Database checking: does an array have key gene symbols for my disease/question at hand?

I pulled relevant genes that were found significant in either breast cancer cell lines or human biopsies from the Broad Insitute and exported these as a text file: http://www.broadinstitute.org/gsea/msigdb/search.jsp

I transposed this table, and then read it into excel (the tranpose is so that each column, or field, in MySQL, will be one gene list)

```
#dummy example - already did this. assuming no extraneous
columns, rows to remove.
geneSignatures <- read.delim(header=TRUE, file = "~/Box
Documents/Atul BC
biomarkers/gene_signatures/existing_BC_gene_signatures.txt")
geneSignatures <- unlist(geneSignatures)
```

I then unlisted all these signatures to get one long list, and corrected for any excel-induced errors and then updated the symbols to the most current symbol using the HGNChelper R package. I noted in the warnings that some gene symbols were denoted in all lowercase, meaning they may have been taken from mouse cell lines or tissue,and HGNC corrected this to the correct human gene symbol. I then take a list of only the unique, non-duplicated symbols.

Note that if there were multiple gene symbols on one line, HGNChelper only takes one - for this rough pass, I'm OK with that (I'm not attaching these gene symbols to expression values or anything.)

```
library('HGNChelper')
geneSignatures <- findExcelGeneSymbols(geneSignatures)
geneSignatures <- checkGeneSymbols(geneSignatures)
geneSignatures <- unique(geneSignatures)
```

I then uploaded this into a MySQL table so that I have it as a reference - I named the table “BC_allGenesEverPublished”. I could further clean up the list and take only those gene signatures from human biopsies, etc. The bigger part is checking whether most of these gene symbols pop up in the microarray studies I use from GEO. I wrote the function below to handle this:

```
# pass a list of genes to a list of gene symbol lists, and
see which ones
# match up - report % found in each gene symbol list helps
rat out very
# poor microarrays that don't have enough gene symbols for
```

```
us. assumption:
# all gene symbols are updated. keyword for HGNChelper?
library("DBI")
library("RMySQL")

# can feed in a list of gene symbols, or a name for a SQL
database you'll
# be pulling from.
testGSList <- function(fieldName = "gene_symbol",
baselineGS_DBName, compareGS_DBName,
    username = "ywrfc09", password = "aveelyau05", host =
"buttelab-db1", dbname = "user_ywrfc09") {

    if (length(baselineGS_DBName) == 1) {

        m <- dbDriver("MySQL")
        con <- dbConnect(m, username = "ywrfc09", password =
"aveelyau05", host = "buttelab-db1",
            dbname = "user_ywrfc09")
        queryBaseline <- paste("SELECT ", fieldName, " FROM
", baselineGS_DBName,
            sep = "")
        res <- dbSendQuery(con, queryBaseline)
        baseline <- fetch(res, -1)
        dbDisconnect(con)

        baseline <- as.vector(t(baseline))


        # then just gave a vector of gene values
    } else {

        baseline <- baselinseGS_DBName

    }

    if (length(compareGS_DBName) == 1) {

        m <- dbDriver("MySQL")
        con <- dbConnect(m, username = "ywrfc09", password =
"aveelyau05", host = "buttelab-db1",
            dbname = "user_ywrfc09")
        queryCompare <- paste("SELECT ", fieldName, " FROM ",
compareGS_DBName,
            sep = "")
        res <- dbSendQuery(con, queryCompare)
        compare <- fetch(res, -1)
        dbDisconnect(con)

        compare <- as.vector(t(compare))

        # then just gave a vector of gene values
    } else {

        compare <- compareGS_DBName
```

```
    }



    # compare may have doubles will include 1 NA value...
    baseline <- unique(baseline)
    compare <- unique(compare)

    # want the BASELINE indices
    matchIndices <- match(compare, baseline)

    # remove NA values

    matchIndices <- matchIndices[which(!is.na(matchIndices))]

    matched <- baseline[matchIndices]
    missed <- baseline[-matchIndices]

    # COME BACK: how handles NAs???? It's being weird when I
try != NA or
    # !is.na()
    matchRate <- length(matched)/length(baseline)

    answer <- list(matched = matched, missed = missed,
matchRate = matchRate)
    return(answer)

}
```

If I run it (here for study GSE12093, which I have a separted linked probe/gene symbol table I can call), I get a list of matched genes, missed genes that were in my reference gene signature list but not in the specific study, and the match rate. The lowest match rate I got was 60%. Most were above 90%. Oddly enough, the Affymetric GPL96 usually had only a 77% match. I assume GPL96 has related gene symbols, but when collapsing datasets and only taking specific gene symbols that are present across all datasets, this still isn't ideal. We can see in the example below that from a reference dataset of about 13k gene symbols, 10k/13k are in GPL96 for study GSE12093.

Note that the DBName keywords can be character string linking to a MySQL table, or just a vector of gene symbols for comparison.

```
answer <- testGSList(baselineGS_DBName =
"BC_allGenesEverPublished", compareGS_DBName =
"probes_GSE12093_GPL96")

answer$matchRate
```

```
## [1] 0.7401
```

```
length(answer$matched)
```

```
## [1] 10339
```

```
print(answer$matched[1:10])
```

```
##  [1] "DDR1"   "RFC2"   "HSPA6"  "PAX8"   "UBA7"   "THRA"
"PTPN21"
##  [8] "CCL5"   "CYP2E1" "EPHB3"
```

```
length(answer$missed)
```

```
## [1] 3631
```