Gene expression microarray: Differential expression, batch effect

Outline

- Scientific goal and potential problems.
- Review of basic statistical concepts:
  - Hypothesis testing.
  - Multiple comparison problem.
- Differential expression (DE) test methods:
  - Empirical Bayesian (EB) methods: limma.
  - SAM.
  - Complex designs.
  - Permutation.
- Batch effect.
- Introduction to tiling arrays (will be skipped).

Input data for DE test

- Assume data are correctly within- and between-array normalized.
- Input data for DE test is a matrix of positive number:
  - Rows for genes and columns for samples.
  - Usually work on logarithm of the data, which are normal-ish.

Goal of DE test

- Goal: find genes that are expressed differently between (among) conditions.
  - Assign a score for each gene to represent its statistical significance of being different.
  - Rank the genes according to the score.
  - Find a proper threshold for the score for calling DE.
- Easy solutions:
  - Hypothesis testing (t-test, ANOVA, linear model, etc.) to get p-values and use as scores.
  - Use canonical cutoff (0.05) to call DE.

Potential problems

- Hypothesis testing:
  - sample sizes are usually small, which lead to unstable test results.
- When data are not normal, p-values are not accurate.
- Use 0.05 as threshold of p-values to call DE - multiple comparison problem:
  - Tests are performed for 20,000 genes. Even if all are null (not DE), 1,000 will have p-value less than 0.05.

Review of statistical inference

- Two-group t-test:
  - data from two groups (cancer and normal): $X_1, \ldots, X_M; Y_1, \ldots, Y_N$.
  - Assume *X*'s and *Y*'s are normally distributed.
  - A "null hypothesis" is that the means of *X*'s and *Y*'s are identical.
  - Test statistics $t = (\overline{X} - \overline{Y}) / \sqrt{S_X^2 / M + S_Y^2 / N}$

    where $S_X^2 = \frac{1}{M-1}\sum_{i=1}^{M}(X_i - \overline{X})^2, S_Y^2 = \frac{1}{N-1}\sum_{i=1}^{N}(Y_i - \overline{Y})^2$
  - t follows t-distribution.
  - P-value: under null hypothesis (that the means are the same), the probability to observe a t-statistics more extreme than the observed.

Why gene-by-gene t-test is a bad idea

- Small sample size (e.g., 3 vs. 3) leads to unstable estimates of variances.
  - By chance some genes have very small variance, which will result in large t-statistics and tiny p-values even when the difference is small.
  - Solution: SAM, EB methods.
- Sometimes data are not normally distributed, lead to incorrect p-values.
  - solution: non-parametric approach to obtain p-values.

SAM t-test

Tusher *et al.* (2001) *PNAS*

- Try to remove (or minimize) the dependence of test statistics on variances (because small variance tend to lead to bigger test statistics).
- Solutions: add a small constant to the denominator in calculating t statistics:

$$d_i = \frac{\bar{y}_i - \bar{x}_i}{s_i + s_0}$$

$\bar{y}_i, \bar{x}_i$: Means of two groups for gene i.

s_i : Standard deviation for gene i, assuming equal variace in both groups.

s_o : "Exchangeability factor" estimated using all genes.

The exchangeability factor

- Chosen to make signal-to-noise ratios independent of signal, e.g., the distribution of the statistics independent of the variance.
- Procedure:
  - Let S^{α} be the α percentile of the s_i values
  - For $\alpha \in (0, 0.01, 0.02, ..., 1.0)$ compute $d_i^{\alpha} = (\overline{y}_i - \overline{x}_i) / (s_i + s^{\alpha})$
  - Compute $v_j^{\alpha} = mad(d_i^{\alpha} \mid s_i \in [q_{j,}, q_{j+1})), \; j = 1, 2, \ldots 99$, here q_j are quantile.
  - Compute cv(α), the coefficient of variation of v_j^{α}
  - Choose $\hat{\alpha} = \underset{a}{\operatorname{argmin}}\{cv(a)\}. \; \hat{s}_o = \hat{s}^{\hat{\alpha}}$

SAM t-test

- Highly cited (>12,000 citations as of 2018), http://www-stat.stanford.edu/~tibs/SAM/.
- Implemented as Bioconductor package `siggenes`, and Excel plugin.
- Follow-up work: SAMSeq on RNA-seq DE test.
- Limitations: solutions for s_0 often sensitive to data.

Empirical Bayes method from `limma`

Smyth *et al.* (2004) *Statistical Applications in Genetics and Molecular Biology*

- Highly cited (~10,000 citations as of 2018).
- Use a Bayesian hierarchical model in multiple regression setting.
- Borrow information from all genes to estimate gene specific variances.
  - As a result, variance estimates will be “shrunk” toward the mean of all variances. So very small variance scenarios will be alleviated.
- Implemented in Bioconductor package “limma”.

The hierarchical model

Let β_{gj} be coefficient (difference in means in two group setting) for gene g, factor j, assume

$$\hat{\beta}_{gj} \mid \beta_{gj}, \sigma_g^2 \sim N(\beta_{gj}, v_{gj}\sigma_g^2) \qquad s_g^2 \mid \sigma_g^2 \sim \frac{\sigma_g^2}{d_g}\chi^2_{d_g}$$ with priors:

$$P(\beta_{gj} \neq 0) = p_j. \quad \beta_{gj} \mid \sigma_g^2, \beta_{gj} \neq 0 \sim N(0, v_{0j}\sigma_g^2). \quad \frac{1}{\sigma_g^2} \sim \frac{1}{d_0 s_0^2}\chi^2_{d_0}.$$

Posterior statistics

Posterior variance estimator: $\tilde{s}_g^2 = \dfrac{d_0 s_0^2 + d_g s_g^2}{d_0 + d_g}.$

Moderated t-statistics for testing β_{gj}=0 : $\tilde{t}_{gj} = \dfrac{\hat{\beta}_{gj}}{\tilde{s}_g \sqrt{v_{gj}}}.$

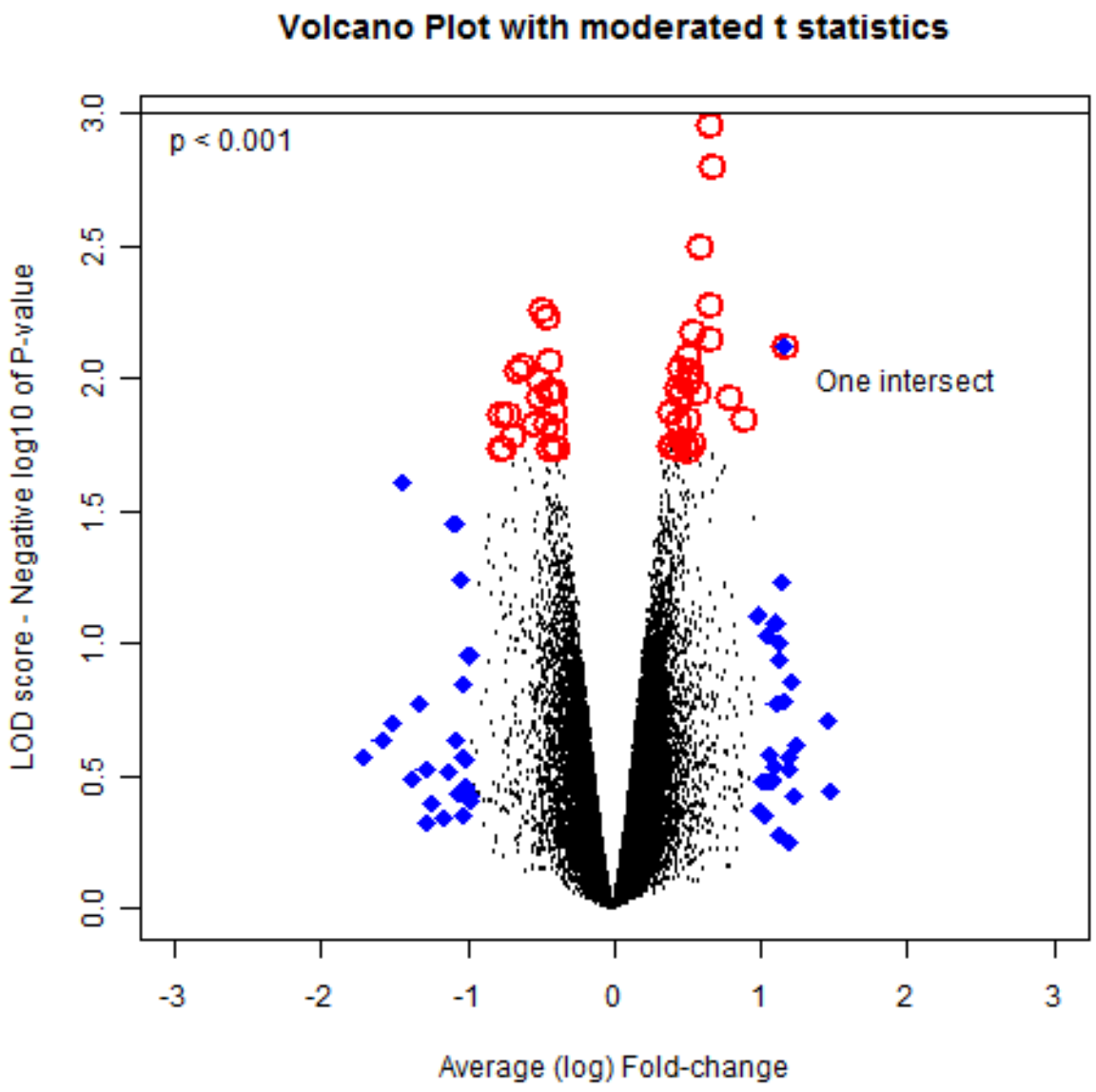
Summary on two-sample DE test

- Try to alleviate the “small sample variance” problem.
- Combine information from all genes.
- Many other variations of the model.
- In practice SAM and limma performs similarly.

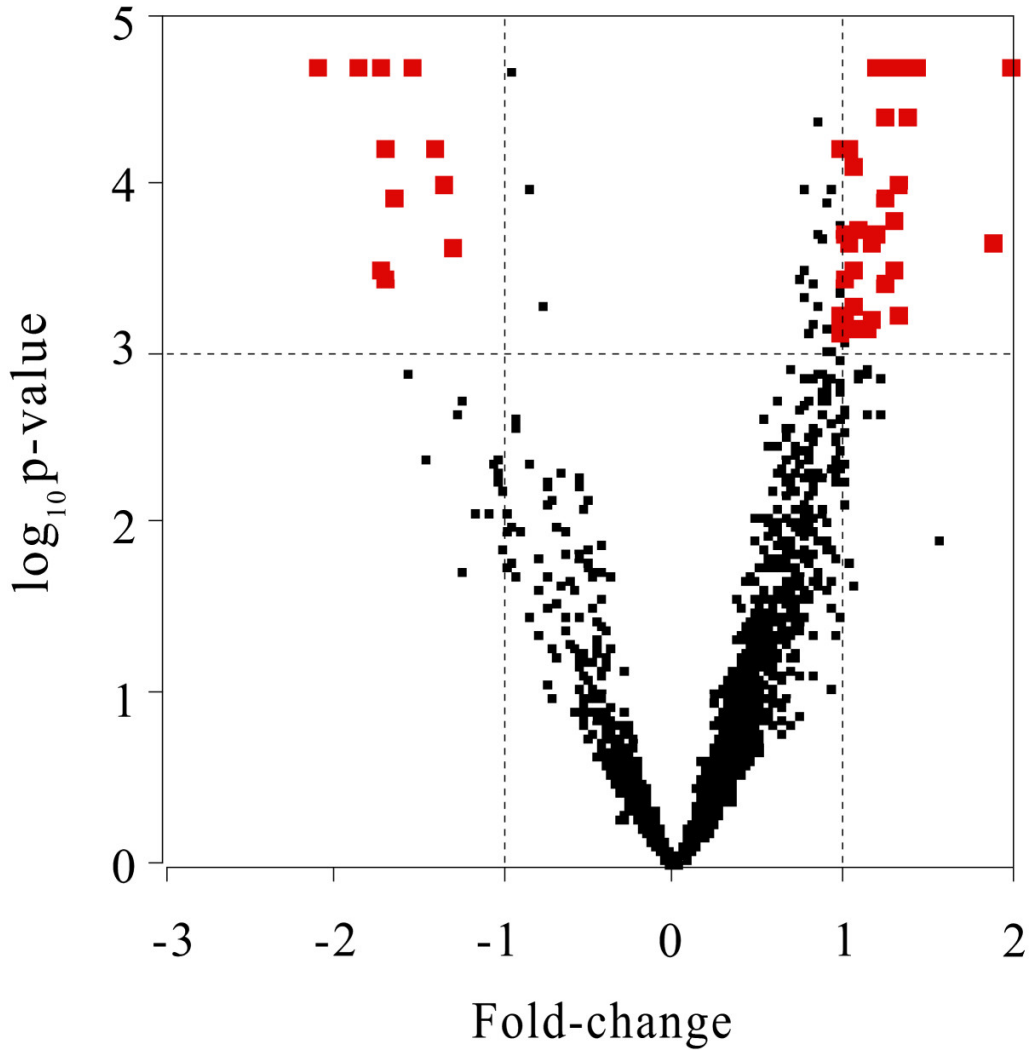
Volcano plot

- A diagnostic plot to visualize the test results.
- Scatter plot of the statistical significance (log p-values) vs. biological significance (log fold change).
- Ideally the two should agree with each other.

