|  |  |
| --- | --- |
| Bioinformatics | Abstract  Notes on the Advanced Molecular Biology course held by Prof. Francesca Cordero and Prof. Marco Beccuti from the 7th of March 2019 to the 30 of May 2019.  Luca Visentin |

Table of Contents

[1 – Introduction 3](#_Toc11229577)

[NGS – Basic Concepts 4](#_Toc11229578)

[Brief history of NGS machines 6](#_Toc11229579)

[2 - Microarrays 6](#_Toc11229580)

[Agilent technology 7](#_Toc11229581)

[Affymetrix technology 8](#_Toc11229582)

[Whole-Transcript approach 9](#_Toc11229583)

[Illumina technology 9](#_Toc11229584)

[Technologies compared 9](#_Toc11229585)

[Experimental Design 10](#_Toc11229586)

[MA plots 10](#_Toc11229587)

[Data analysis for Microarrays - Bioconductor 12](#_Toc11229588)

[3 – Introduction to NGS 13](#_Toc11229589)

[Illumina library prep 15](#_Toc11229590)

[Target sequencing 17](#_Toc11229591)

[The concept of coverage 17](#_Toc11229592)

[Statistical power 18](#_Toc11229593)

[Concluding remarks 19](#_Toc11229594)

[4 – Data Analysis 20](#_Toc11229595)

[Quality Control 20](#_Toc11229596)

[Alignment and Count tables 21](#_Toc11229597)

[Algorithms for alignment 22](#_Toc11229598)

[Transcriptome Assembly 23](#_Toc11229599)

[Quantification 26](#_Toc11229600)

[Normalization 27](#_Toc11229601)

[5. Differential Expression analysis 28](#_Toc11229602)

[Variance, replication, variability, normalization 28](#_Toc11229603)

[Using linear models for DEA 30](#_Toc11229604)

[Method of least squares 31](#_Toc11229605)

[P-value adjustment – Multiple Comparisons 32](#_Toc11229606)

[Plotting differential expression 34](#_Toc11229607)

[Data Interpretation – GO terms 36](#_Toc11229608)

[GO enrichment analysis 38](#_Toc11229609)

[Gene set enrichment analysis (GSEA) 40](#_Toc11229610)

[Fusion gene detection 42](#_Toc11229611)

[Detection of miRNA 44](#_Toc11229612)

[Other Applications for Deep Sequencing 44](#_Toc11229613)

[Using public resource databases 45](#_Toc11229614)

[Types of data 45](#_Toc11229615)

# 1 – Introduction

**High-Throughput Sequencing**, also known as **Next-Generation Sequencing** (**NGS**) is a technology that can sequence **millions of DNA strands in parallel**. NGS allows for unprecedented sequencing speeds with a fraction of the cost in respect to traditional gel-based Sanger sequencing. NGS also allows scientist to **take a snapshot** of what happens inside the cell at a single time point.

The Human genome project was the key steppingstone for HTS. Remember that the human genome project was made using traditional Sanger sequencing, and cost millions of dollars, however, it provided the incentives to develop NGS technology.

However, it is important to note that **Sanger sequencing is still used today**, as it is **the most reliable sequencing method**. However, NGS data is still essential to gather preliminary information in order to target the costly and slow Sanger method to the correct target(s), or to overview the entire genome/transcriptome at once.

Over the years, the technologies that sequence DNA have evolved immensely, from basic techniques (such as slab gels) with a speed of about 10 base pairs/day to modern techniques that can achieve upwards to 10^9 base pairs a day. The first NGS technique was the **microarray**. The general microarray technique involves a **reference genome**, from where we **extrapolate gene sequences of interest**, and we **design *probes* for those sequences**: short DNA molecules that hybridize to these targets. For example, if we have 20’000 genes of interest, we will create 20’000 different probes (although each probe design will be present more than once in the microarray). We then **anchor the probes** to the surface of a support (usually a wafer of materials, with some anchor layer on top), and we **detect whether or not the probe hybridizes with the target in the cell extract** using the incorporated nucleotides on the target themselves. The target might be cDNA (created from extracted mRNA), or any other DNA molecule. The microarray was the first technology to allow the observation of the whole transcriptome of a cell at once.

