CATALYST: Cytometry dATa anALYSis Tools

Helena L Crowell[*1] ***and Mark D Robinson***[†1]

[1]Institute of Molecular Life Sciences, University of Zurich, Switzerland

[*]crowellh@student.ethz.ch [†]mark.robinson@imls.uzh.ch

8 December 2016

Abstract

By addressing the limit of measurable fluorescent parameters due to instrumentation and spectral overlap, mass cytometry (CyTOF) combines heavy metal spectrometry to allow examination of up to 100 parameters at the single cell level. While spectral overlap is significantly less pronounced in CyTOF than flow cytometry, spillover due to detection sensitivity, isotopic impurities, and oxide formation can impede data interpretability. We designed CATALYST (Cytometry dATa anALYSis Tools) to provide tools for preprocessing and analysis of cytometry data, including compensation and in particular, an improved implementation of the single-cell deconvolution algorithm (Zunder et al. 2015, Nature Protocols 10, 316-333).

Package

CATALYST 0.1.3

Contents

1 Introduction

`CATALYST` performs compensation via a two-step approach comprising:

i. identification of single positive populations via single-cell debarcoding (SCD) of single-stained beads (or cells); and,
ii. estimation of a spillover matrix (SM) from the populations identified, followed by compensation via multiplication of measurement intensities by its inverse, the compensation matrix (CM).

As shown in [REF], we can model spillover linearly, with the channel stained for as predictor, and spill-effected channels as response. Thus, the intensity observed in a given channel j are a linear combination of its real signal and contributions of other channels that spill into it. Let s_ij denote the proportion of channel j signal that is due to channel i, and w_j the set of channels that spill into channel j. Then

$$I_{j,observed} = I_{j,real} + \sum_{i \in w_j} s_{ij}$$

In matrix notation, measurement intensities may be viewed as the convolution of real intensities and a spillover matrix with dimensions number of events times number of measurement parameters

$$I_{observed} = I_{real} \cdot SM$$

Therefore, we can estimate the real signal, I_{real}, as:

$$I_{real} = I_{observed} \cdot SM^{-1} = I_{observed} \cdot CM$$

where SM^{-1} is termed compensation matrix (CM).

Because any signal not in a single stain experiment's primary channel j results from channel crosstalk, each spill entry s_{ij} can be approximated by the slope of a linear regression with channel j signal as the response, and channel i signals as the predictors, where $i \in w_j$. To facilitate robust estimates, we calculate this as the slope of a line through the medians (or trimmed means) of stained and unstained populations, m_j^+ and m_i^+, respectively. The medians (or trimmed means) computed from events that are i) negative in the respective channels; and, ii) not assigned to interacting channels; and, iii) not unassigned, m_j^- and m_i^-, respectively, are subtracted as to account for background according to:

$$s_{ij} = \frac{m_j^+ - m_j^-}{m_i^+ - m_i^-}$$

On the basis of their additive nature, spill values are estimated independently for every pair of interacting channels. The current framework exclusively takes account of interactions that are sensible from a chemical and physical point of view. As reasoned before, +/-1M channels (abundance sensitivity), the +16M channel (oxide formation) and channels measuring isotopes (impurities) are taken into consideration. The SM's diagonal entries sii are set to 1 so that spill is relative to the total signal measured in a given channel.

