A guide to use the Polytect: an automatic clustering and labeling method for multi-color digital PCR data

Yao Chen

Introduction

Digital PCR (dPCR) is a state-of-the-art quantification method of target nucleic acids. The technology is based on massive partitioning of a reaction mixture into individual PCR reactions. This results in partition-level end-point fluorescence intensities that are subsequently used to classify partitions as positive or negative, i.e., containing or not containing the target nucleic acid(s). Many automatic dPCR partition classification methods have been proposed, but they are limited to the analysis of single or dual color dPCR data. While general-purpose or flow cytometry clustering methods can be directly applied to multi-color dPCR data, these methods do not exploit the approximate prior knowledge on cluster center locations available in dPCR data.

We developed a novel method, `Polytect`, that relies on crude cluster results from flowPeaks, previously shown to offer good partition classification performance, and subsequently refines flowPeaks' results by cluster merging and automatic cluster labeling, exploiting the prior knowledge on cluster center locations.

Once the package is accepted and added to Bioconductor, you will be able to install it with `BiocManager`:

```
if (!requireNamespace("BiocManager", quietly = TRUE))
    install.packages("BiocManager")
BiocManager::install("Polytect")
```

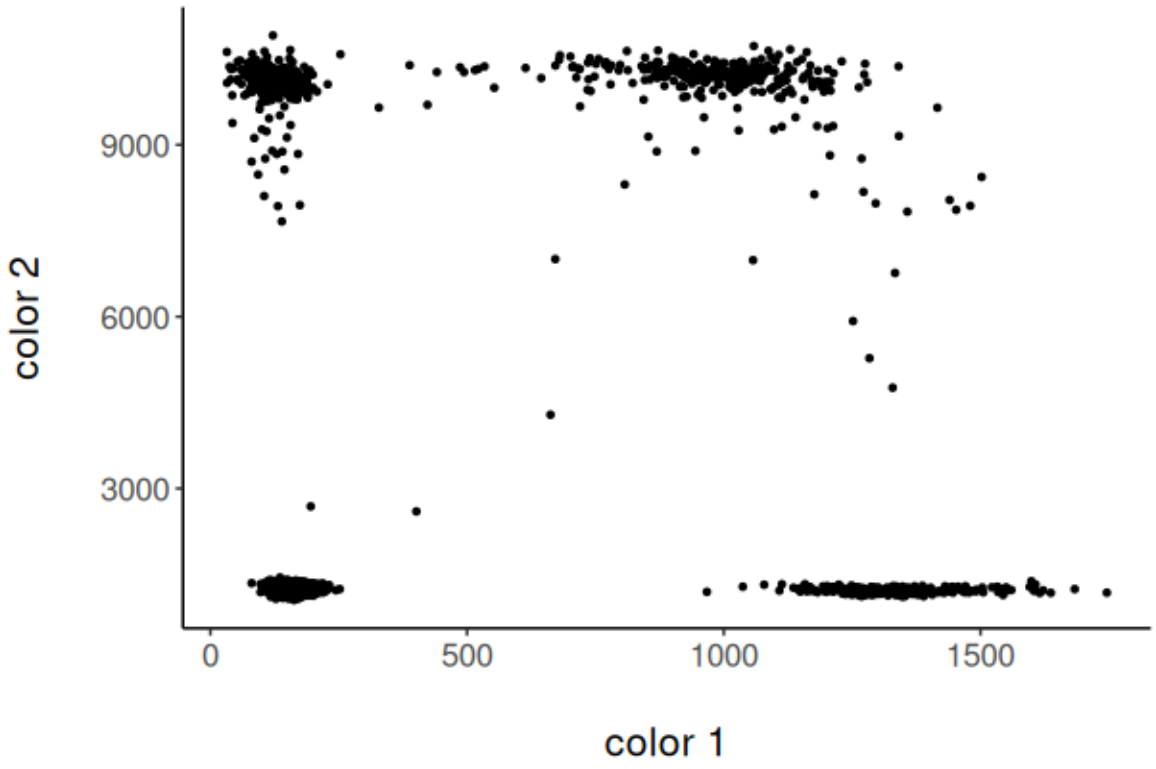
Examples

To demonstrate the core functions of this package, we will use the HR digital PCR dataset, which can be accessed using the command data(HR). The HR dataset is a data frame consisting of 18,233 rows and 2 columns. The dataset has 4 clusters. The clustering analysis can be performed using the following commands.

```
library(Polytect)
library(ggplot2)
library(flowPeaks)
library(ddPCRclust)

data(HR)
head(HR)
#>   channel1  channel2
#> 1 32.25462 10622.230
#> 2 32.55316 10081.106
#> 3 37.41653 10363.010
#> 4 39.65380 10140.198
#> 5 40.31481 10319.560
#> 6 42.58688  9864.532
ggplot(data=HR, aes(channel1, channel2))+
    geom_point(size=0.9,show.legend = FALSE) +
    labs(x = "color 1", y = "color 2") +
    theme(text = element_text(size = 15),
```

```
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_line(colour = "black"),
        plot.margin = margin(t = 20, r = 20, b = 20, l = 20),
        axis.title.x = element_text(margin = margin(t = 20)),
        axis.title.y = element_text(margin = margin(r = 20)))
```

