ChIP-seq

Expression Networks

Peter N. Robinson

Institut für Medizinische Genetik und Humangenetik
Charité Universitätsmedizin Berlin

Genomics: Lecture #14

Gene Expression Networks

The ultimate goal of ChIP-seq experiments is to measure genome wide DNA binding of transcription factors or other proteins in order to understand gene regulatory networks. In particular, we want to understand the relationship between DNA-protein binding and transcription.

- This requires integrative genomics analysis of multiple data sources.
  - ChIP-seq
  - RNA-seq
  - in many cases, epigenetics (DNA-methylation, histone, 3-dimensional chromosomal conformation, etc)

Sit back and enjoy

Today, we will talk about an integrated analysis of genomics data on many levels. Sit back and enjoy!

How to Do Good Bioinformatics for Genomics

1. Read mapping
2. Make calls about basic data (variants, isoforms, differential expression, structural variants, ChIP-seq peaks)
3. Integrative bioinformatics (and wetlab experiments) to answer important questions about biology or medicine!

We have not yet covered (3) in this course, but it will be your challenge for the next decade!

Gene Expression Networks

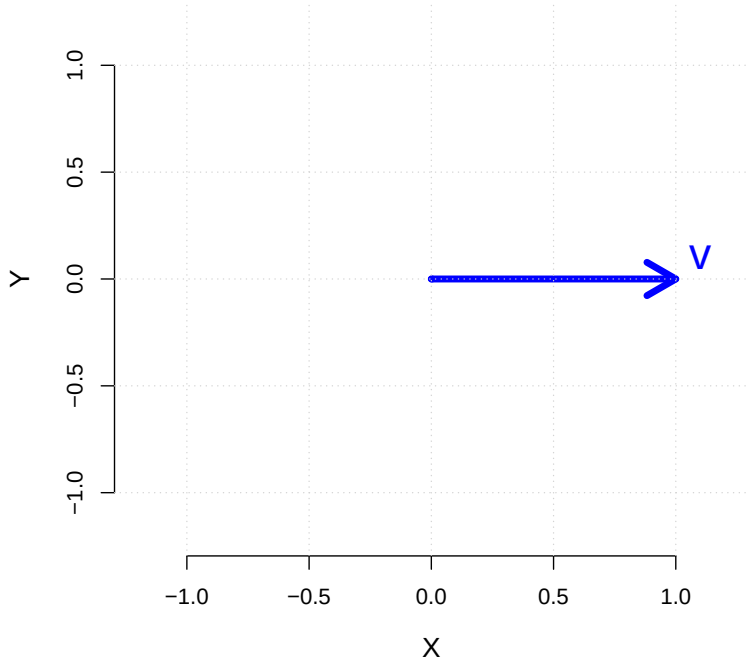
- Today, we will examine the paper Ouyang Z, Zhou Q, Wong WG (2009) ChIP-Seq of transcription factors predicts absolute and differential gene expression in embryonic stem cells. *PNAS* **106**:21251-21526
- We will need to review some material from linear algebra including Principle component analysis (& SVD) before we examine the paper.

Outline

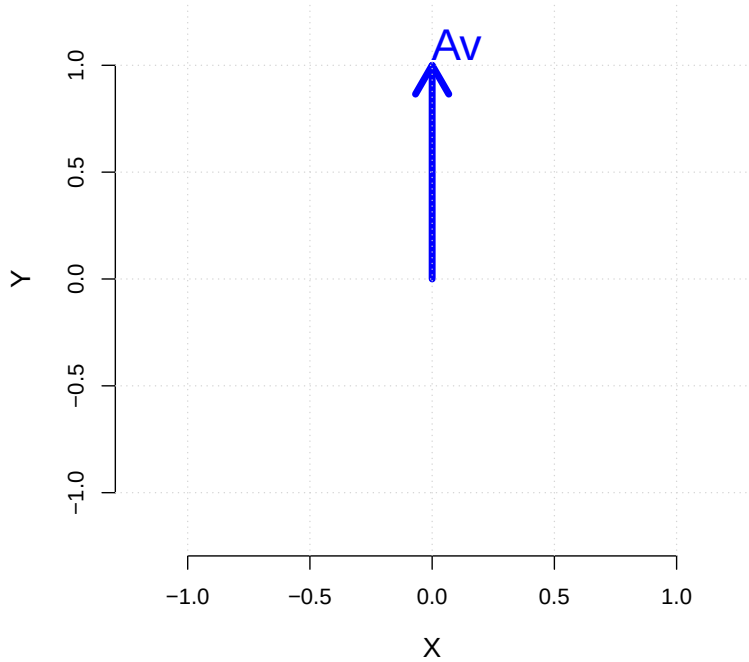
Linear algebra: quick review

- Recall that matrix multiplication can be viewed as a linear mapping, for instance, the matrix $\boldsymbol{A}$ induces a counterclockwise 90°rotation

$$\boldsymbol{v} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$



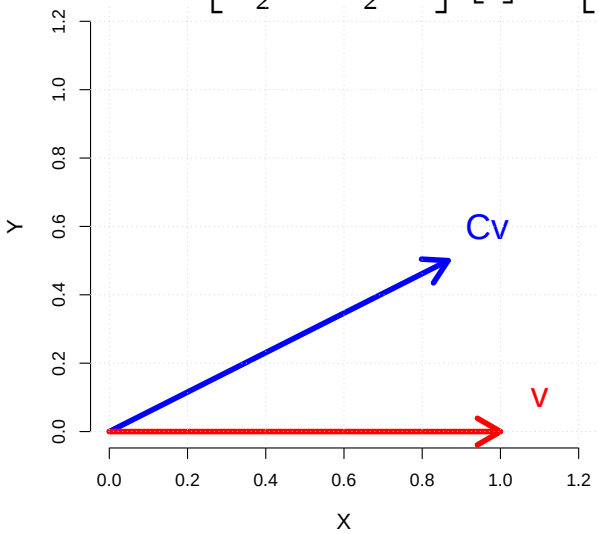
$$\boldsymbol{Av} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$



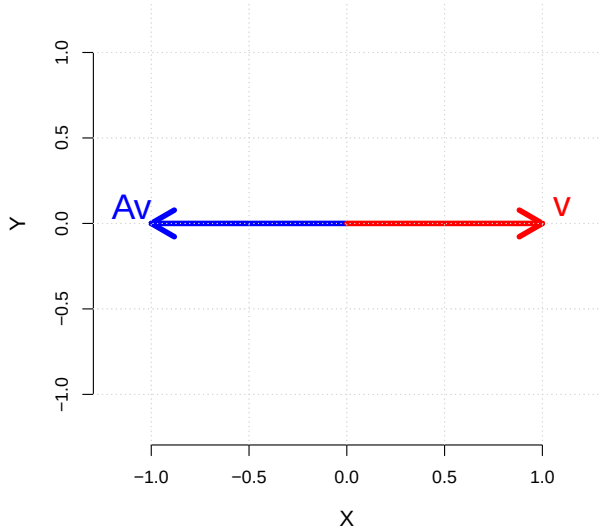
Linear algebra: quick review

- Similarly, the matrix $\boldsymbol{C}$ induces a counterclockwise rotation by and angle of $\theta = \frac{\pi}{6}$=30°and $\boldsymbol{A}$ induces a reflection about the Y axis.

$$\begin{aligned} \boldsymbol{Cv} &= \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \\ &= \begin{bmatrix} \frac{1}{2}\sqrt{3} & -\frac{1}{2} \\ \frac{1}{2} & \frac{1}{2}\sqrt{3} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} \frac{1}{2}\sqrt{3} \\ \frac{1}{2} \end{bmatrix} \end{aligned}$$

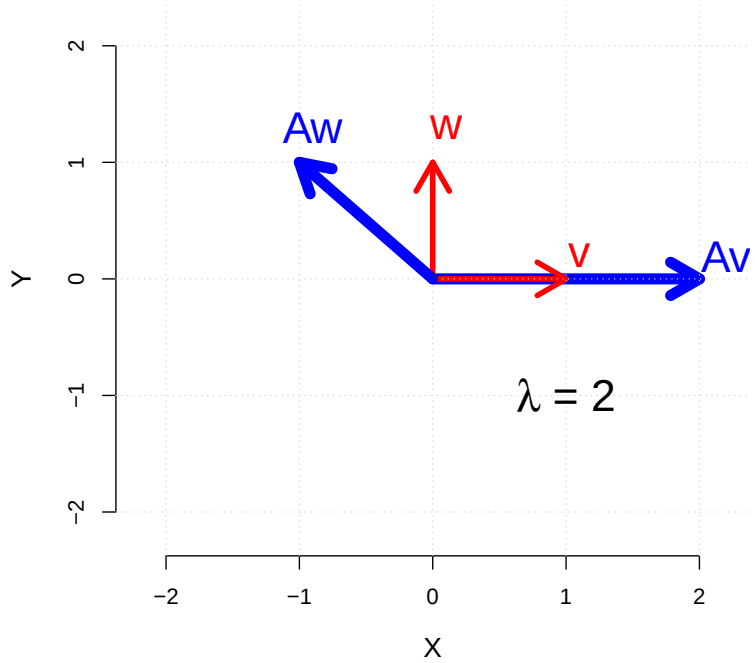


$$\boldsymbol{Av} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} -1 \\ 0 \end{bmatrix} \quad (1)$$



Eigenvalues and eigenvectors

$$\boldsymbol{Av} = \begin{bmatrix} 2 & -1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 2 \\ 0 \end{bmatrix} \quad \text{but} \quad \boldsymbol{Aw} = \begin{bmatrix} 2 & -1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$



- Here, $\boldsymbol{v}$ is an eigenvector of $\boldsymbol{A}$ with eigenvalue 2[1], but $\boldsymbol{w}$ is not an eigenvector of $\boldsymbol{A}$

[1] Corresponding to a "stretch" by a factor of 2.

Eigenvalues and eigenvectors

Definition (eigenpair)

Recall that if $\boldsymbol{A}$ is an $n \times n$ matrix, then $\boldsymbol{x}$ and λ are an eigenvector/eigenvalue pair for $\boldsymbol{A}$ if

$$\boldsymbol{A}\boldsymbol{x} = \lambda \boldsymbol{x},$$

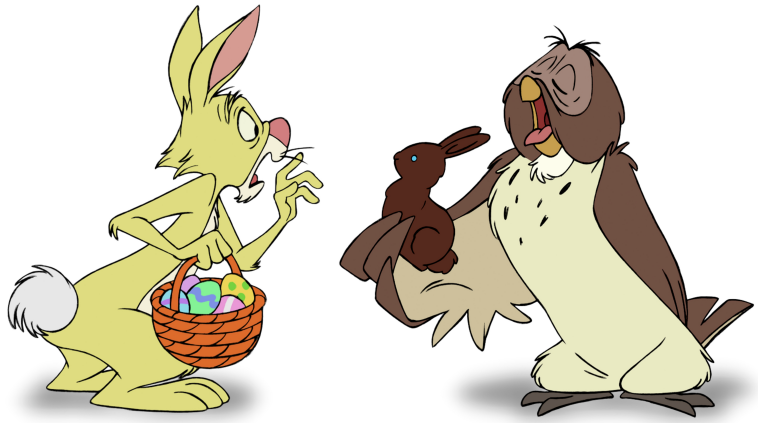
then we say that λ is an eigenvalue of $\boldsymbol{A}$ and that $\boldsymbol{x}$ is the corresponding eigenvector.