A bad volcano plot

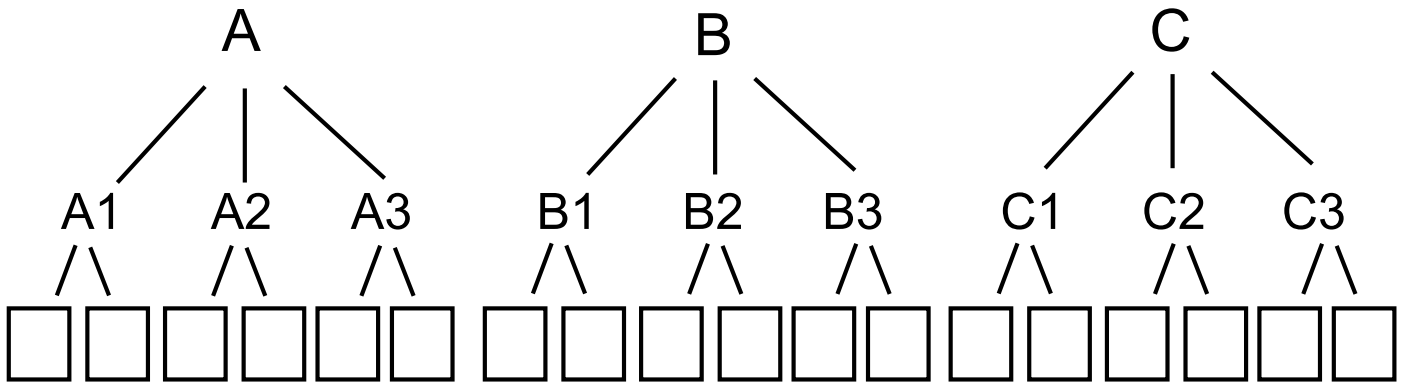


A good one

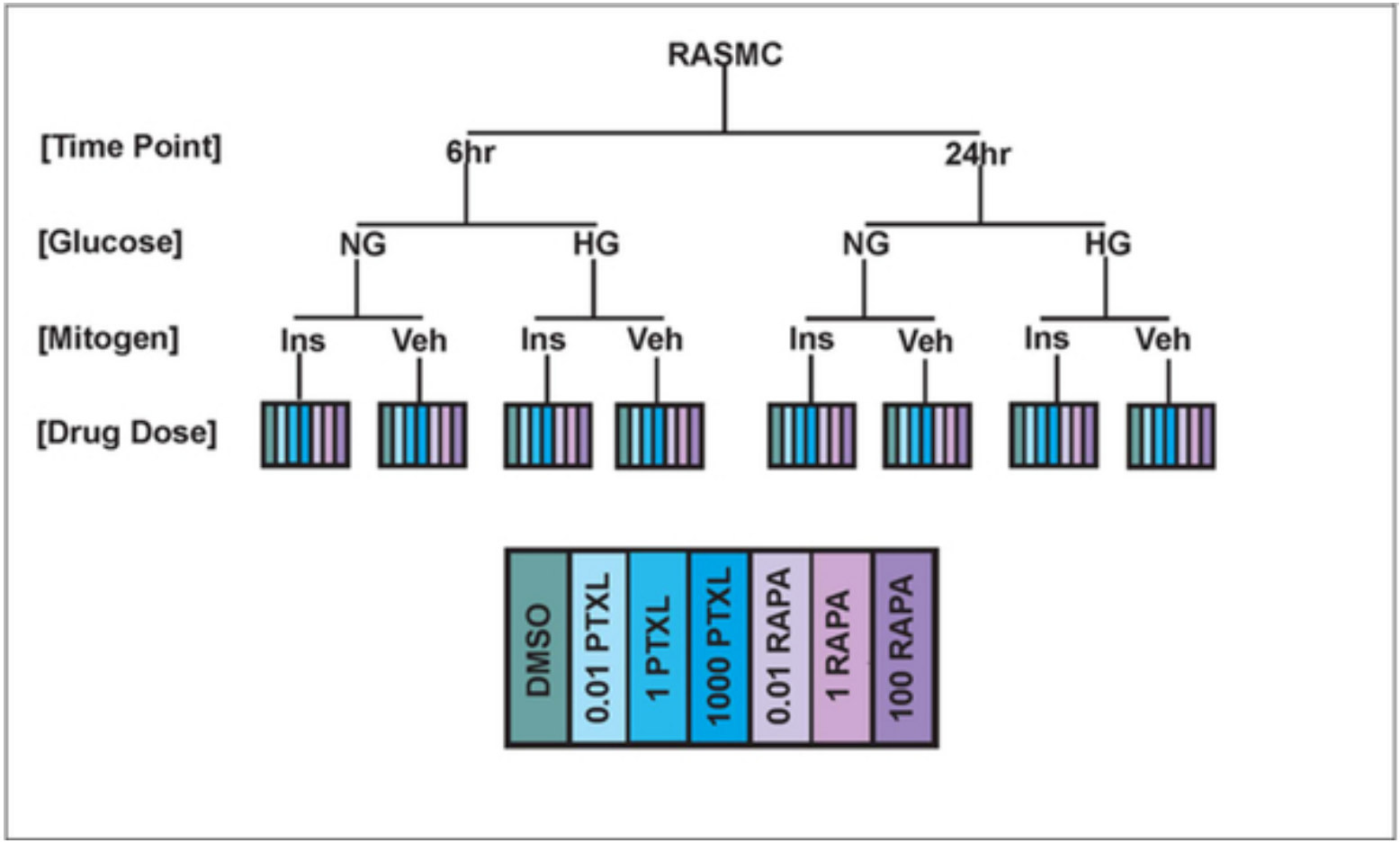


More complex experiments

- Complex experimental designs:
  - multiple (>2) groups.
  - crossed/nested.
  - etc.
- Examples for multiple-group:

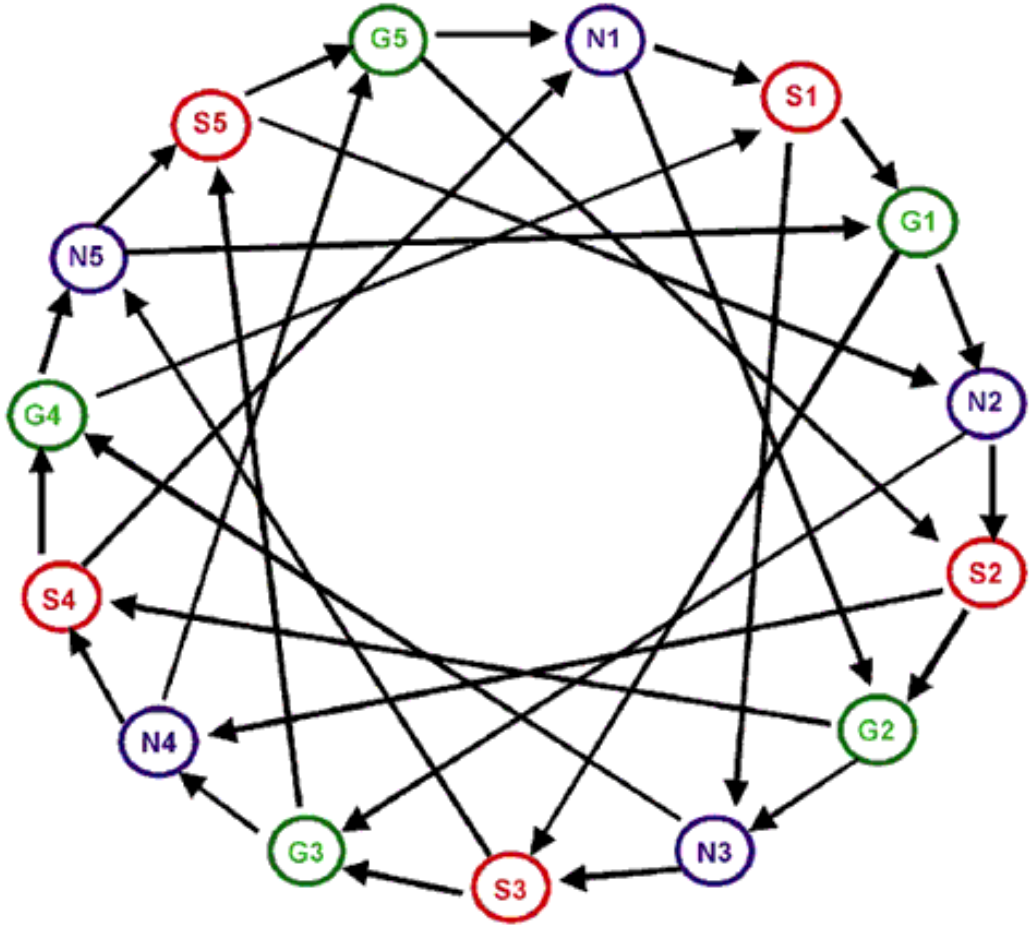


A crossed design



Charles et al. (2007), The Internet Journal of Genomics and Proteomics

A complicated loop design on two-color array



Oleksiak et al. (2002) *Nature Genetics*

DE test for complex design

- Two sample test -> multiple regression.
- The same problems still exist, and similar solutions can be applied.
- Mixed effect models can be used to capture heterogeneity among biological replicates.
- Both SAM and limma provide functions for complex designs.

P-values by randomization

- When the data don't satisfy normal assumption, permutation/bootstrap can be used to derive empirical p-values.
- Procedures for two sample comparison:
  - For each gene, randomly shuffle the data points.
  - Compute the t-statistics on the randomized data.
  - Repeat the procedure for N times, compute p-values as the percent of times that the permuted t-statistics more extreme than the observed.
- The procedure is a little complicated for multiple design. Basically shuffle the data based on null model.

Multiple testing correction

- Multiple testing problem is severe in high throughput data analysis because a large number of tests were performed.
  - Under type I error α=0.05, 1000 out of 20000 genes will be falsely declared DE (false positive) by chance.
  - If there are a total of 2000 genes declared DE, the false discovery rate (FDR) is 0.5!
- Multiple testing correction
  - Bonferroni correction: use α=0.05/20000 (too conservative).
  - FDR control (Benjamini and Hochberg, 1995 JRSS-B)

Bioconductor packages for microarray analysis

Bioconductor for microarray data

- There're a rich collection of bioc packages for microarrays. In fact, Bioconductor started for microarray analysis.
- There are currently 228 packages for microarray.
- Important ones include:
  - affy: one of the earliest bioc packages. Designed for analyzing data from Affymetrix arrays.
  - limma and siggenes: DE detection using limma and SAM-t model.
  - oligo: preprocessing tools for many types of oligonecleotide arrays. This is designed to replace affy package.
  - Many annotation data package to link probe names to genes.

My suggestion

- Use `oligo` to reading in data, normalization and summarization.
- Use `siggenes` or `limma` for detecting DE genes.

An exmple of Analyzing a set of Affymetrix data

- Data generated by MAQC (MicroArray Quality Control) project.
- Five brain samples and five reference samples on human exon arrays.
- Raw data are CEL files (binary file generated by factory).
- Each CEL file is around 65Mb.
- The platform design package (pd.huex.1.0.st.v2) needs to be installed.