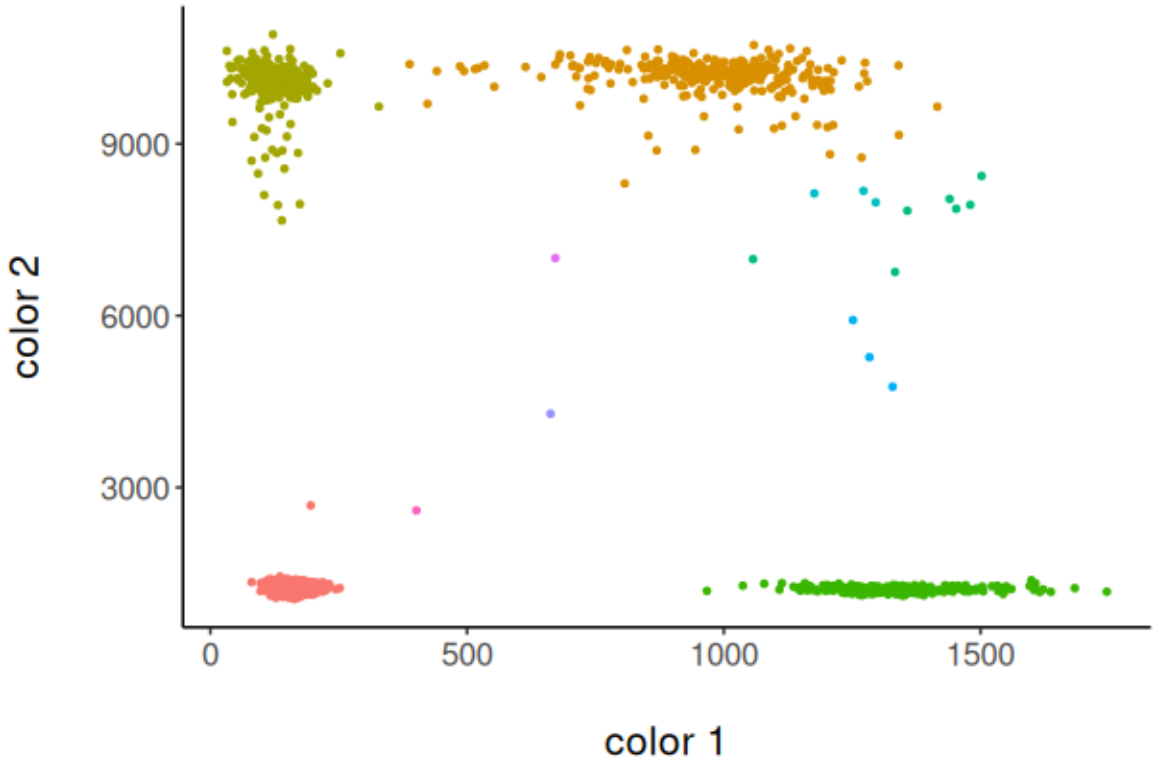


If we perform flowPeaks only, there will be more clusters than expected.

```
data_scaled<-apply(HR,2,function(x) (x-min(x))/(max(x)-min(x)))
data_input<-as.matrix(data_scaled)
fp<-flowPeaks(data_input)
#>         step 0, set the intial seeds, tot.wss=0.0767292
#>         step 1, do the rough EM, tot.wss=0.0519279 at 0.122556 sec
#>         step 2, do the fine transfer of Hartigan-Wong Algorithm
#>                  tot.wss=0.0483346 at 0.259323 sec

ggplot(data=HR, aes(channel1, channel2,colour = factor(fp$peaks.cluster)))+
    geom_point(size=0.9,show.legend = FALSE) +
    labs(x = "color 1", y = "color 2") +
    theme(text = element_text(size = 15),
          panel.grid.major = element_blank(),
          panel.grid.minor = element_blank(),
          panel.background = element_blank(),
          axis.line = element_line(colour = "black"),
          plot.margin = margin(t = 20, r = 20, b = 20, l = 20),
```

```
        axis.title.x = element_text(margin = margin(t = 20)),
        axis.title.y = element_text(margin = margin(r = 20)))
```

