Analysis of Microarray Data

Keith Baggerly and Kevin Coombes
Department of Bioinformatics and Computational Biology
UT M. D. Anderson Cancer Center
kabagg@mdanderson.org
kcoombes@mdanderson.org

6 November 2007
Lecture 20: Genome Browsing

- Learning What BioConductor Contains
- Annotation Environments in R
- AnnBuilder: Rolling Your Own Annotations
- The UCSC Genome Browser
- Chromosome Locations
- Building a Custom Track
- Viewing Your Custom Track
Learning What BioConductor Contains

We are developing (i.e., it is not completed, so may behave strangely at times) a graphical tool to browse through the BioConductor documentation.

http://jonas.mdanderson.org/pub/bioconductor/
The Documentation Graph
Hovering the Mouse Gives a Summary
Biobase

Biobase: Base functions for Bioconductor

Functions that are needed by many other packages or which replace R functions. Suggests: widgetTools, tkWidgets

Author       R. Gentleman, V. Carey, M. Morgan, S. Falcon
Maintainer   Biocon Team

Vignettes (Documentation)

<table>
<thead>
<tr>
<th>Vignette</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Biobase.pdf</td>
<td></td>
</tr>
<tr>
<td>Biocenter.pdf</td>
<td></td>
</tr>
<tr>
<td>esApply.pdf</td>
<td></td>
</tr>
<tr>
<td>HowTo.pdf</td>
<td></td>
</tr>
</tbody>
</table>

Package Downloads

<table>
<thead>
<tr>
<th>Package</th>
<th>Source</th>
<th>OS X binary</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Windows binary</td>
<td>Biobase 1.10.1.zip</td>
</tr>
<tr>
<td>Biobase.pdf</td>
<td></td>
<td>Biobase 1.10.1.tar.gz</td>
</tr>
</tbody>
</table>

Details

<table>
<thead>
<tr>
<th>Category</th>
<th>Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>biocViews</td>
<td>Infrastructure, Statistics</td>
</tr>
<tr>
<td>Depends</td>
<td>R, tools, methods</td>
</tr>
<tr>
<td>Suggests</td>
<td></td>
</tr>
<tr>
<td>Imports</td>
<td></td>
</tr>
<tr>
<td>SystemRequirements</td>
<td></td>
</tr>
</tbody>
</table>

Left-click Takes You to the Documentation
Left-click Also Recenters on a New Selection
Right-click Lets You Hide Part of the Graph
Hiding BioBase Often Clarifies the Structure
Hubs in the Documentation Graph Are Probably Important

We talked about the annotate package previously. It is clear from the graph that this is a central “hub” upon which many of the annotation-related packages depend. (We can also see that affy is another hub, defining the basic tools for Affymetrix arrays, and that the multtest package for multiple testing is another hub.)

One of the annotation tools that is worth exploring is biomaRt, but we are going to leave that for another time. If you want to find out more about the BioMart project, go to http://www.biomart.org.

Right now, we want to look at the AnnBuilder package.
Documentation for the AnnBuilder Package

AnnBuilder

Bioconductor annotation data package builder

Processing annotation data from public data repositories and building annotation data packages or XML data documents using the source data.

Author: J. Zhang
Maintainer: J. Zhang

Vignettes (Documentation)
- ABPrimer.pdf
- AnnBuilder.pdf

Package Downloads

<table>
<thead>
<tr>
<th>Source</th>
<th>AnnBuilder 1.10.5.tar.gz</th>
</tr>
</thead>
<tbody>
<tr>
<td>Windows</td>
<td>AnnBuilder 1.10.5.zip</td>
</tr>
<tr>
<td>OS X</td>
<td>AnnBuilder 1.10.5.tgz</td>
</tr>
</tbody>
</table>

Details

- biocViews: Annotation, Microarray
- Depends: R, methods, Biobase, XML, annotate, utils, RSQLite
- Suggests
- Imports
- SystemRequirements

© Copyright 2004–2007 Kevin R. Coombes and Keith A. Baggerly

GS01 0163: Analysis of Microarray Data
Annotation Environments in R

For most Affymetrix arrays, annotation packages are available directly (and automatically) from BioConductor whenever you need them. These packages were built using AnnBuilder.

