Matching the Potti and Chang Quantifications

Keith A. Baggerly and Kevin R. Coombes

November 13, 2007

1 Introduction

We weren't able to make good predictions of the Chang et al sample classes using the NCI60 data. Potti and Nevins note that they are still able to get good predictions. Checking their supplementary web site, `http://data.cgt.duke.edu/NatureMedicine.php`, we see that it now contains processed data for the docetaxel validation data, in "Breast data (n = 24).txt". We presume these are the values that they used to make their predictions.

It's possible that we missed something in the processing, so we'll try to mimic the steps involved. If we can reproduce their processed data, we may be able to understand our differences better. Thus, we'll load our numbers for the training and test data, then theirs, and try to align them.

2 Options and Loading Data

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

```
> load(file.path("RDataObjects", "chemoPredictors.Rda"))
> doceData <- predictors[, predictorsInfo$drugName == "Doce"]
> doceInfo <- predictorsInfo[predictorsInfo$drugName == "Doce",
+     ]
> rm("predictors", "predictorsInfo", "predictorsSubset")
> load(file.path("RDataObjects", "changData.Rda"))
> rm("changDChip", "changDChipNL", "changSoftNL")
> load(file.path("RDataObjects", "changGEOMapping.Rda"))
```

3 Loading the Duke Quantifications

The table of Duke data was initially in tab-delimited form, with an added comment in column AO, row 3 stating that "Data is standardized due to differences in mean signal intensities of training data and validations". We trimmed off this comment to get a more consistently formatted table for easier loading. There are two header lines indicating (row 1) whether the column is training (Doce) or testing (Chang), and (row 2) whether the column is Sensitive or Resistant.

```
> dukeHeader1 <- read.table(file.path("DukeWebSite", "Breast data (n = 24).txt"),
+     sep = "\t", nrows = 1, header = FALSE)
> dukeHeader1 <- as.vector(t(dukeHeader1))
> dukeHeader2 <- read.table(file.path("DukeWebSite", "Breast data (n = 24).txt"),
```

```
+     sep = "\t", skip = 1, nrows = 1, header = FALSE)
> dukeHeader2 <- as.vector(t(dukeHeader2))
> dukeDoce <- read.table(file.path("DukeWebSite", "Breast data (n = 24).txt"),
+     sep = "\t", skip = 2, header = FALSE)
> dukeHeader1

 [1] "Docetaxel 0" "0"           "0"           "0"           "0"
 [6] "0"           "0"           "1"           "1"           "1"
[11] "1"           "1"           "1"           "Docetaxel1"  "Chang2"
[16] "2"           "2"           "2"           "2"           "2"
[21] "2"           "2"           "2"           "2"           "2"
[26] "2"           "2"           "2"           "2"           "2"
[31] "2"           "2"           "2"           "2"           "2"
[36] "2"           "2"           "Chang2"

> dukeHeader2

 [1] "Sensitive" "Sensitive" "Sensitive" "Sensitive" "Sensitive" "Sensitive"
 [7] "Sensitive" "Resistant" "Resistant" "Resistant" "Resistant" "Resistant"
[13] "Resistant" "Resistant" "Resistant" "Resistant" "Resistant" "Resistant"
[19] "Resistant" "Resistant" "Resistant" "Resistant" "Resistant" "Resistant"
[25] "Resistant" "Sensitive" "Sensitive" "Sensitive" "Sensitive" "Sensitive"
[31] "Sensitive" "Sensitive" "Sensitive" "Sensitive" "Sensitive" "Sensitive"
[37] "Sensitive" "Sensitive"

> dim(dukeDoce)

[1] 12558    38

> dukeDoce[1:3, ]

    V1   V2   V3   V4   V5   V6   V7   V8   V9  V10  V11  V12  V13  V14  V15
1 0.54 2.21 0.59 1.28 1.85 1.20 0.87 1.97 0.22 1.53 0.09 2.18 1.12 3.92 1.00
2 0.29 4.95 0.77 2.55 2.14 1.53 0.43 1.14 3.87 0.15 0.45 1.59 0.82 0.63 0.43
3 5.39 1.83 0.19 1.59 0.55 0.54 0.78 0.77 0.20 2.20 1.04 0.71 5.00 1.15 1.48
   V16  V17  V18  V19  V20  V21  V22  V23  V24  V25  V26  V27  V28  V29  V30
1 1.43 1.10 7.50 0.07 0.84 1.41 1.91 1.45 1.65 1.24 0.84 1.20 2.56 1.20 2.26
2 0.67 1.54 0.65 0.72 0.37 0.17 0.60 1.95 4.50 5.01 0.37 1.66 3.20 2.59 2.99
3 1.45 1.74 1.20 1.26 0.98 0.80 2.71 0.01 1.19 3.37 0.98 0.86 0.90 0.78 2.18
   V31  V32  V33  V34  V35  V36  V37  V38
1 0.06 0.66 0.44 0.77 0.59 1.60 1.39 1.08
2 1.08 0.44 0.75 0.71 0.34 3.65 0.96 1.82
3 1.45 0.75 1.18 0.71 0.54 1.05 1.28 1.74

> range(dukeDoce)

[1]   0.01 109.38
```

