*#### NICE REPRESENTATION*

*```{r, echo=TRUE}*

*set.seed(1) ; obs1 <- data.frame(km = rnorm(1000, 10, 3), time = rnorm(1000, 10, 3), group1 = rep(c("A1", "A2"), 500)) ; obs2 <-data.frame(km = rnorm(1000, 15, 3), time = rnorm(1000, 15, 3), group2 = rep(c("G1", "G2"), 500)) ; set.seed(NULL) ; obs1$L1$km[2:3] <- NA ; fun\_gg\_scatter(data1 = list(L1 = obs1, L2 = obs2), x = list(L1 = "km", L2 = "km"), y = list(L1 = "time", L2 = "time"), categ = list(L1 = "group1", L2 = "group2"), legend.name = NULL, color = list(L1 = 4:5, L2 = 7:8), geom = list(L1 = "geom\_point", L2 = "geom\_point"), alpha = list(L1 = 0.5, L2 = 0.5), dot.size = 3, line.size = 0.5, xlim = c(1, 25), xlab = "KM", xlog = "no", x.tick.nb = 10, x.inter.tick.nb = 1, x.left.extra.margin = 0, x.right.extra.margin = 0, ylim = c(1, 25), ylab = expression(paste("TIME (", 10^-20, " s)")), ylog = "log10", y.tick.nb = 5, y.top.extra.margin = 0, y.bottom.extra.margin = 0, xy.include.zero = TRUE, classic = TRUE)*

*```*

*#### SINGLE GEOMETRIC LAYER*

*# simple example (1) of scatter plot using the classical writting*

*```{r, echo=TRUE}*

*obs1 <- data.frame(km = 2:7, time = (2:7)^2, group = c("A", "A", "A", "B", "B", "B")) ; obs1 ; fun\_gg\_scatter(data1 = obs1, x = "km", y = "time")*

*```*

*# simple example (2) of scatter plot, identical to (1) but using the list writting. Here, a list of one compartment, systematically named L1, is provided to the data1, x, y, categ, geom and alpha. Contrary to example (1), the geom and alpha argument have to be included because the default value are not lists (if data1 is a list, all the x, y, categ, legend.name, color, geom and alpha must also be list if non NULL)*

*```{r, echo=TRUE}*

*obs1 <- data.frame(km = 2:7, time = (2:7)^2, group = c("A", "A", "A", "B", "B", "B")) ; obs1 ; fun\_gg\_scatter(data1 = list(L1 = obs1), x = list(L1 = "km"), y = list(L1 = "time"), geom = list(L1 = "geom\_point"), alpha = list(L1 = 0.5))*

*```*

*# color of dots. Example (1) using the classical writting*

*```{r, echo=TRUE}*

*obs1 <- data.frame(km = 2:7, time = (2:7)^2, group = c("A", "A", "A", "B", "B", "B")) ; obs1 ; fun\_gg\_scatter(data1 = obs1, x = "km", y = "time", color = "blue")*

*```*

*# color of dots. Example (2) using the list writting*

*```{r, echo=TRUE}*

*obs1 <- data.frame(km = 2:7, time = (2:7)^2, group = c("A", "A", "A", "B", "B", "B")) ; obs1 ; fun\_gg\_scatter(data1 = list(L1 = obs1), x = list(L1 = "km"), y = list(L1 = "time"), color = list(L1 = "blue"), geom = list(L1 = "geom\_point"), alpha = list(L1 = 1))*

*```*

*# From here, classical writting is use for single element in data1 and list writting otherwise*

*# color of dots. Example (3) when dots are in different categories. Note that categ argument controls the legend display*

*```{r, echo=TRUE}*

*obs1 <- data.frame(km = 2:7, time = (2:7)^2, group = c("A", "A", "A", "B", "B", "B")) ; obs1 ; fun\_gg\_scatter(data1 = obs1, x = "km", y = "time", categ = "group")*

*```*

*# color of dots. Example (4) when dots are in different categories. A single color mentionned is applied to all the dots*

*```{r, echo=TRUE}*

*obs1 <- data.frame(km = 2:7, time = (2:7)^2, group = c("A", "A", "A", "B", "B", "B")) ; obs1 ; fun\_gg\_scatter(data1 = obs1, x = "km", y = "time", categ = "group", color = "coral")*

*```*

*# color of dots. Example (5) when dots are in different categories. Numbers can be used if ggplot colors are desired*

