Machine Learning Methods for Gene Expression Data

Day 1

Dennis Wylie, UT Bioinformatics Consulting Group

May 22, 2016

Outline

What is Machine Learning?

Perhaps better thought of as "algorithms for learning."

Such algorithms may also be referred to as modeling strategies

M

which, when provided training data

D_{train}

from some particular experiment, "learn" parameters

$\boldsymbol{\theta}$

such that the pair

$(M, \boldsymbol{\theta})$

can be used to predict likely observations

D_{other}

from similar experiments.

Taxonomy of Machine Learning

Often subdivided into three categories:

Supervised $D = (\mathbf{x}, y)$ consists of inputs $\mathbf{x}$ and outcomes y, with focus on predicting y given $\mathbf{x}$.

Unsupervised $D = \mathbf{x}$ with no particular outcome identified; focus instead on identifying common patterns in $\mathbf{x}$ alone.

Reinforcement $D = (a, \mathbf{x}, y)$ in which the outcome y is also influenced by actions a over which the modeler has control and the focus is on identifying those a most likely to generate desirable y.

Reinforcement learning is not currently very highly studied in the context of gene expression data.

Notation

- x_{ig} normalized expression value for gene g in sample i
- $\mathbf{x}$ p-dimensional vector of expression values of all p genes (e.g., in one sample)
- $\underline{x}$ n-dimensional vector of expression values for all n samples (e.g., for one gene)
- $\underline{\mathbf{X}}$ matrix of any dimensionality (e.g., $n \times p$ matrix of gene expression values x_{ig})

Probabilities

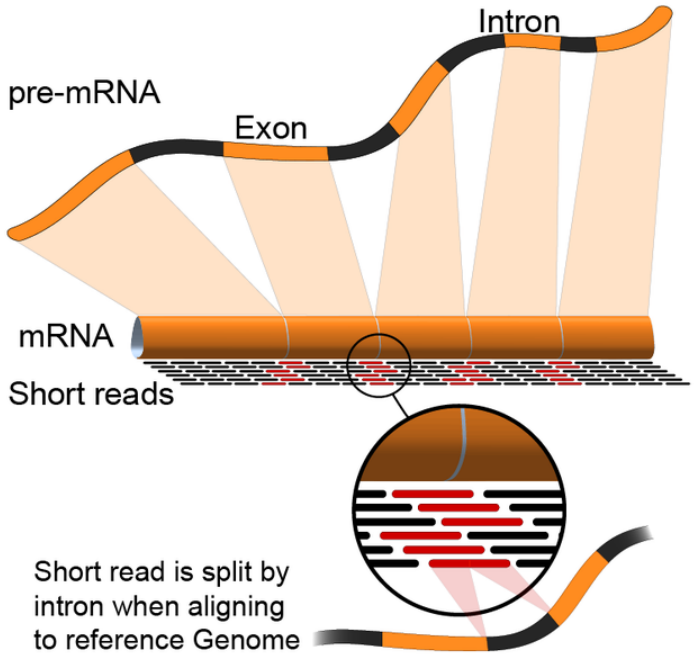
Most work in machine learning can be described probabilistically using random variables $\mathbf{X}$ and/or Y and defining the predictions made by model $(M, \boldsymbol{\theta})$ through the distributions

$$\begin{aligned} \mathbb{P}(\mathbf{X} \mid M, \boldsymbol{\theta}) &\quad \text{(Unsupervised)} \\ \mathbb{P}(\mathbf{X}, Y \mid M, \boldsymbol{\theta}) &\quad \text{(Supervised, Generative)} \\ \mathbb{P}(Y \mid \mathbf{X}, M, \boldsymbol{\theta}) &\quad \text{(Supervised, Discriminative)} \end{aligned}$$

Note that many supervised learning algorithms fit only the conditional probability $\mathbb{P}(Y \mid \mathbf{X}, M, \boldsymbol{\theta})$, thereby remaining agnostic about the distribution of $\mathbf{X}$.

This flexibility can come at a cost . . .

Types of Gene Expression Data: RNA-seq



- Most detailed picture of gene expression
- Can detect novel transcripts, alternative splicing, SNVs
- Analysis can be done at exon, transcript, or gene level

RNA-seq Set: Bottomly *et al.* (2011)

Data set obtained from ReCount (Frazee *et al.* (2011)).

Differential expression study of 21 mouse samples split across two inbred strains.

From the abstract:

> C57BL/6J (B6) and DBA/2J (D2) are two of the most commonly used inbred mouse strains in neuroscience research ...
>
> We show that by using stringent data processing requirements differential expression as determined by RNA-seq is concordant with both the Affymetrix and Illumina platforms in more instances than it is concordant with only a single platform, and that instances of discordance with respect to direction of fold change were rare.

RNA-seq Set: Patel *et al.* (2014): GSE57872

Data set obtained from Gene Expression Omnibus (GEO) using `GEOquery` (Davis & Meltzer (2007)).

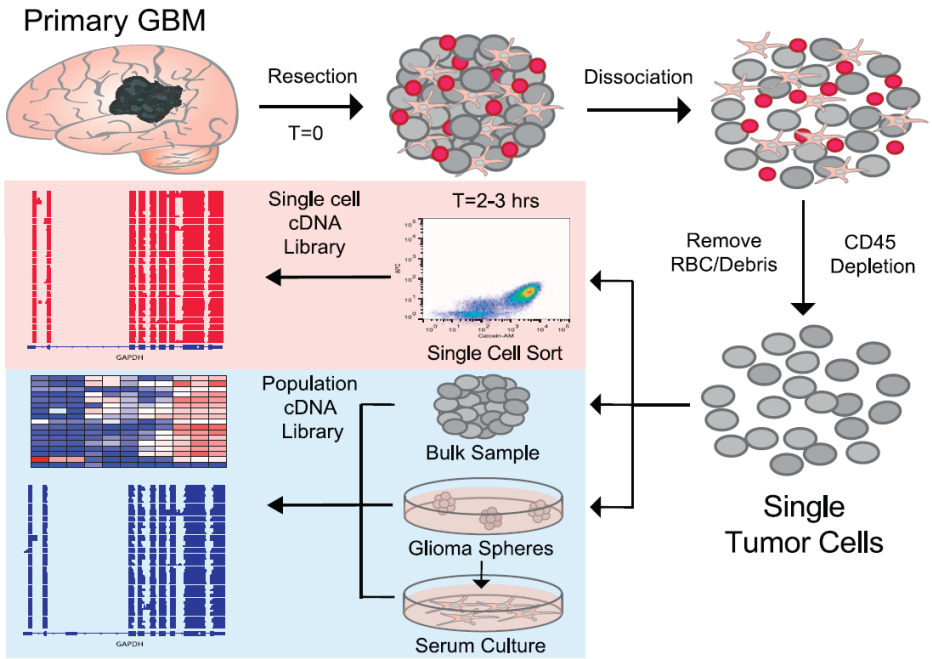
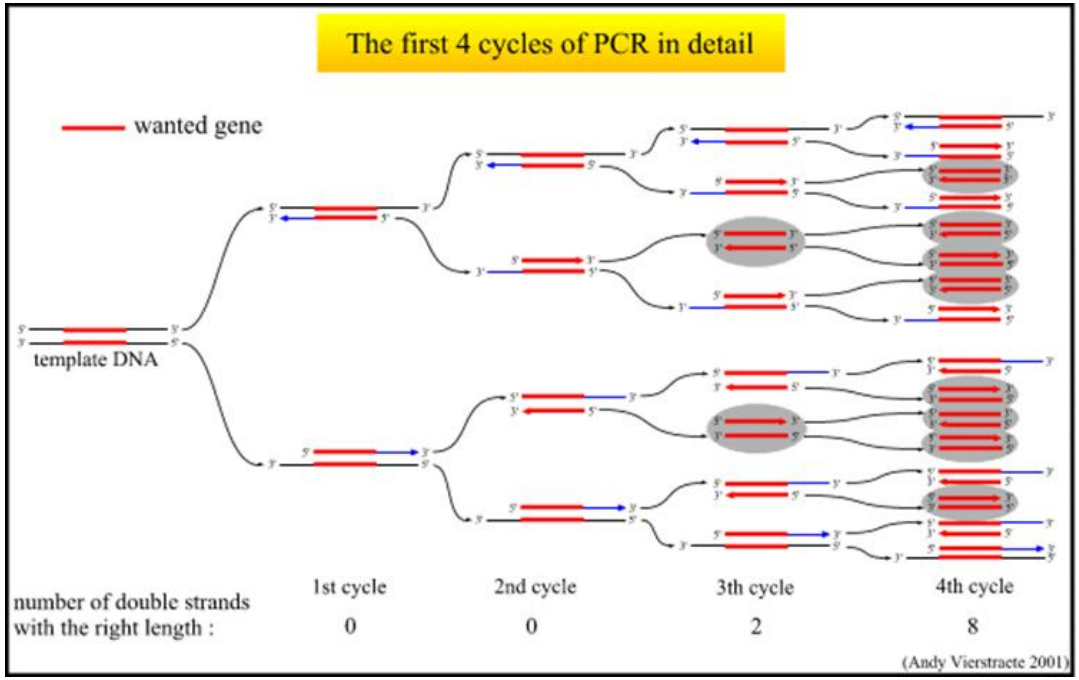


Fig. 1. Intratumoral glioblastoma heterogeneity quantified by single-cell RNA-seq. (A) Workflow depicts rapid dissociation and isolation of glioblastoma cells from primary tumors for generating single-cell and bulk qRNA-seq profiles and deriving glioblastoma culture models.