The header rows match our description above. We note that the testing data columns have resistant first, then sensitive, which is reversed from the way we arranged the Chang data. We'll deal with this later.

Looking at the data, we have 12558 data rows (matching what we saw in our earlier analysis) and 38 data columns (14 training and 24 test). There are no probeset ids, and the sample ids are not present. The numerical values are all positive, and have been truncated to two decimal places. Looking at the first few rows suggests that the values might be on the log scale, but this isn't consistent with the range of the data going up to 109.38.

4 Checking the Training Data

The comment that they make about standardizing the data suggests treating the training and test data separately, with the goal of scaling the data to have mean 0 and variance 1. Given the positivity, we'll try log-transforming a row or two of the data and checking the behavior for the training set.

```
> log(dukeDoce[1, 1:14])
```

```
          V1        V2         V3        V4        V5        V6         V7
1 -0.6161861 0.7929925 -0.5276327 0.2468601 0.6151856 0.1823216 -0.1392621
         V8        V9       V10       V11       V12       V13      V14
1 0.6780335 -1.514128 0.4252677 -2.407946 0.7793249 0.1133287 1.366092
```

```
> mean(log(t(dukeDoce[1, 1:14])))
```

```
[1] -0.0004105719
```

```
> var(log(t(dukeDoce[1, 1:14])))
```

```
          1
1 0.9997468
```

This suggests that the training data values were log-transformed, centered and scaled row by row, and then exponentiated to give the data reported, with truncation happening at some step.

While we aren't sure about the row and column ordering, we will assume that it matches that from the table of chemo predictors we've worked with before.

```
> doceDataStd <- t(scale(t(log(doceData))))
> mismatchMax <- apply(abs(exp(doceDataStd) - dukeDoce[, 1:14]),
+     1, max)
> summary(mismatchMax)
```

```
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
0.002524 0.004529 0.004756 0.004666 0.004897 0.005000
```

All in all, this is as close to a perfect match as we're going to get, given that the tabulated data has been truncated to two decimal places. Indeed, the fact that the data agrees this well (the max absolute deviation is 0.005) suggests that truncation occurred pretty late in the process. In any event, we can see how these numbers were derived from the raw data.