- Many texts refer to the eigenvector as $\boldsymbol{\xi}$, i.e., $\boldsymbol{A}\boldsymbol{\xi} = \lambda \boldsymbol{\xi}$

Eigenvalues and eigenvectors

One of the major uses for eigenanalysis is to decouple equations, which is related to the purpose of PCA/SVD. Therefore, we will finish this linear algebra review with an example of decoupling equations.

- Consider a population of owls and rabbits
- The rabbits breed like mad, but the more rabbits there are, the more the owls have to eat
- If the owls eat more, there will be more owls next year, which will then eat more rabbits



Eigenvalues and eigenvectors

We will use $x_1(n)$ to describe the population of owls (*in hundreds*) in year n, and $x_2(n)$ to describe that of rabbits (*in thousands*). We thus have a system of coupled equations.

$$\begin{aligned} x_1(n) &= a_{11}x_1(n-1) + a_{12}x_2(n-1) \\ x_2(n) &= a_{21}x_1(n-1) + a_{22}x_2(n-1) \end{aligned}$$

where a_{11}, a_{12}, and a_{22} are positive constants and a_{21} is a negative constant (the more owls in year $n-1$, the fewer rabbits in year n). This can be written as

$$\boldsymbol{x}(n) = \boldsymbol{A}\boldsymbol{x}(n-1) \quad \text{with} \quad \boldsymbol{A} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \quad \text{and} \quad \boldsymbol{x}(n) = \begin{bmatrix} x_1(n) \\ x_2(n) \end{bmatrix} \tag{2}$$

Eigenvalues and eigenvectors

Let us use the example

$$\boldsymbol{A} = \begin{bmatrix} 0.4 & 0.6 \\ -0.3 & 1.3 \end{bmatrix} \tag{3}$$

- Thus, in year n, there will be $x_1(n) = 0.4x_1(n-1) + 0.6x_2(n-1)$ owls
  - i.e., the more owls and the more rabbits there are in year $n-1$, the more there will be in year n.
- On the other hand, there will be $x_2(n) = -0.3x_1(n-1) + 1.3x_2(n-1)$ rabbits
  - i.e., the more owls there are in year $n-1$, the less rabbits there will be in year n, but the more rabbits there are in year $n-1$, the more there will be in year n.

Eigenvalues and eigenvectors

- Therefore, we get for the development of the owl and rabbit populations from year $n-1$ to n.

$$\begin{bmatrix} x_1(n) \\ x_2(n) \end{bmatrix} = \boldsymbol{A} \begin{bmatrix} x_1(n-1) \\ x_2(n-1) \end{bmatrix} = \begin{bmatrix} 0.4 & 0.6 \\ -0.3 & 1.3 \end{bmatrix} \begin{bmatrix} x_1(n-1) \\ x_2(n-1) \end{bmatrix} \tag{4}$$

- In general, for the development of the populations starting from some initial conditions $\boldsymbol{x}(\mathbf{0})$, we have

$$\begin{bmatrix} x_1(n) \\ x_2(n) \end{bmatrix} = \boldsymbol{A}^n \begin{bmatrix} x_1(0) \\ x_2(0) \end{bmatrix} \tag{5}$$

- But how do we solve this kind of coupled equation?

Eigenvalues and eigenvectors

We will not explain how to find eigenvalues/eigenvectors, which is standard material. Practically speaking, it is important to understand the concepts of when and why to use eigenpairs, and for larger matrices, software such as matlab or R is used to solve for the eigenvalues and eigenvectors

The matrix $\boldsymbol{A} = \begin{bmatrix} 0.4 & 0.6 \\ -0.3 & 1.3 \end{bmatrix}$ has the following eigenpairs

$$\begin{bmatrix} 0.4 & 0.6 \\ -0.3 & 1.3 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \underbrace{1}_{\lambda_1} \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} 0.4 & 0.6 \\ -0.3 & 1.3 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 1.4 \\ 0.7 \end{bmatrix} = \underbrace{0.7}_{\lambda_2} \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

Eigenvalues and eigenvectors

A square matrix $\boldsymbol{A}$ is called **diagonalizable** if there exists an invertible matrix $\boldsymbol{P}$ such that $\boldsymbol{D} = \boldsymbol{P}^{-1}\boldsymbol{A}\boldsymbol{P}$ is a diagonal matrix.

Theorem

An $n \times n$ matrix $\boldsymbol{A}$ has n linearly independent eigenvectors if and only if it can be written as $\boldsymbol{A} = \boldsymbol{P}\boldsymbol{D}\boldsymbol{P}^{-1}$, where $\boldsymbol{D}$ is a diagonal matrix. In that case, the diagonal entries of $\boldsymbol{D}$ are the eigenvalues of $\boldsymbol{A}$ and the eigenvectors of $\boldsymbol{A}$ are the columns of $\boldsymbol{P}$.

Eigenvalues and eigenvectors

- For example, using the eigenvalues and eigenvectors of our owls and rabbits matrix, we see that

$$\boldsymbol{A} = \boldsymbol{PDP}^{-1}$$

or

$$\begin{bmatrix} 0.4 & 0.6 \\ -0.3 & 1.3 \end{bmatrix} = \begin{bmatrix} 1 & 2 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 0.7 \end{bmatrix} \begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix}$$

In matlab or octave, this corresponds to the following code

```
octave:37> P=[1 2;1 1];
octave:38> D=[1 0;0 0.7];
octave:39> P*D*inv(P)
ans =

   0.40000   0.60000
  -0.30000   1.30000
```

Eigenvalues and eigenvectors

- Let us see how we can use this to solve problems in our owl/rabbit example

We have

$$\boldsymbol{x}(n) = \boldsymbol{A}\boldsymbol{x}(n-1) \tag{6}$$

Since the eigenvectors $\boldsymbol{b}_1$ and $\boldsymbol{b}_2$ are a basis for $\mathbb{R}^2$, we can express $\boldsymbol{x}$ as a linear combination of the eigenvectors

$$\boldsymbol{x}(n) = \alpha_1(n)\boldsymbol{b}_1 + \alpha_2(n)\boldsymbol{b}_2$$

for some coefficients $\alpha_1(n)$ and $\alpha_2(n)$, and analogously

$$\boldsymbol{x}(n-1) = \alpha_1(n-1)\boldsymbol{b}_1 + \alpha_2(n-1)\boldsymbol{b}_2$$

Eigenvalues and eigenvectors

- We can now re-express equation (6) in this basis

$$\begin{aligned}\alpha_1(n)\boldsymbol{b}_1 + \alpha_2(n)\boldsymbol{b}_2 &= \boldsymbol{A}\alpha_1(n-1)\boldsymbol{b}_1 + \boldsymbol{A}\alpha_2(n-1)\boldsymbol{b}_2 \\ &= \alpha_1(n-1)\boldsymbol{A}\boldsymbol{b}_1 + \alpha_2(n-1)\boldsymbol{A}\boldsymbol{b}_2 \\ &= \alpha_1(n-1)\lambda_1\boldsymbol{b}_1 + \alpha_2(n-1)\lambda_2\boldsymbol{b}_2\end{aligned}$$

where the last step follows because of $\boldsymbol{A}\boldsymbol{x} = \lambda\boldsymbol{x}$. Therefore, we have

$$\alpha_i(n) = \lambda_i\alpha_i(n-1)$$

and thus

$$\boldsymbol{x}(n) = \sum_i \lambda_i\alpha_i(n-1)\boldsymbol{b}_i$$

Eigenvalues and eigenvectors

- Let us now return the problem of solving the following equation

$$\begin{bmatrix} x_1(n) \\ x_2(n) \end{bmatrix} = \boldsymbol{A}^n \begin{bmatrix} x_1(0) \\ x_2(0) \end{bmatrix} \tag{7}$$

- Recalling that $A = \boldsymbol{PDP}^{-1}$, we conclude

$$\boldsymbol{A}^n = \underbrace{\boldsymbol{PDP}^{-1}\boldsymbol{PDP}^{-1}\ldots\boldsymbol{PDP}^{-1}}_{n \text{ times}}$$

and thus[2]

$$\boldsymbol{A}^n = \boldsymbol{P}\underbrace{\boldsymbol{DD}\ldots\boldsymbol{D}}_{n \text{ times}}\boldsymbol{P}^{-1} = \boldsymbol{PD}^n\boldsymbol{P}^{-1} \tag{8}$$

which leads to

$$\boldsymbol{x}(n) = \sum_i \lambda_i^n \alpha_i(0)\boldsymbol{b}_i$$

[2] because $\boldsymbol{PP}^{-1} = \boldsymbol{I}$.

Eigenvalues and eigenvectors

- Let's say we start with 200 owls (recall that x_1 was in units of hundreds, so we have $x_1 = 2$) and 3000 rabbits (recall that x_2 was in units of thousands, so we have $x_2 = 3$).
- Then we have that $\boldsymbol{X}(0) = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$. This initial condition now allows us to solve for the coefficients at year zero

$$\boldsymbol{x}(0) = \begin{bmatrix} 2 \\ 3 \end{bmatrix} = \alpha_1(0) \begin{bmatrix} 1 \\ 1 \end{bmatrix} + \alpha_2(0) \begin{bmatrix} 2 \\ 1 \end{bmatrix} = 4 \begin{bmatrix} 1 \\ 1 \end{bmatrix} - \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

- We can now plug the coefficients $\alpha_1(0) = 4$ and $\alpha_2(0) = -1$ into equation (7)

$$\boldsymbol{x}(n) = \boldsymbol{A}^n \boldsymbol{x}(0) = 4\,(1)^n \begin{bmatrix} 1 \\ 1 \end{bmatrix} - 1\,(0.7)^n \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

Eigenvalues and eigenvectors

- Thus we have two equations

$$\begin{aligned} x_1(n) &= 4 - 2(0.7)^n \\ x_2(n) &= 4 - (0.7)^n \end{aligned}$$

- As $n \to \infty$, we get the limiting populations of $x_1(\infty) = 4$ (i.e., 400) owls and $x_2(\infty) = 4$ (i.e., 4000) rabbits.
- Thus, expressing coupled equations using an eigenvector basis has allowed us to decouple a system of coupled equations.

Outline

Symmetric real matrices

Symmetric real matrices have a number of interesting properties that allow special kinds of matrix decompositions and other algorithms

- A **symmetric matrix** is a square matrix that is equal to its transpose, i.e., $a_{ij} = a_{ji}$ for all i and j.