In 2007, the big limitation of the microarray, to have preliminary genomic information, was surpassed. **Machines were created that managed to sequence several fragments in parallel** (so, we moved from a *sequential analysis*, to *parallel analysis*). We stepped into the era of **massive parallel analysis**, and thus NGS. The main NGS technique today is **Illumina Sequencing** (but there are also others, such as Roche). The **basic concept for NGS** is to fragment the genome of various cells into **very short molecules** (called “reads” when sequenced), then use (a version of) **Sanger sequencing methods on several molecules at once**. The output of NGS is a very long list of sequences, created by sequencing small fragments of many copies of the genome. It is intuitive that reconstructing a single sequence from several thousands short reads is a highly computational-intensive task.

The **cost of sequencing also went down** with the introduction of NGS, much more quickly than what Moore's law would predict. From 100 million dollars per genome in 2001 to ~1000 dollars per genome in 2017. This cost reduction is very important, as it allows for even small labs to utilize NGS, and to be available for patient-to-patient clinical analysis in order to customize treatment.

The aid of computational techniques began to be essential as the quantity of generated data increased, especially due to NGS techniques that produce gigabytes of data, useless without further processing. So, the field of bioinformatics was born.

Bioinformatics or computational biology is defined as the **development** and **application** of theoretical **data-analysis methods**, **mathematical modelling** and **computational simulation techniques** in the context of the study of biological systems.

Several sequencing consortiums have popped up as NGS technology got easier to use. Recent databases link clinical information of the patients with their genome, in order to extrapolate additional features.

Each NGS instrument, as well as microarrays, has a set of shared attributes with all others:

* **Library preparation** specific to the type of molecule to be analysed
  + For example, we need to retrotranscribe and then fuse with an adaptor sequence extracted mRNAs.
* **Amplification** of the fragments;
* Sequencing reactions (for NGS) or detection (for Microarrays);
* **Data analysis** (bioinformatics)

## NGS – Basic Concepts

The three basic concepts of NGS are **coverage**, **Read length** and **base calling accuracy**:

