initial_model2

2023-07-10

Null simulation data generating

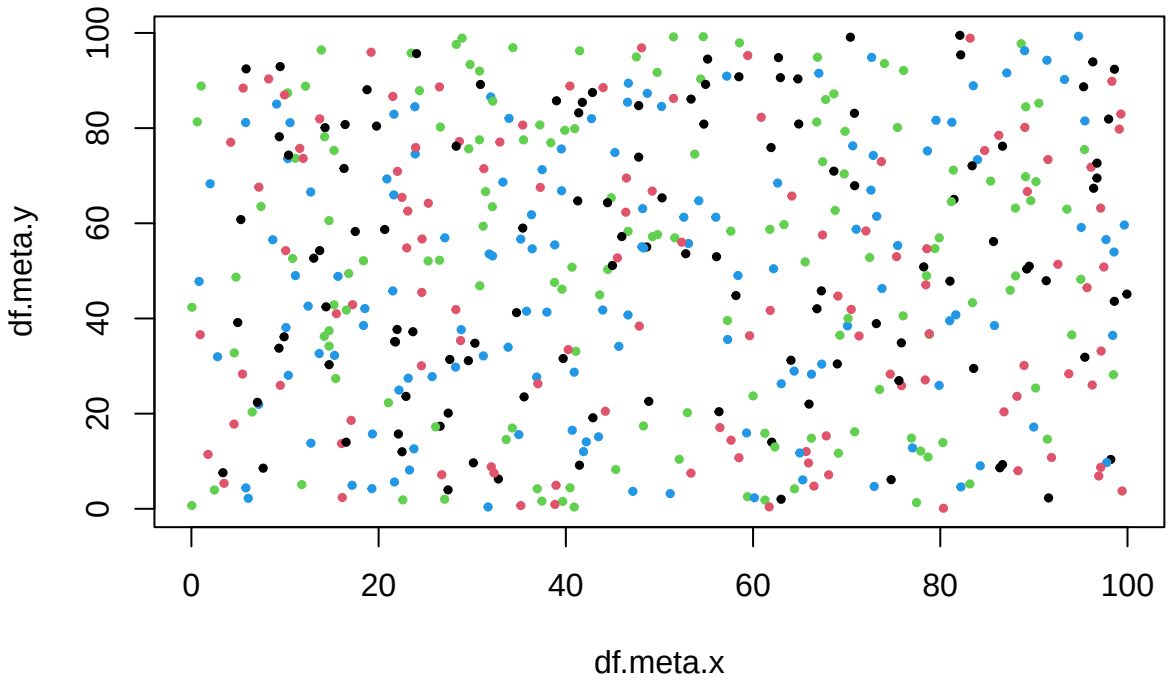
```
set.seed("123")
df <- list()
df$meta <- list()
df$meta$cell <- c(1:500)
df$meta$x <- runif(500, min = 0, max = 100)
df$meta$y <- runif(500, min = 0, max = 100)
df$meta$gene1 <- runif(500, min = 0, max = 1)
df$meta$gene2 <- runif(500, min = 0, max = 1)
df$meta$gene3 <- runif(500, min = 0, max = 1)
df$meta$gene4 <- runif(500, min = 0, max = 1)
df$meta$locat <- data.frame(df$meta$x,df$meta$y)
df$para$dist <- as.matrix(dist(df$meta$locat))
```

cell type decomposing

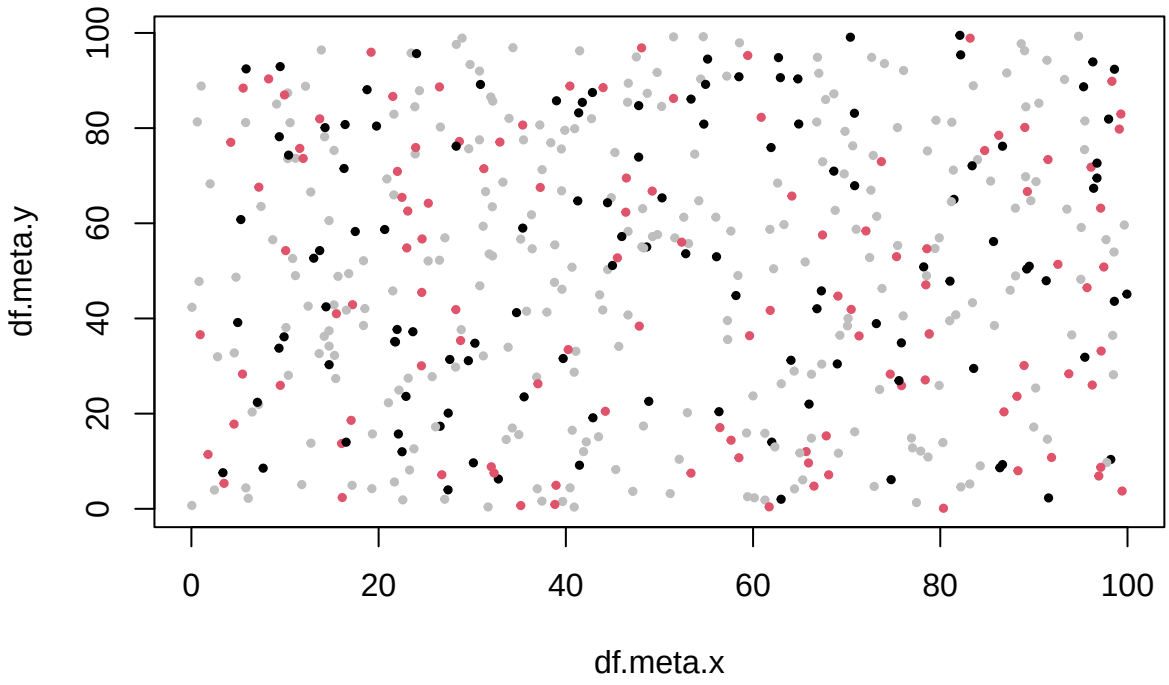
```
df$meta$cell_type<- sample(1:4, length(df$meta$cell), replace = TRUE)
```

plot different cell

```
plot(df$meta$locat, pch=19,cex = 0.5, col = as.numeric(df$meta$cell_type))
```




```
plot(df$meta$locat, pch=19,cex = 0.5, col = ifelse(df$meta$cell_type %in% c(3,4), "grey", df$meta$cell_
```




```
table(df$meta$cell_type)
```

```
##
##   1   2   3   4
## 115 112 142 131
```

