

flowPlots: An R Package for Analyzing Gated ICS Flow Cytometry Data

Natalie Hawkins, Steve Self

April 15, 2025

Version 1.0.0

Abstract

The `flowPlots` package provides graphical displays with embedded statistical tests for gated ICS flow cytometry data, helper functions to prepare data for plotting, and a data class which stores "stacked" data and has methods for computing summary measures on stacked data, such as marginal, polyfunctional degree (pfd) data, and polyfunctional degree parts (pfdParts) data.

1 Introduction

The `flowPlots` package provides a boxplot function with embedded statistical tests to compare groups, data helper functions to prepare data for the boxplot, ternary plot, or bar plot, and a data storage class, *StackedData*, which has methods to create profile and summary data such as marginal, polyfunctional degree (pfd), and polyfunctional degree parts (pfdParts) data from "stacked" data. This document provides examples of the graphical displays which can be used to analyze gated ICS flow cytometry data. Most of these displays are based on summary data. If the summary data are already available, the displays can be made directly. If the summary data are not available, but the data is available in "stacked" format, methods from the *StackedData* class can be used to compute the summary data. Examples are given to illustrate the use of the *StackedData* class. If the summary data are not available and the data is available in a format different from "stacked", the user can take advantage of this package by converting the data to the "stacked" format. The examples are based on a small subset of the data described in Kollmann et al. (2009).

2 Plotting the data: boxplots, ternary plot, bar plot

The boxplot function and the data helper functions are used in the following examples. We load the profile data and the various summary data directly; i.e. we already have it available and do not need to compute it. For each example, we show the first few rows of the data frame being plotted.

2.1 Boxplots to compare groups on profileData

```
> data(profileDF)
> profileDataSubset = subset(profileDF, stim=="LPS" & concGroup==3 & cell=="mDC")
> profileDataSubset[1:3,]
```

	TNFa+IL6+IL12+IFNa+	TNFa+IL6+IL12+IFNa-	TNFa+IL6+IL12-IFNa+
1	0	0.43	0
2	0	0.32	0
3	0	0.44	0

	TNFa+IL6+IL12-IFNa-	TNFa+IL6-IL12+IFNa+	TNFa+IL6-IL12+IFNa-
1	21.86	0	0.29
2	2.75	0	0.50
3	3.06	0	1.97

	TNFa+IL6-IL12-IFNa+	TNFa+IL6-IL12-IFNa-	TNFa-IL6+IL12+IFNa+
1	0	19.71	0
2	0	9.45	0
3	0	7.86	0

	TNFa-IL6+IL12+IFNa-	TNFa-IL6+IL12-IFNa+	TNFa-IL6+IL12-IFNa-
1	0.00	0	2.0
2	0.23	0	1.1
3	0.44	0	2.4

	TNFa-IL6-IL12+IFNa+	TNFa-IL6-IL12+IFNa-	TNFa-IL6-IL12-IFNa+
1	0.14	0.29	1.00
2	0.32	1.05	0.46
3	0.00	2.40	0.00

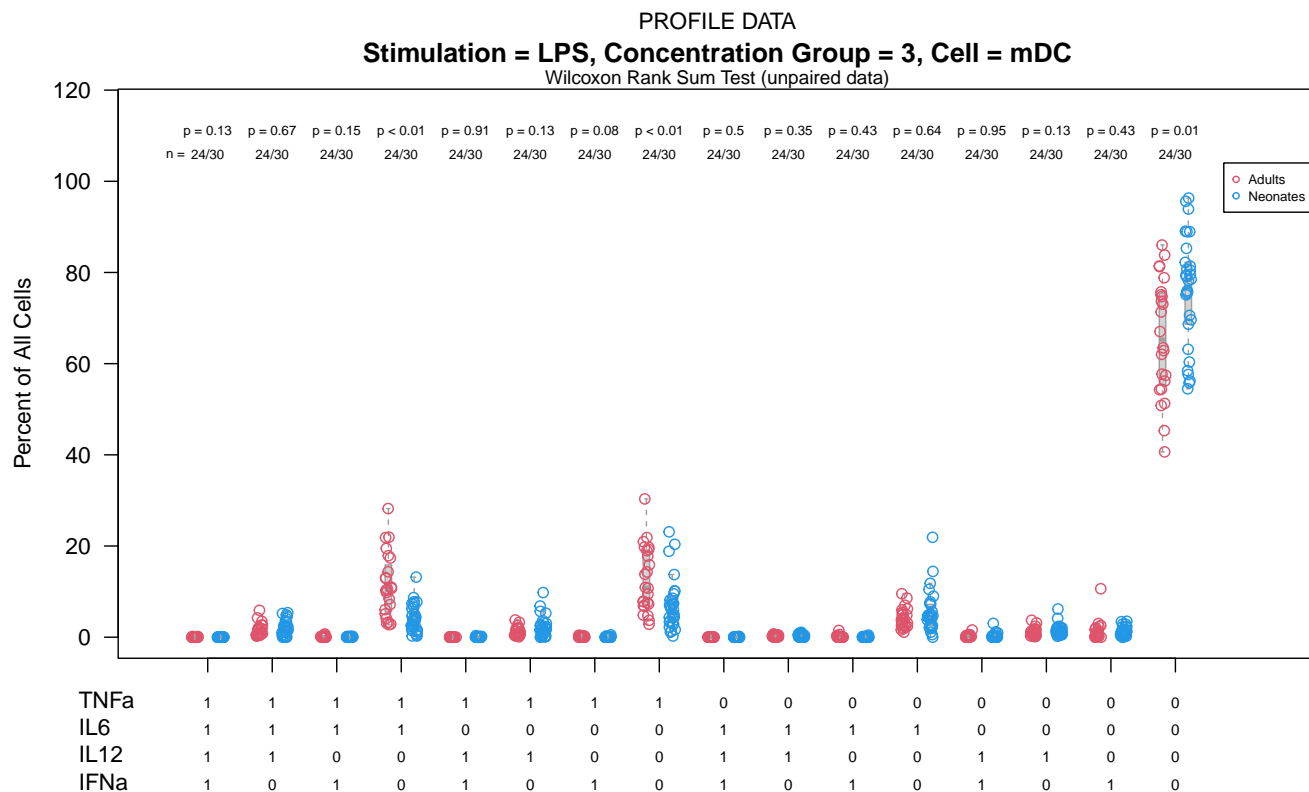
	TNFa-IL6-IL12-IFNa-	stim	concGroup	cell	groups	subjectId	group
1	54.28	LPS	3	mDC	1	a2004	adult
2	83.82	LPS	3	mDC	1	a2005	adult
3	81.43	LPS	3	mDC	1	a2006	adult

```

> # Use data helper function, makeDataList()
> groupDataList = makeDataList(profileDataSubset , "group", 1:16)
> # Make x-axis tick labels
> data(markerMatrix)
> theMarkers = colnames(markerMatrix)
> xTickLabels = cbind(theMarkers, t(markerMatrix))

> GroupListBoxplot(groupDataList, xlabel="",
+   ylabel="Percent of All Cells", boxOutliers=FALSE,
+   xAxisLabels=xTickLabels, xMtext="Marker Category",
+   mainTitle="Stimulation = LPS, Concentration Group = 3, Cell = mDC",
+   legendGroupNames=c("Adults", "Neonates"), pointColor=c(2,4),
+   testTitleCEX=.8, nCEX=.6, pCEX=.6, legendColor=c(2,4), legendCEX=.6,
+   xAxisCEX=.7, xAtMtext=-1)
> mtext("PROFILE DATA", side=3, line=2)