*```{r, echo=TRUE}*

*obs1 <- data.frame(km = 2:7, time = (2:7)^2, group = c("A", "A", "A", "B", "B", "B")) ; obs1 ; fun\_gg\_scatter(data1 = obs1, x = "km", y = "time", categ = "group", color = 2)*

*```*

*# color of dots. Example (6) when dots are in different categories, with one color per category (try also color = 2:1)*

*```{r, echo=TRUE}*

*obs1 <- data.frame(km = 2:7, time = (2:7)^2, group = c("A", "A", "A", "B", "B", "B")) ; obs1 ; fun\_gg\_scatter(data1 = obs1, x = "km", y = "time", categ = "group", color = c("coral", "green"))*

*```*

*# color of dots. Example (7) when dots are in different categories, with colors as a data frame column. BEWARE: one color per category must be respected (try also numbers)*

*```{r, echo=TRUE}*

*obs1 <- data.frame(km = 2:7, time = (2:7)^2, group = c("A", "A", "A", "B", "B", "B"), col = rep(c("coral", "green"), each = 3)) ; obs1 ; fun\_gg\_scatter(data1 = obs1, x = "km", y = "time", categ = "group", color = obs1$col)*

*```*

*# color of dots. Example (8) when dots are in different categories, with colors as a data frame column. Easiest way (ggplot colors)*

*```{r, echo=TRUE}*

*obs1 <- data.frame(km = 2:7, time = (2:7)^2, group = c("A", "A", "A", "B", "B", "B")) ; obs1 ; fun\_gg\_scatter(data1 = obs1, x = "km", y = "time", categ = "group", color = as.numeric(obs1$group))*

*```*

*# legend name*

*```{r, echo=TRUE}*

*obs1 <- data.frame(km = 2:7, time = (2:7)^2, group = c("A", "A", "A", "B", "B", "B")) ; obs1 ; fun\_gg\_scatter(data1 = obs1, x = "km", y = "time", categ = "group", legend.name = "CLASSES")*

*```*

*# different geom features. Example (1) with geom\_line kind of lines*

*```{r, echo=TRUE}*

*obs1 <- data.frame(km = c(1, 3, 2, 6, 4, 5), time = c(1, 3, 2, 6, 4, 5)^2, group = c("A", "A", "A", "B", "B", "B")) ; obs1 ; fun\_gg\_scatter(data1 = obs1, x = "km", y = "time", geom = "geom\_line", categ = "group")*

*```*

*# different geom features. Example (2) with geom\_path kind of lines (see the difference with (1))*

*```{r, echo=TRUE}*

*obs1 <- data.frame(km = c(1, 3, 2, 6, 4, 5), time = c(1, 3, 2, 6, 4, 5)^2, group = c("A", "A", "A", "B", "B", "B")) ; obs1 ; fun\_gg\_scatter(data1 = obs1, x = "km", y = "time", geom = "geom\_path", categ = "group")*

*```*

*# different geom features. Example (3) with geom\_hline kind of lines. Fake\_y y-axis name by default because y argument must be NULL (see ylab argument below to change this)*

*```{r, echo=TRUE}*

*obs1 <- data.frame(km = 1:2, time = (1:2)^2, group = c("A", "B")) ; obs1 ; fun\_gg\_scatter(data1 = obs1, x = NULL, y = "km", geom = "geom\_hline", categ = "group", xlim = c(1,10))*

*```*

*# different geom features. Example (4) with geom\_vline kind of lines. Fake\_y y-axis name by default because y argument must be NULL (see ylab argument below to change this)*

*```{r, echo=TRUE}*

*obs1 <- data.frame(km = 1:2, time = (1:2)^2, group = c("A", "B")) ; obs1 ; fun\_gg\_scatter(data1 = obs1, x = "km", y = NULL, geom = "geom\_vline", categ = "group", ylim = c(1,10))*

*#### MULTI GEOMETRIC LAYERS*

*# Note that in subsequent examples, names of list compartments are systematically referred to as L1, L2, etc., to show the correspondence between the arguments data1, x, y, categ, etc.*

*# single layer (as examples above)*

*```*

*```{r, echo=TRUE}*

*set.seed(1) ; obs1 <- data.frame(km = rnorm(1000, 22, 3), time = rnorm(1000, 22, 3)) ; set.seed(NULL) ; fun\_gg\_scatter(data1 = list(L1 = obs1), x = list(L1 = "km"), y = list(L1 = "time"), geom = list(L1 = "geom\_point"), alpha = list(L1 = 0.5))*

