Towards an Optimized Illumina Microarray Data Analysis Pipeline

Pan Du, Simon Lin

Robert H. Lurie Comprehensive Cancer Center, Northwestern University

Oct 01, 2007
Outline

• Introduction of Illumina Beadarray technology
• Lumi package overview
• nuID and related annotation packages
• VST (variance stabilizing transform)
• RSN (robust spline normalization)
Illumina BeadArray Technology

Uniform pits are etched into the surface of each substrate to a depth of approximately 3 microns prior to assembly.

Each type of bead has about 30 technique replicates on average.

Beads are randomly assembled and held in these microwells.

Multiple arrays on the same slide.

Cost: < $200

Slide: 2 x 7cm

Bead: 3 µm
Each array is different
(Previous) Concerns

<table>
<thead>
<tr>
<th>Challenges</th>
<th>Illumina Solutions</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Uneven distribution of BG (air bubble and washing)</td>
<td>• Larger number of beads</td>
</tr>
<tr>
<td>• Contamination of debris</td>
<td>• Random distribution of beads</td>
</tr>
<tr>
<td>• Scratches on the surface</td>
<td></td>
</tr>
<tr>
<td>Spot morphology and uniformity</td>
<td>Coated beads instead of printing</td>
</tr>
<tr>
<td>Array manufacturing defect</td>
<td>Tested in the decoding process</td>
</tr>
<tr>
<td>Failure in labeling of mRNA</td>
<td>Labeling control on array</td>
</tr>
<tr>
<td>Scanning conditions</td>
<td>Still a concern ?</td>
</tr>
<tr>
<td>Probe Specificity</td>
<td>50-mer design</td>
</tr>
<tr>
<td>Normalization issues</td>
<td>6 to 12 arrays on the same slide</td>
</tr>
</tbody>
</table>
## Affymetrix vs. Illumina

<table>
<thead>
<tr>
<th></th>
<th>Affymetrix</th>
<th>Illumina</th>
</tr>
</thead>
<tbody>
<tr>
<td>Redundancy</td>
<td>Low (usually one)</td>
<td>High (tens of replicates)</td>
</tr>
<tr>
<td>Probe location</td>
<td>Fixed</td>
<td>Random</td>
</tr>
<tr>
<td>Probe length</td>
<td>25 mer</td>
<td>50 mer</td>
</tr>
<tr>
<td>Probe vs. gene</td>
<td>probe → probe-set → gene</td>
<td>probe → gene</td>
</tr>
<tr>
<td>Array layout</td>
<td>One array per chip</td>
<td>Multiple arrays per chip</td>
</tr>
</tbody>
</table>
Overview of *lumi* package
Design Objectives of *lumi* Package

- To provide algorithms uniquely designed for Illumina
- To best utilize the existing functionalities by following the class infrastructure and identifier management framework in Bioconductor
Object Models

- Design based on the S4 Classes.
- One major class: lumiBatch
- Compatible with other Bioconductor packages;
Analysis Pipeline
Example Code

> # load the library
> library(lumi)

> # specify the file name output from Bead Studio
> fileName <- 'Barnes_gene_profile.txt'
> # Read the data and create a LumiBatch object
> example.lumi <- lumiR(fileName, lib='lumiHumanV1')

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

> ## preprocessing and quality control after normalization
> lumi.N.Q <- lumiExpresso(example.lumi)
> ## summary of quality control information after preprocessing
> summary(lumi.N.Q, QC)

> ## plot different plots
> pairs(lumi.N.Q)
> plot(lumi.N.Q, what='sampleRelation')
> boxplot(lumi.N.Q)

> # Extract expression data for further processing
> dataMatrix <- exprs(lumi.N)
nuID and Illumina Annotation Packages
What is nuID

• nuID is the abbreviation of Nucleotide Universal Identifier
• nuID is a novel identifier for oligos, ideal for oligonucleotide-based microarrays
Microarray Information Flow

GO:0051301
Cell Division

entrezID: 19645
symbol: Rb1

retinoblastoma 1
1 ggcggccccgc gtcgggttttt ctctggggga gttcccatta tttttgttaac gggantcggg
tgagagggg gctgcgcgcg cgtgcgcgcg cgaccccgcg ccttcggcccc gcgtggcctc
gtgcgcgcgc ggcggccccgc gttgcgccttc atgcgcgccca aagccccgcc cagagcccgc
gcccccgcgc cccgcccccc cccccggcgc gccctccgggg aggcgcaccgc gcggcaggac
gagcgccccgc aagacgcctcc gttccgcaggg cttcagtggta aagaaattga agaaccgga
301 tttaattgcat tatgtcaaaa gttataggtta cccgatcatg tcagacaag aagcttggtga
...

ID: ??

10/1/07
Bioconductor 2007, Chicago
observed

Probes 1 to N

stable

dynamic

inferred

Gene 1 to M

GO 1 to P

Entiry Mapping
For Illumina microarrays, TargetID was used as the primary ID in the NCBI GEO database.
Challenges of Target IDs

• Not unique: “GI_28476905” and “scl0076846.1_142” are the same gene on Mouse_Ref-8_V1 chip.
  -- Synonyms.