```



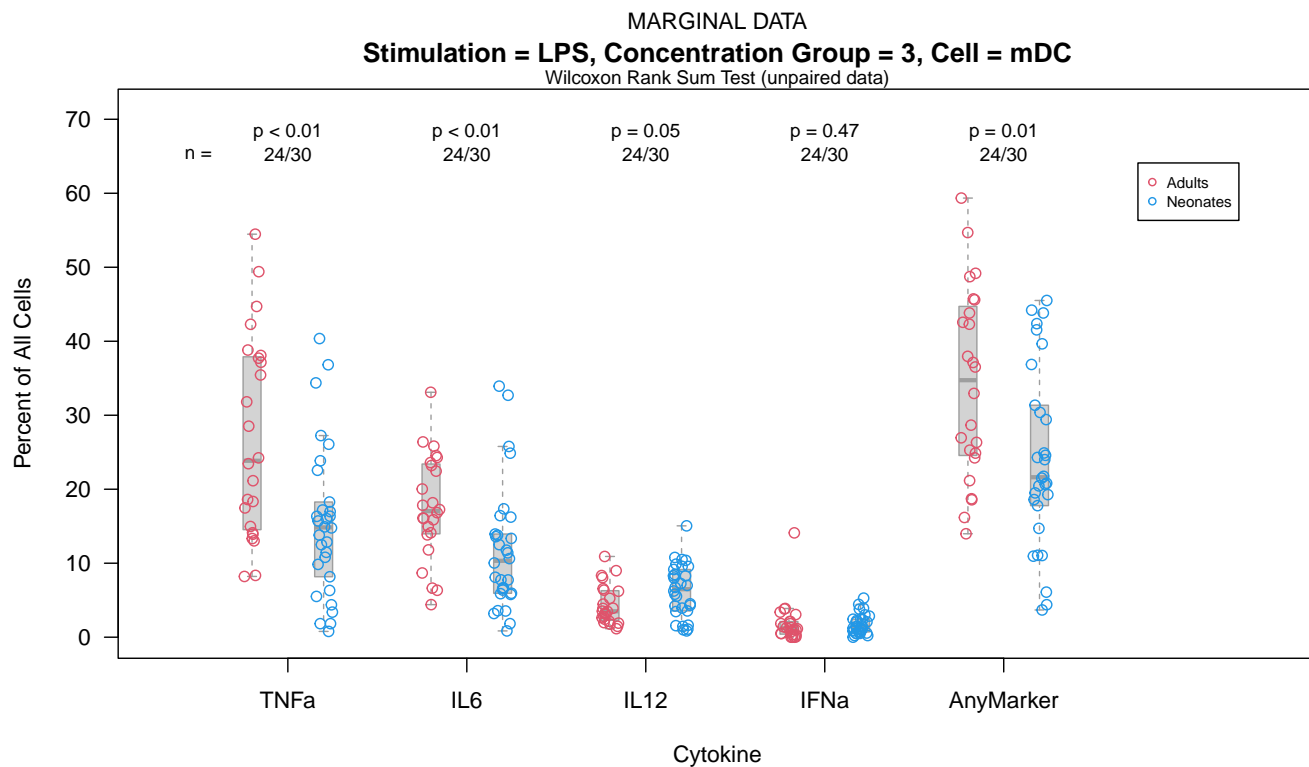
2.2 Boxplots to compare groups on marginalData

```
> data(marginalDF)
> marginalDataSubset = subset(marginalDF, stim=="LPS" & concGroup==3 & cell=="mDC")
> marginalDataSubset[1:3,]
```