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & e & 6 & 9 \\ 3 & 6 & 2 & \pi \\ 4 & 9 & \pi & 1 \end{bmatrix} = A^T$$

Orthogonal matrices

- An orthogonal matrix is a square matrix with real entries whose columns and rows are orthogonal unit vectors:

$$\boldsymbol{q}_i^T \boldsymbol{q}_j = 0 \quad \text{for} \quad i \neq j$$

and

$$\|\boldsymbol{q}_i\| = 1 \quad \forall i$$

- That is, the individual columns of an orthogonal matrix are orthogonal to one another and the length of the vectors is one.
- Note that a matrix $\boldsymbol{Q}$ is orthogonal if its transpose is equal to its inverse:

$$\boldsymbol{Q}^T = \boldsymbol{Q}^{-1}$$

this entails

$$\boldsymbol{Q}\boldsymbol{Q}^T = \boldsymbol{Q}\boldsymbol{Q}^{-1} = \boldsymbol{I}$$

Spectral theorem

Theorem (Spectral theorem)

Any symmetric matrix whose values are real can be diagonalized by an orthogonal matrix. In other words, if $\boldsymbol{A}$ *is a symmetric, real-valued matrix, then there exists a real orthogonal matrix* $\boldsymbol{Q}$ *such that*

$$\Lambda = \boldsymbol{Q}^T \boldsymbol{A} \boldsymbol{Q}$$

- In other words, a matrix $\boldsymbol{A}$ is symmetric $\iff$ $\boldsymbol{A}$ has an orthonormal basis of eigenvectors.
- $\boldsymbol{Q}\Lambda = \boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{A}\boldsymbol{Q} = \boldsymbol{A}\boldsymbol{Q} \rightarrow \boldsymbol{q}_i\lambda_i = \boldsymbol{A}\boldsymbol{q}_i$

Matrix Decompositions

- The spectral theorem entails that a symmetric real-valued matrix $\boldsymbol{A}$ can be decomposed using its eigenpairs:

$$\mathbf{A} = \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^T$$

- Noting that the columns of $\boldsymbol{Q}$ are made up of the eigenvectors $\boldsymbol{q}_i$, we have

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{q}_1 & \boldsymbol{q}_2 & \dots & \boldsymbol{q}_n \end{bmatrix} \begin{bmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \dots & \\ & & & \lambda_n \end{bmatrix} \begin{bmatrix} \boldsymbol{q}_1^T \\ \boldsymbol{q}_2^T \\ \dots \\ \boldsymbol{q}_n^T \end{bmatrix} \tag{9}$$

This implies

$$\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^T = \sum_{i=1}^{n} \lambda_i \boldsymbol{q}_i \boldsymbol{q}_i^T$$

Outline

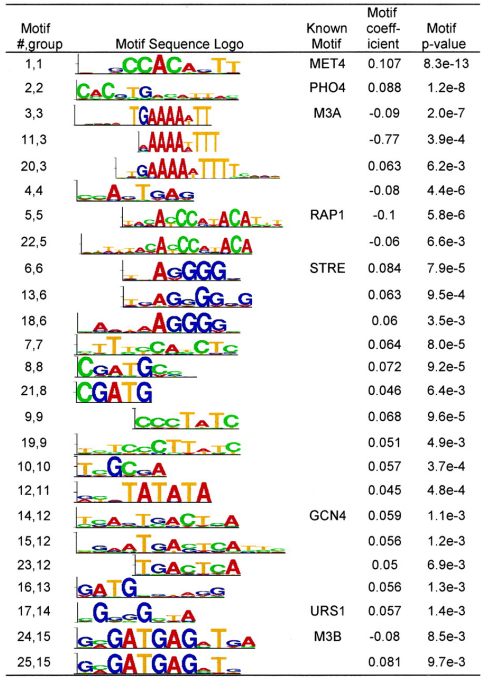
Back to Gene Regulation

Let us now see how these concepts can be brought to bear on the problem of gene regulation in ChIP-seq experiments

- Previous state of the art: Modeling based on linear regression used to predict gene expression
- For instance, predicted TFBS affinity

$$Y_g = \alpha + \sum_{m=1}^{M} \beta_m S_{mg} + \epsilon_g \qquad (10)$$

Conlon EM et al. (2003) Integrating regulatory motif discovery and genome-wide expression analysis. *PNAS* **100**:3339–44.



Motif #,group	Motif Sequence Logo	Known Motif	Motif coeff-icient	Motif p-value
1,1		MET4	0.107	8.3e-13
2,2		PHO4	0.088	1.2e-8
3,3		M3A	-0.09	2.0e-7
11,3			-0.77	3.9e-4
20,3			0.063	6.2e-3
4,4			-0.08	4.4e-6
5,5		RAP1	-0.1	5.8e-6
22,5			-0.06	6.6e-3
6,6		STRE	0.084	7.9e-5
13,6			0.063	9.5e-4
18,6			0.06	3.5e-3
7,7			0.064	8.0e-5
8,8			0.072	9.2e-5
21,8			0.046	6.4e-3
9,9			0.068	9.6e-5
19,9			0.051	4.9e-3
10,10			0.057	3.7e-4
12,11			0.045	4.8e-4
14,12		GCN4	0.059	1.1e-3
15,12			0.056	1.2e-3
23,12			0.05	6.9e-3
16,13			0.056	1.3e-3
17,14		URS1	0.057	1.4e-3
24,15		M3B	-0.08	8.5e-3
25,15			0.081	9.7e-3

Predicting Gene Regulation

However, thus far, the fraction of variation in gene expression (R^2) explained by TF binding has been very moderate, varying between 9.6% and 36.9% on various datasets from yeast to human

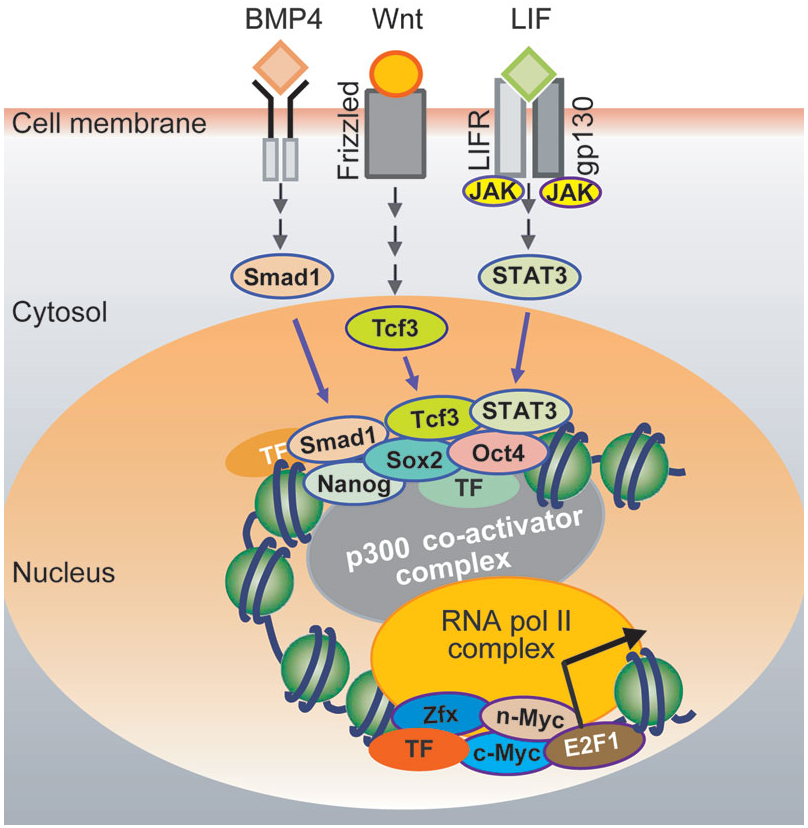
Potential reasons include

- Insufficient data
- suboptimal models
- both.

The authors of Ouyang et al propose a new way to **extract suitable features from the ChIP-Seq data** to serve as explanatory variables in the modeling of gene expression. Additionally, they use SVD/PCA to better model divergent regulatory effects of a TF that may be due to differences in the binding of cofactors and/or the chromatin context.

Embryonic Stem Cells

Transcriptional networks in embryonic stem cells (ESC) maintain self-renewal and pluripotency. Many TFs have been identified as critical in ESCs, among them Oct4, Nanog, and Sox2.



Embryonic Stem Cells

A quantitative dissection of the functional roles of ESC regulators such as Oct4, Nanog, and Sox2 is still lacking.

- Goal of experiment: Use ChIP-seq data from 12 ESC factors[3] and RNA-seq data to perform an analysis of genome-wide gene expression and TF binding data in ESCs.

[3] Smad1, Stat3, Sox2, Oct4, Nanog, Esrrb, Tcfcp2l1, Klf4, Zfx, E2f1, Myc, and Mycn

Transcription Factor Association Strength (TFAS)

Definition (Transcription Factor Association Strength)

The TFAS is a non-observable quantity that reflects the degree to which a transcription factor binds to the regulatory sequences of a gene and thereby stimulates gene expression

- There are innumerable definitions of TFAS or analogous quantities in the literature
- The set of TFAS of all TFs for all Genes can be used for example in network inference algorithms

Binary TFAS

Traditionally, a TF binding peak is usually associated with the nearest gene (uslly based on the distance between the midpoint of the peak and the transcription start site (TSS).

- Denoting the binary TFAS as a_{ij}, then $a_{ij} = 1$ if gene i is associated with a ChIP-seq peak of TF j; otherwise $a_{ij} = 0$.
- A binary TFAS is easy to calculate
- The binary TFAS approach does not take into account the intensity of the peaks and the relative distance between peaks and genes.

Continuous TFAS

Ouyang et al. introduce a continuous TFAS that integrates the peak intensity and the proximity to genes to define the association strength between a TF and a gene.

- It is assumed that the association strength of TF j on gene i is a weighted sum of intensities of all of the peaks of TF j:

$$a_{ij} = \sum_k g_k e^{-\frac{d_k}{d_0}} \tag{11}$$

In this equation,

- g_k is the height of the k^{th} binding peak of the TF j
- d_k is the distance in nucleotides from the k^{th} binding peak and the TSS of gene i
- d_0 is a TF-specific constant (500 nt for E2f1 and 5000 nt for other TFs because E2f1 peaks tend to be close the the TSS)

Continuous TFAS

- When $\frac{d_k}{d_0}$ is very large the contribution of the peak will be effectively zero.
- Therefore, the summation is taken over peaks that are not too far away from the TSS (e.g., $\leq 1 \times 10^6$ nucleotides)
- The TFAS values are then log-transformed[4] and quantile normalized[5]
- For N genes and M TFs, the TFAS profiles are stored in an $N \times M$ matrix $\boldsymbol{A}$.