Finding Neighbour

Weight KNN

```
## Weighted K-NN for lr pairs

get_lr_wknn <- function(obj,sender,receiver, k){

  if(!all(c(sender, receiver)) %in% unique(obj$meta$cell_type)) {
    stop("Can not find sender/receiver in the data")}
  else {

    #distance matrix
    distance <- obj$para$dist[obj$meta$cell_type == sender, obj$meta$cell_type == receiver]
    receiver <- list()

    #the indix of receiver
   receiver$indices <- as.matrix(t(head(apply(distance, 1, function(x) as.numeric(colnames(distance)[or
```

```
   #the distances of receiver from sender
    receiver$distances <- t(head(apply(distance, 1, function(x) sort(x, decreasing = FALSE)), k))

    weighted_counts <- receiver$distances * matrix( 1 , nrow = nrow(receiver$distances), ncol = ncol(re

    percentile_weight <- head(sort(weighted_counts), length(weighted_counts) * 0.5)
    index_wknn <- t(apply(weighted_counts, 1 , function(x) x %in% percentile_weight))

     lr_wknn  <- receiver$indices * index_wknn
     lr_wknn[lr_wknn == 0 ] <- NA

    return(lr_wknn)
  }
}


lr_example <- get_lr_wknn(df,
            sender = 1,
            receiver = 2,
            k = 4)
```

```
## Warning in all(c(sender, receiver)): coercing argument of type 'double' to
## logical
```

```
lr_example
```

```
##     [,1] [,2] [,3] [,4]
## 7    451  374   NA   NA
## 8    261   NA   NA   NA
## 9    185  306   NA   NA
## 19   359   95  217  254
## 23   423   NA   NA   NA
## 26   134  338   NA   NA
## 30   389   NA   NA   NA
## 31   347   NA   NA   NA
## 36   312  310   NA   NA
## 37    84  313  174   NA
## 41   282  486  342   NA
## 42   311   NA   NA   NA
## 43   258   70  162   NA
## 55   451   NA   NA   NA
## 56   154  467  146  284
## 59   261   NA   NA   NA
## 62   408  439  314   NA
## 64    NA   NA   NA   NA
## 65    NA   NA   NA   NA
## 72   185   NA   NA   NA
## 83   258   70   NA   NA
## 90   467  146  154   NA
## 91   292   NA   NA   NA
## 96   413   NA   NA   NA
## 97    58  301  394   NA
## 98   289   NA   NA   NA
## 103   52   NA   NA   NA
```

```
## 104  277  195  202   21
## 112   95  359  254   NA
## 117  204   NA   NA   NA
## 121  185   NA   NA   NA
## 124  210    1  459   NA
## 130  174  313   NA   NA
## 131  108   NA   NA   NA
## 133   33  480  231   NA
## 136  488   NA   NA   NA
## 137  238   NA   NA   NA
## 139  496  400   NA   NA
## 148  328  368   NA   NA
## 156  258   70  162   NA
## 160  149    1   NA   NA
## 165  346   NA   NA   NA
## 168  209  429   NA   NA
## 169  383  463  312   NA
## 170   NA   NA   NA   NA
## 171  331   61  410   NA
## 173    4  220   NA   NA
## 175   33  480  231  313
## 180  231   NA   NA   NA
## 183   NA   NA   NA   NA
## 189  261   11  461   NA
## 191  254  359   95   NA
## 194   94  266  381  198
## 219  238   NA   NA   NA
## 225  273   NA   NA   NA
## 227  204   70   NA   NA
## 230    4  248   NA   NA
## 232   70  258   NA   NA
## 234  408  314  439   NA
## 236  205  346   NA   NA
## 237  396  282  413   NA
## 249  222  380  330   NA
## 268  221   NA   NA   NA
## 270  310  463  312   NA
## 274  282  486   NA   NA
## 280  108   NA   NA   NA
## 297  491  248  220   24
## 299  459   NA   NA   NA
## 317   21  366   NA   NA
## 319   NA   NA   NA   NA
## 320  394   NA   NA   NA
## 322    1  149   NA   NA
## 324   NA   NA   NA   NA
## 325  342  486   18  273
## 326  328  368   NA   NA
## 327   11  461   NA   NA
## 334  151  430   NA   NA
## 344  254  429   NA   NA
## 345  451  374   NA   NA
## 349  149   NA   NA   NA
## 352  222  330   NA   NA
```

```
## 354  342   76  486   NA
## 356   11  461   NA   NA
## 361  459  210   NA   NA
## 373  151  430  106  330
## 375  204   NA   NA   NA
## 376  380  222   88   NA
## 379  184   NA   NA   NA
## 401  347   NA   NA   NA
## 402  292   NA   NA   NA
## 406  374   NA   NA   NA
## 415  149    1   NA   NA
## 421  108  204   NA   NA
## 427    1  221  149   NA
## 431  347  496   NA   NA
## 432  188  266   NA   NA
## 443  313  480   84   33
## 445    4  220   NA   NA
## 453  416  184   NA   NA
## 455    1  221  149   NA
## 458  410   NA   NA   NA
## 464  429  209   NA   NA
## 465  185   NA   NA   NA
## 469  332  396  433  182
## 470  301  394   NA   NA
## 471  486  342   18  282
## 474  281  174  366   NA
## 476  463  310  312   NA
## 477  185   NA   NA   NA
## 479  149    1   NA   NA
## 489   52   NA   NA   NA
## 492  389    6  289   NA
## 493  374  463  451   NA
## 498  302   NA   NA   NA
## 499  134  338   NA   NA
```

create lr index

```
create_lr_index <- function(lrpair, Nopairadd = TRUE ){
  if(Nopairadd) {
   lrpair[,1]<- ifelse(is.na(lrpair[,1]), "0",lrpair[,1] )
  }

   A = rep(row.names(lrpair), each = ncol(lrpair))
   B = as.vector(t(lrpair))
   lr_index <- cbind(A,B)

   lr_index <- lr_index[complete.cases(lr_index),]

   return(lr_index)
 }
```

```
lr_index_example <- create_lr_index(lr_example)
lr_index_example
```

```
##        A     B
##   [1,] "7"   "451"
##   [2,] "7"   "374"
##   [3,] "8"   "261"
##   [4,] "9"   "185"
##   [5,] "9"   "306"
##   [6,] "19"  "359"
##   [7,] "19"  "95"
##   [8,] "19"  "217"
##   [9,] "19"  "254"
##  [10,] "23"  "423"
##  [11,] "26"  "134"
##  [12,] "26"  "338"
##  [13,] "30"  "389"
##  [14,] "31"  "347"
##  [15,] "36"  "312"
##  [16,] "36"  "310"
##  [17,] "37"  "84"
##  [18,] "37"  "313"
##  [19,] "37"  "174"
##  [20,] "41"  "282"
##  [21,] "41"  "486"
##  [22,] "41"  "342"
##  [23,] "42"  "311"
##  [24,] "43"  "258"
##  [25,] "43"  "70"
##  [26,] "43"  "162"
##  [27,] "55"  "451"
##  [28,] "56"  "154"
##  [29,] "56"  "467"
##  [30,] "56"  "146"
##  [31,] "56"  "284"
##  [32,] "59"  "261"
##  [33,] "62"  "408"
##  [34,] "62"  "439"
##  [35,] "62"  "314"
##  [36,] "64"  "0"
##  [37,] "65"  "0"
##  [38,] "72"  "185"
##  [39,] "83"  "258"
##  [40,] "83"  "70"
##  [41,] "90"  "467"
##  [42,] "90"  "146"
##  [43,] "90"  "154"
##  [44,] "91"  "292"
##  [45,] "96"  "413"
##  [46,] "97"  "58"
##  [47,] "97"  "301"
##  [48,] "97"  "394"
##  [49,] "98"  "289"
##  [50,] "103" "52"
```

```
##  [51,] "104" "277"
##  [52,] "104" "195"
##  [53,] "104" "202"
##  [54,] "104" "21"
##  [55,] "112" "95"
##  [56,] "112" "359"
##  [57,] "112" "254"
##  [58,] "117" "204"
##  [59,] "121" "185"
##  [60,] "124" "210"
##  [61,] "124" "1"
##  [62,] "124" "459"
##  [63,] "130" "174"
##  [64,] "130" "313"
##  [65,] "131" "108"
##  [66,] "133" "33"
##  [67,] "133" "480"
##  [68,] "133" "231"
##  [69,] "136" "488"
##  [70,] "137" "238"
##  [71,] "139" "496"
##  [72,] "139" "400"
##  [73,] "148" "328"
##  [74,] "148" "368"
##  [75,] "156" "258"
##  [76,] "156" "70"
##  [77,] "156" "162"
##  [78,] "160" "149"
##  [79,] "160" "1"
##  [80,] "165" "346"
##  [81,] "168" "209"
##  [82,] "168" "429"
##  [83,] "169" "383"
##  [84,] "169" "463"
##  [85,] "169" "312"
##  [86,] "170" "0"
##  [87,] "171" "331"
##  [88,] "171" "61"
##  [89,] "171" "410"
##  [90,] "173" "4"
##  [91,] "173" "220"
##  [92,] "175" "33"
##  [93,] "175" "480"
##  [94,] "175" "231"
##  [95,] "175" "313"
##  [96,] "180" "231"
##  [97,] "183" "0"
##  [98,] "189" "261"
##  [99,] "189" "11"
## [100,] "189" "461"
## [101,] "191" "254"
## [102,] "191" "359"
## [103,] "191" "95"
## [104,] "194" "94"
```

```
## [105,] "194" "266"
## [106,] "194" "381"
## [107,] "194" "198"
## [108,] "219" "238"
## [109,] "225" "273"
## [110,] "227" "204"
## [111,] "227" "70"
## [112,] "230" "4"
## [113,] "230" "248"
## [114,] "232" "70"
## [115,] "232" "258"
## [116,] "234" "408"
## [117,] "234" "314"
## [118,] "234" "439"
## [119,] "236" "205"
## [120,] "236" "346"
## [121,] "237" "396"
## [122,] "237" "282"
## [123,] "237" "413"
## [124,] "249" "222"
## [125,] "249" "380"
## [126,] "249" "330"
## [127,] "268" "221"
## [128,] "270" "310"
## [129,] "270" "463"
## [130,] "270" "312"
## [131,] "274" "282"
## [132,] "274" "486"
## [133,] "280" "108"
## [134,] "297" "491"
## [135,] "297" "248"
## [136,] "297" "220"
## [137,] "297" "24"
## [138,] "299" "459"
## [139,] "317" "21"
## [140,] "317" "366"
## [141,] "319" "0"
## [142,] "320" "394"
## [143,] "322" "1"
## [144,] "322" "149"
## [145,] "324" "0"
## [146,] "325" "342"
## [147,] "325" "486"
## [148,] "325" "18"
## [149,] "325" "273"
## [150,] "326" "328"
## [151,] "326" "368"
## [152,] "327" "11"
## [153,] "327" "461"
## [154,] "334" "151"
## [155,] "334" "430"
## [156,] "344" "254"
## [157,] "344" "429"
## [158,] "345" "451"
```

