

# Package ‘starvz’

February 24, 2024

**Title** R-Based Visualization Techniques for Task-Based Applications

**Version** 0.8.0

**Description** Performance analysis workflow that combines the power of the R language (and the tidyverse realm) and many auxiliary tools to provide a consistent, flexible, extensible, fast, and versatile framework for the performance analysis of task-based applications that run on top of the StarPU runtime (with its MPI (Message Passing Interface) layer for multi-node support). Its goal is to provide a fruitful prototypical environment to conduct performance analysis hypothesis-checking for task-based applications that run on heterogeneous (multi-GPU, multi-core) multi-node HPC (High-performance computing) platforms.

**URL** <https://github.com/schnorr/starvz>

**BugReports** <https://github.com/schnorr/starvz/issues>

**Depends** R (>= 3.6.0)

**Imports** methods, grDevices, stats, utils, magrittr, dplyr, ggplot2, tibble, rlang, tidyr, patchwork, purrr, readr (>= 1.4.0), stringr, yaml, lpSolve, gtools, data.tree, RColorBrewer, zoo, Rcpp

**License** GPL-3

**Encoding** UTF-8

**SystemRequirements** C++, arrow package with gzip codec, StarPU

**LazyData** true

**LinkingTo** Rcpp (>= 1.0.6), BH

**RoxygenNote** 7.2.3

**Collate** 'RcppExports.R' 'starvz\_data.R' 'phase1.R' 'phase1\_outlier.R' 'phase1\_parse\_csv.R' 'phase2.R' 'phase2\_aggregation.R' 'phase2\_applications.R' 'phase2\_atree.R' 'phase2\_config.R' 'phase2\_events.R' 'phase2\_gaps.R' 'phase2\_imbalance.R' 'phase2\_kchart.R' 'phase2\_lackready.R' 'phase2\_memory.R' 'phase2\_metrics.R' 'phase2\_mpi.R' 'phase2\_node\_summary.R'

'phase2\_pmtool.R' 'phase2\_progress.R' 'phase2\_states\_chart.R'  
 'phase2\_themes.R' 'phase2\_time\_integration.R' 'phase2\_util.R'  
 'phase2\_var\_chart.R' 'phase2\_var\_panels.R' 'read\_functions.R'  
 'write\_functions.R'

**Suggests** arrow (>= 3.0.0), testthat, flexmix, car, viridis, ggrepel,  
 mclust

**Additional\_repositories** <https://p3m.dev/cran/2024-02-02>

**NeedsCompilation** yes

**Author** Lucas Mello Schnorr [aut, ths]

(<https://orcid.org/0000-0003-4828-9942>),

Vinicius Garcia Pinto [aut] (<https://orcid.org/0000-0002-6845-9358>),

Lucas Leandro Nesi [aut, cre] (<https://orcid.org/0000-0001-8874-1839>),

Marcelo Cogo Miletto [aut] (<https://orcid.org/0000-0002-1191-3863>),

Guilherme Alles [ctb],

Arnaud Legrand [ctb],

Luka Stanisic [ctb],

Rémy Drouilhet [ctb]

**Maintainer** Lucas Leandro Nesi <lucas.nesi@inf.ufrgs.br>

**Repository** CRAN

**Date/Publication** 2024-02-23 23:50:02 UTC

## R topics documented:

cholesky_colors . . . . .	4
data_name_coordinates . . . . .	4
data_name_handle . . . . .	5
data_name_tag . . . . .	5
handles_presence_states . . . . .	6
lu_colors . . . . .	6
multiple_snaps . . . . .	7
panel_abe_solution . . . . .	8
panel_activenodes . . . . .	8
panel_atree . . . . .	9
panel_atree_structure . . . . .	10
panel_compare_tree . . . . .	11
panel_dist2d . . . . .	12
panel_gflops . . . . .	13
panel_gflops_computed_difference . . . . .	14
panel_gpubandwidth . . . . .	15
panel_handles . . . . .	16
panel_hete_imbalance . . . . .	17
panel_imbalance . . . . .	18
panel_kiteration . . . . .	19
panel_lackready . . . . .	20
panel_memory_heatmap . . . . .	20

panel_memory_snap . . . . .	21
panel_memory_state . . . . .	22
panel_model_gflops . . . . .	23
panel_mpibandwidth . . . . .	24
panel_mpiconcurrent . . . . .	25
panel_mpiconcurrentout . . . . .	26
panel_mplistate . . . . .	27
panel_nodememuse . . . . .	28
panel_node_events . . . . .	28
panel_node_summary . . . . .	29
panel_pmtree_kiteration . . . . .	30
panel_pmtree_st . . . . .	31
panel_power_imbalance . . . . .	32
panel_progress . . . . .	33
panel_ready . . . . .	34
panel_resource_usage_task . . . . .	35
panel_st . . . . .	36
panel_st_agg_dynamic . . . . .	36
panel_st_agg_node . . . . .	37
panel_st_agg_static . . . . .	38
panel_st_raw . . . . .	39
panel_st_runtime . . . . .	40
panel_submitted . . . . .	41
panel_title . . . . .	42
panel_usedmemory . . . . .	42
panel_utilheatmap . . . . .	43
panel_utiltreedepth . . . . .	44
panel_utiltreenode . . . . .	45
plot.starvz_data . . . . .	46
pre_handle_gantt . . . . .	47
print.starvz_data . . . . .	47
qrmumps_colors . . . . .	48
resource_utilization_tree_node . . . . .	48
starvz_assemble . . . . .	49
starvz_check_data . . . . .	49
starvz_phase1 . . . . .	50
starvz_plot . . . . .	51
starvz_plot_list . . . . .	52
starvz_read . . . . .	52
starvz_read_config . . . . .	53
starvz_sample_lu . . . . .	54
starvz_set_log . . . . .	54
starvz_transform_olddata . . . . .	55
summary.starvz_data . . . . .	56

---

cholesky_colors	<i>Colors for lu</i>
-----------------	----------------------

---

**Description**

This will be deprecated

**Usage**

```
cholesky_colors()
```

---

data_name_coordinates	<i>Handles Name coordinates</i>
-----------------------	---------------------------------

---

**Description**

Give handles name by their coordinates

**Usage**

```
data_name_coordinates(df)
```

**Arguments**

df	data_handle table of Starvz data
----	----------------------------------

**Value**

data\_handle table with new column Value with the name

**Examples**

```
data_name_coordinates(starvz_sample_1u$Data_handle)
```

---

data_name_handle	<i>Handles Name address</i>
------------------	-----------------------------

---

**Description**

Give handles name by their address

**Usage**

```
data_name_handle(df)
```

**Arguments**

df                    data\_handle table of Starvz data

**Value**

data\_handle table with new column Value with the name

**Examples**

```
data_name_handle(starvz_sample_1u$Data_handle)
```

---

data_name_tag	<i>Handles Name Tag</i>
---------------	-------------------------

---

**Description**

Give handles name by their tag

**Usage**

```
data_name_tag(df)
```

**Arguments**

df                    data\_handle table of Starvz data

**Value**

data\_handle table with new column Value with the name

**Examples**

```
data_name_tag(starvz_sample_1u$Data_handle)
```

---

handles\_presence\_states

*Computes presence of handles over resources*

---

**Description**

Use for precomputation of other memory-related functions

**Usage**

handles\_presence\_states(data)