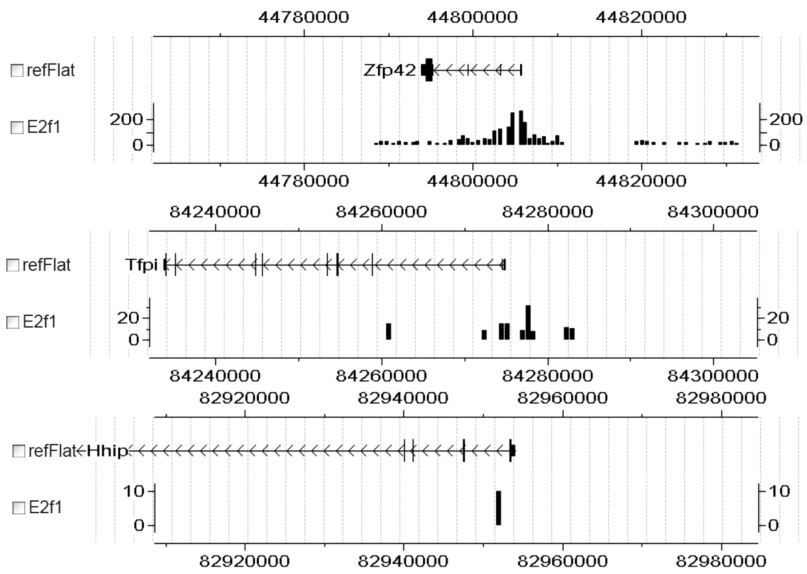
[4] i.e., $a'_{ij} = \log a_{ij}$

[5] i.e., the a'_{ij} are sorted; then, the same number of samples from the reference distribution (e.g., Gaussian) are taken from the cumulative distribution function, and the a'_{ij} are assigned the values of the reference distribution.

Continuous TFAS

Illustration of the binding peaks of E2f1 around three genes. The vertical axis represents the amplitude of the ChIP-Seq signals.

- Zpf42: TFAS=324
- Tfpi: TFAS=19.3
- Hhip: TFAS=0.1



- Note that binary TFAS would have assigned a "1" to all three binding events

Continuous TFAS

- The continuous TFAS was the first major new idea of the paper.
- The authors now validate the utility of continuous TFAS by comparing its performance to that of binary TFAS
- They use a principle-components analysis (PCA) regression model to compare the ability of the binding peaks of the 12 ESC transcription factors with respect to their ability to predict the expression of genes in ESCs (as measured by RNA-seq).
- By examining the quality of the respective regression models, we can determine which method performed best
- To understand this, we will have to review PCA, and how all of this is used to perform regression.

Outline

PCA: Intuition, Goals, Algorithm

We will now present an explanation of the PCA algorithm that is closely based on the document *A Tutorial on Principal Component Analysis* by Jonathon Shlens[a]

[a] Available at http://www.snl.salk.edu/ shlens/

- PCA 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** (PC). The first PC accounts for as much of the variability in the data as possible, and each succeeding PC accounts for as much of the remaining variability as possible.
- An extremely important tool in the repertoire of algorithms for data analysis in bioinformatics

PCA: Intuition, Goals, Algorithm

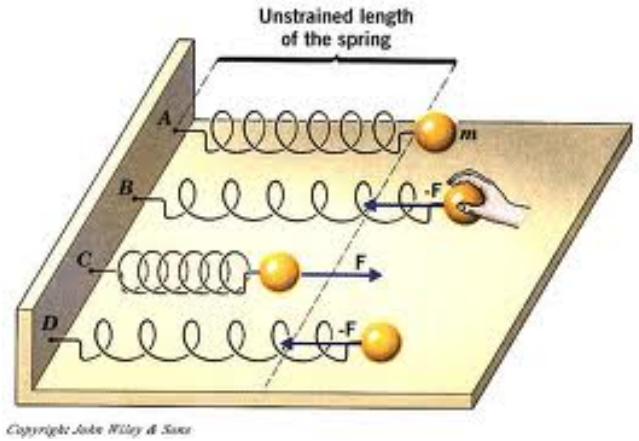
A common problem in bioinformatics

We are trying to understand a complicated biological experiment with lots of genomics data that comes from multiple sources (e.g., ChIP-seq from 12 TFs, RNA-seq data), is noisy, and is partially redundant

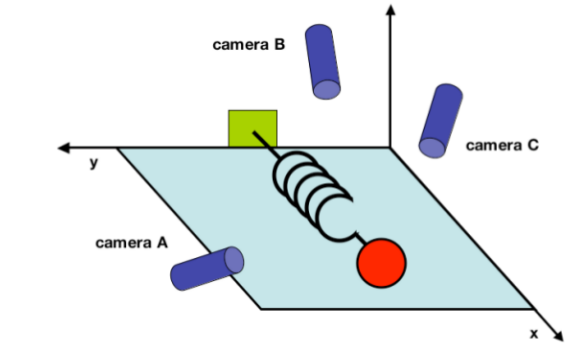
- We want to understand the essential patterns in the data
- We will demonstrate this using a slightly simpler example, and then explain the relevance to the ESC experiment

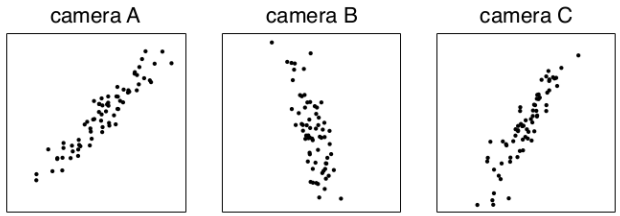
The clueless physicist

Let us imagine we are studying the motion of an ideal spring, consisting of a ball attached to a massless, frictionless spring. The ball is released a small distance away from equilibrium; because it is an ideal spring, it should oscillate indefinitely along its axis of motion.



The clueless physicist





Graphic: Jonathon Shlens

- Let's say we want to determine the motion of the spring as a function of time
- We therefore place three movie cameras around the spring and record images at 120 Hz

The clueless physicist

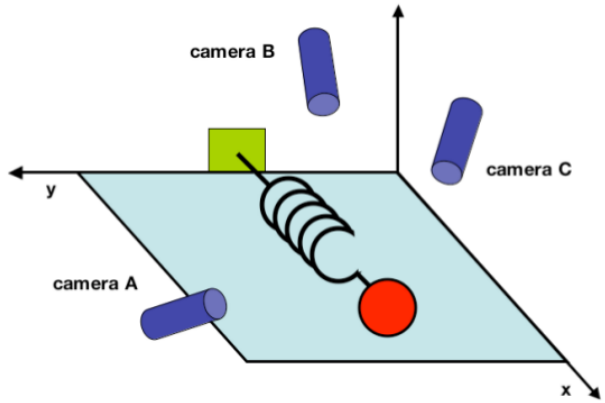
- Our goal: get to a **simple equation** that will describe the dynamics of the system in terms of a single variable x
- But how do we get from our data from the three cameras to this equation?
- In the real world, we do not know which which measurements best reflect the dynamics of the system in question[6]
- Also, there is typically an (unknown) amount of noise in any experimental system that will make our task of recognizing patterns in the data even harder. For instance, friction or poorly focused cameras might interfere with the experiment with the spring

[6] e.g., e do not know a priori which, if any, of the 12 ESC transcription factors will affect the expression of any of the 20,000 genes measured by RNA-seq.

The goals of PCA

Intuitively, the goal of PCA is to identify the most meaningful basis with which to re-express a dataset, in the hope that the new basis will (1) filter out noise and (2) reveal hidden structure.

- Let us continue with our example of the spring
- Clearly, we hope that the method will determine that $\hat{\boldsymbol{x}}$, i.e., the unit basis vector along the x axis, is the important dimension (rather than the clueless axes defined by the three cameras)



Graphic: Jonathon Shlens

The clueless physicist

Now let us see how to use PCA to help us understand the data. Each of the three cameras A, B, and C takes a measurement of the 2-dimensional projection of the ball 120 times a second. For instance, camera A records x_A and y_A.

- One sample (one data measurement) consists of the data from all three cameras

$$\boldsymbol{x} = \begin{bmatrix} x_A \\ y_A \\ x_B \\ y_B \\ x_C \\ y_C \end{bmatrix}$$

- Thus, if we record the ball's position for 100 seconds, we will have $100 \times 120 = 12,000$ of these vectors
- In our ESC example, we essentially have a vector of 12 data points from the ChIP-seq experiments, and we have 20,000 such vectors, one for each gene.

The naive basis

Let us for the moment concentrate on the data sampled by camera A. Each of the measurement vectors represents a linear combination of the unit length basis vectors. The standard naive basis would be $\{\boldsymbol{e}_1, \boldsymbol{e}_2\} = \{(1,0),(0,1)\}$.

- For instance, if camera A records the position $(x_A, y_A) = (2,2)$, this can be expressed as the linear combination

$$2\boldsymbol{e}_1 + 2\boldsymbol{e}_2 = 2\begin{bmatrix}1\\0\end{bmatrix} + 2\begin{bmatrix}0\\1\end{bmatrix}$$

But why select this basis over another one, e.g.

$$2\sqrt{2}\boldsymbol{b}_1' + 0\boldsymbol{b}_2' = 2\sqrt{2}\begin{bmatrix}\frac{2}{\sqrt{2}}\\ \frac{2}{\sqrt{2}}\end{bmatrix} + 0\begin{bmatrix}\frac{2}{\sqrt{2}}\\ -\frac{2}{\sqrt{2}}\end{bmatrix}$$

The naive basis

- Essentially, we use the standard naive basis of $\{\boldsymbol{e}_1, \boldsymbol{e}_2\} = \{(1,0),(0,1)\}$ because this is the way we originally recorded our data (these are the numbers we got out of the camera).
- There is nothing special about this basis, it is just the starting point for most data analysis
- For the 6-dimensional data of the spring experiment, the naive basis can be expressed as a matrix, each row of which is an orthonormal basis vector

$$\boldsymbol{B} = \begin{bmatrix} \boldsymbol{e}_1^T \\ \boldsymbol{e}_2^T \\ \boldsymbol{e}_3^T \\ \boldsymbol{e}_4^T \\ \boldsymbol{e}_5^T \\ \boldsymbol{e}_6^T \end{bmatrix} = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{bmatrix} = \boldsymbol{I} \tag{12}$$

PCA: A more useful basis?

PCA searches for a new basis that is a linear combination of the original basis and that best re-expresses the data set.

- Let $\boldsymbol{X}$ be the original dataset, where each column represents one m-dimensional vector with a single measurement. In the example, $m = 6$ measurements, and there are $n = 12,000$ measurements (one to a column). Thus, $\boldsymbol{X}$ is a $6 \times 12,000$ matrix.
- Now let $\boldsymbol{Y}$ be a new $m \times n$ matrix that is produced from $\boldsymbol{X}$ by means of a linear transformation by a matrix $\boldsymbol{P}$[7]

$$\boldsymbol{Y} = \boldsymbol{P}\boldsymbol{X} \tag{13}$$

Note of course that if $\boldsymbol{P} = \boldsymbol{I}$, then $\boldsymbol{Y} = \boldsymbol{X}$.

[7] At this point, we still have not stated how to find $\boldsymbol{P}$.

PCA: A more useful basis?