2 Data example

As an examplary data set, we provide a flowFrame with 10000 cells and 61 observables, obtained from a 36ab-panel single-staining experiment. Beads were stained for antibodies captured by mass channels 139, 141 through 156, and 158 through 176, respectively, and pooled together. Note that, to demonstrate the debarcoding and compensation work-flow, we sampled 10'000 of all recorded events at the cost of not necessarily arriving at biologically meaningful results.

```
library(CATALYST)
## Warning: replacing previous import 'flowCore::plot' by
## 'graphics::plot' when loading 'CATALYST'
data(ss_beads)
ss_beads[, c(14, 16:31, 33:51)]
## flowFrame object '1.fcs'
## with 10000 cells and 36 observables:
##          name          desc range minRange maxRange
## $P14 La139Di     139La-CD8  2143        0     2142
## $P16 Pr141Di    141Pr-CD64  1292        0     1291
## $P17 Nd142Di    142Nd-CD23  2736        0     2735
## $P18 Nd143Di    143Nd-CD68  3469        0     3468
## $P19 Nd144Di    144Nd-CD36  4803        0     4802
## $P20 Nd145Di     145Nd-CD4  4434        0     4433
## $P21 Nd146Di    146Nd-CD68  7094        0     7093
## $P22 Sm147Di     147Sm-CD3 11467        0    11466
## $P23 Nd148Di    148Nd-CD20  5987        0     5986
## $P24 Sm149Di    149Sm-CD20  8702        0     8701
## $P25 Nd150Di    150Nd-CD68  4200        0     4199
## $P26 Eu151Di   151Eu-CD123  5372        0     5371
## $P27 Sm152Di    152Sm-CD99 13228        0    13227
## $P28 Eu153Di    153Eu-CD68  5381        0     5380
## $P29 Sm154Di    154Sm-CD15  8304        0     8303
## $P30 Gd155Di   155Gd-CD273 14696        0    14695
## $P31 Gd156Di    156Gd-CD93  7180        0     7179
## $P33 Gd158Di    158Gd-CD15  5196        0     5195
## $P34 Tb159Di   159Tb-CD192  7482        0     7481
## $P35 Gd160Di    160Gd-CD45  6003        0     6002
## $P36 Dy161Di 161Dy-CD66ace 11454        0    11453
## $P37 Dy162Di   162Dy-CXCR4 13035        0    13034
## $P38 Dy163Di    163Dy-CD22  5650        0     5649
## $P39 Dy164Di     164Dy-CD7  8496        0     8495
## $P40 Ho165Di     165Ho-CD4  5977        0     5976
## $P41 Er166Di    166Er-CD32  9511        0     9510
## $P42 Er167Di    167Er-CD16 15775        0    15774
## $P43 Er168Di    168Er-CD14  2763        0     2762
## $P44 Tm169Di    169Tm-CD99  6273        0     6272
## $P45 Er170Di     170Er-CD7  3787        0     3786
## $P46 Yb171Di  171Yb-HLA.DR  7678        0     7677
## $P47 Yb172Di 172Yb-HLA.ABC 15478        0    15477
## $P48 Yb173Di     173Yb-CD3  6348        0     6347
## $P49 Yb174Di    174Yb-CD8b  6092        0     6091
## $P50 Lu175Di  175Lu-HLA.DR 10440        0    10439
## $P51 Yb176Di    176Yb-CD45  8924        0     8923
## 323 keywords are stored in the 'description' slot
```

3 The `dbFrame` class

Data returned and used throughout the debarcoding process are stored in a debarcoding frame. Event information, stored in a matrix, is passed from the `flowFrame` specified in `assignedPrelim` to the `exprs` slot, and is accessible via `exprs` as before. The `bc_key` slot is a binary matrix with numeric masses as column names and barcode IDs as row names. If supplied with a numeric vector of masses, `assignPrelim` will generate a concurrent representation.

```
bc_ms <- c(139, 141:156, 158:176)
re <- assignPrelim(x = ss_beads, y = bc_ms, verbose = FALSE)
bc_key(re)[1:5, 1:15]
##     139 141 142 143 144 145 146 147 148 149 150 151 152 153 154
## 139   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0
## 141   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0
## 142   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0
## 143   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0
## 144   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0
```

`bc_ids` is a numeric vector of the ID assignments that have been made. If a given event's population separation falls below its separation cutoff, or above the population's Mahalanobis distance cutoff, it will be give ID 0 for *"unassigned"*. Assignments may be manipulated through standard replacement via `bc_ids<-`. The `deltas` slot contains for each event the separations between positive and nergative populations, that is, between the lowest positive and highest negative intesity. `normed_bcs` are the barcode intensities normalized by population. Here, each event is scaled to the 95% quantile of the population it's been assigned to.