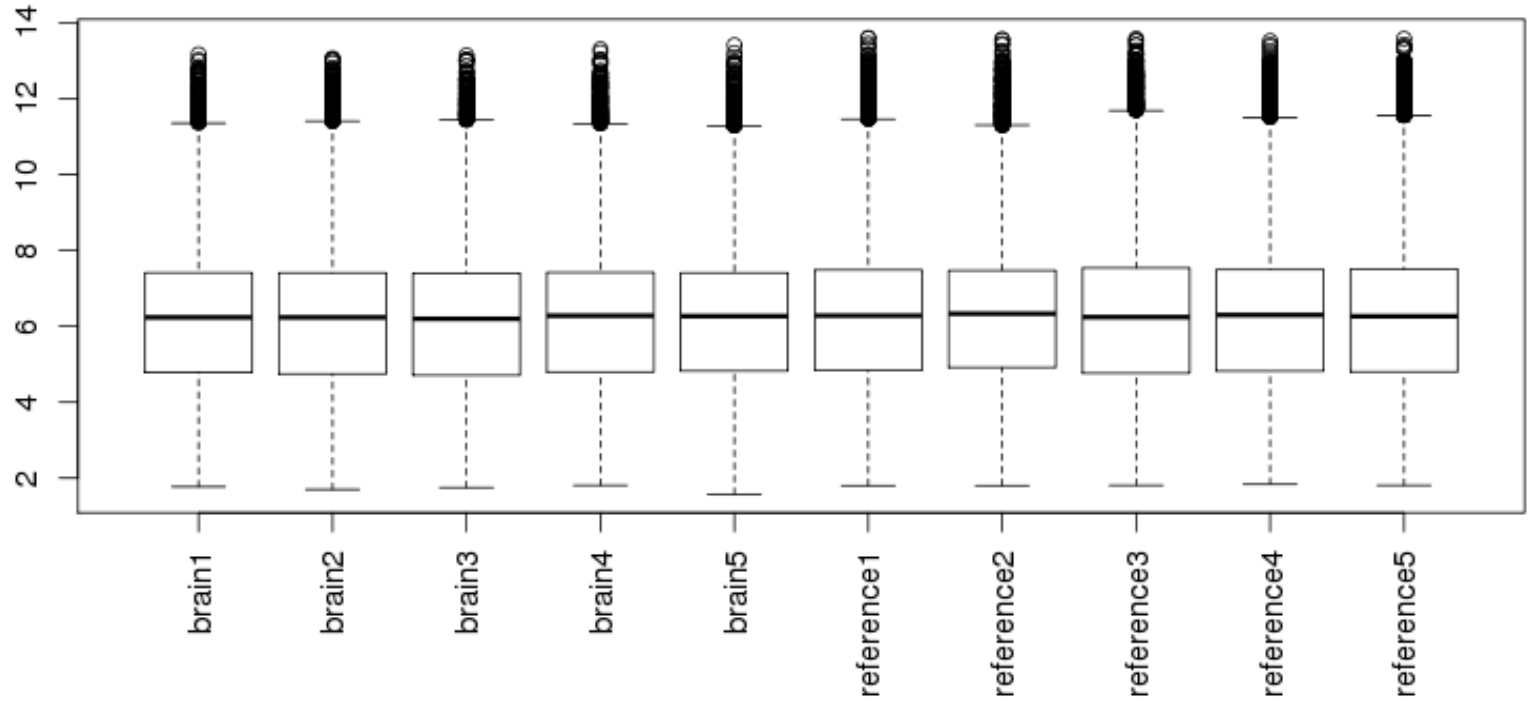
Read in data

```
## load in necessary libraries
> library(oligo)
> library(limma)
## get a list of CEL files
> CELfiles=dir(pattern="CEL")
## read in all raw data
> rawdata=read.celfiles(CELfiles)
> rawdata
ExonFeatureSet (storageMode: lockedEnvironment)
assayData: 6553600 features, 10 samples
  element names: exprs
protocolData
  rowNames: ambion_A1.CEL, ambion_A2.CEL, ..., stratagene_K2.CEL
(10 total)
...
Annotation: pd.huex.1.0.st.v2
```

Normalization and summarization

```
## using RMA
> normdata=rma(rawdata, target = "core")
> normdata
ExpressionSet (storageMode: lockedEnvironment)
assayData: 22011 features, 10 samples
  element names: exprs
...
## extract expression values using expr function
> data=exprs(normdata)
> head(data)
           sample 1  sample 2  sample 3  sample 4
1007_s_at 10.160224 10.214496 10.090697 11.020649
1053_at    9.501826  9.500412  9.574311  7.361141
117_at     5.669447  5.478072  5.648788  6.048142
121_at     8.061479  8.154549  8.156215  7.902597
1255_g_at  4.307739  4.017903  3.992333  4.668972
1294_at    7.108730  7.185586  7.122404  6.597161
```

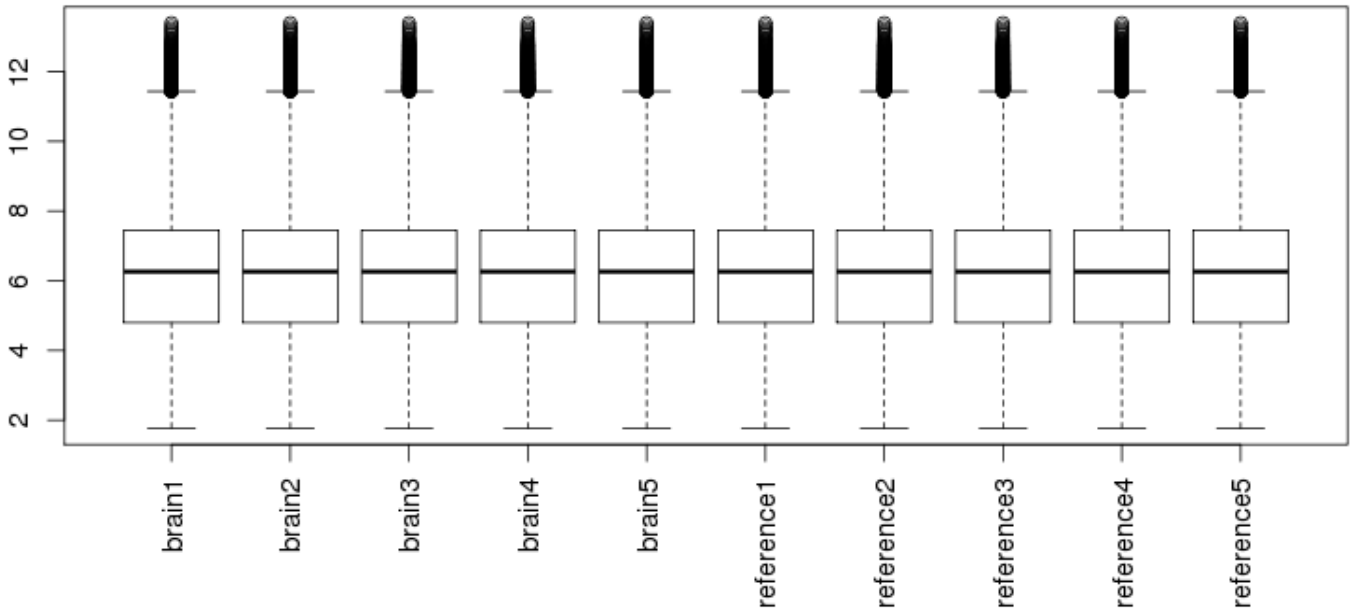
```
## check data distribution after RMA
> boxplot(data)
```



The boxplot looks really good after RMA, so between array normalization is unnecessary. But in case you need it, use `normalizeQuantiles` function from `limma` for quantile normalization :

```
> data2=normalizeQuantiles(data)
```

Now the new boxplot after quantile normalization:



DE detection using SAM t-test

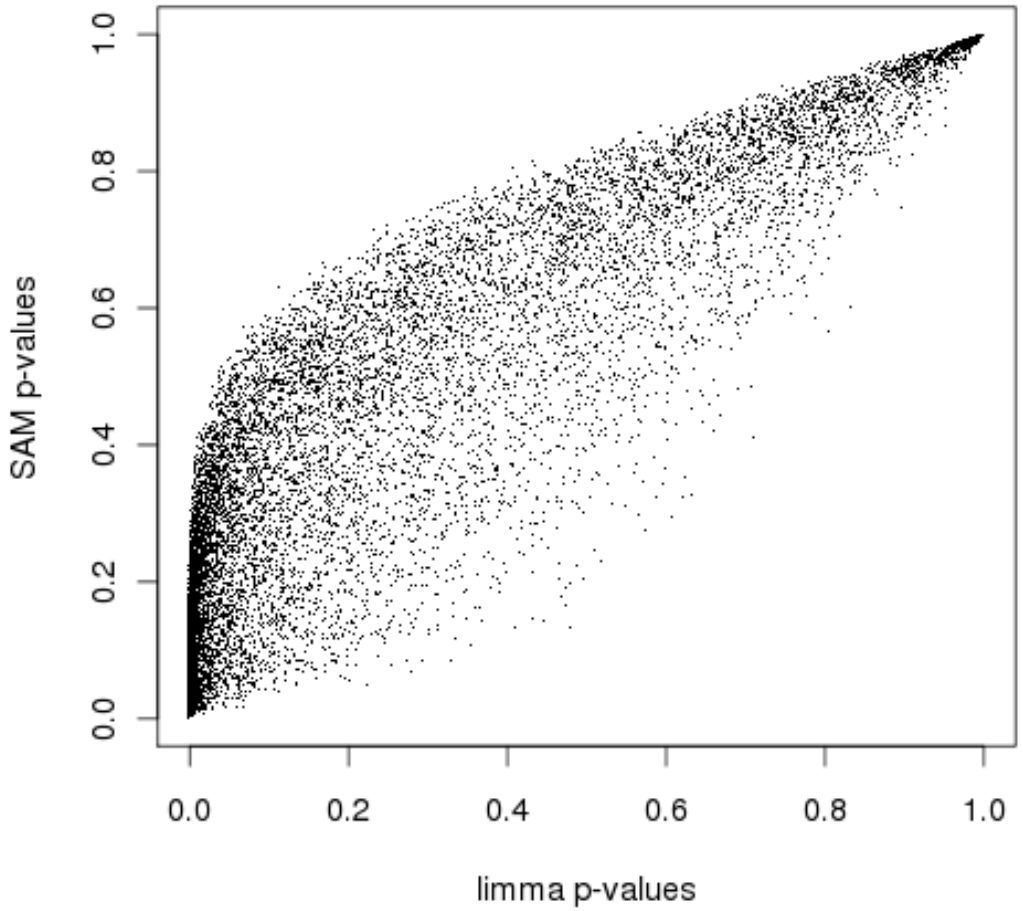
```
> library(siggenes)
## create a vector for design.
> design <- c(rep(0,5),rep(1,5))
> sam.result=sam(data2, cl=design)
> sam.result
SAM Analysis for the Two-Class Unpaired
Case Assuming Unequal Variances
```

DE detection using limma

```
## create design matrix. Intercept must be included
> design=cbind(mu=1,beta=c(rep(0,5),rep(1,5)))
## fit linear model and compute estimates
> limma.result=lmFit(data2, design=design)
## Empirical Bayes method to get p-values
> limma.result=eBayes(limma.result)
## get p-values for the comparison
> pval=limma.result$p.value[,"beta"]
```

Compare results from limma and SAM

- Agreement is good, 0.95 Spearman rank correlation.
- Limma seems to be more liberal.



Obtain gene annotations

- Now you get p-values for all genes, but you also need gene names for generating report.
- There are many annotation packages available for different array platforms. For example, hgu133a.db is for HGU133A arrays.
- These packages contain comprehensive information for all probes, including their sequences, chromosome, position, corresponding gene IDs, GO terms, etc.

- A typical way to convert probeset names to accession number or gene alias is:

```
> library(hgu133a.db)
## convert to accession numbers:
> geneAcc=as.character(hgu133aACCNUM[rownames(data)])
## convert to gene names
> geneNames=as.character(hgu133aSYMBOL[rownames(data)])
```

