GS01 0163
Analysis of Microarray Data

Keith Baggerly and Kevin Coombes
Section of Bioinformatics
Department of Biostatistics and Applied Mathematics
UT M. D. Anderson Cancer Center
kabagg@mdanderson.org
kcoombes@mdanderson.org

28 October 2004

Lecture 17: Clustering Microarray Data I

- How Close are Things? Defining Distance
- How Do We Join Things? Defining Linkage
- Graphical Displays: The Tree
- Which Clusters are “Real”? Cutting the Tree
- Should we Use All the Data? Using Filtered Lists
- Clustering Heatmaps – 2d Clustering (Plaid Models?)

In the beginning was the data...

Say we start with 31 microarrays of melanoma samples (3613 genes/array), all measured against a common reference. The data has been processed (background corrected, normalized, log transformed, etc) and we have reached “the matrix” (welcome, Neo).

How can we cluster the samples in order to reveal potential subtypes of interest?

In the beginning was the data...

Say we start with 31 microarrays of melanoma samples (3613 genes/array), all measured against a common reference. The data has been processed (background corrected, normalized, log transformed, etc) and we have reached “the matrix” (welcome, Neo).

How can we cluster the samples in order to reveal potential subtypes of interest?

At present, this is not a precisely specified problem.

Getting the data

Go to http://linus.nci.nih.gov/BRB-arrayTools.html

get Melanoma.zip

in data file, replace 3' with 3prime, 5' with 5prime, DNA's with DNAs

```
 melanomaData <-
read.table(''Bittner_expression.dat'',
header = TRUE, sep = ''\t'');

 dataMatrix <-
as.matrix(melanomaData[,4:34]);
```

What does a cluster show?

Ideally, hierarchical clustering tells us that some samples form a more coherent set than the data as a whole, where “more coherent” is generally taken to mean that the samples are closer together.

What does a cluster show?

Ideally, hierarchical clustering tells us that some samples form a more coherent set than the data as a whole, where “more coherent” is generally taken to mean that the samples are closer together.

So, how do we define “closer”?

This requires the specification of some type of distance, or more generally “dissimilarity”, matrix.

Some distances

All distances that we will work with for now are calculated between one sample (vector) and another.

Euclidean distance: `sqrt(sum((x - y)^2))`

```
dEuclid <- dist(t(dataMatrix));
```

This is the standard thing that most people come up with. Are there others?

Some distances

All distances that we will work with for now are calculated between one sample (vector) and another.

Euclidean distance: `sqrt(sum((x - y)^2))`

```
dEuclid <- dist(t(dataMatrix));
```

This is the standard thing that most people come up with. Are there others?

The “method” argument to dist will accept “euclidean”, “maximum”, “manhattan”, “canberra”, “binary”, and “minkowski”

Other definitions

Maximum: `abs(max(x-y))`

Manhattan: `sum(abs(x-y))`

Canberra: `sum(abs(x-y)/abs(x+y))`
Canberra: `sum(abs(x-y)/(abs(x)+abs(y)))`

Binary: `sum(xor(x != 0, y != 0))/`
`sum(x != 0 | y != 0)`

Minkowski: `sum(abs(x-y)^p)^(1/p)`

Cool!

Other definitions

Maximum: `abs(max(x-y))`

Manhattan: `sum(abs(x-y))`

Canberra: `sum(abs(x-y)/abs(x+y))`
Canberra: `sum(abs(x-y)/(abs(x)+abs(y)))`

Binary: `sum(xor(x != 0, y != 0))/`
`sum(x != 0 | y != 0)`

Minkowski: `sum(abs(x-y)^p)^(1/p)`

Cool! We won't be using any of these.

More realistic

Correlation! `(1 - cor(x,y))/2`

More realistic

Correlation! `(1 - cor(x,y))/2`

Absolute Correlation? `(1 - abs(cor(x,y)))`

More realistic

Correlation! `(1 - cor(x,y))/2`

Absolute Correlation? `(1 - abs(cor(x,y)))`

Rank Correlation? `(1 - cor(x,y,method = ''spearman''))/2`

More realistic

Correlation! `(1 - cor(x,y))/2`

Absolute Correlation? `(1 - abs(cor(x,y)))`

Rank Correlation? `(1 - cor(x,y,method = ''spearman''))/2`

For now, let's say that we're going to use Correlation as our distance measure.

Occasionally, we center the rows. I haven't.

And we're off!

```
dcorr <- (1 - cor(dataMatrix))/2
```

Looking at all of the pairwise distances between the 31 samples, we find that the smallest pairwise distance, 0.1230, is between samples 4 and 9.

Great! We cluster these two samples.

And we're off!

```
dcorr <- (1 - cor(dataMatrix))/2
```

Looking at all of the pairwise distances between the 31 samples, we find that the smallest pairwise distance, 0.1230, is between samples 4 and 9.

Great! We cluster these two samples.

What is the distance from the remaining samples to the cluster?

Redefining Distance: Linkage

So far, all of the distances that we have assembled work on pairs of vectors. We'd like to extend this idea so that we can work on pairs of clusters instead.

In doing this, we can either work with a new "central vector" that summarizes the cluster, or with specific elements of the cluster (we'll focus on the latter).

For example, the distances between sample 1 and samples 4 and 9 respectively are 0.1478 and 0.1548.

Some linkage values:

d(1,(4,9)) = min(0.1478,0.1548) = 0.1478 (single linkage)

Some linkage values:

d(1,(4,9)) = min(0.1478,0.1548) = 0.1478 (single linkage)

d(1,(4,9)) = max(0.1478,0.1548) = 0.1548 (complete linkage)

Some linkage values:

d(1,(4,9)) = min(0.1478,0.1548) = 0.1478 (single linkage)

d(1,(4,9)) = max(0.1478,0.1548) = 0.1548 (complete linkage)

d(1,(4,9)) = mean(0.1478,0.1548) = 0.1513 (average linkage)

```
hclust(d, method = ‘‘complete’’);
```

Other linkage methods:

“ward”

“median”

“centroid”

“mcquitty”(?)