An overview of the object's dimensionality, current event assignments, deconvolution parameters, and yields achieved upon debarcoding is given by `show`.

```
re
## dbFrame objectect with
##  10000 events, 61 observables and 36 barcodes:
##
## Current assignments:
##       1 events unassigned
## ID    147 162 165 151 173 152 144 176 146 155 169 164 171 174 161
## Count 358 347 344 342 339 336 318 318 311 309 305 303 298 298 295
##
## ID    172 143 149 166 145 148 167 159 175 153 150 160 158 170 156
## Count 293 292 291 288 287 287 285 281 281 278 273 269 253 237 217
##
## ID    163 141 154 142 139 168
## Count 217 215 198 168 152 116
```

4 Deconvolution work-flow

CATALYST provides three functions for debarcoding and two visualizations that guide selection of thresholds and give a sense of barcode assignment quality.

In summary, events are assigned to a sample when i) their positive and negative barcode populations are separated by a distance larger than a threshold value and ii) the combination of their positive barcode channels appears in the barcoding scheme.

4.1 `assignPrelim`: Assignment of preliminary barcode IDs

The debarcoding step commences by assigning each event a preliminary barcode ID. `assignPrelim` thereby takes either a binary barcoding scheme or a vector of numeric masses as input, and accordingly assigns each event the appropirate row index or mass as ID. Depending on the `bc_key` supplied, there are three possible ways of proceeding:

- **Single-staining**:
  If a numeric vector of masses is supplied, the most intense channel is considered positive and its respective mass assigned as ID.

- **Doublet-filtering**:
  Given a binary barcoding scheme with a coherent number k of positive channels for all IDs, the k highest channels are considered positive and $n-k$ channels negative. Separation of positive and negative events equates to the difference between the kth highest and $(n-k)$th lowest intensity value.

- **Non-constant number of 1's**:
  Given an inconsistent number of 1's in the binary codes, the highest separation between consecutive barcodes is looked at.

In both, the doublet-filtering and the latter case, each event is assigned a binary code that, if matched with a code in the barcoding scheme supplied, dictates which row index will be assigned as ID. Cells whose positive barcodes are still very low or whose binary pattern of positive and negative barcodes doesn't occur in the barcoding scheme will be given ID 0 for *"unassigned"*.

FCS files are read into R with read.FCS of the *flowCore* package, and are represented as an object of class flowFrame. Provided with a flowFrame and a compatible barcoding scheme (barcode channel masses have to occur in the measurement data), assignPrelim will return a dbFrame containing exprs passed from the input flowFrame, a numeric vector of event assignments in slot bc_ids, separations between barcode populations on the normalized scale in slot deltas, and normalized barcode intensities in slot normed_bcs. Measurement intensities are normalized by population such that each is scaled to the 95% quantile for asinh transformed measurement intensities of events assigned to the respective barcode.

```
re <- assignPrelim(x = ss_beads, y = bc_ms, verbose = FALSE)
```

4.2 estCutoffs: Estimation of population-specific separatation cutoffs

Here, the choice of thresholds for the distance between negative and positive barcode populations is *i) automated* and *ii) independent for each barcode*. As opposed to a single and global cutoff parameter, estCutoffs will estimate a cutoff value that is specific for each sample to deal with barcode population cell yields that decline in an asynchronous fashion. The function will update the sep_cutoffs, counts and yields slots of the input dbFrame.