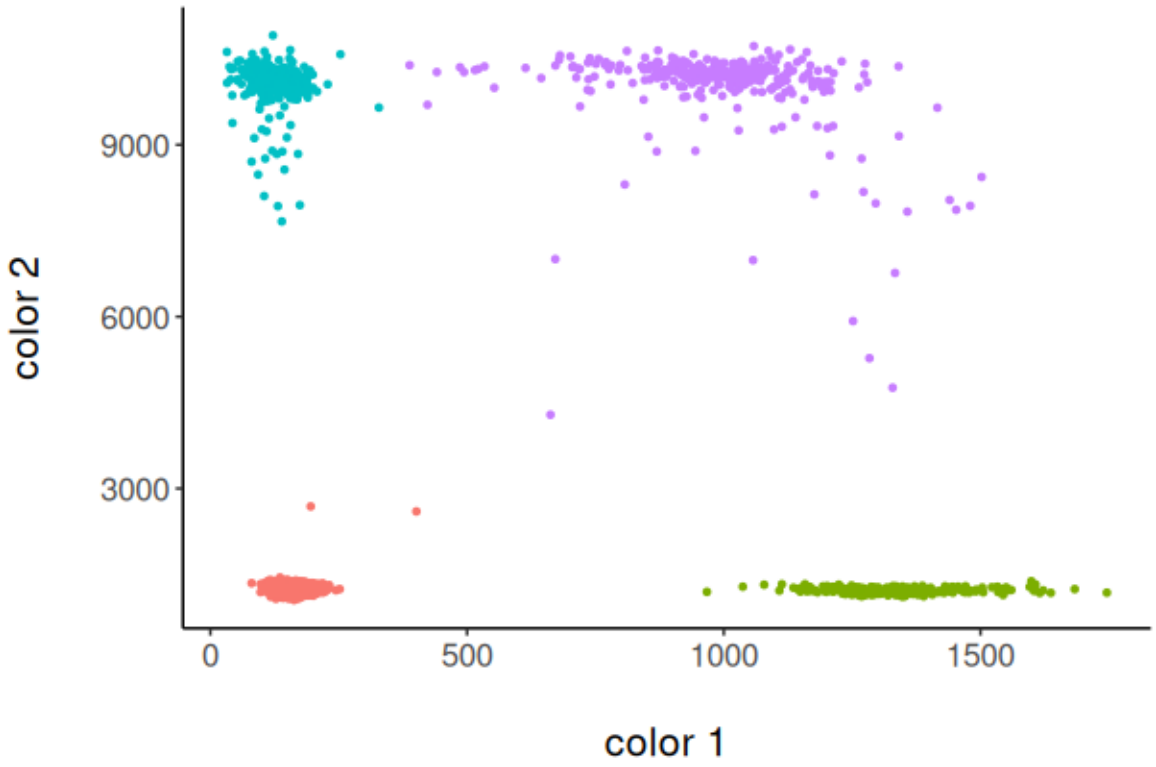


The main function that performs flowPeaks first, then merges the excess clusters.

```
result<-polytect_clust(data=HR,cluster_num=4)
#>          step 0, set the intial seeds, tot.wss=0.0767292
#>          step 1, do the rough EM, tot.wss=0.0519279 at 0.111042 sec
#>          step 2, do the fine transfer of Hartigan-Wong Algorithm
#>                  tot.wss=0.0483346 at 0.252386 sec
print(head(result))
#>   channel1  channel2 cluster
#> 1 32.25462 10622.230       3
#> 2 32.55316 10081.106       3
#> 3 37.41653 10363.010       3
#> 4 39.65380 10140.198       3
#> 5 40.31481 10319.560       3
#> 6 42.58688  9864.532       3

ggplot(data=HR, aes(channel1, channel2,colour = factor(result$cluster)))+
    geom_point(size=0.9,show.legend = FALSE) +
    labs(x = "color 1", y = "color 2") +
    theme(text = element_text(size = 15),
          panel.grid.major = element_blank(),
          panel.grid.minor = element_blank(),
          panel.background = element_blank(),
```

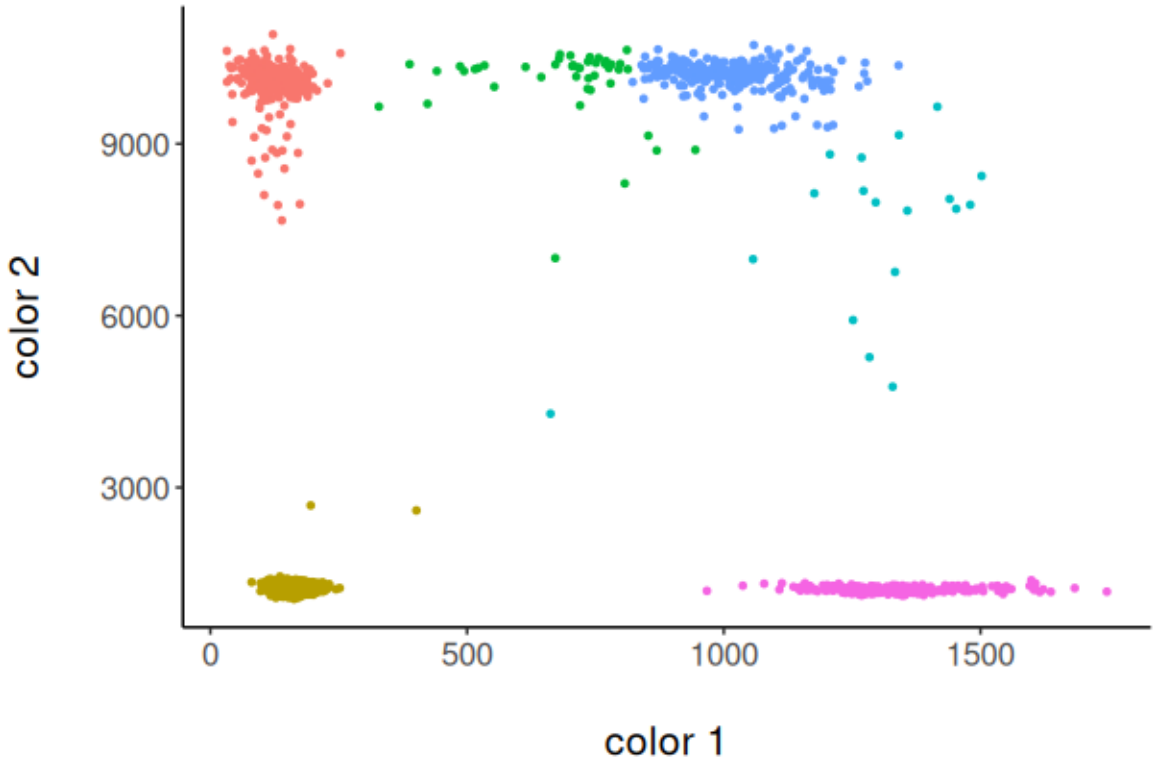
```
        axis.line = element_line(colour = "black"),
        plot.margin = margin(t = 20, r = 20, b = 20, l = 20),
        axis.title.x = element_text(margin = margin(t = 20)),
        axis.title.y = element_text(margin = margin(r = 20)))
```



