beadplexr: Reproducible and automated analysis of multiplex bead assays

Ulrik Stervbo Corresp., [1,2] , **Timm H Westhoff** [1] , **Nina Babel** [1,2]

[1] Center for Translational Medicine, Medical Clinic I, Marien Hospital Herne, University Hospital of the Ruhr-University Bochum, Ruhr-Universität Bochum, Herne, Germany
[2] Berlin-Brandenburg Center for Regenerative Therapies, Charité – Universitätsmedizin, Berlin, Germany

Corresponding Author: Ulrik Stervbo
Email address: ulrik.stervbo-kristensen@charite.de

Multiplex bead assays are an extension of the commonly used sandwich ELISA. The advantage over ELISA is that they make simultaneous evaluation of several analytes possible. Several commercial assay systems, where the beads are acquired on a standard flow cytometer, exist. These assay systems come with their own software tool for analysis and evaluation of the concentration of the analyzed analytes. However, these tools are either tied in to particular commercial software or impose limitations to the number of beads that can be analyzed. In addition, all these solutions are ‘point and click’ which potentially obscures the steps taken in the analysis. Here we present beadplexer, an open-source R-package for the reproducible analysis of multiplex bead assay data. The package makes it possible to automatically identify bead clusters, and makes it easy to fit a standard curve and calculate the concentrations of the analyzed analytes. beadplexer is available from CRAN and from https://gitlab.com/ustervbo/beadplexr.

beadplexr: Reproducible and automated analysis of multiplex bead assays

Ulrik Stervbo[1,2], Timm H. Westhoff[1], Nina Babel[1,2]

[1]Center for Translational Medicine, Medical Clinic I, Marien Hospital Herne, University Hospital of the Ruhr-University Bochum, Herne, Germany

[2]Berlin-Brandenburg Center for Regenerative Therapies, Charité – Universitätsmedizin, Berlin, Germany

Correspondence

Ulrik Stervbo: ulrik.stervbo-kristensen@charite.de

Abstract

Multiplex bead assays are an extension of the commonly used sandwich ELISA. The advantage over ELISA is that they make simultaneous evaluation of several analytes possible. Several commercial assay systems, where the beads are acquired on a standard flow cytometer, exist. These assay systems come with their own software tool for analysis and evaluation of the concentration of the analyzed analytes. However, these tools are either tied in to particular commercial software or impose limitations to the number of beads that can be analyzed. In addition, all these solutions are 'point and click' which potentially obscures the steps taken in the analysis. Here we present `beadplexer`, an open-source R-package for the reproducible analysis of multiplex bead assay data. The package makes it possible to automatically identify bead clusters, and makes it easy to fit a standard curve and calculate the concentrations of the analyzed analytes. `beadplexer` is available from CRAN and from https://gitlab.com/ustervbo/beadplexr.

Introduction

The enzyme-linked immunosorbent assay (ELISA) is a commonly used method to determine the concentration of soluble analytes such as cytokines. It is a single point assay and query into several analytes can be time consuming. Development in polystyrene bead preparations made it possible to construct assays that allow for query of several analytes at the same time. Similar to the ELISA, the analytes of interest are captured by a primary antibody. The captured analytes are subsequently labelled with a secondary antibody and finally detected with a fluorochrome conjugated tertiary antibody (Figure 1A). The detection antibody is specific for the secondary antibody. In a multiplex bead assay, the primary antibody is fixed on a polystyrene bead, and physical properties as well as colors of the beads are used to distinguish the different analytes studied. The data is usually collected using a standard flow cytometer.

The LEGENDplex system from BioLegend, the CBA system from BD Biosciences, and the MACSPlex from Miltenyi Biotec are all bead based multiplex systems. The systems differ slightly in terms of properties and colors used, and in the number of analytes that can be simultaneously identified. The Bio-Plex system from Bio-Rad works in a similar manner as those described here, but requires a dedicated instrument and does not produce files suitable for analysis with `beadplexr`. The individual assays that can be analyzed with `beadplexr` are described below.

LEGENDplex: Beads fall into two large groups based on forward and side scatter properties. Within each group, the individual analytes detected are discriminated by the intensity of Allophycocyanin (APC) of the beads. The concentration of the analyte is related to the intensity of Phycoerythrin (PE).