*```*

*# simple example of two layers*

*```{r, echo=TRUE}*

*set.seed(1) ; obs1 <- data.frame(km = rnorm(1000, 22, 3), time = rnorm(1000, 22, 3)) ; obs2 <-data.frame(km = rnorm(1000, 30, 3), time = rnorm(1000, 30, 3)) ; set.seed(NULL) ; fun\_gg\_scatter(data1 = list(L1 = obs1, L2 = obs2), x = list(L1 = "km", L2 = "km"), y = list(L1 = "time", L2 = "time"), geom = list(L1 = "geom\_point", L2 = "geom\_point"), alpha = list(L1 = 0.5, L2 = 0.5))*

*```*

*# color of dots. Example (1)*

*```{r, echo=TRUE}*

*set.seed(1) ; obs1 <- data.frame(km = rnorm(1000, 22, 3), time = rnorm(1000, 22, 3)) ; obs2 <-data.frame(km = rnorm(1000, 30, 3), time = rnorm(1000, 30, 3)) ; set.seed(NULL) ; fun\_gg\_scatter(data1 = list(L1 = obs1, L2 = obs2), x = list(L1 = "km", L2 = "km"), y = list(L1 = "time", L2 = "time"), geom = list(L1 = "geom\_point", L2 = "geom\_point"), alpha = list(L1 = 0.5, L2 = 0.5), color = list(L1 = "coral", L2 = "green"))*

*```*

*# color of dots. Example (2) of the legend display. The categ argument must be supplied. Make a fake categorical colum in the data frame if necessary (as in this example). The categ argument triggers the legend display. The legend.name argument is used to remove the legend title of each layer*

*```{r, echo=TRUE}*

*set.seed(1) ; obs1 <- data.frame(km = rnorm(1000, 22, 3), time = rnorm(1000, 22, 3), group1 = "GROUP1") ; obs2 <-data.frame(km = rnorm(1000, 30, 3), time = rnorm(1000, 30, 3), group2 = "GROUP2") ; set.seed(NULL) ; fun\_gg\_scatter(data1 = list(L1 = obs1, L2 = obs2), x = list(L1 = "km", L2 = "km"), y = list(L1 = "time", L2 = "time"), categ = list(L1 = "group1", L2 = "group2"), legend.name = list(L1 = NULL, L2 = NULL), geom = list(L1 = "geom\_point", L2 = "geom\_point"), alpha = list(L1 = 0.5, L2 = 0.5), color = list(L1 = "coral", L2 = "green"))*

*```*

*# color of dots. Example (3) when dots are in different categories (default colors)*

*```{r, echo=TRUE}*

*set.seed(1) ; obs1 <- data.frame(km = rnorm(1000, 22, 3), time = rnorm(1000, 22, 3), group1 = rep(c("A1", "A2"), each = 500)) ; obs2 <-data.frame(km = rnorm(1000, 30, 3), time = rnorm(1000, 30, 3), group2 = rep(c("G1", "G2"), each = 500)) ; set.seed(NULL) ; fun\_gg\_scatter(data1 = list(L1 = obs1, L2 = obs2), x = list(L1 = "km", L2 = "km"), y = list(L1 = "time", L2 = "time"), , categ = list(L1 = "group1", L2 = "group2"), geom = list(L1 = "geom\_point", L2 = "geom\_point"), alpha = list(L1 = 0.5, L2 = 0.5))*

*```*

*# color of dots. Example (3) when dots are in different categories. A single color mentionned per layer is applied to all the dots of the layer*

*```{r, echo=TRUE}*

*set.seed(1) ; obs1 <- data.frame(km = rnorm(1000, 22, 3), time = rnorm(1000, 22, 3), group1 = rep(c("A1", "A2"), each = 500)) ; obs2 <-data.frame(km = rnorm(1000, 30, 3), time = rnorm(1000, 30, 3), group2 = rep(c("G1", "G2"), each = 500)) ; set.seed(NULL) ; fun\_gg\_scatter(data1 = list(L1 = obs1, L2 = obs2), x = list(L1 = "km", L2 = "km"), y = list(L1 = "time", L2 = "time"), , categ = list(L1 = "group1", L2 = "group2"), geom = list(L1 = "geom\_point", L2 = "geom\_point"), alpha = list(L1 = 0.5, L2 = 0.5), color = list(L1 = "coral", L2 = "green"))*