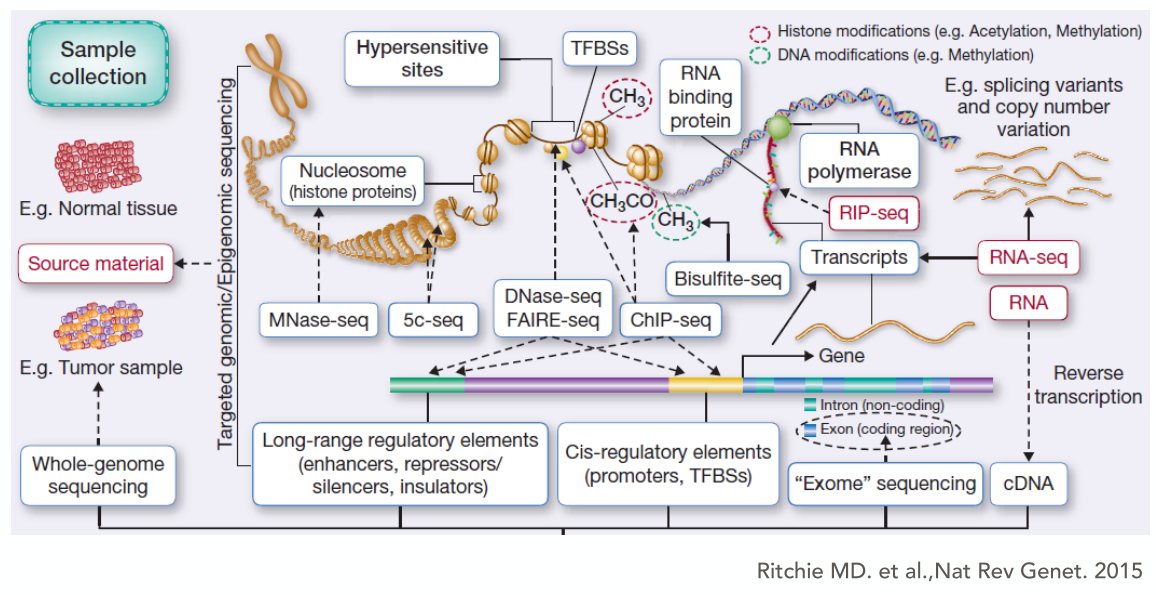
* *Coverage*
  + Depth of Coverage, in layman’s terms, is **how many times a single position in the genome is read** through NGS (also called depth of coverage). The more reads lay on a position on the genome, the more accurate the read is.
  + Breadth of Coverage is “How much of the genome I read”.
  + We would like to have **both deep and broad coverage** in all of the genome.
  + Coverage is related to statistical power and can be calculated with the Lander/Waterman equation (~page 16)
* *Read length*
  + A “read” is a fragment that is sequenced from the NGS technology.
  + Short reads a about 100-250 bp. Long reads are in the orders of thousands of base pairs. It is important to note that *all short reads are given at the same length*. In other words, all reads produced by a single NGS experiment have the same length.
  + Illumina and Roche technologies give short reads, while life biosciences and nanopore give long reads.
  + Reconstruction of a genome (or a generic sequence) is usually **easier with long reads**, and more complex with shorter reads. Long reads might even completely cover the gene/sequence of interest, so that no reconstruction is necessary.
  + Both long and short reads have pro and cons. Long sequences are useful for detection of splicing variants, but are usually very expensive to obtain, as the technologies to produce long and short reads are significantly different. Short read technology is much more widespread, and usually the depth of coverage is enhanced with shorter reads.
* *Base calling accuracy*
  + Each position in a read is **associated with a quality score** (dependant on various parameters measured by the machine, such as pH and temperature). We can analyse the quality by creating several boxplots of each read’s position quality. We then set boundaries for acceptable quality: if the boxplot peak falls outside these ranges, we can see that some problems arose in our analysis, and perhaps even “trim” each sequence to remove low-quality positions. See later in the “data analysis” section.
  + Older machines had high quality for about 15 nucleotides, and quality quickly decreased after that, while new machines can give high qualities for several positions.

Figure 1 – Different targets for NGS techniques.

There are several “-seq” techniques to analyse various targets, such as genes, RNA transcripts, protein-bound regions, splicing variants and more.

NGS is also very useful to **compare various samples**, such as tumours and healthy cells, cells from different individuals (so, different haplotypes...), etc. NGS can also take as an input much more than cell sample, from plasma, urine and saliva to cerebrospinal fluid, so sample acquisition can be much easier than other cell-based techniques.

NGS allows us to discover new things, for example, the microbiome. Through NGS techniques, it has been found that dysbiosis of the microbiome is linked with several diseases, including cancer. NGS allows us also to discover which bacteria are in the microbiome and what are they expressing in a very narrow timeframe. It has been found that the microbiota is "inherited" from the mother to the child, and it follows changes in the microbiota of the mother through the first part of life

NGS is being used to create non-invasive tests for early cancer detection and follow cancer evolution to surpass the challenge of understanding cancer heterogeneity, allowing better design of personalized treatments for chemo-resistant cancers.

The main steps of an experiments with NGS is:

1. **Define a biological question**, which might not be as easy as it sounds, especially when dealing with clinical research;
2. Define what sample types and treatments for this sample are needed to answer meaningfully the question;
3. Define the sampling procedure, including sample harvesting and isolation techniques;
4. **Produce replicates**, usually **biological** (from different patients or healthy humans) and **controls**. For microarrays, we need **technical replicates**, while NGS statistical analysis can be performed without them (although biological replicates need to be available for both technologies);
5. Define the **read type, length, coverage needed**, etc... To balance funding and the required features to answer the biological question;
6. **Analyse data**, which is what we discuss in this course;
7. **Validate** the results using some other method, such as PCR.