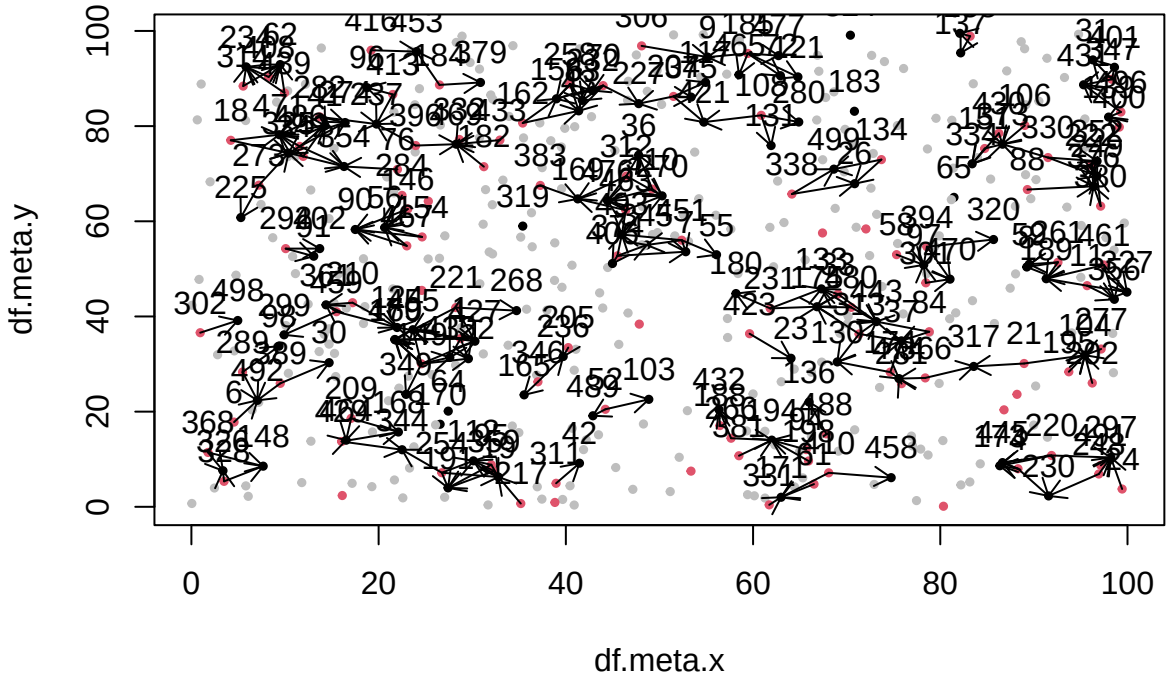
```
## [159,] "345" "374"
## [160,] "349" "149"
## [161,] "352" "222"
## [162,] "352" "330"
## [163,] "354" "342"
## [164,] "354" "76"
## [165,] "354" "486"
## [166,] "356" "11"
## [167,] "356" "461"
## [168,] "361" "459"
## [169,] "361" "210"
## [170,] "373" "151"
## [171,] "373" "430"
## [172,] "373" "106"
## [173,] "373" "330"
## [174,] "375" "204"
## [175,] "376" "380"
## [176,] "376" "222"
## [177,] "376" "88"
## [178,] "379" "184"
## [179,] "401" "347"
## [180,] "402" "292"
## [181,] "406" "374"
## [182,] "415" "149"
## [183,] "415" "1"
## [184,] "421" "108"
## [185,] "421" "204"
## [186,] "427" "1"
## [187,] "427" "221"
## [188,] "427" "149"
## [189,] "431" "347"
## [190,] "431" "496"
## [191,] "432" "188"
## [192,] "432" "266"
## [193,] "443" "313"
## [194,] "443" "480"
## [195,] "443" "84"
## [196,] "443" "33"
## [197,] "445" "4"
## [198,] "445" "220"
## [199,] "453" "416"
## [200,] "453" "184"
## [201,] "455" "1"
## [202,] "455" "221"
## [203,] "455" "149"
## [204,] "458" "410"
## [205,] "464" "429"
## [206,] "464" "209"
## [207,] "465" "185"
## [208,] "469" "332"
## [209,] "469" "396"
## [210,] "469" "433"
## [211,] "469" "182"
## [212,] "470" "301"
```

```
## [213,] "470" "394"
## [214,] "471" "486"
## [215,] "471" "342"
## [216,] "471" "18"
## [217,] "471" "282"
## [218,] "474" "281"
## [219,] "474" "174"
## [220,] "474" "366"
## [221,] "476" "463"
## [222,] "476" "310"
## [223,] "476" "312"
## [224,] "477" "185"
## [225,] "479" "149"
## [226,] "479" "1"
## [227,] "489" "52"
## [228,] "492" "389"
## [229,] "492" "6"
## [230,] "492" "289"
## [231,] "493" "374"
## [232,] "493" "463"
## [233,] "493" "451"
## [234,] "498" "302"
## [235,] "499" "134"
## [236,] "499" "338"
```

plot lr pairs

```
find_row_list_number <- function(lr_index,list_of_lists) {

       for (i in 1: length(list_of_lists)) {
          if (all(lr_index %in% list_of_lists[[i]])) {
                       ind = i
                        break
          }
       }
  return(ind)
}

plot_lr_pair <- function(obj, lr_pair, arrows = T, cluster = list()){
  #obj = df
   cell_index <- unique(as.numeric(c(rownames(lr_pair),as.numeric(lr_pair))))

    lr_index <- create_lr_index(lr_pair,Nopairadd = FALSE)

    cell_meta <- obj$meta$locat[unique(lr_index),]

     plot(df$meta$locat, pch=19,cex = 0.5, col = ifelse(obj$meta$cell_type %in% c(3,4), "grey", df$meta$

  text(df$meta$x[cell_index],df$meta$y[cell_index], cell_index, pos = 3)


for (i in 1:nrow(lr_index)) {
```

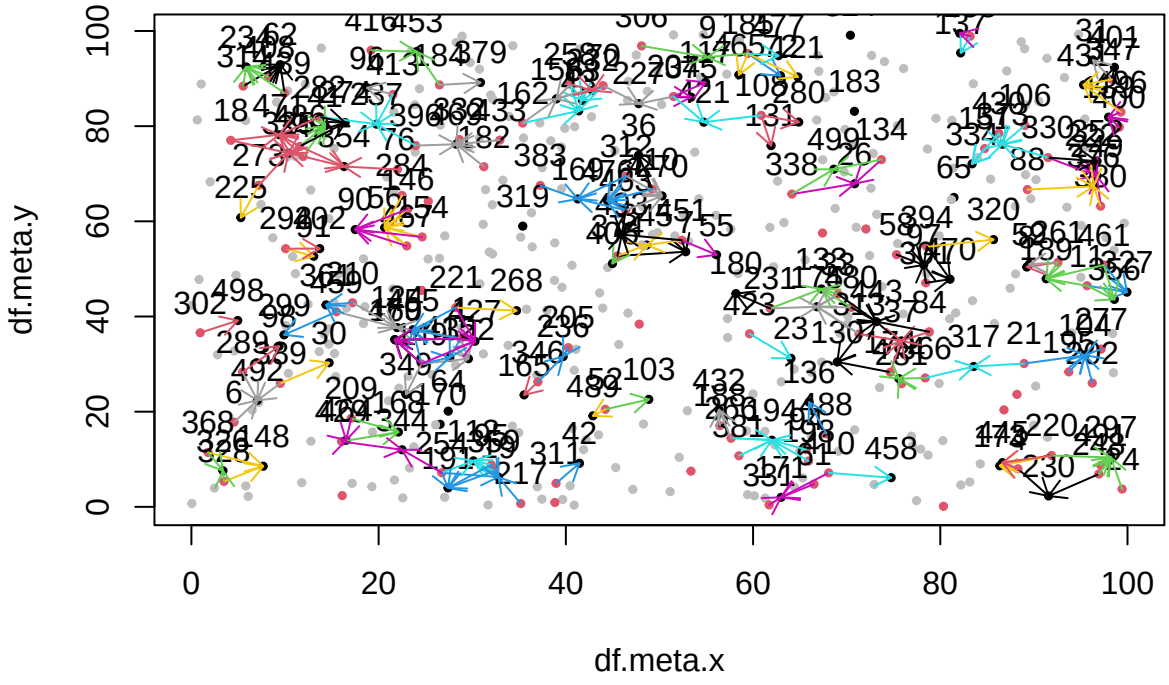
```
      #sender
       A = lr_index[i,2]
      #receiver
      B = lr_index[i,1]

      x1 <- cell_meta[rownames(cell_meta) == A, 1]
      y1 <- cell_meta[rownames(cell_meta) == A, 2]
      x2 <- cell_meta[rownames(cell_meta) == B, 1]
      y2 <- cell_meta[rownames(cell_meta) == B, 2]


if (length(cluster) == 0 ){
if (arrows)  {
   arrows(x1, y1, x2, y2, length = 0.1)
}
else{
    lines(c(x1, x2), c(y1, y2), col = 1 )
}

}
  else{

    if (arrows)  {
   arrows(x1, y1, x2, y2, length = 0.1, col =as.numeric(find_row_list_number(lr_index[i, ], cluster)))
     }
else{
      lines(c(x1, x2), c(y1, y2), col = as.numeric(find_row_list_number(lr_index[i, ],cluster)))
     }
  }

}
}


#example

    plot_lr_pair(df, lr_example)
```