	TNFa	IL6	IL12	IFNa	AnyMarker	stim	concGroup	cell	groups	subjectId	group
1	42.29	24.29	1.15	1.14	45.72	LPS	3	mDC	1	a2004	adult
2	13.02	4.40	2.42	0.78	16.18	LPS	3	mDC	1	a2005	adult
3	13.33	6.34	5.25	0.00	18.57	LPS	3	mDC	1	a2006	adult

```
> # Use data helper function, makeDataList()
> groupDataList = makeDataList(marginalDataSubset, "group", 1:5)

> GroupListBoxplot(groupDataList, xlabel="Cytokine",
+   ylabel="Percent of All Cells", boxOutliers=FALSE,
+   xAxisLabels=c("TNFa", "IL6", "IL12", "IFNa", "AnyMarker"),
+   mainTitle="Stimulation = LPS, Concentration Group = 3, Cell = mDC",
+   legendGroupNames=c("Adults", "Neonates"), pointColor=c(2,4),
+   testTitleCEX=.8, nCEX=.8, pCEX=.8, legendColor=c(2,4), legendCEX=.7)
> mtext("MARGINAL DATA", side=3, line=2)
```



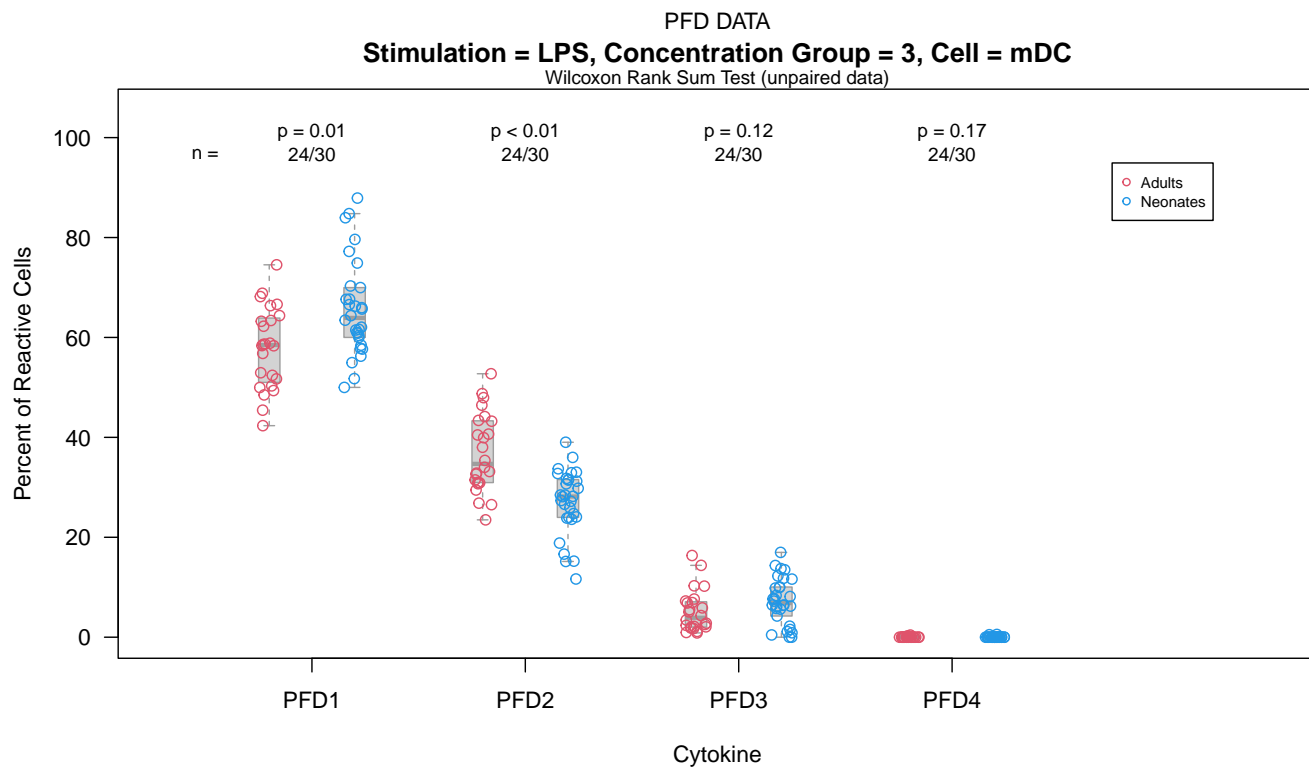
2.3 Boxplots to compare groups on pfdData

```
> data(pfdDF)
> pfdDataSubset = subset(pfdDF, stim=="LPS" & concGroup==3 & cell=="mDC")
> pfdDataSubset[1:3,]
```

	PFD1	PFD2	PFD3	PFD4	stim	concGroup	cell	groups	subjectId	group
1	50.30621	48.75328	0.9405074	0	LPS	3	mDC	1	a2004	adult
2	74.53646	23.48578	1.9777503	0	LPS	3	mDC	1	a2005	adult
3	68.17447	29.45611	2.3694130	0	LPS	3	mDC	1	a2006	adult

```
> # Use data helper function, makeDataList()
> groupDataList = makeDataList(pfdDataSubset, "group", 1:4)

> GroupListBoxplot(groupDataList, xlabel="Cytokine",
+   ylabel="Percent of Reactive Cells", boxOutliers=FALSE,
+   mainTitle="Stimulation = LPS, Concentration Group = 3, Cell = mDC",
+   legendGroupNames=c("Adults", "Neonates"), pointColor=c(2,4),
+   testTitleCEX=.8, nCEX=.8, pCEX=.8, legendColor=c(2,4), legendCEX=.7)
> mtext("PFD DATA", side=3, line=2)
```



