```
# Supplementary File 1- R Script for Figure 1, Figure 1- figure supplement 4
#
#
# This is a program to calculate various parameters for E-cadherin quantifications-----
#
#
#
# "rawdata" contains data read from the csv file(provided by user). It should contain the following
columns:
#   1) Genotype
#   2) Cell name
#   3) Mean
#   4) Length
#   5) Boundary -- If you are doing cell wise analysis(put same letter in the whole column in case
analysis is cell wise and by boundaries)
#
#
# "stack.all" contains stack wise calculated data which includes:
#   1) Height at a particular stack
#   2) Normalized Height
#   3) Normalized Ecad
#   4) Absolute Ecad at each stack
#
# "cell.all" contains cell wise data. Which includes the following:
#   1) The total Height of the Cell
#   2) Total Ecad in the cell
#
# "genotype.all" contains  genotype wise data which includes the following:
#   1) Mean apical perimeter
#   2) Median apical perimeter
#   3) Mean Total Ecad per cell
#   4) Median Total Ecad per cell
#   5) Mean Height of the cell
#   6) Median Height of the cell
#   7) First quantile of Height of cell
#   8) Third quantile of height of cell
#
# "locfit.model" contains the locfit data and the height at which the levels were predicted.
#
# "apical.perimeter" contains the apical most stack used for apical perimeter analysis
#
#
#
#

#clears memory--------
rm(list = ls())

```

```
#sets a working directory----------
#This is where all your graphs will be saved

setwd(
  "C:/Users/prata/Desktop/Test for paper/"
)

#libraries required for the program to work.-----------------
# List of libraries required for the program to run
pkg <- c("scales", "locfit","tidyr","purrr","dplyr","ggplot2","broom",
         "data.table", "RCurl","stringr", "grid", "gridExtra", "RColorBrewer")

# Check if the libraries are installed and install libraries not present
new.pkg <- pkg[!(pkg %in% installed.packages())]
if (length(new.pkg)) {
  install.packages(new.pkg)
}
rm("pkg","new.pkg")

#load libraries in the memory
library(scales)
library(locfit)
library(tidyr)
library(purrr)
library(dplyr)
library(ggplot2)
library(data.table)
library(RCurl)
library(stringr)
library(grid)
library(gridExtra)
library(RColorBrewer)

#stops warnings from being displayed
options(warn = -1)



# read raw data------------------
# the file should should have the following components:
# Genotype        Cell name         Area     Mean    Boundary

#Folder and file names where your input file is saved

Folderadd <-
  "C:/Users/prata/Desktop/Test for paper/"
filename <-
  "Figure 1- source data 1"
```

```
format <- ".csv"
fulladdress <- paste(Folderadd, filename, format, sep = "")

rawdata <-
  read.csv(fulladdress,
           header = TRUE,
           sep = ",") %>% na.omit()

# data arranging----------

#To have Genotype and Cell name, Boundary order as in the file (Otherwise it comes alphabetically
which may not be desirable)
rawdata$Genotype <-
  factor(rawdata$Genotype, levels = unique(rawdata$Genotype))
rawdata$Cell.name <-
  factor(rawdata$Cell.name, levels = unique(rawdata$Cell.name))
rawdata$Boundary <-
  factor(rawdata$Boundary, levels = unique(rawdata$Boundary))


#group by genotype and cell name calculate absolute ecad for each stack, and nest the data to to
help calculations on groups
by_cellname <- rawdata %>%
  group_by(Genotype, Cell.name) %>%
  mutate(AbsoluteEcad = Mean * Area) %>%
  nest()


#function to find height
rawheight <- function(df) {
  height <- vector("double", nrow(df))
  for (i in seq_along(df$Mean)) {
    height[i] <- (i - 1) * 0.28 #the height of the z stack
  }
  height
}


#function to normalize height
normheightfun <- function(df) {
  nheight <- vector("double", nrow(df))
  maxh <- nrow(df)
  minh <- 1
  for (i in seq_along(df$Mean)) {
    nheight[i] <- ((i) - minh) / (maxh - minh)
  }
  nheight
}
```

```
145
146  #function to normalize ecad levels
147  normecadfun <- function(df) {
148    necad <- vector("double", nrow(df))
149    maxe <- max(df$Mean)
150    mine <- min(df$Mean)
151    for (i in seq_along(df$Mean)) {
152      necad[i] <- (df$Mean[i] - mine) * 100 / (maxe - mine)
153    }
154    necad
155  }
156
157  #dataframe which now has height per stack, normalized height, normalized ecad per genotype-cell
158  name
159  stack.all <- by_cellname %>%
160    mutate(
161      Stackheight = data %>% map(rawheight),
162      Normalizedheight = data %>% map(normheightfun),
163      NormalizedEcad = data %>% map(normecadfun),
164      Uniquecellname = paste(Genotype, Cell.name)
165    ) %>% unnest()
166
167
168  #to know the total ecad in a cell and height of a cell
169  cell.all <-  stack.all %>%
170    group_by(Genotype, Cell.name) %>%
171    summarize(Total_Ecad = sum(AbsoluteEcad),
172              Height_of_cell = max(Stackheight))
173
174  #to fit linear regression to find the abnormal polarized distribution----------
175
176  #function for linear regression
177
178  #locfit -------------------------------------------
179  # locfit will fit the data cell wise and predict values at given cell heights
180
181  #nest the data in stack.all with normalized height to use in locfit for prediction
182  locfit.input <- stack.all %>%
183    group_by(Genotype, Cell.name) %>%
184    nest()
185
186  # locfit.input$data[[1]]
187
188
189  #cell heights chosen to interpolate ecad levels
190  predictvalues <- c(0.1, 0.3, 0.5, 0.7, 0.9)
191
192  #To calculate the y-axis limits in locfit graphs
```

```
ylimmax <- max(rawdata$Mean) + 250
ylimmin <- min(rawdata$Mean) - 100


#function to fit locfit
locfit_model <- function(df) {
  fit <- locfit(Mean ~ Normalizedheight, data = df)
  plot(
    fit,
    get.data = TRUE,
    ylim = c(ylimmin, ylimmax),
    xlim = c(0, 1),
    xlab = "Normalized Cell Height",
    ylab = "Ecad Level(in AU)"
  )
  par(new = TRUE)   #To superimpose graphs
  predictions <-  predict(fit, predictvalues)

}

#to plot the locfit graphs
#to start a new graph page and close all old ones,
# so that new graphs don't get superimposed on the old ones
plot.new()
dev.off()
plot.new()




#add locfit data to grouped cells and add cell height(on which prediction is made) to each cell
locfit.model <- locfit.input %>%
  mutate(fitvalues = data %>% map(locfit_model)) %>%
  unnest(fitvalues) %>%
  group_by(Genotype, Cell.name) %>%
  mutate(Cell_height = predictvalues)
locfit.model <- locfit.model %>%  group_by(Genotype, Cell.name)