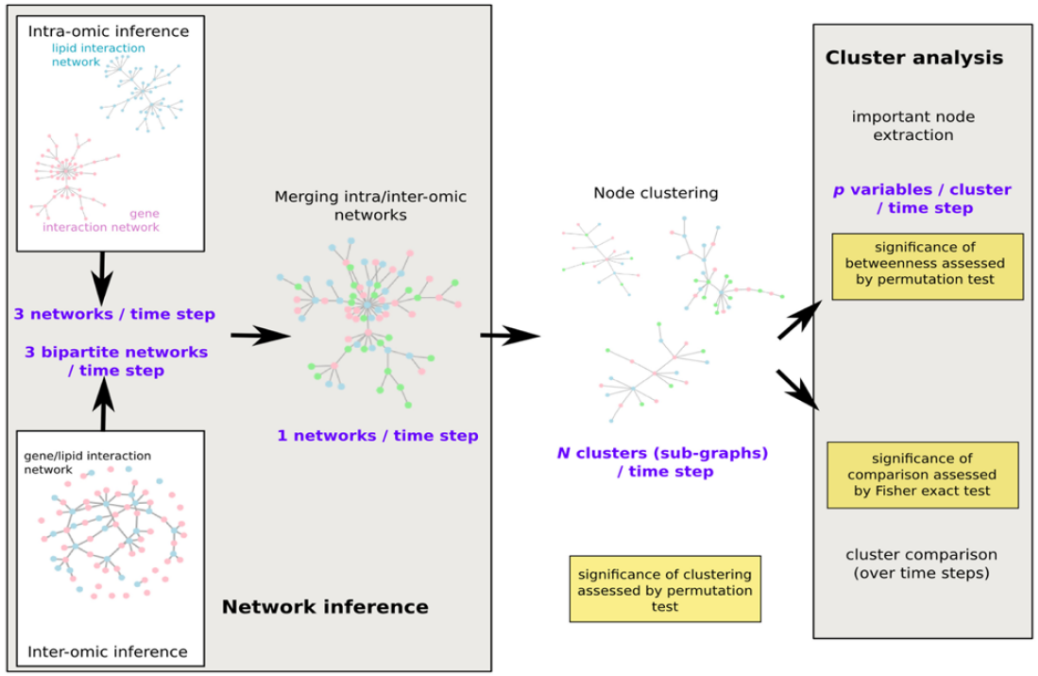
Types of Gene Expression Data: RT-qPCR



- Count number of cycles (Ct) required for fluorescence signal to surpass threshold
  - $\text{Ct} \propto 2^{-(\text{copy number})}$
- Analysis simpler than for RNA-seq
- Need primer pair for gene of interest
- May be cheaper/easier than RNA-seq for measurement of small number of genes

RT-qPCR Set: Montastier *et al.* (2015): GSE60946

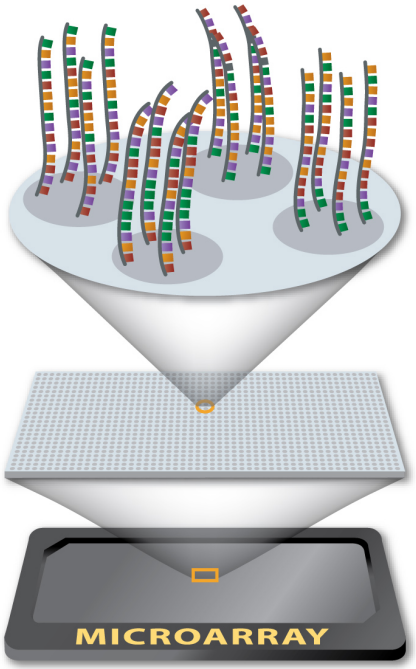
Obtained from GEO using `GEOquery` (Davis & Meltzer (2007)).



AT fatty acids and mRNA levels were quantified in 135 obese women at baseline, after an 8-week low calorie diet (LCD) and after 6 months of ad libitum weight maintenance diet (WMD) . . .

A 3 steps approach . . . consisted in inferring intra-omic networks with sparse partial correlations and inter-omic networks with regularized canonical correlation analysis and finally combining the obtained omic-specific network in a single global model.

Types of Gene Expression Data: Microarray



- Analysis simpler than for RNA-seq
- May be cheaper than RNA-seq
- Throughput intermediate between RT-qPCR and RNA-seq
- Lower sensitivity, dynamic range than RNA-seq

Microarray Set: Hess *et al.* (2006)

Data set downloaded from
`http://bioinformatics.mdanderson.org/pubdata.html`.

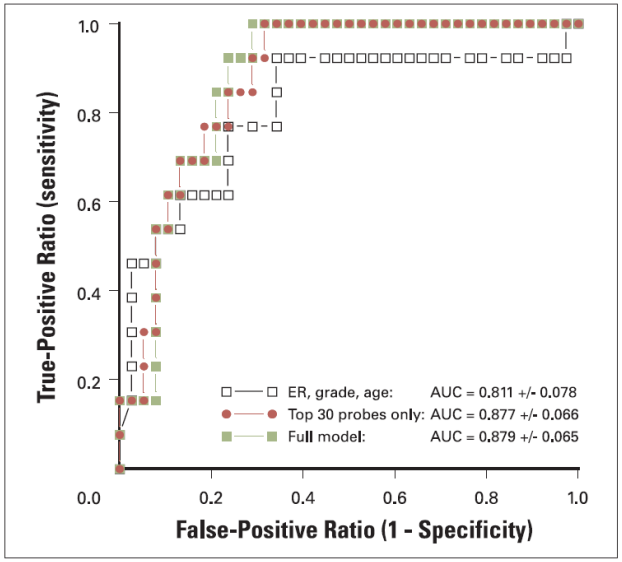


Fig 3. Receiver operating characteristic curves of three distinct pathologic complete response prediction models. The performance of the Diagonal Linear Discriminant Analysis–30 predictor and a predictor based on clinical variables and a combined clinical + pharmacogenomic prediction model are shown in the validation set (n = 51). ER, estrogen receptor; AUC, area under the curve.

We developed a multigene predictor of pathologic complete response (pCR) to preoperative weekly paclitaxel and fluorouracil-doxorubicin-cyclophosphamide (T/FAC) chemotherapy and assessed its predictive accuracy on independent cases.

Loading Tabular Data

For this class, data provided in tab-delimited text files with header in first column and index in first row.

```
# R:

df = read.table(file, header=TRUE, row.names=1, sep='\t')
```

```
# Python:

import pandas
df = pandas.read_csv(file, header=0, index_col=0, sep='\t')
```

I will use the "=" assignment operator in R in order to minimize differences between R and Python.

The pandas library (McKinney (2012)) for Python provides a DataFrame similar (and in some ways superior) to R's data.frame.

Accessing Data — Individual Elements

Assuming column names are capital letters and row names lower-case:

```
# R:

df[1, 2]
df['a', 'B']
df[1, 'B']
df$B[1]
```

```
# Python:

df.ix[0, 1]
df.ix['a', 'B']
df.ix[0, 'B']
df['B'][0]
df.B[0]
```

Accessing Data — Whole Rows or Columns

R:

```
df[1, ]              ## returns row as data.frame
df['a', ]            ##  same
df[ , 2, drop=FALSE] ## returns column as data.frame
df[ , 2]             ## returns column as vector
df[ , 'B']           ##  same
df$B                 ##  same
```

Python:

```
df.ix[0]             ## returns row as pandas.Series
df.ix['a']           ##  same
df.ix[ [0] ]         ## returns row as pandas.DataFrame
df[ df.columns[1] ]  ## returns column as pandas.Series
df['B']              ##  same
df.B                 ##  same
df[ ['B'] ]          ## returns column as pandas.DataFrame
```

Accessing Data — Subframes

In both `R` and `Python`, asking for R rows and C columns simultaneously returns $R \times C$ `[dD]ata\.?[fF]rame`.

```
# R:

df[1:3, 1:3]
df[c('a', 'b', 'c'), c('A', 'B', 'C')]
```

```
# Python:

df.ix[0:3, 0:3]
df.ix[ ['a', 'b', 'c'], ['A', 'B', 'C'] ]
```

Accessing Data — Where...

In both `R` and `Python`, can also select rows or columns of `[dD]ata\.?[fF]rame` using boolean vectors (or matrices).

R:

```
df[df$B > 0, ]            ## all rows where df$B > 0
df[df$B > 0, 'C']         ## col C vals where df$B > 0
df[df$B > 0, 'B'] = 0     ## now all df$B <= 0
df[ , df[1, ] > 0]        ## all cols where first row > 0
```

Python:

```
df.ix[df['B'] > 0]        ## all rows where df.B > 0
df.ix[df['B'] > 0, 'C']   ## col C vals where df.B > 0
df[df.B > 0, 'B'] = 0     ## now all df['B'] <= 0
df.ix[:, df.ix[0] > 0]    ## all cols where first row > 0
```

Normalization — RNA-seq

Basic measurement unit of RNA-seq is count of reads mapped to a given marker (gene, exon, etc.).

Besides biological expression levels, many technical factors influence these counts as well, e.g.:

1. differences in library size (sequencing depth)
2. length of gene

Normalization — RNA-seq

Basic measurement unit of RNA-seq is count of reads mapped to a given marker (gene, exon, etc.).

Besides biological expression levels, many technical factors influence these counts as well, e.g.:

1. differences in library size (sequencing depth)
2. length of gene

Simplest normalization schemes account for these influences by

1. dividing the total library size (and multiplying by 10^6) to obtain CPM or
2. further dividing by gene length (and multiplying by 10^3) to obtain RPKM

(Normalization for gene length may not be necessary in studies which do not attempt to compare expression levels between different genes.)