We will define the following quantities surrounding $\boldsymbol{Y} = \boldsymbol{PX}$

- $\boldsymbol{p}_i$ are the **rows** of $\boldsymbol{P}$.
- $\boldsymbol{x}_i$ are the columns of $\boldsymbol{X}$, representing the individual measurements
- $\boldsymbol{y}_i$ are the columns of $\boldsymbol{Y}$
- Note that $\boldsymbol{P}$ is a matrix that performs a linear transformation of $\boldsymbol{X}$ into $\boldsymbol{Y}$ (rotation and stretch)

PCA: A more useful basis?

It can be seen that the rows of $\boldsymbol{P}$ are thus a new set of basis vectors for expressing the columns of $\boldsymbol{X}$.

$$\boldsymbol{PX} = \begin{bmatrix} \boldsymbol{p}_1 \\ \vdots \\ \boldsymbol{p}_m \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_1 & \dots & \boldsymbol{x}_n \end{bmatrix}$$

and thus

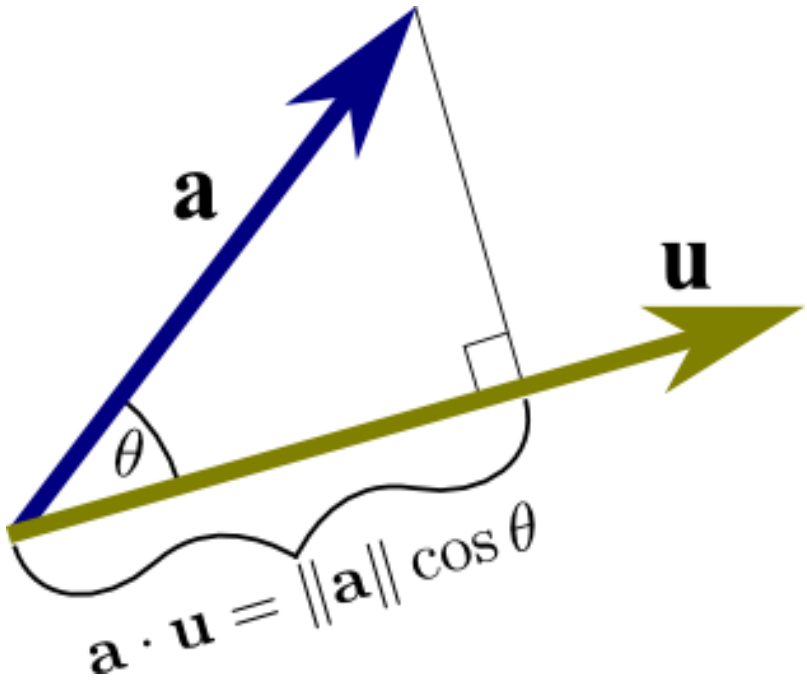
$$\boldsymbol{Y} = \begin{bmatrix} \boldsymbol{p}_1 \cdot \boldsymbol{x}_1 & \dots & \boldsymbol{p}_1 \cdot \boldsymbol{x}_n \\ \vdots & \ddots & \vdots \\ \boldsymbol{p}_m \cdot \boldsymbol{x}_1 & \dots & \boldsymbol{p}_m \cdot \boldsymbol{x}_n \end{bmatrix}$$

PCA: A more useful basis?

Column i of $\boldsymbol{Y}$ is thus the dot product of column i of $\boldsymbol{X}$ with the corresponding rows of $\boldsymbol{P}$:

- The j^{th} coefficient of $\boldsymbol{y}_i$ is a projection of $\boldsymbol{x}_i$ onto the j^{th} row of $\boldsymbol{P}$.

$$\boldsymbol{y}_i = \begin{bmatrix} \boldsymbol{p}_1 \cdot \boldsymbol{x}_i \\ \vdots \\ \boldsymbol{p}_m \cdot \boldsymbol{x}_i \end{bmatrix}$$



PCA: A more useful basis?

We have left out the question of how exactly to find the matrix $\boldsymbol{P}$? The PCA procedures is based upon features that are considered desirable for the matrix $\boldsymbol{Y}$ to exhibit, which we will consider next.

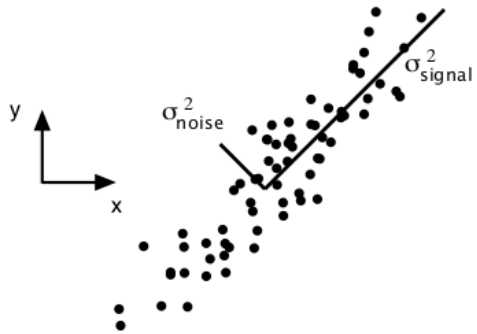
There are two essential topics

- Noise
- Redundancy

Noise

Noise is quantified relative to signal strength. A common measure is the signal to noise ratio:

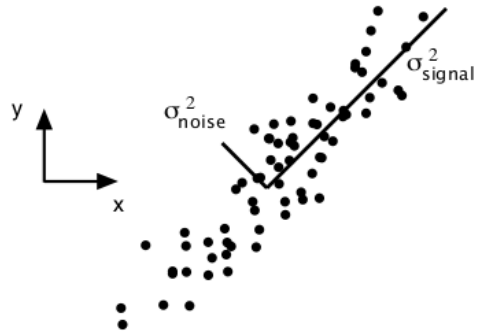
$$SNR = \frac{\sigma^2_{signal}}{\sigma^2_{noise}}$$



- In general, directions with the largest variance correspond to the interesting signal
- Here, σ^2_{signal} is along the straight line traced out by the spring. Any spread deviating from this line is noise, captured here by σ^2_{noise}

Redundancy

- If we could some how rotate the basis to align a basis vector with the direction of maximum variance, we could essentially capture all of the interesting signal in the spring experiment with a single variable instead of 6



Graphic: Jonathon Shlens

- In real life, data can be highly **intercorrelated**, and appropriate dimensionality reduction may be not only intuitive but also improve the performance of downstream statistical tests.

Covariance matrix

A covariance matrix, usually denoted Σ, generalizes the notion of variance to multiple dimensions. Element (i, j) represents the covariance between the i^{th} and j^{th} elements of a vector of random variables.

- Recall that the Variance of a random variable is defined as $\mathrm{Var}(X) = \mathbb{E}\left[(X - \mu)^2\right]$
- e.g., for a discrete with equally probable elements, we have $\mathrm{Var}(X) = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2$.
- The covariance for random variables that are arranged as a column vector $\boldsymbol{X} = \begin{bmatrix} X_1 \\ X_2 \\ \dots \\ X_n \end{bmatrix}$ is then a $n \times n$ matrix $\boldsymbol{\Sigma}$ with

$$\Sigma_{ij} = \mathbb{E}[(X_i - \mu_i)(X_j - \mu_j)]$$

Covariance matrix

Consider two row vectors:

$$\begin{aligned} \boldsymbol{a} &= \begin{bmatrix} a_1 & a_2 & \dots & a_n \end{bmatrix} \text{ and} \\ \boldsymbol{b} &= \begin{bmatrix} b_1 & b_2 & \dots & b_n \end{bmatrix} \end{aligned}$$

We can express their covariance as

$$\sigma^2_{ab} = \frac{1}{n}\boldsymbol{a}\boldsymbol{b}^T$$

Define a new $m \times n$ matrix $\boldsymbol{X}$ whose rows correspond to the measurements, and whose columns corresponding to the components of the **centered** individual measurements (e.g., x_A, y_A). In our example, $\boldsymbol{X}$ has 10,000 rows and 6 columns. The covariance matrix is:

$$\boldsymbol{C_X} = \frac{1}{n}\boldsymbol{X}\boldsymbol{X}^T$$

Covariance matrix

Some important points about covariance matrices

- They are square **symmetric** matrices (clearly, $\Sigma_{ij} = \mathbb{E}[(X_i - \mu_i)(X_j - \mu_j)] = \mathbb{E}[(X_j - \mu_j)(X_i - \mu_i)] = \Sigma_{ji}$)
- The diagonal terms of C_X represent the **variance** of the individual measurement types.
- The off-diagonal terms represent the **covariance** between the individual measurement types.

Thus, to **maximize the signal to noise ratio**, we want to have large values for the diagonal terms, and to **minimize redundancy** we want to have small values for the off-diagonal terms.

PCA: The algorithm

The PCA algorithm can now be understood at a bird's eye level as follows:

Algorithm 1 PCA

1: Select $\boldsymbol{p}_1$, a direction in m-dimensional space along which $\mathrm{var}(X)$ is maximized.
2: Find $\boldsymbol{p}_2$ which maximizes $\mathrm{var}(X)$ s.t. $\boldsymbol{p}_1\boldsymbol{p}_2^T = 0$
3: **repeat**
4: In iteration i, identify a vector $\boldsymbol{p}_i$ that maximizes $\mathrm{var}(X)$ s.t. $\boldsymbol{p}_i\boldsymbol{p}_j^T = 0$ for all $j < i$
5: **until** m PCs are selected

PCA: The algorithm

The goal of PCA is thus: Find an orthonormal matrix $\boldsymbol{P}$ with $\boldsymbol{Y} = \boldsymbol{PX}$ such that the covariance matrix of $\boldsymbol{Y}$ is a diagonal matrix.

- There are many ways of solving PCA, including SVD[8]

That is, we want to find a matrix $\boldsymbol{P}$ such that $\boldsymbol{C_Y} = \frac{1}{n}\boldsymbol{YY}^T$ is diagonal. The rows of $\boldsymbol{P}$ are known as the principle components of $\boldsymbol{X}$.

[8] Which has advantages including numerical stability over the method presented here and is often used in practice.

PCA: The algorithm

- Goal: Find an orthonormal matrix $\boldsymbol{P}$ with $\boldsymbol{Y} = \boldsymbol{PX}$ such that such that $\boldsymbol{C_Y} = \frac{1}{n}\boldsymbol{YY}^T$ is diagonal

$$
\begin{aligned}
\boldsymbol{C_Y} &= \frac{1}{n}\boldsymbol{YY}^T \\
&= \frac{1}{n}(\boldsymbol{PX})(\boldsymbol{PX})^T \\
&= \frac{1}{n}\boldsymbol{PXX}^T\boldsymbol{P}^T \\
&= \boldsymbol{P}\left(\frac{1}{n}\boldsymbol{XX}^T\right)\boldsymbol{P}^T \\
&= \boldsymbol{PC_X P}^T
\end{aligned}
$$

- Thus, $\boldsymbol{C_Y}$ is related to the covariance matrix of $\boldsymbol{X}$

PCA: The algorithm

- Recall from theorem (3) that a symmetric matrix $\boldsymbol{A}$ (such as $\boldsymbol{C}_X$) has an orthonormal basis of eigenvectors such at $\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^T$
- For PCA, the trick is to select the matrix $\boldsymbol{P}$ to be a matrix whose rows $\boldsymbol{p}_i$ are the eigenvectors of $\boldsymbol{C_X} = \frac{1}{n}\boldsymbol{X}\boldsymbol{X}^T$, which implies that $\boldsymbol{P} = \boldsymbol{Q}^T$.

$$\begin{aligned} \boldsymbol{C_Y} &= \boldsymbol{P}\boldsymbol{C}_X\boldsymbol{P}^T \\ &= \boldsymbol{P}(\boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^T)\boldsymbol{P}^T \\ &= \boldsymbol{P}(\boldsymbol{P}^T\boldsymbol{\Lambda}\boldsymbol{P})\boldsymbol{P}^T \\ &= \boldsymbol{\Lambda} \end{aligned}$$

- It is clear that our choice of $\boldsymbol{P}$ diagonalizes $\boldsymbol{C_Y}$, which was our goal for PCA!