#to save graph for locfit data
dev.copy(
  svg,
  file = paste0(filename, "_locfitgraphs.svg"),
  width = 4,
  height = 4
)

dev.off ()

```

```
241
242  #for creating groups for each cell for ggplot graph of lines ----------------------------------------------
243
244  graph.lines <- stack.all %>% mutate (Uniquecellname = paste(Genotype, Cell.name))
245
246
247  #to find out the apical perimeter of the cell
248
249  apical.perimeter <- graph.lines %>%
250   group_by(Genotype, Cell.name) %>%
251   slice(1) %>% select(Genotype, Cell.name, Length)
252
253  cell.all <- cell.all %>% full_join(apical.perimeter, by = c("Genotype", "Cell.name"))
254
255  cell.all <- rename(cell.all, "Apical Perimeter" = Length)
256
257
258  # to calculate Genotype wise parameters-------------------------
259  genotype.peri <-  apical.perimeter %>%   group_by(Genotype) %>%
260   summarise(Mean_peri = mean(Length),
261             Median_peri = median(Length))
262
263
264  genotype.summary <- cell.all %>% group_by(Genotype) %>%
265   summarise(
266     Mean_TotalEcad = mean(Total_Ecad),
267     Median_TotalEcad = median(Total_Ecad),
268     Mean_Heightofcell = mean(Height_of_cell),
269     Median_Heightofcell = median(Height_of_cell),
270     First_quantile_Height = quantile(Height_of_cell, prob = 0.25),
271     Third_quantile_Height = quantile(Height_of_cell, prob = 0.75),
272     count = n()
273   )
274
275  #final dataframe for all calculated parameters by merging the above two data frames
276  genotype.all <- genotype.peri %>% full_join(genotype.summary, by = "Genotype")
277
278  #writing csv-----------------------------------------------------------
279
280
281  #write csv with height, normalized height,absolute ecad in each stack
282  write.csv(
283   stack.all,
284   file = paste0(filename, "_stackwise_calculations.csv"),
285   row.names = FALSE
286  )
287
288  #write csv with cell level parameters
```

```
write.csv(
  cell.all,
  file = paste0(filename, "_cellwise_calculations.csv"),
  row.names = FALSE
)

#write csv with locfit fitdata
write.csv(locfit.model,
          file = paste0(filename, "_locfitdata.csv"),
          row.names = FALSE)

#write csv with genotype level parameters
write.csv(
  genotype.all,
  file = paste0(filename, "_genotypewise_calculations.csv"),
  row.names = FALSE
)

#graphs------------------------------------------------------------

#to plot boxplots for Total Ecad vs Genotype----------
p <-
  ggplot(data = cell.all, aes(x = Genotype, y = Total_Ecad, fill = Genotype)) +
  theme(
    plot.title = element_text(hjust = 0.5),
    text = element_text(size = 14),
    legend.position = "none",
    axis.text.x = element_text(angle = 45, hjust = 1)
  ) +
  labs(title = "Total E-cadherin", x = "Genotype", y = "E-cadherin Levels(in AU)") +
  scale_y_continuous(labels = comma) +
  theme(
    plot.title = element_text(hjust = 0.5),
    legend.position = "none",
    text = element_text(size = 14)
  ) +
  geom_boxplot(outlier.shape = NA, alpha = 0.7) +
  geom_point(position = position_jitter(width = .1),
             shape = 21,
             alpha = 0.4)#+
# coord_cartesian(ylim = c(0, 800000))
print(p)
dev.copy(
  svg,
  file = paste0(filename, "_Total Ecad.svg"),
  width = 4,
  height = 4
)
```

```
337 dev.off ()
338
339
340
341 #to plot boxplots for Height of cell vs Genotype----------
342 q <-
343  q <-
344  ggplot(data = cell.all, aes(x = Genotype, y = Height_of_cell, fill = Genotype)) +
345  theme(
346   plot.title = element_text(hjust = 0.5),
347   text = element_text(size = 14),
348   legend.position = "none",
349   axis.text.x = element_text(angle = 45, hjust = 1)
350  ) +
351  labs(title = "Height of Cells",
352       x = "Genotype",
353       y = expression(paste("Height(in ", mu, "m")")) +
354  scale_y_continuous(labels = comma) +
355  theme(
356   plot.title = element_text(hjust = 0.5),
357   legend.position = "none",
358   text = element_text(size = 14)
359  ) +
360  geom_boxplot(outlier.shape = NA, alpha = 0.7) +
361  geom_point(
362   position = position_jitterdodge(
363    jitter.height = 0,
364    jitter.width = 0.5,
365    dodge.width = .1
366   ),
367   shape = 21,
368   alpha = 0.4
369  )#+
370 # coord_cartesian(ylim = c(0, 8))
371 print(q)
372 dev.copy(
373  svg,
374  file = paste0(filename, "_totalcellheight.svg"),
375  width = 4,
376  height = 4
377 )
378 dev.off ()
379
380 #to plot boxplots for Height of cell vs locfit data----------
381 r <-
382  ggplot(data = locfit.model,
383       aes(
384        x = factor(Cell_height),
```

```
            y = fitvalues,
            fill = Genotype
          )) +
  theme(plot.title = element_text(hjust = 0.5), text = element_text(size =
                                                  14)) +
  labs(title = "E-cadherin levels at various Cell Heights", x = "Normalized Height", y = "E-cadherin
Levels(in AU)") +
  geom_boxplot(position = position_dodge(width = 0.8), outlier.shape = NA) +
  scale_y_continuous(labels = comma) +
  geom_point(
    alpha = 0.4,
    position = position_jitterdodge(jitter.width = .2, dodge.width = 0.8),
    shape = 21
  )#+
# coord_cartesian(ylim = c(0, 1750))
print(r)
dev.copy(
  svg,
  file = paste0(filename, "_locfit_boxplot.svg"),
  width = 4,
  height = 4
)
dev.off ()

#to plot lineplots for cell height  vs Ecad levels----------
s <-
  ggplot(
    data = graph.lines,
    aes(
      x = Stackheight,
      y = Mean,
      group = Uniquecellname,
      fill = Genotype,
      color = Genotype
    )
  ) +
  theme(plot.title = element_text(hjust = 0.5), text = element_text(size =
                                                  14)) +
  labs(title = "Height vs Ecad Levels",
       x = "Height(in " ~ (mu * m),
       y = "E-cadherin Levels(in AU)") +
  scale_y_continuous(labels = comma) +
  geom_point(position = position_jitter(width = .1),
             shape = 21,
             alpha = 0.4) +
  geom_line(alpha = 0.4)#+
# coord_cartesian(ylim = c(0, 1750))
print(s)
```

```
433  dev.copy(
434    svg,
435    file = paste0(filename, "_Rawheight.svg"),
436    width = 4,
437    height = 4
438  )
439  dev.off ()
440
441  #to plot lineplots for Normalized cell height  vs Ecad levels----------------
442  u <-
443    ggplot(
444      data = graph.lines,
445      aes(
446        x = Normalizedheight,
447        y = Mean,
448        group = Uniquecellname,
449        fill = Genotype,
450        color = Genotype
451      )
