Matching Predictors to Microarrays: They Use Novartis A

Kevin R. Coombes, Jing Wang, and Keith A. Baggerly

3 March 2007

1 Problem Definition

Given the Affymetrix expression values in the table "Chemo predictors (U-95).xls", we want to identify which arrays (and hence which samples) are involved.

2 Preparation

We start by loading in the Novartis data and the NSC data, which was prepared previously.

```
> prep <- file.path("Tangled", "prepareData.R")
> Stangle(file.path("RnowebSource", "prepareData.Rnw"),
+     output = prep)

Writing to file Tangled/prepareData.R

> source(prep)
```

3 Getting the Predictors

The columns of "Chemo predictors (U-95).xls" are labeled to indicate sensitivity and resistance to different drugs, but they do not identify the actual cell lines.

This file contains $12,558$ data rows, each identified by an Affymetrix probe set ID. Note that this number is slightly smaller than the number $(12,625)$ of probe sets on an Affymetrix U95A GeneChip. The exact filtering step was unspecified, but probably consisted of leaving out the control probe sets.

The file also contained 121 columns of data. Since there are only 60 cell lines in the NCI-60 (or so we suspect...), we figured that some of the columns were probably repeated (because some cell lines were sensitive or resistant to more than one treatment). To figure this out, we first edited the Excel file to remove a text comment at the right, stating that "All predictor (sic) predict sensitivity (1), nit (sic) resistance (0); except for docetaxel where 0 = sensive (sic) and 1 = resistance", and then saved the table as a csv file. Now we read that file into R for parsing.

```
> dukeDir <- file.path("..", "PublicData", "DukeWebSite")
> predictors <- read.table(file.path(dukeDir, "Chemo predictors (U-95).csv"),
+     sep = ",", header = TRUE)
> rownames(predictors) <- as.character(predictors[, "probe_set"])
> predictors <- predictors[, 2:ncol(predictors)]
> dim(predictors)

[1] 12558   121
```

We now define some information for each column, indicating which drug/level combination it belongs to.

Note that, in spite of the comment, our analysis (see the later file, "GI50ValuesOverlap.Rnw") indicates that, for docetaxel, 0 still indicates resistant and 1 still indicates sensitive.

```
> drugList <- c("Adria", "Etopo", "X5.FU", "Cytox", "Topo",
+     "Taxol", "Doce")
> drugName <- rep("Unknown", ncol(predictors))
> drugLevel <- drugName
> for (i1 in 1:length(drugList)) {
+     temp <- grep(paste("^", drugList[i1], sep = ""),
+         colnames(predictors))
+     drugName[temp[1]:temp[2]] <- drugList[i1]
+     if (any(grep(paste("^", drugList[i1], "0", sep = ""),
+         colnames(predictors)[temp[1]]))) {
+         drugLevel[temp[1]] <- "Resistant"
+         drugLevel[temp[2]] <- "Sensitive"
+     }
+     else {
+         drugLevel[temp[1]] <- "Sensitive"
+         drugLevel[temp[2]] <- "Resistant"
+     }
+ }
> drugName[drugName == "X5.FU"] <- "5-FU"
> drugLevel[grep("^X0", colnames(predictors))] <- "Resistant"
> drugLevel[grep("^X1", colnames(predictors))] <- "Sensitive"
```

We also want to assign a score indicating which columns are identical. To perform this step, we note that the value for the first probe set (36460_at) is sufficient to uniquely determine the entire vector:

```
> rownames(predictors)[1]

[1] "36460_at"

> rownames(novartis)[1]
```

```
[1] "36460_at"
```

```
> dim(novartis)
```

```
[1] 12625   180
```

```
> length(unique(novartis[1, ]))
```

```
[1] 180
```

So, first use this value to identify matches.

```
> temp <- unique(t(predictors["36460_at", ]))
> nLinesUsed <- length(temp)
> lineIndicator <- match(t(predictors["36460_at", ]), temp)
```

Then, check to make sure that equality extends to the rest of the probe sets.

```
> passFail <- rep("Fail", nLinesUsed)
> for (i1 in 1:nLinesUsed) {
+     temp <- which(lineIndicator == i1)
+     if (all(rep(predictors[, temp[1]], length(temp)) ==
+         predictors[, temp])) {
+         passFail[i1] <- "Pass"
+     }
+ }
> if (all(passFail == "Pass")) {
+     "all passed"
+ } else {
+     "some failed"
+ }
```

```
[1] "all passed"
```