**Arguments**

data                    starvz\_data with trace data

**Value**

Time-Step aggregated handle presences

**Examples**

handles\_presence\_states(starvz\_sample\_lu)

---

lu\_colors

*Colors for lu*

---

**Description**

This will be deprecated

**Usage**

lu\_colors()

---

multiple_snaps	<i>Create multiple snapshot of memory</i>
----------------	---

---

**Description**

Create multiple visualizations of memory Useful for continuing views

**Usage**

```
multiple_snaps(  
  data = NULL,  
  start = 0,  
  end = 1000,  
  step = 100,  
  path = ".",  
  scale = 8,  
  width = 4,  
  height = 3  
)
```

**Arguments**

data	starvz_data with trace data
start	start time
end	end time
step	between snaps
path	path to save files
scale	for ggsave
width	for ggsave
height	for ggsave

**Value**

A ggplot object

**Examples**

```
## Not run:  
multiple_snaps(data = starvz_sample_lu, 100, 200, 10, ".")  
  
## End(Not run)
```

---

panel\_abe\_solution      *Create a plot with the solution computed by ABE*

---

### Description

Plot per-node and per-tasktype repartition among resource types

### Usage

```
panel_abe_solution(data, base_size = data$config$base_size)
```

### Arguments

data	starvz_data with trace data
base_size	base_size base font size

### Value

A ggplot object

### Examples

```
panel_abe_solution(data = starvz_sample_1u)
```

---

panel\_activenodes      *Create the active nodes in memory plot*

---

### Description

Use starvz\_data to create a line plot of the number of active nodes per type along the application execution time

### Usage

```
panel_activenodes(
  data = NULL,
  step = data$config$activenodes$aggregation$step,
  aggregation = data$config$activenodes$aggregation$active,
  x_start = data$config$limits$start,
  x_end = data$config$limits$end,
  legend = data$config$activenodes$legend
)
```

**Arguments**

data	starvz_data with trace data
step	size in milliseconds for the time aggregation step
aggregation	enable/disable time aggregation for the plot
x_start	X-axis start value
x_end	X-axis end value
legend	enable/disable plot legends

**Value**

A ggplot object

**Examples**

```
## Not run:
panel_activenodes(data = starvz_sample_lu, step = 100)

## End(Not run)
```

---

panel\_atree

---

*Create the elimination tree plot with some options in the config file*


---

**Description**

Use starvz\_data to create a representation of the elimination tree structure considering initialization, communication, and computational tasks. These representations can be controlled in the configuration file.

**Usage**

```
panel_atree(
  data = NULL,
  step = data$config$atree$step,
  legend = data$config$atree$legend,
  zoom = FALSE,
  computation = data$config$atree$computation$active,
  pruned = data$config$atree$computation$pruned$active,
  initialization = data$config$atree$initialization$active,
  x_start = data$config$limits$start,
  x_end = data$config$limits$end,
  communication = data$config$atree$communication$active,
  anomalies = data$config$atree$anomalies$active,
  performance_metric = "time",
  level = 0,
  end_arrow = "ParentEnd"
)
```

**Arguments**

data	starvz_data with trace data
step	size in milliseconds for the time aggregation step
legend	enable/disable panel legend
zoom	enable/disable vertical zoom in the tree structure
computation	enable/disable computations representations in the tree
pruned	enable/disable pruned computations representations in the tree
initialization	enable/disable initialization tasks representation
x_start	X-axis start value
x_end	X-axis end value
communication	enable/disable communication tasks representation
anomalies	enable/disable anomalies tasks representation
performance_metric	which metric to represent ["time", "gflops"]
level	draw a dashed line to divide the tree at the level h
end_arrow	behavior of the end arrow [ParentEnd, ComputationEnd]

**Value**

A ggplot object

**Examples**

```
## Not run:
panel_atree(starvz_sample_lu, step = 10)
panel_atree(starvz_sample_lu,
  step = 20,
  communication = FALSE, initialization = FALSE
)

## End(Not run)
```

---

panel\_atree\_structure *Create the elimination tree structure plot along time*

---

**Description**

Use Atree and Application data to create the elimination tree structure plot in a ggplot object and return it

**Usage**

```
panel_atree_structure(data = NULL, end_arrow = "ParentEnd")
```

**Arguments**

data	starvz_data with trace data
end_arrow	behavior of the end arrow [ParentEnd, ComputationEnd]

**Value**

A ggplot object

**Examples**

```
## Not run:
panel_atree_structure(starvz_sample_1u)

## End(Not run)
```

---

panel_compare_tree	<i>Combine two atree plots to compare two different executions</i>
--------------------	--

---

**Description**

Use starvz\_data Application and Atree to create a plot that shows the total resource utilization, painted by tree node using geom\_ribbon. The colors are reused between nodes, not tied to a specific tree node.

**Usage**

```
panel_compare_tree(
  data1 = NULL,
  data2 = NULL,
  step = data1$config$utiltreenode$step,
  x_start = data1$config$limits$start,
  x_end = data1$config$limits$end,
  performance_metric = "Time",
  add_diff_line = FALSE,
  add_end_line = FALSE
)
```

**Arguments**

data1	starvz_data with trace data
data2	starvz_data with trace data
step	size in milliseconds for the time aggregation step
x_start	X-axis start value
x_end	X-axis end value
performance_metric	which metric to represent ["time", "gflops"]
add_diff_line	add the computed gflops difference line
add_end_line	add smaller end time vertical line

**Value**

A ggplot object

**Examples**

```
## Not run:  
panel_compare_tree(data1, data2, step = 100)  
  
## End(Not run)
```

---

panel_dist2d	<i>Show the 2D MPI distribution</i>
--------------	-------------------------------------

---

**Description**

Visualize the data distribution across nodes of 2D structured data

**Usage**

```
panel_dist2d(  
  data,  
  legend = data$config$dist2d$legend,  
  base_size = data$config$base_size,  
  expand_x = data$config$expand  
)
```

**Arguments**

data	starvz_data with trace data
legend	enable/disable legends
base_size	base_size base font size
expand_x	expand size for scale_x_continuous padding

**Value**

A ggplot object

**Examples**

```
panel_dist2d(data = starvz_sample_lu)
```

---

panel_gflops	<i>Create a line chart panel with GFlops</i>
--------------	--

---

### Description

Use the Variable traces to create a line chart panel with GFlops per resource, aggregated by a configurable time step

### Usage

```
panel_gflops(  
  data,  
  legend = data$config$gflops$legend,  
  base_size = data$config$base_size,  
  expand_x = data$config$expand,  
  x_start = data$config$limits$start,  
  x_end = data$config$limits$end,  
  y_start = 0,  
  y_end = data$config$gflops$limit,  
  step = data$config$gflops$step  
)
```

### Arguments

data	starvz_data with trace data
legend	enable/disable legends
base_size	base_size base font size
expand_x	expand size for scale_x_continuous padding
x_start	X-axis start value
x_end	X-axis end value
y_start	Y-axis start value
y_end	Y-axis end value
step	time step for aggregation

### Value

A ggplot object

### Examples

```
panel_gflops(data = starvz_sample_lu)
```

---

`panel_gflops_computed_difference`*Plot the total computed GFlops difference over time given two traces*

---

### Description

Use `starvz_data` Application and the `GFlop` columns to create a plot that shows the total computed GFlop difference over time using `geom_line`. The blue color represent the faster execution and the red the slower one.