## Brief history of NGS machines

Here are some examples of sequencing procedures used in the past, and modern machines used to this day:

* **Manual slab gels**
  + Used manual sanger sequencing and the different lengths of the produced fragments in order to sequence very short spans of DNA.
* **Capillary DNA sequencers**
  + Used the same idea as manual gels but ran the fragments using currents through long, thin acrylic fibre capillaries, and they were detected automatically by their fluorophore through excitation via lasers.
  + Many samples were ran at once.
    - Thermo Fisher 3730XL had 96 capillaries, was able to sequence from 400 to 900 bps, and analysed up to 3840 samples per day. It produced about 2.1 Mb of data for each sample.
* **Roche’s GS FLX Titanium XL+** sequenced 400-1000 bases, with he same samples/day of Thermo Fisher, but with substantial more coverage, at 700 Mb per sample.
* **Thermo Fisher Ion Proton** is a newer machine, able to sequence up to 200 bp reads and outputting 10 Gb per run.
* **Illumina’s NextSeq 5000** is the second to last machine from illumine, able to sequence from 75 to 300 bps and produces 120 Gb per run: that is a whole human genome per 30 hours, with a cost of less than 4000 dollars per genome.
* The newest Illumina machine is **NovaSeq 6000**: it outputs up to 6Tb of data per run, and is able to sequence up to 20 Billion reads in less than 2 days.
* Pacific Bioscience latest machine is **Sequel**, sequencing from 400 to 1000 bps, and, per each cell used, produces from 0.5 to 1 Gb of data.
* Nanopore produces **minION**, a disposable NGS machine based on the nanopore technology. It sequences 70 bps/2 days and produces about 6Gb of data while doing it.

# 2 - Microarrays

Microarrays are cheaper than NGS but also provide less information on what is happening in the genome. They can be used if fully-fledged NGS is too expensive to carry out, or if it is not necessary. As was introduced before, the microarray technology is based on the **design of probes that can anneal to target DNA**. The **annealing is then recorded**, so that the **quantity of target present in the samples can be measured**. This recording is achieved by tagging the library to be analysed with some detectable molecule such as a fluorophore and detecting the signal after the microarray is washed. The targets can vary, for example mRNA (for gene expression and splice variant analysis), miRNA, metilation patterns, and DNA (Copy number changes, LOH, Mitochondria mutations, Germline mutation, somatic mutations...). Note that the microarray has the limitation to know *a priori* the sequence of the target.

What can we discover using microarrays?

* **Differential gene expression between two samples** (at the statistical level, we still cannot compare three samples at the same time, but we can then overlap the result for a set of pair from the different samples);
* **Annotation and metadata** for expressed genes in a sample (we try to get more information about what pathways are involved in these genes);
* Visualization of differences between samples;
  + - To make humanly-readable results we use visualization techniques to show the data and share the data with colleagues.
* Distances, prediction, Cluster Analysis of genes
* Class prediction, meaning to predict in which class (such as tumour type) a sample falls in.

When introduced, microarray technology was a jump forward from typical gene expression tests, such as FISH, RT PCR and gel-based methods.

The key steps for microarray analysis are:

* Know the genome of reference and produce probes for the targets;
* **Link the probes** to the wafer support;
  + Chemical knowledge is involved to efficiently carry out this key step;
  + Arrays are either bought pre-made or custom-produced based on various requirements.
* Anneal the samples and **wash**;
* Record the signal from the microarray;
* **Bioinformatics** analysis to make usable results from the data.

A key step is washing the array to remove all unspecific binding with some solution. The stability of hybridization of the target with the probe to the target is influenced by the **condition of hybridization** dictated by the type of washing solution used. In ***high stringent*** conditions, the target must be perfectly complementary or highly complementary to the probe, as even small mismatches in the target sequence (even SNPs) can prevent stable linkage and be removed by the washing step. In ***low stringent*** conditions, SNPs and other changes are tolerated, and even partial complementarity with the probe can survive the washing step.

If we generate short sequences of random nucleotides, we will find several matches in the genome, just because the DNA “alphabet” is only 4 letters long. So, **short probes show low specificity of their targets**, just because of randomness. Between 60 and 70 nucleotides, the specificity is at a maximum, with it getting lower as the probe gets longer (a longer probe can encounter cross-hybridization). Affymetrix makes probes of length 25, while Agilent and Illumina make probes of about 65.