2.4 Boxplots to compare groups on pfdPartsData

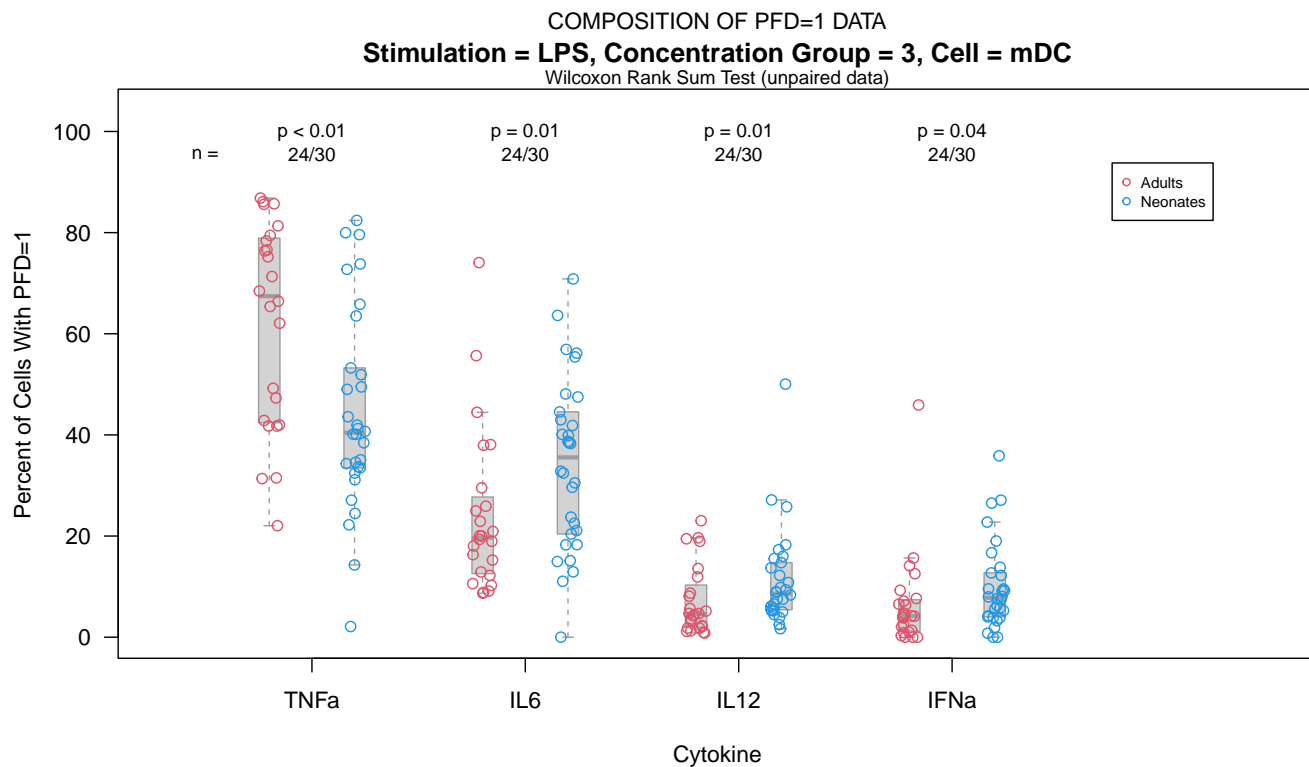
```
> data(pfdPartsList)
> # Look at the composition percentages for the case where PFD=1
> pfdEq1PartsDataSubset = subset(pfdPartsList[[1]], stim=="LPS" & concGroup==3 & cell=="mDC")
> pfdEq1PartsDataSubset[1:3,]
```

	TNFa	IL6	IL12	IFNa	stim	concGroup	cell	groups	subjectId
1	85.69565	8.695652	1.260870	4.347826	LPS	3	mDC	1	a2004
2	78.35821	9.121061	8.706468	3.814262	LPS	3	mDC	1	a2005
3	62.08531	18.957346	18.957346	0.000000	LPS	3	mDC	1	a2006

```
group
1 adult
2 adult
3 adult

> # Use data helper function, makeDataList()
> groupDataList = makeDataList(pfdEq1PartsDataSubset, "group", 1:4)

> GroupListBoxplot(groupDataList, xlabel="Cytokine",
+   ylabel="Percent of Cells With PFD=1", boxOutliers=FALSE,
+   mainTitle="Stimulation = LPS, Concentration Group = 3, Cell = mDC",
+   legendGroupNames=c("Adults", "Neonates"), pointColor=c(2,4),
+   testTitleCEX=.8, nCEX=.8, pCEX=.8, legendColor=c(2,4), legendCEX=.7)
> mtext("COMPOSITION OF PFD=1 DATA", side=3, line=2)
```