You can load one of these packages as follows:

> require(hgu95av2)

[1] TRUE

To see what is in an annotation package, use its name as a function:

> hgu95av2()

Quality control information for hgu95av2
Date built: Created: Mon Apr 23 12:21:36 2007
Number of probes: 12625
Probe number mismatch: None
Probe mismatch: None

Mappings found for probe based rda files:
  hgu95av2ACCNUM found 12625 of 12625
  hgu95av2CHR found 12149 of 12625
  hgu95av2CHRLOC found 11730 of 12625
  hgu95av2ENZYME found 1861 of 12625
  hgu95av2ENTREZID found 12225 of 12625
  hgu95av2GENENAME found 12161 of 12625
  hgu95av2GO found 11421 of 12625
  hgu95av2MAP found 12121 of 12625
  hgu95av2OMIM found 10157 of 12625
  hgu95av2PATH found 4322 of 12625
  hgu95av2PFAM found 12046 of 12625
hgu95av2PMID found 12120 of 12625
hgu95av2PROSITE found 12046 of 12625
hgu95av2REFSEQ found 12004 of 12625
hgu95av2SYMBOL found 12161 of 12625
hgu95av2UNIGENE found 11973 of 12625

Mappings found for non-probe based rda files:
  hgu95av2CHRLENGTHS found 25
  hgu95av2ENZYME2PROBE found 677
  hgu95av2GO2ALLPROBES found 7501
  hgu95av2GO2PROBE found 5339
  hgu95av2PATH2PROBE found 189
  hgu95av2PMID2PROBE found 127350
Getting Annotatons From Environments

Each of the items in the package is an environment, which computer scientists may recognize better if we tell them it is a hash table. The key into the probe-based hash table environments is the manufacturers identifier (i.e., an Affymetrix probe set id such as 1854_at.

> get("1854_at", hgu95av2ACCNUM)

[1] "X13293"

> get("1854_at", hgu95av2UNIGENE)

[1] "Hs.179718"
> get("1854_at", hgu95av2CHR)

[1] "20"

> get("1854_at", hgu95av2MAP)

[1] "20q13.1"

> get("1854_at", hgu95av2CHRLOC)

        20
41729122

> get("1854_at", hgu95av2SYMBOL)

[1] "MYBL2"
> get("1854_at", hgu95av2GENENAME)

[1] "v-myb myeloblastosis viral oncogene homolog (avian)-like"

> get("1854_at", hgu95av2ENTREZID)

[1] 4605

We have also talked previously about how to find the probe set ids if you start with a gene symbol or a UniGene cluster id.
AnnBuilder: Rolling Your Own Annotations

We recently had to analyze some data from an Agilent 44K two-color glass microarray. The corresponding annotation package was not available, so we had to build our own. Finding the manufacturer’s basic annotations was a nontrivial task. We started at the web site (http://www.agilent.com), then followed the link under “Products and Services” for “Life Sciences” to get to the “DNA Microarrays” page.
Follow the Link for “Whole Human Genome”
Follow the Link for “Download Gene Lists”
Reading the Feature Info

In any event, we finally obtained a pair of files that contained the mappings from spots to genomic material. (In addition to the “download gene lists”, you can also follow the link to “Download design files”, but this will only work if you know one of the barcodes on the slides.) We used the `read.table` command to get this file into R:

```r
> featureInfo <- read.table("012391_D_DNAFront_BCBottom_20050601.tsv",
+ header = TRUE, row.names = NULL, sep = "\t",
+ quote = "", comment.char = "")
```
Looking at the Feature Info

