BIOINFORMATICS APPLICATIONS NOTE Vol. 24 no. 13 2008, pages 1547–1548
doi:10.1093/bioinformatics/btn224

Gene expression

lumi: a pipeline for processing Illumina microarray

Pan Du*, Warren A. Kibbe and Simon M. Lin

Robert H. Lurie Comprehensive Cancer Center, Northwestern University, Chicago, IL, 60611, USA

Received on December 20, 2007; revised on February 3, 2008; accepted on May 5, 2008

Advance Access publication May 8, 2008

Associate Editor: Joaquin Dopazo

ABSTRACT

Summary: Illumina microarray is becoming a popular microarray platform. The BeadArray technology from Illumina makes its preprocessing and quality control different from other microarray technologies. Unfortunately, most other analyses have not taken advantage of the unique properties of the BeadArray system, and have just incorporated preprocessing methods originally designed for Affymetrix microarrays. *lumi* is a Bioconductor package especially designed to process the Illumina microarray data. It includes data input, quality control, variance stabilization, normalization and gene annotation portions. In specific, the *lumi* package includes a variance-stabilizing transformation (VST) algorithm that takes advantage of the technical replicates available on every Illumina microarray. Different normalization method options and multiple quality control plots are provided in the package. To better annotate the Illumina data, a vendor independent nucleotide universal identifier (nuID) was devised to identify the probes of Illumina microarray. The nuID annotation packages and output of *lumi* processed results can be easily integrated with other Bioconductor packages to construct a statistical data analysis pipeline for Illumina data.

Availability: The *lumi* Bioconductor package, www.bioconductor.org

Contact: dupan@northwestern.edu

1 INTRODUCTION

Due to cost effectiveness and accuracy, the Illumina microarray (BeadArray) is becoming a popular microarray platform (Kuhn *et al.*, 2004). The Illumina BeadArray technology is based on randomly arranged beads, with each bead binding many identical copies of a gene-specific probe. The BeadArray is constructed so that there are roughly 30 randomly positioned replicates on average for each type of bead. This redundant design yields higher confidence calls and more robust estimations compared with other types of microarrays. However, the uniqueness of Illumina microarray design makes preprocessing and quality control steps significantly different from other types of microarrays. Unfortunately, until now, most other analyses have not taken advantage of the unique properties of the Illumina BeadArray system, and have just incorporated preprocessing methods originally designed for Affymetrix microarrays.

Bioconductor is an open source and open development software project (mainly written in R programming language) for the analysis and comprehension of genomic data (Gentleman *et al.*, 2004). To date, there are two other packages (*beadarray* and *BeadExplorer*) in Bioconductor to extend functionalities provided by the Illumina LIMS software named *BeadStudio*. *Beadarray* package (Dunning *et al.*, 2007) is mainly designed for quality control and bead-level analysis of BeadArrays. *BeadExplorer* is aimed to provide data exploration and quality control by leveraging the output of *BeadStudio* and existing algorithms in the *affy* package; it does not take advantage of larger number of technical replicates available on the Illumina microarray in the preprocessing. The design objectives of the new *lumi* package are 2-fold: first, to provide algorithms uniquely designed for Illumina and second, to best utilize existing algorithms and frameworks by following the class infrastructure and gene annotation framework in Bioconductor.

With the comments and contributions from the users and researchers all around the world, the *lumi* package has made big improvements. The current version of *lumi* package includes methods for data import, quality control, preprocessing and gene annotation of Illumina microarray data. Besides supporting the existing algorithms for microarray data, the *lumi* package includes several unique parts: (1) a variance-stabilizing transformation (VST) that utilizes the technical replicates available on the Illumina microarray (Lin *et al.*, 2008); (2) normalization algorithms [including robust spline normalization (RSN) and simple scaling normalization (SSN)] designed for Illumina microarray data and;3) the nucleotide universal identifier (nuID) annotation packages (Du *et al.*, 2007). The nuID annotation packages allow for version- and vendor-independent annotation of each probe. The nuID also uniquely and exactly encodes the original probe sequence through a process that includes error checking.

2 IMPLEMENTATION

2.1 Preprocessing and quality control

The *lumi* package includes one major class: LumiBatch, which is inherited from ExpressionSet class in Bioconductor to enable interoperability with other Bioconductor packages. The class diagram is shown in Figure 1. LumiBatch class includes numerous methods as discussed below. It extends ExpressionSet class by including three elements, `se.exprs`, `beadNum` and `detection`, in `assayData` slot to hold the additional information unique to Illumina microarrays. The `controlData` slot keeps the control probe information, and `QC` slot keeps the quality control summary. A new `history` slot is added to the class to track all the operations made on the LumiBatch object. This provides a convenient container for data provenance. Users can choose to keep the Illumina annotation information outputted by BeadStudio in the `featureData` of the LumiBach object.

*To whom correspondence should be addressed.