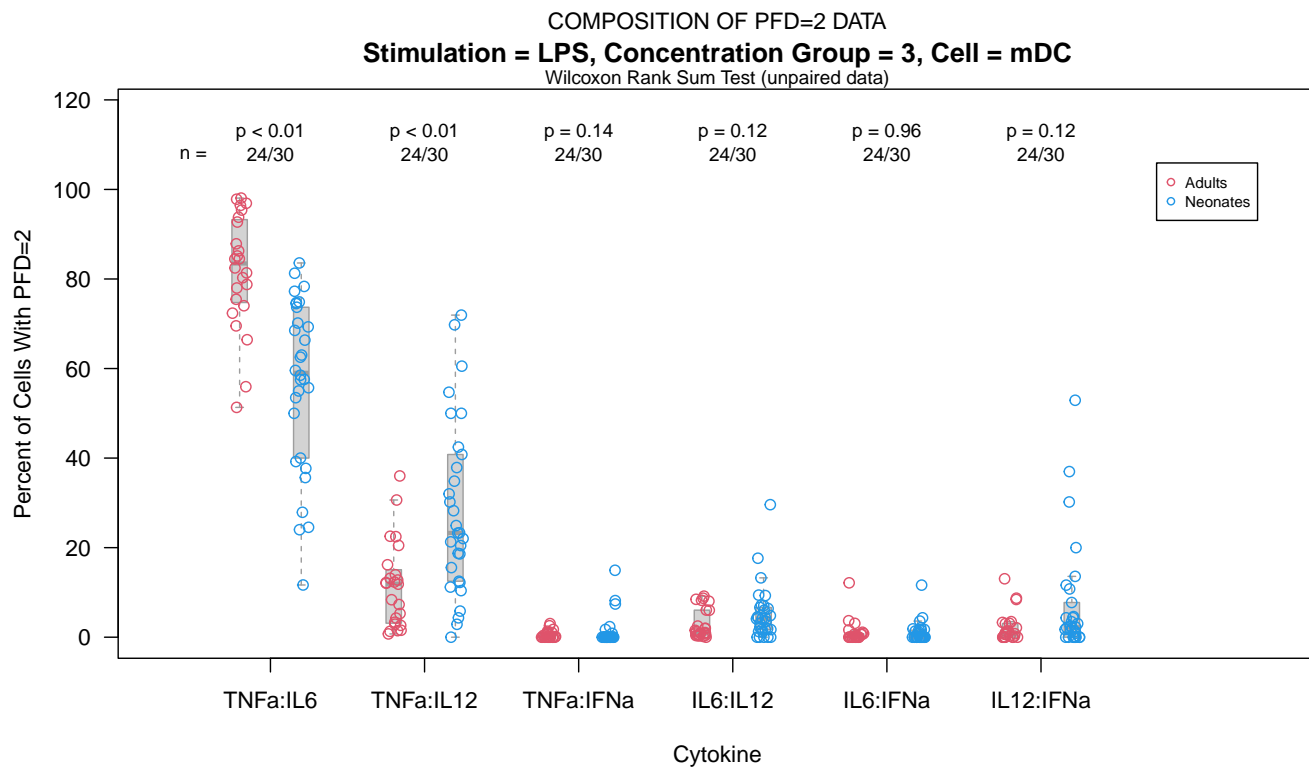
```
> # Look at the composition percentages for the case where PFD=2
> pfdEq2PartsDataSubset = subset(pfdPartsList[[2]], stim=="LPS" & concGroup==3 & cell=="mDC")
> pfdEq2PartsDataSubset[1:3,]
```

	TNFa:IL6	TNFa:IL12	TNFa:IFNa	IL6:IL12	IL6:IFNa	IL12:IFNa	stim	concGroup	cell
1	98.07088	1.301032	0	0.000000	0	0.6280843	LPS	3	mDC
2	72.36842	13.157895	0	6.052632	0	8.4210526	LPS	3	mDC
3	55.94150	36.014625	0	8.043876	0	0.0000000	LPS	3	mDC

groups	subjectId	group
1	1	a2004 adult
2	1	a2005 adult
3	1	a2006 adult

```
> # Use data helper function, makeDataList()
> groupDataList = makeDataList(pfdEq2PartsDataSubset, "group", 1:6)

> GroupListBoxplot(groupDataList, xlabel="Cytokine",
+   ylabel="Percent of Cells With PFD=2", boxOutliers=FALSE,
+   mainTitle="Stimulation = LPS, Concentration Group = 3, Cell = mDC",
+   legendGroupNames=c("Adults", "Neonates"), pointColor=c(2,4),
+   testTitleCEX=.8, nCEX=.8, pCEX=.8, legendColor=c(2,4), legendCEX=.7)
> mtext("COMPOSITION OF PFD=2 DATA", side=3, line=2)
```



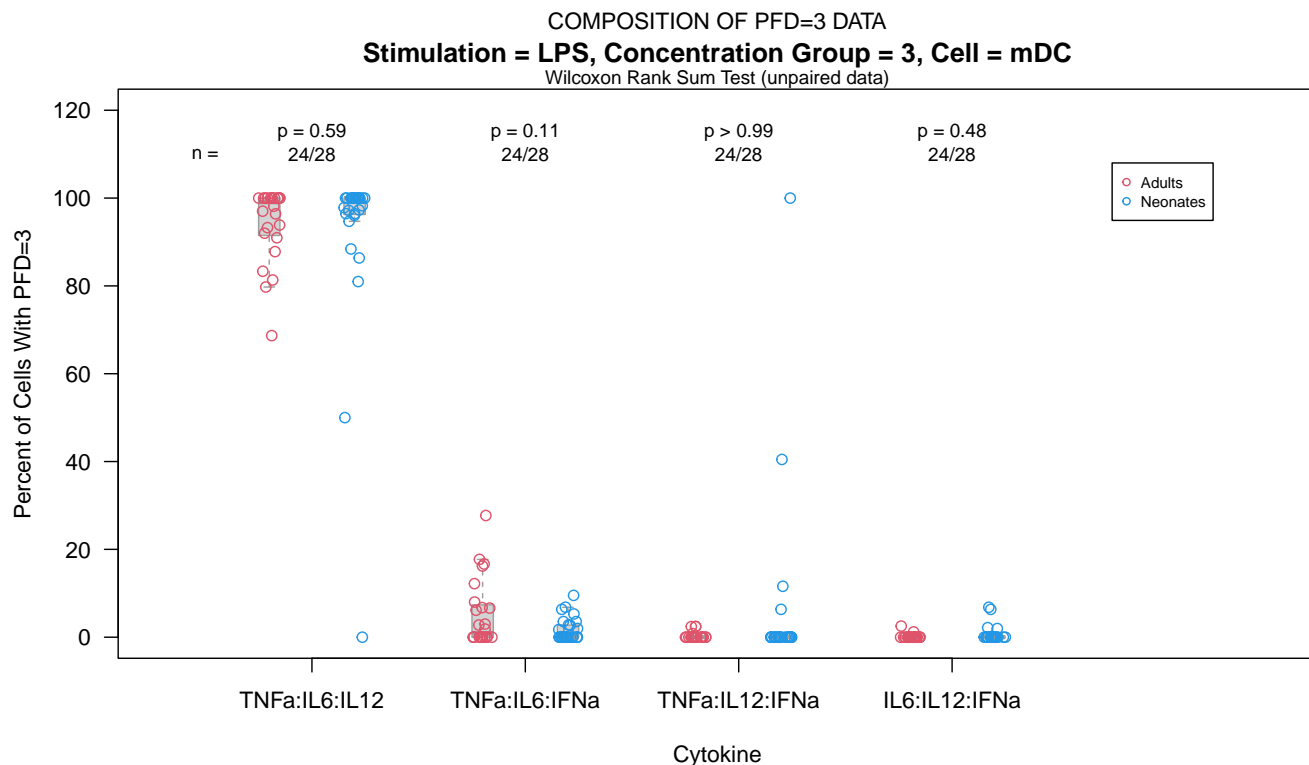
```
> # Look at the composition percentages for the case where PFD=3
> pfdEq3PartsDataSubset = subset(pfdPartsList[[3]], stim=="LPS" & concGroup==3 & cell=="mDC")
> pfdEq3PartsDataSubset[1:3,]

  TNFa:IL6:IL12 TNFa:IL6:IFNa TNFa:IL12:IFNa IL6:IL12:IFNa stim concGroup cell
1             100             0             0             0  LPS           3  mDC
2             100             0             0             0  LPS           3  mDC
3             100             0             0             0  LPS           3  mDC

groups subjectId group
1      1      a2004 adult
2      1      a2005 adult
3      1      a2006 adult

> # Use data helper function, makeDataList()
> groupDataList = makeDataList(pfdEq3PartsDataSubset, "group", 1:4)

> GroupListBoxplot(groupDataList, xlabel="Cytokine",
+   ylabel="Percent of Cells With PFD=3", boxOutliers=FALSE,
+   mainTitle="Stimulation = LPS, Concentration Group = 3, Cell = mDC",
+   legendGroupNames=c("Adults", "Neonates"), pointColor=c(2,4),
+   testTitleCEX=.8, nCEX=.8, pCEX=.8, legendColor=c(2,4), legendCEX=.7)
> mtext("COMPOSITION OF PFD=3 DATA", side=3, line=2)
```