find neighbourhoods

```
find_network <- function(obj, lr_pairs, plot = TRUE) {

  find_neighbor <- function(lr_pairs, Nopairadd = TRUE){

  lr_index <- create_lr_index(lr_pairs)
  neighbor <- list()

  for (value in unique(lr_index[, 1])) {
    # Extract rows with the current value
    rows <- lr_index[lr_index[,1] == value, ]
    # Append the rows to the result list as a matrix
    neighbor[[length(neighbor) + 1]] <- matrix(rows, ncol = 2, byrow = FALSE)
  }

  # Return the result
  return(neighbor)
}

  if(plot){
    neighbor_plot <- find_neighbor(lr_pairs,Nopairadd = FALSE)
    plot_lr_pair(obj, lr_pairs, arrows = T, cluster = neighbor_plot)

  }

  network <- find_neighbor(lr_pairs)
```

```
  return(network)

}

find_network(df,lr_example)
```




```
## [[1]]
##      [,1] [,2]
## [1,] "7"  "451"
## [2,] "7"  "374"
##
## [[2]]
##      [,1] [,2]
## [1,] "8"  "261"
##
## [[3]]
##      [,1] [,2]
## [1,] "9"  "185"
## [2,] "9"  "306"
##
## [[4]]
##      [,1] [,2]
## [1,] "19" "359"
## [2,] "19" "95"
## [3,] "19" "217"
## [4,] "19" "254"
```

```
##
## [[5]]
##      [,1] [,2]
## [1,] "23" "423"
##
## [[6]]
##      [,1] [,2]
## [1,] "26" "134"
## [2,] "26" "338"
##
## [[7]]
##      [,1] [,2]
## [1,] "30" "389"
##
## [[8]]
##      [,1] [,2]
## [1,] "31" "347"
##
## [[9]]
##      [,1] [,2]
## [1,] "36" "312"
## [2,] "36" "310"
##
## [[10]]
##      [,1] [,2]
## [1,] "37" "84"
## [2,] "37" "313"
## [3,] "37" "174"
##
## [[11]]
##      [,1] [,2]
## [1,] "41" "282"
## [2,] "41" "486"
## [3,] "41" "342"
##
## [[12]]
##      [,1] [,2]
## [1,] "42" "311"
##
## [[13]]
##      [,1] [,2]
## [1,] "43" "258"
## [2,] "43" "70"
## [3,] "43" "162"
##
## [[14]]
##      [,1] [,2]
## [1,] "55" "451"
##
## [[15]]
##      [,1] [,2]
## [1,] "56" "154"
## [2,] "56" "467"
## [3,] "56" "146"
```

```
## [4,] "56" "284"
##
## [[16]]
##      [,1] [,2]
## [1,] "59" "261"
##
## [[17]]
##      [,1] [,2]
## [1,] "62" "408"
## [2,] "62" "439"
## [3,] "62" "314"
##
## [[18]]
##      [,1] [,2]
## [1,] "64" "0"
##
## [[19]]
##      [,1] [,2]
## [1,] "65" "0"
##
## [[20]]
##      [,1] [,2]
## [1,] "72" "185"
##
## [[21]]
##      [,1] [,2]
## [1,] "83" "258"
## [2,] "83" "70"
##
## [[22]]
##      [,1] [,2]
## [1,] "90" "467"
## [2,] "90" "146"
## [3,] "90" "154"
##
## [[23]]
##      [,1] [,2]
## [1,] "91" "292"
##
## [[24]]
##      [,1] [,2]
## [1,] "96" "413"
##
## [[25]]
##      [,1] [,2]
## [1,] "97" "58"
## [2,] "97" "301"
## [3,] "97" "394"
##
## [[26]]
##      [,1] [,2]
## [1,] "98" "289"
##
## [[27]]
```

```
##      [,1]  [,2]
## [1,] "103" "52"
##
## [[28]]
##      [,1]  [,2]
## [1,] "104" "277"
## [2,] "104" "195"
## [3,] "104" "202"
## [4,] "104" "21"
##
## [[29]]
##      [,1]  [,2]
## [1,] "112" "95"
## [2,] "112" "359"
## [3,] "112" "254"
##
## [[30]]
##      [,1]  [,2]
## [1,] "117" "204"
##
## [[31]]
##      [,1]  [,2]
## [1,] "121" "185"
##
## [[32]]
##      [,1]  [,2]
## [1,] "124" "210"
## [2,] "124" "1"
## [3,] "124" "459"
##
## [[33]]
##      [,1]  [,2]
## [1,] "130" "174"
## [2,] "130" "313"
##
## [[34]]
##      [,1]  [,2]
## [1,] "131" "108"
##
## [[35]]
##      [,1]  [,2]
## [1,] "133" "33"
## [2,] "133" "480"
## [3,] "133" "231"
##
## [[36]]
##      [,1]  [,2]
## [1,] "136" "488"
##
## [[37]]
##      [,1]  [,2]
## [1,] "137" "238"
##
## [[38]]
```

```
##      [,1]  [,2]
## [1,] "139" "496"
## [2,] "139" "400"
##
## [[39]]
##      [,1]  [,2]
## [1,] "148" "328"
## [2,] "148" "368"
##
## [[40]]
##      [,1]  [,2]
## [1,] "156" "258"
## [2,] "156" "70"
## [3,] "156" "162"
##
## [[41]]
##      [,1]  [,2]
## [1,] "160" "149"
## [2,] "160" "1"
##
## [[42]]
##      [,1]  [,2]
## [1,] "165" "346"
##
## [[43]]
##      [,1]  [,2]
## [1,] "168" "209"
## [2,] "168" "429"
##
## [[44]]
##      [,1]  [,2]
## [1,] "169" "383"
## [2,] "169" "463"
## [3,] "169" "312"
##
## [[45]]
##      [,1]  [,2]
## [1,] "170" "0"
##
## [[46]]
##      [,1]  [,2]
## [1,] "171" "331"
## [2,] "171" "61"
## [3,] "171" "410"
##
## [[47]]
##      [,1]  [,2]
## [1,] "173" "4"
## [2,] "173" "220"
##
## [[48]]
##      [,1]  [,2]
## [1,] "175" "33"
## [2,] "175" "480"
```

```
## [3,] "175" "231"
## [4,] "175" "313"
##
## [[49]]
##      [,1]  [,2]
## [1,] "180" "231"
##
## [[50]]
##      [,1]  [,2]
## [1,] "183" "0"
##
## [[51]]
##      [,1]  [,2]
## [1,] "189" "261"
## [2,] "189" "11"
## [3,] "189" "461"
##
## [[52]]
##      [,1]  [,2]
## [1,] "191" "254"
## [2,] "191" "359"
## [3,] "191" "95"
##
## [[53]]
##      [,1]  [,2]
## [1,] "194" "94"
## [2,] "194" "266"
## [3,] "194" "381"
## [4,] "194" "198"
##
## [[54]]
##      [,1]  [,2]
## [1,] "219" "238"
##
## [[55]]
##      [,1]  [,2]
## [1,] "225" "273"
##
## [[56]]
##      [,1]  [,2]
## [1,] "227" "204"
## [2,] "227" "70"
##
## [[57]]
##      [,1]  [,2]
## [1,] "230" "4"
## [2,] "230" "248"
##
## [[58]]
##      [,1]  [,2]
## [1,] "232" "70"
## [2,] "232" "258"
##
## [[59]]
```