*```*

*# color of dots. Example (5) when dots are in different categories, with one color per category in each layer*

*```{r, echo=TRUE}*

*set.seed(1) ; obs1 <- data.frame(km = rnorm(1000, 22, 3), time = rnorm(1000, 22, 3), group1 = rep(c("A1", "A2"), each = 500)) ; obs2 <-data.frame(km = rnorm(1000, 30, 3), time = rnorm(1000, 30, 3), group2 = rep(c("G1", "G2"), each = 500)) ; set.seed(NULL) ; fun\_gg\_scatter(data1 = list(L1 = obs1, L2 = obs2), x = list(L1 = "km", L2 = "km"), y = list(L1 = "time", L2 = "time"), , categ = list(L1 = "group1", L2 = "group2"), geom = list(L1 = "geom\_point", L2 = "geom\_point"), alpha = list(L1 = 0.5, L2 = 0.5), color = list(L1 = c("coral", "blue"), L2 = c("green", "black")))*

*```*

*# color of dots. Example (4) when dots are in different categories. Numbers can be used if ggplot colors are desired*

*```{r, echo=TRUE}*

*set.seed(1) ; obs1 <- data.frame(km = rnorm(1000, 22, 3), time = rnorm(1000, 22, 3), group1 = rep(c("A1", "A2"), each = 500)) ; obs2 <-data.frame(km = rnorm(1000, 30, 3), time = rnorm(1000, 30, 3), group2 = rep(c("G1", "G2"), each = 500)) ; set.seed(NULL) ; fun\_gg\_scatter(data1 = list(L1 = obs1, L2 = obs2), x = list(L1 = "km", L2 = "km"), y = list(L1 = "time", L2 = "time"), , categ = list(L1 = "group1", L2 = "group2"), geom = list(L1 = "geom\_point", L2 = "geom\_point"), alpha = list(L1 = 0.5, L2 = 0.5), color = list(L1 = 1:2, L2 = c(4, 7)))*

*```*

*# color of dots. Example (7) when dots are in different categories, with colors as a data frame column. BEWARE: one color per category must be respected (try also numbers). BEWARE: in color argument, if the column of the data frame does not exist, color can be still displayed (L2 = obs2$notgood is equivalent to L2 = NULL). Such situation is reported in the warning messages (see below)*

*```{r, echo=TRUE}*

*set.seed(1) ; obs1 <- data.frame(km = rnorm(1000, 22, 3), time = rnorm(1000, 22, 3), group1 = rep(c("A1", "A2"), each = 500), col1 = rep(c("coral", "blue"), each = 500)) ; obs2 <-data.frame(km = rnorm(1000, 30, 3), time = rnorm(1000, 30, 3), group2 = rep(c("G1", "G2"), each = 500), col2 = rep(c("green", "black"), each = 500)) ; set.seed(NULL) ; fun\_gg\_scatter(data1 = list(L1 = obs1, L2 = obs2), x = list(L1 = "km", L2 = "km"), y = list(L1 = "time", L2 = "time"), , categ = list(L1 = "group1", L2 = "group2"), geom = list(L1 = "geom\_point", L2 = "geom\_point"), alpha = list(L1 = 0.5, L2 = 0.5), color = list(L1 = obs1$col1, L2 = obs2$col2))*

*```*

*# color of dots. Example (8) when dots are in different categories, with colors as a data frame column. Easiest way is not recommended with mutiple layers*

*```{r, echo=TRUE}*

*set.seed(1) ; obs1 <- data.frame(km = rnorm(1000, 22, 3), time = rnorm(1000, 22, 3), group1 = rep(c("A1", "A2"), each = 500), col1 = rep(c("coral", "blue"), each = 500)) ; obs2 <-data.frame(km = rnorm(1000, 30, 3), time = rnorm(1000, 30, 3), group2 = rep(c("G1", "G2"), each = 500), col2 = rep(c("green", "black"), each = 500)) ; set.seed(NULL) ; fun\_gg\_scatter(data1 = list(L1 = obs1, L2 = obs2), x = list(L1 = "km", L2 = "km"), y = list(L1 = "time", L2 = "time"), , categ = list(L1 = "group1", L2 = "group2"), geom = list(L1 = "geom\_point", L2 = "geom\_point"), alpha = list(L1 = 0.5, L2 = 0.5), color = list(L1 = as.numeric(obs1$group1), L2 = as.numeric(obs2$group2)))*