2.5 Ternary plots to compare groups on pfdData

```
> data(pfdDF)
> pfdDataSubset = subset(pfdDF, stim=="LPS" & concGroup==3 & cell=="mDC")
> # Use data helper function, makeTernaryData()
> ternaryData = makeTernaryData(pfdDataSubset, 1, 2, 3:4)
> colnames(ternaryData) = c("PFD1", "PFD2", "PFD3-4")
> adultPFDData = subset(pfdDataSubset, group=="adult", select=c(PFD1:PFD4))
> neoPFDData = subset(pfdDataSubset, group=="neonate", select=c(PFD1:PFD4))
> groupPFDDataList = list(adultPFDData, neoPFDData)
> pfdGroupStatsList = computePFDGroupStatsList(groupPFDDataList, pfdValues=1:4,
+   numDigitsMean=3, numDigitsSD=2)
> groupNames = c("Adults", "Neonates")
> legendNames = legendPFDStatsGroupNames(pfdGroupStatsList, groupNames)

> # Load the package, vcd, for the ternaryplot() fcn
> library(vcd)
> ternaryplot(ternaryData, cex=.5, col=as.numeric(pfdDataSubset$group)*2,
+   main="TERNARY PLOT OF PFD DATA,
+   Stimulation = LPS, Concentration Group = 3, Cell = mDC")
> grid_legend(0.8, 0.7, pch=c(20,20), col=c(2,4), legendNames,
+   title = "Group (n), mean/sd:", gp=gpar(cex=.8))
```


TERNARY PLOT OF PFD DATA,
Stimulation = LPS, Concentration Group = 3, Cell = mDC



2.6 Bar plots to compare groups on profileData

```
> data(profileDF)
> profileDataSubset = subset(profileDF, stim=="LPS" & concGroup==3 & cell=="mDC")
> profileColumns = 1:16
> # Use data helper function, makeBarplotData()
> barplotData = makeBarplotData(profileDataSubset, profileColumns, groupVariableName="group")
> barplotDataWithLegend = cbind(barplotData, NA, NA)
> barColors = gray(0:15/15)[16:1]

> barplot(barplotDataWithLegend, col=barColors,
+ main="Stimulation = LPS, Concentration Group = 3, Cell = mDC")
> legendNames = rownames(barplotData)
> legend(2.75,100,legend=legendNames[16:1],col=barColors[16:1],cex=.8,pch=20)
> mtext("BAR PLOT OF PROFILE DATA", side=3, line=3)
```



3 A Boxplot Example to Illustrate Multiple Groups and Point Colors

As many as 11 groups can be compared using the `GroupListBoxplot` function. Data points overlaid on the boxplots can be a single color, a color for each group (by specifying `pointColor` as a vector), varying colors for categories on the x-axis and for groups (by specifying `pointColor` as a matrix, where rows represent groups and cols represent x-cats), or a color can be specifically assigned for each point on the plot (by setting `pointColor` equal to a list of matrices, where each matrix represents a group, rows represent subjects and cols represent x-cats). In the following made-up example, we compare five groups and use point coloring to identify gender.