452    ) +
453    theme(plot.title = element_text(hjust = 0.5), text = element_text(size =
454                                                                        14)) +
455    labs(title = "Normalized Height vs Ecad Levels", x = "Normalized Height", y = "E-cadherin Levels(in
456  AU)") +
457    geom_point(shape = 21, alpha = 0.4) +
458    scale_y_continuous(labels = comma) +
459    geom_line(alpha = 0.4) +
460    #coord_cartesian(ylim = c(0, 1750))#+
461    facet_grid(facets = . ~ Boundary)
462  print(u)
463
464  dev.copy(
465    svg,
466    file = paste0(filename, "_Normalized_height.svg"),
467    width = 4,
468    height = 4
469  )
470  dev.off ()
471
472
473  #to plot lineplots for Normalized cell height  vs Normalized Ecad levels------------
474  t <-
475    ggplot(
476      data = graph.lines,
477      aes(
478        x = Normalizedheight,
479        y = NormalizedEcad,
480        group = Uniquecellname,
```

```
      fill = Genotype
      # color = Genotype
    )
  ) +
  theme(
    plot.title = element_text(hjust = 0.5),
    text = element_text(size = 14),
    legend.position = "none",
    axis.text.x = element_text(angle = 45, hjust = 1)
  ) +
  scale_y_continuous(labels = comma) +
  labs(title = "E-Cadherin Distribution", x = "Normalized Height", y = "Normalized E-cadherin Levels")
+
  # stat_smooth(aes(
  #   x = Normalizedheight,
  #   y = NormalizedEcad,
  #   group = Genotype,
  #   fill = Genotype,
  #   color = Genotype
  # ),method='lm')+
  geom_bin2d(bins = 15, alpha = 0.3) +
  geom_point(shape = 21, alpha = 0.15) + facet_grid(. ~ Boundary) #+

# stat_density_2d(aes(colour = Genotype), geom="density_2d", alpha= 0.2)
print(t)

dev.copy(
  svg,
  file = paste0(filename, "_Normalized height Normalized Ecad.svg"),
  width = 4,
  height = 4
)
dev.off ()



#to plot apical perimeter vs genotype----------
b <-
  ggplot(data = apical.perimeter, aes(x = Genotype, y = Length, fill = Genotype)) +
  theme(
    plot.title = element_text(hjust = 0.5),
    text = element_text(size = 14),
    legend.position = "none",
    axis.text.x = element_text(angle = 45, hjust = 1)
  ) +
  labs(title = "Apical Perimeter of Cells",
       x = "Genotype",
       y = "Length(in" ~ (mu * m)) +
```

```
  geom_boxplot(outlier.shape = NA, alpha = 0.7) +
  scale_y_continuous(labels = comma) +
  theme(
    plot.title = element_text(hjust = 0.5),
    legend.position = "none",
    text = element_text(size = 14)
  ) +
  geom_point(position = position_jitter(width = .1),
             shape = 21,
             alpha = 0.4)#+
# coord_cartesian(ylim = c(40, 120))
print(b)
dev.copy(
  svg,
  file = paste0(filename, "_apical perimeter.svg"),
  width = 4,
  height = 4
)
dev.off ()



# multiplot graphs-----------------------------------
# To plot multiple graphs in one page

grid.newpage()
pushViewport(viewport(layout = grid.layout(17, 6)))
vplayout <- function(x, y)
  viewport(layout.pos.row = x, layout.pos.col = y)
print(r, vp = vplayout(1:6, 2:6))
print(p, vp = vplayout(7:11, 1:2))
print(q, vp = vplayout(7:11, 3:4))
print(b, vp = vplayout(7:11, 5:6))
print(t, vp = vplayout(12:17, 2:5))
dev.copy(
  svg,
  file = paste0(filename, "multiplot_all.svg"),
  width = 10,
  height = 10
)
dev.off()

```

```
# Supplementary File 2- R Script for Figure 2, 3, Figure 2-figure supplement 3, 4
#
#
# This is a program to calculate various parameters for E-cadherin quantifications-----
#
#
#
# "rawdata" contains data read from the csv file(provided by user). It should contain the following
columns:
#   1) Genotype
#   2) Cell name
#   3) Mean
#   4) Length
#   5) Boundary -- If you are doing cell wise analysis(put same letter in the whole column in case
analysis is cell wise and by boundaries)
#
#
# "stack.all" contains stack wise calculated data which includes:
#   1) Height at a particular stack
#   2) Normalized Height
#   3) Normalized Ecad
#   4) Absolute Ecad at each stack
#
# "cell.all" contains cell wise data. Which includes the following:
#   1) The total Height of the Cell
#   2) Total Ecad in the cell
#
# "genotype.all" contains  genotype wise data which includes the following:
#   1) Mean apical perimeter
#   2) Median apical perimeter
#   3) Mean Total Ecad per cell
#   4) Median Total Ecad per cell
#   5) Mean Height of the cell
#   6) Median Height of the cell
#   7) First quantile of Height of cell
#   8) Third quantile of height of cell
#
# "locfit.model" contains the locfit data and the height at which the levels were predicted.
#
# "apical.perimeter" contains the apical most stack used for apical perimeter analysis
#
# "linear.reg.noise"  contains number of cells/percentage cells showing abnormal polarized
distribution
#
# "linear.reg.whole" contains linear regression for both siblings and mutants
#
# The program also classifies the cells according to the cell height.-- Tall cells, medium cells, short
cells
```

```
#
#
#

#clears memory--------
rm(list = ls())

#sets a working directory-----------
#This is where all your graphs will be saved

setwd(
  "C:/User/Desktop/"
)

#libraries required for the program to work.-----------------
# List of libraries required for the program to run
pkg <- c("scales", "locfit","tidyr","purrr","dplyr","ggplot2","broom",
         "data.table", "RCurl","stringr", "grid", "gridExtra", "RColorBrewer")

# Check if the libraries are installed and install libraries not present
new.pkg <- pkg[!(pkg %in% installed.packages())]
if (length(new.pkg)) {
  install.packages(new.pkg)
}
rm("pkg","new.pkg")

#load libraries in the memory
library(scales)
library(locfit)
library(tidyr)
library(purrr)
library(dplyr)
library(ggplot2)
library(data.table)
library(RCurl)
library(stringr)
library(grid)
library(gridExtra)
library(RColorBrewer)

#stops warnings from being displayed
options(warn = -1)



# read raw data------------------
# the file should should have the following components:
# Genotype        Cell name         Area     Mean    Boundary
```

```
667
668  #Folder and file names where your input file is saved
669
670  Folderadd <-
671    "C:/User/Desktop/"
672  filename <-
673    "aPKC mut"
674  format <- ".csv"
675  fulladdress <- paste(Folderadd, filename, format, sep = "")
676
677  rawdata <-
678    read.csv(fulladdress,
679             header = TRUE,
680             sep = ",") %>% na.omit()
681
682  # data arranging----------
683
684  #To have Genotype and Cell name, Boundary order as in the file (Otherwise it comes alphabetically
