Indentifying the Gene List Probesets

Keith A. Baggerly, Shannon Neeley, and Kevin R. Coombes

October 9, 2007

1 Introduction

In table 2 of their paper, Dressman et al provide a list of the "Highest weighted genes in the platinum prediction reponse models using an 83-sample training set and validated in a 36-sample validation set." This table gives Gene Title for each entry, and the Gene Symbol and Representative Public ID are also given for most entries (there are a few gaps). In order to map these over to the data that we have, we need to get the Affymetrix probeset IDs as well. Here, we identify the probesets.

2 Options and Libraries

```
> options(width = 80)
```

3 Loading the List

We first extracted the list from the paper and saved it as a csv file so that we could import it into R (we also removed commas from some of the gene titles in order to have things parse properly).

```
> reportedGenes <- read.table(file.path("OtherData", "gene_list.csv"),
+     sep = ",", quote = "", header = TRUE, colClasses = "character",
+     strip.white = TRUE)
> dim(reportedGenes)

[1] 100   3

> reportedGenes[1:3, ]

                                            Gene.Title Gene.Symbol
1                       Sialidase 1 (lysosomal sialidase)        NEU1
2 Translocated promoter region (to activated MET oncogene)         TPR
3                                            Periplakin         PPL
  Representative.Public.ID
1                   U84246
2                NM_003292
3                NM_002705
```

4 Loading Annotation From GEO

Gene title, gene symbol, and representative public ID are all fields in the annotation available from GEO for various arrays. The U133A is platform GPL96, and we downloaded a text file with the annotation from `http://www.ncbi.nlm.nih.gov/projects/geo/query/acc.cgi?acc=GPL96`. We then pulled all of this into R.

```
> h133aChipInfo <- read.table("OtherData/GEO/GPL96-3386.txt", header = TRUE,
+     skip = 16, sep = "\t", quote = "", row.names = 1)
> dim(h133aChipInfo)
```

```
[1] 22283    15
```

```
> colnames(h133aChipInfo)
```

```
 [1] "GB_ACC"                          "SPOT_ID"
 [3] "Species.Scientific.Name"         "Annotation.Date"
 [5] "Sequence.Type"                   "Sequence.Source"
 [7] "Target.Description"              "Representative.Public.ID"
 [9] "Gene.Title"                      "Gene.Symbol"
[11] "Entrez.Gene"                     "RefSeq.Transcript.ID"
[13] "Gene.Ontology.Biological.Process" "Gene.Ontology.Cellular.Component"
[15] "Gene.Ontology.Molecular.Function"
```

```
> h133aChipInfo[, "Gene.Title"] <- as.character(h133aChipInfo[,
+     "Gene.Title"])
> h133aChipInfo[, "Gene.Symbol"] <- as.character(h133aChipInfo[,
+     "Gene.Symbol"])
> h133aChipInfo[, "Representative.Public.ID"] <- as.character(h133aChipInfo[,
+     "Representative.Public.ID"])
```

5 Matching Genes to Probesets

We do this in two stages. First, we look for the genes where the matching is easy, and then we perform more detailed examinations to better identify the remainder. Below, we use the shorthand "Rep" for "Representative.Public.ID".

```
> probesetMatches <- vector("list", 100)
> for (i1 in 1:100) {
+     probesetMatches[[i1]] <- rownames(h133aChipInfo)[h133aChipInfo$Rep ==
+         reportedGenes$Rep[i1]]
+ }
> nMatches <- unlist(lapply(probesetMatches, length))
> table(nMatches)
```

```
nMatches
 0  1  2
11 87  2
```