Finally generate a report table

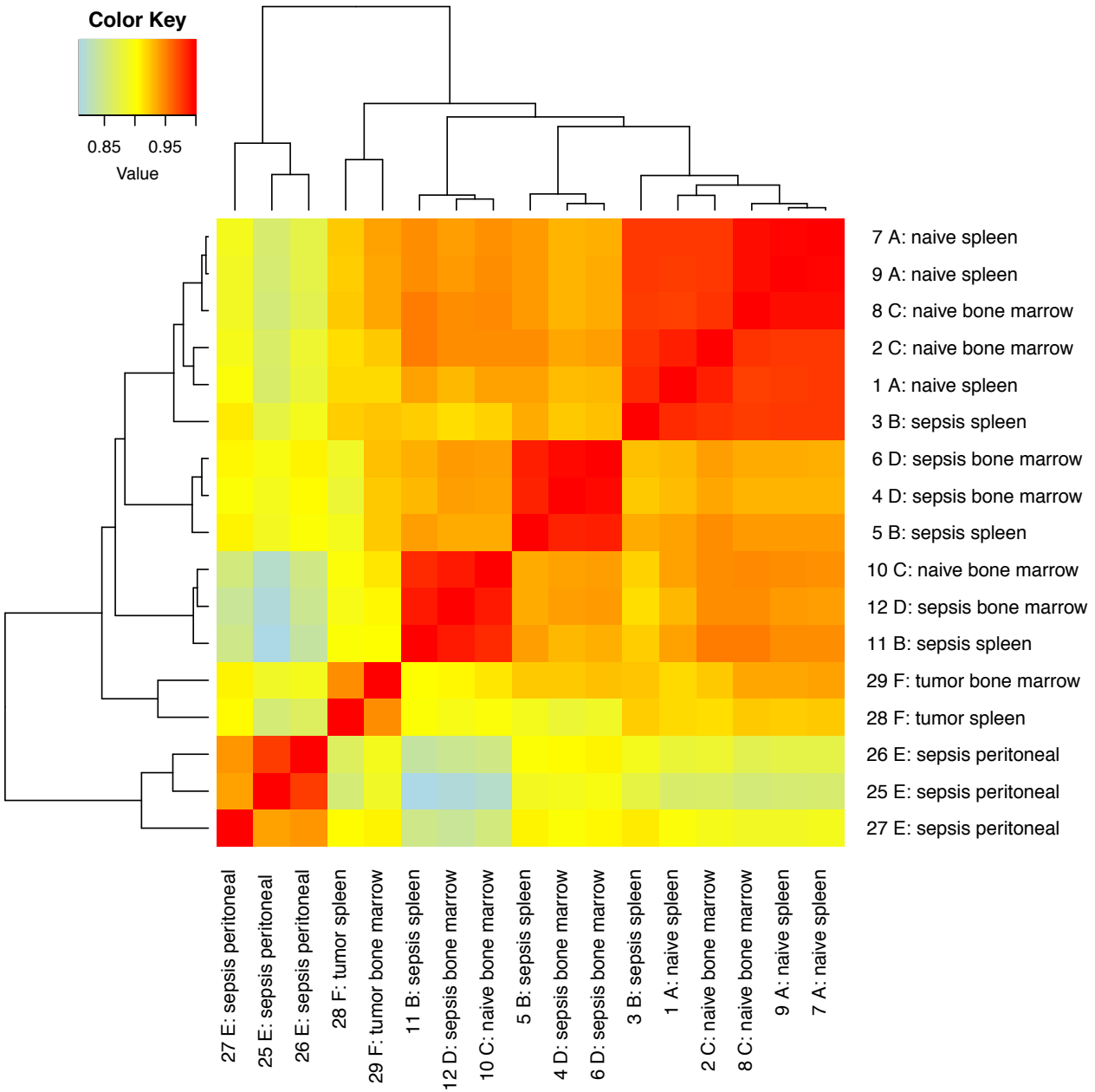
```
> ix=sam.result@q.value<0.1
> result=data.frame(gene=geneNames[ix],
    pvalue=sam.result@p.value[ix],
    fold=sam.result@fold[ix])
## sort by fold change
> ix2=sort(result$fold, decreasing=TRUE, index.return=TRUE)$ix
> result=result[ix2,]
> head(result)
                  gene pvalue     fold
2731192    NM_000477      0 185.5720
3457336    NM_006928      0 155.7143
2772566    NM_144646      0 152.8232
2731230    NM_001134      0 132.8515
> write.table(result, file="report.txt", sep="\t")
```

Batch effect

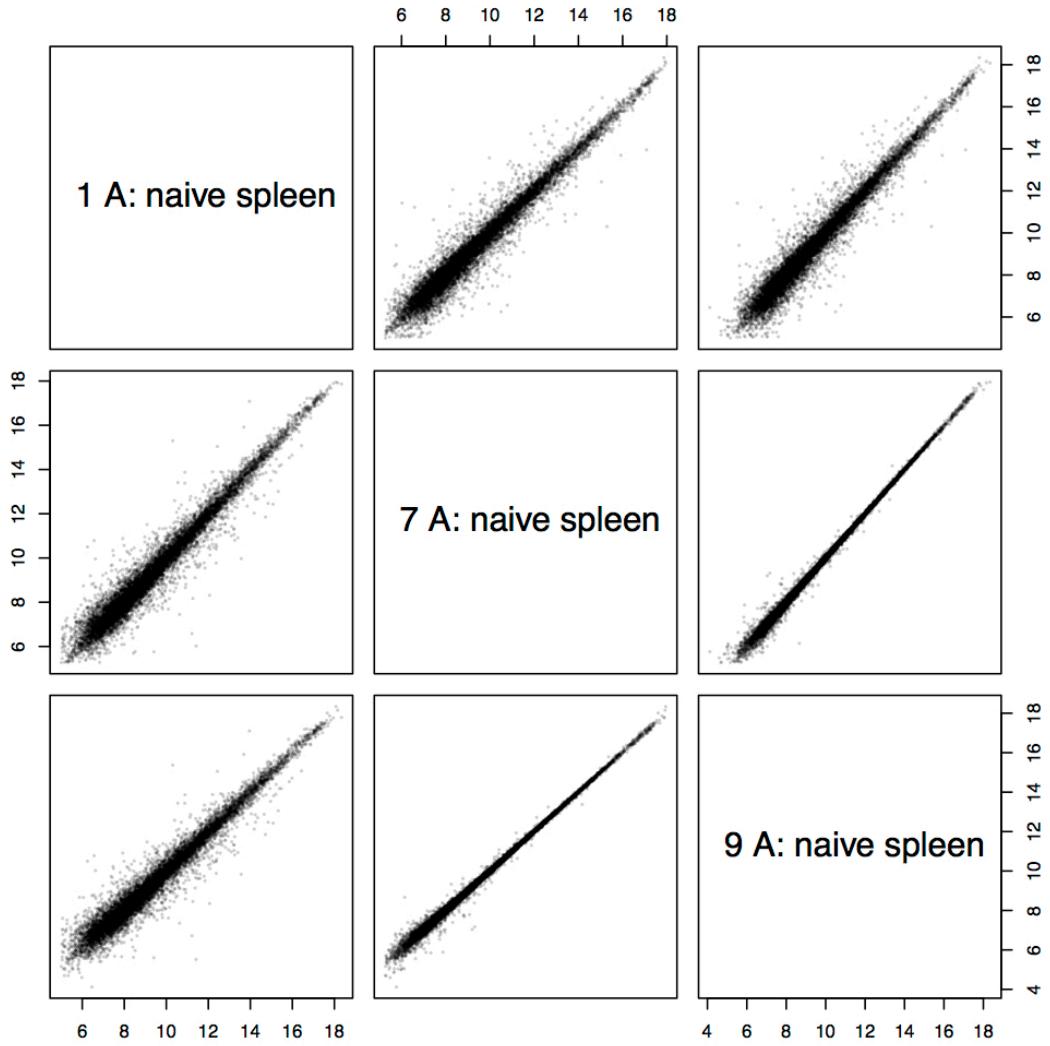
What is batch effect

- Microarray experiments are very sensitive to experimental conditions:
  - Equipment, agents, technicians, etc.
- Data generated from different “batches” (lab, time, etc.) can be quite different, but data from the same batch tend to be more similar.
- So batch effects are structured noise/bias common to all replicates in the same batch, but markedly different from batch to batch.

Example


Color Key
0.85
0.95
Value
7 A: naive spleen
9 A: naive spleen
8 C: naive bone marrow
2 C: naive bone marrow
1 A: naive spleen
3 B: sepsis spleen
6 D: sepsis bone marrow
4 D: sepsis bone marrow
5 B: sepsis spleen
10 C: naive bone marrow
12 D: sepsis bone marrow
11 B: sepsis spleen
29 F: tumor bone marrow
28 F: tumor spleen
26 E: sepsis peritoneal
25 E: sepsis peritoneal
27 E: sepsis peritoneal
27 E: sepsis peritoneal
25 E: sepsis peritoneal
26 E: sepsis peritoneal
28 F: tumor spleen
29 F: tumor bone marrow
11 B: sepsis spleen
12 D: sepsis bone marrow
10 C: naive bone marrow
5 B: sepsis spleen
4 D: sepsis bone marrow
6 D: sepsis bone marrow
3 B: sepsis spleen
1 A: naive spleen
2 C: naive bone marrow
8 C: naive bone marrow
9 A: naive spleen
7 A: naive spleen

Variation within and between batches



Methods to remove batch effects

- Based on linear model: batches cause location/scale changes (e.g., combat).
- Based on dimension reduction technique: SVD, PCA, factor analysis, etc. (e.g., sva).
  - The singular vectors/PCs/factors that are correlated with batch are deemed from batch effects.
  - Remove batch effects from data, leftovers are biological signals.

sva package in Bioconductor

- Contains `ComBat` function for removing effects of known batches.
- Assume we have
  - `edata`: a matrix for raw expression values
  - `batch`: a vector named for batch numbers.

```
modcombat = model.matrix(~1, data=as.factor(batch))
combat_edata = ComBat(dat=edata, batch=batch,
  mod=modcombat, par.prior=TRUE, prior.plot=FALSE)
```

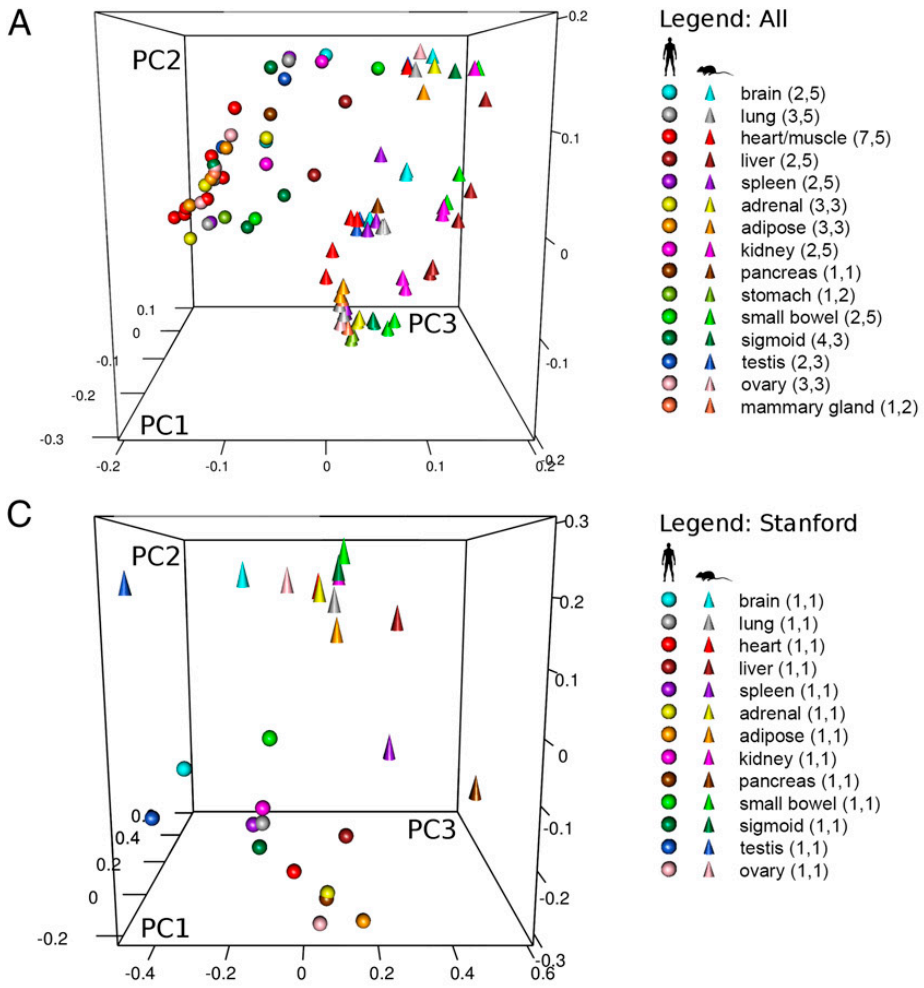
BatchQC - Batch Effects Quality Control

- A Bioconductor package with a GUI (shiny app).
- https://github.com/mani2012/BatchQC

Comparison of the transcriptional landscapes between human and mouse tissues

Shin Lin[a,b,1], Yiing Lin[c,1], Joseph R. Nery[d], Mark A. Urich[d], Alessandra Breschi[e,f], Carrie A. Davis[g], Alexander Dobin[g], Christopher Zaleski[g], Michael A. Beer[h], William C. Chapman[c], Thomas R. Gingeras[g,i], Joseph R. Ecker[d,j,2], and Michael P. Snyder[a,2]

- One major conclusion is that tissues are more similar within a species, compared with the same tissue across species.