685  which may not be desirable)
686  rawdata$Genotype <-
687    factor(rawdata$Genotype, levels = unique(rawdata$Genotype))
688  rawdata$Cell.name <-
689    factor(rawdata$Cell.name, levels = unique(rawdata$Cell.name))
690  rawdata$Boundary <-
691    factor(rawdata$Boundary, levels = unique(rawdata$Boundary))
692
693
694  #group by genotype and cell name calculate absolute ecad for each stack, and nest the data to to
695  help calculations on groups
696  by_cellname <- rawdata %>%
697    group_by(Genotype, Cell.name) %>%
698    mutate(AbsoluteEcad = Mean * Area) %>%
699    nest()
700
701
702  #function to find height
703  rawheight <- function(df) {
704    height <- vector("double", nrow(df))
705    for (i in seq_along(df$Mean)) {
706      height[i] <- (i - 1) * 0.28 #the height of the z stack
707    }
708    height
709  }
710
711
712  #function to normalize height
713  normheightfun <- function(df) {
714    nheight <- vector("double", nrow(df))
```

```
  maxh <- nrow(df)
  minh <- 1
  for (i in seq_along(df$Mean)) {
    nheight[i] <- ((i) - minh) / (maxh - minh)
  }
  nheight
}

#function to normalize ecad levels
normecadfun <- function(df) {
  necad <- vector("double", nrow(df))
  maxe <- max(df$Mean)
  mine <- min(df$Mean)
  for (i in seq_along(df$Mean)) {
    necad[i] <- (df$Mean[i] - mine) * 100 / (maxe - mine)
  }
  necad
}

#dataframe which now has height per stack, normalized height, normalized ecad per genotype-cell
name
stack.all <- by_cellname %>%
  mutate(
    Stackheight = data %>% map(rawheight),
    Normalizedheight = data %>% map(normheightfun),
    NormalizedEcad = data %>% map(normecadfun),
    Uniquecellname = paste(Genotype, Cell.name)
  ) %>% unnest()


#to know the total ecad in a cell and height of a cell
cell.all <-  stack.all %>%
  group_by(Genotype, Cell.name) %>%
  summarize(Total_Ecad = sum(AbsoluteEcad),
            Height_of_cell = max(Stackheight))

#to fit linear regression to find the abnormal polarized distribution----------

#function for linear regression

linearfit.noise <- function(df) {

  # To give specific colors in linear regressions, a column called colour1 can be added to input file.
  # all "colourcodes" lines can then be uncommented.

  # colourcodes <- df %>% distinct(colour1)
  # colourcodes <- as.character(colourcodes[['colour1']])
  # print(colourcodes)
```

```
763
764
765     # select sibling data for linear regression
766
767     df.lm <- df %>% filter(str_detect(df$Genotype, "sib"))
768
769     # fit linear regression on the sibling data
770     fit <- lm(NormalizedEcad ~ Normalizedheight, data = df.lm)
771
772     # find the upper and lower prediction limits at 99% prediction level
773     pred <-
774       as.data.frame(predict(fit, df, level = 0.99, interval = "prediction"))
775
776     # check if the data is within the prediction limits
777     # TRUE= point is within the predction limit
778     # FALSE = point lies outside the prediction limit
779
780     dat <-
781       data.frame(df,
782                  Distribution = df$NormalizedEcad <= pred$upr &
783                    df$NormalizedEcad >= pred$lwr)
784
785     # Replace TRUE with normal and FALSE with Abnormal
786
787     dat <- dat %>%  mutate(Distribution=replace(Distribution, Distribution=="TRUE", "Normal"))
788     dat <- dat %>%  mutate(Distribution=replace(Distribution, Distribution=="FALSE", "Abnormal"))
789
790     # plot graph for the linear regression fit data showing the prediction limits according to the siblings
791
792     plot.lm <-
793       ggplot(data = dat, aes(
794         x = dat$Normalizedheight,
795         y = dat$NormalizedEcad
796       )) +
797       # scale_colour_manual(values = colourcodes) +
798       coord_cartesian(ylim = c(0, 100)) +
799       theme(
800         plot.title = element_text(hjust = 0.5),
801         text = element_text(size = 14),
802         axis.text.x = element_text(angle = 45, hjust = 1)
803       ) +
804       labs(title = "E-Cadherin Distribution", x = "Normalized Height", y = "Normalized E-cadherin Levels")
805     +
806       stat_smooth(method = 'lm',
807                   # color = colourcodes[1],
808                   se = FALSE
809       )  +
810       geom_ribbon(
```

```
811         data = pred,
812         aes(ymin = lwr, ymax = upr),
813         # fill = colourcodes[1],
814         alpha = 0.2
815       ) +
816       geom_point(aes(colour = Genotype), alpha = 0.4)
817     print(plot.lm)
818
819
820     # save the plot
821     dev.copy(svg,
822              file = paste0(
823                filename,
824                "_",
825                df$Genotype[1],
826                "_linear_regrs_sibling_pred_interval.svg"
827              ))
828     dev.off()
829
830     # Calculate abnormal vs normal distribution numbers and percentage
831     dat.summary <-
832       dat %>% group_by(Genotype, Distribution) %>% distinct(Uniquecellname)  %>%
833   arrange(Distribution)
834     dat.summary <-
835       dat.summary %>% ungroup() %>% group_by(Genotype, Uniquecellname) %>% slice(1)
836     dat.summary <-
837       dat.summary %>% group_by(Genotype, Distribution) %>% summarize(Count =
838                                                                        n())
839     dat.summary2 <-
840       dat.summary %>% ungroup() %>% group_by(Genotype) %>% summarize(total =
841                                                                        sum(Count))
842     dat.summary3 <- full_join(dat.summary, dat.summary2)
843     dat.summary3 <-
844       dat.summary3 %>% mutate(Percentage = ((Count / total) * 100))
845     dat.summary3
846   }
847   # show_col(hue_pal()(4)) #to show color codes
848
849   # Group according the boundary types to get values for boudnaries only (eg. Clone-Clone)
850   linear.reg.input <-  stack.all %>%
851     group_by(Boundary) %>%
852     nest()
853
854   # to get table for all normal vs abnormal distribution according to boundary type
855   linear.reg.noise <- linear.reg.input %>%
856     mutate(lmfit = data %>% map(linearfit.noise)) %>% select(-data)  %>% unnest(lmfit)
857
858   write.csv(
```

```
  linear.reg.noise,
  file = paste0(filename, "_abnormal_by_linear_regression.csv"),
  row.names = FALSE
 )


 # function to check if linear regressions are close by in both mutants and siblings

 linearfit.whole <- function(df) {
