Documentation of the RMAGEML package

Steffen Durinck‡∗, Joke Allemeersch‡†, Vincent J Carey¶, Yves Moreau‡, Bart De Moor‡

April 21, 2009

and ¶Channing Laboratory, Brigham and Women’s Hospital, 75 Francis Street, Boston 02115, USA

Contents

1 Introduction 2

2 Prerequisites 2

3 Getting started 3

4 Import to marray packages 3

4.1 One step import and creation of an marrayRaw object from MAGE-ML files

4.2 Creation of a Gnames marrayInfo object 7

4.3 Creation of an marrayLayout object 8

4.4 Make an marrayRaw object 9

5 Import to limma package 9

5.1 One step import and creation of a limma RGList object from MAGE-ML files

5.2 Creating the genes dataframe of an RGList object 12

5.3 Make an RGList object 12

∗Steffen.Durinck@esat.kuleuven.ac.be
†Joke.Allemeersch@esat.kuleuven.ac.be
1 Introduction

MAGE-ML or Microarray Gene Expression Markup Language is a language designed to describe and exchange information about microarray experiments. MAGE-ML is based on XML and can describe microarray designs, microarray experiment setups, gene expression data, and data analysis results.

This package provides the link between MAGE-ML files and BioConductor. It gives the possibility to read in MAGE-ML files that describe cDNA microarray experiments. The functions convert the MAGE-ML files into the customary BioConductor objects (i.e., mareyLayout, marrayInfo and marrayRaw objects or limma RGList objects).

Here we give a short introduction to the Microarray and GeneExpression Object Model (MAGE-OM) and how we implemented the extraction of information necessary to make BioConductor objects. For a full description of MAGE-OM, we refer to the Gene Expression Specification: http://www.omg.org/cgi-bin/doc?formal/03-02-03


In MAGE-ML these translate into packages with the same name. The packages needed for building BioConductor objects are BioAssayData, BioAssay, BioMaterial, BioSequence, ArrayDesign, and DesignElement.

The DesignElement package contains a mapping of Features, which are the actual features present on the array, to Reporters, the reporter a feature represents. The DesignElement package also provides a mapping from Reporters to their corresponding BioSequence references. These BioSequence objects are characterized by their name and database entries in the BioSequence package. The ArrayDesign package contains information on the layout of the array. From this package, we can derive the position of each Feature on the array in terms of Zone (block or grid) and row and column within each Zone. The BioAssayData package describes the feature references that were assayed and the measured and derived QuantitationTypes. The BioAssay package describes the different steps in the microarray experiment. The last package used to make BioConductor objects is the BioMaterial package and describes how a sample is treated to obtain, for example, labeled samples used for hybridization.

2 Prerequisites

The RMAGEML package depends on SJava(>= 0.68) and a Java VM, e.g. j2resdk1.4.0. Other dependencies are as the Java-MAGEstk API and Java Xerces included in the package itself.
3 Getting started

Installing the package. The package can be installed as a normal R package: download the RMAGEML_2.0.4.tar.gz package and under Unix use the command

R CMD INSTALL RMAGEML_2.1.0.tar.gz.

The equivalent command for Windows is

Rcmd INSTALL RMAGEML_2.1.0.zip.

The package automatically loads the Biobase and marrayInput packages from BioConductor and the SJava libraries, so these should be installed as well.

Starting R. Before starting R one should be aware that the RMAGEML package uses SJava and that SJava requires to set the LD_LIBRARY_PATH environment variable before starting R.

Without setting this variable the package won’t work.

Loading the package. You can load the package into R by typing

> library(RMAGEML)

4 Import to marray packages

4.1 One step import and creation of an marrayRaw object from MAGE-ML files

In the marray packages of BioConductor the design of an array experiment is typically described by an marrayLayout and marrayInfo object. The function importMAGEML parses all MAGE-ML files present in the directory, which is given as a parameter to the function. From these files it creates an marrayLayout object, containing the Layout of one type of microarrays, and an marrayInfo object containing the gene names and database entries of the features spotted on the array. The name of the database to which the entries refer, is given in the ‘notes’ slot of the Gnames object. Next the function will extract the raw data values and output a complete marrayRaw object as a result.