## Agilent technology

**Agilent** was the first company to produce a microarray. They used **a single probe for each gene**, designed on the **3' of the gene** (which is the most gene-specific region in a gene), of **60-70 nucleotides in length**. They seed these probes onto a plate in **spots**: each spot is a different probe, and it is deposited on the wafer with a **printer-like machine**, with a density of about **10-30k spots per array**. The spots **may be spatially separated in subarrays**. mRNAs or other targets are extracted from the cells, and retrotranscribed into cDNA while being **linked with fluorescent molecules** (usually done during amplification, by linking the dyes cy3, green, and cy5, red). For example, we can label mRNA from normal cells in green, and from cancer in red. The two differently labelled cDNAs are mixed and plated on the microarray, then we take a fluorescent picture. The Red, green and Yellow (Green + Red) colours will determine the quantity of each transcript type.

Due to changes in dye stability and output, among other variables, **both biological and technical replicates are needed for meaningful statistics**. Some steps that add randomness in the microarrays are: the **different incorporation of the dyes** in the cDNA library, the **mixing of the two samples** (which is done by a human, so the proportion of the two samples can be different), and **the scanning**, which needs to detect colours (where colours can be non-homogeneous, can vary, and are difficult to interpret). To reduce errors, samples are mixed randomly with each other, and other methodologies can be followed.

## Affymetrix technology

Affymetrix (produced by Thermo Fisher) produces 1.28 cm square chips. For each gene, **13 different probes of l=25 are designed from different positions in the gene itself** (however, still in the 3' of the gene). Also, **for each probe, an additional set of probes is produced** where every probe contains a single mismatch. In total, we have 2 x 13 probes per gene. Why the use of mismatch probes? In the Affymetrix technique, **the washing process was designed to be able to completely remove a target with even a single SNP from the probe**. The mismatch probe serves as a **control of this washing step**: if the PM (perfect match probe) is positive, and the MM is negative, the process has worked correctly, and the PM is actually a perfect match for the target. If both the PM and MM are negative, the gene is actually not expressed. This means that the washing step is a very highly stringent condition, but the hybridization specificity is low, since the probes are very short. **The low specificity of the probe is taken care of by the repetition of the probe for each gene**. **DNA is still labelled here** (with biotin) before fragmentation. Scanning is easier since we do not need to care about colours, and only pick up grayscale signals. However, this technology is never actually true in practice: both the PM and MM matches are grey, and additional statistical methods need to be used to interpret the data. Usually, the signals from each probe for a gene is averaged and conclusions are extrapolated from that. The MM probes are usually unused if not for a rough background signal. For Affymetrix, **instead of mixing two samples, we just compare the signal between two different chips**, one for normal and one for cancer, for instance. **The necessity for technical replicates is less pronounced than the one for Agilent**, since several steps that add randomness (such as sample mixing) are skipped. The specificity of this chip is very high thanks to the many probes per gene: it can discriminate between genes with up to 90% sequence identity.

Instead of focusing our probes in the 3' of our gene, we can **spread the probes over all the exons in the transcript** to try to determine **changes in splicing**. This technique is called ***gene 1.0***. Another similar technique is ***exon 1.0***, the same for gene 1.0 but with many more probes. We can also create probes that **span over two exons**, so we can detect for specific splicing variants. These techniques all have to increase the number of probes in the chip: the complexity of the chip and of the bioinformatic calculations needed increase quickly.

In order to fit all of these probes on the chip, they are synthetized on the chip itself through photolithography. The silica chip is linked with nucleotides capped with a special photosensitive cap. The cap can be removed by UV light: masks with tiny holes are created and deposed over the chip, and UV is shined on it. New nucleotides are then deposed, and will “stick” only to the portions without a cap. The mask is then changed and the process repeats.

The most used applications of the Affymetrix technology are Gene discovery, Biomarker/gene signatures, Global expression changes, Profiling a large number of genes with relative ease and Genotyping.