Here is part of the file:

```r
> colnames(featureInfo)

[1] "Column"    "Row"    "Name"    "ID"
[5] "RefNumber" "ControlType" "GeneName" "TopHit"
[9] "Description"

> featureInfo[1:5, 1:4]

<table>
<thead>
<tr>
<th>Column</th>
<th>Row</th>
<th>Name</th>
<th>ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>103</td>
<td>426 NM_001003689</td>
<td>A_23_P80353</td>
</tr>
<tr>
<td>4</td>
<td>103</td>
<td>424 NM_005503</td>
<td>A_23_P158231</td>
</tr>
<tr>
<td>5</td>
<td>103</td>
<td>422 NM_004672</td>
<td>A_32_P223017</td>
</tr>
</tbody>
</table>
```
The critical information is given by the columns that contain the manufacturers identifier (ID) and the GenBank or RefSeq accession number (Name). The function we are going to use to build annotations requires only these two columns (in the reverse order) to be present in a file. So we make them available:

```r
> temp <- featureInfo[, c(4, 3)]
> write.table(temp, "agilentGenes.tsv", sep = "\t",
+             quote = FALSE, col.names = NA)
```
Setting Up the Annotation Package

> library(AnnBuilder)
> baseName <- "agilentGenes.tsv"
> baseType <- "gb"
> srcUrls <- getSrcUrl("all", organism = "Homo sapiens")
> myDir <- getwd()
Building the Annotation Package

The next command takes a very long time, since it makes calls to databases all over the internet for every one of the 44,000 probes on the array. Be prepared to go get lunch while it executes.

```r
ABPkgBuilder(baseName = baseName, srcUrls = srcUrls,
baseMapType = baseType, pkgName = "Agilent44K",
pkgPath = myDir, organism = "Homo sapiens",
version = "1.0", author = list(authors = "krc@mdacc.tmc.edu",
maintainer = "krc@mdacc.tmc.edu"), fromWeb = TRUE)
```
Producing the Final Package

This command produces the **source** for a package, which must still be compiled and zipped into a binary package that can be installed easily. This task is most easily accomplished on a UNIX based machine:

```
helios% R CMD build Agilent44K
helios% R CMD build --binary Agilent44K
```

You can then convert the resulting `.tar.gz` file to a `.zip` file, which is the preferred form for distributing a Windows package.

You can check out the results by getting the annotation package from our course web site.
The Agilent 44K Annotations

> library(Agilent44K)
> Agilent44K()

Quality control information for Agilent44K
Date built: Created: Sun Sep 03 07:50:38 2006

Number of probes: 41001
Probe number mismatch: None
Probe mismatch: None
Mappings found for probe based rda files:
  Agilent44KACCNUM found 41001 of 41001
  Agilent44KCHR found 31185 of 41001
  Agilent44KCHRLOC found 28795 of 41001
  Agilent44KENZYME found 3056 of 41001
Introduction to Microarrays

Agilent44KGENENAME found 27824 of 41001
Agilent44KGO found 23644 of 41001
Agilent44KLOCUSID found 31224 of 41001
Agilent44KMAP found 30939 of 41001
Agilent44KOMIM found 17942 of 41001
Agilent44KPATH found 6715 of 41001
Agilent44KPMID found 30361 of 41001
Agilent44KREFSEQ found 30057 of 41001
Agilent44KSUMFUNC found 0 of 41001
Agilent44KSYMBOL found 31217 of 41001
Agilent44KUNIGENE found 31010 of 41001

Mappings found for non-probe based rda files:
    Agilent44KCHRLENGTHS found 25
    Agilent44KENZYME2PROBE found 794
    Agilent44KGO2ALLPROBES found 6883
    Agilent44KGO2PROBE found 5117
Agilent44KORGANISM found 1
Agilent44KPATH2PROBE found 183
Agilent44KPFAM found 21902
Agilent44KPMID2PROBE found 131104
Agilent44KPROSITE found 15055
The UCSC Genome Browser

We are going to shift gears slightly:

http://genome.ucsc.edu/
Follow the Link to “Genome Browser”

About the Human Mar. 2006 (hg18) assembly (sequences)

The March 2006 human reference sequence (NCBI Build 36.1) was produced by the International Human Genome Sequencing Consortium.