Assembling a cluster

Define the distance matrix

```
dcorr <- (1 - cor(dataMatrix))/2
```

Define the linkages

```
dcorrTree <-
hclust(as.dist(dcorr),method=''complete'')
```

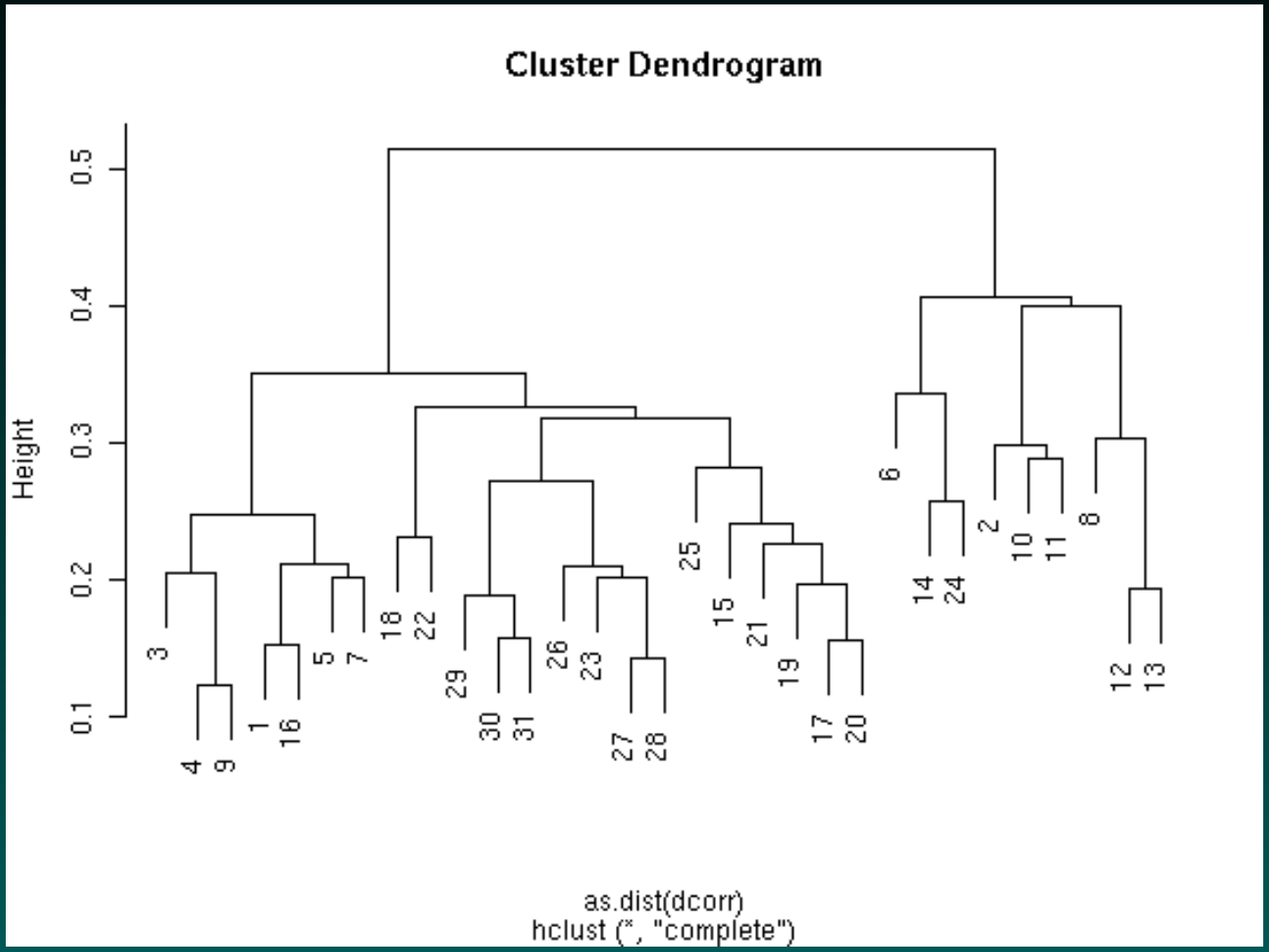
Set the labels (if not inherited)

```
dcorrTree$labels <- as.character(1:31)
```

