flowTrans: A Package for Optimizing Data Transformations for Flow Cytometry

Greg Finak, Raphael Gottardo

March 30, 2012

greg.finak@ircm.qc.ca, raphael.gottardo@ircm.qc.ca

Contents

1 Licensing **2**

2 Overview **2**

3 Example: Optimizing Data Transformations for the GvHD Data Set **2**

4 Future Improvements **4**

1 Licensing

Under the Artistic License, you are free to use and redistribute this software.

> Greg Finak, Juan–Manuel Perez, Andrew Weng, Raphael Gottardo. Optimizing Data Transformations for Flow Cytometry. (In Preparation)

2 Overview

flowTrans is an R package for optimizing the parameters of the most commonly used flow cytometry data transformations with the goal of making the data appear more normally distributed, have decreased skewness or kurtosis. The end result is often a transformation that makes flow cytometry cell populations appear more symmetric, better resolved, thus simplifying population gating. In high throughput experiments, flowTrans reduces the variability in the location of discovered cell populations when compared to untransformed data, or data transformed with default transformation parameters.

The package implements a single user – callable function `flowTrans`, which takes a *flowFrame* as input, as well as the names of the dimensions to be transformed, and the name of the transformation function to be applied. There are four transformations available in `flowTrans`. The transforms are multivariate, are meant to transform two or more dimensions, and will estimate a common set of transformation parameters for all dimensions simultaneously via profile maximum likelihood, with the global distribution of the transformed data tending towards multivariate normal. All the transformation functions in the package are summarized in Table 1.

Function Name	Transformation	Dimensionality	Parameters	Optimization Criteria
mclMultivBoxCox	Box–Cox	Multivariate	Common	Normality
mclMultivArcSinh	ArcSinh	Multivariate	Common	Normality
mclMultivBiexp	Biexponential	Multivariate	Common	Normality
mclMultivLinLog	LinLog	Multivariate	Common	Normality

Table 1: A summary of the transformations that can be called via the `flowTrans` function.

3 Example: Optimizing Data Transformations for the GvHD Data Set

We present a simple example, transforming the samples in the `GvHD` data set. We begin by attaching the data, then transforming each sample in the forward and side scatter dimensions with the parameter–optimized multivariate arcsinh transform.

```
> library(flowTrans)

Scalable Robust Estimators with High Breakdown Point (version 1.3-01)

> data(GvHD)
> transformed<-lapply(as(GvHD[1:4],"list"),function(x)flowTrans(dat=x,fun="mclMultivArcSinh'
```

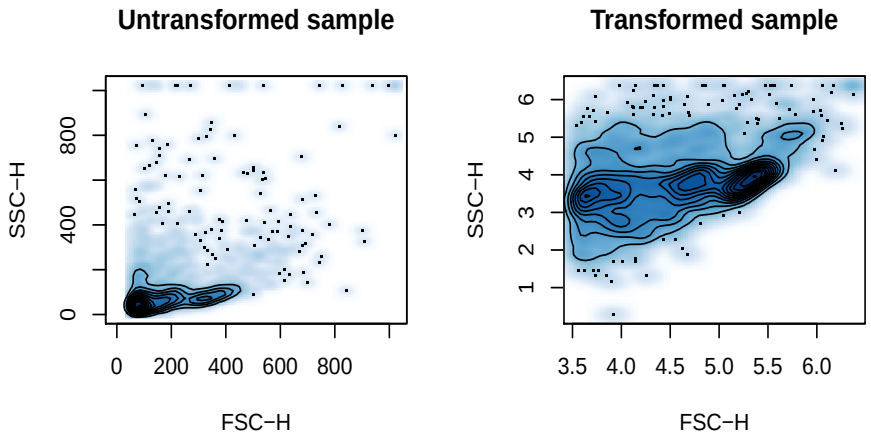
We can extract the parameters used for the transformation of each dimension in each sample. The returned object is a list of transformed samples, each sample is itself composed of a list with the first element corresponding to the transformed *flowFrame*, and the second element a list of the transformation parameters applied to each dimension. For example, to extract the parameters used to transform sample 2, dimension 1 (FSC) we would call `extractParams(x=transformed[[2]],dims=colnames(transformed[[2]])[1])`.

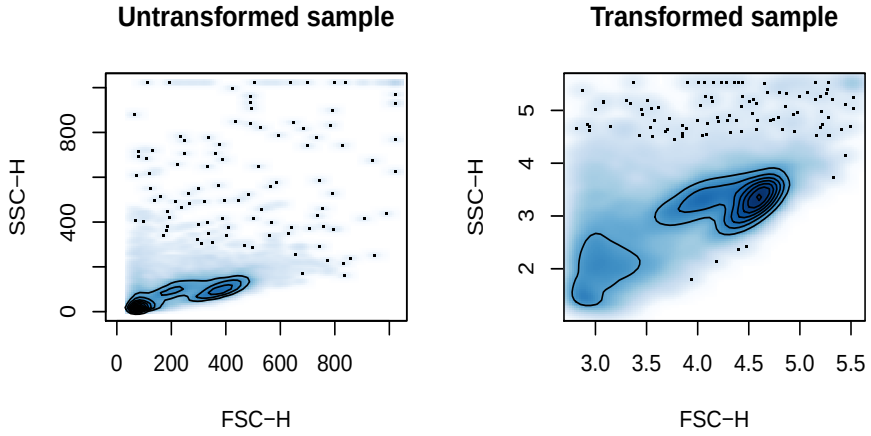
```
> parameters<-do.call(rbind,lapply(transformed,function(x)extractParams(x)[[1]]))
> parameters;

              a          b c
s5a01 0.9999977 0.99999390 0
s5a02 0.0000000 0.28811803 0
s5a03 0.0000000 0.05392773 0
s5a04 1.0000000 0.12207816 0
```

We see that the parameters, a and b, of `arcsinh` vary considerably across the different samples, due to the data-dependence of the optimal transformation. We can plot the transformed and untransformed samples for comparison (samples 2 and 4 in this case).

```
> par(mfrow = c(2, 2))
> plot(GvHD[[2]], c("FSC-H", "SSC-H"), main = "Untransformed sample")
> contour(GvHD[[2]], c("FSC-H","SSC-H"),add=TRUE);
> plot(transformed[[2]]$result, c("FSC-H", "SSC-H"), main = "Transformed sample")
> contour(transformed[[2]]$result,c("FSC-H","SSC-H"),add=TRUE);
> plot(GvHD[[4]], c("FSC-H", "SSC-H"), main = "Untransformed sample")
> contour(GvHD[[4]], c("FSC-H","SSC-H"),add=TRUE);
> plot(transformed[[4]]$result, c("FSC-H", "SSC-H"), main = "Transformed sample")
> contour(transformed[[4]]$result, c("FSC-H","SSC-H"),add=TRUE);
```





Similarly, we can apply the multivariate transformation of the forward and side scatter channels, but return only the parameters, for inclusion in a flowCore workflow.

```
> transformed2<-flowTrans(dat=GvHD[[2]],fun="mclMultivArcSinh",dims=c("FSC-H","SSC-H"),n2f=F
> transformed2

       a        b        c
0.000000 0.288118 0.000000
```

4 Future Improvements

In the near future, we plan to implement inverse transformation functions via S4 classes and objects, such that transformed data can be inverse transformed

without the need to call internal `flowTrans` functions.