### Usage

```
panel_gflops_computed_difference(  
  data1 = NULL,  
  data2 = NULL,  
  legend = FALSE,  
  x_start = NULL,  
  x_end = NULL,  
  add_end_line = TRUE  
)
```

### Arguments

<code>data1</code>	starvz_data with trace data
<code>data2</code>	starvz_data with trace data
<code>legend</code>	enable/disable plot legends
<code>x_start</code>	X-axis start value
<code>x_end</code>	X-axis end value
<code>add_end_line</code>	add smaller end time vertical line

### Value

A ggplot object

### Examples

```
## Not run:  
panel_gflops_computed_difference(data1, data2)  
  
## End(Not run)
```

---

panel\_gpubandwidth      *Create a line chart panel with GPU bandwidth*

---

### Description

Use the Variable traces to create a line chart panel with GPU bandwidth per resource, aggregated by a configurable time step

### Usage

```
panel_gpubandwidth(
  data,
  legend = data$config$gpubandwidth$legend,
  base_size = data$config$base_size,
  expand_x = data$config$expand,
  x_start = data$config$limits$start,
  x_end = data$config$limits$end,
  y_start = 0,
  y_end = data$config$gpubandwidth$limit,
  step = data$config$gpubandwidth$step,
  total = data$config$gpubandwidth$total
)
```

### Arguments

data	starvz_data with trace data
legend	enable/disable legends
base_size	base_size base font size
expand_x	expand size for scale_x_continuous padding
x_start	X-axis start value
x_end	X-axis end value
y_start	Y-axis start value
y_end	Y-axis end value
step	time step for aggregation
total	show total bandwidth of the execution per resource

### Value

A ggplot object

### Examples

```
panel_gpubandwidth(data = starvz_sample_1u)
```

---

panel\_handles                      *Create a space time visualization of data handles*

---

### Description

Visualize data handles movement To accelerate the process:

```
data$handle_states <- handles_presence_states(data)
data$handle_gantt_data <- pre_handle_gantt(data)
To Select time:
handles_gantt(data, JobId=c(jobid))
snap_data <- pre_snap(data, data$handle_states)
memory_snap(snap_data, 1000, tasks_size=200, step=1)
```

### Usage

```
panel_handles(
  data,
  JobId = NA,
  lines = NA,
  lHandle = NA,
  name_func = NULL,
  legend = data$config$handles$legend,
  base_size = data$config$base_size,
  expand_x = data$config$expand,
  x_start = data$config$limits$start,
  x_end = data$config$limits$end
)
```

### Arguments

data	starvz_data with trace data
JobId	Select handles of jobid
lines	vertical lines
lHandle	select handles
name_func	function to give names to handles
legend	enable/disable legends
base_size	base_size base font size
expand_x	expand size for scale_x_continuous padding
x_start	X-axis start value
x_end	X-axis end value

### Value

A ggplot object

**Examples**

```
panel_handles(data = starvz_sample_lu)
```

---

panel\_hete\_imbalance *Create a line chart with heterogeneous resources and tasks imbalance metrics*

---

**Description**

This function creates a line chart with imbalance metrics. The function applies the metrics on fixed time-steps defined by the user. The metrics consider that the resources are heterogeneous, and each task has a different performance per resource.

**Usage**

```
panel_hete_imbalance(
  data,
  legend = data$config$hete_imbalance$legend,
  base_size = data$config$base_size,
  expand_x = data$config$expand,
  x_start = data$config$limits$start,
  x_end = data$config$limits$end,
  y_start = 0,
  y_end = data$config$hete_imbalance$limit,
  step = data$config$hete_imbalance$step
)
```

**Arguments**

data	starvz_data with trace data
legend	enable/disable legends
base_size	base_size base font size
expand_x	expand size for scale_x_continuous padding
x_start	X-axis start value
x_end	X-axis end value
y_start	Y-axis start value
y_end	Y-axis end value
step	time step for aggregation

**Value**

A ggplot object

**Examples**

```
panel_hete_imbalance(data = starvz_sample_lu)
```

---

panel_imbalance	<i>Create a line chart with homogeneous imbalance metrics.</i>
-----------------	--

---

**Description**

This function creates a line chart with imbalance metrics. The function applies the metrics on fixed time-steps defined by the user. The metrics consider that the resources are homogeneous.

**Usage**

```
panel_imbalance(  
  data,  
  legend = data$config$imbalance$legend,  
  base_size = data$config$base_size,  
  expand_x = data$config$expand,  
  x_start = data$config$limits$start,  
  x_end = data$config$limits$end,  
  y_start = 0,  
  y_end = data$config$imbalance$limit,  
  step = data$config$imbalance$step  
)
```

**Arguments**

data	starvz_data with trace data
legend	enable/disable legends
base_size	base_size base font size
expand_x	expand size for scale_x_continuous padding
x_start	X-axis start value
x_end	X-axis end value
y_start	Y-axis start value
y_end	Y-axis end value
step	time step for aggregation

**Value**

A ggplot object

**Examples**

```
panel_imbalance(data = starvz_sample_lu)
```

---

panel_kiteration	<i>Create a special chart for applications with iterations</i>
------------------	--

---

**Description**

Plot iterations Y over Time X

**Usage**

```
panel_kiteration(  
  data = NULL,  
  middle_lines = data$config$kiteration$middlelines,  
  base_size = data$config$base_size,  
  expand_x = data$config$expand,  
  legend = data$config$kiteration$legend,  
  x_start = data$config$limits$start,  
  x_end = data$config$limits$end,  
  per_node = data$config$kiteration$pernode,  
  sub_ite = data$config$kiteration$subite  
)
```

**Arguments**

data	starvz_data with trace data
middle_lines	plot a middle line
base_size	base_size base font size
expand_x	expand size for scale_x_continuous padding
legend	enable/disable legends
x_start	X-axis start value
x_end	X-axis end value
per_node	Create node facets
sub_ite	Use Subiteration as Y

**Value**

A ggplot object

**Examples**

```
panel_kiteration(data = starvz_sample_lu)
```

---

panel_lackready	<i>Shows if the runtimes is lacking ready tasks</i>
-----------------	---

---

**Description**

Plot a bar over time that shows when the runtime is lacking ready tasks

**Usage**

```
panel_lackready(  
  data = NULL,  
  x_start = data$config$limits$start,  
  x_end = data$config$limits$end  
)
```

**Arguments**

data	starvz_data with trace data
x_start	X-axis start value
x_end	X-axis end value

**Value**

A ggplot object

**Examples**

```
panel_lackready(data = starvz_sample_lu)
```

---

panel_memory_heatmap	<i>Heatmap of memory presence</i>
----------------------	-----------------------------------

---

**Description**

Visualize the presence of memory handles across memory managers

**Usage**

```
panel_memory_heatmap(  
  data,  
  legend = data$config$memory_heatmap$legend,  
  base_size = data$config$base_size,  
  expand_x = data$config$expand  
)
```