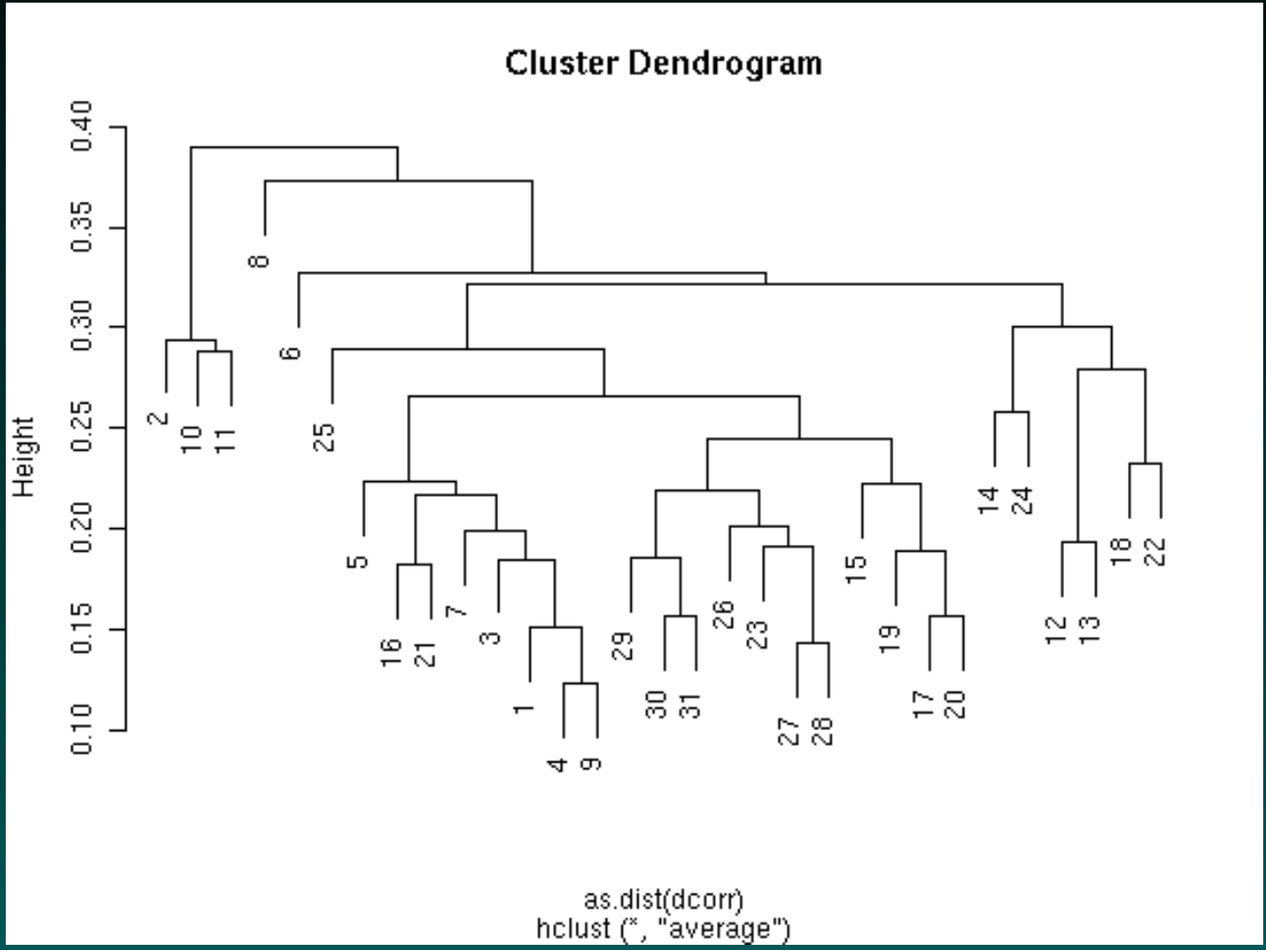
Plot the tree

```
plot(dcorrTree)
```

Cluster (Take 1 - Complete)

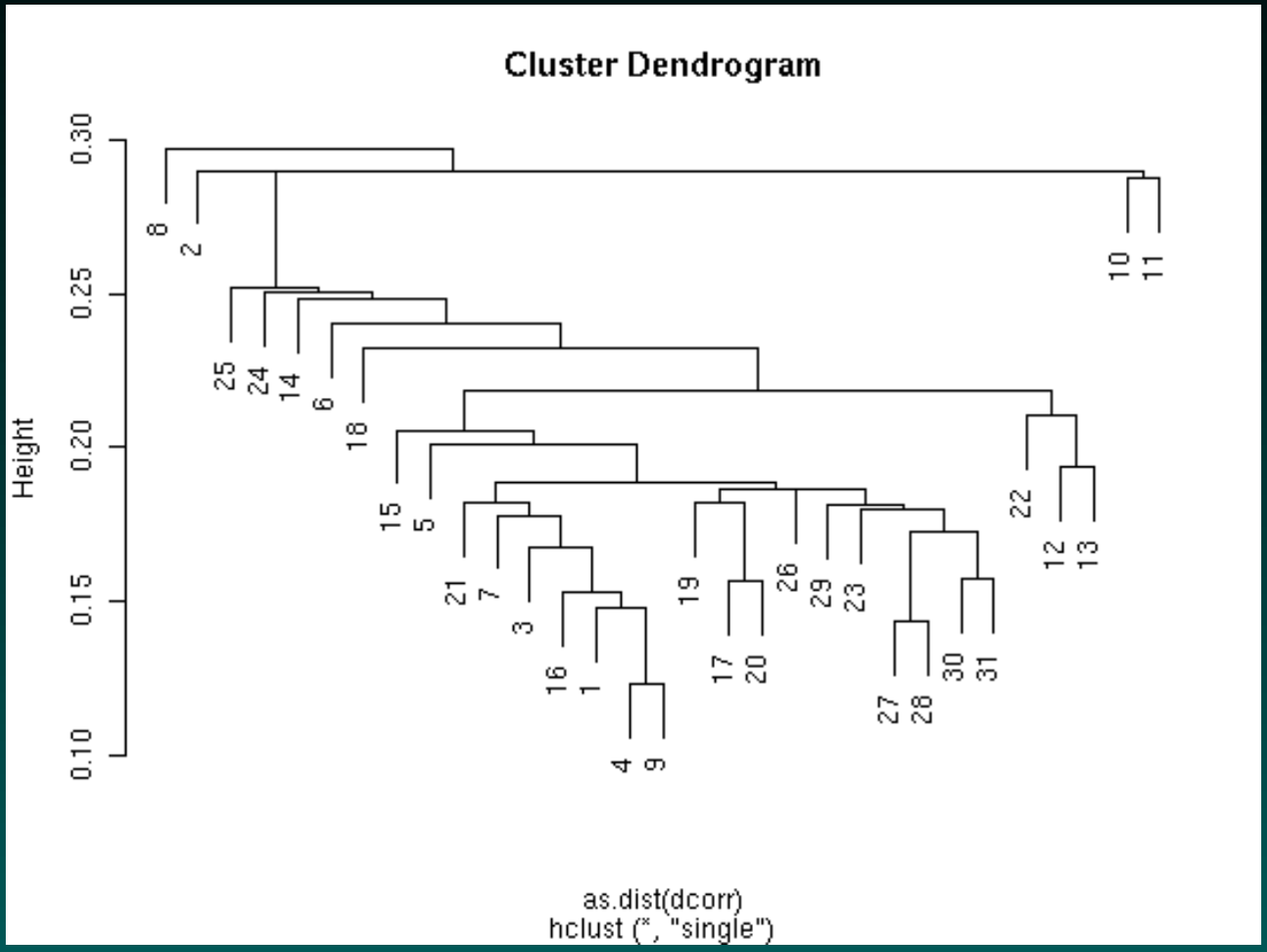


Cluster (Take 2 - Average)




```
 dcorrTree <-
hclust(as.dist(dcorr),method=''average'')
```