RESEARCH ARTICLE

A reanalysis of mouse ENCODE comparative gene expression data [version 1; referees: 3 approved, 1 approved with reservations]

Yoav Gilad, Orna Mizrahi-Man

Department of Human Genetics, University of Chicago, Chicago, IL, 60637, USA

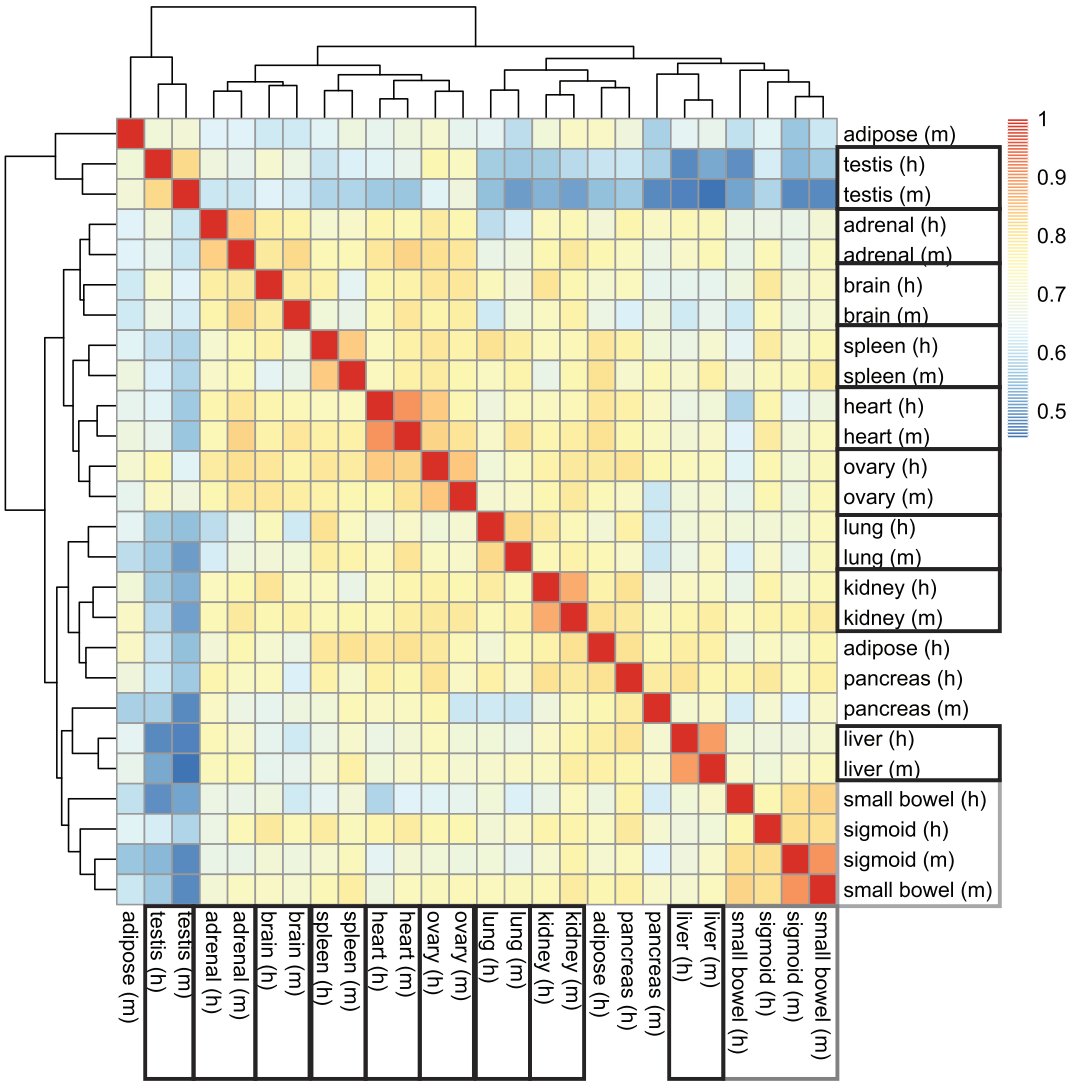
- Experimental design: data are from 5 batches.

D87PMJN1 (run 253, flow cell D2GUAACXX, lane 7)	D87PMJN1 (run 253, flow cell D2GUAACXX , lane 8)	D4LHBFN1 (run 276, flow cell C2HKJACXX , lane 4)	MONK (run 312, flow cell C2GR3ACXX , lane 6)	HWI-ST373 (run 375, flow cell C3172ACXX , lane 7)
heart	adipose	adipose	heart	brain
kidney	adrenal	adrenal	kidney	pancreas
liver	sigmoid colon	sigmoid colon	liver	brain
small bowel	lung	lung	small bowel	spleen
spleen	ovary	ovary	testis	Human
testis		pancreas		Mouse

After correcting for batch effects

- Tissues tend to cluster together more.

b



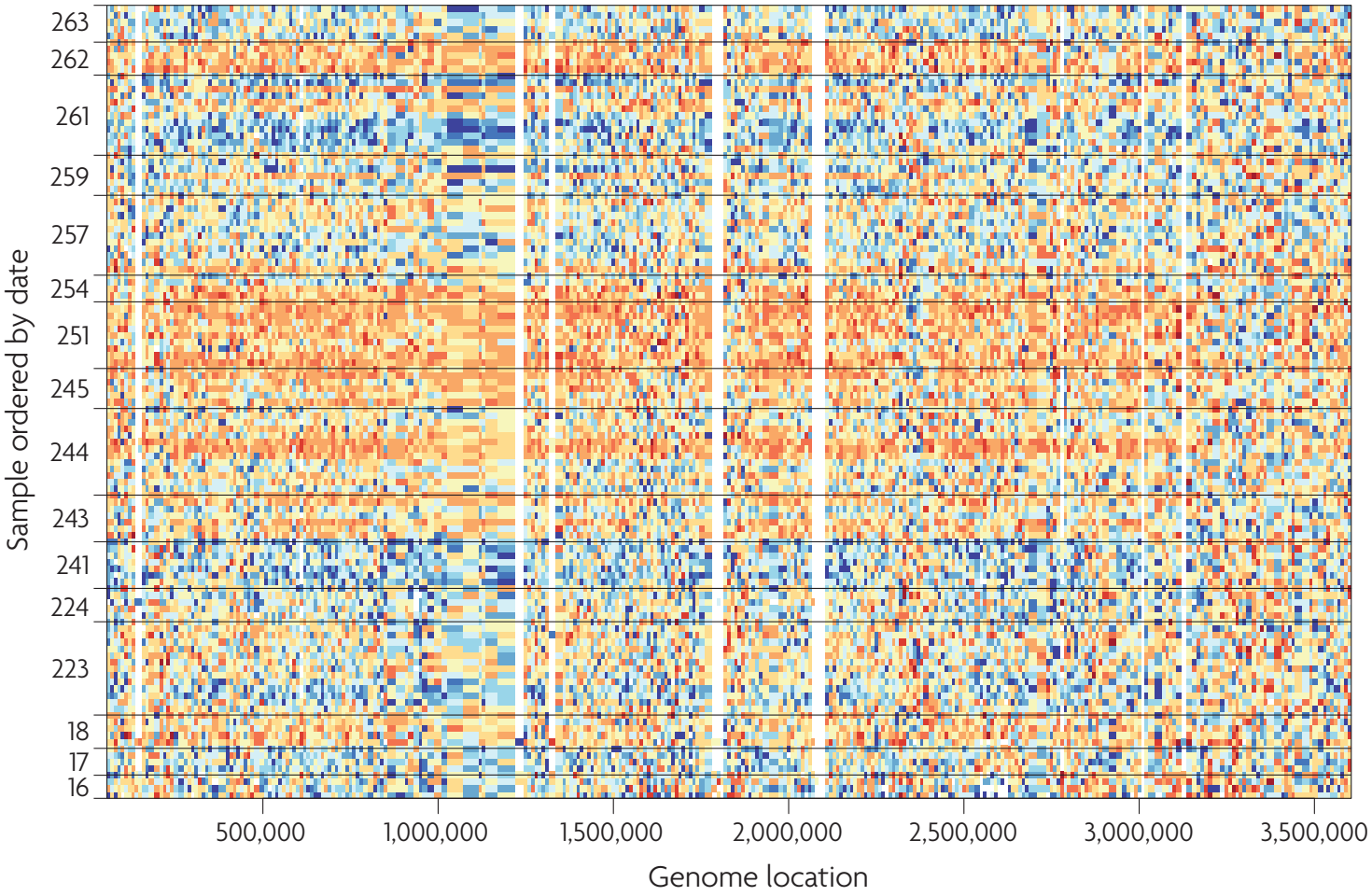
Batch effects are prevalent

- Observed in many high-throughput experiments: microarray, different types of sequencing, even brain imaging.
- Methods for identifying and removing batch effects is under continuous developments.

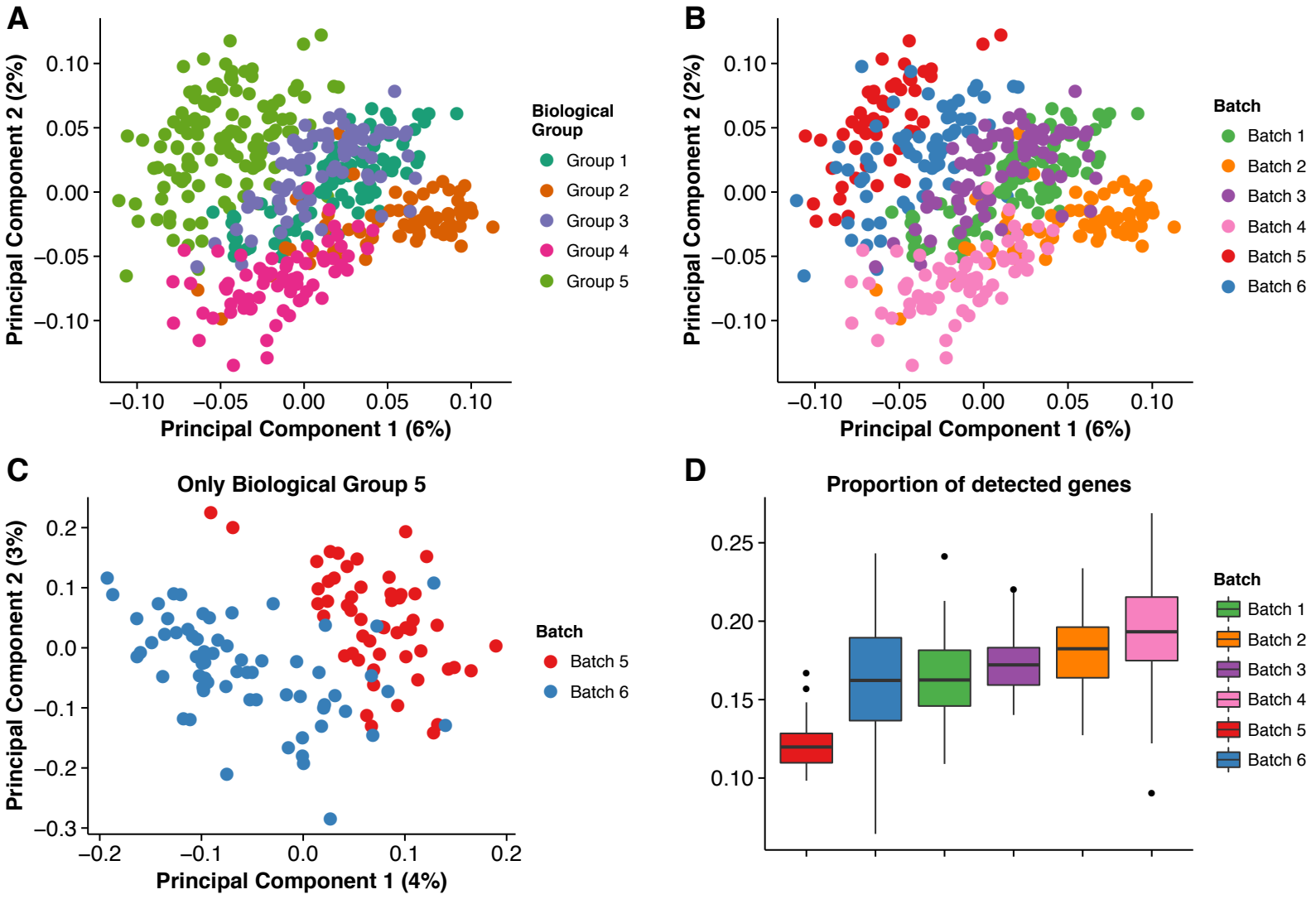
OPINION

Tackling the widespread and critical impact of batch effects in high-throughput data

Jeffrey T. Leek, Robert B. Scharpf, Héctor Corrada Bravo, David Simcha, Benjamin Langmead, W. Evan Johnson, Donald Geman, Keith Baggerly and Rafael A. Irizarry



In single cell RNA-seq
(Hicks et al. 2016, bioRxiv)



Review

- We have covered microarray analysis DE test, including:
  - SAM t-test.
  - EB method: Limma.
  - A little on complex design.
  - Permutation test.
  - Multiple testing.
  - R/Bioconductor packages for DE analysis.
- Batch effects.