CBA: All beads have similar forward and side scatter properties. The individual analytes are discriminated by the intensity of APC and APC-Cy7 of the bead. The concentration of the analyte is related to the intensity of PE.

MACSPlex: All beads have similar forward and side scatter properties. The individual analytes are discriminated by the intensity of PE and Fluorescein isothiocyanate (FITC) of the bead. The concentration of the analyte is related to the intensity of APC.

All multiplex systems come with their own analysis software. However, these solutions might come with an added price tag because of binding to a particular piece of software, or because the analysis software allows analysis of only a limited number of beads. In addition, the usability and flexibility of the analysis solutions are restricted and often impractical for experiments with a large number of samples. Currently no open source alternative exists.

Here the general usage of the `beadplexr` package for R (R Core Team, 2018) is introduced. It will be demonstrated how to load the FACS-files, identify bead populations, draw standard curves and calculate concentration of the test samples.

Materials & Methods

The functionality of the package is illustrated using the LEGENDplex example data set included in the package. The data are from an unpublished "Human Growth Factor Panel (13-plex)" (BioLegend) experiment performed in our laboratory. The dataset consists of eight controls samples and a serum sample from a single healthy volunteer. All samples were processed in duplicates and per manufacturer’s instructions. The data was acquired on a CytoFLEX cytometer (Beckman Coulter). An example fcs-file is also included in the package.

Data was analyzed with R, version 3.4.4, (R Core Team, 2018) and plots created with ggplot2 (Wickham, 2009) and cowplot (Wilke, 2017).

Results

Package overview

The released package can be installed from CRAN and the development version from GitLab:

```
# From CRAN
install.packages("beadplexr")
# From GitLab using devtools
# install.packages("devtools")
devtools::install_git("https://gitlab.com/ustervbo/beadplexr")
#
# Or with vignettes built
# devtools::install_git("https://gitlab.com/ustervbo/beadplexr",
#                       build_vignettes = TRUE)
```

The package provides several steps to extract the analyte concentration from the raw data (Figure 1B). The functions for interacting with the data are flexible, but sensible defaults make them accessible to the novice R-user.

Reading fcs-files

`beadplexr` works with Flow Cytometry Standard (FCS) files (Seamer et al., 1997), which is the usual output of a flow cytometer. The function `read_fcs()` loads the given fcs-file using the functionality provided by the Bioconductor package `flowcore` (Ellis et al., 2017) and performs the following steps:

1. Apply an arcsinh transformation of the bead channels
2. Remove boundary events of the forward and side scatter channels
3. Optionally subset the channels to contain just bead events – this might improve identification of the beads
4. Convert the FACS-data to a `data.frame`

```
library(beadplexr)

.file_name <- system.file("extdata",
                          "K2-C07-A7.fcs",
                          package = "beadplexr")
.data <- read_fcs(
  .file_name = .file_name,
  .fsc_ssc = c("FSC-H", "SSC-H"),
  .bead_channels = c("FL6-H", "FL2-H"),
  .filter = list(
    "FSC-H" = c(3.75e5L, 5.5e5L),
    "SSC-H" = c(4e5L, 1e6L),
    "FL6-H" = c(7L, Inf)
  )
)
```

It might get a bit of trying to get the settings correct for an experiment, but once established it should remain stable. However, this provides that there are no change of cytometer, and that there is no particular drift in the used cytometer.

The loaded FACS data can be visualized with the `facs_plot()` convenience function (Figure 2), which make use of `ggplot2` (Wickham, 2009).

Identification of analyte MFI

The first step of the experiment analysis is to get the mean fluorescence intensity (MFI) of each analyte. In the case of LEGENDplex data, this requires the identification of each analyte per size and APC intensity and the MFI of in the PE channel. The function `identify_legendplex_analyte()` is a convenience function for identifying the bead populations and their MFIs for an experiment based on the LEGENDplex assay.

Identification of analytes is achieved with the function `identify_analyte()`, which finds analyte clusters and sort the analyte clusters based on their centres. Several methods for automatic cluster identification can be evoked by the function. The default function is `clara()` (clustering large applications) from the package `cluster`. (Maechler et al., 2017). The base `kmeans()` as well as `dbscan()` from the `fpc` package, and `mclust()` from the `mclust` package are also interfaced (Hennig, 2015; Scrucca et al., 2016). We have found that `dbscan()` is the best clustering for the forward-side scatter population identification, but requires some (and sometimes a lot) trial and error to get the parameters right. `clara()` is a great all-rounder, but in some cases - such as this - `mclust()` is the better choice. The best function depends on the data, and finding it is a matter of trial and error.