**Arguments**

data	starvz_data with trace data
legend	enable/disable legends
base_size	base_size base font size
expand_x	expand size for scale_x_continuous padding

**Value**

A ggplot object

**Examples**

```
panel_memory_heatmap(data = starvz_sample_lu)
```

---

panel_memory_snap	<i>Create a snapshot of memory</i>
-------------------	------------------------------------

---

**Description**

Visualizate memory in a specific time

**Usage**

```
panel_memory_snap(
  data,
  selected_time,
  step,
  legend = data$config$memory_snap$legend,
  base_size = data$config$base_size,
  expand_x = data$config$expand,
  x_start = data$config$limits$start,
  x_end = data$config$limits$end,
  tasks_size = 30
)
```

**Arguments**

data	starvz_data with trace data
selected_time	time
step	for discreate events
legend	enable/disable legends
base_size	base_size base font size
expand_x	expand size for scale_x_continuous padding

x_start	X-axis start value
x_end	X-axis end value
tasks_size	size of tasks in the visualization

**Value**

A ggplot object

**Examples**

```
panel_memory_snap(data = starvz_sample_lu, 100, 10)
```

---

panel_memory_state	<i>Create a memory state space time</i>
--------------------	---

---

**Description**

Show memory events

**Usage**

```
panel_memory_state(
  data = NULL,
  combined = data$config$memory$combined,
  legend = data$config$memory$legend,
  base_size = data$config$base_size,
  expand_x = data$config$expand,
  x_start = data$config$limits$start,
  x_end = data$config$limits$end,
  show_state_total = data$config$memory$state$total,
  show_transfer_total = data$config$memory$transfer$total
)
```

**Arguments**

data	starvz_data with trace data
combined	shows links
legend	enable/disable legends
base_size	base_size base font size
expand_x	expand size for scale_x_continuous padding
x_start	X-axis start value
x_end	X-axis end value
show_state_total	Show the percentage of selected state
show_transfer_total	Show total transfer amount

**Value**

A ggplot object

**Examples**

```
panel_memory_state(data = starvz_sample_1u)
```

---

panel\_model\_gflops      *Create the diagnostig plot for the regression model*

---

**Description**

Use the starvz Application data to observe how the regression model used in the task anomaly classification fits the data.

**Usage**

```
panel_model_gflops(data,  
  freeScales = TRUE, model_type = "LOG_LOG")
```

**Arguments**

data	starvz_data with trace data
freeScales	free X,Y scales for each task and resource type combination
model_type	Choose the regression model type to use

**Value**

A ggplot object

**Examples**

```
## Not run:  
panel_model_gflops(data = starvz_sample_sample)  
  
## End(Not run)
```

---

panel\_mpibandwidth      *Create a line chart panel with MPI bandwidth*

---

### Description

Use the Variable traces to create a line chart panel with MPI bandwidth per node, aggregated by a configurable time step

### Usage

```
panel_mpibandwidth(
  data,
  legend = data$config$mpibandwidth$legend,
  base_size = data$config$base_size,
  expand_x = data$config$expand,
  x_start = data$config$limits$start,
  x_end = data$config$limits$end,
  y_start = 0,
  y_end = data$config$mpibandwidth$limit,
  step = data$config$mpibandwidth$step
)
```

### Arguments

data	starvz_data with trace data
legend	enable/disable legends
base_size	base_size base font size
expand_x	expand size for scale_x_continuous padding
x_start	X-axis start value
x_end	X-axis end value
y_start	Y-axis start value
y_end	Y-axis end value
step	time step for aggregation

### Value

A ggplot object

### Examples

```
panel_mpibandwidth(data = starvz_sample_1u)
```

---

panel\_mpiconcurrent *Create a line chart panel with MPI concurrent*

---

### Description

Use the Variable traces to create a line chart panel with MPI concurrent per node, aggregated by a configurable time step

### Usage

```
panel_mpiconcurrent(  
  data,  
  legend = data$config$mpiconcurrent$legend,  
  base_size = data$config$base_size,  
  expand_x = data$config$expand,  
  x_start = data$config$limits$start,  
  x_end = data$config$limits$end,  
  y_start = 0,  
  y_end = data$config$mpiconcurrent$limit,  
  step = data$config$mpiconcurrent$step  
)
```

### Arguments

data	starvz_data with trace data
legend	enable/disable legends
base_size	base_size base font size
expand_x	expand size for scale_x_continuous padding
x_start	X-axis start value
x_end	X-axis end value
y_start	Y-axis start value
y_end	Y-axis end value
step	time step for aggregation

### Value

A ggplot object

### Examples

```
panel_mpiconcurrent(data = starvz_sample_lu)
```

---

panel\_mpiconcurrentout

*Create a line chart panel with MPI concurrent out*

---

### Description

Use the Variable traces to create a line chart panel with MPI concurrent out per node, aggregated by a configurable time step

### Usage

```
panel_mpiconcurrentout(
  data,
  legend = data$config$mpiconcurrentout$legend,
  base_size = data$config$base_size,
  expand_x = data$config$expand,
  x_start = data$config$limits$start,
  x_end = data$config$limits$end,
  y_start = 0,
  y_end = data$config$mpiconcurrentout$limit,
  step = data$config$mpiconcurrentout$step
)
```

### Arguments

data	starvz_data with trace data
legend	enable/disable legends
base_size	base_size base font size
expand_x	expand size for scale_x_continuous padding
x_start	X-axis start value
x_end	X-axis end value
y_start	Y-axis start value
y_end	Y-axis end value
step	time step for aggregation

### Value

A ggplot object

### Examples

```
panel_mpiconcurrentout(data = starvz_sample_lu)
```

---

panel_mpistate	<i>Create a space-time view of MPI controlers</i>
----------------	---

---

### Description

Create a space-time view of MPI controlers

### Usage

```
panel_mpistate(  
  data = NULL,  
  legend = data$config$mpistate$legend,  
  base_size = data$config$base_size,  
  expand_x = data$config$expand,  
  x_start = data$config$limits$start,  
  x_end = data$config$limits$end,  
  y_start = 0,  
  y_end = data$config$mpistate$limit  
)
```

### Arguments

data	starvz_data with trace data
legend	enable/disable legends
base_size	base_size base font size
expand_x	expand size for scale_x_continuous padding
x_start	X-axis start value
x_end	X-axis end value
y_start	Y-axis start value
y_end	Y-axis end value

### Value

A ggplot object

### Examples

```
panel_mpistate(data = starvz_sample_lu)
```

---

panel\_nodememuse      *Create the node memory usage plot*

---

### Description

Use starvz\_data to create a line plot of the memory usage in MB of active nodes along the application execution time

### Usage

```
panel_nodememuse(
  data = NULL,
  step = data$config$activenodes$aggregation$step,
  aggregation = data$config$activenodes$aggregation$active,
  x_start = data$config$limits$start,
  x_end = data$config$limits$end,
  legend = data$config$activenodes$nodememuse$legend
)
```

### Arguments

data	starvz_data with trace data
step	size in milliseconds for the time aggregation step
aggregation	enable/disable time aggregation for the plot
x_start	X-axis start value
x_end	X-axis end value
legend	enable/disable plot legends