However, there is a problem.

```
> doceInfo
```

```
    index drugName responseStatus    Source NovartisName
108   108     Doce      Resistant      EKVX       A.EKVX
109   109     Doce      Resistant    IGROV1     A.IGROV1
110   110     Doce      Resistant   OVCAR-4    A.OVCAR-4
111   111     Doce      Resistant     786-0      A.786-0
112   112     Doce      Resistant    CAKI-1     A.CAKI-1
113   113     Doce      Resistant     SN12C      A.SN12C
114   114     Doce      Resistant     TK-10      A.TK-10
115   115     Doce      Sensitive HL-60(TB)  A.HL-60(TB)
116   116     Doce      Sensitive    SF-539     A.SF-539
117   117     Doce      Sensitive      HT29       A.HT29
118   118     Doce      Sensitive    HOP-62     A.HOP-62
119   119     Doce      Sensitive  SK-MEL-2   A.SK-MEL-2
120   120     Doce      Sensitive  SK-MEL-5   A.SK-MEL-5
121   121     Doce      Sensitive  NCI-H522   A.NCI-H522
```

```
> dukeHeader2
```

```
 [1] "Sensitive" "Sensitive" "Sensitive" "Sensitive" "Sensitive" "Sensitive"
 [7] "Sensitive" "Resistant" "Resistant" "Resistant" "Resistant" "Resistant"
[13] "Resistant" "Resistant" "Resistant" "Resistant" "Resistant" "Resistant"
[19] "Resistant" "Resistant" "Resistant" "Resistant" "Resistant" "Resistant"
[25] "Resistant" "Sensitive" "Sensitive" "Sensitive" "Sensitive" "Sensitive"
[31] "Sensitive" "Sensitive" "Sensitive" "Sensitive" "Sensitive" "Sensitive"
[37] "Sensitive" "Sensitive"
```

The resistant samples are listed first and the sensitive samples listed second in our data from the chemo predictors, but the processed data reverses this labeling. Checking the "Description of the Process of Predictor generation.doc" file, we see that the chemo predictor labeling is correct, and the labeling given in the processed data is wrong.

5 Checking the Test Data

Here, we're pretty sure we know what the probesets should be, but we're not as sure about the identities of the individual samples. Since we put responders before nonresponders, and they did the reverse, there's bound to be some reorganization even before we deal with the two reclassified samples.

We begin by standardizing the Chang data just as we did the cell line data.

```
> changDataStd <- t(scale(t(log(changSoft))))
```

Since we think we know what the probesets are, it's natural to check the correlation between the columns of changDataStd and those of dukeDoce. Let's try this.

```
> temp <- exp(changDataStd[rownames(doceDataStd), ])
```

This works ok, but when we try the following line (not evaluated here) we get a problem.

```
> tempCor <- cor(temp, dukeDoce[, 15:38])
```

Specifically, we get the error message "Error in cor(temp, dukeDoce[, 15:38]) : missing observations in cov/cor". Time to find the missing observations.

```
> sum(is.na(temp))

[1] 240

> tempLoc <- which(is.na(changDataStd), arr.ind = TRUE)
> table(tempLoc[, 1])

 1116  1325  1691  1962  2464  3910  6758  7475 11549 12085
   24    24    24    24    24    24    24    24    24    24

> table(tempLoc[, 2])

 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
```

Evidently, we have some problems with the data in 10 rows of changDataStd. Time to figure out what's going on.