Normalization — RNA-seq: Better

Some studies have found that RPKM normalization may not appropriately control for association between gene length and read counts (Dillies *et al.* (2013)).

Further, both CPM and RPKM may overweight influence of few very highly expressed genes which may actually be differentially expressed across samples.

Normalization — RNA-seq: Better

Some studies have found that RPKM normalization may not appropriately control for association between gene length and read counts (Dillies *et al.* (2013)).

Further, both CPM and RPKM may overweight influence of few very highly expressed genes which may actually be differentially expressed across samples.

Simple alternatives are to use upper quartile- or median-read count as sample normalization factor instead of sum; Dillies *et al.* (2013) found these options preferable.

Normalization — RNA-seq: Better

Some studies have found that RPKM normalization may not appropriately control for association between gene length and read counts (Dillies *et al.* (2013)).

Further, both CPM and RPKM may overweight influence of few very highly expressed genes which may actually be differentially expressed across samples.

Simple alternatives are to use upper quartile- or median-read count as sample normalization factor instead of sum; Dillies *et al.* (2013) found these options preferable.

More complex normalization methods offered by the `R` packages `DESeq` and `edgeR`; may offer better performance in some circumstances.

Normalization — RNA-seq: Upper Quartile

R:

```
uqnormalize = function(x, MARGIN=1, scale=100) {
    geneDetected = (apply(X=x, MARGIN=3-MARGIN, FUN=sum) > 0)
    return(scale * sweep(
        x = x,
        MARGIN = MARGIN,
        STATS = apply(X=x, MARGIN=MARGIN,
                FUN=function(z) {quantile(z[geneDetected], 0.75)}),
        FUN = `/`
    ))
}
```

Python:

```
def uqnormalize(x, axis=0, scale=100):
    geneDetected = (x.sum(axis=axis) > 0)
    if axis == 0:
        xgd = x[ x.columns[geneDetected] ]
    elif axis == 1:
        xgd = x.ix[geneDetected]
    normfacs = numpy.percentile(xgd, q=75, axis=1-axis)
    return scale * x.divide(normfacs, axis=axis)
```

Normalization — RT-qPCR

Basic measurement of RT-qPCR is Ct for given gene (primer pair).

Once again, technical factors such as quantity or quality of nucleic acid in sample may influence measured Ct values.

Since Ct values are already in log-copy number space, simple sample-mean-centering approach can work well...

$$\Delta x_{ig} = x_{ig} - \frac{1}{p}\sum_{h=1}^{p} x_{ih}$$

Normalization — RT-qPCR

Basic measurement of RT-qPCR is Ct for given gene (primer pair).

Once again, technical factors such as quantity or quality of nucleic acid in sample may influence measured Ct values.

Since Ct values are already in log-copy number space, simple sample-mean-centering approach can work well...

$$\Delta x_{ig} = x_{ig} - \frac{1}{p}\sum_{h=1}^{p} x_{ih}$$

...if many genes are measured with expectation that most are not differentially expressed and ...

Normalization — RT-qPCR

Basic measurement of RT-qPCR is Ct for given gene (primer pair).

Once again, technical factors such as quantity or quality of nucleic acid in sample may influence measured Ct values.

Since Ct values are already in log-copy number space, simple sample-mean-centering approach can work well...

$$\Delta x_{ig} = x_{ig} - \frac{1}{p}\sum_{h=1}^{p} x_{ih}$$

...if many genes are measured with expectation that most are not differentially expressed and ...

...if none of the Ct values x_{ig} are missing/undefined.

Normalization — RT-qPCR: Mean-Centering

R:

```
meanCenter = function(x, MARGIN=1) {
    geneHasNAs = apply(x, 3-MARGIN, function(z) {any(is.na(z))})
    means = apply(x, MARGIN, function(z) {mean(z[!geneHasNAs])})
    return(sweep(x, MARGIN, means, `-`))
}
```

Python:

```
def meanCenter(x, axis=0):
    geneHasNans = (numpy.isnan(x).sum(axis=axis) > 0)
    if axis == 0:
        xnonans = x[ x.columns[~geneHasNans] ]
    elif axis == 1:
        xnonans = x.ix[~geneHasNans]
    means = xnonans.mean(axis=1-axis)
    return x.add(-means, axis=axis)
```

Normalization — RT-qPCR: Normalizers

Conceptually more difficult to deal with RT-qPCR data normalization when most measured genes are differentially expressed.

Usual answer in this case is to include a few "stably expressed" **normalizer** genes in panel.

How does one know what genes are stably expressed?

Normalization — RT-qPCR: Normalizers

Conceptually more difficult to deal with RT-qPCR data normalization when most measured genes are differentially expressed.

Usual answer in this case is to include a few "stably expressed" **normalizer** genes in panel.

How does one know what genes are stably expressed?

1. Use genes other people have declared stable in literature, or

Normalization — RT-qPCR: Normalizers

Conceptually more difficult to deal with RT-qPCR data normalization when most measured genes are differentially expressed.

Usual answer in this case is to include a few "stably expressed" **normalizer** genes in panel.

How does one know what genes are stably expressed?

1. Use genes other people have declared stable in literature, or
2. First apply algorithm to identify normalizers (e.g., Vandesompele *et al.* (2002); Andersen *et al.* (2004); Wylie *et al.* (2011)) to large panel where most genes are not expected to be differentially expressed.

Unsupervised Learning

$D = \mathbf{x}$ with no particular outcome identified; focus on identifying common patterns in $\mathbf{x}$ alone.

What do we mean by "patterns?"

- clusters (subgroupings of "similar" samples or genes)
- relationships between variables (gene expression levels or other covariates)
  - strong relationships may lead to identification of hidden/latent factors simultaneously influencing many variables
  - useful for **dimensionality reduction**

While most approaches *can* be represented as probabilistic model

$$\mathbb{P}(\mathbf{X} \mid M, \boldsymbol{\theta})$$

some may be more simply presented without the extra theoretical baggage.

Clustering

Want to find groups of samples i or genes g such that:

- similarity of objects within same group tends to be high
- similarity between objects in different groups tends to be low.

Wide range of complexity in clustering methods; here focus on relatively simple approaches.

Useful way to check data quality/confirm expectations (or, alternatively, spot unexpected structure in data).

Clustering

Want to find groups of samples i or genes g such that:

- similarity of objects within same group tends to be high
- similarity between objects in different groups tends to be low.

Wide range of complexity in clustering methods; here focus on relatively simple approaches.

Useful way to check data quality/confirm expectations (or, alternatively, spot unexpected structure in data).

- if replicates are present, do they cluster together?

Clustering

Want to find groups of samples i or genes g such that:

- similarity of objects within same group tends to be high
- similarity between objects in different groups tends to be low.

Wide range of complexity in clustering methods; here focus on relatively simple approaches.

Useful way to check data quality/confirm expectations (or, alternatively, spot unexpected structure in data).

- if replicates are present, do they cluster together?
- do samples taken from similar tissues, conditions, time points, etc. cluster together?

Clustering

Want to find groups of samples i or genes g such that:

- similarity of objects within same group tends to be high
- similarity between objects in different groups tends to be low.

Wide range of complexity in clustering methods; here focus on relatively simple approaches.

Useful way to check data quality/confirm expectations (or, alternatively, spot unexpected structure in data).

- if replicates are present, do they cluster together?
- do samples taken from similar tissues, conditions, time points, etc. cluster together?
- do samples cluster by processing batch or order?

Similarity, Dissimilarity, and Distance

Often work with dissimilarity measures (often distance metrics) as opposed to similarities.

Common dissimilarity metrics:

1. **Euclidean distance** $d(\mathbf{x}_1, \mathbf{x}_2) = \|\mathbf{x}_1 - \mathbf{x}_2\|_2$
2. **Pearson correlation dissimilarity**

$$d(\mathbf{x}_1, \mathbf{x}_2) = 1 - \frac{\Delta\mathbf{x}_1 \cdot \Delta\mathbf{x}_2}{\|\Delta\mathbf{x}_1\|\|\Delta\mathbf{x}_2\|}$$

where $\Delta\mathbf{x} = \mathbf{x} - \frac{1}{p}\sum_{g=1}^{p} x_g$.

3. **Spearman correlation dissimilarity**

$$d(\mathbf{x}_1, \mathbf{x}_2) = 1 - \frac{\Delta\text{rank}(\mathbf{x}_1) \cdot \Delta\text{rank}(\mathbf{x}_2)}{\|\Delta\text{rank}(\mathbf{x}_1)\|\|\Delta\text{rank}(\mathbf{x}_2)\|}$$

k-Means Clustering (MacQueen (1967))

Algorithm:

1. Initialize k "centroids" $\mathbf{c}_a$.
2. Assign each datum $\mathbf{x}_i$ to nearest cluster:

$$\text{clust}(\mathbf{x}_i) = \arg\min_a \|\mathbf{x}_i - \mathbf{c}_a\|$$

3. Reset centroids to mean of associated data:

$$\mathbf{c}_a = \frac{1}{|S_a|} \sum_{i \in S_a} \mathbf{x}_i$$

   where the set $S_a = \{i \mid \text{clust}(\mathbf{x}_i) = a\}$.
4. Repeat steps 2-3 until convergence.

k-Means Clustering (MacQueen (1967))

Algorithm:

1. Initialize k "centroids" $\mathbf{c}_a$.
2. Assign each datum $\mathbf{x}_i$ to nearest cluster:

$$\text{clust}(\mathbf{x}_i) = \arg\min_a \|\mathbf{x}_i - \mathbf{c}_a\|$$

3. Reset centroids to mean of associated data:

$$\mathbf{c}_a = \frac{1}{|S_a|} \sum_{i \in S_a} \mathbf{x}_i$$

   where the set $S_a = \{i \mid \text{clust}(\mathbf{x}_i) = a\}$.
4. Repeat steps 2-3 until convergence.