### Examples

```
## Not run:
panel_nodememuse(starvz_sample_lu, step = 100)

## End(Not run)
```

---

panel\_node\_events      *Shows nodes events*

---

### Description

Plot a Gantt chart for all nodes where program events are states An example of event is the fxt\_flush

**Usage**

```
panel_node_events(
  data = NULL,
  legend = data$config$node_events$legend,
  base_size = data$config$base_size,
  expand_x = data$config$expand,
  x_start = data$config$limits$start,
  x_end = data$config$limits$end
)
```

**Arguments**

data	starvz_data with trace data
legend	enable/disable legends
base_size	base_size base font size
expand_x	expand size for scale_x_continuous padding
x_start	X-axis start value
x_end	X-axis end value

**Value**

A ggplot object

**Examples**

```
panel_node_events(data = starvz_sample_lu)
```

---

panel\_node\_summary     *Create a bar plot with node information*

---

**Description**

Bar plot with makespan and abe per node

**Usage**

```
panel_node_summary(
  data,
  legend = data$config$summary_nodes$legend,
  base_size = data$config$base_size,
  expand_x = data$config$expand,
  x_start = data$config$limits$start,
  x_end = data$config$limits$end
)
```

**Arguments**

data	starvz_data with trace data
legend	enable/disable legends
base_size	base_size base font size
expand_x	expand size for scale_x_continuous padding
x_start	X-axis start value
x_end	X-axis end value

**Value**

A ggplot object

**Examples**

```
panel_node_summary(data = starvz_sample_1u)
```

---

```
panel_pmtool_kiteration
```

*Create a special chart for applications with iterations with PMtool data*

---

**Description**

Plot iterations Y over Time X of PMtool data

**Usage**

```
panel_pmtool_kiteration(  
  data = NULL,  
  legend = data$config$pmtool$kiteration$legend,  
  x_start = data$config$limits$start,  
  x_end = data$config$limits$end  
)
```

**Arguments**

data	starvz_data with trace data
legend	enable/disable legends
x_start	X-axis start value
x_end	X-axis end value

**Value**

A ggplot object

**Examples**

```
panel_pmtool_kiteration(data = starvz_sample_lu)
```

---

panel_pmtool_st	<i>Create a space time visualization of pmtool application as a Gantt chart</i>
-----------------	---

---

**Description**

Use the PMTOOL Application trace data to plot the task computations by ResourceId over the execution time.

**Usage**

```
panel_pmtool_st(
  data = NULL,
  legend = data$config$pmtool$state$legend,
  base_size = data$config$base_size,
  expand_x = data$config$expand,
  x_start = data$config$limits$start,
  x_end = data$config$limits$end
)
```

**Arguments**

data	starvz_data with trace data
legend	enable/disable legends
base_size	base_size base font size
expand_x	expand size for scale_x_continuous padding
x_start	X-axis start value
x_end	X-axis end value

**Value**

A ggplot object

**Examples**

```
panel_pmtool_st(data = starvz_sample_lu)
```

---

panel\_power\_imbalance *Create a line chart with heterogeneous imbalance metrics.*

---

## Description

This function creates a line chart with imbalance metrics. The function applies the metrics on fixed time-steps defined by the user. The metrics consider that the resources are heterogeneous and defined by a constant power factor. For the effects of this function, one task is select for computing the relative power between resources.

## Usage

```
panel_power_imbalance(  
  data,  
  legend = data$config$power_imbalance$legend,  
  base_size = data$config$base_size,  
  expand_x = data$config$expand,  
  x_start = data$config$limits$start,  
  x_end = data$config$limits$end,  
  y_start = 0,  
  y_end = data$config$power_imbalance$limit,  
  step = data$config$power_imbalance$step,  
  task = data$config$power_imbalance$task  
)
```

## Arguments

data	starvz_data with trace data
legend	enable/disable legends
base_size	base_size base font size
expand_x	expand size for scale_x_continuous padding
x_start	X-axis start value
x_end	X-axis end value
y_start	Y-axis start value
y_end	Y-axis end value
step	time step for aggregation
task	Task used to computer relative resource power.

## Value

A ggplot object

**Examples**

```
panel_power_imbalance(data = starvz_sample_lu)
```

---

panel_progress	<i>Create the progress panel</i>
----------------	----------------------------------

---

**Description**

The progress panel show a progress metric per node, clustering nodes with a similar metric

**Usage**

```
panel_progress(df_app, nsteps, cluster_option,
               func_progress_node = NULL,
               func_progress_global = NULL,
               cluster_func = "Mode Density",
               show_abe = TRUE,
               plot_node_lines = FALSE,
               plot_cluster_info = FALSE,
               abe_label_pos_y = 0.05,
               abe_label_pos_x = 1.02)
```

**Arguments**

df_app	starvz_data Application data
nsteps	integer Number of times steps
cluster_option	numeric In case of "Mode Density", the bandwidth
func_progress_node	function progress funcion per node that return [0-1]
func_progress_global	function progress function globaly that return [0-1]
cluster_func	string "Mode Density" or "GMM"
show_abe	boolean Add abe to plots
plot_node_lines	boolean Add to return list the progress metric non-clustered
plot_cluster_info	boolean Add to return list the plot of cluster information
abe_label_pos_y	numeric ajust ABE label in y
abe_label_pos_x	numeric ajust ABE label in x

**Value**

List, steps - numeric list of steps, step - numeric the number of steps, original\_metrics - ggplot, if plot\_node\_lines is true, plot\_den - ggplot, if plot\_cluster\_info is true, joined\_data - tibble, cluster data computed, cluster\_metrics - ggplot the progress cluster visualization

**Examples**

```
panel_progress(starvz_sample_lu$Application, 20, 0.01, show_abe = FALSE)
```

---

panel_ready	<i>Create a line chart panel with ready tasks submission</i>
-------------	--

---

**Description**

Use the Variable traces to create a line chart panel with ready tasks submission per node, aggregated by a configurable time step

**Usage**

```
panel_ready(
  data,
  legend = data$config$ready$legend,
  base_size = data$config$base_size,
  expand_x = data$config$expand,
  x_start = data$config$limits$start,
  x_end = data$config$limits$end,
  y_start = 0,
  y_end = data$config$ready$limit,
  step = data$config$ready$step,
  lack_ready = data$config$ready$lack_ready$active
)
```

**Arguments**

data	starvz_data with trace data
legend	enable/disable legends
base_size	base_size base font size
expand_x	expand size for scale_x_continuous padding
x_start	X-axis start value
x_end	X-axis end value
y_start	Y-axis start value
y_end	Y-axis end value
step	time step for aggregation
lack_ready	show lack ready area in this panel

**Value**

A ggplot object

**Examples**

```
panel_ready(data = starvz_sample_lu)
```

---

```
panel_resource_usage_task
```

*Plot resource utilization using tasks as color*

---

**Description**

Use data Application to create a panel of the total resource utilization that helps to observe the time related resource utilization by task

**Usage**

```
panel_resource_usage_task(  
  data = NULL,  
  step = NULL,  
  legend = FALSE,  
  x_start = data$config$limits$start,  
  x_end = data$config$limits$end  
)
```

**Arguments**

data	starvz_data with trace data
step	size in milliseconds for the time aggregation step
legend	enable/disable plot legends
x_start	X-axis start value
x_end	X-axis end value

**Value**

A ggplot object

**Examples**

```
panel_resource_usage_task(data = starvz_sample_lu)
```

---

panel\_st                      *Create a space time visualization of the application as a Gantt chart*

---

### Description

Use the Application trace data to plot the task computations by ResourceId over the execution time. It will select the aggregation mode if requested.