The function can be tested on the MEXP-14 dataset. This example is available from ArrayExpress at [http://www.ebi.ac.uk/arrayexpress/](http://www.ebi.ac.uk/arrayexpress/)

If one knows which DesignElement Dimension, QuantitationType Dimension and Quantitation Types are required, the import function can be used as:
> datadir <- system.file("MAGEMLdata", package = "RMAGEML")
> raw <- importMAGEML(directory = datadir, package = "marray",
+     name.Rf = "QT:F635 Mean", name.Rb = "QT:B635 Median", name.Gf = "QT:F532 Mean",
+     name.Gb = "QT:B532 Median")

Java Virtual Machine is running -
parsing MAGEML files
making Layout and Gnames objects
Reading am2730miame.txt
Reading am2731miame.txt
Reading am2732m.txt
Reading am2736m.txt
Reading am2737m.txt
Reading tm1826m.txt
Reading tm1827m.txt
Reading tm1829m.txt
Reading tm1830m.txt
Reading tm1831m.txt

> print(raw)

An object of class "marrayRaw"
@maRf

 [1,]  5841  2030   2968  45 1828  1975  2077  1775  2202   841
 [2,]  2002  1312   421  96  399  557  295  748  465   83
 [3,]  2254  2057  1097 1163  649  917  755 1276  985  335
 [4,]  2212  1492   782  767  1163  649  917 1276  985  335
 [5,]    73    76    42    54    45    49    46    47    43
955 more rows ...

@maGf

 [1,]   852   750  1587  135 1625  1183  1598  1108  807  1746
 [2,]   652   529  404   162  397  515  285  390  291   90
 [3,]   576   615   634   386   734  820   696  573  457  572
 [4,]   781   589   733   366   758  848  667   597  559  716
 [5,]   157   143  111  124  130  148  135  137  146  131
955 more rows ...

@maRb

 [1,]  123  123  123  123  123  123  123  123  123  123
 [2,]  123  123  123  123  123  123  123  123  123  123
 [3,]  123  123  123  123  123  123  123  123  123  123
 [4,]  123  123  123  123  123  123  123  123  123  123
 [5,]  123  123  123  123  123  123  123  123  123  123
955 more rows ...

4
[1,]  42  42  36  39  41  47  40  45  49  42
[2,]  41  41  35  39  40  45  39  43  42  41
[3,]  41  42  34  41  40  44  40  43  42  42
[4,]  41  40  34  41  40  43  39  43  41  41
[5,]  41  39  34  40  40  42  39  43  41  41
955 more rows ...

@maGb

[1,]  150  130  87  120  104  135  117  137  168  127
[2,]  147  128  88  111  105  136  116  131  137  121
[3,]  140  124  85  106  105  133  116  128  135  121
[4,]  138  122  88  108  106  133  116  128  134  120
[5,]  138  122  87  106  104  133  114  128  133  118
955 more rows ...

@maW

<0 x 0 matrix>

@maLayout
An object of class "marrayLayout"
@maNgr
[1]  4

@maNgc
[1]  4

@maNsra
[1]  10

@maNsc
[1]  6

@maNspots
[1]  960

@maSub
[1]  TRUE

@maPlate
factor(0)
Levels:
@maControls
factor(0)
Levels:

@maNotes
[1] ""

@maGnames
An object of class "marrayInfo"
@maLabels
[1] "none" "none" "none" "none" "none"
955 more elements ...

@maInfo
[1] aj508733 V00618 aj291984 aj306233 aj310439
142 Levels: af025843 af034412 af135499 aj132353 aj291832 aj291833 ... y17187
955 more rows ...

@maNotes
[1] "Identifiers refer to database: DB:embl"

@maTargets
An object of class "marrayInfo"
@maLabels
[1] "am2730miame.txt" "am2731miame.txt" "am2732m.txt" "am2736m.txt"
[5] "am2737m.txt" "tm1826m.txt" "tm1827m.txt" "tm1829m.txt"
[9] "tm1830m.txt" "tm1831m.txt"

@maInfo
  Cy3   Cy5
  1 AM-Pool AM2730-I
  2 AM-Pool AM2731-I
  3 AM-Pool AM2732-I
  4 AM-Pool AM2736-I
  5 AM-Pool AM2737-I
  6 AM-Pool TM1826-I
  7 AM-Pool TM1827-I
  8 AM-Pool TM1829-I
  9 AM-Pool TM1830-I
If however you do not know which DesignElement Dimension, QuantitationType Dimension and Quantitation Types to use, you can call the function as follows:

```r
> datadir <- system.file("MAGEMLdata", package = "RMAGEML")
> if (interactive()) {
+   raw <- importMAGEML(directory = datadir, package = "marray")
+ }
```

This will generate a few selection panels which allow selection of the appropriate DesignElement Dimension, QuantitationType Dimension and Quantitation Types.

### 4.2 Creation of a Gnames marrayInfo object

If one just wants to make an marrayInfo object containing the gene names and database identifiers of the spotted features the function getGnames can be used.

```r
> data <- system.file("MAGEMLdata", package = "RMAGEML")
> mageom <- importMAGEOM(directory = data)

parsing MAGEML files

> getGnames(mageom, arrayID = "A-MEXP-14", DED = "DED:707", package = "marray")
```

An object of class "marrayInfo"

@maLabels
[1] "none" "none" "none" "none" "none"
955 more elements ...