`identify_legendplex_analyte()` make use of structured Panel Information to identify the two bead populations according to their forward and side scatter properties, and for each of these subpopulations, the analytes are identified. The package comes with a set of LEGENDplex panel definitions, and more are being added. The Panel Information can be loaded by passing the name or a name pattern to the `load_panel()` function. The information file is in YAML format, and contains name and start concentration for each bead ID, the name of the panel, the fold dilution of the standards, and the units of the analytes. It is also possible to specify a file outside the package. The panel definition is not required, but makes sense if the assay is repeated across several projects.

```
library(dplyr)
library(purrr)

data(lplex)

panel_info <-
  load_panel(.panel_name = "Human Growth Factor Panel (13-plex)")

args_ident_analyte <-
  list(
    fs = list(
      .parameter = c("FSC-A", "SSC-A"),
      .column_name = "Bead group",
      .method = "mclust"
    ),
    analytes = list(.parameter = "FL6-H",
                    .column_name = "Analyte ID")
  )

analytes_identified <- lplex %>%
  lapply(
    identify_LEGENDplex_analyte,
    .analytes = panel_info$analytess,
    .method_args = args_ident_analyte
  )
```

The analyte IDs for the “Human Growth Factor Panel (13-plex)” bead group A are A4, A5, A6, A7, A8, A10 and for group B the analyte IDs are B2, B3, B4, B5, B6, B7, B9. In this case, the beads are arranged from low to high, that is the lowest analyte ID has lowest intensity in the APC channel (Figure 3).

This initial and crucial step of the analysis has been successfully performed with unpublished data from a CBA experiment and from a MACSPlex experiment using the function `identify_analyte()`.

With the analytes identified and the bead populations documented the MFI of each analyte can finally be calculated. `beadplexr` provides the possibility to calculate geometric, harmonic, and arithmetic mean. We combine the list of FACS-data to a `data.frame` with Sample, analyte ID, and the MFI because the creation of a standard curve in the next steps needs the MFI of several standard samples.

```
analyte_mfi <- analytes_identified %>%
  filter(!is.na(`Analyte ID`)) %>%
  map_df(calc_analyte_mfi,
         .parameter = "FL2-H",
         .column_name = "Analyte ID",
         .mean_fun = "geometric",
         .id = "Sample") %>%
  mutate(`FL2-H` = log10(`FL2-H`))
```

Calculation of standard and test samples

The calculation of the analytes requires two steps:

1. Create a standard curve by fitting a function to the MFI of the standard analytes and their known concentration.
2. Estimate the concentration of each sample analyte from the fitted function.

Split the `data.fame` with the MFI information into standard and samples.

```
# All standard samples have the pattern C[number]
standard_data <- analyte_mfi %>%
  filter(grepl("C[0-9]", Sample))

# All non-standards are samples... we could also filter on S[number]
sample_data <- analyte_mfi %>%
  filter(!grepl("C[0-9]", Sample))
```

The first thing needed to calculate the standard curve for each analyte is the concentration of the analyte. This we can calculate using the function `calc_std_conc()`, when we know the order of the samples, the start concentration, and the dilution factor.

The order of the samples usually ranges from 0 to 7. If the nomenclature on the LEGENDplex assay protocol is followed, 7 indicates the highest concentration of the standard analyte, 1 indicates the lowest concentration and 0 indicates blank. The start concentration for each analyte is stored in the Panel Information (depending on the panel it might differ from analyte to analyte). The dilution factor is also given in the Panel Information and is usually 4 (the concentration of each standard analyte is 4 times lower than the previous concentration). If - for some reason - you do not use the same dilution factor throughout all analytes and standards you need to indicate the appropriate dilution factors by hand.

```
# Helper function to extract the sample number
as_numeric_standard_id <- function(.s){
  .s %>%
    stringr::str_extract("C[0-9]") %>%
    stringr::str_sub(start = -1L) %>%
    stringr::as.numeric()
}

standard_data <- standard_data %>%
  mutate(`Sample number` = as_numeric_standard_id(Sample)) %>%
  left_join(as_data_frame_analyte(panel_info$analytes),
            by = "Analyte ID") %>%
  group_by(`Analyte ID`) %>%
  mutate(
    Concentration = calc_std_conc(
      `Sample number`,
      concentration,
      .dilution_factor = panel_info$std_dilution
    )
  ) %>%
  mutate(Concentration = log10(Concentration)) %>%
  select(-concentration, -`Bead group`)
```