Here, we see that a large portion of the mapping job is easy. For 87 of the entries, the representative public id (RPID) specifies a unique probeset. For two others, we can only narrow the target down to one of two options. Let's look at those two a bit more closely.

```
> which(nMatches == 2)

[1]  6 62

> probesetMatches[[which(nMatches == 2)[1]]]

[1] "210459_at"   "210460_s_at"

> probesetMatches[[which(nMatches == 2)[2]]]

[1] "200737_at"   "200738_s_at"

> reportedGenes[c(6, 62), ]

                                          Gene.Title Gene.Symbol
6  Proteasome (prosome macropain) 26S subunit non-ATPase 4       PSMD4
62                            Phosphoglycerate kinase 1        PGK1
   Representative.Public.ID
6                  AB033605
62                NM_000291

> h133aChipInfo[h133aChipInfo$Rep == "AB033605", c("Gene.Title",
+     "Gene.Symbol")]

                                                            Gene.Title
210459_at   proteasome (prosome, macropain) 26S subunit, non-ATPase, 4
210460_s_at proteasome (prosome, macropain) 26S subunit, non-ATPase, 4
            Gene.Symbol
210459_at         PSMD4
210460_s_at       PSMD4

> h133aChipInfo[h133aChipInfo$Rep == "NM_000291", c("Gene.Title",
+     "Gene.Symbol")]

                         Gene.Title Gene.Symbol
200737_at   phosphoglycerate kinase 1        PGK1
200738_s_at phosphoglycerate kinase 1        PGK1
```

Given the RPID, the gene title and symbol fail to clarify the story further, so we're stuck with some ambiguity here.

Our question now is whether we can recover anything for the 11 genes with no reported gene symbols or representative public ids. Unfortunately, this may involve some individualized assessment. Let's get started.

```
> badRows <- which(nMatches == 0)
> badRows

 [1] 40 43 67 69 72 77 78 85 92 98 99
```

5.1 Matching 1/11

Checking the first illustrates the approach and some of the issues.

```
> reportedGenes[badRows[1], ]
```

```
                                                                                                    Gene.Tit
40 Protein kinase interferon-inducible double stranded RNAdependent inhibitor repressor of (P58 represso
   Gene.Symbol Representative.Public.ID
40
```

```
> goodRows <- grep("P58 repressor", h133aChipInfo$Gene.Title)
> as.character(h133aChipInfo[goodRows, "Gene.Title"])
```

```
[1] "protein-kinase, interferon-inducible double stranded RNA dependent inhibitor, repressor of (P58 rep
```

```
> as.character(h133aChipInfo[goodRows, "Gene.Symbol"])
```

```
[1] "PRKRIR"
```

```
> as.character(h133aChipInfo[goodRows, "Representative.Public.ID"])
```

```
[1] "AF081567"
```

```
> rownames(h133aChipInfo[goodRows, ])
```

```
[1] "209323_at"
```

```
> probesetMatches[[badRows[1]]] <- rownames(h133aChipInfo[goodRows,
+     ])
```

We look at the title to subjectively identify a potentially unique substring, and look for that substring in the GEO annotation. Here we were successful, and got a unique hit. Looking at the full gene title from GEO illustrates that while the names are quite similar, there are some distinctions that would have caused looking for the full string to fail. First, there are commas in the name from GEO, which we stripped out in assembling our csv file. Second, the first word in the name from GEO is not capitalized, but it is in the table from Dressman et al. This is likely an artifact automatically introduced by Excel. We note that for this hit, we also have a gene symbol and a representative public id; it is not clear why these did not make it to the initial table in Dressman et al. Finally, we record the probeset identified.

5.2 Matching 2/11

On to the second one.

```
> reportedGenes[badRows[2], ]
```

```
                                                                    Gene.Title
43 Low-density lipoproteinâĂŞrelated protein 1 (alpha-2-macroglobulin receptor)
   Gene.Symbol Representative.Public.ID
43
```

```
> goodRows <- grep("alpha-2-macroglobulin receptor", h133aChipInfo$Gene.Title)
> as.character(h133aChipInfo[goodRows, "Gene.Title"])
```

```
[1] "low density lipoprotein-related protein 1 (alpha-2-macroglobulin receptor)"
[2] "low density lipoprotein-related protein 1 (alpha-2-macroglobulin receptor)"
```

```
> as.character(h133aChipInfo[goodRows, "Gene.Symbol"])
```

```
[1] "LRP1" "LRP1"
```

```
> as.character(h133aChipInfo[goodRows, "Representative.Public.ID"])
```

```
[1] "BF304759"  "NM_002332"
```

```
> rownames(h133aChipInfo[goodRows, ])
```

```
[1] "200784_s_at" "200785_s_at"
```

```
> probesetMatches[[badRows[2]]] <- rownames(h133aChipInfo[goodRows,
+     ])
```

Again, the title lets us narrow things down pretty far; this time to 2 probesets. Again, there are values for the gene symbol and the representative public id. As the latter are different, specifying the public id would uniquely identify the probeset, but we don't have this information.