PCA: The algorithm

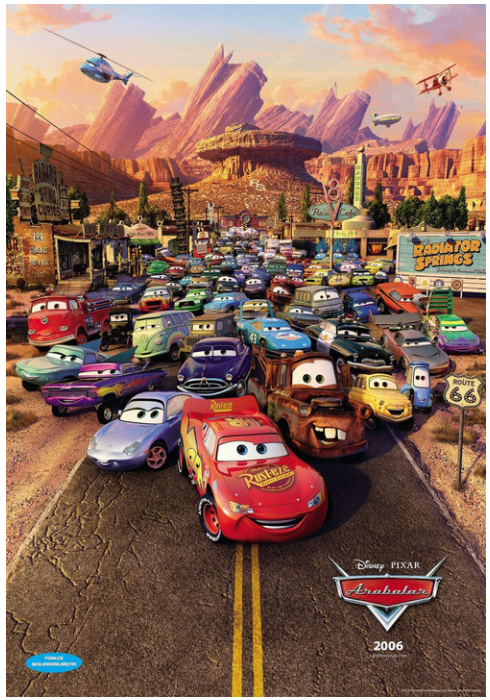
So that's it. The PCA algorithm entails

1. Subtract the mean of each measurement type
2. Compute the eigenvectors of $\boldsymbol{C_X}$.
3. The principle components (PCs) of $\boldsymbol{X}$ are the eigenvectors of $\boldsymbol{C_X} = \frac{1}{n}\boldsymbol{X}\boldsymbol{X}^T$
4. The i^{th} diagonal value of $\boldsymbol{C_Y}$ is the variance of $\boldsymbol{X}$ along $\boldsymbol{p}_i$.

PCA: Application

To give intuition about the PCA, we will show how it is used to examine and visualize a dataset about cars. Specifications are given for 428 new vehicles for the 2004 year. The variables recorded include price, measurements relating to the size of the vehicle, and fuel efficiency.

- Suggested Retail Price
- Dealer Cost
- Engine Size
- Number of Cylinders
- Horsepower
- City Miles Per Gallon
- Highway Miles Per Gallon
- Weight (Pounds)
- Wheel Base (inches)
- Length (inches)
- Width (inches)



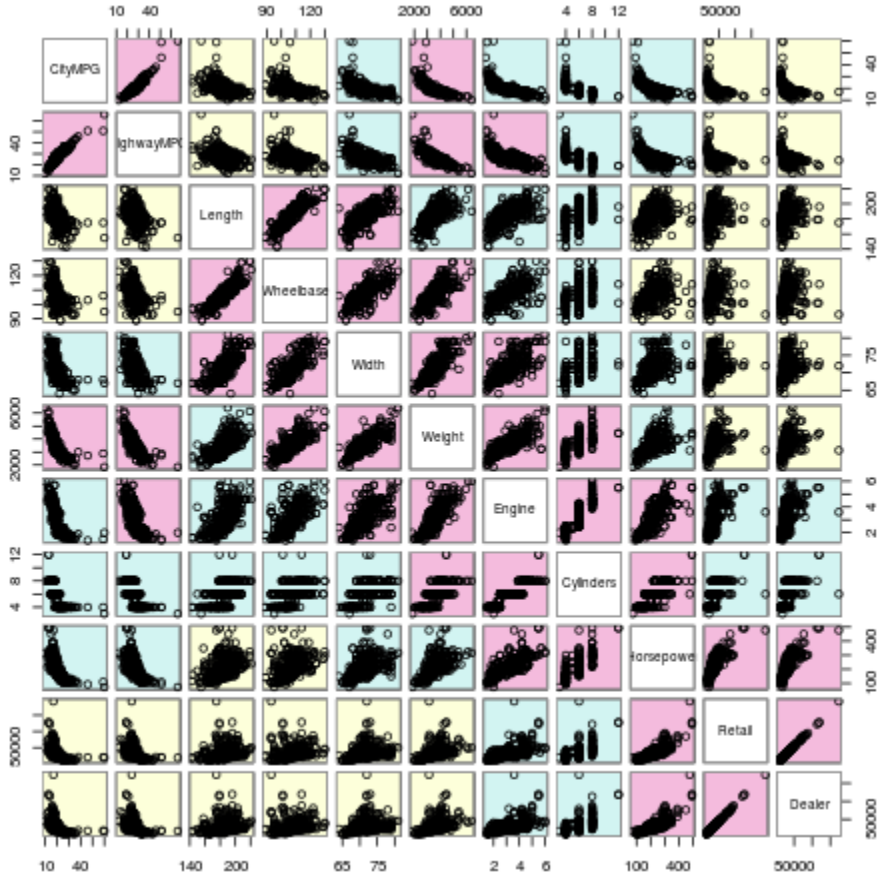
PCA: Application

The next several slides were adapted from a script by Cosma Shalizi at Carnegie Mellon University[9]

```
> cars = read.csv("cars-fixed04.dat")
> head(cars[,8:18])
                      Retail Dealer Engine Cylinders Horsepower CityMPG
Acura 3.5 RL           43755  39014    3.5         6        225      18
Acura 3.5 RL Navigation  46100  41100    3.5         6        225      18
Acura MDX              36945  33337    3.5         6        265      17
Acura NSX S            89765  79978    3.2         6        290      17
Acura RSX              23820  21761    2.0         4        200      24
Acura TL               33195  30299    3.2         6        270      20
                      HighwayMPG Weight Wheelbase Length Width
Acura 3.5 RL                  24   3880       115    197    72
Acura 3.5 RL Navigation       24   3893       115    197    72
Acura MDX                     23   4451       106    189    77
Acura NSX S                   24   3153       100    174    71
Acura RSX                     31   2778       101    172    68
Acura TL                      28   3575       108    186    72
```

[9] Data file available at http://www.stat.cmu.edu/ cshalizi/490/pca/cars-fixed04.dat

PCA: Application



There are complex correlations between different attributes of the cars. (Red: highly correlated, blue: so-so, yellow: low correlation)

PCA: Application

The R function `prcomp` performs PCA via SVD.

```
> cars.pca = prcomp(cars[,8:18],
        scale.=TRUE)
> round(cars.pca$rotation[,1:2],2)
              PC1   PC2
Retail      -0.26 -0.47
Dealer      -0.26 -0.47
Engine      -0.35  0.02
Cylinders   -0.33 -0.08
Horsepower  -0.32 -0.29
CityMPG      0.31  0.00
HighwayMPG   0.31  0.01
Weight      -0.34  0.17
Wheelbase   -0.27  0.42
Length      -0.26  0.41
Width       -0.30  0.31
```

PC1

- All the variables except the gas-mileages have a negative projection on to the first component. This means that there is a negative correlation between mileage and everything else. The first principal component tells us about whether we are getting a big, expensive gas-guzzling car with a powerful engine, or whether we are getting a small, cheap, fuel-efficient car with a wimpy engine.

PCA: Application

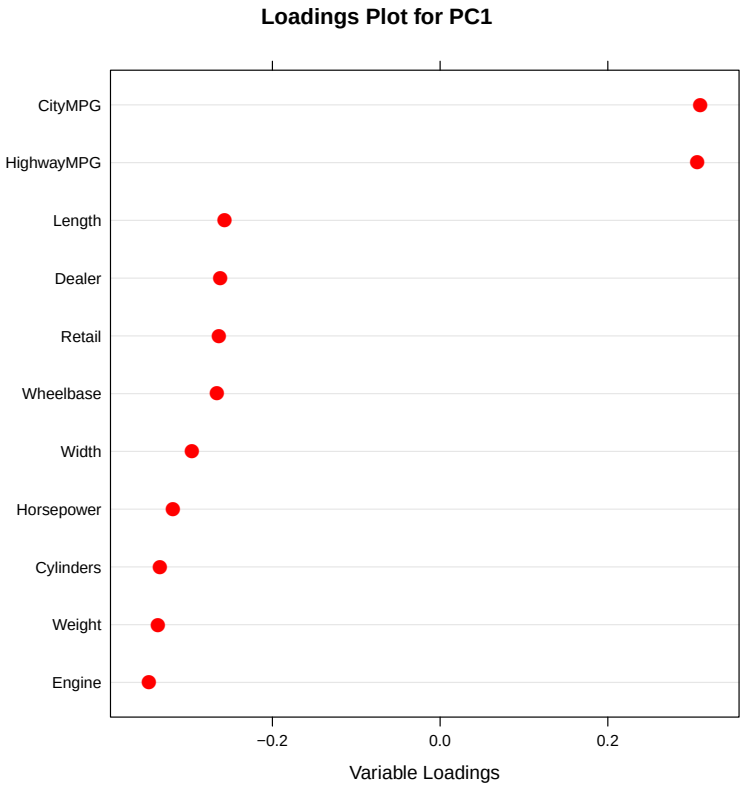

```
> round(cars.pca$rotation[,1:2],2)
              PC1   PC2
Retail      -0.26 -0.47
Dealer      -0.26 -0.47
Engine      -0.35  0.02
Cylinders   -0.33 -0.08
Horsepower  -0.32 -0.29
CityMPG      0.31  0.00
HighwayMPG   0.31  0.01
Weight      -0.34  0.17
Wheelbase   -0.27  0.42
Length      -0.26  0.41
Width       -0.30  0.31
```

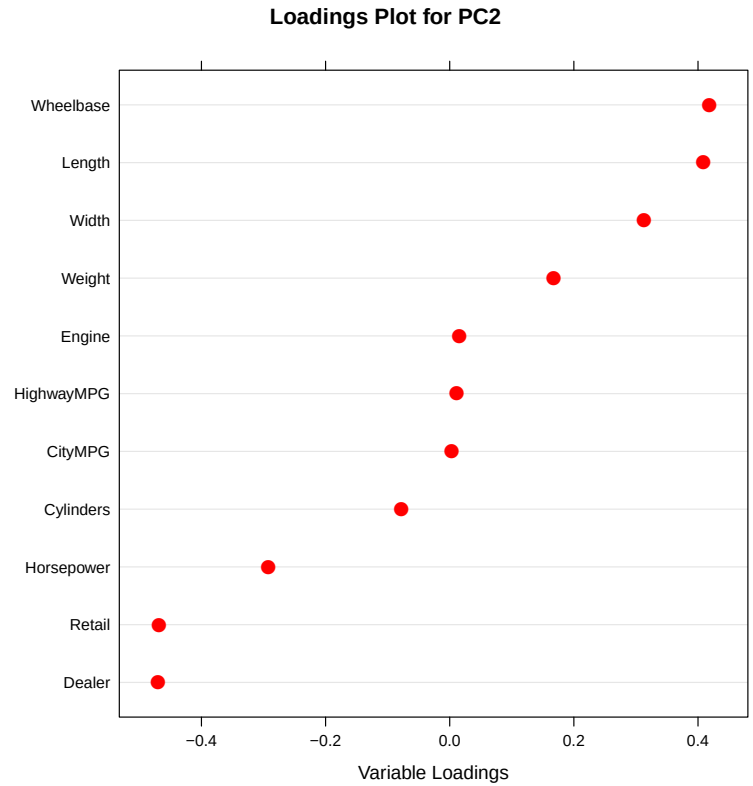
Note: MPG=miles per gallon

PC2

- Engine size and gas mileage hardly project on to PC2 at all. Instead we have a contrast between the physical size of the car (positive projection) and the price and horsepower. This axis separates mini-vans, trucks and SUVs (big, not so expensive, not so much horse-power) from sports-cars (small, expensive, lots of horse-power).

