BeadDataPackR: Compression Tools for Raw Illumina Beadarray Data

Mike L. Smith and Andy G. Lynch

February 9, 2011

Introduction

Raw Illumina BeadArray data consists of *.tif* images produced by the scanner, accompanied by *.txt* and *.locs* files containing details of bead locations and their intensities within the image. For a single channel human expression array these data typically occupy ≈ 125MB of storage. The size of these files can prove a hinderance to both their storage and distribution. Whilst the images can be compressed using a variety of common tools, the BeadDataPackR package aims to provide tools for the efficient compression of the *.txt* and *.locs* files.

Disclaimer: BeadDataPackR has been tested on data from a variety of Illumina BeadArray platforms and, to the best of our knowledge, data compressed using lossless settings has always been restored successfully. However, we cannot take responsibility for any loss or damage to data that results from its use.

Citing the package

If you make use of BeadDataPackR to ease distribution of your data please cite the following paper:

BeadDataPackR: A Tool to Facilitate the Sharing of Raw Data from Illumina BeadArray Studies. *Cancer Informatics*, 9:217-227, 2010.

Compressing Data

The first step before using any of the functionality is to load the package. For the purpose of this vignette we also get the path to the example data that is distributed with the package.

```
> library(BeadDataPackR)
> dataPath <- system.file("data", package = "BeadDataPackR")
```

BeadDataPackR has two primary functions, namely to compress raw Illumina data, or decompress a file already created with the package. We'll begin with file compression, which is carried out using the following commands:

```
> compressBeadData(txtFile = "example.txt", locsGrn = "example_Grn.locs",
+     outputFile = "example.bab", path = dataPath, nBytes = 4,
+     nrow = 326, ncol = 4)
```

The `txtFile` and `locsGrn` arguments specify the names of the files to be compressed. For two channel data there is an additional argument, `locsRed`, giving the name of the *.locs* file for the red channel. These files should be found within the directory specified in the `path` argument. A future revision of the package will hopefully alter this behaviour, so all arrays within a specified folder will be automatically identified and compressed.

The argument `nBytes` specifies how many bytes should be used to store the fractional parts of the bead coordinates. For a single channel array the maximum value is 4 (8 for a two channel array). If the maximum value is used the coordinates are stored in the *.bab* file as single precision floating point numbers, as they are in the *.locs* files. If a value smaller than the maximum is choosen then the integer parts of each coordinate are stored seperately. The requested number of bytes are then used to store the fractional parts, with a corresponding loss of precision as the number of bytes decreases.

The `nrow` and `ncol` arguments can normally be left blank. They specify the dimensions of each grid segment on the array and, if left blank, can normally be infered from the grid coordinates. However, this can fail for particularly small grids or cases of misregistration where segments overlap. If one wants or needs to specify them explicitly, these values can be found in the *.sdf* which accompanies the bead level output from the scanner. The number of columns and rows per segment can be found within the tags `<SizeGridX>` and `<SizeGridY>` respectively.

Decompressing Data

To decompress a *.bab* file that was created by BeadDataPackR, use the following function:

```
> decompressBeadData(inputFile = "example.bab", inputPath = dataPath,
+     outputMask = "restored", outputPath = ".", outputNonDecoded = FALSE,
+     roundValues = TRUE)
```

The `inputFile` argument specifies the name of the *.bab* that should be decompressed. This file should be located in the folder indicated by `inputPath`, which by default is the current working directory.

When an array is compressed its name is stored in the resulting *.bab* file. By default when it is decompressed this name is used for the restored files. However, this can be troublesome if you don't want to overwrite an exisiting file. The `outputMask` argument allows the user to define the names of the restored *.txt* and *.locs* files. In this case the restored files will be named 'restored.txt' and 'restored_Grn.locs'. If this was two channel data a further file, 'restored_Red.locs', would also be produced. The files are created in the location specified by the `outputPath` argument. If this is left blank the current working directory is used.

The *.txt* file that are produced by Illumina's scanner do not included the locations of beads that failed their decoding process. Since their location are retained in the *.locs* file, we have to option of including them in the restored files. `outputNonDecoded` toggles whether to include them or not. Illumina's *.txt* files also give the bead centre coordinates to 7 significant figures, resulting in a different number of decimal places as we move across the array, rather than the single precision values held in the *.locs* files. `roundValues` allows the user to choose between mimicing this behaviour when recreating the *.txt* files or ouputing the maximum available precision. The default for both these arguments is to reproduce the original Illumina files.

Extracting data directly

Rather than decompressing the *.bab* file into the two original files, it may be useful to extract data directly from it. This can be achieved in the following way:

```
> readCompressedData(inputFile = "example.bab", path = dataPath,
+     probeIDs = c(10008, 10010))
```

This function takes a *.bab* file found in the directory specified by the path argument. In this case it extracts the data relating to the beads with IDs 10008 and 10010, which is returned as a matrix in the same format as the *.txt* file that would be created if the file were to be decompressed. If the probeIDs argument is left NULL then a matrix containing data for every probe on the array is returned.

Similarly, we can extract the information contained in the original *.locs* file, i.e. the bead centre coordinates in *.locs* file order, by using the following command:

```
> locs <- extractLocsFile(inputFile = "example.bab", path = dataPath)
```

Using coordinates in the grid order provided by the *.locs* can be useful for quickly determining neighbouring beads and is used in several instances by the beadarray package.

Session Info

```
> toLatex(sessionInfo())
```

- R version 2.12.1 (2010-12-16), x86_64-unknown-linux-gnu
- Locale: LC_CTYPE=en_US.UTF-8, LC_NUMERIC=C, LC_TIME=en_US.UTF-8, LC_COLLATE=C, LC_MONETARY=C, LC_MESSAGES=en_US.UTF-8, LC_PAPER=en_US.UTF-8, LC_NAME=C, LC_ADDRESS=C, LC_TELEPHONE=C, LC_MEASUREMENT=en_US.UTF-8, LC_IDENTIFICATION=C
- Base packages: base, datasets, grDevices, graphics, methods, stats, utils
- Other packages: BeadDataPackR 1.2.1
- Loaded via a namespace (and not attached): tools 2.12.1

Table 1: The output of sessionInfo on the build system after running this vignette.