Sample position queries

A genome position can be specified by the accession number of a sequenced genomic clone, an mRNA or EST or STS marker, or a cytological band, a chromosomal coordinate range, or keywords from the GenBank description of an mRNA. The following list shows examples of valid position queries for the human genome. See the User’s Guide for more information.

Request: Genome Browser Response:

chr7 Displays all of chromosome 7
Introduction to Microarrays

Press “Submit” to Start Browsing
About the Genome Browser

The genome browser lets you see a great deal of information laid out along the latest completed build of the human genome. The most obvious thing to look at are the known genes, which are typically displayed in such a way that you can see the individual introns and exons (provided you zoom in closely).

For our purposes (as people who analyze microarray data), an extremely interesting feature of the Genome Browser is that it lets you add your own “Custom Tracks”, which is their name for a set of annotations you can define.
Custom Tracks

To learn about the genome (custom) tracks, go to the FAQ.
BED Format

BED format provides a flexible way to define the data lines that are displayed in an annotation track. BED lines have three required fields and nine additional optional fields. The number of fields per line must be consistent throughout any single set of data in an annotation track.

The first three required BED fields are:

1. chrom - The name of the chromosome (e.g. chr3, chrY, chr2_random) or scaffold (e.g. scaffold10671).
2. chromStart - The starting position of the feature in the chromosome or scaffold. The first base in a chromosome is numbered 0.
3. chromEnd - The ending position of the feature in the chromosome or scaffold. The chromEnd base is not included in the display of the feature. For example, the first 100 bases of a chromosome are defined as chromStart=0, chromEnd=100, and span the bases numbered 0-99.

The 9 additional optional BED fields are:

4. name - Defines the name of the BED line. This label is displayed to the left of the BED line in the Genome Browser window when the track is open to full display mode or directly to the left of the item in pack mode.
5. score - A score between 0 and 1000. If the track line useScore attribute is set to 1 for this annotation data set, the score value will determine the level of gray in which this feature is displayed (higher numbers = darker gray).
6. strand - Defines the strand - either ‘+’ or ‘-’.
7. thickStart - The starting position at which the feature is drawn thickly (for example, the start codon in gene displays).
8. thickEnd - The ending position at which the feature is drawn thickly (for example, the stop codon in gene displays).
9. itemRgb - An RGB value of the form R,G,B (e.g. 255,0,0). If the track line itemRgb attribute is set to “On”, this RGB value
Chromosome Locations

You can read more of the custom track documentation on your own; here, we are going to focus on how to build a custom track in R. The first thing we want to point out is that we need to know both the starting base location and the ending base location in order to build a custom track. Thus, the CHRLOC annotations that the AnnBuilder BioConductor package constructs are not adequate.

Fortunately, we can get start and end points directly from the folks at the UCSC Genome Browser. Go back to the main page, then follow the link for “Downloads”.
Sequence and Annotation Downloads

This page contains links to sequence and annotation data downloads for the genome assemblies featured in the UCSC Genome Browser. Table downloads are also available via the Genome Browser FTP server. For quick access to the most recent assembly of each genome, see the current genomes directory. This directory may be useful to individuals with automated scripts that must always reference the most recent assembly.

To view the current descriptions and formats of the tables in the annotation database, use the "describe table schema" button in the Table Browser. The Description of the annotation database page (infrequently maintained) also provides descriptions of selected tables in the database.

All tables in the Genome Browser are freely usable for any purpose except as indicated in the README.txt files in the download directories. To view restrictions specific to a particular data set, click on the corresponding download link and review the README text. These data were contributed by many researchers, as listed on the Genome Browser credits page. Please acknowledge the contributor(s) of the data you use.