Or you can use any initial clustering results as an input to the polytect_merge function

```
## it is advised to standardize the data
dist_matrix <- dist(data_input)
hc <- hclust(dist_matrix, method = "ward.D2")
# the number of clusters is specified at 6, which is larger than 4
hc_clusters <- cutree(hc, k = 6)

ggplot(data=HR, aes(channel1, channel2,colour = factor(hc_clusters)))+
    geom_point(size=0.9,show.legend = FALSE) +
    labs(x = "color 1", y = "color 2") +
    theme(text = element_text(size = 15),
          panel.grid.major = element_blank(),
          panel.grid.minor = element_blank(),
          panel.background = element_blank(),
          axis.line = element_line(colour = "black"),
          plot.margin = margin(t = 20,  r = 20, b = 20, l = 20),
          axis.title.x = element_text(margin = margin(t = 20)),
          axis.title.y = element_text(margin = margin(r = 20)))
```

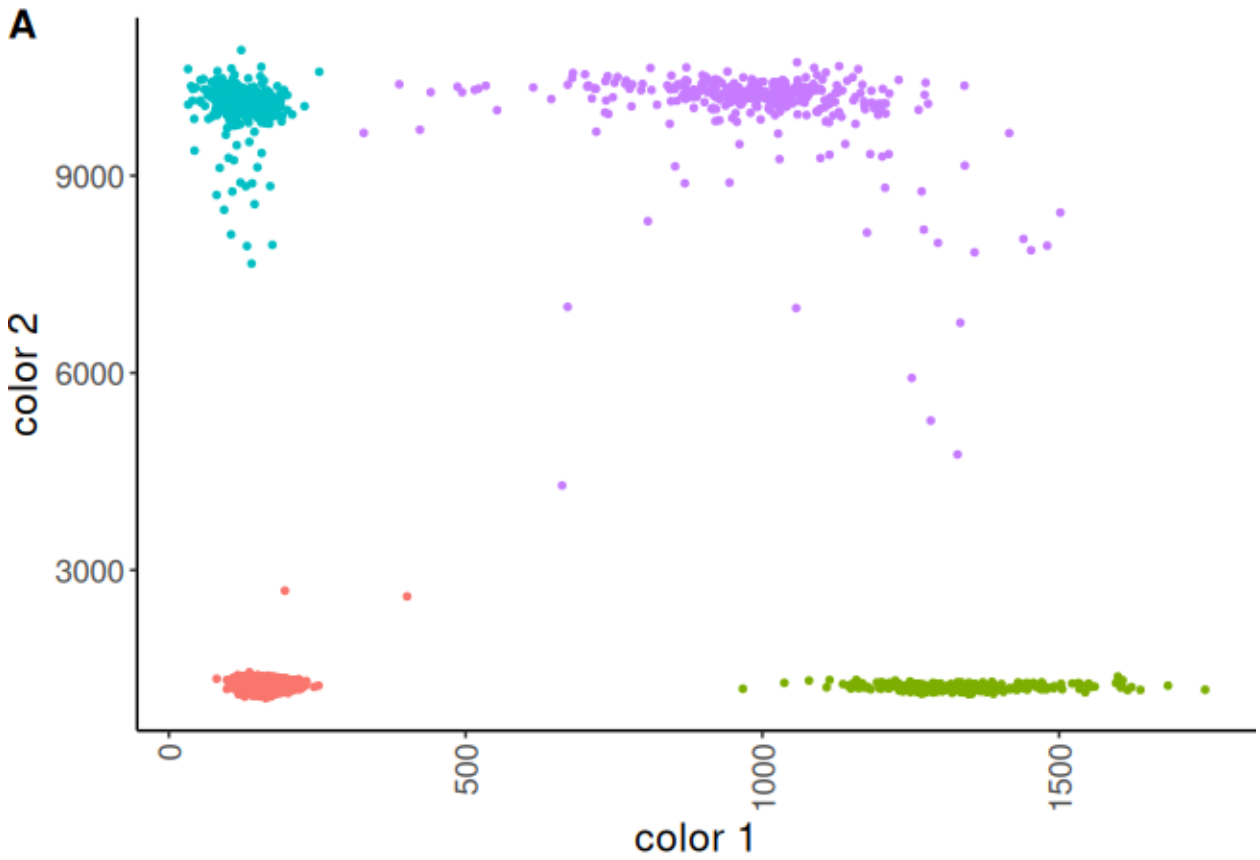



```
hc_parse<-list()
hc_parse$cluster<-hc_clusters

result<-polytect_merge(data=HR,cluster_num=4,base_clust=hc_parse)
print(head(result))
#>   channel1  channel2 cluster
#> 1 32.25462 10622.230       3
#> 2 32.55316 10081.106       3
#> 3 37.41653 10363.010       3
#> 4 39.65380 10140.198       3
#> 5 40.31481 10319.560       3
#> 6 42.58688  9864.532       3
```

The clustering results can be visualized by 2-d plots.

```
polytect_plot(result,cluster_num=4)
```