@maInfo
[1] aj508733 V00618 aj291984 aj306233 aj310439
142 Levels: af025843 af034412 af135499 aj132353 aj291832 aj291833 ... y17187
955 more rows ...

@maNotes
[1] "Identifiers refer to database: DB:embl"

Again leaving out the ‘DED’ parameter will cause selection panels to pop up displaying the available DesignElement Dimensions.
4.3 Creation of an marrayLayout object

In the marray packages the information on the array layout is stored in an marrayLayout object which can be created by the getArrayLayout function.

```r
> data <- system.file("MAGEMLdata", package = "RMAGEML")
> mageom <- importMAGEOM(directory = data)

parsing MAGEML files

> getArrayLayout(mageom, arrayID = "A-MEXP-14", DED = "DED:707")

An object of class "marrayLayout"
@maNgr
[1] 4

@maNgc
[1] 4

@maNsr
[1] 10

@maNsc
[1] 6

@maNspots
[1] 960

@maSub
[1] TRUE

@maPlate
factor(0)
Levels:

@maControls
factor(0)
Levels:

@maNotes
[1] ""
```
4.4 Make an marrayRaw object

The function makeMarrayRaw takes a Gnames and Layout object and parameters corresponding to the DesignElement Dimension, QuantitationType Dimension and Quantitation Types to create an marrayRaw object.

```r
> data <- system.file("MAGEML\text{data}\", package = "RMAGEML")
> mageom <- importMAGEOM(directory = data)
```

parsing MAGEML files

```r
> gnames <- getGnames(mageom, arrayID = "A-MEXP-14", DED = "DED:707",
+    package = "marray")
> layout <- getArrayLayout(mageom, arrayID = "A-MEXP-14", DED = "DED:707")
> raw <- makeMarrayRaw(mageOM = mageom, layout = layout, gnames = gnames,
+    directory = data, arrayID = "A-MEXP-14", DED = "DED:707",
+    QTD = "QTD:707", name.Rf = "QT:F635 Mean", name.Rb = "QT:B635 Median",
+    name.Gf = "QT:F532 Mean", name.Gb = "QT:B532 Median")
```

Reading am2730miame.txt
Reading am2731miame.txt
Reading am2732m.txt
Reading am2736m.txt
Reading am2737m.txt
Reading tm1826m.txt
Reading tm1827m.txt
Reading tm1829m.txt
Reading tm1830m.txt
Reading tm1831m.txt

5 Import to limma package

5.1 One step import and creation of a limma RGLList object from MAGE-ML files

In the limma package of BioConductor the raw data is stored in an RGLList object. The function importMAGEML parses all MAGE-ML files present in the directory which is given as a parameter to the function. From these files it creates the RGLList object, containing the layout, gene names and database entries of the features spotted on the array and the foreground and background intensities for the green and red channels.

The function can be tested on the MEXP-14 dataset. This example is available from ArrayExpress at [http://www.ebi.ac.uk/arrayexpress/]

For import to limma the same function as MAGEML import to marray packages can be used, just adapt the name of the package into limma as follows:
```r
> datadir <- system.file("MAGEMLdata", package = "RMAGEML")
> raw <- importMAGEML(directory = datadir, package = "limma", arrayID = "A-MEXP-14",
+     DED = "DED:707", QTD = "QTD:707", name.Rf = "QT:F635 Mean",
+     name.Rb = "QT:B635 Median", name.Gf = "QT:F532 Mean", name.Gb = "QT:B532 Median")

parsing MAGEML files
Reading am2730miame.txt
Reading am2731miame.txt
Reading am2732m.txt
Reading am2736m.txt
Reading am2737m.txt
Reading tm1826m.txt
Reading tm1827m.txt
Reading tm1829m.txt
Reading tm1830m.txt
Reading tm1831m.txt

> print(raw)

An object of class "RGList"

$R

  am2730miame.txt    am2731miame.txt    am2732m.txt    am2736m.txt    am2737m.txt
[1,]        5841        2030         2968          45          1828
[2,]        2002        1312         421          96          399
[3,]        2254        2057        1097        1163          649
[4,]        2212        1492         782          767          709
[5,]          73          76          42          54          45

  tm1826m.txt    tm1827m.txt    tm1829m.txt    tm1830m.txt    tm1831m.txt
[1,]        1975        2077         1775        2202          841
[2,]         557         295         748          465          83
[3,]         917         755        1276          985          335
[4,]        1114         620        1004          860          488
[5,]          49          49          46          47          43
955 more rows ...

$G

  am2730miame.txt    am2731miame.txt    am2732m.txt    am2736m.txt    am2737m.txt
[1,]         852         750        1587          135         1625
[2,]         652         529         404          162          397
[3,]         576         615         634          386          734
[4,]         781         589         733          366          758
[5,]         157         143         111          124          130

  tm1826m.txt    tm1827m.txt    tm1829m.txt    tm1830m.txt    tm1831m.txt
```