- Human
- Chimpanzee
- Rhesus
- Dog
- Cow
- Mouse
- Rat
Follow the link for “Human”

- Full data set
- Data set by chromosome
- Annotation database
- Human self alignments
- Human/Chimp (panTro2) pairwise alignments
- Human/Chimp (panTro1) pairwise alignments
- Human/Rhesus (rheMac2) pairwise alignments
- Human/Cow (bosTat2) pairwise alignments
- Human/Dog (canFam2) pairwise alignments
- Human/Mouse (mm8) pairwise alignments
- Human/Mouse (mm7) pairwise alignments
- Human/Rat (m4) pairwise alignments
- Human/Opossum (monDom4) pairwise alignments
- Human/Chicken (galGal3) pairwise alignments
- Human/Chicken (galGal2) pairwise alignments
- Human/Zebrafish (danRer4) pairwise alignments
In “Annotation Database”, Scroll To “refGene”
Using the RefGene locations in R

Load the file.

```r
> refgene <- read.table("refGene.txt", header = FALSE,
+     sep = "\t", comment.char = "", quote = "")
```

Add the column names, which are not included.

```r
> colnames(refgene) <- c("bin", "name", "chrom",
+     "strand", "txStart", "txEnd", "cdsStart",
+     "cdsEnd", "exonCount", "exonStarts", "exonEnds",
+     "id", "name2", "cdsStartStat", "cdsEndStat",
+     "exonFrames")
```

We are going to ignore the intron and exon boundaries. We are also going to remove duplicate entries, which seem for some reason to exist;
the search to identify these is time consuming.

```r
> temprg <- refgene[, c(1:9, 13:15)]
> omit <- unlist(lapply(levels(temprg$name), function(x, n) {
+     which(n == x)[1]
+ }, as.character(temprg$name)))
> summary(omit)
> refgene <- temprg[omit, ]
> rownames(refgene) <- as.character(refgene[, "name"])
```

Finally, we save this as a binary object that we can load later.

```r
> save(refgene, file = "refgene.rda")
```
Linking the Agilent Array to RefGene locations

First, convert the environment in the AnnBuilder package for the Agilent 44K arrays to a list.

```r
> temp2 <- as.list(Agilent44KREFSEQ)
```

Next, we produce a list that maps the annotations to the spots. This code works because the ID column of the `featureInfo` object contains RefSeq IDs (primarily), which are the names of the rows in the `temp2` object we just created.

```r
> ag.annoList <- temp2[as.character(featureInfo[, +   "ID"])]
```
Alternative Splicing

> ag.annoList[1]

$A_23_P80353
[1] "NM_001003689" "NP_001003689" "NM_031488"
[4] "NP_113676"

Notice that some probes are associated with more than one RefSeq gene; this happens because different isoforms (produced by alternative splicing) of the same gene have different RefSeq identifiers. That is, the same piece of DNA can give rise to different mRNA molecules. So, we now search through and select just the first annotation for each spot.

> agilent.lc <- unlist(lapply(ag.annoList, length))
> agilentREFSEQ <- unlist(lapply(ag.annoList, function(x) { 

\[ Copyright 2004–2007 Kevin R. Coombes and Keith A. Baggerly
+    if (length(x) == 0) {
+        return(NA)
+    }
+    if (length(x) == 1) {
+        return(x)
+    }
+    idx <- 1
+    while (idx <= length(x)) {
+        if (x[[idx]] == "") {
+            idx <- idx + 1
+            next
+        }
+        return(x[[idx]])
+    }
+    return(NA)
+})
> agilentREFSEQ[agilentREFSEQ == ""] <- NA

> length(agilentREFSEQ)

[1] 41675

> sum(!is.na(agilentREFSEQ))

[1] 30612

Finally, we use the updated RefSeqs (that we just constructed in the agilentREFSEQ object) as indices into the refgene chromosome locations above. This computation is also slow, since it uses a search in a list instead of in a hash.