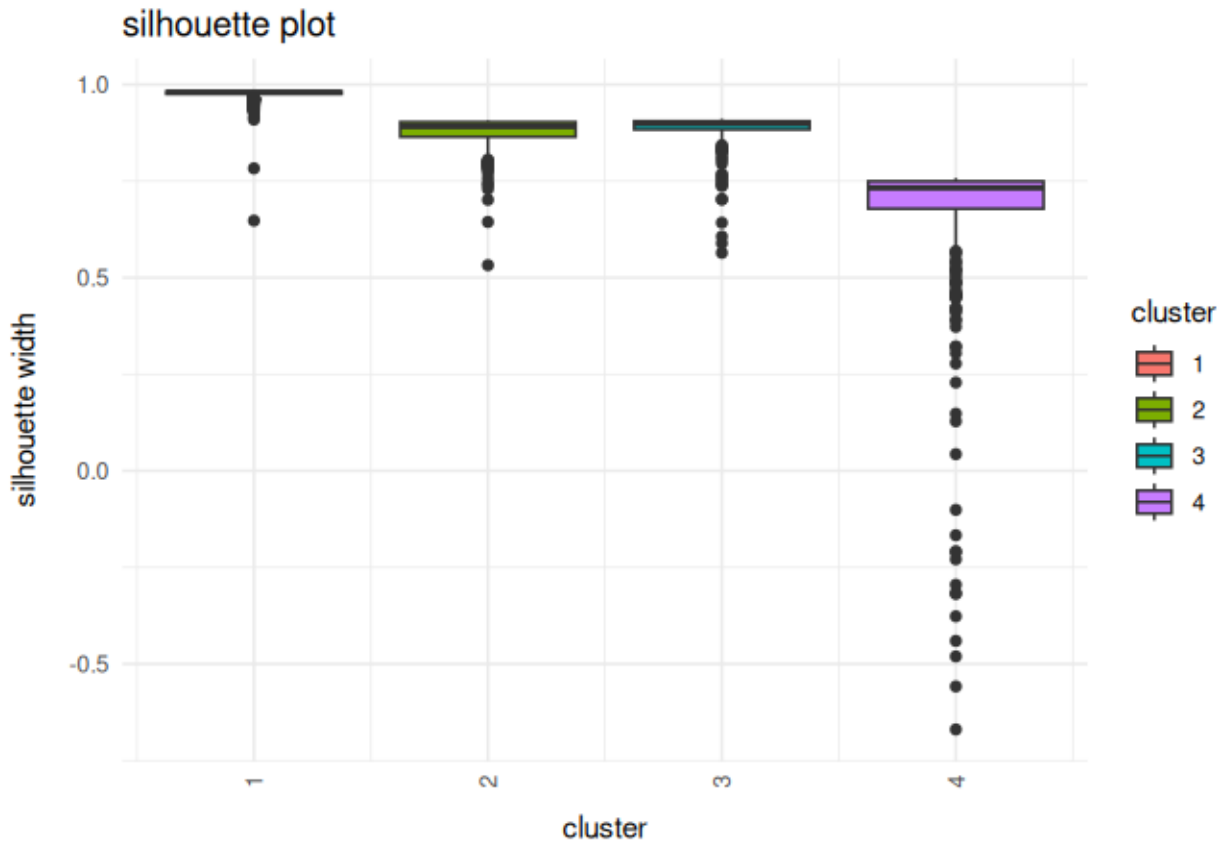


You can also summarise the results, which will give you cluster centers, group sizes and silhouette coefficients for each group

```
result_summary<-polytect_summary(result)
print(result_summary)
#>   cluster mean_channel1 mean_channel2 cluster_size silhouette_coef
#> 1       1      155.2730      1249.602        17342            0.98
#> 2       2     1330.4427      1216.940          259            0.87
#> 3       3      122.9398     10046.944          291            0.88
#> 4       4      988.5556     10044.658          341            0.65
```

You can also plot the individual silhouette coefficient in each cluster

```
sil_plot(result)
```