The next step is to fit a standard curve for each analyte. With the standard curve we can calculate the concentration of the experimental samples (the purpose of the initial work), we can check the quality of the measurements - and the standard curve - by back calculate the standard concentration and compare this to the expected concentration, and we can plot the experimental samples on the standard curve (`beadplexr` provides easy functions for all of this).

However, in each case we need to ensure that the correct standard curve is used with the correct data, which means we have to juggle at least three structures: A `data.frame` with the standard data, a `data.frame` with the experimental sample data and the models for each analyte (probably a list). It quickly becomes tedious to ensure that everything is in the correct order - and to be sure it is error prone.

To circumvent this, we can use the `nest()` and its inverse `unnest()` functions of the `tidyr` package (Wickham & Henry, 2018).

```
library(tidyr)

standard_data <- standard_data %>%
  ungroup() %>%
  nest(-`Analyte ID`, -name, .key = "Standard data")

sample_data <- sample_data %>%
  ungroup() %>%
  nest(-`Analyte ID`, .key = "Sample data")

```

```
plex_data <-
  inner_join(standard_data, sample_data, by = "Analyte ID")
```

With everything in a neatly arranged `data.frame` we can now focus on the actual task at hand, namely calculation of the standard curve for each analyte. For this we use the function `fit_standard_curve()`, which interfaces the `drm()` function from the `drc` package (Ritz et al., 2015).

```
# When clustering is performed with mclust, the package mclust is
# loaded in thebackground (an unfortunate necessity). The mclust
# package also has a function called `map`, so an unlucky side effect
# of clustering with mclust, is that we need to be specify which map
# function we use
plex_data <- plex_data %>%
  group_by(`Analyte ID`) %>%
  mutate(`Model fit` = purrr::map(`Standard data`,
                                  fit_standard_curve))
```

We can plot the standard curve using the built in `plot_std_curve()` function (Figure 4A). With the standard curve created we can estimate the concentrations of the samples, but also of the standards. The latter is to help us verify that the standard measurements were all fine, and that we can trust the estimation of the sample concentrations.

After calculating the concentrations we can plot the known standard concentrations versus the estimated standard concentrations using the function `plot_target_est_conc()` (Figure 4B) and visualize where the samples fall on the standard curve with `plot_target_est_conc()` (Figure 4C).

```
plex_data <- plex_data %>%
  group_by(analyte.ID) %>%
  mutate(`Standard data` =
           purrr::map2(`Standard data`, `Model fit`,
                       calculate_concentration)) %>%
  mutate(`Sample data` =
           purrr::map2(`Sample data`, `Model fit`,
                       calculate_concentration)) %>%
  mutate(`Std conc` =
           purrr::map(`Standard data`,
                      plot_target_est_conc)) %>%
  mutate(`Est curve` =
           purrr::pmap(
             list(`Sample data`, `Standard data`, `Model fit`, name),
             plot_estimate
           ))
```

Lastly we fulfill the purpose of everything above and extract the concentration of each analyte for each sample.

```
plex_data %>%
  unnest(`Sample data`) %>%
  mutate(Calc.conc = 10 ^ Calc.conc,
         `Calc.conc error` = 10 ^ `Calc.conc error`)
```

Discussion

Multiplex bead assays make simultaneous evaluation of several analytes possible. Because of this, they are an attractive alternative to the commonly used sandwich ELISA. Commercial systems are available for acquisition on a standard flow cytometer. For the data analysis, these commercial systems make use of their own proprietary software, which impose different limitations to the analysis. The R-package `beadplexr`, released under the MIT license, is meant as an open-source alternative to these commercial systems. The package is available from CRAN and from https://gitlab.com/ustervbo/beadplexr.