$Rb$

<table>
<thead>
<tr>
<th>am2730miame.txt</th>
<th>am2731miame.txt</th>
<th>am2732m.txt</th>
<th>am2736m.txt</th>
<th>am2737m.txt</th>
</tr>
</thead>
<tbody>
<tr>
<td>42</td>
<td>42</td>
<td>36</td>
<td>39</td>
<td>41</td>
</tr>
<tr>
<td>41</td>
<td>41</td>
<td>35</td>
<td>39</td>
<td>40</td>
</tr>
<tr>
<td>41</td>
<td>42</td>
<td>34</td>
<td>41</td>
<td>40</td>
</tr>
<tr>
<td>41</td>
<td>40</td>
<td>34</td>
<td>41</td>
<td>40</td>
</tr>
<tr>
<td>41</td>
<td>39</td>
<td>34</td>
<td>40</td>
<td>40</td>
</tr>
</tbody>
</table>

$Gb$

<table>
<thead>
<tr>
<th>am2730miame.txt</th>
<th>am2731miame.txt</th>
<th>am2732m.txt</th>
<th>am2736m.txt</th>
<th>am2737m.txt</th>
</tr>
</thead>
<tbody>
<tr>
<td>150</td>
<td>130</td>
<td>87</td>
<td>120</td>
<td>104</td>
</tr>
<tr>
<td>147</td>
<td>128</td>
<td>88</td>
<td>111</td>
<td>105</td>
</tr>
<tr>
<td>140</td>
<td>124</td>
<td>85</td>
<td>106</td>
<td>105</td>
</tr>
<tr>
<td>138</td>
<td>122</td>
<td>88</td>
<td>108</td>
<td>106</td>
</tr>
<tr>
<td>138</td>
<td>122</td>
<td>87</td>
<td>106</td>
<td>104</td>
</tr>
</tbody>
</table>

$genes$

<table>
<thead>
<tr>
<th>Block</th>
<th>Row</th>
<th>Column</th>
<th>ID</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>aj508733 none</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>V00618 none</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>aj291984 none</td>
</tr>
</tbody>
</table>
Similarly if one only specifies the ‘directory’ and the ‘package’, selection panels will pop up to select the DesignElement Dimension, QuantitationType Dimension and Quantitation Types.

### 5.2 Creating the genes dataframe of an RGLList object

In limma the gene names, gene identifiers and layout information is stored in a dataframe which can be created by the getArrayLayoutLimma function.

```r
> data <- system.file("MAGEMLdata", package = "RMAGEML")
> mageom <- importMAGEOM(directory = data)
```

parsing MAGEML files

```r
genes <- getArrayLayoutLimma(mageom, arrayID = "A-MEXP-14", DED = "DED:707")
> print(genes[1:10, ])
```

<table>
<thead>
<tr>
<th>Block</th>
<th>Row</th>
<th>Column</th>
<th>ID</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>aj508733 none</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>VO0618 none</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>aj291984 none</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>aj306233 none</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td>aj310439 none</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>1</td>
<td>6</td>
<td>aj409363 none</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>1</td>
<td>7</td>
<td>aj310516 none</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>1</td>
<td>8</td>
<td>aj306230 none</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>1</td>
<td>9</td>
<td>aj310436 none</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>1</td>
<td>10</td>
<td>aj291834 none</td>
</tr>
</tbody>
</table>
```

### 5.3 Make an RGLList object

The function makeRG takes a genes dataframe (containing the layout, gene identifiers and gene names), and parameters corresponding to DesignElement Dimension, QuantitationType Dimension and Quantitation Types to create a limma RGLList object.

```r
> data <- system.file("MAGEMLdata", package = "RMAGEML")
> mageom <- importMAGEOM(directory = data)
```

parsing MAGEML files
> genes <- getArrayLayoutLimma(mageom, arrayID = "A-MEXP-14", DED = "DED:707")
> raw <- makeRG(mageOM = mageom, genes = genes, directory = data,
+     name.Rf = "QT:F635 Mean", name.Rb = "QT:B635 Median", name.Gf = "QT:F532 Mean",
+     name.Gb = "QT:B532 Median")

Reading am2730miame.txt
Reading am2731miame.txt
Reading am2732m.txt
Reading am2736m.txt
Reading am2737m.txt
Reading tm1826m.txt
Reading tm1827m.txt
Reading tm1829m.txt
Reading tm1830m.txt
Reading tm1831m.txt