  # colourcodes <- df %>% distinct(colour1)
  # colourcodes <- as.character(colourcodes[['colour1']])

  # plot regressions for bith siblings and mutants
  plot.lm <-
   ggplot(data = df,
          aes(
           x = df$Normalizedheight,
           y = df$NormalizedEcad,
           colour = Genotype
          )) +
   # scale_colour_manual(values = colourcodes) +
   coord_cartesian(ylim = c(0, 100)) +
   theme(
    plot.title = element_text(hjust = 0.5),
    text = element_text(size = 14),
    axis.text.x = element_text(angle = 45, hjust = 1)
   ) +
   labs(title = "E-Cadherin Distribution", x = "Normalized Height", y = "Normalized E-cadherin Levels")
 +
   stat_smooth(method = 'lm', se = FALSE)  +
   geom_point(aes(), alpha = 0.4)
  print(plot.lm)

  # save the plot
  dev.copy(svg, file = paste0(filename, df$Genotype[1], "_linear_regrs_whole.svg"))
  dev.off()

  # fit linear regression in both sibling and mutant

  fit <- lm(NormalizedEcad ~ Normalizedheight, data = df)
  radjsq <- summary(fit)$adj.r.squared
  rsq <- summary(fit)$r.squared
  slope <- coef(summary(fit))["Normalizedheight", "Estimate"]
  intercept <- coef(summary(fit))["(Intercept)", "Estimate"]
  tablefit <- data.frame(radjsq, rsq, intercept, slope)
  tablefit <-
   tablefit %>% rename("Adjusted R Squared" = radjsq, "R Squared" = rsq)
  tablefit
```

```
}

# Make table according with linear regressions fit according to boundary types

linear.reg.whole <- linear.reg.input %>%
  mutate(lmfit = data %>% map(linearfit.whole)) %>% select(-data)  %>% unnest(lmfit)

#write csv for linear fit for Normalized ecad and normalized height----------

write.csv(
  linear.reg.whole,
  file = paste0(filename, "_genotypewise_linearfit_coefficients.csv"),
  row.names = FALSE
)


#locfit ------------------------------------------
# locfit will fit the data cell wise and predict values at given cell heights

#nest the data in stack.all with normalized height to use in locfit for prediction
locfit.input <- stack.all %>%
  group_by(Genotype, Cell.name) %>%
  nest()

# locfit.input$data[[1]]


#cell heights chosen to interpolate ecad levels
predictvalues <- c(0.1, 0.3, 0.5, 0.7, 0.9)

#To calculate the y-axis limits in locfit graphs
ylimmax <- max(rawdata$Mean) + 250
ylimmin <- min(rawdata$Mean) - 100


#function to fit locfit
locfit_model <- function(df) {
  fit <- locfit(Mean ~ Normalizedheight, data = df)
  plot(
    fit,
    get.data = TRUE,
    ylim = c(ylimmin, ylimmax),
    xlim = c(0, 1),
    xlab = "Normalized Cell Height",
    ylab = "Ecad Level(in AU)"
  )
  par(new = TRUE)   #To superimpose graphs
  predictions <-  predict(fit, predictvalues)
```

```
955
956     }
957
958     #to plot the locfit graphs
959     #to start a new graph page and close all old ones,
960     # so that new graphs don't get superimposed on the old ones
961     plot.new()
962     dev.off()
963     plot.new()
964
965
966
967
968     #add locfit data to grouped cells and add cell height(on which prediction is made) to each cell
969     locfit.model <- locfit.input %>%
970       mutate(fitvalues = data %>% map(locfit_model)) %>%
971       unnest(fitvalues) %>%
972       group_by(Genotype, Cell.name) %>%
973       mutate(Cell_height = predictvalues)
974     locfit.model <- locfit.model %>%  group_by(Genotype, Cell.name)
975
976     #to save graph for locfit data
977     dev.copy(
978       svg,
979       file = paste0(filename, "_locfitgraphs.svg"),
980       width = 4,
981       height = 4
982     )
983
984     dev.off ()
985
986
987     #for creating groups for each cell for ggplot graph of lines ----------------------------------------------
988
989     graph.lines <- stack.all %>% mutate (Uniquecellname = paste(Genotype, Cell.name))
990
991
992     #to find out the apical perimeter of the cell
993
994     apical.perimeter <- graph.lines %>%
995       group_by(Genotype, Cell.name) %>%
996       slice(1) %>% select(Genotype, Cell.name, Length)
997
998     cell.all <- cell.all %>% full_join(apical.perimeter, by = c("Genotype", "Cell.name"))
999
1000    cell.all <- rename(cell.all, "Apical Perimeter" = Length)
1001
1002
```

```
# to calculate Genotype wise parameters-------------------------
genotype.peri <-  apical.perimeter %>%  group_by(Genotype) %>%
 summarise(Mean_peri = mean(Length),
           Median_peri = median(Length))


genotype.summary <- cell.all %>% group_by(Genotype) %>%
 summarise(
  Mean_TotalEcad = mean(Total_Ecad),
  Median_TotalEcad = median(Total_Ecad),
  Mean_Heightofcell = mean(Height_of_cell),
  Median_Heightofcell = median(Height_of_cell),
  First_quantile_Height = quantile(Height_of_cell, prob = 0.25),
  Third_quantile_Height = quantile(Height_of_cell, prob = 0.75),
  count = n()
 )

#final dataframe for all calculated parameters by merging the above two data frames
genotype.all <- genotype.peri %>% full_join(genotype.summary, by = "Genotype")

#filter out tall cells or short cells-------------------------

# This part of the program will not run if the following names are in the name of the file
# (so that the same program can be run on different hieghted cells without really re- filtering the
cells according to the height again)
namecontains <- c("tall_cells", "short_cells","median_cells")

if (!any(str_detect(filename, namecontains))) {


 #filter tall cells
 sibthirdquantile <- genotype.summary %>% filter(str_detect(genotype.summary$Genotype, "sib"))
%>% select(Third_quantile_Height)
 sibfirstquantile <- genotype.summary %>% filter(str_detect(genotype.summary$Genotype, "sib"))
%>% select(First_quantile_Height)

 tall_cells_cellwise <- cell.all %>%
  full_join(genotype.all, by = "Genotype") %>% filter(Height_of_cell >
sibthirdquantile$Third_quantile_Height) %>% select(Genotype, Cell.name, Total_Ecad,
Height_of_cell)

 tall_cells_all <-  stack.all %>%
  semi_join(tall_cells_cellwise, by = c("Genotype", "Cell.name"))


 #write csv with tall cells filtered out
 write.csv(
  tall_cells_cellwise,
```

```
    file = paste0(filename, "_tall_cells_cellwise.csv"),
    row.names = FALSE
  )
  write.csv(
    tall_cells_all,
    file = paste0(filename, "_tall_cells_all.csv"),
    row.names = FALSE
  )

  #filter short cells
  short_cells_cellwise <- cell.all %>%
    full_join(genotype.all, by = "Genotype") %>% filter(Height_of_cell <
sibfirstquantile$First_quantile_Height) %>% select(Genotype, Cell.name, Total_Ecad, Height_of_cell)

  short_cells_all <-  stack.all %>%
    semi_join(short_cells_cellwise, by = c("Genotype", "Cell.name"))