• Not stable over time: “GI_21070949-S” in the Mouse_Ref-8_V1 chip but as “scl022190.1_154-S” in the later Mouse-6_V1 chip.
  -- IDs can be recycled or retired.

• Not universal across manufacturers
  -- Homonyms.

• Not interpretable without metadata: However, metadata (lookup table) is not always available in reality.
How to ensure one ID per item?

– How to enforce 1:1 mapping?

– How can it be globally unique?

– How can it be permanent?
Solution I: Central Authority

• GenBank/ EMBL / DDBJ
• May help enforcing 1:1 mapping of an ID and an entity
  – HUGO Nomenclature Committee
  – “Giving unique and meaningful names to every human gene”
• May be infeasible either technically or socially
Solution II: nuID

- Unique, guaranteed
  - Each name identifies only one entity
  - Inherently enforces 1:1 mapping
  - Uniquely resolvable

- Globally unique, guaranteed
  - Decentralized
  - No ID registry necessary

- Permanent, guaranteed

- Carries information about the entity
  - White box
  - No need for a lookup table
nuID: the idea

• Sequence itself as the ID
• Combined with the following four features
  – Compression: make it shorter
  – Checksum
    • Prevent transmission error
    • Provide self-identification
  – Encryption: in cases where the sequence identity is proprietary
  – Digital watermark: identify issuer
How does nuID work?

Figure 2
The encoding and decoding process of nuID. The solid arrows represent the encoding process, and the dashed arrows represent the decoding process. The bold-italic number $II$ is the numeric value of the checking code "L". The "AA" at the end of sequence is the padded nucleotides.
Example of nuID

<table>
<thead>
<tr>
<th>Array Type</th>
<th>Manufacturer’s Proprietary Identifier</th>
<th>Nucleotide Sequence</th>
<th>nuID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Affymetrix Human</td>
<td>206064_s_at_probe1</td>
<td>TGTATATGTCGTTTTTCTTACCCC</td>
<td>a7M7ev98VQ</td>
</tr>
<tr>
<td>Illumina Human</td>
<td>GI_23097300-A</td>
<td>GCTTCATCGCTTCCCAGGGGCCTCCTGTCACCAACTACATGAGCTACACG</td>
<td>cn0dn1Sqd8UHE4nEY</td>
</tr>
<tr>
<td>Illumina Mouse</td>
<td>TRBV23_AE000664_T_cell_receptor_beta_variable_23_106-S</td>
<td>GCCCTTCAAGTGAAGAGCACAAGTCATGTATATGGTATAGTTCATGGT</td>
<td>9hX2C4BET08zRMT0s</td>
</tr>
</tbody>
</table>
Performance of checksum

Table 2: The error detection power of the nulD checksum algorithm (N = 21)

<table>
<thead>
<tr>
<th>L</th>
<th>1-character</th>
<th>2-character</th>
<th>3-character</th>
<th>Random</th>
</tr>
</thead>
<tbody>
<tr>
<td>25mer</td>
<td>0.97780</td>
<td>0.97918</td>
<td>0.98689</td>
<td>0.99924</td>
</tr>
<tr>
<td>50mer</td>
<td>0.97724</td>
<td>0.97838</td>
<td>0.98607</td>
<td>0.99997</td>
</tr>
<tr>
<td>100mer</td>
<td>0.97894</td>
<td>0.97825</td>
<td>0.98617</td>
<td>1*</td>
</tr>
</tbody>
</table>

L and N are defined in Equation (3) and (4) in Methods. The column "1-character" is the error detection rate of an nulD with only one character mutated. Similar definition for column "2-character" and "3-character". "Random" column is error detection rate of a random ASCII string. The optimum detection power is 1.0.

* We realize the detection of nulD's for 100mers is not guaranteed, but in none of our simulations did we ever encounter a randomly assembled string that was a valid nulD.
Implementation of nuID

• We have build nuID based annotation packages for all Illumina expression chips.
• We have set up a website for nuID conversion and check latest annotation for the probe.
• The implementation is also included in the lumi package.
Illumina Annotation Packages

• Produced nuID indexed annotation packages for all Illumina expression chips. (named as lumiHumanV1, …)

• In the future, the packages will be based on the most updated RefSeq matches with nuID and their annotations.
Summary

• For microarray reporting: Probe-level data is preferred over gene-level data.

• nuID is universal, globally unique, and permanent.