Cluster (Take 3 - Single)




```
dcorrTree <-
hclust(as.dist(dcorr),method=''single'')
```

What types of clusters are produced?

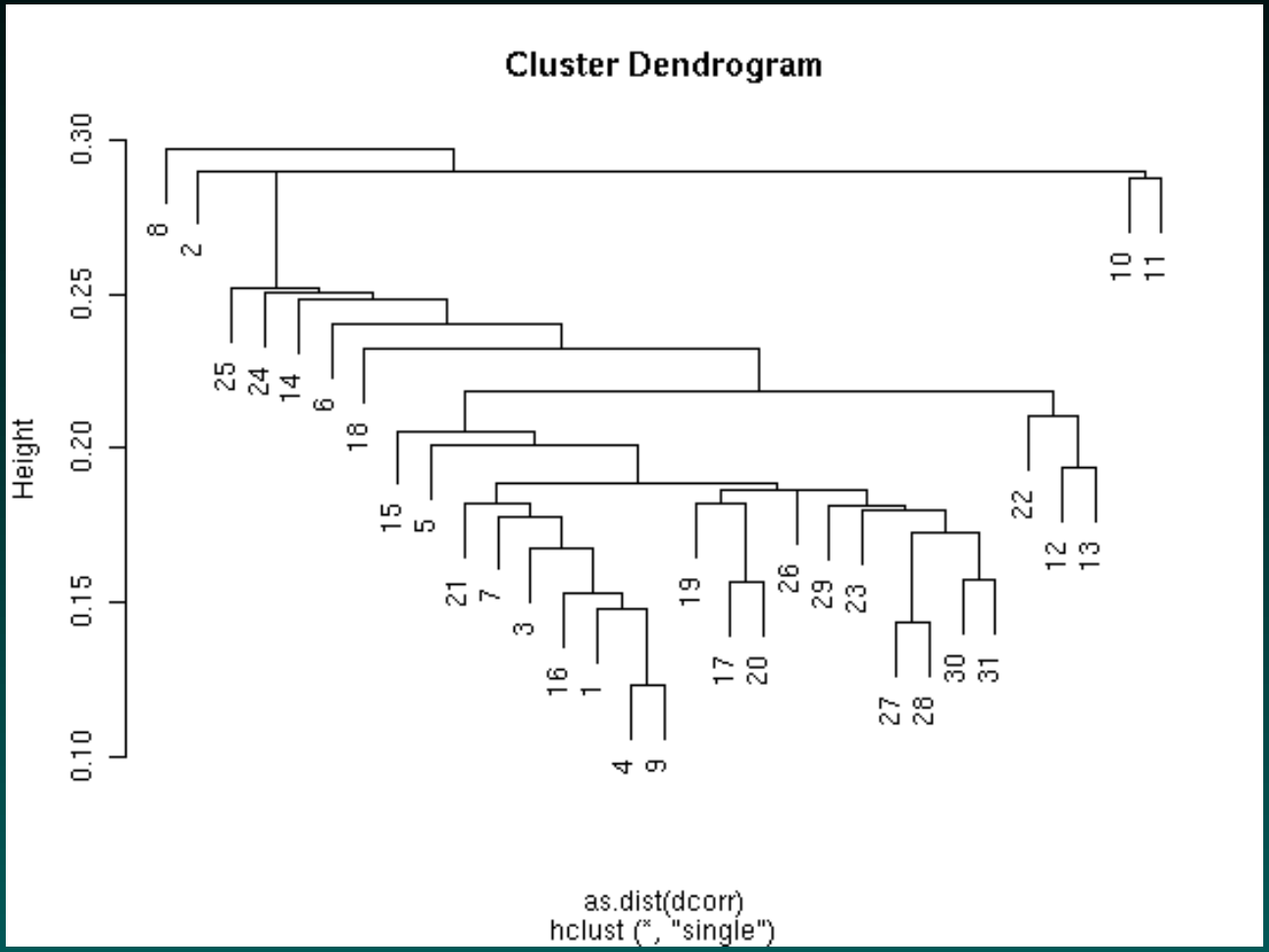
Single linkage can lead to a lot of singleton clusters, and to clusters that look “stringlike” in high dimensions.

Complete linkage, by contrast, tends to find more compact (“spherical”) clusters. This property is also shared with Ward’s method. (I like this one.)