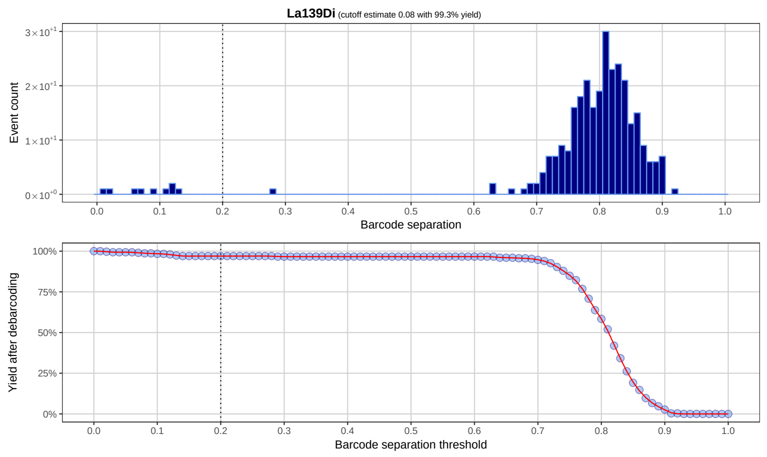
For the estimation of cutoff parameters we concider yields upon debarcoding as a function of the applied cutoffs. Commonly, this function will characterized by an initial weak decline, where doublets are excluded, and subsequent rapid decline in yields to zero. Inbetween, low numbers of counts with intermediate barcode separation give rise to a plateau. The separation cutoff value should be chosen such that it appropriately balances confidence in barcode assignment and cell yield. We thus fit the yields function, its first and second derivative, and compute the first turning point, marking the on-set of the plateu regime, as an adequte cutoff estimate.

```
# estimate separation cutoffs, and
# get counts and yields as a function of barcode separations
(re <- estCutoffs(x = re, verbose = FALSE))
## dbFrame objectect with
##  10000 events, 61 observables and 36 barcodes:
##
## Current assignments:
##       1 events unassigned
## ID    147 162 165 151 173 152 144 176 146 155 169 164 171 174 161
## Count 358 347 344 342 339 336 318 318 311 309 305 303 298 298 295
##
## ID    172 143 149 166 145 148 167 159 175 153 150 160 158 170 156
## Count 293 292 291 288 287 287 285 281 281 278 273 269 253 237 217
##
## ID    163 141 154 142 139 168
## Count 217 215 198 168 152 116
##
## Separation cutoffs:
## ID    139  141  142  143  144  145  146  147  148  149  150  151
## Yield 0.08 0.07 0.08 0.15 0.08 0.07 0.22 0.29 0.08 0.19 0.08 0.19
##
## ID    152  153  154  155  156  158  159  160  161  162  163  164
## Yield 0.14 0.13 0.23 0.30 0.14 0.18 0.22 0.23 0.41 0.13 0.10 0.27
##
## ID    165  166  167  168  169  170  171  172  173  174  175  176
## Yield 0.19 0.30 0.14 0.52 0.17 0.14 0.08 0.38 0.16 0.20 0.13 0.13
##
## Yields upon debarcoding:
##       95.23% overall yield
## ID    139    141    142   143    144    145    146    147
## Yield 99.34% 99.53% 99.4% 98.63% 98.43% 99.65% 97.75% 96.65%
##
## ID    148    149    150    151    152    153    154    155
## Yield 98.95% 98.28% 99.27% 95.61% 94.94% 96.76% 93.94% 94.5%
##
## ID    156    158    159    160    161    162    163    164
## Yield 94.93% 89.72% 92.17% 95.17% 88.81% 92.22% 96.31% 91.75%
##
## ID    165    166    167    168  169    170    171    172    173
## Yield 90.99% 91.32% 96.84% 100% 92.13% 95.78% 97.65% 78.84% 93.22%
##
## ID    174    175    176
## Yield 96.98% 95.37% 96.54%
```

4.3 plotYields: Distribution of barcode separations and cell yields upon debarcoding