A critical step in the analysis multiplex bead assays is the identification of bead populations corresponding to each analyte. A single function in `beadplexr` acts as an interface to several common, and tested, cluster functions, making it easy to find the best suited cluster function. Future versions of the package will see improvements in this part, with inclusion of other clustering methods and perhaps a heuristic for automatic method selection.

Flow cytometry data are inherently noisy. The R-package only provides a rudimentary function for removing lonely points and lets the cluster functions determine which events are considered noisy. However, a very noisy data set might make it difficult for an optimal identification of the bead clusters. De-noising multidimensional data is not trivial, but work is planned in this direction for a future release.

Conclusion

`beadplexr` provides a frame work for easy and reproducible analysis of multiplex bead assays for the experienced and the novice user alike.

Acknowledgments

The authors wish to thank Miltenyi Biotec, Bergisch Gladbach, Germany and C. McGuckin, CTIBIOTECH, Lyon, France for the example data to test the package. We further acknowledge the support from the German Research Foundation (DFG) and the Open Access Publication Fund of Charité – Universitätsmedizin Berlin.

References

Ellis B., Haaland P., Hahne F., Meur NL., Gopalakrishnan N., Spidlen J., Jiang M. 2017. *flowCore: flowCore: Basic structures for flow cytometry data*.

Hennig C. 2015. *fpc: Flexible Procedures for Clustering*.

Maechler M., Rousseeuw P., Struyf A., Hubert M., Hornik K. 2017. *cluster: Cluster Analysis Basics and Extensions*.

R Core Team 2018. *R: A Language and Environment for Statistical Computing*. Vienna, Austria.

Ritz C., Baty F., Streibig JC., Gerhard D. 2015. Dose-Response Analysis Using R. *PLOS ONE* 10.

Scrucca L., Fop M., Murphy TB., Raftery AE. 2016. mclust 5: clustering, classification and density estimation using Gaussian finite mixture models. *The R Journal* 8:205–233.

Seamer LC., Bagwell CB., Barden L., Redelman D., Salzman GC., Wood JCS., Murphy RF. 1997. Proposed new data file standard for flow cytometry, version FCS 3.0. *Cytometry* 28:118–122. DOI: 10.1002/(SICI)1097-0320(19970601)28:2<118::AID-CYTO3>3.0.CO;2-B.

Wickham H. 2009. *ggplot2: elegant graphics for data analysis*. Springer.

Wickham H., Henry L. 2018. *tidyr: Easily Tidy Data with "spread()" and "gather()" Functions*.

Wilke CO. 2017. *cowplot: Streamlined Plot Theme and Plot Annotations for "ggplot2."*

Funding

This work was supported by BMBF grant e:KID.

Competing Interests

The authors declare that they have no competing interests.

Author Contributions

US performed the experiments, authored the R-package, prepared the figures, and authored the paper. TW and NB designed the study and authored the paper.

Figure legends

Figure 1: Overview of assay principle and the package workflow.

A) Schematic overview of the principle of a LEGENDplex assay. B) Steps in analysis of a multiplex bead assay with accompanying visualizations.

Figure 2: Visualization of FACS data.

A) Common monochrome dot-plot. B) Pseudo-colored dot-plot (requires the R-package `hexbin` to be installed).

Figure 3: Bead identification and visualization of LEXENDplex data.

A) Identification of the two bead populations 'A' and 'B' in a LEGENDplex data set. B) Identification of analytes of the bead population 'A'. B) Identification of analytes of the bead population 'B'. Noisy data points are assigned the group 'None' or 'NA'.

Figure 4: Visualization of standard and test samples for Angiopoietin-2.

A) The standard curve of Angiopoietin-2 ranging from 50,000.0 to 12.21 pg/ml. B) Correlation between the standard concentration (x-axis) and the calculated concentration of the standard samples (y-axis). The back calculation is done using the fit in (A) and the MFI of the samples. C) Position of the test samples on the standard curve.

Figure 1(on next page)

Overview of assay principle and the package workflow

A) Schematic overview of the principle of a LEGENDplex assay. B) Steps in analysis of a multiplex bead assay with accompanying visualizations.