5.3 Matching 3/11

On to the third.

```
> reportedGenes[badRows[3], ]
```

```
                                                                                Gene.Title
67 DiGeorge syndrome critical region gene 6///DiGeorge syndrome critical region gene 6-like
   Gene.Symbol Representative.Public.ID
67
```

```
> goodRows <- grep("DiGeorge syndrome critical region gene 6",
+     h133aChipInfo$Gene.Title)
> as.character(h133aChipInfo[goodRows, "Gene.Title"])
```

```
[1] "DiGeorge syndrome critical region gene 6 /// DiGeorge syndrome critical region gene 6-like"
[2] "DiGeorge syndrome critical region gene 6-like"
```

```
> goodRows <- grep("DiGeorge syndrome critical region gene 6 ///",
+     h133aChipInfo$Gene.Title)
> as.character(h133aChipInfo[goodRows, "Gene.Symbol"])
```

```
[1] "DGCR6 /// DGCR6L"
```

```
> as.character(h133aChipInfo[goodRows, "Representative.Public.ID"])
```

```
[1] "NM_005675"
```

```
> rownames(h133aChipInfo[goodRows, ])
```

```
[1] "208024_s_at"
```

```
> probesetMatches[[badRows[3]]] <- rownames(h133aChipInfo[goodRows,
+     ])
```

Here, the name is sufficient to uniquely identify the probeset id. Again, a gene symbol and public id are available.

5.4 Matching 4/11

On to the fourth.

```
> reportedGenes[badRows[4], ]
```

```
                                                          Gene.Title
69 Tankyrase TRF1-interacting ankyrin-related ADP-ribose polymerase 2
   Gene.Symbol Representative.Public.ID
69
```

```
> goodRows <- grep("TRF1-interacting", h133aChipInfo$Gene.Title)
> as.character(h133aChipInfo[goodRows, "Gene.Title"])
```

```
[1] "tankyrase, TRF1-interacting ankyrin-related ADP-ribose polymerase"
[2] "tankyrase, TRF1-interacting ankyrin-related ADP-ribose polymerase"
[3] "tankyrase, TRF1-interacting ankyrin-related ADP-ribose polymerase"
[4] "tankyrase, TRF1-interacting ankyrin-related ADP-ribose polymerase 2"
```

```
> goodRows <- grep("TRF1-interacting ankyrin-related ADP-ribose polymerase 2",
+     h133aChipInfo$Gene.Title)
> as.character(h133aChipInfo[goodRows, "Gene.Symbol"])
```

```
[1] "TNKS2"
```

```
> as.character(h133aChipInfo[goodRows, "Representative.Public.ID"])
```

```
[1] "NM_025235"
```

```
> rownames(h133aChipInfo[goodRows, ])
```

```
[1] "218228_s_at"
```

```
> probesetMatches[[badRows[4]]] <- rownames(h133aChipInfo[goodRows,
+     ])
```

Here, the name is sufficient to uniquely identify the probeset id. Again, a gene symbol and public id are available.

5.5 Matching 5/11

On to the fifth.

```
> reportedGenes[badRows[5], ]
```

```
                                                                Gene.Title
72 Mitochondrial ribosomal protein L9///mitochondrial ribosomal protein L9
   Gene.Symbol Representative.Public.ID
72
```

```
> goodRows <- grep("mitochondrial ribosomal protein L9", h133aChipInfo$Gene.Title)
> as.character(h133aChipInfo[goodRows, "Gene.Title"])
```

```
[1] "mitochondrial ribosomal protein L9" "mitochondrial ribosomal protein L9"
```

```
> as.character(h133aChipInfo[goodRows, "Gene.Symbol"])
```

```
[1] "MRPL9" "MRPL9"
```

```
> as.character(h133aChipInfo[goodRows, "Representative.Public.ID"])
```

```
[1] "BC004517" "AB049636"
```

```
> rownames(h133aChipInfo[goodRows, ])
```

```
[1] "209609_s_at" "211594_s_at"
```

```
> probesetMatches[[badRows[5]]] <- rownames(h133aChipInfo[goodRows,
+     ])
```

The title lets us narrow things down to 2 probesets. Again, there are values for the gene symbol and the representative public ids; as the latter are different, specifying the public id would have uniquely identified the probeset.