Locally minimizes $\sum_{a=1}^{k} \sum_{i \in S_a} (\mathbf{x}_i - \mathbf{c}_a)^2$.

k-Means Clustering

k-means clustering is fast and intuitive . . .

. . . but it tends to produce (higher-dimensional) spherical, equal-sized clusters whether they are appropriate or not.

k-Means Clustering

k-means clustering is fast and intuitive . . .

. . . but it tends to produce (higher-dimensional) spherical, equal-sized clusters whether they are appropriate or not.

k-means clustering can be derived from the small σ limit of probabilistic mixture-of-Gaussians model M with parameters $\boldsymbol{\theta} = (\mathbf{c}, \sigma)$ (Ghahramani (2004)):

$$\mathbb{P}(\mathbf{X} = \mathbf{x} \mid M, \mathbf{c}, \sigma) = \sum_{a=1}^{k} \frac{1}{k\sqrt{(2\pi\sigma^2)^p}} \exp\left[\frac{(\mathbf{x} - \mathbf{c}_a)^2}{2\sigma^2}\right]$$

where each Gaussian in the mixture has common spherical covariance matrix $\sigma^2 I$.

Hierarchical Clustering

Also known as agglomerative (bottom-up) clustering (Mary-Huard *et al.* (2006); Hastie *et al.* (2009)).

Requires extension of (dis)similarity metric from pairs of data $d(\mathbf{x}_i, \mathbf{x}_j)$ to pairs of *clusters*:

$$d(S_a, S_b) = ???$$

For example, hierarchical clustering with so-called "average linkage" defines

$$d(S_a, S_b) = \sum_{i \in S_a} \sum_{j \in S_b} \frac{d(\mathbf{x}_i, \mathbf{x}_j)}{|S_a||S_b|}$$

. . . but there are many other possible choices of aggregation criterion as well.

Hierarchical Clustering

Algorithm:

1. Initialize each datum to own cluster, $S_i = \{i\}$, define initial set of active clusters $A_0 = \{1, 2, \ldots, n\}$.
2. For iteration t, select two most similar active clusters and merge:

$$
\begin{aligned}
(a_t, b_t) &= \underset{(a,b) \in A_{t-1} \times A_{t-1} \mid a<b}{\arg\min} d(S_a, S_b) \\
S_{n+t} &= S_{a_t} \cup S_{b_t} \\
A_t &= (A_{t-1} \setminus \{a_t, b_t\}) \cup \{n + t\}
\end{aligned}
$$

3. If $t < (n - 1)$, increment t and repeat step 2. (Note: if you know you want exactly k clusters, stop when $t = n - k$.)

Hierarchical Clustering

Algorithm:

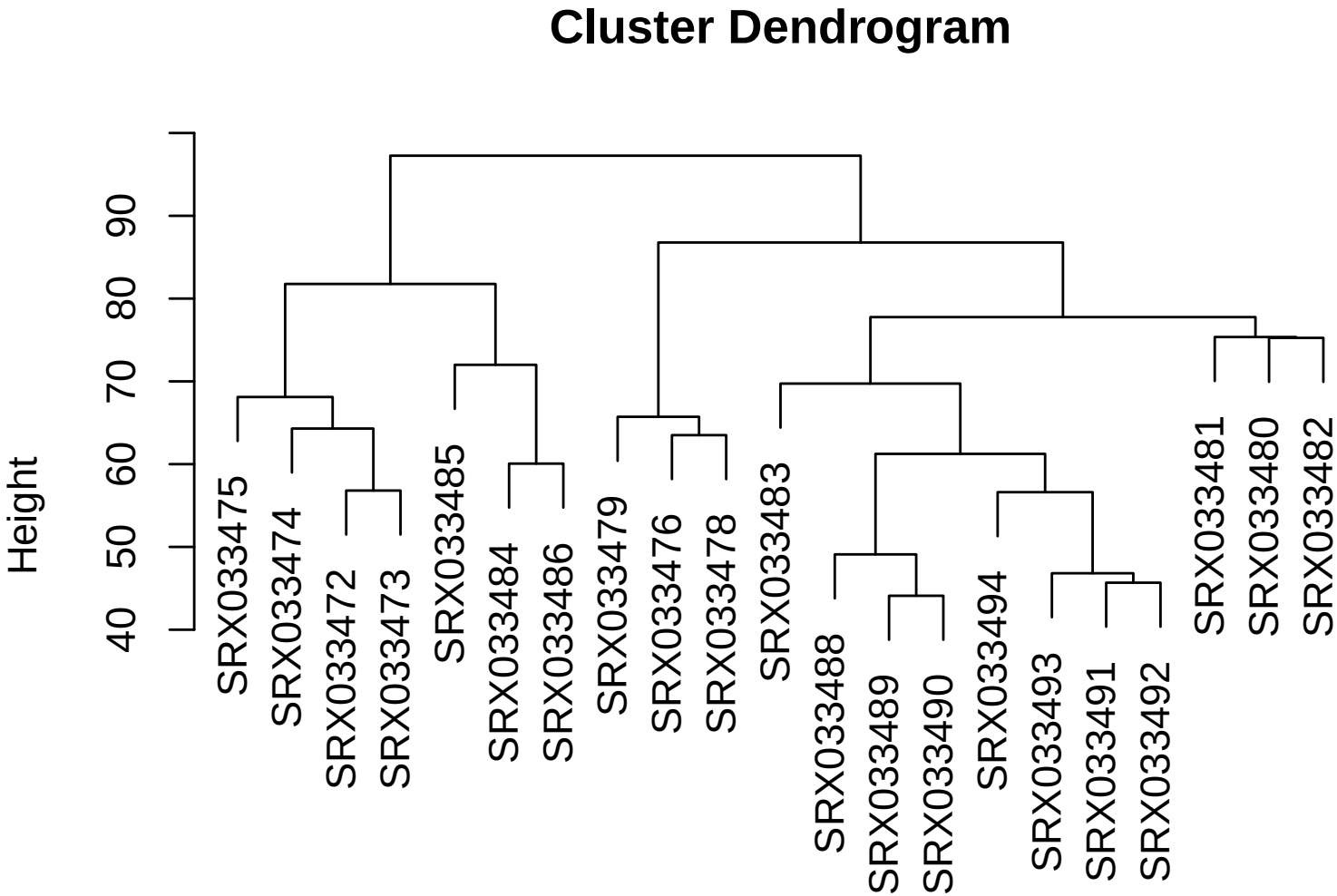
1. Initialize each datum to own cluster, $S_i = \{i\}$, define initial set of active clusters $A_0 = \{1, 2, \ldots, n\}$.
2. For iteration t, select two most similar active clusters and merge:

$$(a_t, b_t) = \underset{(a,b) \in A_{t-1} \times A_{t-1} \mid a<b}{\arg\min} d(S_a, S_b)$$
$$S_{n+t} = S_{a_t} \cup S_{b_t}$$
$$A_t = (A_{t-1} \setminus \{a_t, b_t\}) \cup \{n + t\}$$

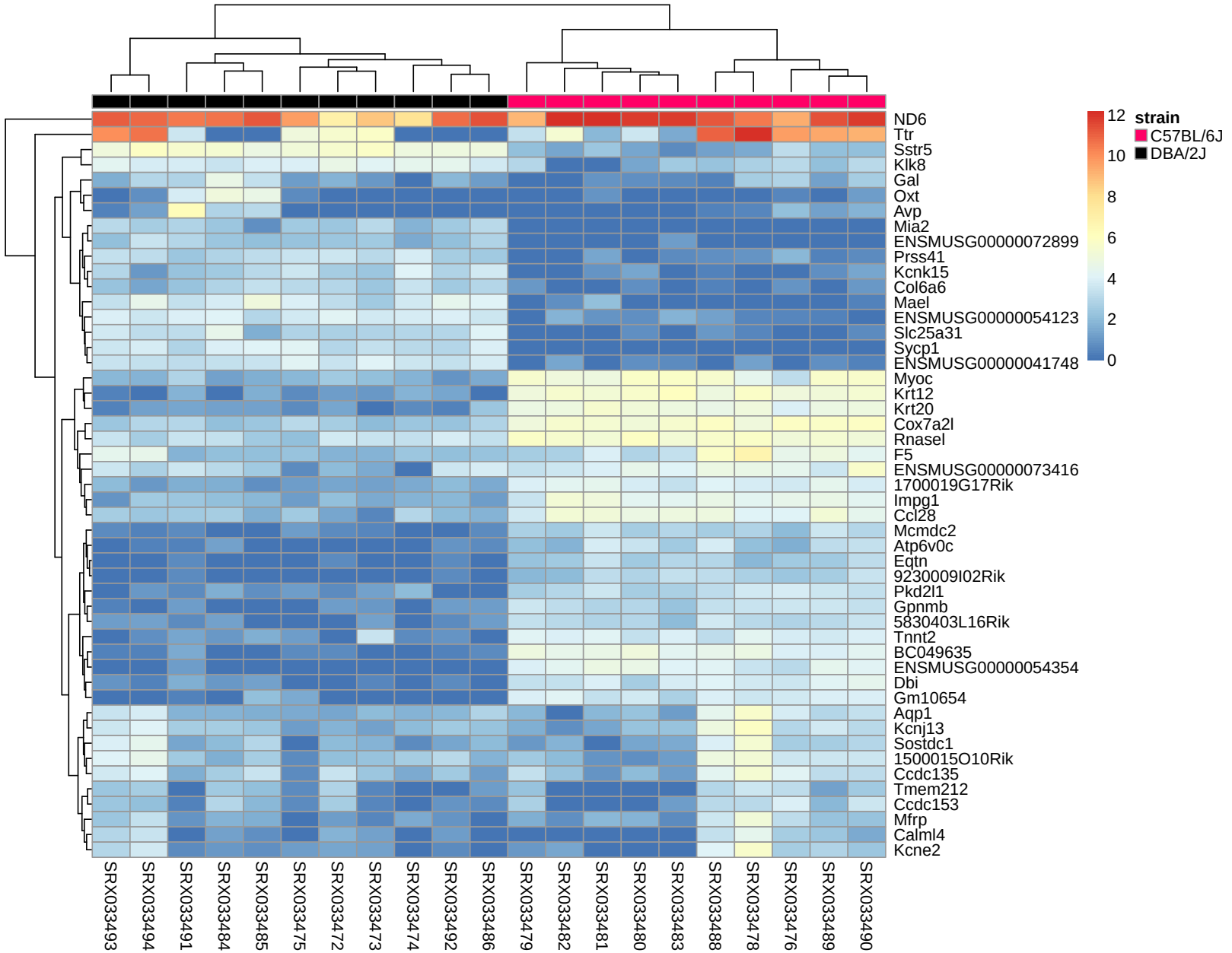
3. If $t < (n-1)$, increment t and repeat step 2. (Note: if you know you want exactly k clusters, stop when $t = n - k$.)