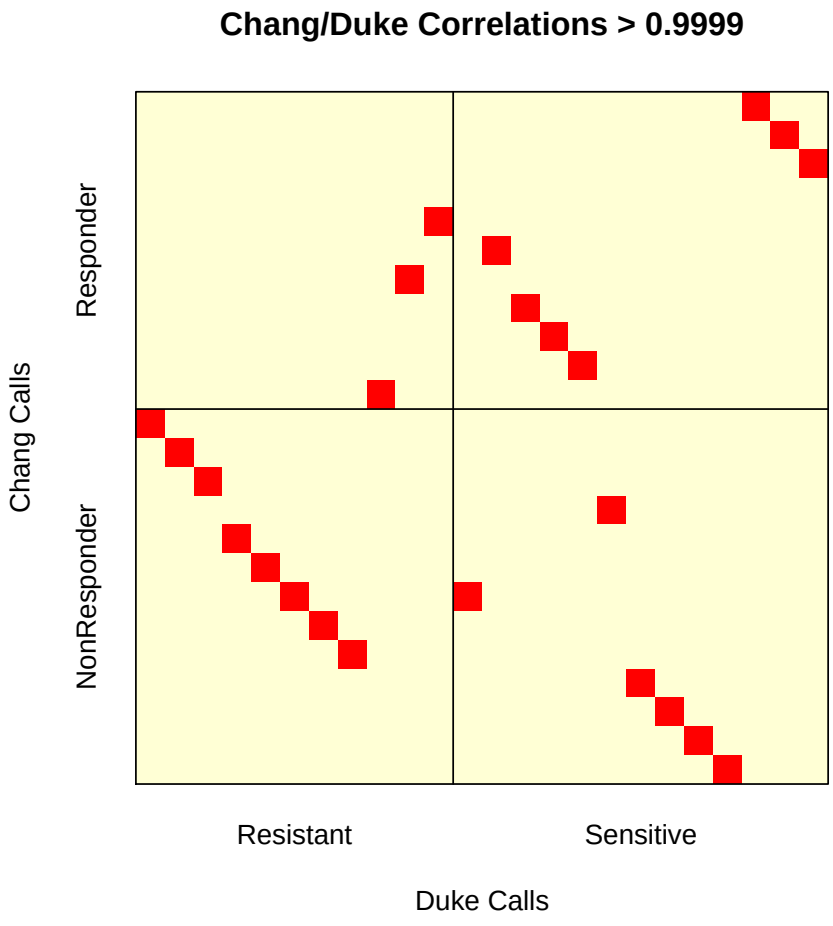
```
> changSoft[rownames(doceDataStd)[tempLoc[1, 1]], ]

         GSM4903 GSM4907 GSM4908 GSM4914 GSM4915 GSM4917 GSM4919 GSM4920
37566_at 5.89822 5.89822 5.89822 5.89822 5.89822 5.89822 5.89822 5.89822
         GSM4921 GSM4923 GSM4913 GSM4901 GSM4902 GSM4904 GSM4905 GSM4906
37566_at 5.89822 5.89822 5.89822 5.89822 5.89822 5.89822 5.89822 5.89822
         GSM4909 GSM4910 GSM4911 GSM4912 GSM4916 GSM4918 GSM4922 GSM4924
37566_at 5.89822 5.89822 5.89822 5.89822 5.89822 5.89822 5.89822 5.89822

> dukeDoce[tempLoc[1, 1], 15:38]

      V15  V16  V17  V18  V19  V20  V21  V22  V23  V24  V25  V26  V27  V28  V29
1116 2.66 2.66 2.66 2.66 2.66 2.66 2.66 2.66 2.66 2.66 2.66 2.66 2.66 2.66 2.66
      V30  V31  V32  V33  V34  V35  V36  V37  V38
1116 2.66 2.66 2.66 2.66 2.66 2.66 2.66 2.66 2.66
```

Ah. The initial Chang quantifications used dChip, and there was a lower bound to the values that could be obtained. For these 10 probesets, all samples hit the lower bound. There's no variation, so standardization explodes. If we look at the corresponding extries from dukeDoce, they're all close to exp(1) (2.66 instead of 2.71), so rows of NAs in log space were (we think) replaced with 1s. Now that we know what's going on, a fix is simple.

```
> temp[is.na(temp)] <- exp(1)
> tempCor <- cor(temp, dukeDoce[, 15:38])
```

Now let's take a look at where the correlations are really high. Due to the reversal, we expect all of the first 11 columns of changSoft to map to some of the last 13 columns in dukeDoce, and we expect 11 of the last 13 columns in changSoft to map to the first 11 columns in dukeDoce.

```
> oldPin <- par()$pin
> par(pin = c(min(oldPin), min(oldPin)))
> image(1:24, 1:24, t(tempCor < 0.9999), ylab = "Chang Calls",
+     xlab = "Duke Calls", main = "Chang/Duke Correlations > 0.9999",
+     ylim = c(24.5, 0.5), axes = FALSE)
> abline(h = 11.5, v = 11.5)
> axis(1, at = c(6, 18), labels = c("Resistant", "Sensitive"),
+     tick = 0)
> axis(2, at = c(6, 18), labels = c("Responder", "NonResponder"),
+     tick = 0)
> box()
> par(pin = oldPin)
```