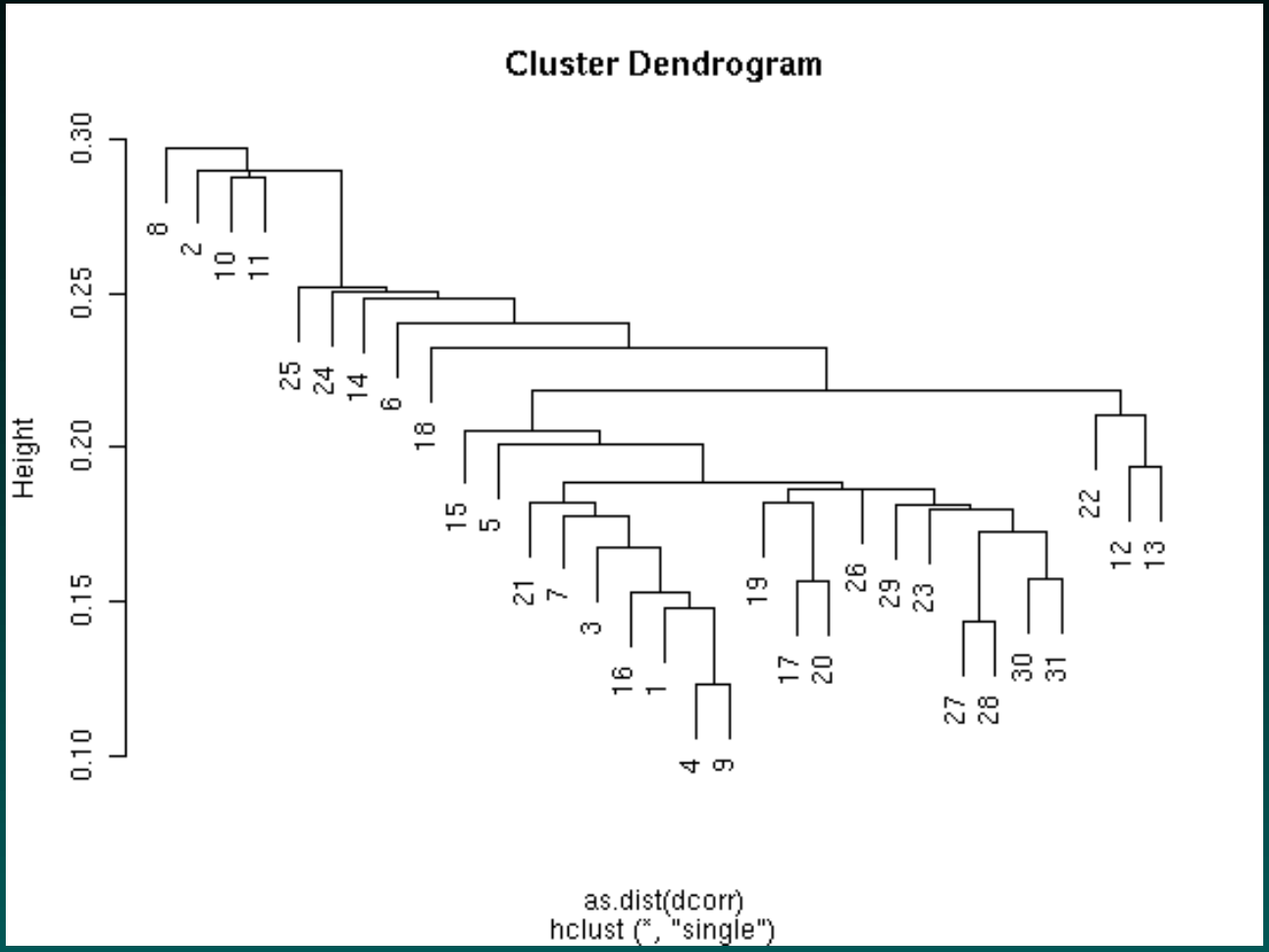
Average linkage (and most others) produce clusters that are intermediate between those produced by single and complete linkage.

Warning: Some of the tree orderings are meaningless!

Visual Games 1 (Single)