  #write csv with short cells filtered out
  write.csv(
    short_cells_cellwise,
    file = paste0(filename, "_short_cells_cellwise.csv"),
    row.names = FALSE
  )
  write.csv(
    short_cells_all,
    file = paste0(filename, "_short_cells_all.csv"),
    row.names = FALSE
  )


  #filter median heighted cells
  median_cells_cellwise <- cell.all %>%
    full_join(genotype.all, by = "Genotype") %>% filter(Height_of_cell >=
sibfirstquantile$First_quantile_Height & Height_of_cell <= sibthirdquantile$Third_quantile_Height)
%>% select(Genotype, Cell.name, Total_Ecad, Height_of_cell)

  median_cells_all <-  stack.all %>%
    semi_join(median_cells_cellwise, by = c("Genotype", "Cell.name"))

  #write csv with short cells filtered out
  write.csv(
    median_cells_cellwise,
    file = paste0(filename, "_median_cells_cellwise.csv"),
    row.names = FALSE
  )
  write.csv(
    median_cells_all,
    file = paste0(filename, "_median_cells_all.csv"),
    row.names = FALSE
```

```
  )

}

#writing csv-------------------------------------------------------


#write csv with height, normalized height,absolute ecad in each stack
write.csv(
  stack.all,
  file = paste0(filename, "_stackwise_calculations.csv"),
  row.names = FALSE
)

#write csv with cell level parameters
write.csv(
  cell.all,
  file = paste0(filename, "_cellwise_calculations.csv"),
  row.names = FALSE
)

#write csv with locfit fitdata
write.csv(locfit.model,
          file = paste0(filename, "_locfitdata.csv"),
          row.names = FALSE)

#write csv with genotype level parameters
write.csv(
  genotype.all,
  file = paste0(filename, "_genotypewise_calculations.csv"),
  row.names = FALSE
)

#graphs-----------------------------------------------------------

#to plot boxplots for Total Ecad vs Genotype-------------
p <-
  ggplot(data = cell.all, aes(x = Genotype, y = Total_Ecad, fill = Genotype)) +
  theme(
    plot.title = element_text(hjust = 0.5),
    text = element_text(size = 14),
    legend.position = "none",
    axis.text.x = element_text(angle = 45, hjust = 1)
  ) +
  labs(title = "Total E-cadherin", x = "Genotype", y = "E-cadherin Levels(in AU)") +
  scale_y_continuous(labels = comma) +
  theme(
    plot.title = element_text(hjust = 0.5),
```

```
    legend.position = "none",
    text = element_text(size = 14)
  ) +
  geom_boxplot(outlier.shape = NA, alpha = 0.7) +
  geom_point(position = position_jitter(width = .1),
             shape = 21,
             alpha = 0.4)#+
# coord_cartesian(ylim = c(0, 800000))
print(p)
dev.copy(
  svg,
  file = paste0(filename, "_Total Ecad.svg"),
  width = 4,
  height = 4
)
dev.off ()



#to plot boxplots for Height of cell vs Genotype----------
q <-
  q <-
  ggplot(data = cell.all, aes(x = Genotype, y = Height_of_cell, fill = Genotype)) +
  theme(
    plot.title = element_text(hjust = 0.5),
    text = element_text(size = 14),
    legend.position = "none",
    axis.text.x = element_text(angle = 45, hjust = 1)
  ) +
  labs(title = "Height of Cells",
       x = "Genotype",
       y = expression(paste("Height(in ", mu, "m)"))) +
  scale_y_continuous(labels = comma) +
  theme(
    plot.title = element_text(hjust = 0.5),
    legend.position = "none",
    text = element_text(size = 14)
  ) +
  geom_boxplot(outlier.shape = NA, alpha = 0.7) +
  geom_point(
    position = position_jitterdodge(
      jitter.height = 0,
      jitter.width = 0.5,
      dodge.width = .1
    ),
    shape = 21,
    alpha = 0.4
  )#+
```

```
# coord_cartesian(ylim = c(0, 8))
print(q)
dev.copy(
  svg,
  file = paste0(filename, "_totalcellheight.svg"),
  width = 4,
  height = 4
)
dev.off ()

#to plot boxplots for Height of cell vs locfit data----------
r <-
  ggplot(data = locfit.model,
         aes(
           x = factor(Cell_height),
           y = fitvalues,
           fill = Genotype
         )) +
  theme(plot.title = element_text(hjust = 0.5), text = element_text(size =
                                                                      14)) +
  labs(title = "E-cadherin levels at various Cell Heights", x = "Normalized Height", y = "E-cadherin
Levels(in AU)") +
  geom_boxplot(position = position_dodge(width = 0.8), outlier.shape = NA) +
  scale_y_continuous(labels = comma) +
  geom_point(
    alpha = 0.4,
    position = position_jitterdodge(jitter.width = .2, dodge.width = 0.8),
    shape = 21
  )#+
# coord_cartesian(ylim = c(0, 1750))
print(r)
dev.copy(
  svg,
  file = paste0(filename, "_locfit_boxplot.svg"),
  width = 4,
  height = 4
)
dev.off ()

#to plot lineplots for cell height  vs Ecad levels----------
s <-
  ggplot(
    data = graph.lines,
    aes(
      x = Stackheight,
      y = Mean,
      group = Uniquecellname,
      fill = Genotype,
```

```
      color = Genotype
    )
  ) +
  theme(plot.title = element_text(hjust = 0.5), text = element_text(size =
                                                    14)) +
  labs(title = "Height vs Ecad Levels",
       x = "Height(in " ~ (mu * m),
       y = "E-cadherin Levels(in AU)") +
  scale_y_continuous(labels = comma) +
  geom_point(position = position_jitter(width = .1),
             shape = 21,
             alpha = 0.4) +
  geom_line(alpha = 0.4)#+
# coord_cartesian(ylim = c(0, 1750))
print(s)
dev.copy(
  svg,
  file = paste0(filename, "_Rawheight.svg"),
  width = 4,
  height = 4
)
dev.off ()

#to plot lineplots for Normalized cell height  vs Ecad levels----------------
u <-
  ggplot(
    data = graph.lines,
    aes(
      x = Normalizedheight,
      y = Mean,
      group = Uniquecellname,
      fill = Genotype,
      color = Genotype
    )
  ) +
  theme(plot.title = element_text(hjust = 0.5), text = element_text(size =
                                                    14)) +
  labs(title = "Normalized Height vs Ecad Levels", x = "Normalized Height", y = "E-cadherin Levels(in
AU)") +
  geom_point(shape = 21, alpha = 0.4) +
  scale_y_continuous(labels = comma) +
  geom_line(alpha = 0.4) +
  #coord_cartesian(ylim = c(0, 1750))#+
  facet_grid(facets = . ~ Boundary)
print(u)

dev.copy(
  svg,
```

```
  file = paste0(filename, "_Normalized_height.svg"),
  width = 4,
  height = 4
 )
 dev.off ()


 #to plot lineplots for Normalized cell height  vs Normalized Ecad levels----------
 t <-
  ggplot(
   data = graph.lines,
   aes(
    x = Normalizedheight,
    y = NormalizedEcad,
    group = Uniquecellname,
    fill = Genotype
    # color = Genotype
   )
  ) +
  theme(