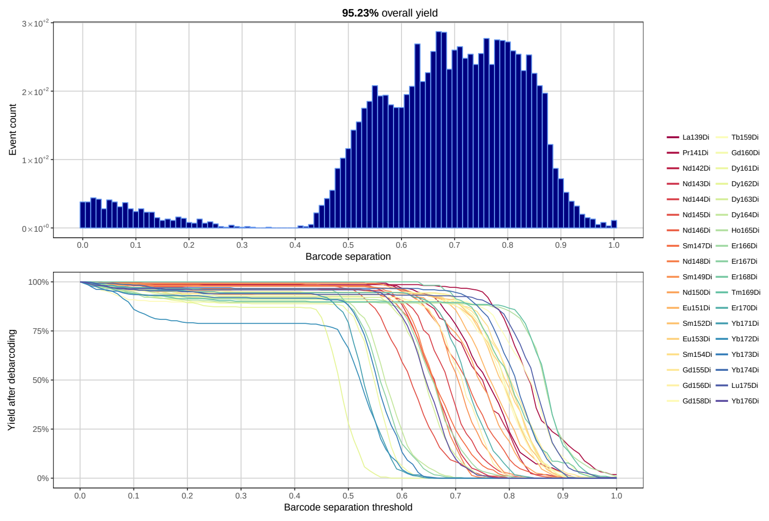
For each barcode, plotYields will generate a histogram of events separated by a given distance, as well as yields upon debarcoding as a function of separation cutoffs. The currently used separation cutoff as well as its resulting yield within the population is indicated in the plot's main title.

```
# generate yields plot for barcode 174
plotYields(x = re, which_bc = 174)
```



Option `which_bc` will render a summary plot of all barcodes. Here, the overall yield achieved by applying the current set of cutoff values will be shown. All yield functions should behave as described above: decline, stagnation, decline. Convergence to 0 yield at low cutoffs is a strong indicator that staining in this channel did not work, and excluding the channel entirely is sensible in this case. It is thus recommended to always view the all-barcodes yield plot to eliminate uninformative populations as a too small population size may cause difficulties, especially when computing spill estimates.

```
# generate summary yields plot
plotYields(x = re, which_bc = 0)
```