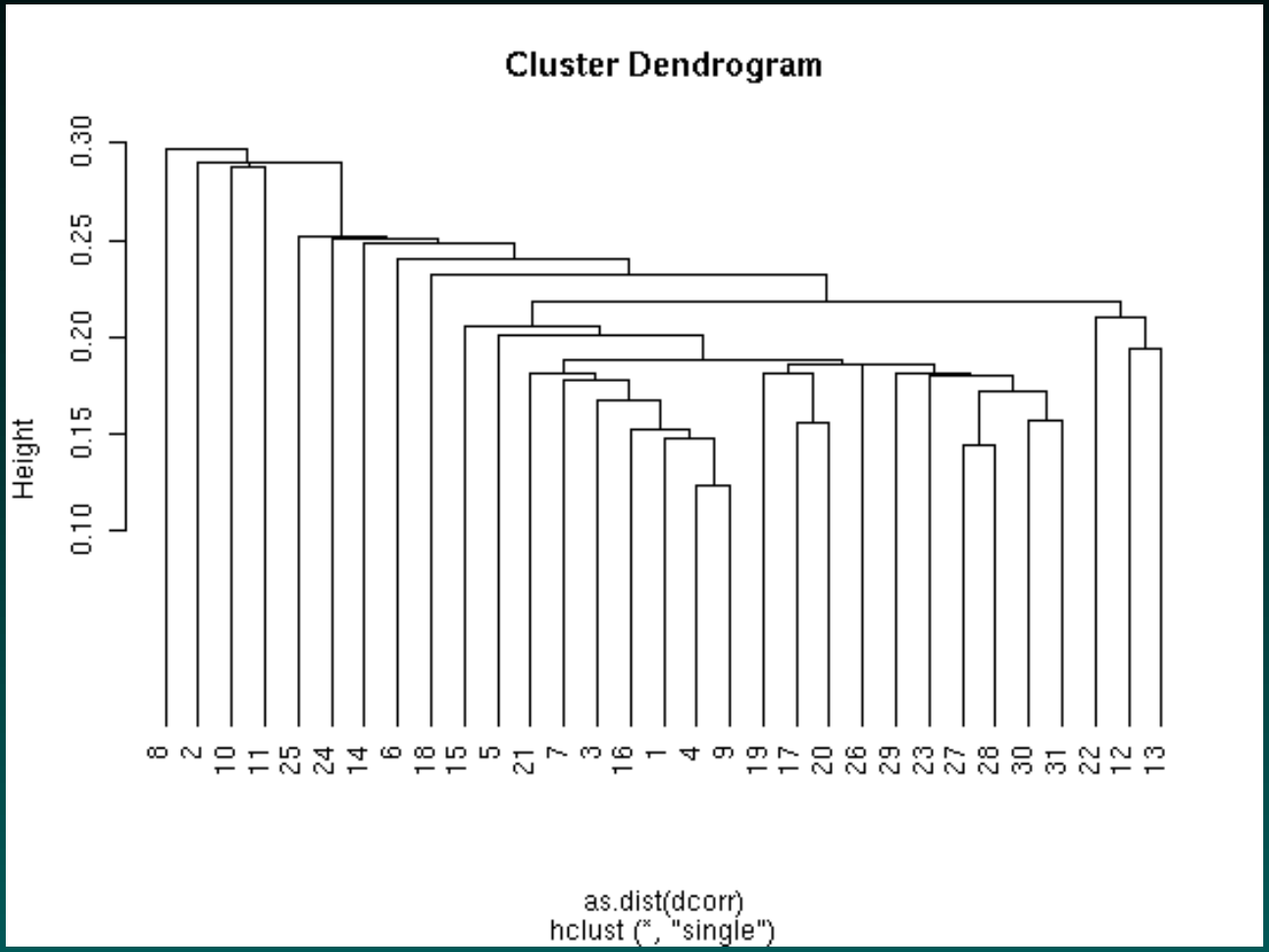


Visual Games 1 (Single)




```
 dcorrTree$order <-
dcorrTree$order[c(1:2,30:31,3:29)];
```

Visual Games 2 (Single)




```
plot(dcorrTree, hang = -1);
```

What's Stable?

We can flip branches around without affecting the underlying structure of the data, or changing the meaning of the clustering.

What things are left unchanged by such flips?

Say we flip a branch at height h_{flip}.

Membership of the sub-branches does not change, but the order can change across the boundary.

How do I define a "cluster"?

Clusters should be Stable

If we cut the dendrogram at height h, then the sub-branches of each cut branch define clusters.

Within a cluster, everything is closer than h to the rest.

By varying the cut height, we can produce an arbitrary number of clusters.

Defining Cluster Validity

What is the right level to cut things at?

Defining Cluster Validity

What is the right level to cut things at?

Testing this requires “perturbing” the data.

Defining Cluster Validity

What is the right level to cut things at?

Testing this requires “perturbing” the data.

A cluster is comprised of pairs of items that are grouped together. If we repeatedly perturb the data, and the pairs still cluster together, this is a good sign that the cluster is “stable”.

Samples that cluster in other groups are more questionable.

Disturbing the Universe

The simplest way to perturb the data here is to "bootstrap" the individual genes, or rows of the data matrix.

```
 tempData <-
dataMatrix[sample(3613,replace=T),];

 tempCorr <- (1 - cor(tempData))/2;

 tempCorrTree <-
hclust(as.dist(tempCorr), method =
''average'');
```

Checking the Pairings

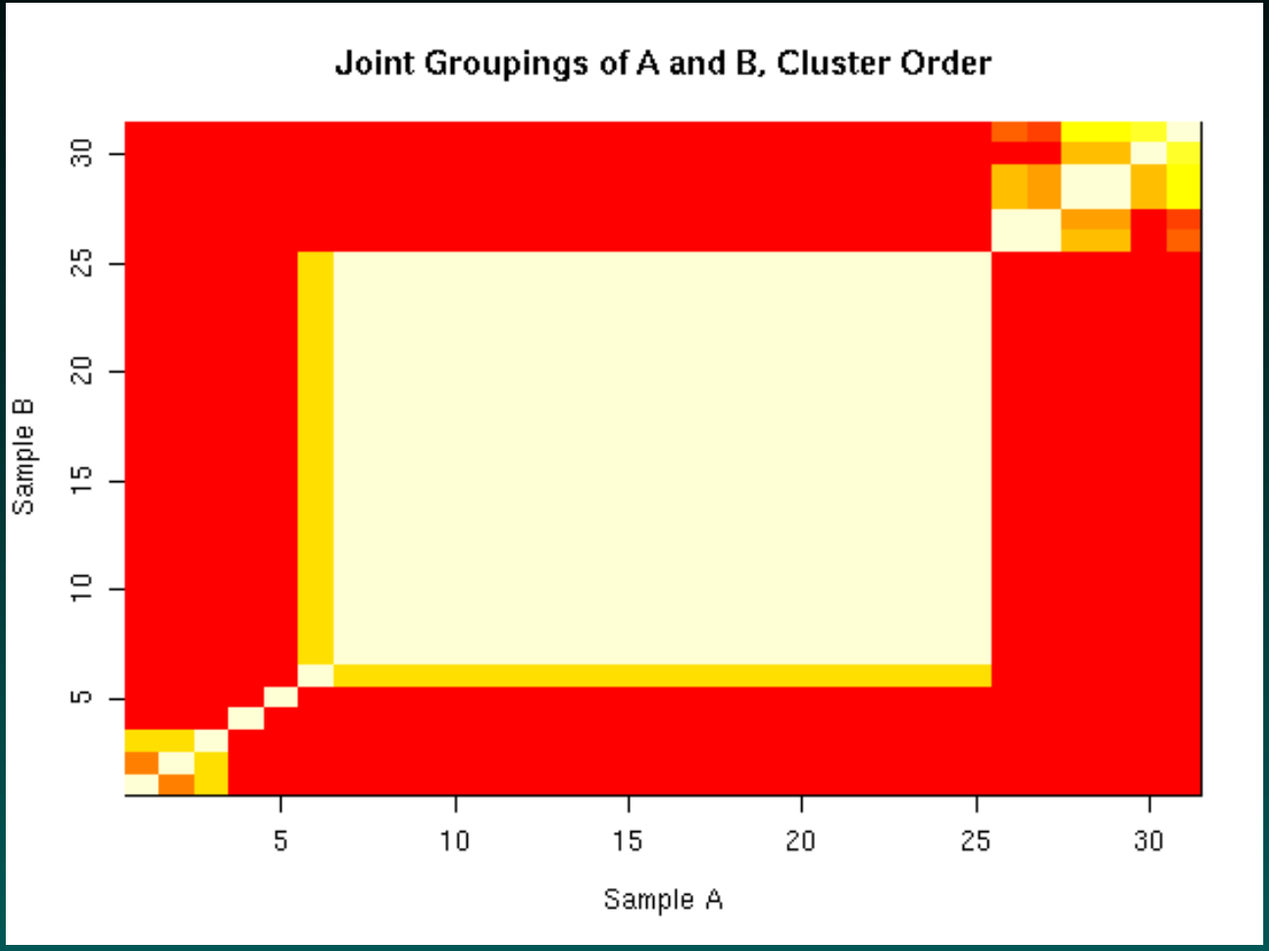
Using 7 groups, 31 samples, and an outer loop of 1000 bootstrap samples:

```
tempCut <- cutree(tempCorrTree, k = 7);
tempMatch <- matrix(0, nrow=31, ncol=31);

for(i2 in 1:7){
 tempMatch[tempCut==i2,tempCut==i2] = 1;
}

bootMatch <- bootMatch + tempMatch;
```