Now collect the info and tidy up.

```
> predictorsInfo <- data.frame(index = c(1:(dim(predictors)[2])),
+     drugName = as.factor(drugName), responseStatus = as.factor(drugLevel),
+     lineIndicator = as.factor(lineIndicator))
> rm(i1, temp, passFail, drugList, drugName, drugLevel)
> rm(nLinesUsed, lineIndicator)
```

4 Identifying the Data Source, take 1

Because we wanted to able to identify the cell lines, we downloaded the Affymetrix data for the NCI60 cell lines from the NCI web site at `http://dtp.nci.nih.gov/mtargets/download.html`. There were three files:

- "WEB_HOOKS_NOV_GC_ALL.TXT" contained data from Novartis, on individual arrays where each cell line was processed in triplicate.
- "WEB_HOOKS_NOVARTIS.TXT" contained averages of the triplicate data from Novartis
- "WEB_HOOKS_GENELOGIC_U95.TXT" contained U95A data on the NCI60 cell lines from GeneLogic.

Note that only the individual Novartis data included the Affymetrix probe set IDs; the averaged Novartis data and the GeneLogic data only contained GenBank accession numbers and other derived quantities.

We decided to focus first on the individual Novartis data, since it would be easiest to match the genes on Affymetrix probe sets from this data set. Because the data looked more like a database dump than the usual "genes-by-samples" matrix, we started by cutting-and-pasting the values for the probe set 36460_at from the Novartis individual data and put it into a small file called "Novartis36460_at.csv". It is interesting to note that this probe set is the first one in the Novartis data set as well as being the first row in the Chemo predictors table.

Now we read in the Novartis data for probe set 36460_at. As described above, this data supposedly came from three replicates of each cell line. In fact, the data appears to come from four batches (given by the "ID" column); there are a total of 59 cell lines used, and four cell lines are run four times instead of three (and one cell line is only run twice).

```
> nov <- data.frame(novartisInfo, Signal = as.vector(as.matrix(novartis["36460_at",
+     ])))
> summary(nov)
```

```
 ID          cellname        pname       panelnbr       cellnbr
 A:59   HL-60(TB):  4   NSCLC   :27   Min.   : 1.0   Min.   : 1.000
 B:59   KM12     :  4   Melanoma:25   1st Qu.: 4.0   1st Qu.: 3.000
 C:58   OVCAR-5  :  4   Renal   :23   Median : 7.0   Median : 8.000
 D: 4   SK-MEL-2 :  4   Colon   :22   Mean   : 6.7   Mean   : 9.717
        A498     :  3   Breast  :21   3rd Qu.:10.0   3rd Qu.:15.000
        A549/ATCC:  3   Leukemia:19   Max.   :12.0   Max.   :29.000
        (Other)  :158   (Other) :43
    Signal
 Min.   :  7.131
 1st Qu.: 59.228
 Median : 77.881
```

```
 Mean   : 86.419
 3rd Qu.:110.200
 Max.   :262.895
```

```
> length(levels(nov[, "cellname"]))
```

```
[1] 59
```

For each column in Chemo predictors, we searched for a match in the Novartis data in order to figure out which cell lines—and which arrays—were the original source of the data. We were able to identify the source for all cell lines. We also note that, for those sources that we can identify, all data was taken from the "A" batch of Novartis arrays.

```
> x <- nov[, "Signal"]
> y <- as.vector(as.matrix(predictors["36460_at", ]))
> src <- rep(NA, length(y))
> for (i in 1:length(src)) {
+     temp <- which(x == y[i])
+     if (length(temp) > 1) {
+         stop("multiple hits")
+     }
+     else if (length(temp) == 1) {
+         src[i] <- temp
+     }
+ }
> predictorsInfo[, "SourceIndex"] <- src
> predictorsInfo[, "SourceID"] <- nov[src, "ID"]
> predictorsInfo[, "Source"] <- nov[src, "cellname"]
> rm(i, x, y, temp)
> predictorsInfo[, c(2, 3, 5, 6, 7)]
```

```
    drugName responseStatus SourceIndex SourceID          Source
1      Adria      Resistant          44        A          SF-539
2      Adria      Resistant          51        A          SNB-75
3      Adria      Resistant          27        A      MDA-MB-435
4      Adria      Resistant          30        A         NCI-H23
5      Adria      Resistant          23        A             M14
6      Adria      Resistant          24        A        MALME-3M
7      Adria      Resistant          45        A        SK-MEL-2
8      Adria      Resistant          46        A       SK-MEL-28
9      Adria      Resistant          47        A        SK-MEL-5
10     Adria      Resistant          58        A         UACC-62
11     Adria      Sensitive          34        A     NCI/ADR-RES
```