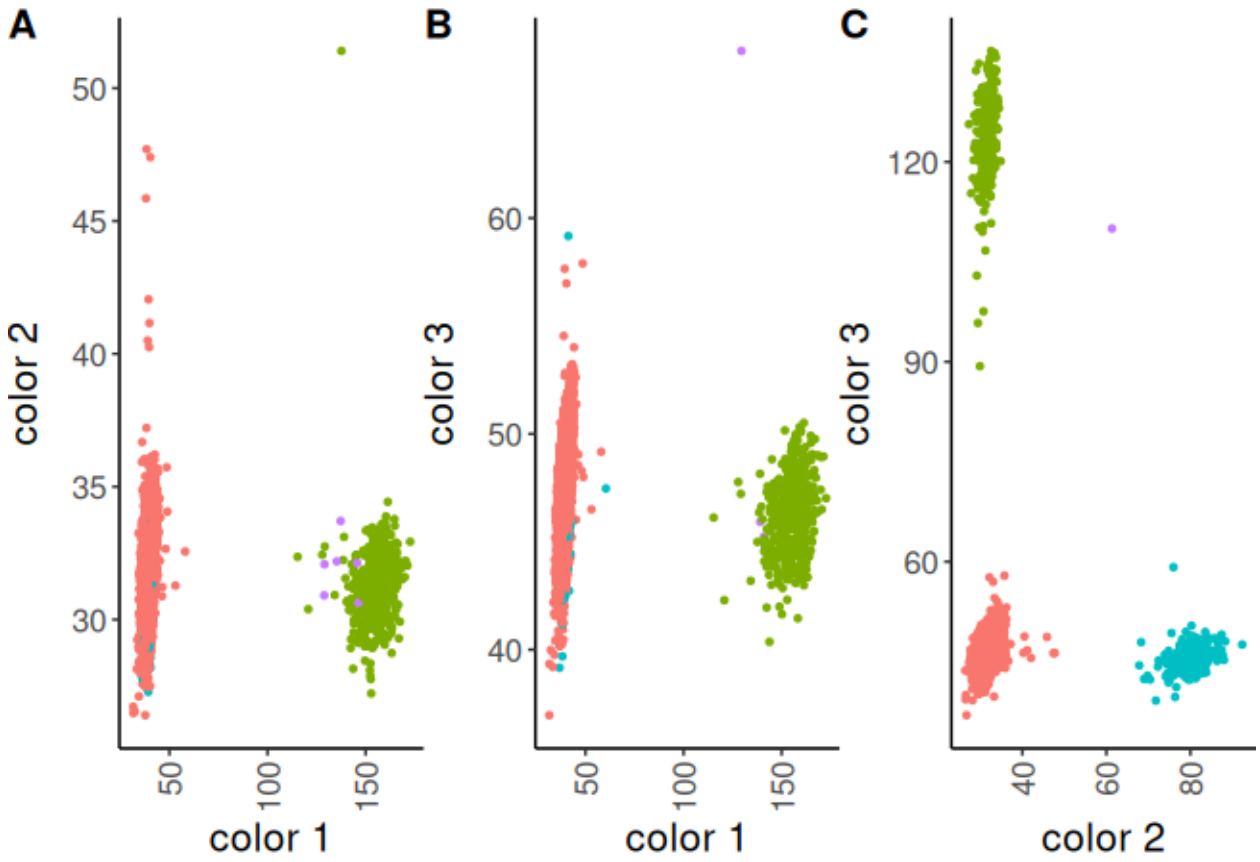


There is a function to calculate the concentration of the targets.

```
target_conc<-conc_cal(result,cluster_num=4,sampvol=0.91,volmix=20,voltemp=20)
print(target_conc)
#>   target concentration
#> 1      1      36.77032
#> 2      2      38.76640
```

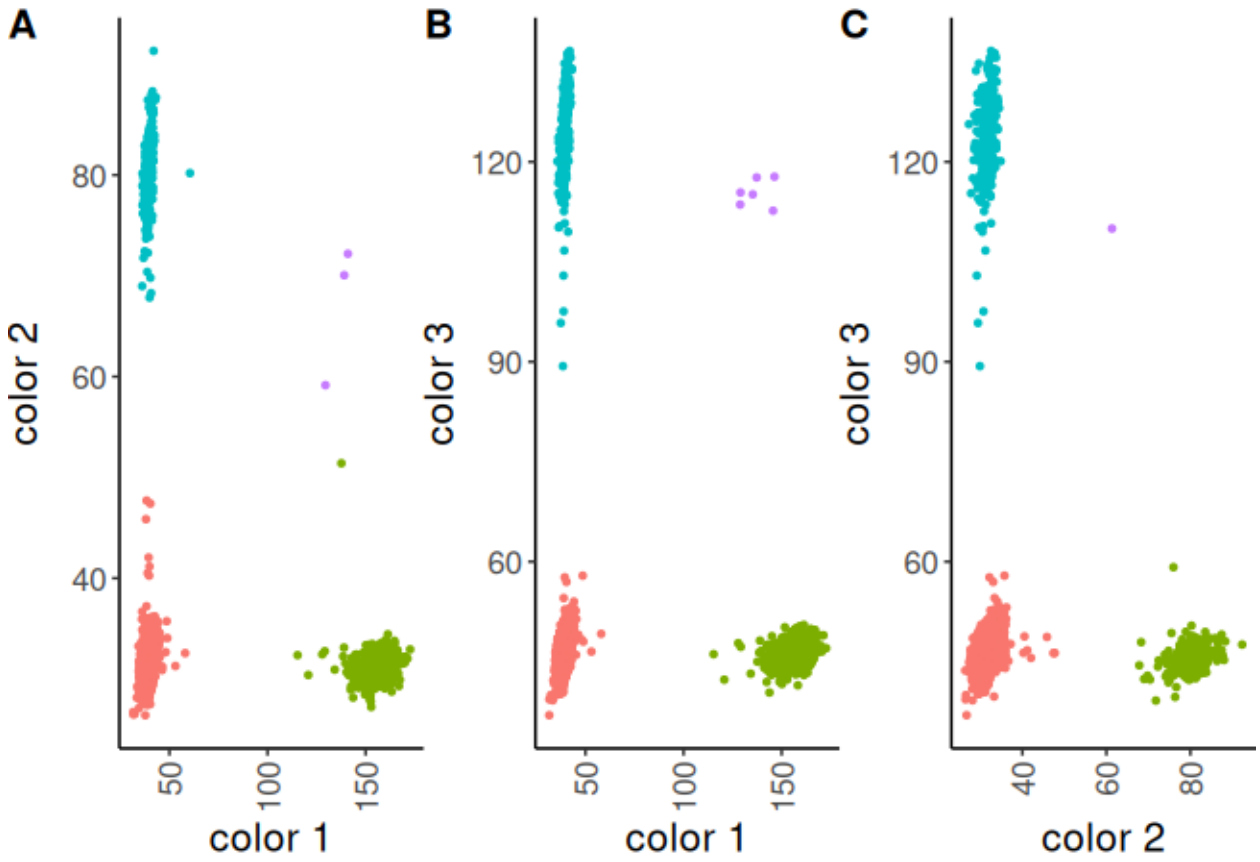
This package can also handle 3-up to 6-color dPCR data. We first perform flowPeaks only.

```
data(BPV)
data_scaled<-apply(BPV,2,function(x) (x-min(x))/(max(x)-min(x)))
data_input<-as.matrix(data_scaled)
fp<-flowPeaks(data_input)
#>          step 0, set the intial seeds, tot.wss=1.00728
#>          step 1, do the rough EM, tot.wss=0.721981 at 0.368387 sec
#>          step 2, do the fine transfer of Hartigan-Wong Algorithm
#>                  tot.wss=0.708295 at 0.645131 sec
table(fp$peaks.cluster)
#>
#>     1     2     3     4     5     6     7
#> 24401   482   323   245     6     3     1
df_data<-as.data.frame(cbind(BPV,cluster=fp$peaks.cluster))
polytect_plot(df_data,cluster_num=8)
```