The Results Here




```
image(1:31,1:31,bootMatch[dcorrTree$order,
dcorrTree$order], ...);
```

Some Caveats

We need to specify the number of clusters to use; simply working with the distance matrices will not work, due to the problem of going from distance to linkage.

The picture is much more interpretable if the rows and columns of the matching matrix are reordered to match the ordering supplied by the clustering.

This ordering can likewise be used to order the data matrix itself before display.

Other Perturbations

Instead of resampling the genes, we can “add noise” to the data, say from a Gaussian (normal) distribution.

The scale of noise to use is not necessarily obvious.

McShane et al (2002) recommend computing all of the gene-wise standard deviations, and using the median of these values to define σ.

I personally prefer bootstrapping.

Do we need ALL the Genes?

Do we need ALL the Genes?

Maybe most of them are noise.

Do we need ALL the Genes?

Maybe most of them are noise.

Maybe we just want to work with a well-chosen subset.

Do we need ALL the Genes?

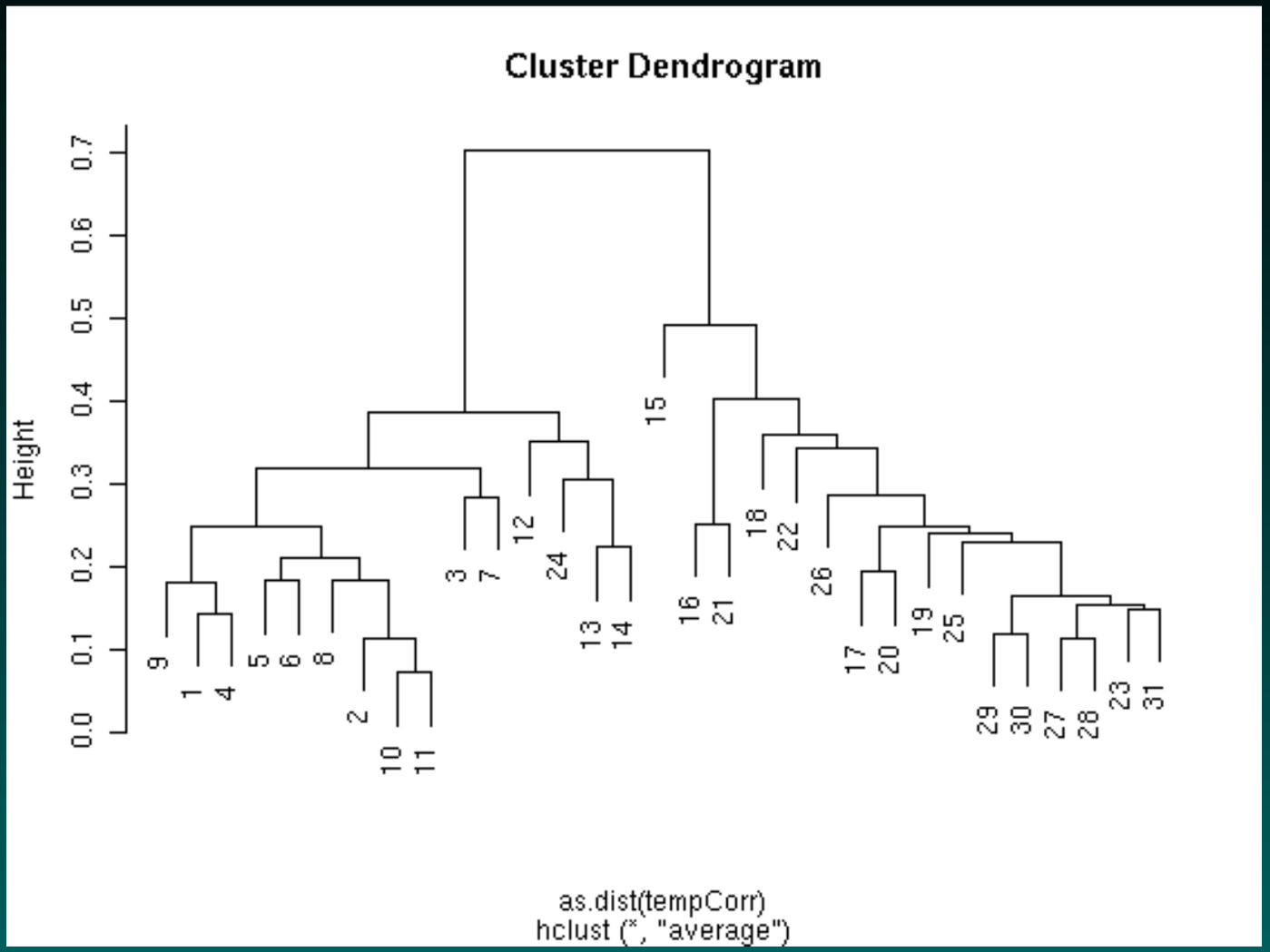
Maybe most of them are noise.