   plot.title = element_text(hjust = 0.5),
   text = element_text(size = 14),
   legend.position = "none",
   axis.text.x = element_text(angle = 45, hjust = 1)
  ) +
  scale_y_continuous(labels = comma) +
  labs(title = "E-Cadherin Distribution", x = "Normalized Height", y = "Normalized E-cadherin Levels")
 +
  # stat_smooth(aes(
  #   x = Normalizedheight,
  #   y = NormalizedEcad,
  #   group = Genotype,
  #   fill = Genotype,
  #   color = Genotype
  # ),method='lm')+
  geom_bin2d(bins = 15, alpha = 0.3) +
  geom_point(shape = 21, alpha = 0.15) + facet_grid(. ~ Boundary) #+

 # stat_density_2d(aes(colour = Genotype), geom="density_2d", alpha= 0.2)
 print(t)

 dev.copy(
  svg,
  file = paste0(filename, "_Normalized height Normalized Ecad.svg"),
  width = 4,
  height = 4
 )
 dev.off ()
```

```
1339
1340
1341    #to plot Percent Abnormal Distribution----------
1342    z <-
1343     ggplot(linear.reg.noise,
1344            aes(x = Genotype, y = Percentage, fill = Distribution)) +
1345     geom_bar(stat = "identity", position = "stack") + scale_fill_brewer(palette = "Dark2") +
1346     scale_y_continuous(labels = comma) +
1347     theme(
1348       plot.title = element_text(hjust = 0.5),
1349       text = element_text(size = 14),
1350       axis.text.x = element_text(angle = 45, hjust = 1)
1351     ) +
1352     labs(title = "Percent Boundaries showing \nAbnormal Distribution", x = "Genotype", y =
1353    "Percentage")
1354
1355
1356    print(z)
1357    dev.copy(
1358     svg,
1359     file = paste0(filename, "_abnormal distribution.svg"),
1360     width = 4,
1361     height = 4
1362    )
1363    dev.off ()
1364
1365    #to plot apical perimeter vs genotype----------
1366    b <-
1367     ggplot(data = apical.perimeter, aes(x = Genotype, y = Length, fill = Genotype)) +
1368     theme(
1369       plot.title = element_text(hjust = 0.5),
1370       text = element_text(size = 14),
1371       legend.position = "none",
1372       axis.text.x = element_text(angle = 45, hjust = 1)
1373     ) +
1374     labs(title = "Apical Perimeter of Cells",
1375          x = "Genotype",
1376          y = "Length(in" ~ (mu * m)) +
1377     geom_boxplot(outlier.shape = NA, alpha = 0.7) +
1378     scale_y_continuous(labels = comma) +
1379     theme(
1380       plot.title = element_text(hjust = 0.5),
1381       legend.position = "none",
1382       text = element_text(size = 14)
1383     ) +
1384     geom_point(position = position_jitter(width = .1),
1385                shape = 21,
1386                alpha = 0.4)#+
```

```
# coord_cartesian(ylim = c(40, 120))
print(b)
dev.copy(
  svg,
  file = paste0(filename, "_apical perimeter.svg"),
  width = 4,
  height = 4
)
dev.off ()



# multiplot graphs-----------------------------------
# To plot multiple graphs in one page

grid.newpage()
pushViewport(viewport(layout = grid.layout(17, 6)))
vplayout <- function(x, y)
  viewport(layout.pos.row = x, layout.pos.col = y)
print(r, vp = vplayout(1:6, 2:6))
print(p, vp = vplayout(7:11, 1:2))
print(q, vp = vplayout(7:11, 3:4))
print(b, vp = vplayout(7:11, 5:6))
print(t, vp = vplayout(12:17, 1:4))
print(z, vp = vplayout(12:17, 5:6))
dev.copy(
  svg,
  file = paste0(filename, "multiplot_all.svg"),
  width = 10,
  height = 10
)
dev.off()

```

```
# Supplementary File 2- R Script for Figure 4, 5, Figure 4-figure supplement 2
#
#
# This is a program to calculate various parameters for E-cadherin quantifications-----
#
#
#
# "rawdata" contains data read from the csv file(provided by user). It should contain the following
columns:
#   1) Genotype
#   2) Cell name
#   3) Mean
#   4) Length
#   5) Boundary -- If you are doing cell wise analysis(put same letter in the whole column in case
analysis is cell wise and by boundaries)
#
#
# "stack.all" contains stack wise calculated data which includes:
#   1) Height at a particular stack
#   2) Normalized Height
#   3) Normalized Ecad
#   4) Absolute Ecad at each stack

#clears memory--------
rm(list = ls())

#sets a working directory-----------
#This is where all your graphs will be saved

setwd(
  "D:/Prateek/TIFR/MS Lab/Ecad Gradient/locfit analysis/R files version control/Ecad_project/"
)

#libraries required for the program to work.-----------------
# List of libraries required for the program to run
pkg <- c("scales", "locfit","tidyr","purrr","dplyr","ggplot2","broom",
         "data.table", "RCurl","stringr", "grid", "gridExtra", "RColorBrewer")

# Check if the libraries are installed and install libraries not present
new.pkg <- pkg[!(pkg %in% installed.packages())]
if (length(new.pkg)) {
  install.packages(new.pkg)
}
rm("pkg","new.pkg")

#load libraries in the memory
library(scales)
library(locfit)
```

```
library(tidyr)
library(purrr)
library(dplyr)
library(ggplot2)
library(data.table)
library(RCurl)
library(stringr)
library(grid)
library(gridExtra)
library(RColorBrewer)

#stops warnings from being displayed
options(warn = -1)



# read raw data------------------
# the file should should have the following components:
# Genotype        Cell name         Area     Mean    Boundary

#Folder and file names where your input file is saved

Folderadd <-
  "D:/Prateek/TIFR/MS Lab/Ecad Gradient/locfit analysis/R files version control/Ecad_project/"
filename <-
  "Autonomous peri Lgl 05052019 14122018_outliers removed"
format <- ".csv"
fulladdress <- paste(Folderadd, filename, format, sep = "")

rawdata <-
  read.csv(fulladdress,
           header = TRUE,
           sep = ",") %>% na.omit()

# data arranging--------------------

#To have Genotype and Cell name, Boundary order as in the file (Otherwise it comes alphabetically
which may not be desirable)
rawdata$Genotype <-
  factor(rawdata$Genotype, levels = unique(rawdata$Genotype))
rawdata$Cell.name <-
  factor(rawdata$Cell.name, levels = unique(rawdata$Cell.name))
rawdata$Boundary <-
  factor(rawdata$Boundary, levels = unique(rawdata$Boundary))



#group by genotype and cell name calculate absolute ecad for each stack, and nest the data to to
help calculations on groups
```

```
by_cellname <- rawdata %>%
  group_by(Genotype, Cell.name) %>%
  mutate(AbsoluteEcad = Mean * Area) %>%
  nest()


#function to find height
rawheight <- function(df) {
  height <- vector("double", nrow(df))
  for (i in seq_along(df$Mean)) {
    height[i] <- (i - 1) * 0.28 #the height of the z slice
  }
  height
}


#function to normalize height
normheightfun <- function(df) {