There a quite a few problems here. The top 11 rows correspond to the samples that Chang et al designated as Responders, and the bottom 13 are NRs. The leftmost 11 columns are labeled in the Duke table as Resistant (NR), and the last 13 as Sensitive (Responders). According to notes posted on the Duke

website, they reclassified two of the 13 Chang NRs as Sensitive. Thus,

- There should be 11 squares in the upper right, as all of the Chang Responders should be classed as Sensitive. There are 7.
- There should not be any squares in the upper left, by the same reasoning as above. There are 3.
- There should be 11 squares in the top 11 rows. There are 10; there is no match for the fourth Chang sample.
- There should be 11 squares in the bottom left, as 11 of the 13 Chang NRs should be classed as Resistant. There are 8.
- There should be 2 squares in the bottom right, as 2 of the 13 Chang NRs should be reclassed as Sensitive. There are 6.
- There should be 13 squares in the bottom 13 rows. There are 14; Chang sample 18 is present in two columns of the Duke table (once as Sensitive and once as Resistant, a neat trick).

By our count, 4 of the 11 Chang Responders have been misclassified (3 are called Resistant, and 1 is omitted). Similarly, 6 of the 13 Chang NRs have been misclassified (5 are called Sensitive and Chang sample 18 is classified both ways).

6 More Details

There are one or two questions that we wish to touch on briefly before closing.

6.1 Does Relabeling Matter?

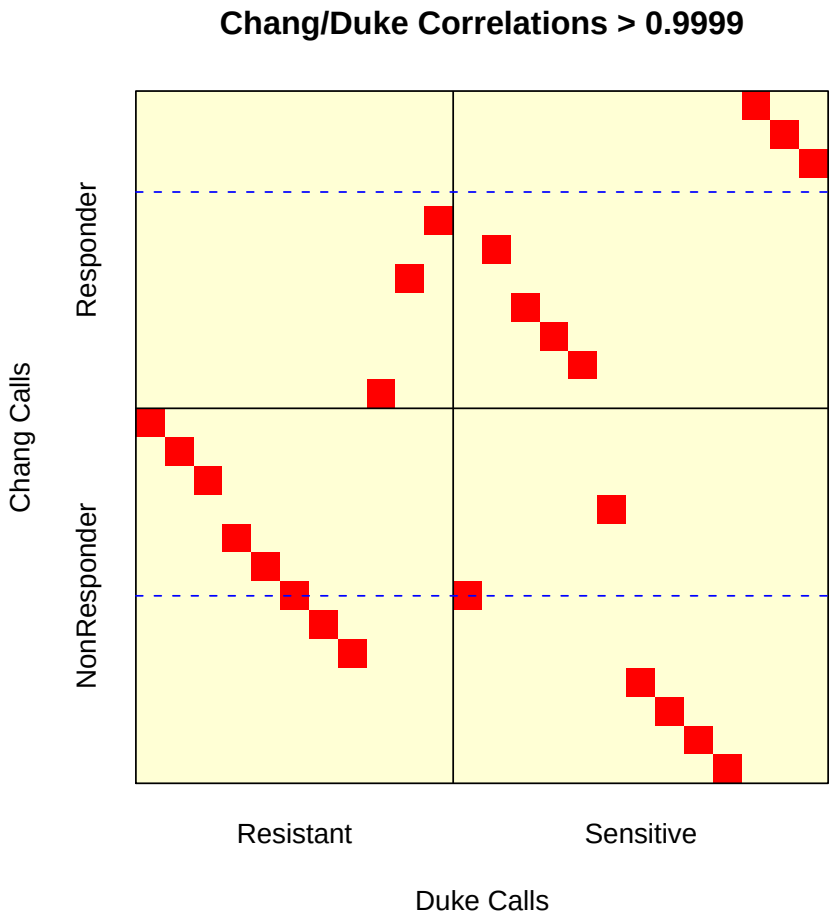
They relabeled two of the samples. What effect did this have? The two samples needing to be relabeled are N12 and N13. The corresponding Chang GEO names are given below.

```
> changGEOMapping[c("N12", "N13")]

      N12       N13
"GSM4912" "GSM4918"

> match(changGEOMapping[c("N12", "N13")], rownames(tempCor))

[1] 20 22

> which.max(tempCor[20, ])

V22
  8

> which.max(tempCor[22, ])

V33
 19
```

In the image, the rows corresponding to the samples needing their status labels swapped are rows 20 and 22. However, looking at the columns that these map to, one is clearly labeled as Resistant and the other as Sensitive. Relabeling these two samples has no real effect on the amount of discordance.