Maybe we just want to work with a well-chosen subset.

An example – let's say we center the data rows, so as to focus primarily on changes.

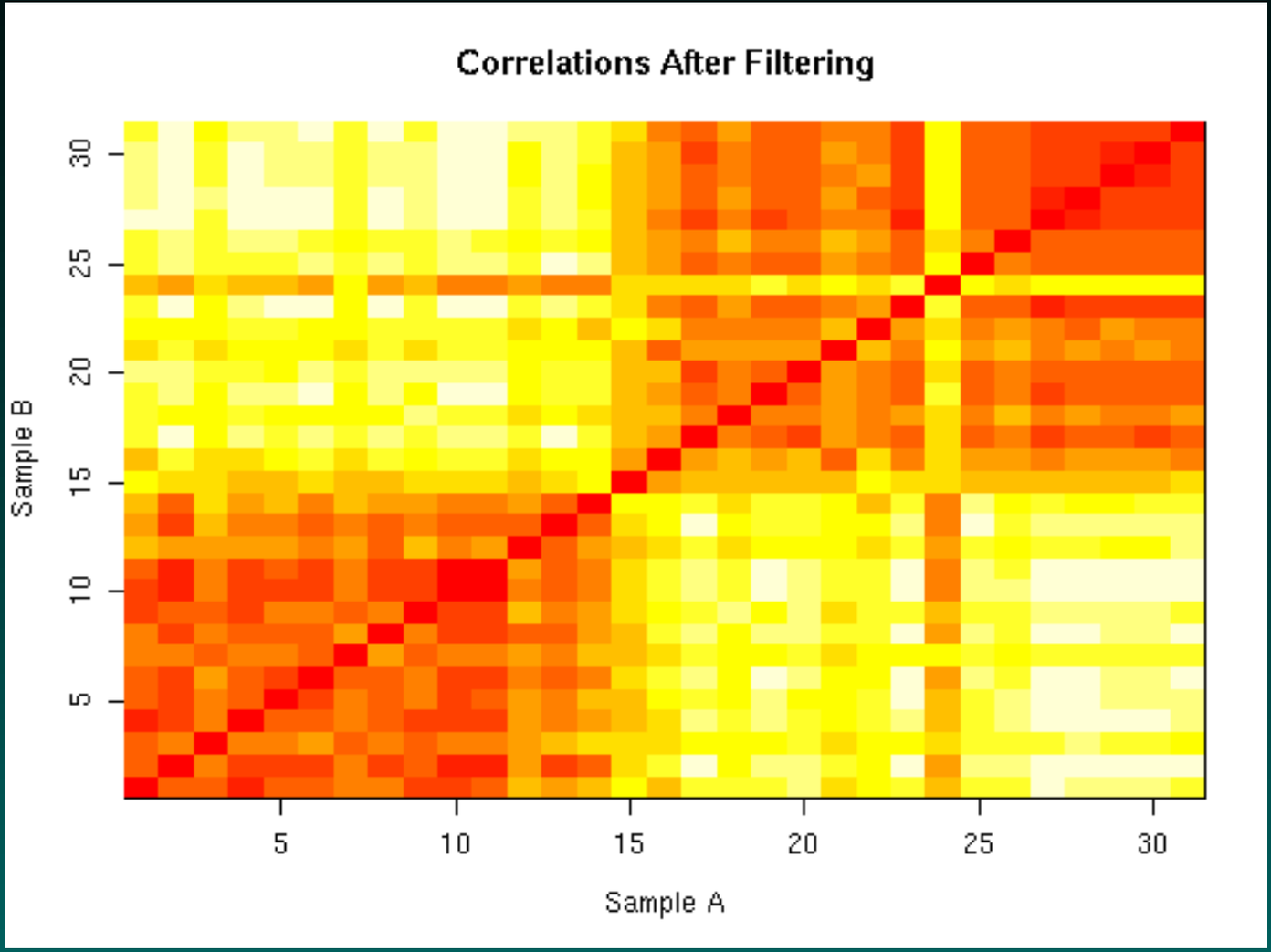
Then, let's just focus on the genes that show a big difference between the first 15 columns and the last 16 (abs t-value $>$ 4).

The Cluster



Amazing!

The Correlation Matrix



What are the odds?!

Filtering Sans Sarcasm

The above results are not surprising.

Filters should not be related to a specific contrast if an overall view is desired.

More natural filters exist:

Filtering Sans Sarcasm

The above results are not surprising.

Filters should not be related to a specific contrast if an overall view is desired.

More natural filters exist:

total variation

Filtering Sans Sarcasm

The above results are not surprising.

Filters should not be related to a specific contrast if an overall view is desired.

More natural filters exist:

total variation

all genes on a given chromosome

Filtering Sans Sarcasm

The above results are not surprising.

Filters should not be related to a specific contrast if an overall view is desired.

More natural filters exist:

total variation

all genes on a given chromosome

all genes in a given ontology category

Filtering and Size

Filtering does serve at least on practical purpose – it reduces the number of genes a lot. This is important because we may want to cluster the genes as well as the samples, and clustering thousands of things may make R complain (not to mention validation).

If we can cluster on both axes, we can sort the genes as well as the samples for display purposes.

These are the really cool pictures.