A **dendrogram** can be obtained from this process by connecting the two merged clusters a_t and b_t to the newly created merged cluster $(n + t)$ sequentially for each iteration t.

Hierarchical Clustering (Bottomly Samples)



Hierarchical Clustering (High Variance Genes)



Hierarchical Clustering

Some commonly used aggregation criteria:

Average Linkage

$$d(S_a, S_b) = \sum_{i \in S_a} \sum_{j \in S_b} \frac{d(\mathbf{x}_i, \mathbf{x}_j)}{|S_a||S_b|}$$

Single Linkage

$$d(S_a, S_b) = \min_{i \in S_a, j \in S_b} d(\mathbf{x}_i, \mathbf{x}_j)$$

Complete Linkage

$$d(S_a, S_b) = \max_{i \in S_a, j \in S_b} d(\mathbf{x}_i, \mathbf{x}_j)$$

Centroid (where $\mathbf{c}_a$ is centroid of cluster a)

$$d(S_a, S_b) = d(\mathbf{c}_a, \mathbf{c}_b)$$

Ward

$$d^2(S_a, S_b) = \frac{|S_a||S_b|}{|S_a| + |S_b|} d^2(\mathbf{c}_a, \mathbf{c}_b)$$

Hierarchical Clustering: A Warning

From Mary-Huard *et al.* (2006):

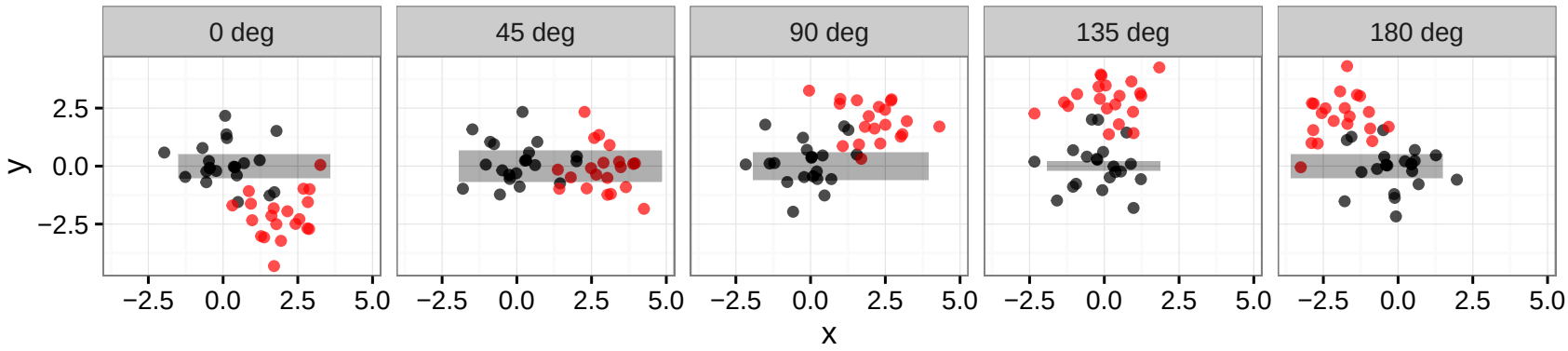
> It has to be noted that the tree structure is the result of the clustering history, but does not reveal some presupposed underlying structure ...
>
> ... Hierarchical algorithms always provide a tree, even if the data are not structured according to a tree ...
>
> ... This is a major drawback of these 'algorithmic' approaches: because of the lack of statistical modeling, the fit of the representation to the data is difficult to assess.

What is PCA?

From `https://en.wikipedia.org/wiki/Principal_component_analysis`:

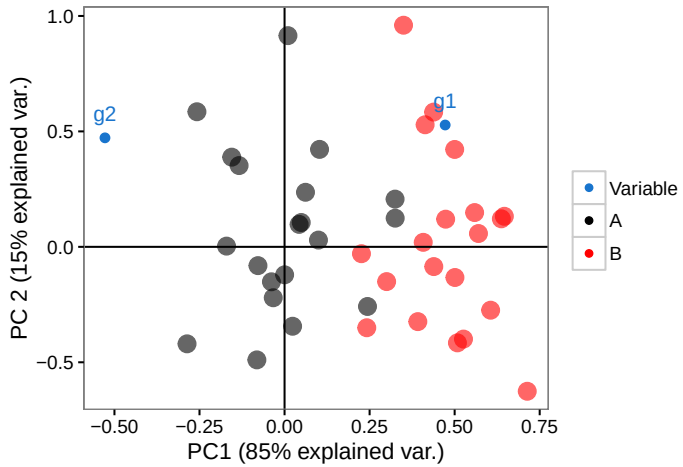
> . . . a statistical procedure that uses an **orthogonal transformation** to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components . . .
>
> This transformation is defined in such a way that the **first principal component** has the **largest possible variance** (that is, accounts for as much of the variability in the data as possible),
>
> and each succeeding component in turn has the highest variance possible under the constraint that it is orthogonal to the preceding components.
>
> The resulting vectors are an uncorrelated orthogonal basis set. The principal components are orthogonal because they are the eigenvectors of the covariance matrix, which is symmetric.
>
> PCA is **sensitive to the relative scaling** of the original variables.

Rotating Simulated Data



Rotation by 45° maximizes variance in x-direction.

PCA finds rotation for which this variance is maximized . . .



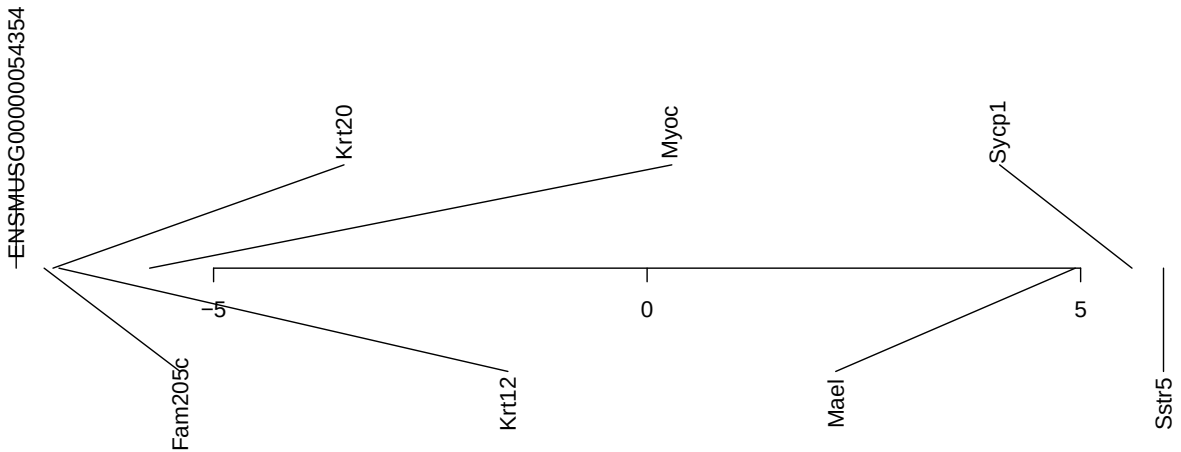
. . . but generalized to higher dimensionalities.

Another view of PCA

Find a single linear "pattern" w_g which explains as much of the variation in expression levels* as possible by minimizing $\sum_{i,g} \epsilon_{ig}^2$ in

$$\tilde{x}_{ig} = u_i w_g + \epsilon_{ig}$$

- $\underline{u}$ composed of PC1 coordinates u_i of samples i.
- $\mathbf{w}$ proportional to PC1 coordinates w_g of genes g.