```
##      [,1]  [,2]
## [1,] "234" "408"
## [2,] "234" "314"
## [3,] "234" "439"
##
## [[60]]
##      [,1]  [,2]
## [1,] "236" "205"
## [2,] "236" "346"
##
## [[61]]
##      [,1]  [,2]
## [1,] "237" "396"
## [2,] "237" "282"
## [3,] "237" "413"
##
## [[62]]
##      [,1]  [,2]
## [1,] "249" "222"
## [2,] "249" "380"
## [3,] "249" "330"
##
## [[63]]
##      [,1]  [,2]
## [1,] "268" "221"
##
## [[64]]
##      [,1]  [,2]
## [1,] "270" "310"
## [2,] "270" "463"
## [3,] "270" "312"
##
## [[65]]
##      [,1]  [,2]
## [1,] "274" "282"
## [2,] "274" "486"
##
## [[66]]
##      [,1]  [,2]
## [1,] "280" "108"
##
## [[67]]
##      [,1]  [,2]
## [1,] "297" "491"
## [2,] "297" "248"
## [3,] "297" "220"
## [4,] "297" "24"
##
## [[68]]
##      [,1]  [,2]
## [1,] "299" "459"
##
## [[69]]
##      [,1]  [,2]
```

```
## [1,] "317" "21"
## [2,] "317" "366"
##
## [[70]]
##      [,1]  [,2]
## [1,] "319" "0"
##
## [[71]]
##      [,1]  [,2]
## [1,] "320" "394"
##
## [[72]]
##      [,1]  [,2]
## [1,] "322" "1"
## [2,] "322" "149"
##
## [[73]]
##      [,1]  [,2]
## [1,] "324" "0"
##
## [[74]]
##      [,1]  [,2]
## [1,] "325" "342"
## [2,] "325" "486"
## [3,] "325" "18"
## [4,] "325" "273"
##
## [[75]]
##      [,1]  [,2]
## [1,] "326" "328"
## [2,] "326" "368"
##
## [[76]]
##      [,1]  [,2]
## [1,] "327" "11"
## [2,] "327" "461"
##
## [[77]]
##      [,1]  [,2]
## [1,] "334" "151"
## [2,] "334" "430"
##
## [[78]]
##      [,1]  [,2]
## [1,] "344" "254"
## [2,] "344" "429"
##
## [[79]]
##      [,1]  [,2]
## [1,] "345" "451"
## [2,] "345" "374"
##
## [[80]]
##      [,1]  [,2]
```

```
## [1,] "349" "149"
##
## [[81]]
##      [,1]  [,2]
## [1,] "352" "222"
## [2,] "352" "330"
##
## [[82]]
##      [,1]  [,2]
## [1,] "354" "342"
## [2,] "354" "76"
## [3,] "354" "486"
##
## [[83]]
##      [,1]  [,2]
## [1,] "356" "11"
## [2,] "356" "461"
##
## [[84]]
##      [,1]  [,2]
## [1,] "361" "459"
## [2,] "361" "210"
##
## [[85]]
##      [,1]  [,2]
## [1,] "373" "151"
## [2,] "373" "430"
## [3,] "373" "106"
## [4,] "373" "330"
##
## [[86]]
##      [,1]  [,2]
## [1,] "375" "204"
##
## [[87]]
##      [,1]  [,2]
## [1,] "376" "380"
## [2,] "376" "222"
## [3,] "376" "88"
##
## [[88]]
##      [,1]  [,2]
## [1,] "379" "184"
##
## [[89]]
##      [,1]  [,2]
## [1,] "401" "347"
##
## [[90]]
##      [,1]  [,2]
## [1,] "402" "292"
##
## [[91]]
##      [,1]  [,2]
```

```
## [1,] "406" "374"
##
## [[92]]
##      [,1]  [,2]
## [1,] "415" "149"
## [2,] "415" "1"
##
## [[93]]
##      [,1]  [,2]
## [1,] "421" "108"
## [2,] "421" "204"
##
## [[94]]
##      [,1]  [,2]
## [1,] "427" "1"
## [2,] "427" "221"
## [3,] "427" "149"
##
## [[95]]
##      [,1]  [,2]
## [1,] "431" "347"
## [2,] "431" "496"
##
## [[96]]
##      [,1]  [,2]
## [1,] "432" "188"
## [2,] "432" "266"
##
## [[97]]
##      [,1]  [,2]
## [1,] "443" "313"
## [2,] "443" "480"
## [3,] "443" "84"
## [4,] "443" "33"
##
## [[98]]
##      [,1]  [,2]
## [1,] "445" "4"
## [2,] "445" "220"
##
## [[99]]
##      [,1]  [,2]
## [1,] "453" "416"
## [2,] "453" "184"
##
## [[100]]
##      [,1]  [,2]
## [1,] "455" "1"
## [2,] "455" "221"
## [3,] "455" "149"
##
## [[101]]
##      [,1]  [,2]
## [1,] "458" "410"
```

```
##
## [[102]]
##      [,1]  [,2]
## [1,] "464" "429"
## [2,] "464" "209"
##
## [[103]]
##      [,1]  [,2]
## [1,] "465" "185"
##
## [[104]]
##      [,1]  [,2]
## [1,] "469" "332"
## [2,] "469" "396"
## [3,] "469" "433"
## [4,] "469" "182"
##
## [[105]]
##      [,1]  [,2]
## [1,] "470" "301"
## [2,] "470" "394"
##
## [[106]]
##      [,1]  [,2]
## [1,] "471" "486"
## [2,] "471" "342"
## [3,] "471" "18"
## [4,] "471" "282"
##
## [[107]]
##      [,1]  [,2]
## [1,] "474" "281"
## [2,] "474" "174"
## [3,] "474" "366"
##
## [[108]]
##      [,1]  [,2]
## [1,] "476" "463"
## [2,] "476" "310"
## [3,] "476" "312"
##
## [[109]]
##      [,1]  [,2]
## [1,] "477" "185"
##
## [[110]]
##      [,1]  [,2]
## [1,] "479" "149"
## [2,] "479" "1"
##
## [[111]]
##      [,1]  [,2]
## [1,] "489" "52"
##
```

```
## [[112]]
##      [,1]  [,2]
## [1,] "492" "389"
## [2,] "492" "6"
## [3,] "492" "289"
##
## [[113]]
##      [,1]  [,2]
## [1,] "493" "374"
## [2,] "493" "463"
## [3,] "493" "451"
##
## [[114]]
##      [,1]  [,2]
## [1,] "498" "302"
##
## [[115]]
##      [,1]  [,2]
## [1,] "499" "134"
## [2,] "499" "338"
```