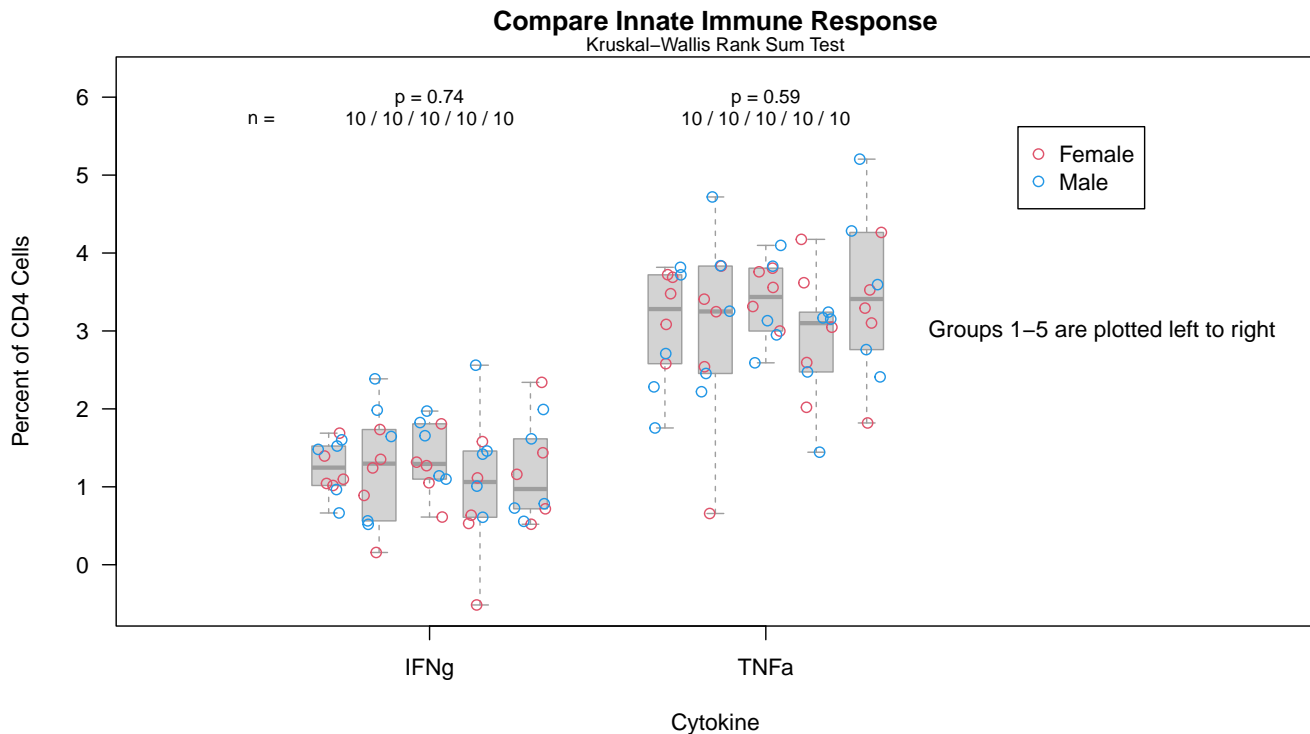
```
> # Create Example data list
> group1 = as.data.frame(cbind(rnorm(10,1.2,.6),rnorm(10,3.2,.8)))
> group2 = as.data.frame(cbind(rnorm(10,1.2,.6),rnorm(10,3.2,.8)))
> group3 = as.data.frame(cbind(rnorm(10,1.2,.6),rnorm(10,3.2,.8)))
> group4 = as.data.frame(cbind(rnorm(10,1.2,.6),rnorm(10,3.2,.8)))
> group5 = as.data.frame(cbind(rnorm(10,1.2,.6),rnorm(10,3.2,.8)))
> dataList = list(group1,group2,group3,group4,group5)
```

```

> # Create pointColor list
> colorGroup1 = cbind(c(rep(2,5),rep(4,5)),c(rep(2,5),rep(4,5)))
> colorGroup2 = cbind(c(rep(2,5),rep(4,5)),c(rep(2,5),rep(4,5)))
> colorGroup3 = cbind(c(rep(2,5),rep(4,5)),c(rep(2,5),rep(4,5)))
> colorGroup4 = cbind(c(rep(2,5),rep(4,5)),c(rep(2,5),rep(4,5)))
> colorGroup5 = cbind(c(rep(2,5),rep(4,5)),c(rep(2,5),rep(4,5)))
> pointColorList = list(colorGroup1,colorGroup2,colorGroup3,colorGroup4, colorGroup5)

> GroupListBoxplot(dataList, xlabel="Cytokine", ylabel="Percent of CD4 Cells",
+ xAxisLabels=c("IFNg","TNFa"), mainTitle="Compare Innate Immune Response",
+ legendGroupNames=c("Female","Male"), legendColors=c(2,4), boxOutliers=FALSE,
+ pointColor=pointColorList, testTitleCEX=.8, nCEX=.8, pCEX=.8)
> text(3,3,"Groups 1-5 are plotted left to right")

```



4 The *StackedData* Class

"Stacked" data refers to data originating from an ICS Flow Cytometry experiment, after gating has been applied. The marker categories (for example, cytokine combinations) are "stacked" in the table of data. An example subset:

```

> # load an .rda file of stacked data
> data(adultsNeonates)
> adultsNeonates[1:16,]

  id group stim concGroup cell percentAll count totalCount percentReactive
1 a2004 adult LPS      3 mDC      0.00      0      700      0.000000

```

2	a2004	adult	LPS	3	mDC	0.43	3	700	0.940625
3	a2004	adult	LPS	3	mDC	0.00	0	700	0.000000
4	a2004	adult	LPS	3	mDC	21.86	153	700	47.818750
5	a2004	adult	LPS	3	mDC	0.00	0	700	0.000000
6	a2004	adult	LPS	3	mDC	0.29	2	700	0.634375
7	a2004	adult	LPS	3	mDC	0.00	0	700	0.000000
8	a2004	adult	LPS	3	mDC	19.71	138	700	43.115625
9	a2004	adult	LPS	3	mDC	0.00	0	700	0.000000
10	a2004	adult	LPS	3	mDC	0.00	0	700	0.000000
11	a2004	adult	LPS	3	mDC	0.00	0	700	0.000000
12	a2004	adult	LPS	3	mDC	2.00	14	700	4.375000
13	a2004	adult	LPS	3	mDC	0.14	1	700	0.306250
14	a2004	adult	LPS	3	mDC	0.29	2	700	0.634375
15	a2004	adult	LPS	3	mDC	1.00	7	700	2.187500
16	a2004	adult	LPS	3	mDC	54.28	380	700	NA