Introduction to genome tiling microarray analysis

Biological motivations

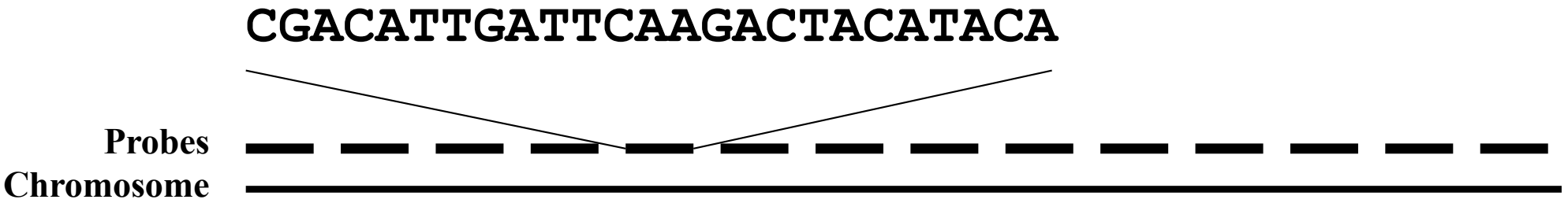
- There are many types of “events” happen at different locations on the genome. For example, protein bindings, epigenetic modifications (DNA methylation and histone modifications), copy number variations, etc.
- It is often of great interests to detect the genomic locations where a specific event happens, or quantify the events along the genome.

An example: transcription factor(TF) binding

- Transcription factors (TF): proteins that binds to specific DNA sequences and control the transcription from DNA to mRNA.
- There are many different types of TFs, each recognize different DNA sequences (motifs).
- The functions of the TFs are important for understanding gene regulatory mechanisms.
- The first step toward the understanding is to detect the TF binding sites (TFBS).

Tiling arrays

- The goal is to quantify the events of interests along the genomes, and/or detect the genomic coordinates for the events.
- Work the same as gene expression array (hybridization based), except that the probes are designed to tile up the genome at non-repeat regions.
- Data for probes in the location of interest often behave differently from backgrounds (e.g., bigger intensities).

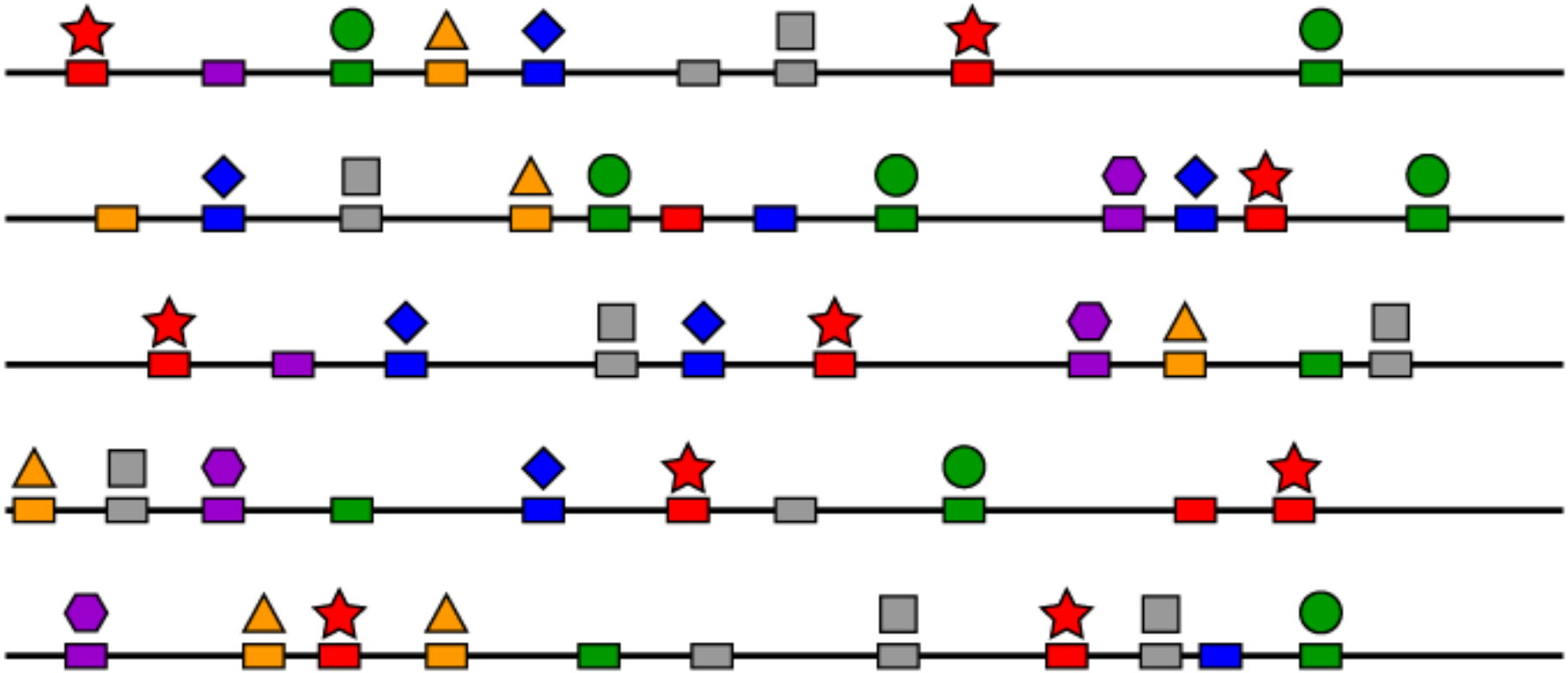


Types of tiling arrays

- ChIP-chip: Chromatin ImmunoPrecipitation (ChIP) + tiling array (chip) for detecting transcription factor binding sites or measuring histone modification levels.
- MeDIP-chip: Methyl-DNA ImmunoPrecipitation (MeDIP) + tiling array (chip) for measuring DNA methylation level.
- ArrayCGH (Comparative Genomic Hybridization) for detecting copy number variations.

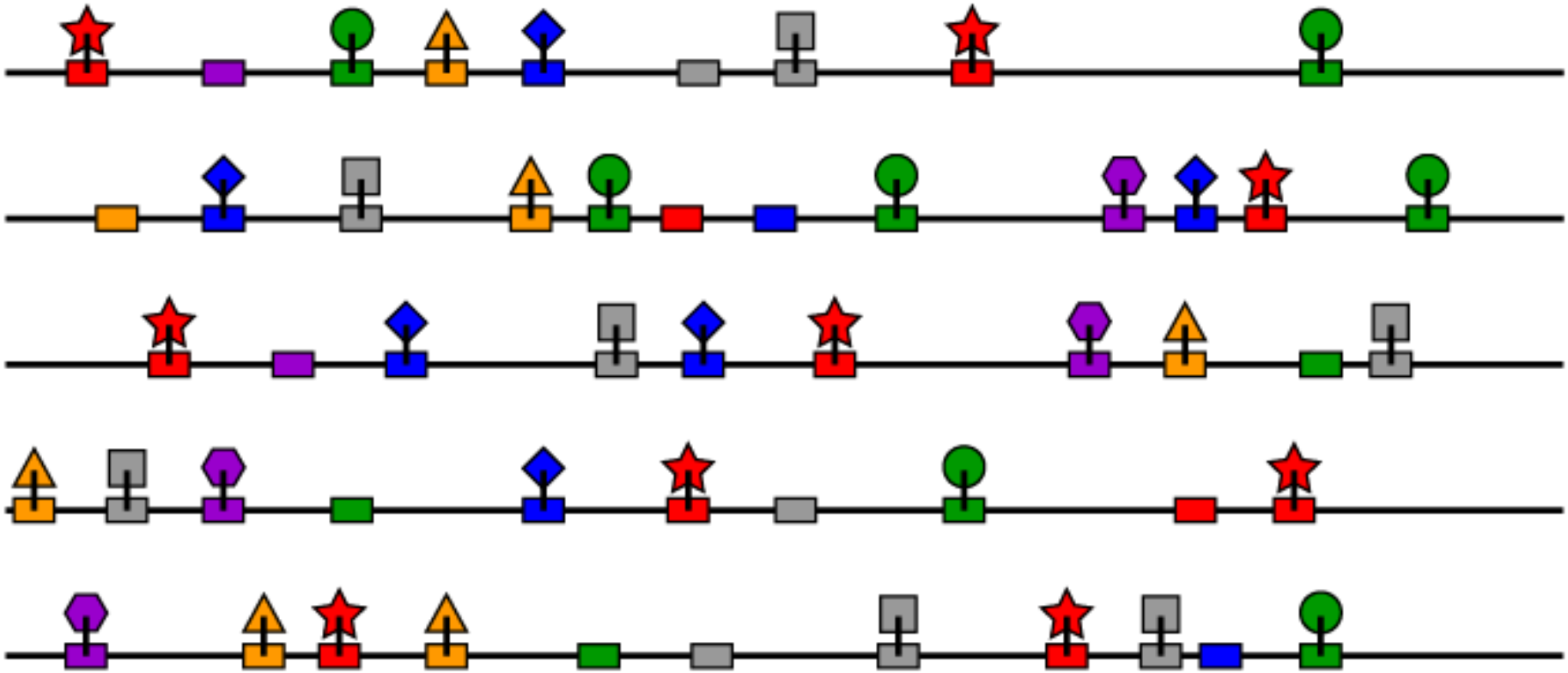
There's no major differences in array designs. Difference are the ways to prepare biological samples.

Chromatin ImmunoPrecipitation (ChIP)