12	Adria	Sensitive	13	A	HCT-15
13	Adria	Sensitive	18	A	HT29
14	Adria	Sensitive	10	A	EKVX
15	Adria	Sensitive	31	A	NCI-H322M
16	Adria	Sensitive	19	A	IGROV1
17	Adria	Sensitive	35	A	OVCAR-3
18	Adria	Sensitive	36	A	OVCAR-4
19	Adria	Sensitive	37	A	OVCAR-5
20	Adria	Sensitive	38	A	OVCAR-8
21	Adria	Sensitive	48	A	SK-OV-3
22	Adria	Sensitive	6	A	CAKI-1
23	Etopo	Resistant	44	A	SF-539
24	Etopo	Resistant	5	A	BT-549
25	Etopo	Resistant	26	A	MDA-MB-231/ATCC
26	Etopo	Resistant	34	A	NCI/ADR-RES
27	Etopo	Resistant	15	A	HOP-62
28	Etopo	Resistant	29	A	NCI-H226
29	Etopo	Resistant	46	A	SK-MEL-28
30	Etopo	Resistant	57	A	UACC-257
31	Etopo	Resistant	1	A	786-0
32	Etopo	Sensitive	25	A	MCF7
33	Etopo	Sensitive	11	A	HCC-2998
34	Etopo	Sensitive	13	A	HCT-15
35	Etopo	Sensitive	53	A	SW-620
36	Etopo	Sensitive	31	A	NCI-H322M
37	Etopo	Sensitive	39	A	PC-3
38	Etopo	Sensitive	36	A	OVCAR-4
39	Etopo	Sensitive	37	A	OVCAR-5
40	5-FU	Resistant	25	A	MCF7
41	5-FU	Resistant	8	A	COLO 205
42	5-FU	Resistant	12	A	HCT-116
43	5-FU	Resistant	32	A	NCI-H460
44	5-FU	Resistant	22	A	LOX IMVI
45	5-FU	Resistant	47	A	SK-MEL-5
46	5-FU	Resistant	2	A	A498
47	5-FU	Resistant	59	A	UO-31
48	5-FU	Sensitive	34	A	NCI/ADR-RES
49	5-FU	Sensitive	27	A	MDA-MB-435
50	5-FU	Sensitive	53	A	SW-620
51	5-FU	Sensitive	10	A	EKVX
52	5-FU	Sensitive	23	A	M14
53	5-FU	Sensitive	38	A	OVCAR-8