6.2 Are the Apparent Ties Exact?

The missing and tied columns bother us. Let's take a look at the figure again, highlighting the problem rows.



The two tied entries are in the sixth and twelfth columns of the test data, or the 20th and 26th columns of the Duke table. Let's confirm that they are completely tied.

```
> sum(dukeDoce[, 20] == dukeDoce[, 26])
```

```
[1] 12558
```

The agreement is indeed perfect.

6.3 When Were the Ties Introduced?

Given that two columns have been replicated, when did this replication take place? We may be able to address this by looking at how well the Chang and Duke numbers agree when they do. For a few case studies, we note that the first three of the Chang samples are the last three in the Duke set.

```
> min(apply(tempCor, 2, max))

[1] 0.9999968

> sum(abs(temp[, 1] - dukeDoce[, 36]) > 0.0050001)

[1] 10

> sum(abs(temp[, 2] - dukeDoce[, 37]) > 0.0050001)

[1] 10

> sum(abs(temp[, 3] - dukeDoce[, 38]) > 0.0050001)

[1] 10

> temp[which(abs(temp[, 1] - dukeDoce[, 36]) > 0.0050001), 1]

   37566_at    37773_at    38136_at    38405_at  38902_r_at    40335_at
   2.718282    2.718282    2.718282    2.718282    2.718282    2.718282
160021_r_at    31359_at    35394_at    35924_at
   2.718282    2.718282    2.718282    2.718282

> dukeDoce[which(abs(temp[, 1] - dukeDoce[, 36]) > 0.0050001),
+     36]

 [1] 2.66 2.66 2.66 2.66 2.66 2.66 2.66 2.66 2.66 2.66
```

The only cases that disagree by more than the amount induced by truncation are those where we replaced the entire set of tied values with exp(1), and there we have the difference between 2.66 (theirs) and 2.71828 (ours). The centered and scaled values would not agree this well if the ties were present before the centering and scaling; the mixup was introduced later. Some related checks:

```
> summary(apply(log(dukeDoce[, 1:14]), 1, mean))

      Min.    1st Qu.     Median       Mean    3rd Qu.       Max.
-8.859e-03 -1.288e-03 -2.478e-05 -2.455e-05  1.260e-03  8.017e-03

> summary(apply(log(dukeDoce[, 15:38]), 1, mean))

     Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
-0.205900 -0.025050  0.004787  0.009964  0.042060  0.978300

> -sort(apply(-log(dukeDoce[, 15:38]), 1, mean))[11]

    10117
0.2429957

> dukeDoce[10117, 15:38]

       V15  V16  V17  V18  V19  V20  V21  V22  V23  V24  V25  V26  V27  V28
10117 0.62 0.72 0.38 0.88 1.28 40.3 1.09 1.72 0.94 0.47 0.68 40.3 0.74 2.03
       V29  V30  V31  V32  V33  V34  V35 V36  V37  V38
10117 1.93 1.57 0.65 0.77 1.39 0.95 1.29 1.4 0.63 0.57
```

The mean values are smaller for the training set (where no mixup occurred). The 10 biggest mean values are associated with the exp(1) replacement noted above, so we check the eleventh. Looking at the data for this row, we see that the two tied values by far the largest ones for this probeset, introducing the distortion.

7 Conclusions

The sensitive/resistant labeling for the training data has been reversed. The sensitive/resistant labeling for the test data samples has been scrambled for 10/24 samples, so we have difficulty with the high levels of accuracy claimed. Some data columns are used twice, and others not at all.

The posted data is wrong.

8 Appendix

8.1 Saves

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] "stats"     "graphics"  "grDevices" "utils"     "datasets"  "methods"
[7] "base"

other attached packages:
R.matlab     R.oo
 "1.1.3"  "1.3.0"
```