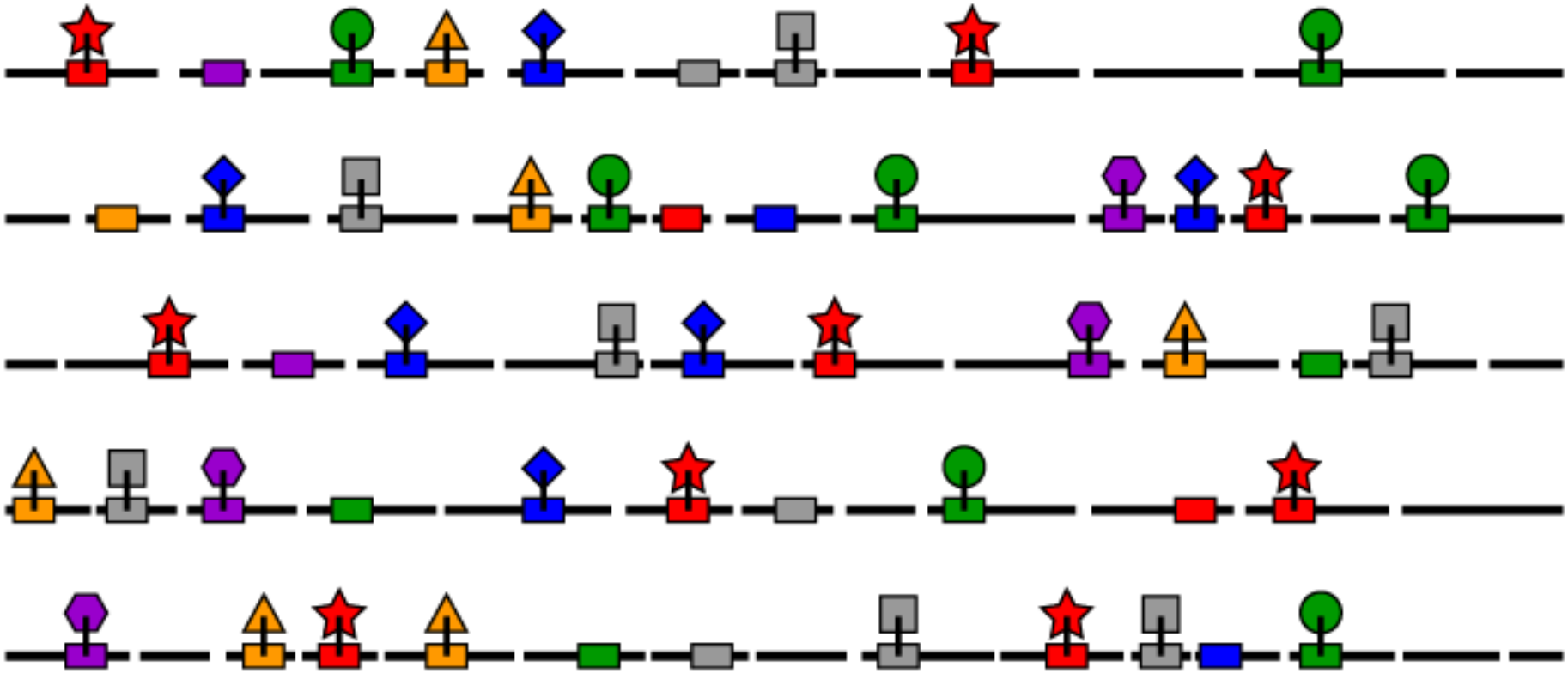
By Richard Bourgon at UC Berkley

TF/DNA Crosslinking *in vivo*



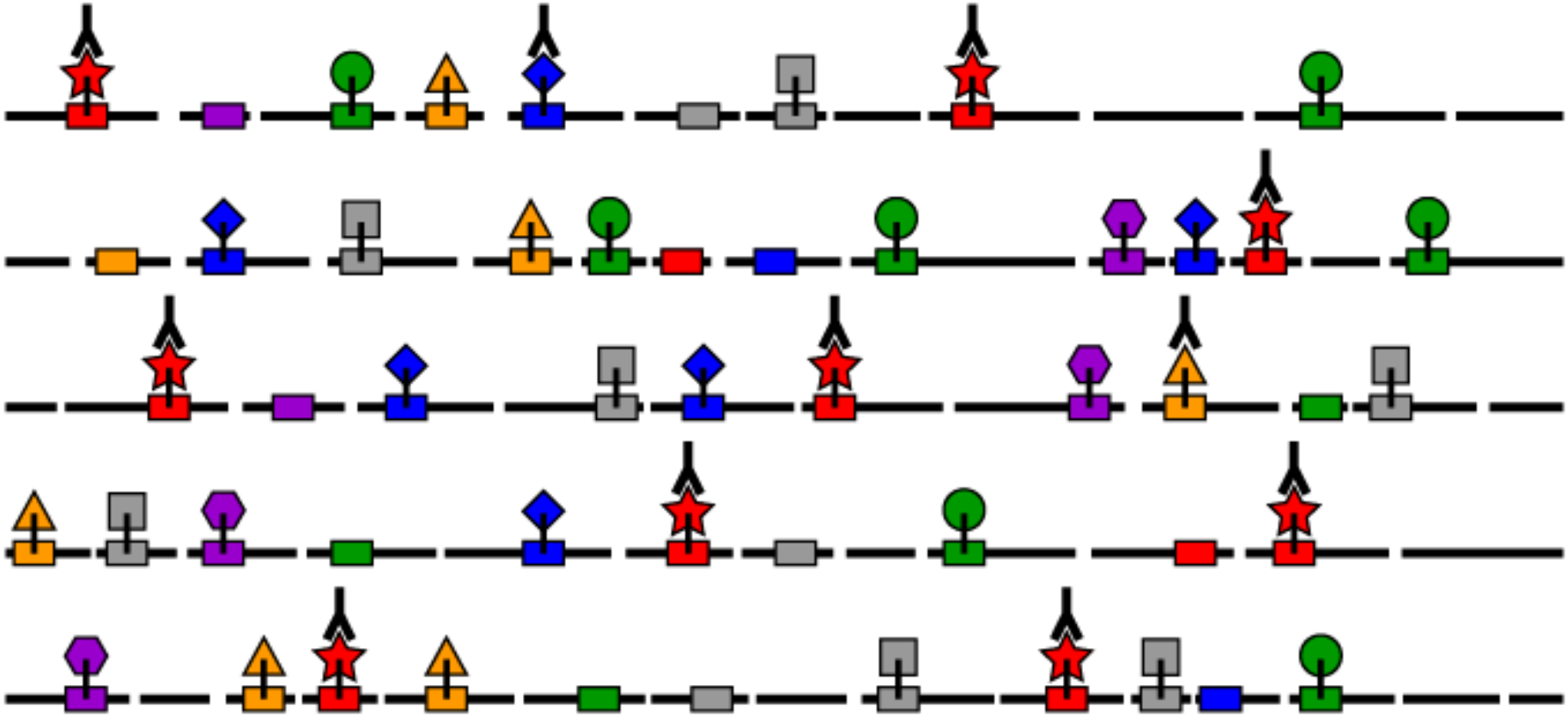
By Richard Bourgon at UC Berkley

Sonication (~500bp)



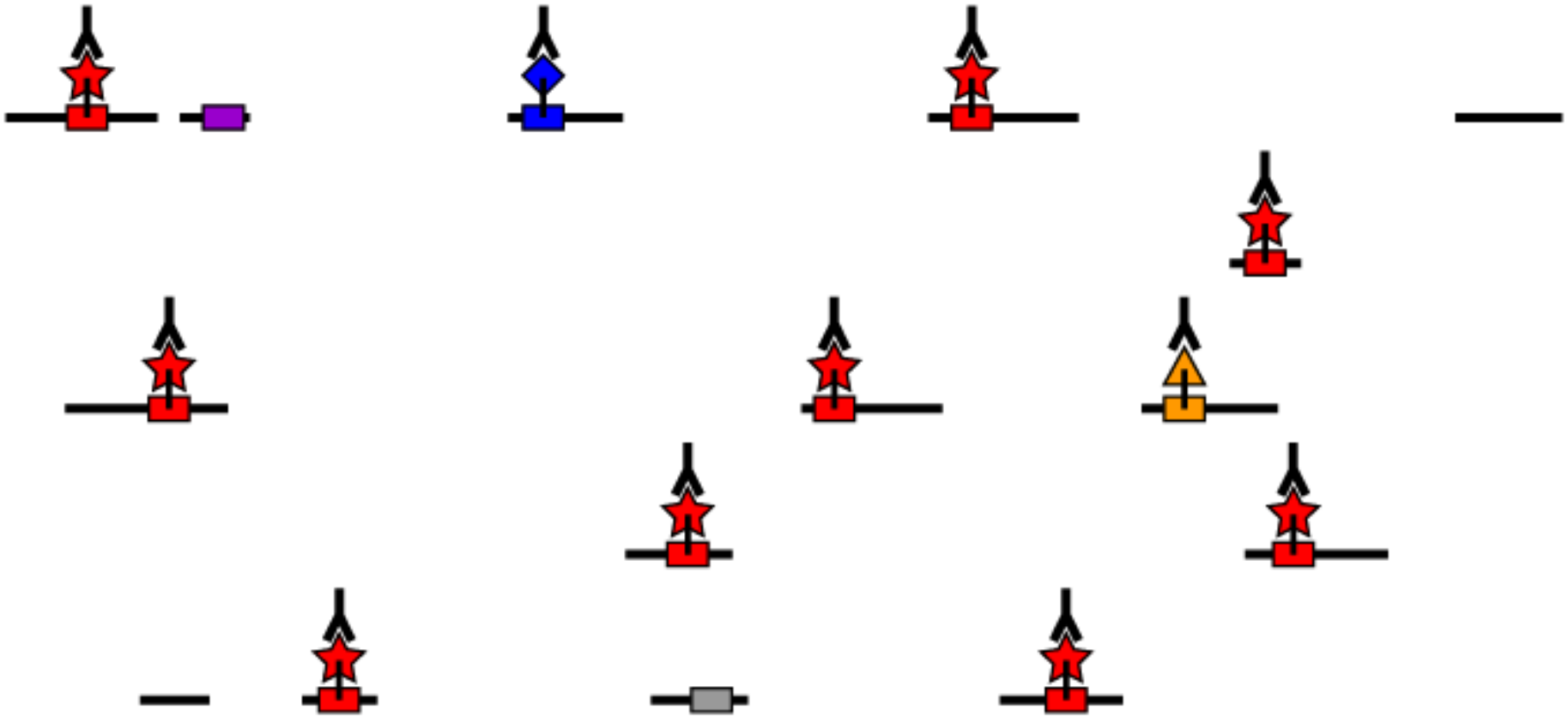
By Richard Bourgon at UC Berkley

TF-specific Antibody



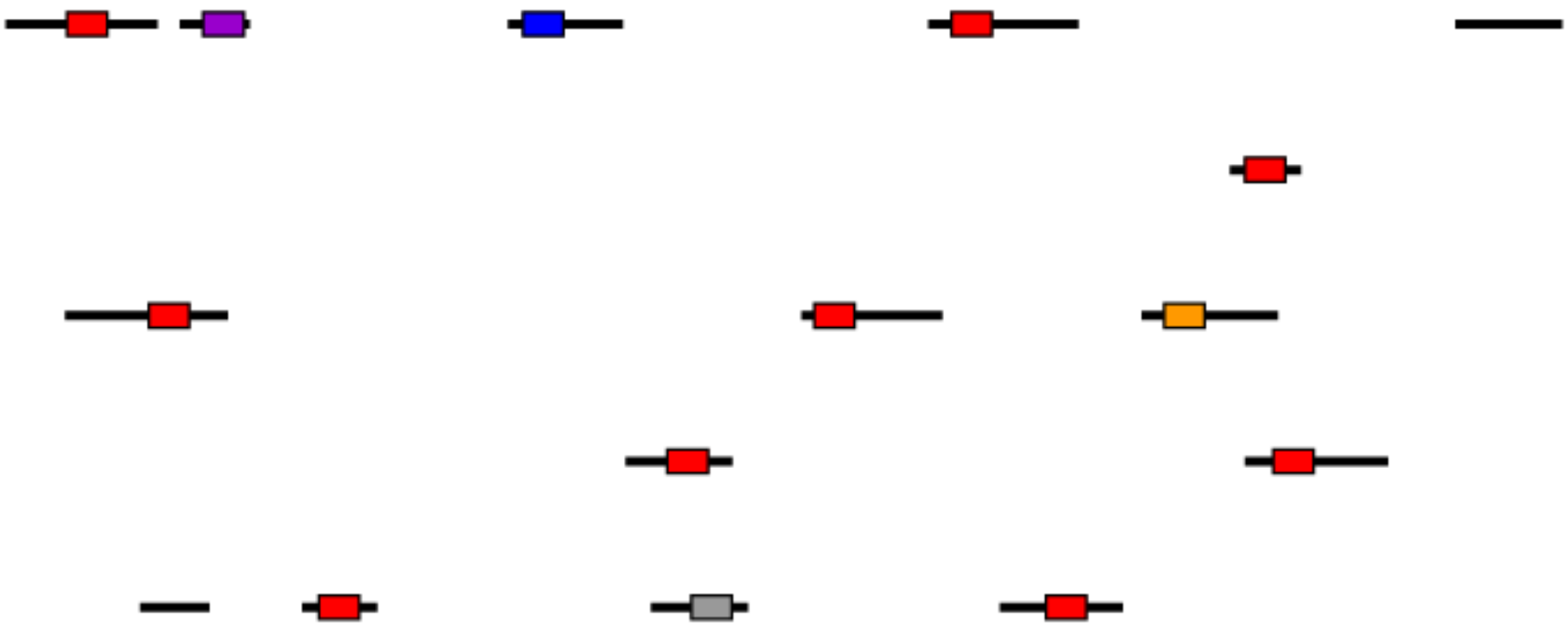
By Richard Bourgon at UC Berkley

Immunoprecipitation (IP)