cytCombo

1	TNFa+IL6+IL12+IFNa+
2	TNFa+IL6+IL12+IFNa-
3	TNFa+IL6+IL12-IFNa+
4	TNFa+IL6+IL12-IFNa-
5	TNFa+IL6-IL12+IFNa+
6	TNFa+IL6-IL12+IFNa-
7	TNFa+IL6-IL12-IFNa+
8	TNFa+IL6-IL12-IFNa-
9	TNFa-IL6+IL12+IFNa+
10	TNFa-IL6+IL12+IFNa-
11	TNFa-IL6+IL12-IFNa+
12	TNFa-IL6+IL12-IFNa-
13	TNFa-IL6-IL12+IFNa+
14	TNFa-IL6-IL12+IFNa-
15	TNFa-IL6-IL12-IFNa+
16	TNFa-IL6-IL12-IFNa-

The *StackedData* class has 6 slots for data:

1. stackedData, a `data.frame`
2. profileData, a `data.frame`
3. marginalData, a `data.frame`.
4. pfdData, a `data.frame`.
5. pfdPartsData, a list of `data.frame`'s.
6. markers, a `matrix`.

The `stackedData` often is provided by the lab as a table in CSV format.

The `markers` data is a matrix of 0's and 1's in which each row of the matrix represents one of the possible marker combinations in the "stacked" data and each column represents a marker. The `stackedData` data frame

should be sorted within each subset of interest (for example, subjectID, cellType, stimulation, and concentration) so that the marker combinations match the order of the rows in the `markers` matrix before using the *StackedData* class methods to compute summary data.

The *StackedData* class provides methods to compute the matrix of `markers`, and the `profileData`, `marginalData`, `pfdData`, and `pfdPartsData` data frames.

4.1 Marker Data: `markers`

The marker categories (eg. cytokine combinations) should be stacked in this order within each subset of interest in the `stackedData`.

```
> data(markerMatrix)
> print(markerMatrix)
```

	TNFa	IL6	IL12	IFNa
[1,]	1	1	1	1
[2,]	1	1	1	0
[3,]	1	1	0	1
[4,]	1	1	0	0
[5,]	1	0	1	1
[6,]	1	0	1	0
[7,]	1	0	0	1
[8,]	1	0	0	0
[9,]	0	1	1	1
[10,]	0	1	1	0
[11,]	0	1	0	1
[12,]	0	1	0	0
[13,]	0	0	1	1
[14,]	0	0	1	0
[15,]	0	0	0	1
[16,]	0	0	0	0

4.2 The Other Data Slots: `stackedData`, `profileData`, `marginalData`, `pfdData`, `pfdPartsData`

Examples of the data held in these slots have been given elsewhere in this document: `stackedData` (Section 4), `profileData` (Section 2.1), `marginalData` (Section 2.2), `pfdData` (Section 2.3), and `pfdPartsData` (Section 2.4).

5 Using the *StackedData* Class

In this section we create a *StackedData* object and use the class methods to compute profile and summary data.

5.1 Create a `stackedDataObject`

```
> # create the StackedData object based on the adultsNeonates data
> stackedDataObject = new("StackedData", stackedData = adultsNeonates)
```

5.2 Create the markers

The `stackedData` can either include or exclude the "all-negative" combination of markers (eg. the TNFa-IL6-IL12-IFNa- row). If that row is included, set the `computeMarkers` parameter `includeAllNegativeRow` to `TRUE`, else set it to `FALSE`. If you create this matrix on your own, create a matrix which matches the data.

```
> markerNames = c("TNFa", "IL6", "IL12", "IFNa")
> markers = computeMarkers(markerNames, includeAllNegativeRow=TRUE)
> markers(stackedDataObject) = markers
```

5.3 Sort the stackedData

The `stackedData` should be sorted within subsets of interest (such as: `subjectID`, `cellType`, `stimulation`, and `concentration`) so that the data within each subset matches the marker combinations specified by the marker matrix. Usually the lab provides data sorted in this way. If not, sort it before applying the *StackedData* class methods.

5.4 Create the profileData

```
> # The byVarNames specify the subsets in the data
> byVarNames = c("stim", "concGroup", "cell")
> profileData = computeProfileData(stackedDataObject, byVarNames, "id", "percentAll", "group")
> profileData(stackedDataObject) = profileData
```

5.5 Create the marginalData

```
> # The byVarNames specify the subsets in the data
> byVarNames = c("stim", "concGroup", "cell")
> marginalData = computeMarginalData(stackedDataObject, byVarNames, "id", "percentAll", "group")
> marginalData(stackedDataObject) = marginalData
```

5.6 Create the pfdData

```
> # The byVarNames specify the subsets in the data
> byVarNames = c("stim", "concGroup", "cell")
> pfdData = computePFDDData(stackedDataObject, byVarNames, "id", "percentAll", "group")
> pfdData(stackedDataObject) = pfdData
```

5.7 Create the pfdPartsData

```
> # The byVarNames specify the subsets in the data
> byVarNames = c("stim", "concGroup", "cell")
> pfdPartsData = computePFDPData(stackedDataObject, byVarNames, "id", "percentAll", "group")
> pfdPartsData(stackedDataObject) = pfdPartsData
```

6 Acknowledgments

We would like to thank Chris Wilson, et al, for allowing us to use the adultsNeonates data subset as the example data set in this package.

References

TR Kollmann, J Crabtree, A Rein-Weston, D Blimkie, F Thomai, X Wang, J Furlong, E Fortuno III, A Hajjar, N Hawkins, S Self, and C Wilson. The neonatal innate immune system is not less responsive than the adult but responds differently. *J Immunology*, 183:7150–7160, 2009.