*```*

*# legend name*

*```{r, echo=TRUE}*

*set.seed(1) ; obs1 <- data.frame(km = rnorm(1000, 22, 3), time = rnorm(1000, 22, 3), group1 = rep(c("A1", "A2"), each = 500)) ; obs2 <-data.frame(km = rnorm(1000, 30, 3), time = rnorm(1000, 30, 3), group2 = rep(c("G1", "G2"), each = 500)) ; set.seed(NULL) ; fun\_gg\_scatter(data1 = list(L1 = obs1, L2 = obs2), x = list(L1 = "km", L2 = "km"), y = list(L1 = "time", L2 = "time"), , categ = list(L1 = "group1", L2 = "group2"), legend.name = list(L1 = "CLASS A", L2 = "CLASS G"), geom = list(L1 = "geom\_point", L2 = "geom\_point"), alpha = list(L1 = 0.5, L2 = 0.5))*

*```*

*# different geom features. Example (1) with 5 layers. Note that order in data1 defines the overlay order (from below to above) and the order in the legend (from top to bottom)*

*```{r, echo=TRUE}*

*set.seed(1) ; obs1 <- data.frame(km = rnorm(1000, 22, 3), time = rnorm(1000, 22, 3), group1 = rep(c("A1", "A2"), each = 500)) ; obs2 <-data.frame(km = rnorm(1000, 30, 3), time = rnorm(1000, 30, 3), group2 = rep(c("G1", "G2"), each = 500)) ; set.seed(NULL) ; obs3 <- data.frame(time = c(29, 31), group3 = c("HORIZ.THRESHOLD.1", "HORIZ.THRESHOLD.2")) ; obs4 <- data.frame(km = 26, group4 = "VERTIC.THRESHOLD") ; obs5 <- data.frame(km = seq(1, 100, 0.1), time = 7\*seq(1, 100, 0.1)^0.5, group5 = "FUNCTION") ; fun\_gg\_scatter(data1 = list(L1 = obs1, L2 = obs2, L3 = obs3, L4 = obs4, L5 = obs5), x = list(L1 = "km", L2 = "km", L3 = NULL, L4 = "km", L5 = "km"), y = list(L1 = "time", L2 = "time", L3 = "time", L4 = NULL, L5 = "time"), categ = list(L1 = "group1", L2 = "group2", L3 = "group3", L4 = "group4", L5 = "group5"), geom = list(L1 = "geom\_point", L2 = "geom\_point", L3 = "geom\_hline", L4 = "geom\_vline", L5 = "geom\_line"), alpha = list(L1 = 0.5, L2 = 0.5, L3 = 0.5, L4 = 0.5, L5 = 0.5), xlim = c(10, 40), ylim = c(10, 40), classic = TRUE, line.size = 0.75)*

*# layer transparency. One transparency defined by layer (from 0 invisible to 1 opaque). Note that for lines, transparency in not applied in the legend to prevent a ggplot2 bug (https://github.com/tidyverse/ggplot2/issues/2452)*

*```{r, echo=TRUE}*

*set.seed(1) ; obs1 <- data.frame(km = rnorm(1000, 22, 3), time = rnorm(1000, 22, 3), group1 = rep(c("A1", "A2"), each = 500)) ; obs2 <-data.frame(km = rnorm(1000, 30, 3), time = rnorm(1000, 30, 3), group2 = rep(c("G1", "G2"), each = 500)) ; set.seed(NULL) ; fun\_gg\_scatter(data1 = list(L1 = obs1, L2 = obs2), x = list(L1 = "km", L2 = "km"), y = list(L1 = "time", L2 = "time"), , categ = list(L1 = "group1", L2 = "group2"), geom = list(L1 = "geom\_point", L2 = "geom\_point"), alpha = list(L1 = 1, L2 = 0.1))*

*```*

*# other different example of mutiple geom features are shown in the fun\_segmentation function*

*#### OTHER GRAPHIC ARGUMENTS*

*# dot size (line.size argument controls size of lines)*

*```{r, echo=TRUE}*

*obs1 <- data.frame(km = 2:7, time = (2:7)^2, group = c("A", "A", "A", "B", "B", "B")) ; obs1 ; fun\_gg\_scatter(data1 = obs1, x = "km", y = "time", dot.size = 5)*