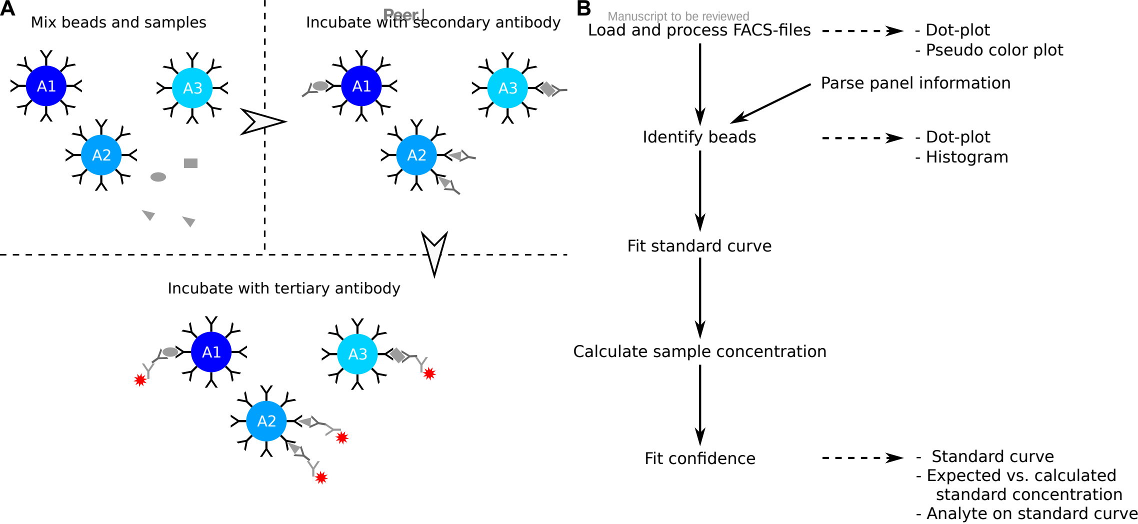

A
Mix beads and samples
A1
A3
A2
Incubate with secondary antibody
A1
A3
A2
Incubate with tertiary antibody
A1
A3
A2
B
Load and process FACS-files
- Dot-plot
- Pseudo color plot
Parse panel information
Identify beads
- Dot-plot
- Histogram
Fit standard curve
Calculate sample concentration
Fit confidence
- Standard curve
- Expected vs. calculated standard concentration
- Analyte on standard curve

Figure 2(on next page)

Visualization of FACS data

A) Common monochrome dot-plot. B) Pseudo-colored dot-plot (requires the R-package hexbin to be installed).

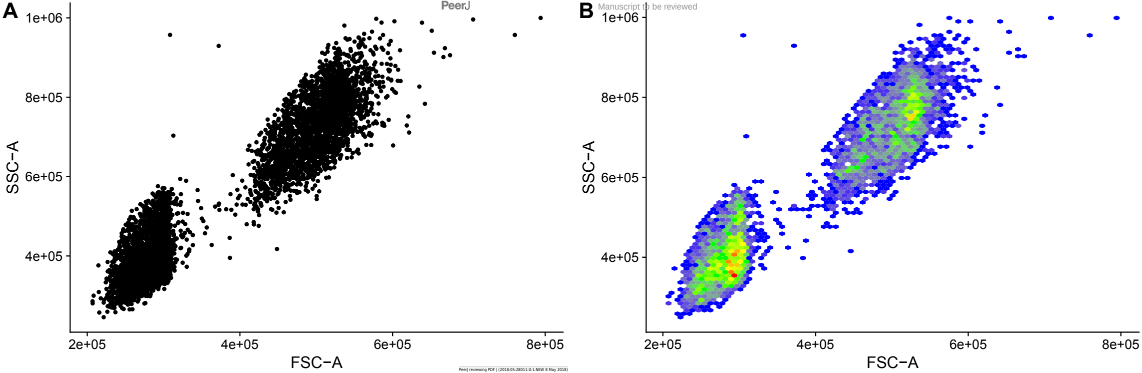

A
1e+06
8e+05
6e+05
4e+05
SSC−A
2e+05
4e+05
6e+05
8e+05
FSC−A
B
1e+06
8e+05
6e+05
4e+05
SSC−A
2e+05
4e+05
6e+05
8e+05
FSC−A

Figure 3(on next page)

Bead identification and visualization of LEXENDplex data

A) Identification of the two bead populations ‘A’ and ‘B’ in a LEGENDplex data set. B) Identification of analytes of the bead population ‘A’. B) Identification of analytes of the bead population ‘B’. Noisy data points are assigned the group ‘None’ or ‘NA’.