> agilent2refgene <- refgene[agilentREFSEQ, ]
> `agilent2refgene[1:3, ]`

<table>
<thead>
<tr>
<th>bin</th>
<th>name</th>
<th>chrom</th>
<th>strand</th>
<th>txStart</th>
<th>txEnd</th>
<th>cdsStart</th>
<th>cdsEnd</th>
<th>exonCount</th>
<th>name2</th>
</tr>
</thead>
<tbody>
<tr>
<td>889</td>
<td>NM_001003689</td>
<td>chr22</td>
<td>+</td>
<td>39931258</td>
<td>39957220</td>
<td>39931312</td>
<td>39953547</td>
<td>18</td>
<td>L3MBTL2</td>
</tr>
<tr>
<td>98</td>
<td>NM_005503</td>
<td>chr15</td>
<td>+</td>
<td>27001144</td>
<td>27197806</td>
<td>27133379</td>
<td>27196628</td>
<td>14</td>
<td>APBA2</td>
</tr>
<tr>
<td>795</td>
<td>NM_004672</td>
<td>chr1</td>
<td>-</td>
<td>27554256</td>
<td>27565924</td>
<td>27554468</td>
<td>27565675</td>
<td>29</td>
<td>MAP3K6</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>cdsStartStat</th>
<th>cdsEndStat</th>
</tr>
</thead>
<tbody>
<tr>
<td>cmpl</td>
<td>cmpl</td>
</tr>
<tr>
<td>cmpl</td>
<td>cmpl</td>
</tr>
<tr>
<td>cmpl</td>
<td>cmpl</td>
</tr>
</tbody>
</table>
Building a Custom Track

We analyzed the Agilent 44K microarray data using a linear model. The results are contained in an object called ourResults:

```r
> summary(ourResults)

 UntreatedMeanLog  Beta  PValue
  Min.  :4.870  Min.  :-3.15530  Min.  :2.024e-09
  1st Qu.:6.907  1st Qu.:-0.19572  1st Qu.:8.142e-02
  Median :8.058  Median :-0.05431  Median :2.749e-01
  Mean  :8.742  Mean  :-0.04300  Mean  :3.511e-01
  3rd Qu.:9.982  3rd Qu.: 0.10075  3rd Qu.:5.823e-01
  Max.  :16.523  Max.  : 3.27672  Max.  :1.000e+00
```
Computing a Displayable Score

We are going to us the p-values to decide which genes to display, and we are going to use the coefficient (Beta) to compute a score that shows the amount of differential expression. The allowed scores for a custom track range from 0 to 1000. Since the true values of Beta range between $-3$ and $+3$ (more or less), we are going to multiply by 300 to get a useful score.

```r
> score <- 300 * ourResults[, "Beta"]
> score[score > 1000] <- 1000
> score[score < -1000] <- -1000
> score <- abs(score)
```
A Track Data Frame

Now we build a data frame that includes the information we need for a custom track in the desired order:

```r
> temp <- data.frame(agilent2refgene[, c("chrom", "+ "txStart", "txEnd", "name2")], score = score, 
+                     strand = agilent2refgene[, "strand"]
> temp[1:3, 1:5]

  chrom  txStart  txEnd  name2  score
NM_001003689 chr22  39931258 39957220 L3MBTL2   96.902254
NM_005503 chr15  27001144 27197806 APBA2       74.415391
NM_004672 chr1   27554256 27565924 MAP3K6      2.281971
```
Significant Overexpressed Genes

We built this data frame for all genes; now we are going to select the ones that are significant (p-value < 0.02) and are overexpressed in response to the treatment (β > 0). We further restrict to those genes that we are able to map to the genome.

```r
> trackInfo <- temp[!is.na(temp[, "chrom"])] & ourResults[, + "PValue"] < 0.02 & ourResults[, "Beta"] > + 0, ]
```

We also have to create a header line that tells the browser to make use of the scores.

```r
> trackheader <- paste("track name=upNormal", + "description=\"Increased in Normal Cells\"", + "useScore=1 color=0,60,120")
```
Writing the Track Info to a File

We can now write the header line followed by the track data:

```r
> write(trackheader, file = "upNormalRNA.tsv", 
+       append = FALSE)
> write.table(trackInfo, file = "upNormalRNA.tsv", 
+       append = TRUE, quote = FALSE, sep = "\t", 
+       row.names = FALSE, col.names = FALSE)
```
Finally, we do the same thing for the genes that are underexpressed.

```r
> trackInfo <- temp[!is.na(temp[, "chrom"]) & ourResults[, "PValue"] < 0.02 & ourResults[, "Beta"] < 0, ]

> trackheader <- paste("track name=downNormal",
+ "description=""Decreased in Normal Cells"",
+ "useScore=1 color=100,50,0")

> write(trackheader, file = "dnNormalRNA.tsv", append = FALSE)
> write.table(trackInfo, file = "dnNormalRNA.tsv", append = TRUE, quote = FALSE, sep = "\t",
+ row.names = FALSE, col.names = FALSE)
```
Viewing Your Custom Track