### Usage

```
panel_st(
  data,
  agg = data$config$st$aggregation$active,
  agg_met = data$config$st$aggregation$method
)
```

### Arguments

data	starvz_data with trace data
agg	boolean Active or not aggregation
agg_met	Aggregation method, possible: static, dynamic, nodes

### Value

A ggplot object

### Examples

```
panel_st(data = starvz_sample_lu)
```

---

panel\_st\_agg\_dynamic    *Create a space-time visualization with dynamic aggregation.*

---

### Description

Use any state trace data to plot the task computations by ResourceId over the execution time with Gantt Chart. This function dynamically aggregate states with a dynamic/automatic time-step.

### Usage

```
panel_st_agg_dynamic(
  data = NULL,
  x_start = data$config$limits$start,
  x_end = data$config$limits$end,
  expand_x = data$config$expand,
  expand_y = data$config$st$expand
)
```

**Arguments**

data	starvz_data with trace data
x_start	X-axis start value
x_end	X-axis end value
expand_x	expand size for scale_x_continuous padding
expand_y	expand size for scale_y_continuous padding

**Value**

A ggplot object

**Examples**

```
panel_st_agg_dynamic(data = starvz_sample_lu)
```

---

panel\_st\_agg\_node      *Create a space-time visualization with node aggregation.*

---

**Description**

Use any state trace data to plot the task computations by Node over the execution time with Gantt Chart. This function aggregate states within the same resource type.

**Usage**

```
panel_st_agg_node(
  data,
  x_start = data$config$limits$start,
  x_end = data$config$limits$end,
  step = data$config$st$aggregation$step,
  legend = data$config$st$legend,
  selected_nodes = data$config$selected_nodes
)
```

**Arguments**

data	starvz_data with trace data
x_start	X-axis start value
x_end	X-axis end value
step	time-step
legend	option to activate legend
selected_nodes	select only some nodes in some plots

**Value**

A ggplot object

**Examples**

```
panel_st_agg_node(data = starvz_sample_lu)
```

---

panel\_st\_agg\_static     *Create a space-time visualization with static aggregation.*

---

**Description**

Use any state trace data to plot the task computations by ResourceId over the execution time with Gantt Chart. This function aggregate states with a static/user-defined time-step.

**Usage**

```
panel_st_agg_static(
  data = NULL,
  runtime = FALSE,
  x_start = data$config$limits$start,
  x_end = data$config$limits$end,
  expand_x = data$config$expand,
  expand_y = data$config$st$expand,
  outliers = data$config$st$outliers,
  step = data$config$st$aggregation$step
)
```

**Arguments**

data	starvz_data with trace data
runtime	if this is runtime data
x_start	X-axis start value
x_end	X-axis end value
expand_x	expand size for scale_x_continuous padding
expand_y	expand size for scale_y_continuous padding
outliers	print outliers on top
step	time-step

**Value**

A ggplot object

**Examples**

```
panel_st_agg_static(data = starvz_sample_lu)
```

---

panel_st_raw	<i>Create a space time visualization as a Gantt chart</i>
--------------	---

---

**Description**

Use the Application trace data to plot the task computations by ResourceId over the execution time.

**Usage**

```
panel_st_raw(
  data = NULL,
  ST.Outliers = data$config$st$outliers,
  base_size = data$config$base_size,
  expand_x = data$config$expand,
  expand_y = data$config$st$expand,
  selected_nodes = data$config$selected_nodes,
  labels = data$config$st$labels,
  alpha = data$config$st$alpha,
  idleness = data$config$st$idleness,
  taskdeps = data$config$st$tasks$active,
  tasklist = data$config$st$tasks$list,
  levels = data$config$st$tasks$levels,
  makespan = data$config$st$makespan,
  abe = data$config$st$abe$active,
  pmtoolbounds = data$config$pmtool$bounds$active,
  cpb = data$config$st$cpb,
  cpb_mpi = data$config$st$cpb_mpi$active,
  legend = data$config$st$legend,
  x_start = data$config$limits$start,
  x_end = data$config$limits$end,
  drop_small = data$config$st$drop_small,
  runtime = FALSE
)
```

**Arguments**

data	starvz_data with trace data
ST.Outliers	enable/disable the anomalous task highlighting
base_size	base_size base font size
expand_x	expand size for scale_x_continuous padding
expand_y	expand size for scale_y_continuous padding

selected_nodes	select only some nodes in some plots
labels	labels: [ALL, 1CPU_per_NODE, 1GPU_per_NODE, FIRST_LAST]
alpha	alpha value for non-anomalous tasks
idleness	enable/disable idleness percentages in the plot
taskdeps	enable/disable task deps path highlighting
tasklist	list of JobIds to highlight the dependencies
levels	number of dependencies to be shown
makespan	enable/disable application makespan at the end of the plot
abe	enable/disable ABE metric
pmtreebounds	enable/disable pmtree theoretical bounds
cpb	enable/disable critical path bound makespan metric
cpb_mpi	enable/disable critical path bound makespan considering MPI
legend	enable/disable legends
x_start	X-axis start value
x_end	X-axis end value
drop_small	Drop states smaller than this value
runtime	TODO I think we should create a separated function for it

**Value**

A ggplot object

**Examples**

```
panel_st_raw(data = starvz_sample_lu)
```

---

panel\_st\_runtime      *Create a space time visualization of the runtime as a Gantt chart*

---

**Description**

Use the runtime trace data to plot the task computations by ResourceId over the execution time. It will select the aggregation mode if requested, only static aggregation is available for runtime.

**Usage**

```
panel_st_runtime(data,
                 agg = data$config$starpup$aggregation$active)
```

**Arguments**

data	starvz_data with trace data
agg	Active or not static aggregation

**Value**

A ggplot object

**Examples**

```
panel_st_runtime(data = starvz_sample_lu)
```

---

panel_submitted	<i>Create a line chart panel with submitted tasks submission</i>
-----------------	--

---

**Description**

Use the Variable traces to create a line chart panel with submitted tasks submission per node, aggregated by a configurable time step

**Usage**

```
panel_submitted(
  data,
  legend = data$config$submitted$legend,
  base_size = data$config$base_size,
  expand_x = data$config$expand,
  x_start = data$config$limits$start,
  x_end = data$config$limits$end,
  y_start = 0,
  y_end = data$config$submitted$limit,
  step = data$config$submitted$step
)
```

**Arguments**

data	starvz_data with trace data
legend	enable/disable legends
base_size	base_size base font size
expand_x	expand size for scale_x_continuous padding
x_start	X-axis start value
x_end	X-axis end value
y_start	Y-axis start value
y_end	Y-axis end value
step	time step for aggregation

**Value**

A ggplot object

**Examples**

```
panel_submitted(data = starvz_sample_lu)
```

---

panel_title	<i>Create the title of StarVZ plot</i>
-------------	--

---

**Description**

Use the directory of traces name to create a plot title

**Usage**

```
panel_title(data, title = data$config$title$text)
```

**Arguments**

data	starvz_data with trace data
title	title text, if NULL it will fallback to data\$Origin then to "Null Title"