SVC model

##dataset generating

```
build_dataset <- function(df, lr) {
  lr_network <- find_network(df, lr, plot = FALSE)
  dataset <- vector("list", length(lr_network))

  # Initialize dataset as a list with the required number of elements


  for (j in 1:length(lr_network)) {
    single_network <- lr_network[[j]]

    for (i in 1:(length(single_network)/2)) {

      if (length(single_network)/2 < 2) {
        ligand <- as.numeric(single_network[[1]])
        receptor <- as.numeric(single_network[[2]])
      } else {
        row <- single_network[i, ]
        ligand <- as.numeric(row[1])
        receptor <- as.numeric(row[2])
      }

      gene_class_L <- df$meta$cell_type[ligand]
      gene_class_R <- df$meta$cell_type[receptor]

      expression <- cbind(df$meta$gene1, df$meta$gene2, df$meta$gene3, df$meta$gene4)
      x <- expression[ligand, gene_class_L]
      y <- expression[receptor, gene_class_R]
     if(length(y) == 0){
```

```
      y = 0
    }

    new_expression_row <- c(x, y)
    dataset[[j]] <- rbind(dataset[[j]], new_expression_row)
   }
  }

  return(dataset)
}

ds_example <- build_dataset(df, lr_example)

ds_example
```

```
## [[1]]
##                          [,1]      [,2]
## new_expression_row 0.7736921 0.1359953
## new_expression_row 0.7736921 0.3696462
##
## [[2]]
##                          [,1]      [,2]
## new_expression_row 0.2954001 0.6833322
##
## [[3]]
##                           [,1]       [,2]
## new_expression_row 0.06562811 0.32333609
## new_expression_row 0.06562811 0.04999978
##
## [[4]]
##                          [,1]      [,2]
## new_expression_row 0.4684321 0.6668018
## new_expression_row 0.4684321 0.3278221
## new_expression_row 0.4684321 0.5583189
## new_expression_row 0.4684321 0.6228258
##
## [[5]]
##                          [,1]      [,2]
## new_expression_row 0.5855656 0.9892651
##
## [[6]]
##                          [,1]      [,2]
## new_expression_row 0.3185631 0.6543813
## new_expression_row 0.3185631 0.3146913
##
## [[7]]
##                          [,1]      [,2]
## new_expression_row 0.5551772 0.3235622
##
## [[8]]
##                          [,1]      [,2]
## new_expression_row 0.2312309 0.6424513
##
## [[9]]
```

```
##                         [,1]      [,2]
## new_expression_row 0.5479943 0.5973777
## new_expression_row 0.5479943 0.5184672
##
## [[10]]
##                         [,1]      [,2]
## new_expression_row 0.1492442 0.1044625
## new_expression_row 0.1492442 0.7955761
## new_expression_row 0.1492442 0.7452819
##
## [[11]]
##                         [,1]      [,2]
## new_expression_row 0.8332948 0.2720803
## new_expression_row 0.8332948 0.1380302
## new_expression_row 0.8332948 0.4816463
##
## [[12]]
##                         [,1]      [,2]
## new_expression_row 0.7195873 0.2661275
##
## [[13]]
##                         [,1]       [,2]
## new_expression_row 0.4568868 0.40104325
## new_expression_row 0.4568868 0.86505856
## new_expression_row 0.4568868 0.03717817
##
## [[14]]
##                         [,1]      [,2]
## new_expression_row 0.5553136 0.1359953
##
## [[15]]
##                         [,1]      [,2]
## new_expression_row 0.2764399 0.3827119
## new_expression_row 0.2764399 0.3026620
## new_expression_row 0.2764399 0.9398556
## new_expression_row 0.2764399 0.6133537
##
## [[16]]
##                         [,1]      [,2]
## new_expression_row 0.2882662 0.6833322
##
## [[17]]
##                        [,1]      [,2]
## new_expression_row 0.173043 0.7278129
## new_expression_row 0.173043 0.2805975
## new_expression_row 0.173043 0.6892860
##
## [[18]]
##                         [,1] [,2]
## new_expression_row 0.3699073    0
##
## [[19]]
##                          [,1] [,2]
## new_expression_row 0.07772155    0
```

```
##
## [[20]]
##                          [,1]      [,2]
## new_expression_row 0.8489141 0.3233361
##
## [[21]]
##                          [,1]      [,2]
## new_expression_row 0.3196421 0.4010432
## new_expression_row 0.3196421 0.8650586
##
## [[22]]
##                          [,1]      [,2]
## new_expression_row 0.4801264 0.3026620
## new_expression_row 0.4801264 0.9398556
## new_expression_row 0.4801264 0.3827119
##
## [[23]]
##                          [,1]       [,2]
## new_expression_row 0.8062479 0.03515099
##
## [[24]]
##                          [,1]      [,2]
## new_expression_row 0.7570985 0.5873414
##
## [[25]]
##                          [,1]      [,2]
## new_expression_row 0.9924553 0.2846149
## new_expression_row 0.9924553 0.1552641
## new_expression_row 0.9924553 0.7352699
##
## [[26]]
##                         [,1]      [,2]
## new_expression_row 0.624553 0.6774541
##
## [[27]]
##                         [,1]       [,2]
## new_expression_row 0.217048 0.08848456
##
## [[28]]
##                          [,1]       [,2]
## new_expression_row 0.7442385 0.22225309
## new_expression_row 0.7442385 0.85205914
## new_expression_row 0.7442385 0.08506997
## new_expression_row 0.7442385 0.70532928
##
## [[29]]
##                          [,1]      [,2]
## new_expression_row 0.8263316 0.3278221
## new_expression_row 0.8263316 0.6668018
## new_expression_row 0.8263316 0.6228258
##
## [[30]]
##                          [,1]      [,2]
## new_expression_row 0.4218101 0.9359073
```

```
##
## [[31]]
##                          [,1]      [,2]
## new_expression_row 0.2021846 0.3233361
##
## [[32]]
##                          [,1]      [,2]
## new_expression_row 0.3135179 0.6154085
## new_expression_row 0.3135179 0.9380283
## new_expression_row 0.3135179 0.5582040
##
## [[33]]
##                          [,1]      [,2]
## new_expression_row 0.5647945 0.7452819
## new_expression_row 0.5647945 0.7955761
##
## [[34]]
##                          [,1]      [,2]
## new_expression_row 0.5402562 0.1280091
##
## [[35]]
##                          [,1]      [,2]
## new_expression_row 0.7029795 0.8991245
## new_expression_row 0.7029795 0.1881985
## new_expression_row 0.7029795 0.9946604
##
## [[36]]
##                          [,1]     [,2]
## new_expression_row 0.7524442 0.451247
##
## [[37]]
##                          [,1]      [,2]
## new_expression_row 0.3814226 0.8093463
##
## [[38]]
##                          [,1]      [,2]
## new_expression_row 0.5316833 0.8934178
## new_expression_row 0.5316833 0.5357779
##
## [[39]]
##                          [,1]      [,2]
## new_expression_row 0.6284048 0.8814340
## new_expression_row 0.6284048 0.8863063
##
## [[40]]
##                          [,1]       [,2]
## new_expression_row 0.3719787 0.40104325
## new_expression_row 0.3719787 0.86505856
## new_expression_row 0.3719787 0.03717817
##
## [[41]]
##                          [,1]      [,2]
## new_expression_row 0.2099981 0.3944627
## new_expression_row 0.2099981 0.9380283
```

```
##
## [[42]]
##                          [,1]      [,2]
## new_expression_row 0.1186903 0.0152991
##
## [[43]]
##                          [,1]      [,2]
## new_expression_row 0.2869576 0.9898528
## new_expression_row 0.2869576 0.5384723
##
## [[44]]
##                          [,1]      [,2]
## new_expression_row 0.1494125 0.3594329
## new_expression_row 0.1494125 0.1303785
## new_expression_row 0.1494125 0.5973777
##
## [[45]]
##                         [,1] [,2]
## new_expression_row 0.972432    0
##
## [[46]]
##                          [,1]      [,2]
## new_expression_row 0.4105144 0.8635937
## new_expression_row 0.4105144 0.7920057
## new_expression_row 0.4105144 0.6144502
##
## [[47]]
##                          [,1]      [,2]
## new_expression_row 0.6485563 0.2306149
## new_expression_row 0.6485563 0.1098016
##
## [[48]]
##                          [,1]      [,2]
## new_expression_row 0.2166621 0.8991245
## new_expression_row 0.2166621 0.1881985
## new_expression_row 0.2166621 0.9946604
## new_expression_row 0.2166621 0.7955761
##
## [[49]]
##                          [,1]      [,2]
## new_expression_row 0.6348964 0.9946604
##
## [[50]]
##                          [,1] [,2]
## new_expression_row 0.6790228    0
##
## [[51]]
##                          [,1]      [,2]
## new_expression_row 0.6929137 0.6833322
## new_expression_row 0.6929137 0.4520856
## new_expression_row 0.6929137 0.9894184
##
## [[52]]
##                          [,1]      [,2]
```

```