Now we can return to the genome browser and look at our custom tracks. Unfortunately, their web page only lets you attach one at a time unless you can make them available from a web site:
Agilent Differential Expression Study

This page is a front end to the UCSC "Golden Path" Human Genome Browser. When you press the "Submit" button, it will upload custom tracks to the browser to represent the locations that are differentially expressed in normal cells. There are two annotation tracks available:

http://bioinformatics.mdanderson.org/MicroarrayCourse/upNormal
http://bioinformatics.mdanderson.org/MicroarrayCourse/dnNormal
Displaying Our Tracks
Searching for a Gene
Searching for a Gene

**Known Genes**

TP53 (NM_000546) at chr17:7512465-7531642 - tumor protein p53
TP53 (DQ186649) at chr17:7512447-7531524 - Del133 p53 gamma isoform.
TP53 (DQ186649) at chr17:7512447-7531524 - Del133 p53 gamma isoform.
TP53 (DQ186649) at chr17:7512447-7531524 - Del133 p53 beta isoform.
TP53 (DQ186649) at chr17:7512447-7531524 - Del133 p53 beta isoform.
C2orf10 (NM_014477) at chr20:43435935-43440371 - TP53-target gene 5 protein
TP53AP1 (NM_007233) at chr7:86792477-86812767 - TP53 activated protein 1
TP53RK (BC019621) at chr20:44747581-44751486 - TP53 regulating kinase.
PERP (NM_022121) at chr6:138453619-138470280 - PERP, TP53 apoptosis effector
PRM1 (NM_019845) at chr1:154042098-154043586 - replimo, TP53 dependent G2 arrest mediator
RMM2B (NM_015713) at chr8:103285907-103320522 - ribonucleotide reductase M2 B (TP53 inducible
TP53 (BC003596) at chr17:7512465-7531511 - Del133 p53 isoform.
VRK1 (BC103761) at chr14:9633459-96417696 - vaccinia related kinase 1
TP53INP1 (NM_032385) at chr8:96007337-96030767 - tumor protein p53 inducible nuclear protein
TP53INP1 (AF209114) at chr8:96007337-96030767 - tumor protein p53 inducible nuclear protein 1
UBE2L6 (BC032491) at chr11:57075712-57091756 - ubiquitin-conjugating enzyme E2L 6
PPIL13B (NM_015316) at chr14:103269842-103383555 - protein phosphatase 1, regulatory (inhibi
TP53BP2 (BC058918) at chr1:222034413-222100255 - tumor protein p53 binding protein, 2
ENCL (NM_003633) at chr5:73958991-73972273 - ectodermal-neural cortex (with BTB-like domain)
PEM1D (BC042418) at chr17:56032412-56069616 - protein phosphatase 1D magnesium-dependent, del
TP53BP2 (NM_00136685) at chr1:222034413-222100255 - tumor protein p53 binding protein, 2 iso
GNYL3 (NM_206825) at chr3:52694976-52703548 - guanine nucleotide binding protein-like 3
ING5 (NM_003229) at chr2:242290129-24317563 - inhibitor of growth family, member 5
EDG1 (NM_030815) at chr20:29996420-30003544 - p53 and DNA damage-regulated protein
ING4 (NM_016162) at chr12:6629707-6642565 - inhibitor of growth family, member 4 isoform 1
Searching for a Gene
Searching for a Gene

UCSC Genome Browser on Human Mar. 2006 Assembly

position/search chr17:7,202,421-7,841,687

Click on a feature for details.
Click on base position to
zoom in around cursor. Click

move start
move end

© Copyright 2004–2007 Kevin R. Coombes and Keith A. Baggerly
GS01 0163: Analysis of Microarray Data