Then the main function.

```
result<-polytect_clust(data=BPV,cluster_num=8)
#>          step 0, set the intial seeds, tot.wss=1.00728
#>          step 1, do the rough EM, tot.wss=0.721981 at 0.372294 sec
#>          step 2, do the fine transfer of Hartigan-Wong Algorithm
#>                    tot.wss=0.708295 at 0.648927 sec
table(result$cluster)
#>
#>     1     2     3     4     5     6     7
#> 24401   482   245   323     3     6     1
polytect_plot(result,cluster_num = 8)
```



Polytect does not change the clustering result of flowPeaks. What it did is relabeling.

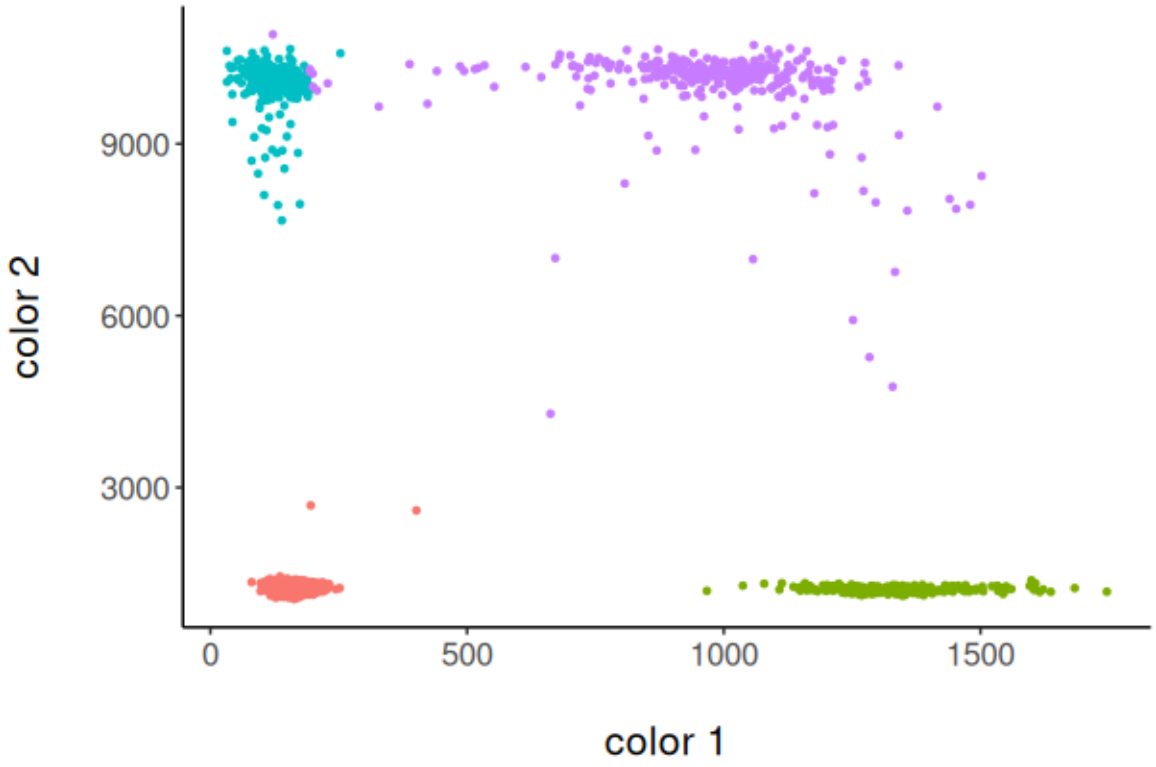
Comparison to the existing ddpcr package ddpcrclust

```
exampleFiles <- list.files(paste0(find.package('ddPCRclust'), '/extdata'), full.names = TRUE)
template <- readTemplate(exampleFiles[9])
template$template[1,3]<-2
template$template[1,c(6,7)]<-''
data_list<-list()
data_list$ids<-'B01'
data_list$data$B01<-data.frame(Ch1.Amplitude=HR[,1],Ch2.Amplitude=HR[,2])

result <- ddPCRclust(data_list,template = template)
#>         step 0, set the intial seeds, tot.wss=1.22273e+06
#>         step 1, do the rough EM, tot.wss=848836 at 0.073833 sec
#>         step 2, do the fine transfer of Hartigan-Wong Algorithm
#>                 tot.wss=804148 at 0.185742 sec

ggplot(data=HR,
       aes(channel1, channel2,colour = factor(result$B01$data$Cluster)))+
    geom_point(size=0.9,show.legend = FALSE) +
    labs(x = "color 1", y = "color 2") +
    theme(text = element_text(size = 15),
          panel.grid.major = element_blank(),
          panel.grid.minor = element_blank(),
          panel.background = element_blank(),
          axis.line = element_line(colour = "black"),
          plot.margin = margin(t = 20, r = 20, b = 20, l = 20),
```

```
        axis.title.x = element_text(margin = margin(t = 20)),
        axis.title.y = element_text(margin = margin(r = 20)))
```