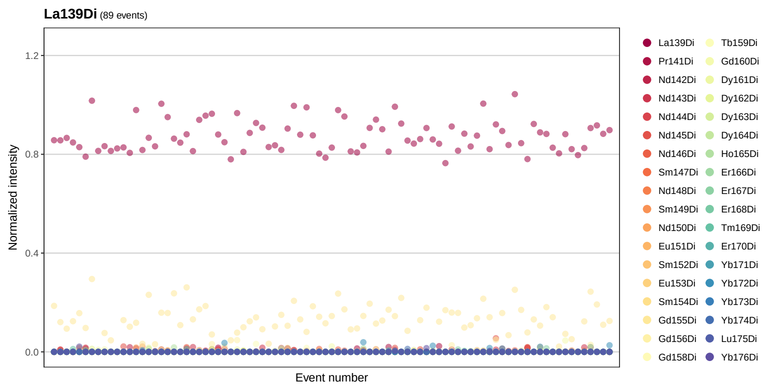


4.4 applyCutoffs: Applying separation and mahalanobis distance thresholds

```
# apply separation and Mahalanobis distance thresholds
(re <- applyCutoffs(x = re))
## dbFrame objectect with
##  10000 events, 61 observables and 36 barcodes:
##
## Current assignments:
##       3664 events unassigned
## ID    162 173 147 151 144 176 152 165 171 167 169 174 146 153 155
## Count 239 229 221 219 215 215 213 211 208 198 196 196 194 188 186
##
## ID    145 149 159 143 164 175 148 166 160 161 172 150 158 170 163
## Count 185 183 183 180 178 178 176 173 171 165 158 154 144 143 140
##
## ID    141 156 168 154 142 139
## Count 138 137 116 115 102  89
##
## Separation cutoffs:
## ID    139  141  142  143  144  145  146  147  148  149  150  151
## Yield 0.08 0.07 0.08 0.15 0.08 0.07 0.22 0.29 0.08 0.19 0.08 0.19
##
## ID    152  153  154  155  156  158  159  160  161  162  163  164
## Yield 0.14 0.13 0.23 0.30 0.14 0.18 0.22 0.23 0.41 0.13 0.10 0.27
##
## ID    165  166  167  168  169  170  171  172  173  174  175  176
## Yield 0.19 0.30 0.14 0.52 0.17 0.14 0.08 0.38 0.16 0.20 0.13 0.13
##
## Yields upon debarcoding:
##       95.23% overall yield
## ID    139    141    142   143    144    145    146    147
## Yield 99.34% 99.53% 99.4% 98.63% 98.43% 99.65% 97.75% 96.65%
##
## ID    148    149    150    151    152    153    154    155
## Yield 98.95% 98.28% 99.27% 95.61% 94.94% 96.76% 93.94% 94.5%
##
## ID    156    158    159    160    161    162    163    164
## Yield 94.93% 89.72% 92.17% 95.17% 88.81% 92.22% 96.31% 91.75%
##
## ID    165    166    167    168  169    170    171    172    173
## Yield 90.99% 91.32% 96.84% 100% 92.13% 95.78% 97.65% 78.84% 93.22%
##
## ID    174    175    176
## Yield 96.98% 95.37% 96.54%
```

4.5 `plotEvents`: Normalized intensities for each barcode population

```
# generate events plot
plotEvents(x = re, which_bc = 139, n_events = "all")
```



5 Compensation work-flow

5.1 `computeCompmat`: Computation of the compensation matrix

Given a flowFrame of single-stained beads (or cells) and a numeric vector of barcode masses and barcode IDs, `computeCompmat` estimates the compensation matrix as follows. Let s_{ij} denote the portion of signal measured in channel j that is due to spill of channel i. Assuming spillover to be a linear phenomenon, we compute this fraction as the median intensity measured in the affected channel, m_j^+, over the median intensity of the spilling channel, m_i^+. The median signal of events that are i) negative in the given channel, ii) are not assigned to potentially interacting channels, and iii) are not unassigned, m_j^- and m_i^-, respectively, are substracted as to account for background:

$$s_{ij} = \frac{m_j^+ - m_j^-}{m_i^+ - m_i^-}$$

On the basis of their additive nature, spill values are estimated independently for every pair of interacting channels. The current framework exclusively takes account of interactions that are sensible from a chemical and physical point of view. Precisely, $M \pm 1$ channels (*abundance sensitivity*), the $M + 16$ channel (*oxide formation*) and channels that measure potentially contaminated metals (*isotopic impurities*; @ref()) are taken into consideration. By default, the SM's diagonal entries s_{ii} are set to 1 to make spill relative to the total.

Table 1: List of mass channels that may contain isotopic contaminations, and are considered in computation of the compensation matrix in addition to $M \pm 1$ (*detection sensitivity*), $M + 16$ channels (*oxide formation*).