5.6 Matching 6/11

On to the sixth.

```
> reportedGenes[badRows[6], ]
```

```
                                                                    Gene.Title
77 Prosaposin (variant Gaucher disease and variant metachromatic leukodystrophy)
   Gene.Symbol Representative.Public.ID
77
```

```
> goodRows <- grep("variant Gaucher disease", h133aChipInfo$Gene.Title)
> as.character(h133aChipInfo[goodRows, "Gene.Title"])
```

```
[1] "prosaposin (variant Gaucher disease and variant metachromatic leukodystrophy)"
[2] "prosaposin (variant Gaucher disease and variant metachromatic leukodystrophy)"
```

```
> as.character(h133aChipInfo[goodRows, "Gene.Symbol"])
```

```
[1] "PSAP" "PSAP"
```

```
> as.character(h133aChipInfo[goodRows, "Representative.Public.ID"])
```

```
[1] "M32221"    "NM_002778"
```

```
> rownames(h133aChipInfo[goodRows, ])
```

```
[1] "200866_s_at" "200871_s_at"
```

```
> probesetMatches[[badRows[6]]] <- rownames(h133aChipInfo[goodRows,
+     ])
```

The title lets us narrow things down to 2 probesets. Again, there are values for the gene symbol and the representative public ids; as the latter are different, specifying the public id would have uniquely identified the probeset.

5.7 Matching 7/11

On to the seventh.

```
> reportedGenes[badRows[7], ]
```

```
                                                                Gene.Title
78 Solute carrier family 25 (mitochondrial carrier; oxoglutarate carrier)
   Gene.Symbol Representative.Public.ID
78
```

```
> goodRows <- grep("oxoglutarate carrier", h133aChipInfo$Gene.Title)
> as.character(h133aChipInfo[goodRows, "Gene.Title"])
```

```
[1] "solute carrier family 25 (mitochondrial carrier; oxoglutarate carrier), member 11"
[2] "solute carrier family 25 (mitochondrial carrier; oxoglutarate carrier), member 11"
```

```
> as.character(h133aChipInfo[goodRows, "Gene.Symbol"])
```

```
[1] "SLC25A11" "SLC25A11"
```

```
> as.character(h133aChipInfo[goodRows, "Representative.Public.ID"])
```

```
[1] "NM_003562" "AF070548"
```

```
> rownames(h133aChipInfo[goodRows, ])
```

```
[1] "207088_s_at" "209003_at"
```

```
> probesetMatches[[badRows[7]]] <- rownames(h133aChipInfo[goodRows,
+     ])
```

The title lets us narrow things down to 2 probesets. Again, there are values for the gene symbol and the representative public ids; as the latter are different, specifying the public id would have uniquely identified the probeset.

5.8 Matching 8/11

On to the eighth.

```
> reportedGenes[badRows[8], ]
```

```
                                                    Gene.Title Gene.Symbol
85 ATP synthase H  transporting mitochondrial F0 complex subunit d
   Representative.Public.ID
85
```

```
> goodRows <- grep("mitochondrial F0 complex, subunit d", h133aChipInfo$Gene.Title)
> as.character(h133aChipInfo[goodRows, "Gene.Title"])
```

```
[1] "ATP synthase, H+ transporting, mitochondrial F0 complex, subunit d"
```

```
> as.character(h133aChipInfo[goodRows, "Gene.Symbol"])
```

```
[1] "ATP5H"
```

```
> as.character(h133aChipInfo[goodRows, "Representative.Public.ID"])
```

```
[1] "AF061735"
```

```
> rownames(h133aChipInfo[goodRows, ])
```

```
[1] "210149_s_at"
```

```
> probesetMatches[[badRows[8]]] <- rownames(h133aChipInfo[goodRows,
+     ])
```

Here, the name is sufficient to uniquely identify the probeset id. Again, a gene symbol and public id are available. We note that the positioning of a comma in the midst of the string to be matched was found by looking at the table in Dressman et al.