*# axis management: examples are shown for x-axis but are identical for y-axis*

*# x-axis limits. Example (1)*

*```{r, echo=TRUE}*

*obs1 <- data.frame(km = 2:7, time = (2:7)^2, group = c("A", "A", "A", "B", "B", "B")) ; obs1 ; fun\_gg\_scatter(data1 = obs1, x = "km", y = "time", xlim = c(-1, 25))*

*```*

*# x-axis limits. Example (2) showing that order matters in ylim argument*

*```{r, echo=TRUE}*

*obs1 <- data.frame(km = 2:7, time = (2:7)^2, group = c("A", "A", "A", "B", "B", "B")) ; obs1 ; fun\_gg\_scatter(data1 = obs1, x = "km", y = "time", xlim = c(25, -1))*

*```*

*# log scale. Example (1). BEWARE: x column must be log, otherwise incoherent scale (see below warning message with the return argument)*

*```{r, echo=TRUE}*

*obs1 <- data.frame(km = 2:7, time = (2:7)^2, group = c("A", "A", "A", "B", "B", "B")) ; obs1 ; fun\_gg\_scatter(data1 = obs1, x = "km", y = "time", xlog = "log10")*

*# log scale. Example (2). BEWARE: values of the xlim must be in the corresponding log*

*```{r, echo=TRUE}*

*obs1 <- data.frame(km = 2:7, time = (2:7)^2, group = c("A", "A", "A", "B", "B", "B")) ; obs1 ; fun\_gg\_scatter(data1 = obs1, x = "km", y = "time", xlog = "log10", xlim = c(1, 10))*

*```*

*# tick number. Example (1). Note that the final number shown is approximate*

*```{r, echo=TRUE}*

*obs1 <- data.frame(km = 2:7, time = (2:7)^2, group = c("A", "A", "A", "B", "B", "B")) ; obs1 ; fun\_gg\_scatter(data1 = obs1, x = "km", y = "time", x.tick.nb = 6)*

*# tick number. Example (2) using a log2 scale*

*```{r, echo=TRUE}*

*obs1 <- data.frame(km = 2:7, time = (2:7)^2, group = c("A", "A", "A", "B", "B", "B")) ; obs1 ; fun\_gg\_scatter(data1 = obs1, x = "km", y = "time", xlog = "log2", x.tick.nb = 6)*

*# tick number. Example (3) using a log10 scale*

*```{r, echo=TRUE}*

*obs1 <- data.frame(km = 2:7, time = (2:7)^2, group = c("A", "A", "A", "B", "B", "B")) ; obs1 ; fun\_gg\_scatter(data1 = obs1, x = "km", y = "time", xlog = "log10", x.tick.nb = 6)*

*# tick number. Example (4) using a log10 scale: the reverse x-axis correctly deal with log10 scale*

*```{r, echo=TRUE}*

*obs1 <- data.frame(km = 2:7, time = (2:7)^2, group = c("A", "A", "A", "B", "B", "B")) ; obs1 ; fun\_gg\_scatter(data1 = obs1, x = "km", y = "time", xlog = "log10", xlim = c(7, 2))*

*```*

*# secondary tick number. Example (1)*

*```{r, echo=TRUE}*

*obs1 <- data.frame(km = 2:7, time = (2:7)^2, group = c("A", "A", "A", "B", "B", "B")) ; obs1 ; fun\_gg\_scatter(data1 = obs1, x = "km", y = "time", x.inter.tick.nb = 4)*

*# secondary ticks. Example (2) not for log2 and log10 scales (see below warning message with the return argument)*

*```{r, echo=TRUE}*

*obs1 <- data.frame(km = 2:7, time = (2:7)^2, group = c("A", "A", "A", "B", "B", "B")) ; obs1 ; fun\_gg\_scatter(data1 = obs1, x = "km", y = "time", xlog = "log10", x.inter.tick.nb = 4)*

*# extra margins. To avoid dot cuts*

*```{r, echo=TRUE}*

*obs1 <- data.frame(km = 2:7, time = (2:7)^2, group = c("A", "A", "A", "B", "B", "B")) ; obs1 ; fun\_gg\_scatter(data1 = obs1, x = "km", y = "time", x.left.extra.margin = 0.25, x.right.extra.margin = 0.25)*