  nheight <- vector("double", nrow(df))
  maxh <- nrow(df)
  minh <- 1
  for (i in seq_along(df$Mean)) {
    nheight[i] <- ((i) - minh) / (maxh - minh)
  }
  nheight
}

#function to normalize ecad levels
normecadfun <- function(df) {
  necad <- vector("double", nrow(df))
  maxe <- max(df$Mean)
  mine <- min(df$Mean)
  for (i in seq_along(df$Mean)) {
    necad[i] <- (df$Mean[i] - mine) * 100 / (maxe - mine)
  }
  necad
}

#dataframe which now has height per stack, normalized height, normalized ecad per genotype-cell
name
stack.all <- by_cellname %>%
  mutate(
    Stackheight = data %>% map(rawheight),
    Normalizedheight = data %>% map(normheightfun),
    NormalizedEcad = data %>% map(normecadfun),
    Uniquecellname = paste(Genotype, Cell.name)
  ) %>% unnest()

```

```
1564
1565  #locfit ----------------------------------------
1566  # locfit will fit the data cell wise and predict values at given cell heights
1567
1568  #nest the data in stack.all with normalized height to use in locfit for prediction
1569  locfit.input <- stack.all %>%
1570    group_by(Genotype, Cell.name) %>%
1571    nest()
1572
1573  # locfit.input$data[[1]]
1574
1575
1576  #cell heights chosen to interpolate ecad levels
1577  predictvalues <- c(0.1, 0.3, 0.5, 0.7, 0.9)
1578
1579  #To calculate the y-axis limits in locfit graphs
1580  ylimmax <- max(rawdata$Mean) + 250
1581  ylimmin <- min(rawdata$Mean) - 100
1582
1583
1584  #function to fit locfit
1585  locfit_model <- function(df) {
1586    fit <- locfit(Mean ~ Normalizedheight, data = df)
1587    plot(
1588      fit,
1589      get.data = TRUE,
1590      ylim = c(ylimmin, ylimmax),
1591      xlim = c(0, 1),
1592      xlab = "Normalized Cell Height",
1593      ylab = "Ecad Level(in AU)"
1594    )
1595    par(new = TRUE)   #To superimpose graphs
1596    predictions <-  predict(fit, predictvalues)
1597
1598  }
1599
1600  #to plot the locfit graphs
1601  #to start a new graph page and close all old ones,
1602  # so that new graphs don't get superimposed on the old ones
1603  plot.new()
1604  dev.off()
1605  plot.new()
1606
1607
1608
1609
1610  #add locfit data to grouped cells and add cell height(on which prediction is made) to each cell
1611  locfit.model <- locfit.input %>%
```

```
  mutate(fitvalues = data %>% map(locfit_model)) %>%
  unnest(fitvalues) %>%
  group_by(Genotype, Cell.name) %>%
  mutate(Cell_height = predictvalues)
locfit.model <- locfit.model %>%  group_by(Genotype, Cell.name)

#to save graph for locfit data
dev.copy(
  svg,
  file = paste0(filename, "_locfitgraphs.svg"),
  width = 4,
  height = 4
)

dev.off ()


#for creating groups for each cell for ggplot graph of lines ----------------------------------------------------

graph.lines <- stack.all %>% mutate (Uniquecellname = paste(Genotype, Cell.name))




#writing csv-----------------------------------------------------------


#write csv with height, normalized height,absolute ecad in each stack
write.csv(
  stack.all,
  file = paste0(filename, "_stackwise_calculations.csv"),
  row.names = FALSE
)


#write csv with locfit fitdata
write.csv(locfit.model,
          file = paste0(filename, "_locfitdata.csv"),
          row.names = FALSE)

#graphs----------------------------------------------------------------

#to plot boxplots for Height of cell vs locfit data------------------
r <-
  ggplot(data = locfit.model,
         aes(
           x = factor(Cell_height),
           y = fitvalues,
```

```
            fill = Genotype
          )) +
  theme(plot.title = element_text(hjust = 0.5), text = element_text(size =
                                                    14)) +
  labs(title = "E-cadherin levels at various Cell Heights", x = "Normalized Height", y = "E-cadherin
Levels(in AU)") +
  geom_boxplot(position = position_dodge(width = 0.8), outlier.shape = NA) +
  scale_y_continuous(labels = comma) +
  geom_point(
    alpha = 0.4,
    position = position_jitterdodge(jitter.width = .2, dodge.width = 0.8),
    shape = 21
  )#+
# coord_cartesian(ylim = c(0, 1750))
print(r)
dev.copy(
  svg,
  file = paste0(filename, "_locfit_boxplot.svg"),
  width = 4,
  height = 4
)
dev.off ()

#to plot lineplots for cell height  vs Ecad levels----------
s <-
  ggplot(
    data = graph.lines,
    aes(
      x = Stackheight,
      y = Mean,
      group = Uniquecellname,
      fill = Genotype,
      color = Genotype
    )
  ) +
  theme(plot.title = element_text(hjust = 0.5), text = element_text(size =
                                                    14)) +
  labs(title = "Height vs Ecad Levels",
       x = "Height(in " ~ (mu * m),
       y = "E-cadherin Levels(in AU)") +
  scale_y_continuous(labels = comma) +
  geom_point(position = position_jitter(width = .1),
             shape = 21,
             alpha = 0.4) +
  geom_line(alpha = 0.4)#+
# coord_cartesian(ylim = c(0, 1750))
print(s)
dev.copy(
```

```
  svg,
  file = paste0(filename, "_Rawheight.svg"),
  width = 4,
  height = 4
)
dev.off ()

#to plot lineplots for Normalized cell height  vs Ecad levels----------
u <-
  ggplot(
    data = graph.lines,
    aes(
      x = Normalizedheight,
      y = Mean,
      group = Uniquecellname,
      fill = Genotype,
      color = Genotype
    )
  ) +
  theme(plot.title = element_text(hjust = 0.5), text = element_text(size =
                                                      14)) +
  labs(title = "Normalized Height vs Ecad Levels", x = "Normalized Height", y = "E-cadherin Levels(in
AU)") +
  geom_point(shape = 21, alpha = 0.4) +
  scale_y_continuous(labels = comma) +
  geom_line(alpha = 0.4) +
  #coord_cartesian(ylim = c(0, 1750))#+
  facet_grid(facets = . ~ Boundary)
print(u)

dev.copy(
  svg,
  file = paste0(filename, "_Normalized_height.svg"),
  width = 4,
  height = 4
)
dev.off ()


# multiplot graphs-----------------------------------
# To plot multiple graphs in one page

grid.newpage()
pushViewport(viewport(layout = grid.layout(17, 6)))
vplayout <- function(x, y)
  viewport(layout.pos.row = x, layout.pos.col = y)
print(r, vp = vplayout(1:6, 2:6))
# print(p, vp = vplayout(7:11, 1:2))
```

```
1756    # print(q, vp = vplayout(7:11, 3:4))
1757    # print(b, vp = vplayout(7:11, 5:6))
1758    # print(t, vp = vplayout(12:17, 1:4))
1759    # print(z, vp = vplayout(12:17, 5:6))
1760    dev.copy(
1761      svg,
1762      file = paste0(filename, "multiplot_all.svg"),
1763      width = 10,
1764      height = 10
1765    )
1766    dev.off()
1767
```