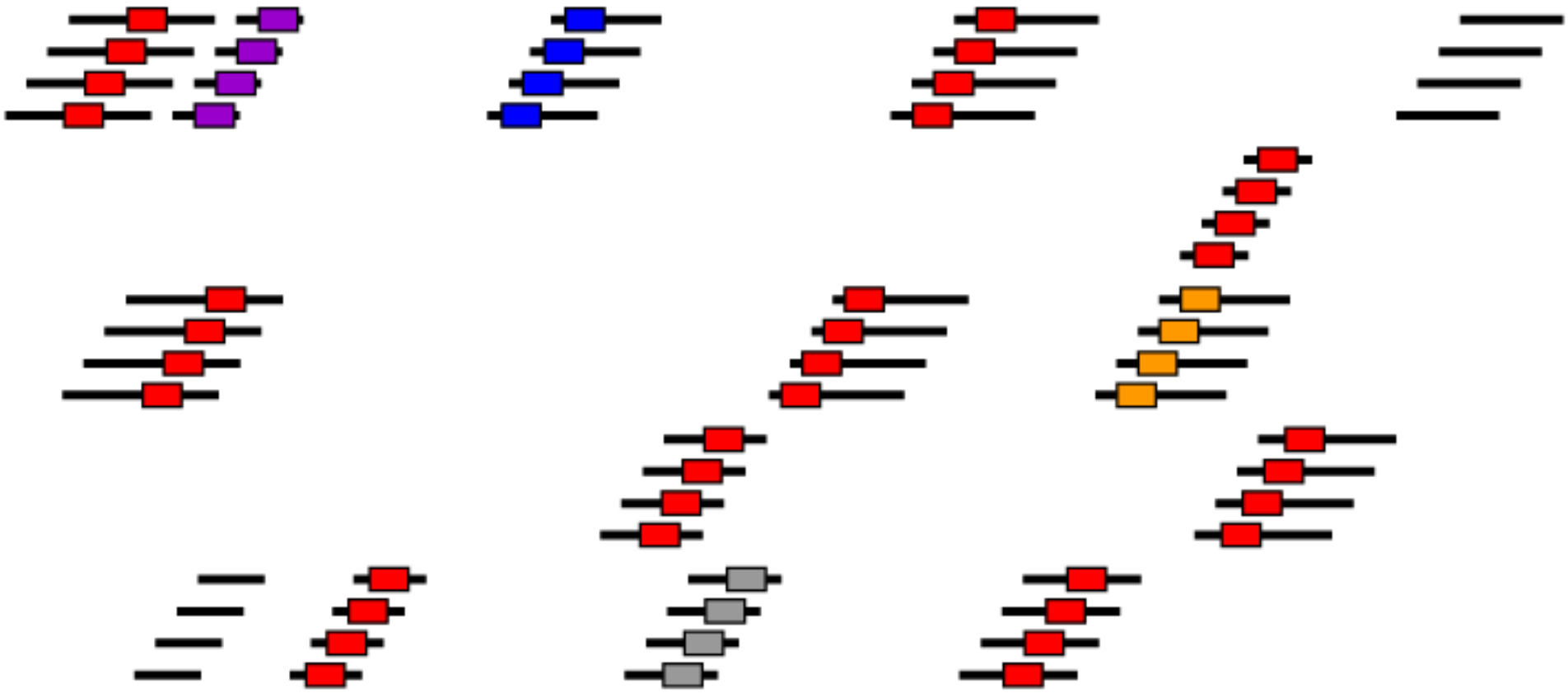
By Richard Bourgon at UC Berkley

Reverse Crosslink and DNA Purification



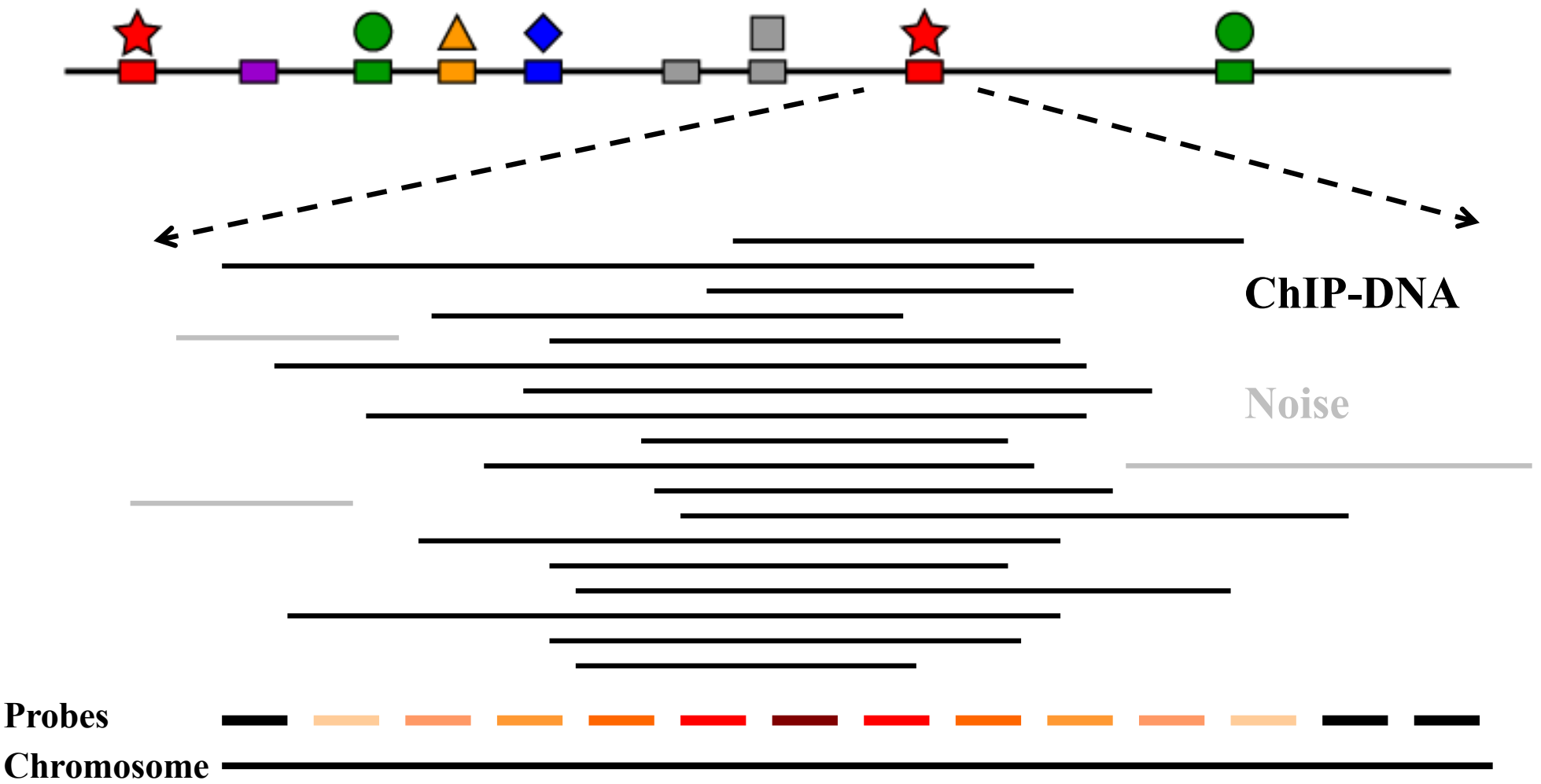
By Richard Bourgon at UC Berkley

Amplification



By Richard Bourgon at UC Berkley

ChIP-chip Hybridization

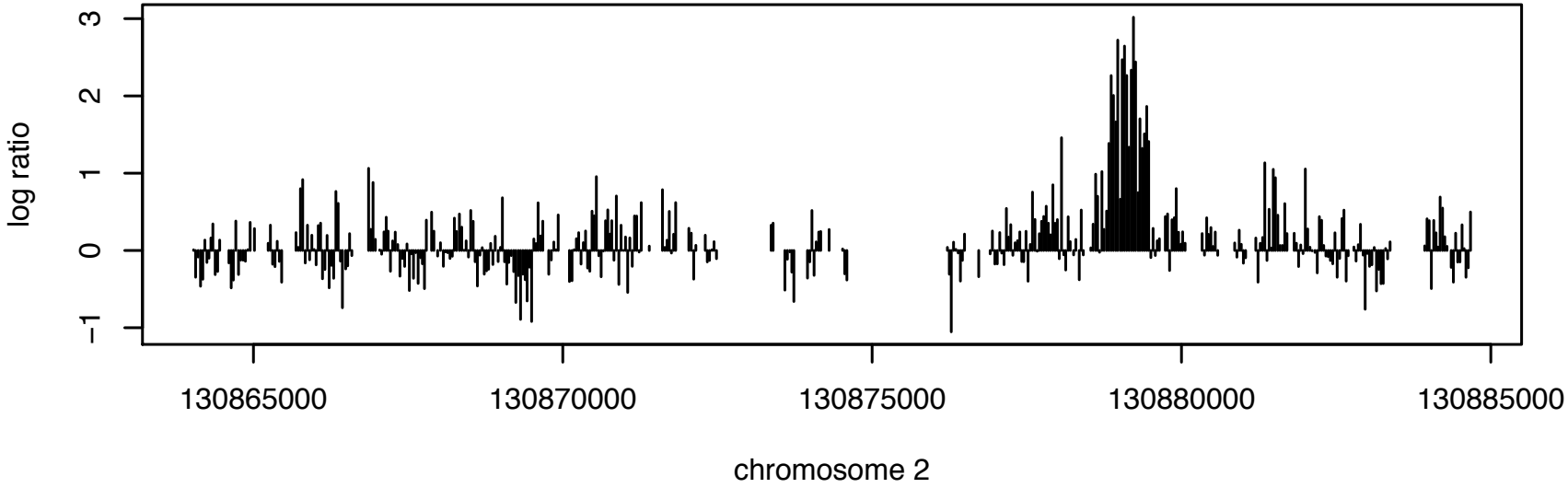


Based on Xiaole Shirley Liu at Harvard

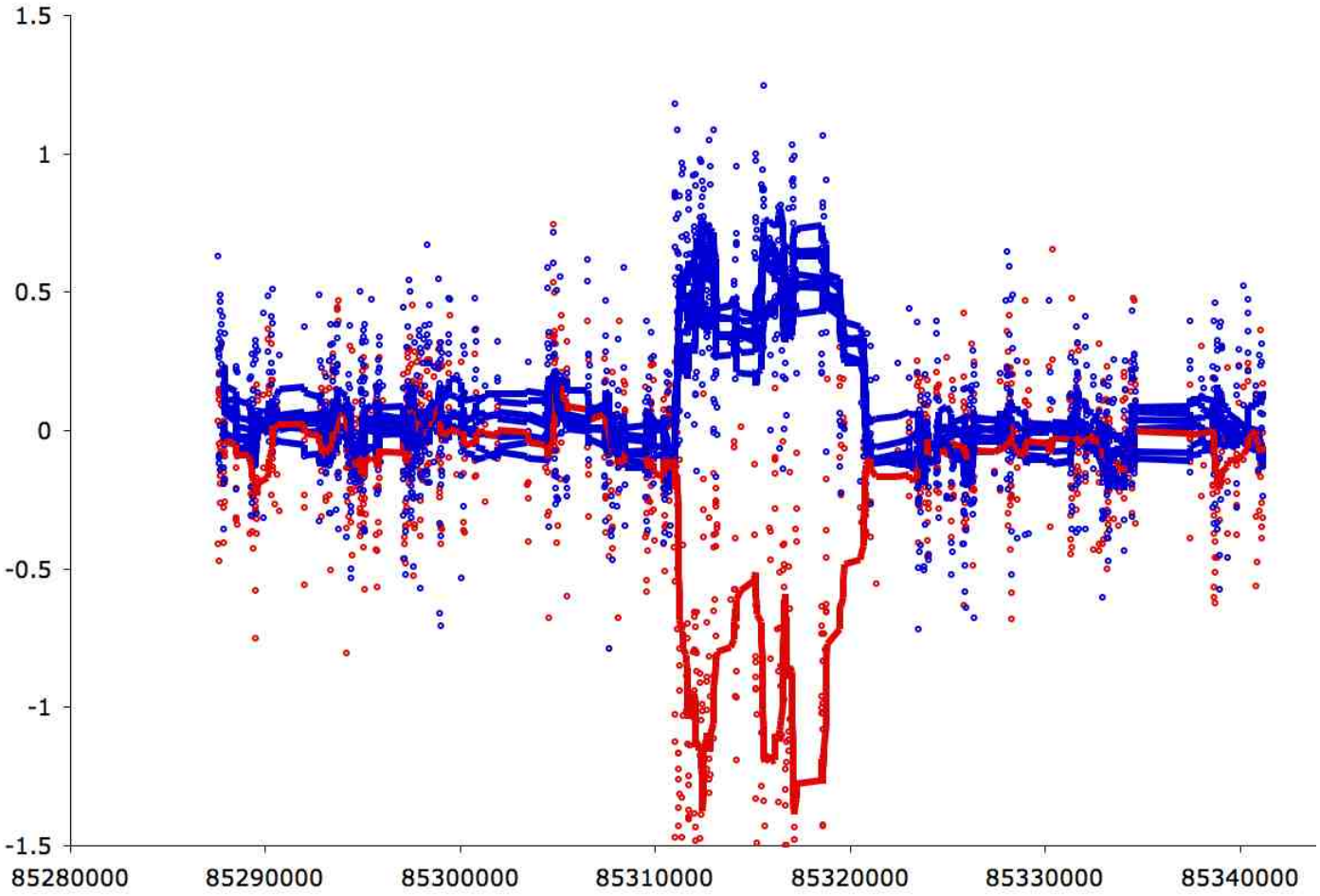
Data from ChIP-chip

- Can be thought as a file with millions of rows and three columns.
  - Each row is for a probe.
  - Columns are chromosome number, probe location on the genome, and signal (intensity values or log fold change).
- To visualize: plot the probe signals against probe locations.

Example ChIP-seq data



Example CNV array data



Tiling array data analysis

- Goal: detect locations of interests (also called “peaks”) based on probe locations and signals.
- Normalization: remove technical artifacts.
- Detection for regions of interests:
  - Data from neighboring probes need to be combined to make inference, because the regions of interests often overlap many probes.
  - Easiest method: moving average, then use an arbitrary cutoff.
  - Many different methods.

Popular software

- For ChIP-chip:
  - CisGenome
  - MAT
- CNV arrays:
  - Affymetrix:
    - APT: uses a hidden Markov model
    - R package VanillaICE: HMM base. R. Scharpf *et al.* (2008) AOAS
    - R package DNAcopy: Circular Binary Segmentation. Olshen *et al.* (2004) Biostatistics
  - Illumina:
    - QuantiSNP: S. Colella *et al.* (2007), NAR
    - PennCNV: K. Wang *et al.* (2008), NAR

Summary

- Tiling arrays are DNA microarrays for detecting locational modifications of genome.
- Probes tile up a part of whole genome.
- Still hybridization based (DNA segments stick to probes), same as gene expression arrays.
- Location of interests shows some patterns: peaks for TFBS, or plateau for CNV.
- Need to combine data from neighboring probes to make calls.
- Being replaced by sequencing (e.g., ChIP-seq).