*# include zero in both the x-axis and y-xis*

*```{r, echo=TRUE}*

*obs1 <- data.frame(km = 2:7, time = (2:7)^2, group = c("A", "A", "A", "B", "B", "B")) ; obs1 ; fun\_gg\_scatter(data1 = obs1, x = "km", y = "time", xy.include.zero = TRUE)*

*# graph title, text size and legend display*

*```{r, echo=TRUE}*

*obs1 <- data.frame(km = 2:7, time = (2:7)^2, group = c("A", "A", "A", "B", "B", "B")) ; obs1 ; fun\_gg\_scatter(data1 = obs1, x = "km", y = "time", categ = "group", text.size = 8, title = "GRAPH1", title.text.size = 16, show.legend = TRUE)*

*# raster display. This switchs from vectorial mode to raster mode. The display can takes some time, but this is easier to export and handle than vectorial display*

*```{r, echo=TRUE}*

*set.seed(1) ; obs1 <- data.frame(km = rnorm(100000, 22, 3), time = rnorm(100000, 22, 3)) ; set.seed(NULL) ; fun\_gg\_scatter(data1 = obs1, x = "km", y = "time", raster = TRUE)*

*# classic representation (use grid = TRUE to display the background lines of the y axis ticks)*

*```{r, echo=TRUE}*

*obs1 <- data.frame(km = 2:7, time = (2:7)^2, group = c("A", "A", "A", "B", "B", "B")) ; obs1 ; fun\_gg\_scatter(data1 = obs1, x = "km", y = "time", classic = TRUE, grid = FALSE)*

*# graphic info. Example (1)*

*```{r, echo=TRUE}*

*obs1 <- data.frame(km = 2:7, time = (2:7)^2, group = c("A", "A", "A", "B", "B", "B")) ; fun\_gg\_scatter(data1 = obs1, x = "km", y = "time", return = TRUE)*

*# graphic info. Example (2) of assignation and warning message display*

*```{r, echo=TRUE}*

*obs1 <- data.frame(km = 2:7, time = (2:7)^2, group = c("A", "A", "A", "B", "B", "B")) ; output <- fun\_gg\_scatter(data1 = obs1, x = "km", y = "time", xlog = "log10", return = TRUE) ; cat(output$warnings)*

*# add ggplot2 functions*

*```{r, echo=TRUE}*

*obs1 <- data.frame(km = 2:7, time = (2:7)^2, group = c("A", "A", "A", "B", "B", "B")) ; obs1 ; fun\_gg\_scatter(data1 = obs1, x = "km", y = "time", add = "+ggplot2::theme\_classic()")*

*# all the arguments*

*```{r, echo=TRUE}*

*set.seed(1) ; obs1 <- data.frame(km = rnorm(1000, 10, 3), time = rnorm(1000, 10, 3), group1 = rep(c("A1", "A2"), 500)) ; obs2 <-data.frame(km = rnorm(1000, 15, 3), time = rnorm(1000, 15, 3), group2 = rep(c("G1", "G2"), 500)) ; set.seed(NULL) ; obs1$L1$km[2:3] <- NA ; fun\_gg\_scatter(data1 = list(L1 = obs1, L2 = obs2), x = list(L1 = "km", L2 = "km"), y = list(L1 = "time", L2 = "time"), categ = list(L1 = "group1", L2 = "group2"), legend.name = NULL, color = list(L1 = 4:5, L2 = 7:8), geom = list(L1 = "geom\_point", L2 = "geom\_point"), alpha = list(L1 = 0.5, L2 = 0.5), dot.size = 3, line.size = 0.5, xlim = c(1, 25), xlab = "KM", xlog = "no", x.tick.nb = 10, x.inter.tick.nb = 1, x.left.extra.margin = 0, x.right.extra.margin = 0, ylim = c(1, 25), ylab = "TIME (s)", ylog = "log10", y.tick.nb = 5, y.inter.tick.nb = NULL, y.top.extra.margin = 0, y.bottom.extra.margin = 0, xy.include.zero = TRUE, text.size = 12, title = "", title.text.size = 8, show.legend = TRUE, classic = FALSE, grid = FALSE, raster = FALSE, vectorial.limit = NULL, return = FALSE, plot = TRUE, add = NULL, warn.print = TRUE, path.lib = NULL)*