54	5-FU	Sensitive	49	A	SN12C
55	Cytox	Resistant	20	A	K-562
56	Cytox	Resistant	28	A	MOLT-4
57	Cytox	Resistant	14	A	HL-60(TB)
58	Cytox	Resistant	25	A	MCF7
59	Cytox	Resistant	11	A	HCC-2998
60	Cytox	Sensitive	50	A	SNB-19
61	Cytox	Sensitive	17	A	HS 578T
62	Cytox	Sensitive	26	A	MDA-MB-231/ATCC
63	Cytox	Sensitive	27	A	MDA-MB-435
64	Cytox	Sensitive	29	A	NCI-H226
65	Cytox	Sensitive	23	A	M14
66	Cytox	Sensitive	24	A	MALME-3M
67	Cytox	Sensitive	45	A	SK-MEL-2
68	Topo	Resistant	44	A	SF-539
69	Topo	Resistant	51	A	SNB-75
70	Topo	Resistant	56	A	U251
71	Topo	Resistant	17	A	HS 578T
72	Topo	Resistant	15	A	HOP-62
73	Topo	Resistant	29	A	NCI-H226
74	Topo	Resistant	30	A	NCI-H23
75	Topo	Resistant	22	A	LOX IMVI
76	Topo	Resistant	38	A	OVCAR-8
77	Topo	Resistant	2	A	A498
78	Topo	Resistant	4	A	ACHN
79	Topo	Resistant	6	A	CAKI-1
80	Topo	Resistant	59	A	UO-31
81	Topo	Sensitive	20	A	K-562
82	Topo	Sensitive	40	A	RPMI-8226
83	Topo	Sensitive	27	A	MDA-MB-435
84	Topo	Sensitive	47	A	SK-MEL-5
85	Topo	Sensitive	11	A	HCC-2998
86	Topo	Sensitive	12	A	HCT-116
87	Topo	Sensitive	13	A	HCT-15
88	Topo	Sensitive	31	A	NCI-H322M
89	Topo	Sensitive	46	A	SK-MEL-28
90	Topo	Sensitive	8	A	COLO 205
91	Taxol	Resistant	43	A	SF-295
92	Taxol	Resistant	44	A	SF-539
93	Taxol	Resistant	17	A	HS 578T
94	Taxol	Resistant	27	A	MDA-MB-435
95	Taxol	Resistant	8	A	COLO 205

```
96     Taxol      Resistant          11        A        HCC-2998
97     Taxol      Resistant          18        A            HT29
98     Taxol      Resistant          35        A         OVCAR-3
99     Taxol      Resistant           9        A          DU-145
100    Taxol      Sensitive           7        A        CCRF-CEM
101    Taxol      Sensitive          53        A          SW-620
102    Taxol      Sensitive           3        A       A549/ATCC
103    Taxol      Sensitive          10        A            EKVX
104    Taxol      Sensitive          24        A        MALME-3M
105    Taxol      Sensitive          46        A       SK-MEL-28
106    Taxol      Sensitive          38        A         OVCAR-8
107    Taxol      Sensitive           1        A           786-0
108     Doce      Resistant          10        A            EKVX
109     Doce      Resistant          19        A          IGROV1
110     Doce      Resistant          36        A         OVCAR-4
111     Doce      Resistant           1        A           786-0
112     Doce      Resistant           6        A          CAKI-1
113     Doce      Resistant          49        A           SN12C
114     Doce      Resistant          55        A           TK-10
115     Doce      Sensitive          14        A       HL-60(TB)
116     Doce      Sensitive          44        A          SF-539
117     Doce      Sensitive          18        A            HT29
118     Doce      Sensitive          15        A          HOP-62
119     Doce      Sensitive          45        A        SK-MEL-2
120     Doce      Sensitive          47        A        SK-MEL-5
121     Doce      Sensitive          33        A        NCI-H522
```

We also wanted to know how many distinct cell lines were used, and to double check our assignment. First, label use.

```
> used <- rep(FALSE, nrow(nov))
> temp <- unique(src)
> used[temp] <- TRUE
> nov[, "Used"] <- used
> sum(nov[, "Used"])
```

```
[1] 53
```

Then, tabulate the results.

```
> attach(nov)
> table(ID, Used)
```

```
   Used
ID  FALSE TRUE
```

```
  A     6   53
  B    59    0
  C    58    0
  D     4    0
```

```
> detach()
```

5 The Affymetrix probe sets for the drugs match the Novartis data

Because we were concerned about the annotations, we loaded the full individual Novartis data into R for comparison with the data in the Chemo predictors table. As mentioned above, the Novartis file was in the form of a database dump, and needed to be reprocessed into matrix form before being read into R. This processing was performed using a perl script ("krc-parse.pl") that we wrote; this script is in this directory. It is an idiosyncracy of R that there is an extra column of all missing data at the end of this tab-separated-values file.

We now define a logical vector that picks out the rows in the Novartis data that are also included in the Chemo predictors. Since these are identified by the Affymetrix probe set ID, we should get 12,558 such rows.

```
> predictorsSubset <- rownames(novartis) %in% rownames(predictors)
> sum(predictorsSubset)
```

```
[1] 12558
```