PCA: Application





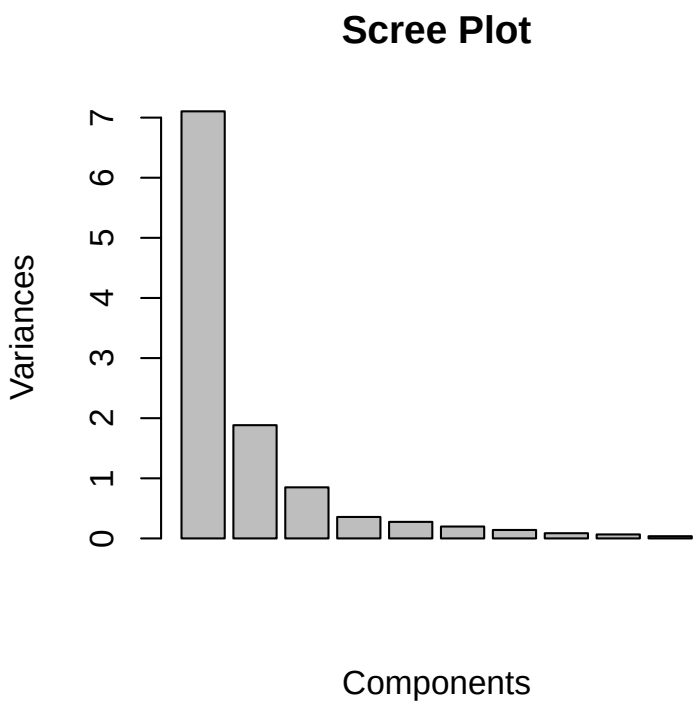
- The elements of an eigenvector are the weights p_{ij}, and are also known as loadings[10].
- The figures show the **loadings** of $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$, i.e., the coefficients representing the linear combinations of the original variables to together make up the eigenvectors

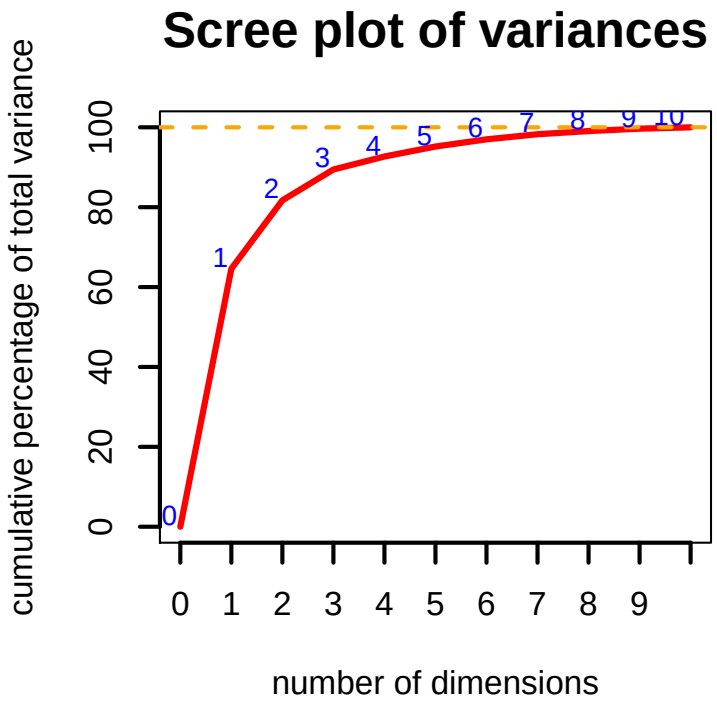
[10] loadings are called rotations in some texts

PCA: Application

How many principles components are required to represent the essential parts of the data? This can be estimated by a **scree plot**.

```
> screeplot(cars.pca,main="Scree Plot",xlab="Components")
```

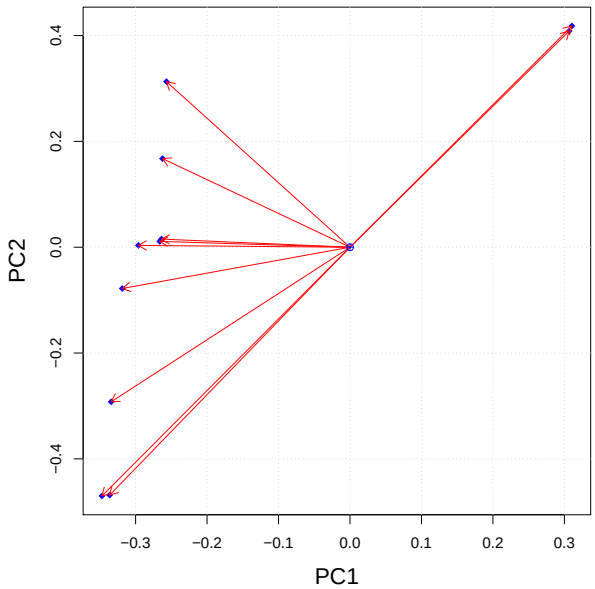




PCA: Understanding the biplot: Loadings

The biplot is often used to display the results of PCA. Biplots show both the loadings and the scores in a single plot. Let us first examine each component separately.

```
load = cars.pca$rotation
PC1 = load[order(load[,1]),1]
PC2 = load[order(load[,2]),2]
plot(PC1,PC2,pch=18,col="blue",cex.lab=1.5)
grid()
n<-length(PC1)
arrows(rep(0,n),rep(0,n),PC1,PC2,length=0.1,col="red")
points(0,0,pch=10,col="blue")
```



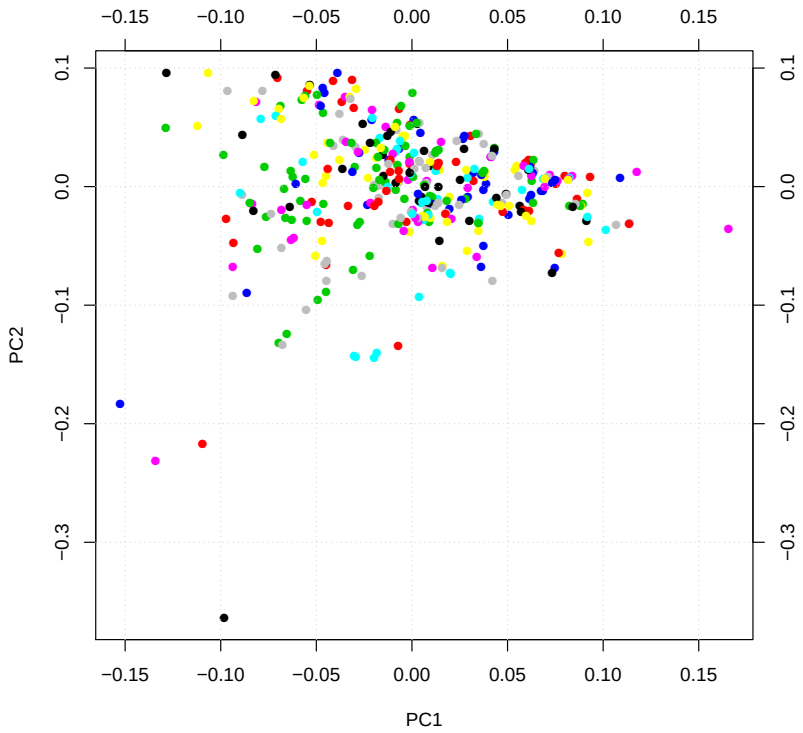
- Each point consists of the loadings for PC1 and PC2 for one coeeficient, e.f., price or miles-per-gallon

PCA: Understanding the biplot: Scores

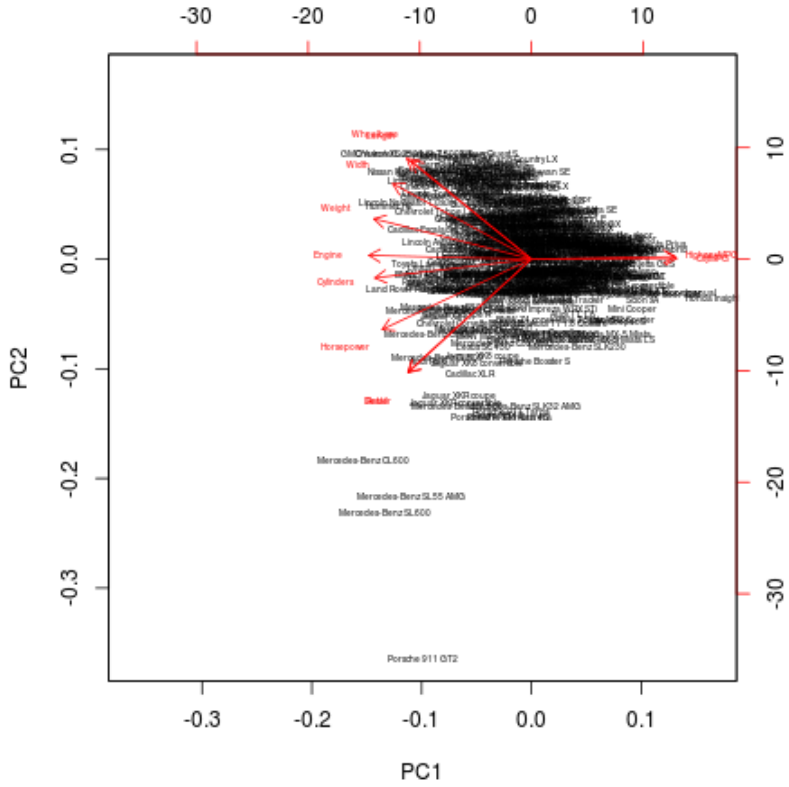
The positions of each observation in this new coordinate system of principal components are called **scores** and are calculated as linear combinations of the original variables and the weights p_{ij}. For example, the score for the r^{th} sample on the k^{th} principal component is calculated as

$$Y_{kr} = p_{k1}x_{k1} + p_{k2}x_{k2} + \ldots + p_{kp}x_{kp} \tag{14}$$

The figure shows the Y_{k1} scores (on x-axis) and the Y_{k2} (on Y axis)