5.9 Matching 9/11

On to the ninth.

```
> reportedGenes[badRows[9], ]
```

```
                                                    Gene.Title Gene.Symbol
92 Nucleolar protein family A member 3 (H/ACA small nucleolar RNPs)
   Representative.Public.ID
92
```

```
> goodRows <- grep("H/ACA small nucleolar RNPs", h133aChipInfo$Gene.Title)
> as.character(h133aChipInfo[goodRows, "Gene.Title"])
```

```
[1] "nucleolar protein family A, member 2 (H/ACA small nucleolar RNPs)"
[2] "nucleolar protein family A, member 3 (H/ACA small nucleolar RNPs)"
[3] "nucleolar protein family A, member 1 (H/ACA small nucleolar RNPs)"
```

```
> goodRows <- grep("nucleolar protein family A, member 3", h133aChipInfo$Gene.Title)
> as.character(h133aChipInfo[goodRows, "Gene.Symbol"])
```

```
[1] "NOLA3"
```

```
> as.character(h133aChipInfo[goodRows, "Representative.Public.ID"])
```

```
[1] "NM_018648"
```

```
> rownames(h133aChipInfo[goodRows, ])
```

```
[1] "217962_at"
```

```
> probesetMatches[[badRows[9]]] <- rownames(h133aChipInfo[goodRows,
+     ])
```

Here, the name is sufficient to uniquely identify the probeset id. Again, a gene symbol and public id are available.

5.10 Matching 10/11

On to the tenth.

```
> reportedGenes[badRows[10], ]
```

```
                                                          Gene.Title
98 Shank-interacting protein-like 1///shank-interacting protein-like 1
   Gene.Symbol Representative.Public.ID
98
```

```
> goodRows <- grep("SHANK", h133aChipInfo$Gene.Title)
> as.character(h133aChipInfo[goodRows, "Gene.Title"])
```

```
[1] "SHANK-associated RH domain interactor"
```

```
> as.character(h133aChipInfo[goodRows, "Gene.Symbol"])
```

```
[1] "SHARPIN"
```

```
> as.character(h133aChipInfo[goodRows, "Representative.Public.ID"])
```

```
[1] "NM_030974"
```

```
> rownames(h133aChipInfo[goodRows, ])
```

```
[1] "220973_s_at"
```

```
> probesetMatches[[badRows[10]]] <- rownames(h133aChipInfo[goodRows,
+     ])
```

This one was more difficult; none of the variants of "shank-interacting" that we initially thought of gave hits. Google found a webpage, `http://www.genomika.pl/thyroidcancer/3mode.txt`, linking the probeset 220973_s_at, with symbol SHARPIN and public ID NM_030974. Trying this in GEO suggested using SHANK in all caps, as done above. Checking RefSeq, the public ID has tweaked slightly to NM_030974.3. Trying this on Gene Cards gives a hit, listing both the title found in GEO and the title given in their gene list as variants. That Gene Cards entry lists just one U133A probeset, which is the one found and recorded above.

5.11 Matching 11/11

On to the eleventh (and last).

```
> reportedGenes[badRows[11], ]
```

```
                                                                        Gene.Title
99 Natriuretic peptide receptor A/guanylate cyclase A (atrionatriuretic peptide receptor A)
   Gene.Symbol Representative.Public.ID
99
```

```
> goodRows <- grep("atrionatriuretic peptide receptor A", h133aChipInfo$Gene.Title)
> as.character(h133aChipInfo[goodRows, "Gene.Title"])
```

```
[1] "natriuretic peptide receptor A/guanylate cyclase A (atrionatriuretic peptide receptor A)"
[2] "natriuretic peptide receptor A/guanylate cyclase A (atrionatriuretic peptide receptor A)"
```

```
> as.character(h133aChipInfo[goodRows, "Gene.Symbol"])
```

```
[1] "NPR1" "NPR1"
```

```
> as.character(h133aChipInfo[goodRows, "Representative.Public.ID"])
```

```
[1] "NM_000906" "X15357"
```

```
> rownames(h133aChipInfo[goodRows, ])
```

```
[1] "204648_at" "32625_at"
```

```
> probesetMatches[[badRows[11]]] <- rownames(h133aChipInfo[goodRows,
+     ])
```

The title lets us narrow things down to 2 probesets. Again, there are values for the gene symbol and the representative public ids; as the latter are different, specifying the public id would have uniquely identified the probeset.