We now confirm that the probe sets that were omitted were the controls:

```
> rownames(novartis)[!predictorsSubset]
```

```
 [1] "AFFX-BioB-3_at"              "AFFX-BioB-3_st"
 [3] "AFFX-BioB-5_at"              "AFFX-BioB-5_st"
 [5] "AFFX-BioB-M_at"              "AFFX-BioB-M_st"
 [7] "AFFX-BioC-3_at"              "AFFX-BioC-3_st"
 [9] "AFFX-BioC-5_at"              "AFFX-BioC-5_st"
[11] "AFFX-BioDn-3_at"             "AFFX-BioDn-3_st"
[13] "AFFX-BioDn-5_at"             "AFFX-BioDn-5_st"
[15] "AFFX-CreX-3_at"              "AFFX-CreX-3_st"
[17] "AFFX-CreX-5_at"              "AFFX-CreX-5_st"
[19] "AFFX-DapX-3_at"              "AFFX-DapX-5_at"
[21] "AFFX-DapX-M_at"              "AFFX-LysX-3_at"
[23] "AFFX-LysX-M_at"              "AFFX-M27830_3_at"
[25] "AFFX-M27830_5_at"            "AFFX-M27830_M_at"
[27] "AFFX-MurFAS_at"              "AFFX-MurIL10_at"
```

```
[29] "AFFX-MurIL2_at"              "AFFX-MurIL4_at"
[31] "AFFX-PheX-3_at"              "AFFX-PheX-5_at"
[33] "AFFX-PheX-M_at"              "AFFX-ThrX-3_at"
[35] "AFFX-ThrX-5_at"              "AFFX-ThrX-M_at"
[37] "AFFX-TrpnX-3_at"             "AFFX-TrpnX-5_at"
[39] "AFFX-TrpnX-M_at"             "AFFX-YEL002c/WBP1_at"
[41] "AFFX-YEL018w/_at"            "AFFX-YEL021w/URA3_at"
[43] "AFFX-YEL024w/RIP1_at"        "AFFX-hum_alu_at"
[45] "AFFX-LysX-5_at"              "AFFX-HSAC07/X00351_3_at"
[47] "AFFX-HSAC07/X00351_3_st"     "AFFX-HSAC07/X00351_5_at"
[49] "AFFX-HSAC07/X00351_5_st"     "AFFX-HSAC07/X00351_M_at"
[51] "AFFX-HSAC07/X00351_M_st"     "AFFX-HUMGAPDH/M33197_3_at"
[53] "AFFX-HUMGAPDH/M33197_3_st"   "AFFX-HUMGAPDH/M33197_5_at"
[55] "AFFX-HUMGAPDH/M33197_5_st"   "AFFX-HUMGAPDH/M33197_M_at"
[57] "AFFX-HUMGAPDH/M33197_M_st"   "AFFX-HUMISGF3A/M97935_3_at"
[59] "AFFX-HUMISGF3A/M97935_5_at"  "AFFX-HUMISGF3A/M97935_MA_at"
[61] "AFFX-HUMISGF3A/M97935_MB_at" "AFFX-HUMRGE/M10098_3_at"
[63] "AFFX-HUMRGE/M10098_5_at"     "AFFX-HUMRGE/M10098_M_at"
[65] "AFFX-HUMTFRR/M11507_3_at"    "AFFX-HUMTFRR/M11507_5_at"
[67] "AFFX-HUMTFRR/M11507_M_at"
```

Next, we confirm that the order of the rows (based on the probe set ID) is the same in the Chemo predictors data set and in the subset of the Novartis data:

```
> sum(rownames(predictors) == rownames(novartis)[predictorsSubset])
```

```
[1] 12558
```

```
> sum(rownames(predictors) != rownames(novartis)[predictorsSubset])
```

```
[1] 0
```

Now we simply pick one column in the Potti data and what should be the corresponding column in the Novartis data, and check that they contain the same numerical values.

```
> i <- 5
> identifier <- paste(predictorsInfo[i, "SourceID"], predictorsInfo[i,
+     "Source"], sep = ".")
> matching <- which(colnames(novartis) == identifier)
> x <- predictors[, i]
> y <- novartis[predictorsSubset, matching]
> summary(x - y)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
      0       0       0       0       0       0
```

Finally, clean up and arrange stuff for later.

```
> rm(i, identifier, matching, src, temp, used, x, y, dukeDir)
> predictorsInfo[, "NovartisName"] <- colnames(novartis)[match(predictors["36460_at",
+     ], novartis["36460_at", ])]
> predictorsInfo <- predictorsInfo[, c("index", "drugName",
+     "responseStatus", "Source", "NovartisName")]
> save(predictors, predictorsInfo, predictorsSubset, file = file.path("RDataObjects",
+     "chemoPredictors.Rda"))
```

6 Conclusions

The data in the Chemo predictors table can be identified as being taken directly from the individual Novartis data, with all columns coming from the "A" subset. The probe set IDs for this subset match what is in the Novartis data.