**Value**

A ggplot object

**Examples**

```
panel_title(data = starvz_sample_lu)
```

---

panel_usedmemory	<i>Create a line chart panel with used memory</i>
------------------	---

---

**Description**

Use the Variable traces to create a line chart panel with used memory per resource, aggregated by a configurable time step

**Usage**

```
panel_usedmemory(
  data,
  legend = data$config$usedmemory$legend,
  base_size = data$config$base_size,
  expand_x = data$config$expand,
  x_start = data$config$limits$start,
  x_end = data$config$limits$end,
  y_start = 0,
  y_end = data$config$usedmemory$limit,
  step = data$config$usedmemory$step
)
```

**Arguments**

data	starvz_data with trace data
legend	enable/disable legends
base_size	base_size base font size
expand_x	expand size for scale_x_continuous padding
x_start	X-axis start value
x_end	X-axis end value
y_start	Y-axis start value
y_end	Y-axis end value
step	time step for aggregation

**Value**

A ggplot object

**Examples**

```
panel_usedmemory(data = starvz_sample_lu)
```

---

panel_utilheatmap	<i>Create a heatmap of resource utilization</i>
-------------------	---

---

**Description**

Similar to the other resource oriented plots but shows the utilization per time step

**Usage**

```

panel_utilheatmap(
  data,
  legend = data$config$utilheatmap$legend,
  base_size = data$config$base_size,
  expand_x = data$config$expand,
  x_start = data$config$limits$start,
  x_end = data$config$limits$end,
  y_start = 0,
  y_end = NA,
  step = data$config$utilheatmap$step
)

```

**Arguments**

data	starvz_data with trace data
legend	enable/disable legends
base_size	base_size base font size
expand_x	expand size for scale_x_continuous padding
x_start	X-axis start value
x_end	X-axis end value
y_start	Y-axis start value
y_end	Y-axis end value
step	time step for aggregation

**Value**

A ggplot object

**Examples**

```
panel_utilheatmap(data = starvz_sample_lu)
```

---

panel\_utiltreedepth *Create the resource utilization by tree depth plot*

---

**Description**

Use starvz\_data Application and Atree to create a plot that shows the total resource utilization, painted by tree depth level using geom\_ribbon

**Usage**

```
panel_utiltreedepth(
  data,
  step = data$config$utiltreenode$step,
  x_start = data$config$limits$start,
  x_end = data$config$limits$end,
  legend = data$config$utiltreedepth$legend
)
```

**Arguments**

data	starvz_data with trace data
step	size in milliseconds for the time aggregation step
x_start	X-axis start value
x_end	X-axis end value
legend	enable/disable plot legends

**Value**

A ggplot object

**Examples**

```
## Not run:
panel_utiltreedepth(starvz_sample_lu, step = 100, legend = TRUE)

## End(Not run)
```

---

panel\_utiltreenode      *Create the resource utilization by tree node plot*

---

**Description**

Use starvz\_data Application and Atree to create a plot that shows the total resource utilization, painted by tree node using geom\_ribbon. The colors are reused between nodes, not tied to a specific tree node.

**Usage**

```
panel_utiltreenode(
  data = NULL,
  step = data$config$utiltreenode$step,
  x_start = data$config$limits$start,
  x_end = data$config$limits$end
)
```

**Arguments**

data	starvz_data with trace data
step	size in milliseconds for the time aggregation step
x_start	X-axis start value
x_end	X-axis end value

**Value**

A ggplot object

**Examples**

```
## Not run:  
panel_utiltreenode(data = starvz_sample_lu, step = 100)  
  
## End(Not run)
```

---

plot.starvz\_data      *Plot starvz\_data*

---

**Description**

Plot starvz\_data

**Usage**

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

**Arguments**

x	A starvz_data
...	optional

**Value**

Nothing

**Examples**

```
plot(starvz_sample_lu)
```

---

pre\_handle\_gantt      *Pre-Computation for memory handles panel*

---

**Description**

Use for precomputation of memory handles panel

**Usage**

```
pre_handle_gantt(data, name_func = NULL)
```

**Arguments**

data	starvz_data with trace data
name_func	function to give names to handles

**Value**

Pre-Computed data for panel\_handles

**Examples**

```
pre_handle_gantt(data = starvz_sample_lu)
```

---

print.starvz\_data      *Print starvz\_data*

---

**Description**

Print starvz\_data

**Usage**

```
## S3 method for class 'starvz_data'  
print(x, ...)
```

**Arguments**

x	A starvz_data
...	optional

**Value**

Nothing

**Examples**

```
print(starvz_sample_lu)
```

---

qrmumps_colors	<i>Colors for qr mumps</i>
----------------	----------------------------

---

**Description**

This will be deprecated

**Usage**

```
qrmumps_colors()
```

---

resource_utilization_tree_node	<i>Create the node memory usage plot</i>
--------------------------------	--

---

**Description**

Use starvz\_data to create a line plot of the memory usage in MB of active nodes along the application execution time

**Usage**

```
resource_utilization_tree_node(  
  Application = NULL,  
  Atree = NULL,  
  step = 100,  
  group_pruned = FALSE,  
  performance_metric = "Time"  
)
```

**Arguments**

Application	starvz application data
Atree	starvz elimination tree data
step	size in milliseconds for the time aggregation step
group_pruned	aggregate computations of the same parent pruned nodes
performance_metric	Performance metric to save in Value1 [Time, GFlops]

---

starvz_assemble	<i>Assemble multiple StarVZ panel lists</i>
-----------------	---

---

**Description**

Take a panel list, or a list of list of panels and assemble it

**Usage**

```
starvz_assemble(
  ...,
  config = NULL,
  remove_Y_info = TRUE,
  remove_legends = TRUE
)
```

**Arguments**

...	Panel list or list of panel lists
config	StarVZ configurations for determining panels heights
remove_Y_info	remove Y labels for a second and subsequent list of panels
remove_legends	remove legends for a second and subsequent list of panels

**Value**

The ggplot plot

**Examples**

```
starvz_assemble(starvz_plot_list(starvz_sample_lu),
  config = starvz_sample_lu$config
)
```

---

starvz_check_data	<i>Check if all required data is available</i>
-------------------	--

---

**Description**

The following conditions are checked in order and return FALSE if any fail - If data is not NULL - If data is a StarVZ Class - If data has all tables (given by the names of the list tables) - If each respective table has all columns (given the associated vector) - Execute extra\_func on data (that should return true or false)

**Usage**

```
starvz_check_data(data = NULL,
                 tables = list(), extra_func = NULL)
```

**Arguments**

data	starvz_data with trace data
tables	A named list (names are tables of data) of vectors (elements are columns), if tables is null continue
extra_func	Extra function to be applied on data to do a last check

**Value**

Logical, TRUE if data pass all tests

**Examples**

```
starvz_check_data(starvz_sample_lu,
                 tables = list("Comm_state" = c("Node")))
)
```

---

starvz_phase1	<i>Execute StarVZ Phase one.</i>
---------------	----------------------------------

---

**Description**

This function calls all CSV-converter inner-functions to pre-process they into StarVZ files. Although this can be directly used in a folder where all CSV compressed (gzip) files reside, we suggest to use the shell tool starvz or phase1-workflow.sh in the tools/ directory.