• Do not need a central authority to issue nuID.
Variance Stabilization Transformation
Variance Stabilization

- General assumption of statistical tests to microarray data: variance is independent of intensity
- In reality, larger intensities tend to have larger variations
- Current implementation:
  - Log2 transform is widely used
- Variance stabilization through a generalized log transformation
Example of Mean and Variance Relation

(A) Raw

(B) Log2 transformed
Variance Stabilization

• Mathematical model

\[ Y = \alpha + \mu e^{\eta} + \varepsilon \]  \tag{1}

• Asymptotic variance-stabilizing transformation

\[ h(y) = \int_{1}^{y} \frac{1}{\sqrt{v(u)}} du \]

• Mean and variance relation

\[ h(y) = \begin{cases} 
1/c_1 \arcsin h(c_2/\sqrt{c_3} + c_1/\sqrt{c_3} \cdot y), & \text{when } c_3 > 0 \\
1/c_1 \ln(c_2 + c_1 \cdot y), & \text{when } c_3 = 0 
\end{cases} \]
VSN (Variance Stabilizing Normalization)

• Estimation the mean and variance relation based on limited technique replicates
• Combines variance stabilizing and normalization based on the limited replicates across chips
• Assumption: most genes are not differentially expressed
• Sometimes unstable due to the above reasons.
Variance Stabilizing Transformation (VST)

Illumina BeadArray technology enables better variance stabilizing

Better fit the relations between mean and standard deviation

Relations between log2 and VST (arcsinh)
Variance Stabilization of the Technical Replicates
Comparison of Log2, VSN and VST

<table>
<thead>
<tr>
<th></th>
<th>log2</th>
<th>VSN</th>
<th>VST</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error model for each individual array</td>
<td>None</td>
<td>Equation (1)</td>
<td>Equation (1)</td>
</tr>
<tr>
<td>Estimated from</td>
<td>None</td>
<td>Between-array replicates</td>
<td>Within-array replicates</td>
</tr>
<tr>
<td>Requires built-in normalization</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Negative value</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Parameter estimation method</td>
<td>Fixed mathematical transformation</td>
<td>Maximum likelihood integrated with normalization</td>
<td>Linear fitting</td>
</tr>
<tr>
<td>Assumptions of the replicates</td>
<td>None</td>
<td>Most of the genes are not differentially expressed; thus, they can be treated as replicates.</td>
<td>No such assumption required because the probes are in the same array.</td>
</tr>
<tr>
<td>Observed or assumed replicates</td>
<td>Not used</td>
<td>Usually less than a dozen</td>
<td>Usually over 30</td>
</tr>
</tbody>
</table>
Robust Spline Normalization
Robust Spline Normalization (RSN)

- Quantile normalization:
  - Pros: computational efficiency, preserves the rank order
  - Cons: The intensity transformation is discontinuous
- Loess and other curve-fitting based normalization:
  - Pros: continuous
  - Cons: cannot guarantee the rank order. Strong assumption (majority genes unexpressed and symmetric distributed)
- RSN combines the good features of the quantile and loess normalization
Comparison of curve fitting and quantile normalization

<table>
<thead>
<tr>
<th></th>
<th>Curve fitting based normalization</th>
<th>Quantile normalization</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Assumption</strong></td>
<td>Most genes are not differentially expressed.</td>
<td>All samples have the same distribution.</td>
</tr>
<tr>
<td><strong>Approximation</strong></td>
<td>Based on curve fitting</td>
<td>Replaced by the average of the probes with the same rank</td>
</tr>
<tr>
<td><strong>Problems</strong></td>
<td>Does not work well when lots of genes are differentially expressed.</td>
<td>Will lose small difference between samples, and the change is un-recoverable. Normalize across all samples, memory intensive.</td>
</tr>
<tr>
<td><strong>Strengths</strong></td>
<td>The value mapping is continuous. Normalize in pairwise, memory save.</td>
<td>Rank invariant Computationally efficient</td>
</tr>
</tbody>
</table>
Robust Spline Normalization (RSN)

• Combining the strength of curve fitting and quantile normalization
  – Continuous mapping
  – Rank invariant
  –Insensitive to differentially expressed genes.

• Basic Ideas of RSN
  – Perform a quantile normalization of the entire microarray dataset for the purpose of estimating the fold-changes between samples
  – Fit a weighted monotonic-constraint spline by Gaussian window to down-weight the probes with high fold-changes
  – Normalize each microarray against a reference microarray
Algorithms Evaluation
Evaluation Data Sets

• Barnes data: (Barnes, M., et al., 2005)
  – measured a dilution series (two replicates and six dilution ratios: 100:0, 95:5, 75:25, 50:50, 25:75 and 0:100) of two human tissues: blood and placenta.
Performance Evaluation

VST improves the concordance between the expression profiles and the real dilution ratio profiles

Based on Barnes titration data
Conclusions and Future Plan

• Lumi package provides a pipeline of Illumina microarray preprocessing and annotation
• Provide algorithms uniquely designed for Illumina
• Options to use other traditional algorithms and compatible with other Bioconductor packages
• In the future,
  – enhance the quality control part
  – extend the lumi package to other Illumina data:
    • DNA copy number analysis
    • Methylation profiling
    • SNP and genotyping
Acknowledgements

• Robert H. Lurie Comprehensive Cancer Center, Northwestern University
  – Warren A. Kibbe and other members in the Bioinformatics group
  – Nadereh Jafari, microarray core
• European Bioinformatics Institute, UK
  – Wolfgang Huber
• The Walter and Eliza Hall Institute of Medical Research, Australia
  – Gordon Smyth