create\_sensitivities\_figure.R

z3254626

2021-07-07

.. content for (no empty lines) ..

.. content for ..

@title @param variable

@param variable

create\_sensitivities\_figure <- function(measures = c("H1b.MI.AvLast"),   
 colours = c("black"),  
 p.start = 0,  
 p.end = 1,  
 n.samples = 20,  
 n.loci = 1000,  
 n.pops = 10) {  
   
 #Plot the intensity of a step as Distance (d) vs. Allele Proportion (p)  
 vis.steps <- function(step) {  
   
 step\_plot <- ggplot(data = simulate\_data(step = step,  
 p.start = p.start,   
 p.end = p.end,  
 n.samples = n.samples,  
 n.loci = n.loci,  
 n.pops = n.pops)[[2]],  
 aes(d, p\_mean)) +  
 geom\_point() +  
 geom\_line() +  
 xlab("Distance (d)") +  
 ylab("Allele Proportion (p)") +  
 theme\_classic()  
   
 return(step\_plot)  
 }  
   
 #Plot the step sensitivities while varying an input (n.samples, n.loci, n.pops)  
 vis.sensitivities <- function(step, variable) {  
   
 #Which tested variable is being visualised?  
 if (variable == "n.samples") {  
 data <- readd(step\_sensitivities\_samples)  
   
 label\_for\_x <- "Number of samples (n)"  
 }  
   
 if (variable == "n.loci") {  
 data <- readd(step\_sensitivities\_loci)  
   
 label\_for\_x <- "Number of loci (L)"  
 }  
   
 if (variable == "n.pops") {  
 data <- readd(step\_sensitivities\_pops)  
   
 label\_for\_x <- "Number of populations (K)"  
 }  
   
 p <- ggplot()+  
 ylab("Step detections (out of 100)") +  
 geom\_vline(xintercept = get(variable),   
 linetype = "dotted") +  
 ylim(0,100) +  
 theme\_classic()  
   
 for (i in seq(1, length(measures))) {  
 p <- p +  
 geom\_line(data = filter(data, step == !!step & p.start == !!p.start & p.end == !!p.end),  
 aes(x= !!sym(variable), y = !!sym(measures[[i]])), colour = colours[[i]], linetype = "dashed") +  
 xlab(label = label\_for\_x)  
   
 if (step != 0) { #this makes sure no "correct" steps are plotted for step = 0  
 p <- p +  
 geom\_line(data = filter(data, step == !!step & p.start == !!p.start & p.end == !!p.end),  
 aes(x = !!sym(variable), y = !!sym(paste0(measures[[i]], "\_correct"))), colour = colours[[i]]) +  
 xlab(label = label\_for\_x)  
   
 }  
 }  
   
 return(p)  
 }  
   
 #Plot an example single simulation with the specified inputs  
 vis.betas <- function(step){  
   
 beta\_plot <- plot\_betas(simulate\_data(step = step,  
 p.start = p.start,   
 p.end = p.end,  
 n.samples = n.samples,  
 n.loci = n.loci,  
 n.pops = n.pops),  
 measures = measures,  
 colours = colours,  
 insert = F)[[1]]  
   
 return(beta\_plot)  
 }  
   
 #Make a 4 by 3 grid of plots  
 step\_row <- plot\_grid(vis.steps(0),   
 vis.steps(1),   
 vis.steps(5),   
 vis.steps(50),  
 nrow = 1)  
   
 samples\_row <- plot\_grid(  
 vis.sensitivities(0, "n.samples"),  
 vis.sensitivities(1, "n.samples"),  
 vis.sensitivities(5, "n.samples"),  
 vis.sensitivities(50, "n.samples"),  
 nrow = 1  
 )  
   
 loci\_row <- plot\_grid(  
 vis.sensitivities(0, "n.loci"),  
 vis.sensitivities(1, "n.loci"),  
 vis.sensitivities(5, "n.loci"),  
 vis.sensitivities(50, "n.loci"),  
 nrow = 1  
 )  
   
 pops\_row <- plot\_grid(  
 vis.sensitivities(0, "n.pops"),  
 vis.sensitivities(1, "n.pops"),  
 vis.sensitivities(5, "n.pops"),  
 vis.sensitivities(50, "n.pops"),  
 nrow = 1  
 )  
   
   
 beta\_row <- plot\_grid(vis.betas(0),  
 vis.betas(1),  
 vis.betas(5),  
 vis.betas(50),  
 nrow = 1)  
   
 title\_row <- ggdraw() +  
 draw\_label(paste0("Measure = ", str\_flatten(measures, "\_"),   
 ", p.start = ", p.start,  
 ", p.end = ", p.end),   
 fontface = 'bold', x = 0, hjust = 0) +  
 theme(  
 # add margin on the left of the drawing canvas,  
 # so title is aligned with left edge of first plot  
 plot.margin = margin(0, 0, 0, 7)  
 )  
   
   
 final\_figure <- plot\_grid(title\_row, step\_row, beta\_row,  
 samples\_row, loci\_row, pops\_row,  
 nrow = 6, rel\_heights = c(0.3, 1, 1, 1, 1, 1))  
   
 folder\_name <- ifelse(length(measures) > 1,   
 "Comparison",   
 str\_flatten(measures, "\_"))  
   
 dir.create(paste0("./Outputs/", folder\_name))  
   
 ggsave(final\_figure,   
 filename = paste0("./Outputs/", folder\_name,  
 "/", folder\_name,   
 "\_", p.start, "\_", p.end, ".pdf"),  
 height = 297, width = 210, unit = "mm")  
   
   
 ggsave(final\_figure,   
 filename = paste0("./Outputs/", folder\_name,  
 "/", folder\_name,   
 "\_", p.start, "\_", p.end, ".png"),  
 height = 410 \*0.8, width = 250, unit = "mm")  
   
 return(final\_figure)  
}