plotSelectedWorkflow(experimentName, whichQueue, sampleFile, path = path)
```

### Arguments

experimentName	the experiment name (representing a pipeline run) from which to extract the workflow
whichQueue	"pre-processing" or "scale transform"
sampleFile	in case 'whichQueue' is set to 'pre-processing', which sample file to look at. This can be a number or a character. <ul style="list-style-type: none"> <li>if whichQueue == "scale transform", the sampleFile is ignored</li> <li>if NULL and whichQueue == "pre-processing", the sampleFile is defaulted to the first one belonging to the experiment</li> </ul>
path	the root path to look for the CytoPipeline experiment cache

### Value

nothing, but displays the plot as a side effect

## Examples

```
# run CytoPipeline object first

outputDir <- base::tempdir()

rawDataDir <-
  system.file("extdata", package = "CytoPipeline")
experimentName <- "OMIP021_PeacoQC"
sampleFiles <-
  file.path(
    rawDataDir,
    list.files(rawDataDir, pattern = "Donor"))
jsonDir <- system.file("extdata", package = "CytoPipeline")
jsonPath <- file.path(jsonDir, "pipelineParams.json")

pipL2 <- CytoPipeline(
  jsonPath,
  experimentName = experimentName,
  sampleFiles = sampleFiles)

suppressWarnings(execute(
  pipL2,
  rmCache = TRUE,
  path = outputDir))

plotSelectedWorkflow(
  experimentName = experimentName,
  whichQueue = "pre-processing",
  sampleFile = sampleFiles[1],
  path = outputDir)

plotSelectedWorkflow(
  experimentName = experimentName,
  whichQueue = "scale transform",
  sampleFile = NULL,
  path = outputDir)
```

---

ScaleTransformApp

*interactive display and modification of scale transform list*

---

## Description

this application allows the user to visualize a scale transformation list, possibly amending it channel after channel, and save the results on disk. The needed input transformation list and flow frame for visualization needs to be read from a CytoPipeline experiments stored in cache.

**Usage**

```
ScaleTransformApp(dir = ".")
```

**Arguments**

`dir` the root directory into which the engine will look for existing CytoPipeline experiments

**Value**

no return value

**Examples**

```
# run CytoPipeline object first

outputDir <- base::tempdir()

rawDataDir <-
  system.file("extdata", package = "CytoPipeline")
experimentName <- "OMIP021_PeacoQC"
sampleFiles <-
  file.path(rawDataDir, list.files(rawDataDir, pattern = "Donor"))
jsonDir <- system.file("extdata", package = "CytoPipeline")
jsonPath <- file.path(jsonDir, "pipelineParams.json")

pipL2 <-
  CytoPipeline(
    jsonPath,
    experimentName = experimentName,
    sampleFiles = sampleFiles)

suppressWarnings(execute(
  pipL2,
  rmCache = TRUE,
  path = outputDir))

# run shiny app

if (interactive())
  ScaleTransformApp(dir = outputDir)
```

# Index

## \* **internal**

CytoPipelineGUI, [3](#)

CytoPipeline, [4](#)

CytoPipelineCheckApp, [2](#), [4](#)

CytoPipelineGUI, [3](#)

CytoPipelineGUI-package  
(CytoPipelineGUI), [3](#)

plotDiffFlowFrame, [4](#)

plotScaleTransformedChannel, [7](#)

plotSelectedFlowFrame, [9](#)

plotSelectedWorkflow, [12](#)

ScaleTransformApp, [4](#), [13](#)