## new_expression_row 0.1238577 0.6228258
## new_expression_row 0.1238577 0.6668018
## new_expression_row 0.1238577 0.3278221
##
## [[53]]
##                          [,1]      [,2]
## new_expression_row 0.7661665 0.5928633
## new_expression_row 0.7661665 0.9920531
## new_expression_row 0.7661665 0.8529309
## new_expression_row 0.7661665 0.6526816
##
## [[54]]
##                          [,1]      [,2]
## new_expression_row 0.1979236 0.8093463
##
## [[55]]
##                         [,1]      [,2]
## new_expression_row 0.954255 0.1496045
##
## [[56]]
##                          [,1]      [,2]
## new_expression_row 0.4936162 0.9359073
## new_expression_row 0.4936162 0.8650586
##
## [[57]]
##                          [,1]      [,2]
## new_expression_row 0.5309357 0.2306149
## new_expression_row 0.5309357 0.7821636
##
## [[58]]
##                          [,1]      [,2]
## new_expression_row 0.2775601 0.8650586
## new_expression_row 0.2775601 0.4010432
##
## [[59]]
##                          [,1]      [,2]
## new_expression_row 0.5887063 0.7278129
## new_expression_row 0.5887063 0.6892860
## new_expression_row 0.5887063 0.2805975
##
## [[60]]
##                           [,1]      [,2]
## new_expression_row 0.04311728 0.8452937
## new_expression_row 0.04311728 0.0152991
##
## [[61]]
##                          [,1]      [,2]
## new_expression_row 0.1106788 0.5580462
## new_expression_row 0.1106788 0.2720803
## new_expression_row 0.1106788 0.5873414
##
## [[62]]
##                          [,1]      [,2]
## new_expression_row 0.1907175 0.3773421
```

```
## new_expression_row 0.1907175 0.3476414
## new_expression_row 0.1907175 0.7698103
##
## [[63]]
##                          [,1]      [,2]
## new_expression_row 0.9373274 0.8189076
##
## [[64]]
##                          [,1]      [,2]
## new_expression_row 0.9728035 0.5184672
## new_expression_row 0.9728035 0.1303785
## new_expression_row 0.9728035 0.5973777
##
## [[65]]
##                          [,1]      [,2]
## new_expression_row 0.5661824 0.2720803
## new_expression_row 0.5661824 0.1380302
##
## [[66]]
##                          [,1]      [,2]
## new_expression_row 0.3921965 0.1280091
##
## [[67]]
##                         [,1]      [,2]
## new_expression_row 0.770597 0.4641535
## new_expression_row 0.770597 0.7821636
## new_expression_row 0.770597 0.1098016
## new_expression_row 0.770597 0.4327241
##
## [[68]]
##                         [,1]     [,2]
## new_expression_row 0.838906 0.558204
##
## [[69]]
##                          [,1]      [,2]
## new_expression_row 0.7790689 0.7053293
## new_expression_row 0.7790689 0.2232563
##
## [[70]]
##                          [,1] [,2]
## new_expression_row 0.5480433    0
##
## [[71]]
##                           [,1]      [,2]
## new_expression_row 0.03390152 0.7352699
##
## [[72]]
##                          [,1]      [,2]
## new_expression_row 0.3377528 0.9380283
## new_expression_row 0.3377528 0.3944627
##
## [[73]]
##                          [,1] [,2]
## new_expression_row 0.3571064    0
```

```
##
## [[74]]
##                          [,1]      [,2]
## new_expression_row 0.975365 0.4816463
## new_expression_row 0.975365 0.1380302
## new_expression_row 0.975365 0.3804480
## new_expression_row 0.975365 0.1496045
##
## [[75]]
##                           [,1]      [,2]
## new_expression_row 0.3843785 0.8814340
## new_expression_row 0.3843785 0.8863063
##
## [[76]]
##                           [,1]      [,2]
## new_expression_row 0.4886735 0.4520856
## new_expression_row 0.4886735 0.9894184
##
## [[77]]
##                           [,1]      [,2]
## new_expression_row 0.6173444 0.2769614
## new_expression_row 0.6173444 0.9610610
##
## [[78]]
##                           [,1]      [,2]
## new_expression_row 0.2490469 0.6228258
## new_expression_row 0.2490469 0.5384723
##
## [[79]]
##                           [,1]      [,2]
## new_expression_row 0.6546267 0.1359953
## new_expression_row 0.6546267 0.3696462
##
## [[80]]
##                           [,1]      [,2]
## new_expression_row 0.5363599 0.3944627
##
## [[81]]
##                            [,1]      [,2]
## new_expression_row 0.00484174 0.3773421
## new_expression_row 0.00484174 0.7698103
##
## [[82]]
##                           [,1]      [,2]
## new_expression_row 0.0248171 0.4816463
## new_expression_row 0.0248171 0.7460936
## new_expression_row 0.0248171 0.1380302
##
## [[83]]
##                           [,1]      [,2]
## new_expression_row 0.4755582 0.4520856
## new_expression_row 0.4755582 0.9894184
##
## [[84]]
```

```
##                          [,1]      [,2]
## new_expression_row 0.6488825 0.5582040
## new_expression_row 0.6488825 0.6154085
##
## [[85]]
##                          [,1]       [,2]
## new_expression_row 0.4055415 0.27696142
## new_expression_row 0.4055415 0.96106100
## new_expression_row 0.4055415 0.04583463
## new_expression_row 0.4055415 0.76981030
##
## [[86]]
##                          [,1]      [,2]
## new_expression_row 0.6856713 0.9359073
##
## [[87]]
##                          [,1]      [,2]
## new_expression_row 0.8235256 0.3476414
## new_expression_row 0.8235256 0.3773421
## new_expression_row 0.8235256 0.3908899
##
## [[88]]
##                           [,1]      [,2]
## new_expression_row 0.01883216 0.3289165
##
## [[89]]
##                          [,1]      [,2]
## new_expression_row 0.2332397 0.6424513
##
## [[90]]
##                          [,1]       [,2]
## new_expression_row 0.2306536 0.03515099
##
## [[91]]
##                          [,1]      [,2]
## new_expression_row 0.7581855 0.3696462
##
## [[92]]
##                          [,1]      [,2]
## new_expression_row 0.1829407 0.3944627
## new_expression_row 0.1829407 0.9380283
##
## [[93]]
##                          [,1]      [,2]
## new_expression_row 0.8556767 0.1280091
## new_expression_row 0.8556767 0.9359073
##
## [[94]]
##                          [,1]      [,2]
## new_expression_row 0.4654494 0.9380283
## new_expression_row 0.4654494 0.8189076
## new_expression_row 0.4654494 0.3944627
##
## [[95]]
```

```
##                          [,1]      [,2]
## new_expression_row 0.8697207 0.6424513
## new_expression_row 0.8697207 0.8934178
##
## [[96]]
##                          [,1]      [,2]
## new_expression_row 0.3003542 0.5968069
## new_expression_row 0.3003542 0.9920531
##
## [[97]]
##                           [,1]      [,2]
## new_expression_row 0.04196682 0.7955761
## new_expression_row 0.04196682 0.1881985
## new_expression_row 0.04196682 0.1044625
## new_expression_row 0.04196682 0.8991245
##
## [[98]]
##                          [,1]      [,2]
## new_expression_row 0.7395082 0.2306149
## new_expression_row 0.7395082 0.1098016
##
## [[99]]
##                          [,1]       [,2]
## new_expression_row 0.7135778 0.08227004
## new_expression_row 0.7135778 0.32891654
##
## [[100]]
##                          [,1]      [,2]
## new_expression_row 0.5751206 0.9380283
## new_expression_row 0.5751206 0.8189076
## new_expression_row 0.5751206 0.3944627
##
## [[101]]
##                          [,1]      [,2]
## new_expression_row 0.7248558 0.6144502
##
## [[102]]
##                          [,1]      [,2]
## new_expression_row 0.1899177 0.5384723
## new_expression_row 0.1899177 0.9898528
##
## [[103]]
##                           [,1]      [,2]
## new_expression_row 0.02596878 0.3233361
##
## [[104]]
##                          [,1]      [,2]
## new_expression_row 0.4776486 0.5007421
## new_expression_row 0.4776486 0.5580462
## new_expression_row 0.4776486 0.0322670
## new_expression_row 0.4776486 0.5390194
##
## [[105]]
##                          [,1]      [,2]
```

```
## new_expression_row 0.4629615 0.1552641
## new_expression_row 0.4629615 0.7352699
##
## [[106]]
##                          [,1]      [,2]
## new_expression_row 0.7001786 0.1380302
## new_expression_row 0.7001786 0.4816463
## new_expression_row 0.7001786 0.3804480
## new_expression_row 0.7001786 0.2720803
##
## [[107]]
##                          [,1]      [,2]
## new_expression_row 0.3959316 0.1609630
## new_expression_row 0.3959316 0.7452819
## new_expression_row 0.3959316 0.2232563
##
## [[108]]
##                          [,1]      [,2]
## new_expression_row 0.7252994 0.1303785
## new_expression_row 0.7252994 0.5184672
## new_expression_row 0.7252994 0.5973777
##
## [[109]]
##                          [,1]      [,2]
## new_expression_row 0.4700026 0.3233361
##
## [[110]]
##                          [,1]      [,2]
## new_expression_row 0.8459358 0.3944627
## new_expression_row 0.8459358 0.9380283
##
## [[111]]
##                          [,1]       [,2]
## new_expression_row 0.5821439 0.08848456
##
## [[112]]
##                           [,1]      [,2]
## new_expression_row 0.01152176 0.3235622
## new_expression_row 0.01152176 0.5566323
## new_expression_row 0.01152176 0.6774541
##
## [[113]]
##                          [,1]      [,2]
## new_expression_row 0.9964457 0.3696462
## new_expression_row 0.9964457 0.1303785
## new_expression_row 0.9964457 0.1359953
##
## [[114]]
##                          [,1]     [,2]
## new_expression_row 0.2007596 0.845851
##
## [[115]]
##                          [,1]      [,2]
## new_expression_row 0.6460372 0.6543813
```

```
## new_expression_row 0.6460372 0.3146913
```