6 Checking the List

Now that we've "filled in the blanks" to the extent possible, let's recheck the final tally.

```
> nMatches <- unlist(lapply(probesetMatches, length))
> table(nMatches)
```

```
nMatches
 1  2
93  7
```

We have 93 unique matches, and 7 "doublets", so we're probably looking at 107 distinct probeset IDs. Given that we're willing to work with this list, let's reassemble it using the GEO annotations, and save it so that we don't have to track it down again.

```
> reportedProbesets <- unlist(probesetMatches)
> temp <- match(c("Gene.Symbol", "Representative.Public.ID", "Gene.Title"),
+     colnames(h133aChipInfo))
> reportedGenesGEO <- h133aChipInfo[reportedProbesets, temp]
```

Let's do some quick quality checking.

```
> length(unique(reportedGenesGEO[, "Gene.Symbol"]))
```

```
[1] 99
```

```
> length(unique(reportedGenesGEO[, "Representative.Public.ID"]))
```

```
[1] 105
```

```
> length(unique(reportedGenesGEO[, "Gene.Title"]))

[1] 99
```

Some of this is surprising. As expected, the number of representative public ids is more than 100, as some of the genes that we have identified by title alone have multiple entries. We weren't expecting there to be less than 100 unique gene symbols or titles, however. Let's take a look at the ones that occur more than once.

```
> table(reportedGenesGEO[, "Gene.Symbol"])[table(reportedGenesGEO[,
+     "Gene.Symbol"]) > 1]

    LRP1    MRPL9     NPR1     PGK1     PSAP    PSMD4 SLC25A11      TPR
       2        2        2        2        2        2        2        2

> which(reportedGenesGEO$Gene.Symbol == "LRP1")

[1] 44 45

> which(reportedGenesGEO$Gene.Symbol == "MRPL9")

[1] 75 76

> which(reportedGenesGEO$Gene.Symbol == "NPR1")

[1] 105 106

> which(reportedGenesGEO$Gene.Symbol == "PGK1")

[1] 64 65

> which(reportedGenesGEO$Gene.Symbol == "PSAP")

[1] 81 82

> which(reportedGenesGEO$Gene.Symbol == "PSMD4")

[1] 6 7

> which(reportedGenesGEO$Gene.Symbol == "SLC25A11")

[1] 83 84

> which(reportedGenesGEO$Gene.Symbol == "TPR")

[1]  2 28
```

For 7 of the 8 doublets, the occurrences are positioned sequentially in the overall list, corresponding to the 7 genes where the information was not sufficient to let us identify a unique id. The last, TPR, is found in 2 nonsequential rows.

```
> reportedGenesGEO[which(reportedGenesGEO$Gene.Symbol == "TPR"),
+     ]
```

```
           Gene.Symbol Representative.Public.ID
201731_s_at         TPR                NM_003292
201730_s_at         TPR                 BF110993
                                                                  Gene.Title
201731_s_at translocated promoter region (to activated MET oncogene)
201730_s_at translocated promoter region (to activated MET oncogene)
```

What we see here are 2 sequential probesets, with the same symbol and title, but with different public ids. Both of these entries are present in the Dressman et al table (rows 2 and 27 in the paper), providing some redundancy for the the assertion that this gene is relevant.

7 Summary

1. 87 of the 100 genes reported by Dressman (using gene symbols and GenBank accession numbers) can be uniquely (and automatrically) matched to a probe set on the U133 array. Two more genes have ambiguous matches to two possible probe sets.
2. The remaining 11 genes could be matched by hand to one or two probe sets on the U133 array.

8 Appendix

8.1 Saves

```
> save(reportedGenesGEO, file = paste("RDataObjects", "reportedGenesGEO.Rda",
+     sep = .Platform$file.sep))
> save(h133aChipInfo, file = paste("RDataObjects", "h133aChipInfo.Rda",
+     sep = .Platform$file.sep))
```

8.2 SessionInfo

```
> sessionInfo()
```

```
R version 2.5.1 (2007-06-27)
i386-pc-mingw32

locale:
LC_COLLATE=English_United States.1252;LC_CTYPE=English_United States.1252;LC_MONETARY=English_United Sta

attached base packages:
[1] "splines"   "tools"     "stats"     "graphics"  "grDevices" "utils"
[7] "datasets"  "methods"   "base"

other attached packages:
       survival  ClassDiscovery         cluster ClassComparison      PreProcess
         "2.32"         "2.5.0"        "1.11.7"         "2.5.0"         "2.5.0"
      oompaBase     geneplotter         lattice        annotate            affy
        "2.5.0"        "1.14.0"       "0.15-11"        "1.14.1"        "1.14.2"
         affyio         Biobase
        "1.4.1"        "1.14.1"
```