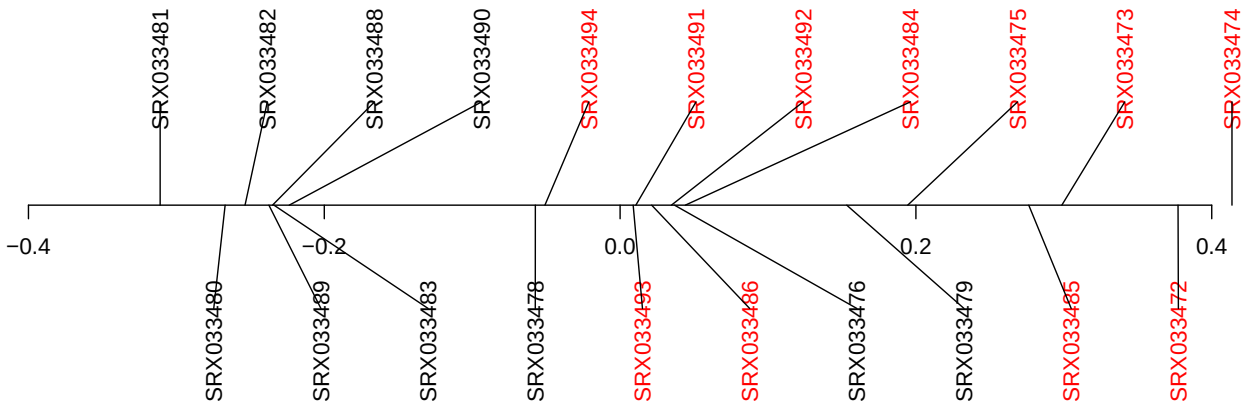
largest magnitude w_g values for Bottomly data set
(after centering both rows and columns of data matrix $\underline{\mathbf{X}}$ to obtain $\underline{\tilde{\mathbf{X}}}$).

Another view of PCA

Find a single linear "pattern" w_g which explains as much of the variation in expression levels* as possible by minimizing $\sum_{i,g} \epsilon_{ig}^2$ in

$$\tilde{x}_{ig} = u_i w_g + \epsilon_{ig}$$

- $\underline{u}$ composed of PC1 coordinates u_i of samples i.
- $\mathbf{w}$ proportional to PC1 coordinates w_g of genes g.



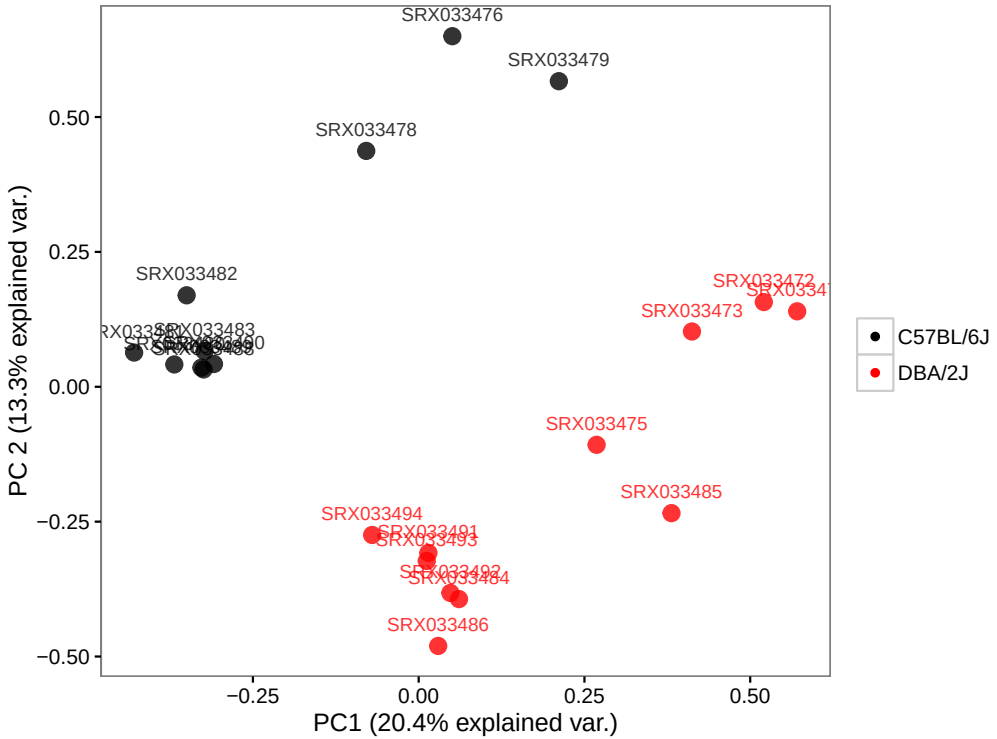
u_i values for all samples i for Bottomly data set
(after centering both rows and columns of data matrix $\underline{\mathbf{X}}$ to obtain $\underline{\tilde{\mathbf{X}}}$).

PCA Plot

It seems unreasonable to expect a single linear pattern to explain everything in our data, though. . . let's try two:

$$\tilde{x}_{ig} = \left(u_{i1} w_{g1} + u_{i2} w_{g2}\right) + \epsilon_{ig}$$

and again select u_{iq}, w_{gq}, and ϵ_{ig} so as to minimize $\sum_{i,g} \epsilon_{ig}^2$ . . .



Here I've rescaled both axes so that range of x-axis is 1 unit.

PCA by SVD

Can continue this process to obtain n principal components (assuming $n \leq p$). These can be calculated via the singular value decomposition (SVD) of $\underline{\tilde{\mathbf{X}}}$ (note $\underline{\tilde{\mathbf{X}}}$ here denotes matrix, not random variable)

$$\underline{\tilde{\mathbf{X}}} = \underline{\mathbf{U}}\underline{\mathbf{D}}\underline{\mathbf{V}}^T$$

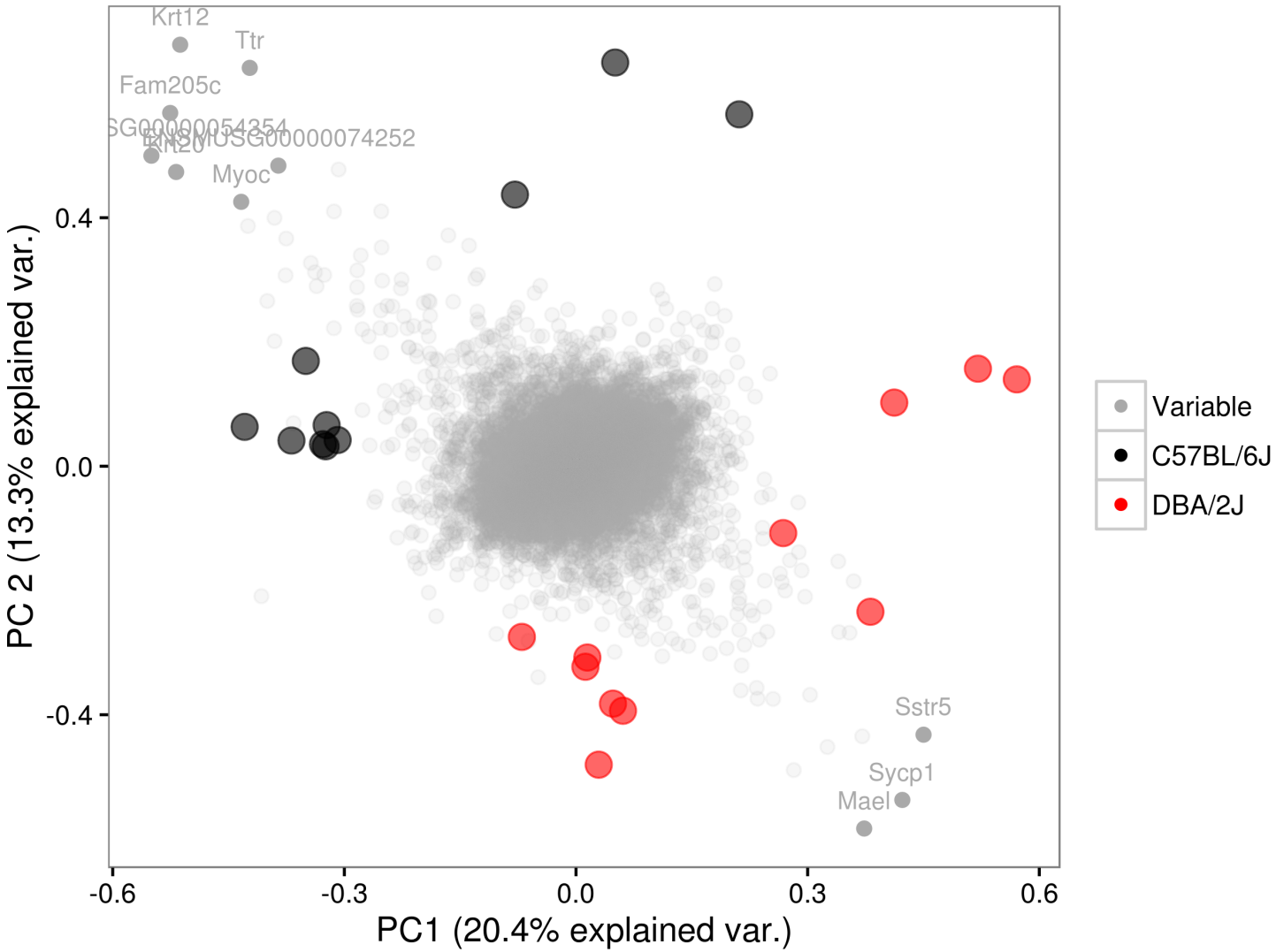
where $\underline{\mathbf{U}}$ and $\underline{\mathbf{V}}$ are orthogonal matrices and $\underline{\mathbf{D}}$ is an $n \times n$ diagonal matrix with the diagonal sorted in descending order. In terms of the components $\tilde{x}_{ig}$:

$$\tilde{x}_{ig} = \sum_q u_{iq} d_{qq} v_{gq}$$

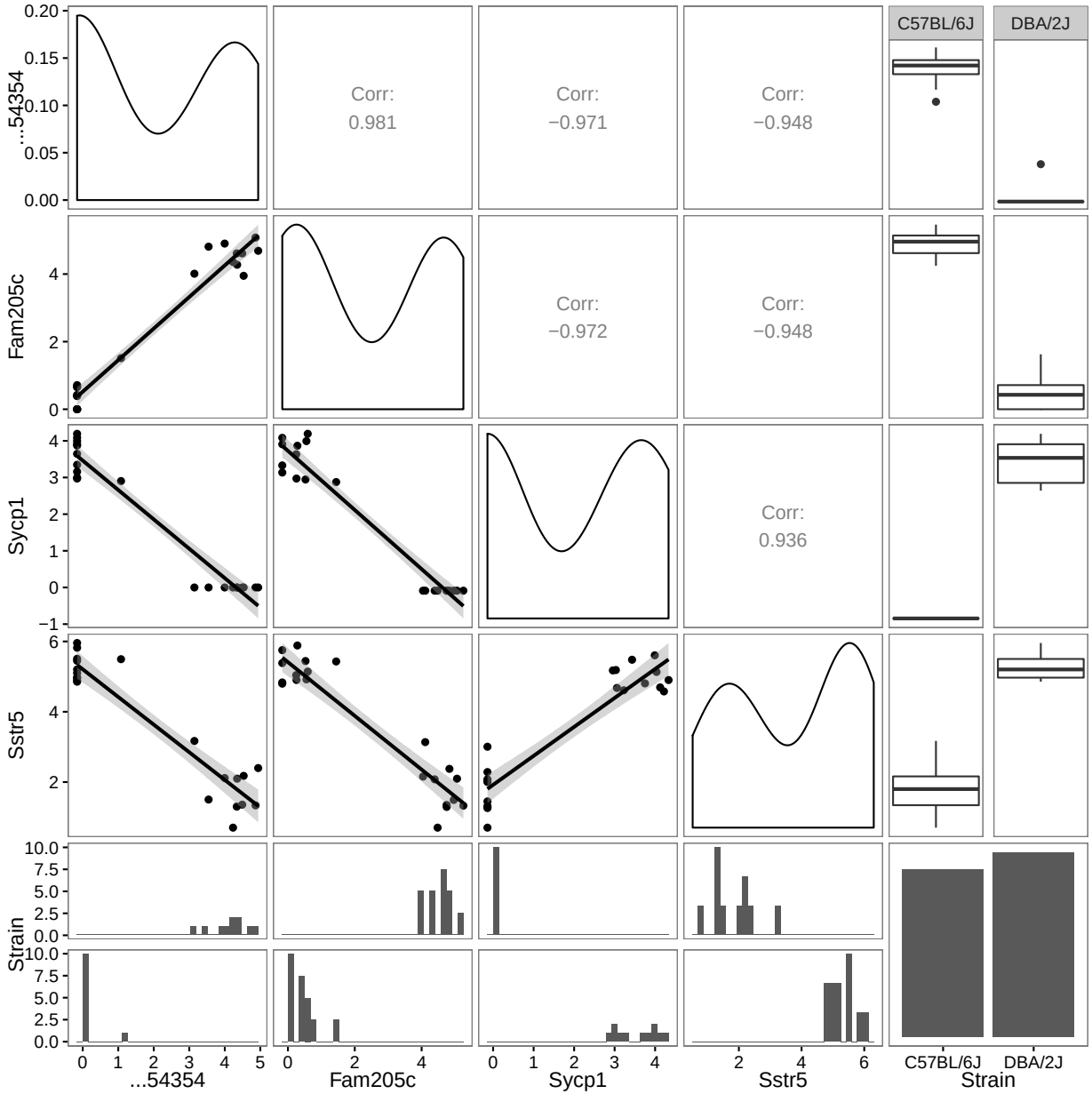
This looks familiar if we take $w_{gq} = d_{qq} v_{gq}$ ...

PCA Biplot

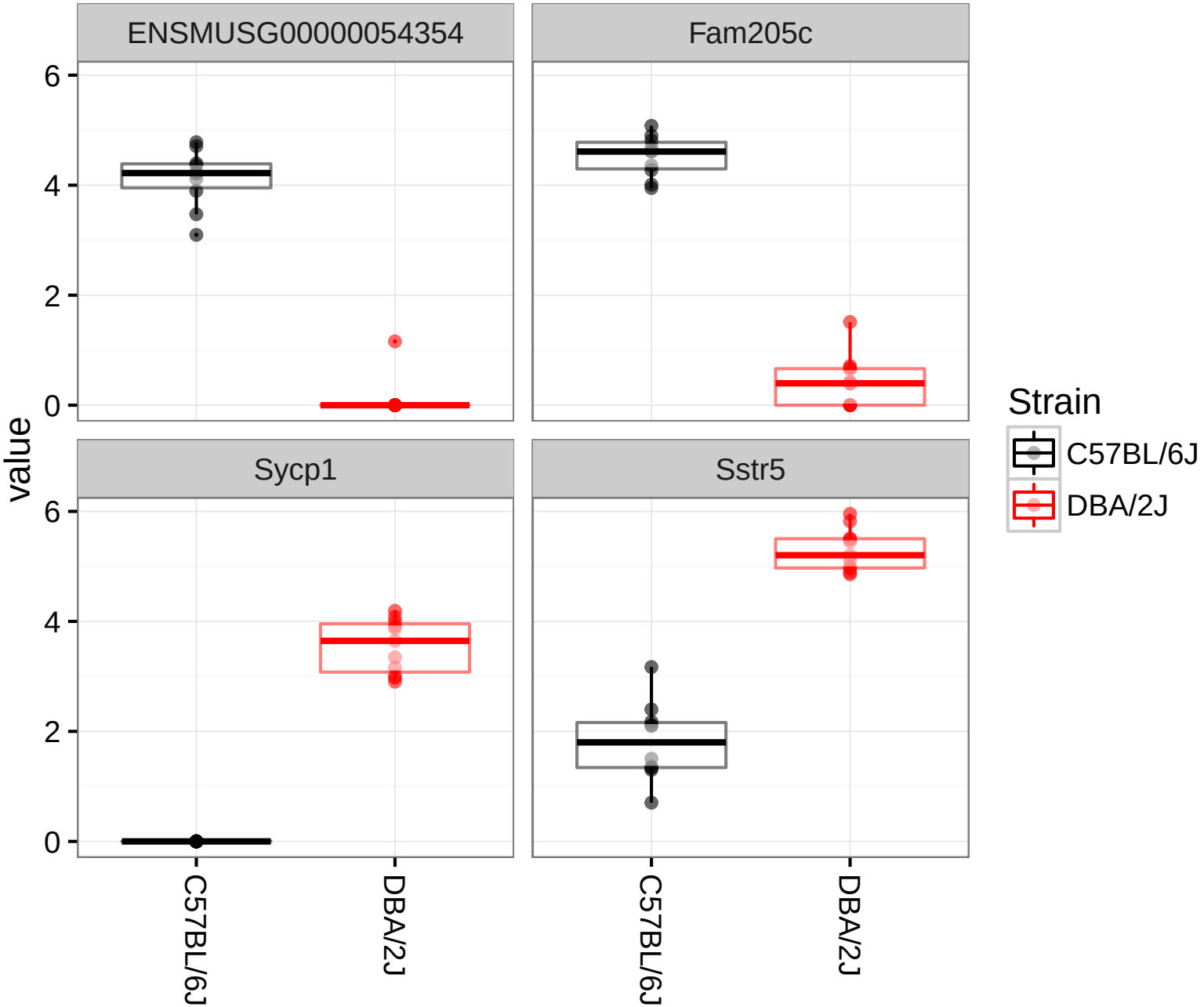
Useful to also plot the points (v_{g1}, v_{g2}) for those genes g with large contributions to the first two principal components:



PCA Biplot Gene Expression Levels



PCA Biplot Gene Expression Levels



PCA and Eigendecomposition

The SVD of $\underline{\tilde{\mathbf{X}}}$ also has the following interesting properties:

- $\underline{\mathbf{D}}$ the diagonal of $\underline{\mathbf{D}}$ contains the square roots of the eigenvalues of $\underline{\tilde{\mathbf{X}}\tilde{\mathbf{X}}}^T$ (and thus also of $\underline{\tilde{\mathbf{X}}}^T\underline{\tilde{\mathbf{X}}}$).
- $\underline{\mathbf{U}}$ the columns of $\underline{\mathbf{U}}$ are the eigenvectors of the matrix $\underline{\tilde{\mathbf{X}}\tilde{\mathbf{X}}}^T$, so that $\underline{\tilde{\mathbf{X}}\tilde{\mathbf{X}}}^T\underline{\mathbf{U}} = \underline{\mathbf{U}\mathbf{D}^2}$.
- $\underline{\mathbf{V}}$ the columns of $\underline{\mathbf{V}}$ are the n eigenvectors of the matrix $\underline{\tilde{\mathbf{X}}}^T\underline{\tilde{\mathbf{X}}}$ with non-zero eigenvalues, so that $\underline{\tilde{\mathbf{X}}}^T\underline{\tilde{\mathbf{X}}\mathbf{V}} = \underline{\mathbf{V}\mathbf{D}^2}$.