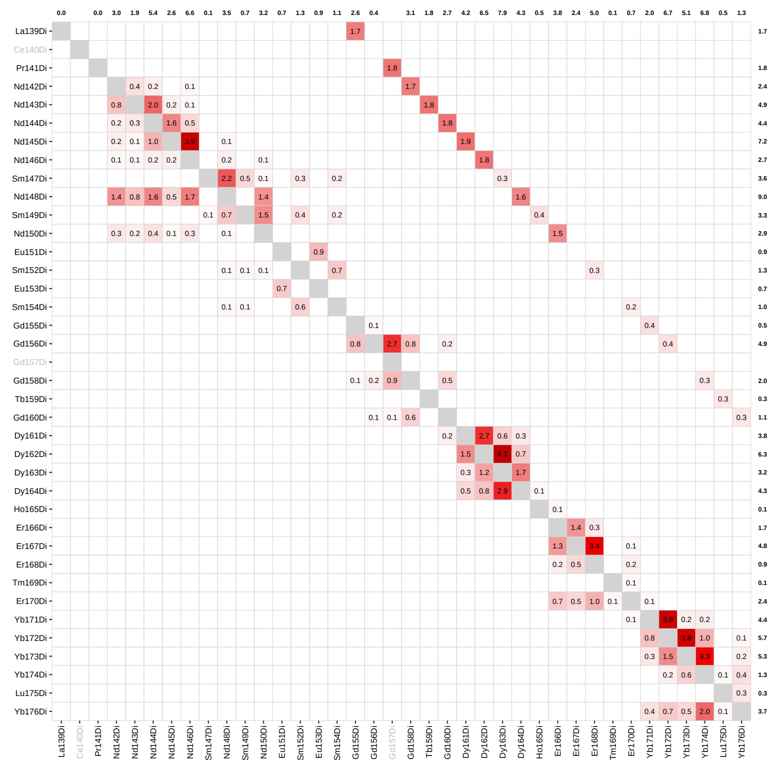
Metal	Isotope masses
La	138, 139
Pr	141
Nd	142, 143, 144, 145, 146, 148, 150
Sm	144, 147, 148, 149, 150, 152, 154
Eu	151, 153
Gd	152, 154, 155, 156, 157, 158, 160
Dy	156, 158, 160, 161, 162, 163, 164
Tb	159
Er	162, 164, 166, 167, 168, 170
Ho	165
Yb	168, 170, 171, 172, 173, 174, 176
Tm	169
Lu	175, 176

```
compMat <- computeCompmat(x = re)
```

5.2 plotSpillmat: Spillover matrix heat map representation

plotSpillmat provides a visualization of estimated spill percentages as a heat map. Channels not corresponding to a barcode are annotated in grey, and colours are ramped to the highest spillover value present.

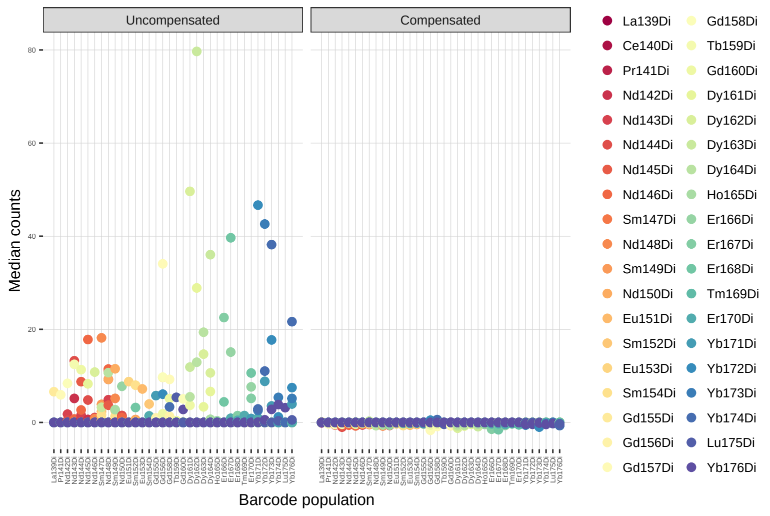
```
plotSpillmat(bc_ms = bc_ms, CM = compMat)
```



5.3 `plotScatter`: Raw vs. compensated data at a glance

For each barcode population, `plotScatters` compares the channel-wise median intensities between measurement and compensated data.

```r
plotScatter(x = re, CM = compMat)
```


Uncompensated
Compensated
Median counts
Barcode population
La139Di
Ce140Di
Pr141Di
Nd142Di
Nd143Di
Nd144Di
Nd145Di
Nd146Di
Sm147Di
Nd148Di
Sm149Di
Nd150Di
Eu151Di
Sm152Di
Eu153Di
Sm154Di
Gd155Di
Gd156Di
Gd157Di
Gd158Di
Tb159Di
Gd160Di
Dy161Di
Dy162Di
Dy163Di
Dy164Di
Ho165Di
Er166Di
Er167Di
Er168Di
Tm169Di
Er170Di
Yb171Di
Yb172Di
Yb173Di
Yb174Di
Lu175Di
Yb176Di