PCA: Biplot



- Biplot: combined view of loadings and scores for the top two PCs
- The left and bottom axes show the loadings; the top and right axes show principal component scores.
- By comparing the score and loading plot, We can **identify the relationships between samples and variables**

Outline

Gene regulation and PCA

- Consider now the matrix $\boldsymbol{A}$ of TFAS profiles. There are $N \approx 20,000$ genes and $M \approx 10$ TFs that are stored in an $N \times M$ matrix
- First, this matrix is centered, i.e., the mean of each row is subtracted from the values of that row.

$$A'_{ij} = A_{ij} - \mu_i$$

- The mean value μ_i is the mean TFAS for gene i.
- Furthermore, the values A'_{ij} are divided by the standard deviation.
- This procedure is equivalent to replacing each value by its Z-score:

$$Z = \frac{A_{ij} - \mu}{\sigma(X)}$$

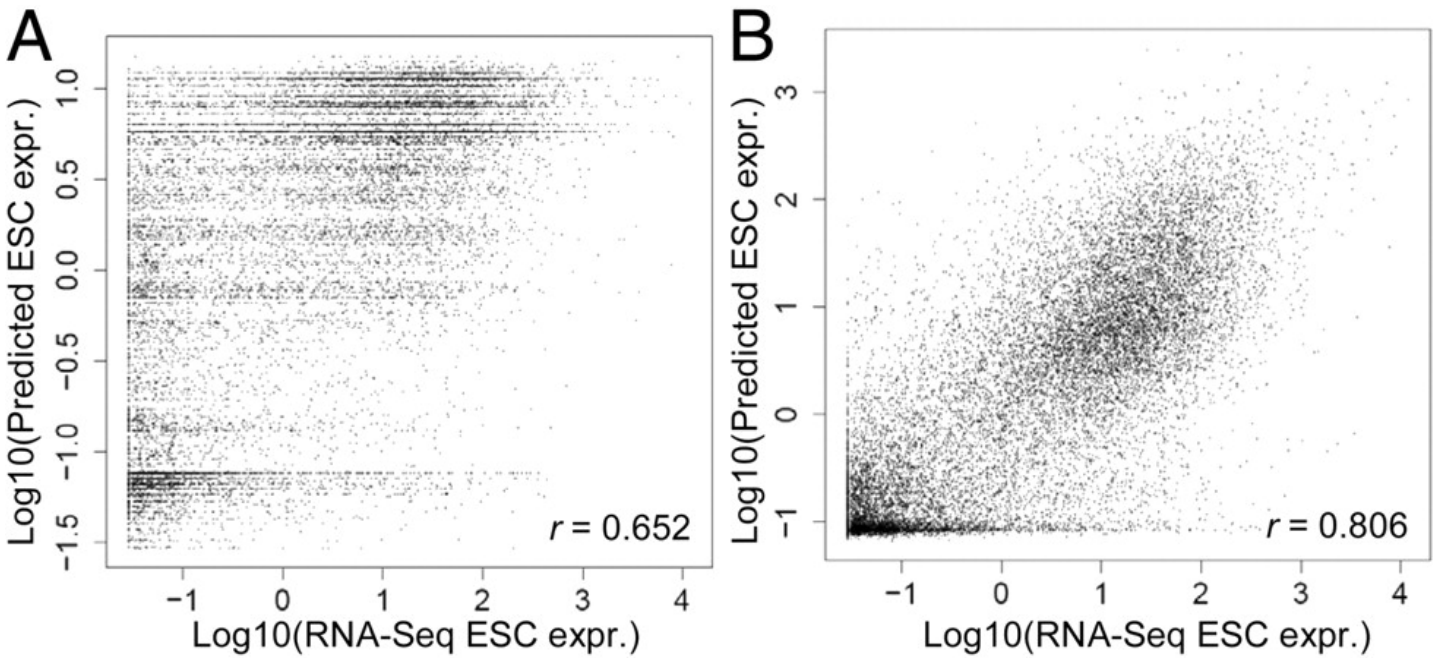
Gene regulation and PCA

The authors first decomposed the TFAS profiles into 12 principal components by PCA. Then they performed a log-linear regression on gene expression using the extracted principal components.

TFAS	R^2
Continuous	0.650
Binary	0.425

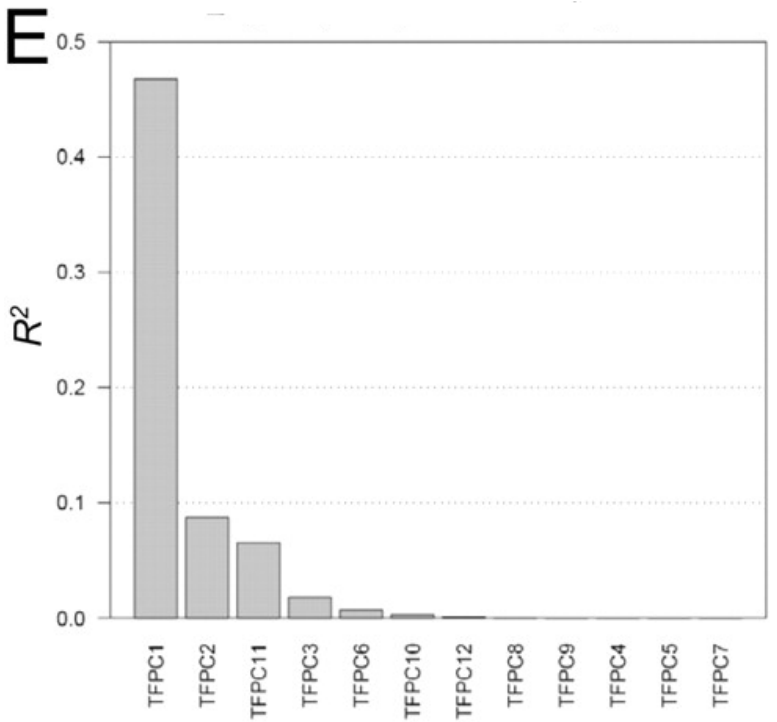
- Substantial improvement over most previous methods (R^2 between 9.6% and 36.9%)

Gene regulation and PCA



- Predicted versus observed ESC gene expression values for the RNA-Seq dataset on the binary TFAS. (PCA regression)

Gene regulation and PCA

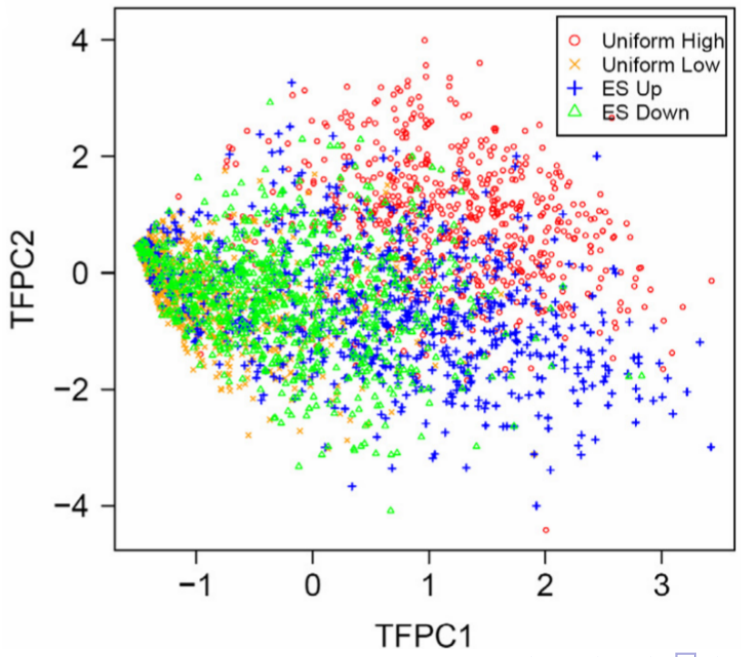


- Scree plot: The R^2 statistics of individual TFPCs for the prediction of RNA-Seq gene expression.
- The top three PC account for about 97% of the gene expression variation.

Gene regulation and PCA

The authors then look at 668 genes highly expressed in both ESCs and differentiated cells (**Uniform High**), 838 genes lowly expressed in both (**Uniform Low**), 782 genes up-regulated in ESCs (**ES Up**), and 831 genes down-regulated in ESCs (**ES Down**).

Visualization in the TFPC1–TFPC2 plane shows that the four sets of genes form clear clusters (Fig. S3A), suggesting that they are regulated by different combinations of the TFs.

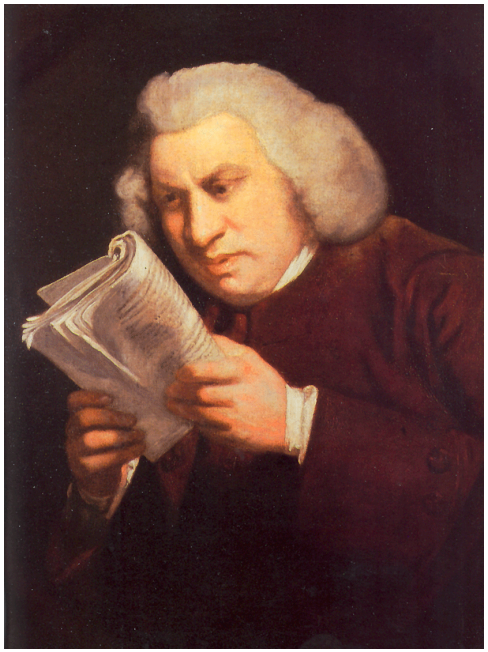


Gene regulation and PCA

- Finally, The authors claimed to learn regulatory rules that are combinations of TFPCs.
- For example, the Uniform Low gene set can be determined by $\mathrm{TFPC1} < -0.77$ (score of a gene) AND $\mathrm{TFPC2} < 0.25$
- The paper rewards more close reading, but let us stop here.
- In sum, joint modeling of ChIP-Seq and gene expression data (RNA- Seq and microarray) was used to quantify the contribution of TF binding on gene expression regulation.
- PCA was used to capture signal within noisy and partially redundant data
- Interpretation of the patterns of the PC loadings offers some insight into the gene regulation of ESCs

The End of the Lecture as We Know It

- Email: peter.robinson@charite.de
- Office hours by appointment



Lectures were once useful; but now, when all can read, and books are so numerous, lectures are unnecessary. If your attention fails, and you miss a part of a lecture, it is lost; you cannot go back as you do upon a book... People have nowadays got a strange opinion that everything should be taught by lectures. Now, I cannot see that lectures can do as much good as reading the books from which the lectures are taken. I know nothing that can be best taught by lectures, except where experiments are to be shown. You may teach chymistry by lectures. You might teach making shoes by lectures!

Samuel Johnson, quoted in Boswell's Life of Johnson (1791).