Note that:

- $\frac{1}{n-1}\underline{\tilde{\mathbf{X}}}^T\underline{\tilde{\mathbf{X}}}$ is the estimated gene-gene covariance matrix, and
- $\frac{1}{p-1}\underline{\tilde{\mathbf{X}}\tilde{\mathbf{X}}}^T$ is the estimated sample-sample covariance matrix.

So SVD relates the eigendecompositions of the estimated gene- and sample-covariance matrices.

PCA: Centering and Scaling

Why have I been writing $\underline{\tilde{\mathbf{X}}}$ instead of just $\underline{\mathbf{X}}$?

PCA: Centering and Scaling

Why have I been writing $\underline{\tilde{\mathbf{X}}}$ instead of just $\underline{\mathbf{X}}$?

PCA methods are generally applied to a transformed—**centered** and possibly **scaled** in row and/or column space—version of the original data matrix $\underline{\mathbf{X}}$.

Different transformations lead to different PCA variants (Wouters *et al.* (2003)).

PCA: Centering and Scaling

Why have I been writing $\underline{\tilde{\mathbf{X}}}$ instead of just $\underline{\mathbf{X}}$?

PCA methods are generally applied to a transformed—**centered** and possibly **scaled** in row and/or column space—version of the original data matrix $\underline{\mathbf{X}}$.

Different transformations lead to different PCA variants (Wouters *et al.* (2003)).

My default choice is to center both rows and columns:

$$\tilde{x}_{ig} = x_{ig} - \frac{1}{p}\sum_{h=1}^{p} x_{ih} - \frac{1}{n}\sum_{j=1}^{n} x_{jg} + \frac{1}{np}\sum_{j,h} x_{jh}$$

but to do no variance scaling.

PCA: Centering and Scaling

Why have I been writing $\underline{\tilde{\mathbf{X}}}$ instead of just $\underline{\mathbf{X}}$?

PCA methods are generally applied to a transformed—**centered** and possibly **scaled** in row and/or column space—version of the original data matrix $\underline{\mathbf{X}}$.

Different transformations lead to different PCA variants (Wouters *et al.* (2003)).

My default choice is to center both rows and columns:

$$\tilde{x}_{ig} = x_{ig} - \frac{1}{p}\sum_{h=1}^{p} x_{ih} - \frac{1}{n}\sum_{j=1}^{n} x_{jg} + \frac{1}{np}\sum_{j,h} x_{jh}$$

but to do no variance scaling.

Variance scaling is arguably not needed for gene expression data because all variables have same units. Use of variance scaling in such contexts can overweight very low-variance genes and amplify noise.

Probabilistic PCA and Factor Analysis

PCA using $k \leq \min(n, p)$ principal components can be derived from the small σ limit of a probabilistic model:

$$\mathbb{P}\left(\tilde{\mathbf{X}} = \tilde{\mathbf{x}} \mid M, [w_{ga}], (z_a)\right) = \frac{1}{\sqrt{2\pi\sigma^2}^p} \exp\left[-\frac{1}{2\sigma^2}\sum_{g=1}^{p}\left(\tilde{x}_g - \sum_{a=1}^{k} w_{ga} z_a\right)^2\right]$$

where the k-dimensional vector $\mathbf{z} \sim \mathcal{N}(0, I)$ (with components z_a) is a *latent* (unobserved) variable.

Probabilistic PCA and Factor Analysis

PCA using $k \leq \min(n, p)$ principal components can be derived from the small σ limit of a probabilistic model:

$$\mathbb{P}\left(\tilde{\mathbf{X}} = \tilde{\mathbf{x}} \mid M, [w_{ga}], (z_a)\right) = \frac{1}{\sqrt{2\pi\sigma^2}^p} \exp\left[-\frac{1}{2\sigma^2}\sum_{g=1}^{p}\left(\tilde{x}_g - \sum_{a=1}^{k} w_{ga} z_a\right)^2\right]$$

where the k-dimensional vector $\mathbf{z} \sim \mathcal{N}(0, I)$ (with components z_a) is a *latent* (unobserved) variable.

If instead of assuming that $\widetilde{X}_g - \sum_a w_{ga} Z_a \sim \mathcal{N}(0, \sigma^2)$ for some common fixed infinitesimal σ we allow:

$$E_g = \widetilde{X}_g - \sum_{a=1}^{k} w_{ga} Z_a \sim \mathcal{N}(0, \psi_g^2)$$

where ψ_g is no longer assumed small (but where we retain the assumption of between-gene independence of these error terms), we derive the standard *factor analysis* model (Roweis & Ghahramani (1999)).

References I

Andersen, Claus Lindbjerg, Jensen, Jens Ledet, & Ørntoft, Torben Falck. 2004. Normalization of real-time quantitative reverse transcription-PCR data: a model-based variance estimation approach to identify genes suited for normalization, applied to bladder and colon cancer data sets. *Cancer Research*, **64**(15), 5245–5250.

Bottomly, Daniel, Walter, Nicole AR, Hunter, Jessica Ezzell, Darakjian, Priscila, Kawane, Sunita, Buck, Kari J, Searles, Robert P, Mooney, Michael, McWeeney, Shannon K, & Hitzemann, Robert. 2011. Evaluating gene expression in C57BL/6J and DBA/2J mouse striatum using RNA-Seq and microarrays. *PloS One*, **6**(3), e17820.

Davis, Sean, & Meltzer, Paul S. 2007. GEOquery: a bridge between the Gene Expression Omnibus (GEO) and BioConductor. *Bioinformatics*, **23**(14), 1846–1847.

Dillies, Marie-Agnès, Rau, Andrea, Aubert, Julie, Hennequet-Antier, Christelle, Jeanmougin, Marine, Servant, Nicolas, Keime, Céline, Marot, Guillemette, Castel, David, Estelle, Jordi, *et al.* . 2013. A comprehensive evaluation of normalization methods for Illumina high-throughput RNA sequencing data analysis. *Briefings in Bioinformatics*, **14**(6), 671–683.

Frazee, Alyssa C, Langmead, Ben, & Leek, Jeffrey T. 2011. ReCount: a multi-experiment resource of analysis-ready RNA-seq gene count datasets. *BMC Bioinformatics*, **12**(1), 449.

Ghahramani, Zoubin. 2004. Unsupervised Learning. *Pages 72–112 of: Advanced Lectures on Machine Learning*. Springer.

Hastie, Trevor, Tibshirani, Robert, & Friedman, Jerome. 2009. *The Elements of Statistical Learning*. Springer.

Hess, Kenneth R, Anderson, Keith, Symmans, W Fraser, Valero, Vicente, Ibrahim, Nuhad, Mejia, Jaime A, Booser, Daniel, Theriault, Richard L, Buzdar, Aman U, Dempsey, Peter J, *et al.* . 2006. Pharmacogenomic predictor of sensitivity to preoperative chemotherapy with paclitaxel and fluorouracil, doxorubicin, and cyclophosphamide in breast cancer. *Journal of Clinical Oncology*, **24**(26), 4236–4244.

References II

MacQueen, James. 1967. Some methods for classification and analysis of multivariate observations. *Pages 281–297 of: Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1. University of California Press.

Mary-Huard, Tristan, Picard, Franck, & Robin, Stéphane. 2006. Introduction to statistical methods for microarray data analysis. *Mathematical and Computational Methods in Biology. Paris: Hermann.*

McKinney, Wes. 2012. *Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython.* O'Reilly Media, Inc.

Montastier, Emilie, Villa-Vialaneix, Nathalie, Caspar-Bauguil, Sylvie, Hlavaty, Petr, Tvrzicka, Eva, Gonzalez, Ignacio, Saris, Wim HM, Langin, Dominique, Kunesova, Marie, & Viguerie, Nathalie. 2015. System Model Network for Adipose Tissue Signatures Related to Weight Changes in Response to Calorie Restriction and Subsequent Weight Maintenance. *PLoS Computational Biology*, **11**(1).

Patel, Anoop P, Tirosh, Itay, Trombetta, John J, Shalek, Alex K, Gillespie, Shawn M, Wakimoto, Hiroaki, Cahill, Daniel P, Nahed, Brian V, Curry, William T, Martuza, Robert L, *et al.* . 2014. Single-cell RNA-seq highlights intratumoral heterogeneity in primary glioblastoma. *Science*, **344**(6190), 1396–1401.

Roweis, Sam, & Ghahramani, Zoubin. 1999. A unifying review of linear Gaussian models. *Neural Computation*, **11**(2), 305–345.

Vandesompele, Jo, De Preter, Katleen, Pattyn, Filip, Poppe, Bruce, Van Roy, Nadine, De Paepe, Anne, & Speleman, Frank. 2002. Accurate normalization of real-time quantitative RT-PCR data by geometric averaging of multiple internal control genes. *Genome Biology*, **3**(7), research0034.

Wouters, Luc, Göhlmann, Hinrich W, Bijnens, Luc, Kass, Stefan U, Molenberghs, Geert, & Lewi, Paul J. 2003. Graphical exploration of gene expression data: a comparative study of three multivariate methods. *Biometrics*, **59**(4), 1131–1139.

Wylie, Dennis, Shelton, Jeffrey, Choudhary, Ashish, & Adai, Alex T. 2011. A novel mean-centering method for normalizing microRNA expression from high-throughput RT-qPCR data. *BMC Research Notes*, **4**(1), 555.