**Usage**

```
starvz_phase1(
  directory = ".",
  app_states_fun = lu_colors,
  state_filter = 0,
  whichApplication = "",
  input.parquet = "1",
  config_file = NULL
)
```

**Arguments**

directory	Directory of CSV files
app_states_fun	Function to determine application
state_filter	Type of filter
whichApplication	Name of Application
input.parquet	Use or not of parquet files
config_file	StarVZ config structure, this function uses only the app_tasks

**Value**

ggplot object with all starvz plots

**Examples**

```
example_folder <- system.file("extdata", "lu_trace", package = "starvz")
starvz_phase1(directory = example_folder)
```

---

starvz\_plot

*Make a StarVZ plot*


---

**Description**

Create a StarVZ plot considering the data supplied

**Usage**

```
starvz_plot(
  data = NULL,
  name = NULL,
  save = FALSE,
  guided = data$config$guided$active,
  dpi = 120
)
```

**Arguments**

data	starvz_data class with \$config
name	Path for saved image
save	call ggplot to save the image
guided	compute ideal figure height
dpi	dpi for ggsave

**Value**

ggplot object with all starvz plots

**Examples**

```
starvz_plot(starvz_sample_lu)
```

---

starvz_plot_list	<i>Generate the StarVZ Plots</i>
------------------	----------------------------------

---

**Description**

Use data to create the list of StarVZ plots

**Usage**

```
starvz_plot_list(data = NULL)
```

**Arguments**

data                    starvz\_data with trace data

**Value**

A list of ggplot plots

**Examples**

```
starvz_plot_list(starvz_sample_lu)
```

---

starvz_read	<i>Read starvz trace files</i>
-------------	--------------------------------

---

**Description**

Read the directory of trace files (feather or parquet)  
and the configuration file, and return a starvz\_data class used in starvz functions

**Usage**

```
starvz_read(directory = ".",  
            config_file = NULL, selective = TRUE)
```

**Arguments**

directory	Directory path of trace files
config_file	Path for configuration yaml file
selective	if True, only read data needed for creating panels activated in config

**Value**

The starvz\_data with all tables

**Examples**

```
starvz_read("folder_to_parquet_files/")
starvz_read(
  directory = "folder_to_parquet_files/",
  config_file = "path_to_config.yaml"
)
starvz_read() # Read current directory
```

---

starvz\_read\_config     *Read config files*

---

**Description**

Read starvz configuration yaml files. This function is design to replace an already existing configuration on starvz data.

**Usage**

```
starvz_read_config(file = NULL, warn = TRUE)
```

**Arguments**

file	The path to a file
warn	Give a warn in case the config file is not found

**Value**

A list containing starvz configuration

**Examples**

```
example_file <- system.file("extdata", "config.yaml", package = "starvz")
config <- starvz_read_config(example_file)
```

---

starvz\_sample\_lu      *Small StarVZ data of LU Factorization*

---

**Description**

A small StarVZ data object obtained from Chameleon+StarPU LU Factorization Generated by:

```
library(starvz)
pre_phase1 <- starvz_phase1(system.file("extdata", "lu_trace",
                                       package = "starvz"),
                           lu_colors,
                           state_filter=2,
                           whichApplication="lu")
starvz_sample_lu <- starvz_read(system.file("extdata",
                                           "lu_trace",
                                           package = "starvz"),
                              system.file("extdata",
                                           "config.yaml",
                                           package = "starvz"),
                              selective=FALSE)
usethis::use_data(starvz_sample_lu)
```

**Usage**

```
starvz_sample_lu
```

**Format**

An object of class starvz\_data of length 24.

**Source**

```
starvz_phase1 and starvz_read
```

---

starvz\_set\_log      *Active internal debug logs*

---

**Description**

Active internal debug logs

**Usage**

```
starvz_set_log(state)
```

**Arguments**

state            Active or not logs

**Value**

Nothing

**Examples**

```
starvz_set_log(FALSE)
```

---

starvz\_transform\_olddata

*Try to convert old StarVZ data to the new type*

---

**Description**

Old StarVZ data are usually just a tibble

**Usage**

```
starvz_transform_olddata(data)
```

**Arguments**

data            starvz\_data old data

**Value**

starvz\_data the data converted

**Examples**

```
starvz_transform_olddata(starvz_sample_lu)
```

---

summary.starvz\_data    *Summary starvz\_data*

---

**Description**

Summary starvz\_data

**Usage**

```
## S3 method for class 'starvz_data'  
summary(object, ...)
```

**Arguments**

object	A starvz_data
...	optional

**Value**

Nothing

**Examples**

```
summary(starvz_sample_1u)
```

# Index

- \* **datasets**
  - starvz\_sample\_lu, 54
- \* **phase1 functions**
  - starvz\_phase1, 50
- cholesky\_colors, 4
- data\_name\_coordinates, 4
- data\_name\_handle, 5
- data\_name\_tag, 5
- handles\_presence\_states, 6
- lu\_colors, 6
- multiple\_snaps, 7
- panel\_abe\_solution, 8
- panel\_activenodes, 8
- panel\_atree, 9
- panel\_atree\_structure, 10
- panel\_compare\_tree, 11
- panel\_dist2d, 12
- panel\_gflops, 13
- panel\_gflops\_computed\_difference, 14
- panel\_gpubandwidth, 15
- panel\_handles, 16
- panel\_hete\_imbalance, 17
- panel\_imbalance, 18
- panel\_kiteration, 19
- panel\_lackready, 20
- panel\_memory\_heatmap, 20
- panel\_memory\_snap, 21
- panel\_memory\_state, 22
- panel\_model\_gflops, 23
- panel\_mpibandwidth, 24
- panel\_mpiconcurrent, 25
- panel\_mpiconcurrentout, 26
- panel\_mpistate, 27
- panel\_node\_events, 28
- panel\_node\_summary, 29
- panel\_nodememuse, 28
- panel\_pmtree\_kiteration, 30
- panel\_pmtree\_st, 31
- panel\_power\_imbalance, 32
- panel\_progress, 33
- panel\_ready, 34
- panel\_resource\_usage\_task, 35
- panel\_st, 36
- panel\_st\_agg\_dynamic, 36
- panel\_st\_agg\_node, 37
- panel\_st\_agg\_static, 38
- panel\_st\_raw, 39
- panel\_st\_runtime, 40
- panel\_submitted, 41
- panel\_title, 42
- panel\_usedmemory, 42
- panel\_utilheatmap, 43
- panel\_utiltreedepth, 44
- panel\_utiltreenode, 45
- plot.starvz\_data, 46
- pre\_handle\_gantt, 47
- print.starvz\_data, 47
- qrmumps\_colors, 48
- resource\_utilization\_tree\_node, 48
- starvz\_assemble, 49
- starvz\_check\_data, 49
- starvz\_phase1, 50
- starvz\_plot, 51
- starvz\_plot\_list, 52
- starvz\_read, 52
- starvz\_read\_config, 53
- starvz\_sample\_lu, 54
- starvz\_set\_log, 54
- starvz\_transform\_olddata, 55
- summary.starvz\_data, 56