Session Information

The following information was obtained from the R session that generated this vignette:

```
sessionInfo()
#> R version 4.5.0 beta (2025-04-02 r88102)
#> Platform: x86_64-pc-linux-gnu
#> Running under: Ubuntu 24.04.2 LTS
#>
#> Matrix products: default
#> BLAS:   /home/biocbuild/bbs-3.22-bioc/R/lib/libRblas.so
#> LAPACK: /usr/lib/x86_64-linux-gnu/lapack/liblapack.so.3.12.0  LAPACK version 3.12.0
#>
#> locale:
#>  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
#>  [3] LC_TIME=en_GB              LC_COLLATE=C
#>  [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
#>  [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
#>  [9] LC_ADDRESS=C               LC_TELEPHONE=C
#> [11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C
#>
#> time zone: America/New_York
#> tzcode source: system (glibc)
#>
```

```
#> attached base packages:
#> [1] stats     graphics  grDevices utils     datasets  methods   base
#>
#> other attached packages:
#> [1] ddPCRclust_1.29.0 flowPeaks_1.55.0  ggplot2_3.5.2     Polytect_1.1.0
#>
#> loaded via a namespace (and not attached):
#>  [1] tidyselect_1.2.1    farver_2.1.2        dplyr_1.1.4
#>  [4] R.utils_2.13.0      bitops_1.0-9        fastmap_1.2.0
#>  [7] ParamHelpers_1.14.2 digest_0.6.37       lifecycle_1.0.4
#> [10] cluster_2.1.8.1     survival_3.8-3      parallelMap_1.5.1
#> [13] magrittr_2.0.3      smoof_1.6.0.3       compiler_4.5.0
#> [16] rlang_1.1.6         tools_4.5.0         plotrix_3.8-4
#> [19] yaml_2.3.10         data.table_1.17.0   knitr_1.50
#> [22] sn_2.1.1            labeling_0.4.3      interp_1.1-6
#> [25] mnormt_2.1.1        RColorBrewer_1.1-3  mlrMBO_1.1.5.1
#> [28] abind_1.4-8         KernSmooth_2.23-26  withr_3.0.2
#> [31] RProtoBufLib_2.21.0 numDeriv_2016.8-1.1 R.oo_1.27.0
#> [34] BiocGenerics_0.55.0 grid_4.5.0          polyclip_1.10-7
#> [37] rgenoud_5.9-0.11    stats4_4.5.0        latticeExtra_0.6-30
#> [40] mlr_2.19.2          caTools_1.18.3      SamSPECTRAL_1.63.0
#> [43] colorspace_2.1-1    MASS_7.3-65         scales_1.3.0
#> [46] gtools_3.9.5        BBmisc_1.13         cli_3.6.4
#> [49] mvtnorm_1.3-3       rmarkdown_2.29      generics_0.1.3
#> [52] IDPmisc_1.1.21      flowCore_2.21.0     splines_4.5.0
#> [55] parallel_4.5.0      BiocManager_1.30.25 matrixStats_1.5.0
#> [58] vctrs_0.6.5         Matrix_1.7-3        carData_3.0-5
#> [61] cytolib_2.21.0      car_3.1-3           S4Vectors_0.47.0
#> [64] Formula_1.2-5       clue_0.3-66         jpeg_0.1-11
#> [67] DiceKriging_1.6.0   flowDensity_1.43.0  hexbin_1.28.5
#> [70] glue_1.8.0          cowplot_1.1.3       stringi_1.8.7
#> [73] gtable_0.3.6        deldir_2.0-4        munsell_0.5.1
#> [76] tibble_3.2.1        pillar_1.10.2       htmltools_0.5.8.1
#> [79] gplots_3.2.0        R6_2.6.1            lhs_1.2.0
#> [82] evaluate_1.0.3      tidyverse_2.0.0     lattice_0.22-7
#> [85] Biobase_2.69.0      R.methodsS3_1.8.2   png_0.1-8
#> [88] backports_1.5.0     flowViz_1.73.0      Rcpp_1.0.14
#> [91] fastmatch_1.1-6     checkmate_2.3.2     xfun_0.52
#> [94] pkgconfig_2.0.3
```

References

Ge Y, Sealfon S (2012). "flowPeaks: a fast unsupervised clustering for flow cytometry data via K-means and density peak finding." Bioinformatics. R package version 4.4.0.

Lun A (2024). bluster: Clustering Algorithms for Bioconductor. R package version 1.14.0.