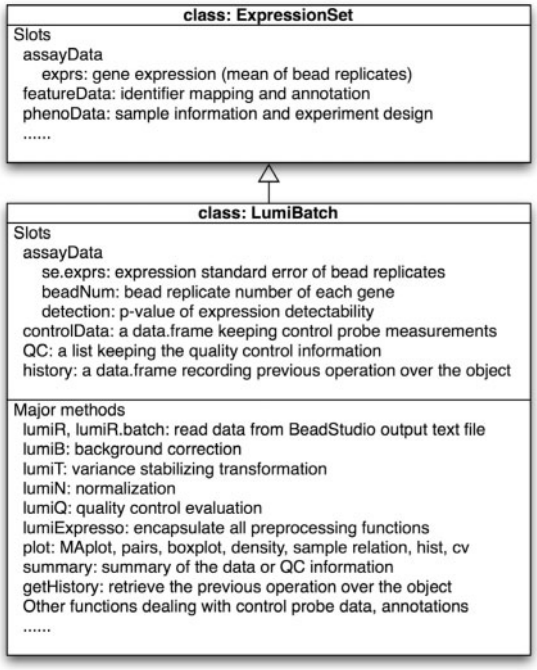

Fig. 1. Object model relationships in the *lumi* package.

There are several major processing methods in the *lumi* package. `lumiR` initializes the LumiBatch object by intelligently reading raw data from all versions of Illumina *BeadStudio* software and a `lumiR.batch` method is designed to read a batch of data files. `lumiB` adjusts the array background; `lumiT` performs variance stabilization of data; `lumiN` normalizes the variance stabilized data and `lumiQ` assesses the data quality. All these methods constitute a preprocessing pipeline. For convenience, a `lumiExpresso` method encapsulates all four methods as one. The methods also include options to call other processing methods previously designed for Affymetrix data. For better visualization and quality control purpose, the *lumi* package also provides different kinds of plot functions. These plot functions can handle both expression and control probe data. Please refer to the tutorial and function help files for more details.

2.2 Annotation packages

A good annotation package is important for interpreting the analysis results. Illumina *BeadStudio* uses TargetID or ProbeId to identify individual genes. However, the identifiers are not consistent among different versions of BeadArray chips or even between different batches. This causes difficulties when combining the results using different versions of the chips. We designed a nuID (Du *et al*., 2007) to address these issues. A nuID is a loss-less compression of the 50mer oligonucleotide sequence and contains error checking and self-identification code.

The Illumina annotation packages were built by using Bioconductor annotation tools with the nuID of each probe used as the identifier. The mappings from TargetID or ProbeId to nuID are also included in the annotation package. Because all the Illumina microarrays use 50mers, by using the nuID universal identifier, we are able to build one annotation database for different versions

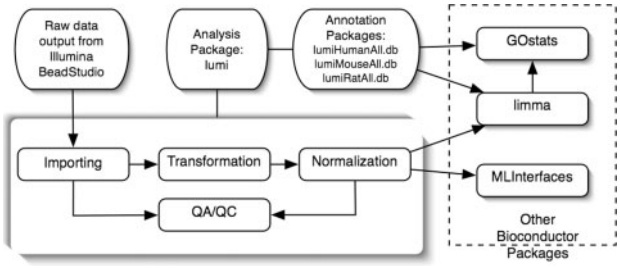

Fig. 2. Flow chart of the use case.

```
> library(lumi)          # load the library

> # Read Bead Studio output file and create a LumiBatch object
> example.lumi <- lumiR('exampleData.txt')

> ## summary of data
> example.lumi
> ## summary of quality control information
> summary(example.lumi, QC )

> ## preprocessing and quality control after normalization
> lumi.N <- lumiExpresso(example.lumi)

> ## make different plots: pairs, MAplot, sampleRelation, boxplot
> ## Make further analysis based on processed data lumi.N
```

Fig. 3. Example R code for Illumina data preprocessing.

of the chips of same species. Moreover, a nuID can be directly converted to the probe sequence, and used to get the most updated refSeq matches and annotations. Annotation packages for all current Illumina expression chips (the package names are prefixed with 'lumi', and followed by the species name and version number, e.g. *lumiHumanAll.db*) can be downloaded from Bioconductor.

3 A USE CASE

Figure 2 shows the data processing flow chart of the use case. The R source code for preprocessing is shown in Figure 3. Since the classes in *lumi* package is extended from the class ExpressionSet, lots of data analysis packages in Bioconductor can be directly applied to the results of *lumi* methods. Figure 2 graphs a scenario using the *lumi* package plus *limma*, *GOstats* and *MLInterfaces*. The more details of the implementation can be found in the tutorial of the *lumi* package.

In conclusion, the *lumi* package provides class infrastructure and associated methods to construct an Illumina analysis workflow pipeline starting with raw data through functional analysis.

Conflict of Interest: none declared.

REFERENCES

Du,P. *et al*. (2007) nuID: a universal naming scheme of oligonucleotides for Illumina, Affymetrix, and other microarrays. *Biol. Direct*, **2**, 16.

Dunning,M.J. *et al*. (2007) beadarray: R classes and methods for Illumina bead-based data. *Bioinformatics*, **23**, 2183–2184.

Gentleman,R.C. *et al*. (2004) Bioconductor: open software development for computational biology and bioinformatics. *Genome Biol*., **5**, R80.

Kuhn,K. *et al*. (2004) A novel, high-performance random array platform for quantitative gene expression profiling. *Genome Res*., **14**, 2347–2356.

Lin,S.M. *et al*. (2008) Model-based variance-stabilizing transformation for Illumina microarray data. *Nucleic Acid Res*., **36**, e11.