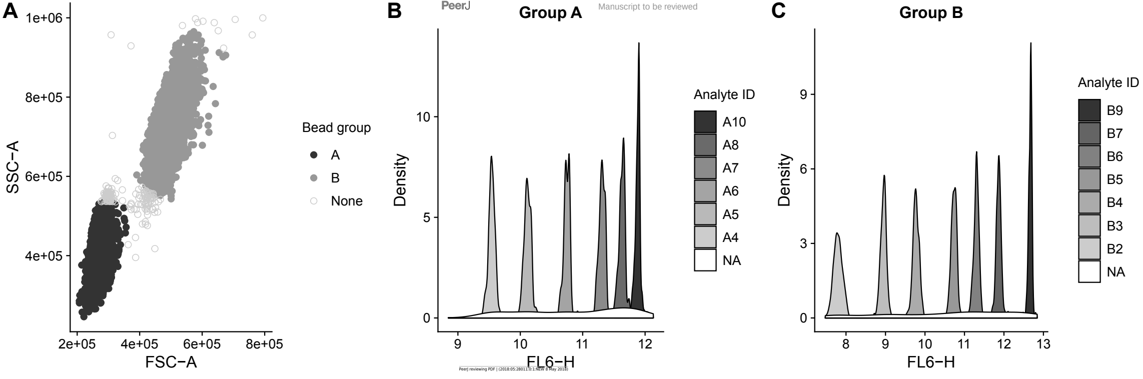
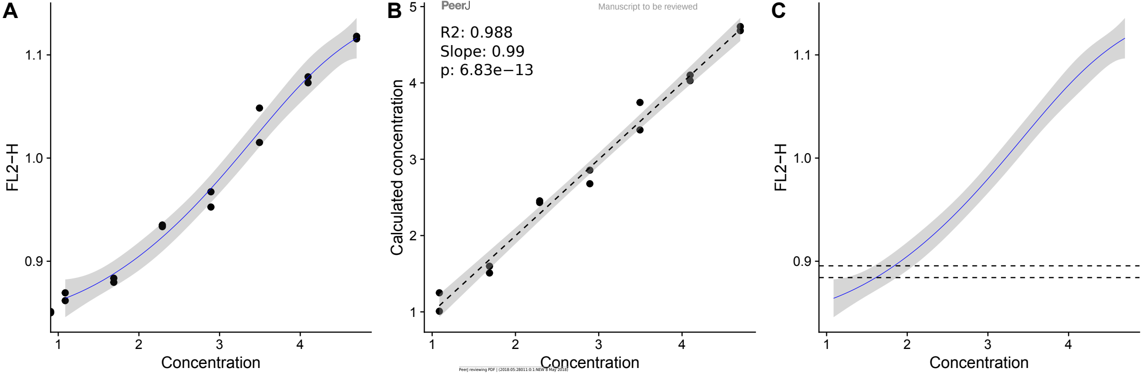

A
1e+06
8e+05
6e+05
4e+05
SSC−A
2e+05
4e+05
6e+05
8e+05
FSC−A
Bead group
A
B
None
B
Group A
Density
0
5
10
9
10
11
12
FL6−H
Analyte ID
A10
A8
A7
A6
A5
A4
NA
C
Group B
Density
0
3
6
9
8
9
10
11
12
13
FL6−H
Analyte ID
B9
B7
B6
B5
B4
B3
B2
NA

Figure 4(on next page)

Visualization of standard and test samples for Angiopoietin-2

A) The standard curve of Angiopoietin-2 ranging from 50,000.0 to 12.21 pg/ml. B) Correlation between the standard concentration (x-axis) and the calculated concentration of the standard samples (y-axis). The back calculation is done using the fit in (A) and the MFI of the samples. C) Position of the test samples on the standard curve.


A
FL2−H
1.1
1.0
0.9
1
2
3
4
Concentration
B
R2: 0.988
Slope: 0.99
p: 6.83e−13
Calculated concentration
5
4
3
2
1
1
2
3
4
Concentration
C
FL2−H
1.1
1.0
0.9
1
2
3
4
Concentration