##SVC model

```
#model1 optimal method = Nelder-Mead, initial value = 0
#the objecctive function is not converging for this method
SVC_model1 <- function(obj, lr, lambda1, lambda2) {

  ds <- build_dataset(obj, lr)
  ds_matrix<- do.call(rbind, ds)
  receiver <- obj$meta$cell_type[as.numeric(rownames(lr))[1]]
  sender <- obj$meta$cell_type[na.omit(as.numeric(lr_example))[[1]]]

  Y <- as.matrix(ds_matrix[,1],ncol = 1)
  X <- cbind(rep(1,nrow(ds_matrix)),as.matrix(ds_matrix[,2],ncol = 1))
  lr[,1]<- ifelse(is.na(lr[,1]), "0",lr[,1] )


   count_non_na <- function(row) {
          sum(!is.na(row))
   }

        non_na_counts <- as.matrix(apply(lr, 1, count_non_na),ncol = 1)


    objective_function <- function(beta) {

       beta <- matrix(beta, ncol = 2)
       beta2 <- beta[rep(seq_len(nrow(beta)), times = non_na_counts), ]

       residuals <- Y - matrix(rowSums(X * beta2), ncol = 1)

      # Calculate the squared sum of residuals
      L <- sum(residuals^2) / (2 * nrow(X))

      # Calculate the penalty terms
      R1 <- lambda1 * sum(abs(beta))

      # Calculate the penalty term R2


      Dis <- obj$para$dist[obj$meta$cell_type == receiver, obj$meta$cell_type == receiver]


     weight <- 1/Dis
     weight[weight == Inf] <- 0

      R2 <- lambda2 * sum(abs(as.matrix(dist(beta[,1]))) * weight +  abs(as.matrix(dist(beta[,1]))) * w

      objective_value <- L + R1 + R2
      return(objective_value)
    }

    result <- optim(par = matrix(rep(0, 2*length(ds)),ncol = 2), fn = objective_function, method = "Nelc
```

```
    beta_list <- result

    return(beta_list)

  }


beta_coefficients <- SVC_model1(df, lr_example,
                                lambda1 = 0.001,
                                lambda2 = 0.001)


beta_coefficients
```

```
 # model2 optimal method = BFGS, initial value = 0
SVC_model2 <- function(obj, lr, lambda1, lambda2) {

  ds <- build_dataset(obj, lr)
  ds_matrix<- do.call(rbind, ds)
  receiver <- obj$meta$cell_type[as.numeric(rownames(lr))[1]]
  sender <- obj$meta$cell_type[na.omit(as.numeric(lr_example))[[1]]]

  Y <- as.matrix(ds_matrix[,1],ncol = 1)
  X <- cbind(rep(1,nrow(ds_matrix)),as.matrix(ds_matrix[,2],ncol = 1))
  lr[,1]<- ifelse(is.na(lr[,1]), "0",lr[,1] )

   count_non_na <- function(row) {
          sum(!is.na(row))
   }

        non_na_counts <- as.matrix(apply(lr, 1, count_non_na),ncol = 1)


    objective_function <- function(beta) {

       beta <- matrix(beta, ncol = 2)
       beta2 <- beta[rep(seq_len(nrow(beta)), times = non_na_counts), ]

       residuals <- Y - matrix(rowSums(X * beta2), ncol = 1)

      # Calculate the squared sum of residuals
      L <- sum(residuals^2) / (2 * nrow(X))

      # Calculate the penalty terms
      R1 <- lambda1 * sum(abs(beta))

      # Calculate the penalty term R2
      Dis <- obj$para$dist[obj$meta$cell_type == receiver, obj$meta$cell_type == receiver]


     weight <- 1/Dis
     weight[weight == Inf] <- 0

      R2 <- lambda2 * sum(abs(as.matrix(dist(beta[,1]))) * weight +  abs(as.matrix(dist(beta[,1]))) * w
```

```
      objective_value <- L + R1 + R2

      return(objective_value)
    }


    result <- optim(par = matrix(rep(0, 2*length(ds)),ncol = 2), fn = objective_function, method = "BFGS

    beta_list <- result

    #likelihood test for /beta = 0
    beta1 <- beta_list$par
    beta2 <- beta1[rep(seq_len(nrow( beta1)), times = non_na_counts), ]

    l_hypo =log(prod(matrix(rowSums(X * beta2), ncol = 1)))

    l_null = log(prod(Y))
    likelihood_score = 2*(l_hypo- l_null)
    beta_list$score = likelihood_score

    return(beta_list)

  }


beta_coefficients <- SVC_model2(df, lr_example,
                                lambda1 = 0.0001,
                                lambda2 = 0.001)
```

```
## Warning in log(prod(matrix(rowSums(X * beta2), ncol = 1))): NaNs produced
```

```
beta_coefficients
```

```
## $par
##              [,1]          [,2]
##   [1,] 0.4446009  5.843730e-01
##   [2,] 0.4443093 -1.361243e-01
##   [3,] 0.4445285 -3.234657e-01
##   [4,] 0.4453286 -1.339660e-03
##   [5,] 0.4444368  1.126844e-01
##   [6,] 0.4446285 -1.533113e-01
##   [7,] 0.4446313  1.348321e-01
##   [8,] 0.4441162 -2.235191e-01
##   [9,] 0.4441276  1.919164e-01
##  [10,] 0.4441633 -3.914164e-01
##  [11,] 0.4451410  9.359924e-01
##  [12,] 0.4445986  2.646984e-01
##  [13,] 0.4446798 -1.686848e-02
##  [14,] 0.4440746  2.028121e-02
##  [15,] 0.4445532 -2.049972e-01
##  [16,] 0.4446667 -1.463496e-01
##  [17,] 0.4448179 -4.211621e-01
##  [18,] 0.4443583  0.000000e+00
##  [19,] 0.4448794  0.000000e+00
```

```
##  [20,] 0.4447450  4.641884e-01
##  [21,] 0.4444840 -1.584313e-01
##  [22,] 0.4450986  9.914109e-03
##  [23,] 0.4454908  6.631406e-03
##  [24,] 0.4446114  4.745576e-01
##  [25,] 0.4446445  1.052262e+00
##  [26,] 0.4444620  2.643444e-01
##  [27,] 0.4447631 -1.094880e-02
##  [28,] 0.4445811  3.968620e-01
##  [29,] 0.4444530  6.501110e-01
##  [30,] 0.4456243 -2.247477e-02
##  [31,] 0.4445044 -1.659553e-01
##  [32,] 0.4452145 -1.281818e-01
##  [33,] 0.4449893  1.148154e-01
##  [34,] 0.4452181  6.853523e-03
##  [35,] 0.4441945  2.821095e-01
##  [36,] 0.4450285  4.440725e-01
##  [37,] 0.4446394 -6.408953e-02
##  [38,] 0.4448530  7.590225e-02
##  [39,] 0.4424598  1.788697e-01
##  [40,] 0.4447214 -1.028846e-01
##  [41,] 0.4447416 -2.903052e-01
##  [42,] 0.4447282  5.356144e-03
##  [43,] 0.4447583 -1.669254e-01
##  [44,] 0.4443608 -5.324141e-01
##  [45,] 0.4448725  0.000000e+00
##  [46,] 0.4465525  7.977630e-03
##  [47,] 0.4448090  2.742221e-01
##  [48,] 0.4450571 -2.256712e-01
##  [49,] 0.4439534  1.626892e-01
##  [50,] 0.4446229  0.000000e+00
##  [51,] 0.4448544  2.887275e-01
##  [52,] 0.4455295 -5.604066e-01
##  [53,] 0.4446662  4.102137e-01
##  [54,] 0.4445501 -2.724896e-01
##  [55,] 0.4455586  2.613167e-01
##  [56,] 0.4437255  4.570230e-02
##  [57,] 0.4454198  1.272323e-01
##  [58,] 0.4446474 -2.199982e-01
##  [59,] 0.4451090  1.856587e-01
##  [60,] 0.4447558 -4.672931e-01
##  [61,] 0.4450244 -6.391749e-01
##  [62,] 0.4435396 -4.411997e-01
##  [63,] 0.4447803  6.236015e-01
##  [64,] 0.4446180  1.084722e+00
##  [65,] 0.4440285  2.115564e-01
##  [66,] 0.4444450 -1.222715e-02
##  [67,] 0.4438297  5.656028e-01
##  [68,] 0.4459305  5.946181e-01
##  [69,] 0.4447797  5.895556e-01
##  [70,] 0.4447026  0.000000e+00
##  [71,] 0.4428735 -4.899441e-01
##  [72,] 0.4439647 -1.236583e-01
##  [73,] 0.4439000  0.000000e+00
```

```
##  [74,] 0.4451698  1.352388e+00
##  [75,] 0.4446660 -3.120750e-02
##  [76,] 0.4436368  1.571297e-02
##  [77,] 0.4446742  1.827877e-01
##  [78,] 0.4452348 -3.010568e-01
##  [79,] 0.4447959  3.890687e-01
##  [80,] 0.4449528  1.371341e-01
##  [81,] 0.4435295 -6.854966e-01
##  [82,] 0.4448536 -7.217965e-01
##  [83,] 0.4445905 -1.768085e-05
##  [84,] 0.4456539  3.580742e-01
##  [85,] 0.4444238 -1.177564e-02
##  [86,] 0.4448659  2.404362e-01
##  [87,] 0.4434468  1.002632e+00
##  [88,] 0.4443076 -3.222213e-01
##  [89,] 0.4449629 -2.202435e-01
##  [90,] 0.4447362  5.049019e-03
##  [91,] 0.4447868  4.020813e-01
##  [92,] 0.4444239 -3.282743e-01
##  [93,] 0.4444683  4.840310e-01
##  [94,] 0.4448108  6.322236e-02
##  [95,] 0.4447514  4.890172e-01
##  [96,] 0.4443271 -1.438234e-01
##  [97,] 0.4436432 -4.825047e-01
##  [98,] 0.4451149  3.873939e-01
##  [99,] 0.4448205  4.097204e-01
## [100,] 0.4443287  1.553023e-01
## [101,] 0.4448450  4.241922e-01
## [102,] 0.4442690 -2.808934e-01
## [103,] 0.4446662 -3.089772e-01
## [104,] 0.4443123  3.507374e-02
## [105,] 0.4445250  2.919625e-02
## [106,] 0.4446878  7.158391e-01
## [107,] 0.4444553 -6.047811e-02
## [108,] 0.4451417  5.754938e-01
## [109,] 0.4425166  3.938460e-02
## [110,] 0.4447071  4.881280e-01
## [111,] 0.4449459 -6.205303e-03
## [112,] 0.4450345 -7.931974e-01
## [113,] 0.4447008  1.162201e+00
## [114,] 0.4449375 -2.643618e-01
## [115,] 0.4448751  3.890629e-01
##
## $value
## [1] 0.01916233
##
## $counts
## function gradient
##      174      100
##
## $convergence
## [1] 1
##
## $message
```

```
## NULL
##
## $score
## [1] NaN
```