### Whole-Transcript approach

To not limit the analysis to only the 3’ of the gene (which is biased in many ways), we can approach the microarray design with a “Whole Transcript” methodology: probes are created for the whole length of the genome, instead of only the 3’ end. This removes the 3’ bias, and allows the detection of transcripts without defined 3’, without poly-A tails, truncated transcripts, alternative 3’, genomic deletions, etc…

## Illumina technology

Illumina created a microarray called Illumina Whole-genome Gene expression Bead Chips. Silica beads linked with an Address-probe construct are placed on a chip. The probe is long, with a single probe per gene, similar to what Agilent does. Illumina makes microarrays for metilation studies and other, where the microarray is superior to NGS.

Library preparation is slightly different from the others, as amplified RNA is needed instead of cDNA:

* Purify samples, and extract mRNA. This is carried out by hybridizing an oligo containing both the poly-T and a sequence for the viral T7 RNA polymerase promoter;
* Produce a cDNA copy of the mRNA, and double-strand it;
* Add T7 RNA polymerase and BrdU-tagged nucleotides to produce RNA transcripts;
* Hybridize the biotin-modified aRNA to the beadChip;
* Stain with cyanin3-streptadivin
* Scan

## Technologies compared

* Agilent
  + **Spotted technology**: Spots with identical probes of l=60 for a single gene on the wafer. Number of spots = 10k /30K
  + No controls in regards with sequence specificity, as the length of the probe is enough to be considered highly specific.
  + Two colours, as two marked genome samples are mixed together. Green, red and yellow.
  + The results are intrinsically comparative.
* Affymetrix
  + **High density oligonucleotides**: High density as N = 500k, organized as sets of perfect match /mismatch probes for 13 places in a single gene (a total of 26 probes per gene, translating to many more probes than the ones used in Agilent). They are seeded onto a 1.28 cm square chip as PM/MM probe pairs.
  + The thirteen probe pairs for each gene are distributed around in the chip randomly. This is done to still use the chip if it gets damaged, for example, by touching it.
  + To place the probes into the chip, we use **photolithographic synthesis**. A wafer of silicon (Si) with free OH groups is covered with DNA linkers having a cap to prevent them to ligate to other nucleotides. However, the cap is sensible to UV light, and is removed when it is shined upon them. By using a mask with tiny holes and shining UV light through it, we can focus it to **only remove the cap on a part of the chip**. Then, **the first (capped in the same way as the linker) nucleotide is deposed**, linked, and UV light is shined again, repeating the process several times to completely create each probe.
  + To save time, **several spots are deposited at a time**: all adenines, then all cytosines, then all guanines..., to save washing steps. This means that the mask will have several holes, according to where in that step (and according to previous steps) the nucleotide has to be deposited onto.

## Experimental Design

All of these techniques are **comparative**, meaning that we need at least two conditions, a test and a control, to compare. Indeed, experimental design and in particular the number of replicates and samples has to be carefully determined in advance. We can be in two cases:

* In the first case, the *direct* approach, we compare two samples by analysing both of them, and comparing the signals produced by each. For example, treated/untreated, treatment1/treatment2 and normal/tumour. We do this in order to test whether our drug does something, if a tumour is different from the normal tissues, what genes are activated during different treatments, etc.
* In the second case, the *indirect* approach, we have an already-measured reference that we can compare to our samples, such as to determine which tumour type we have.

How many and what type of replicates must we use?

The ***variance*** of something is how much that something changes when we repeat the test multiple times. The variance of biological experiments is intrinsically high, but experimental design can be useful to reduce it, for example, by how we compare the samples.

In the so-called **indirect approach**, the samples are compared to a control or a reference, but not with themselves, and have the highest variance (We can quantify this with a symbolic “2”). If we **increase the number of samples**, so we increase the **biological replicates**, for example double them, we can reduce the variance to about 1. So, by increasing the biological replicates, we increase the **power** of our experiments. The “**power**” of any test of statistical significance is defined as the probability that it will reject a false null hypothesis; in other terms, statistical power is the likelihood that a study will detect an effect when there is an effect there to be detected.

On the other hand, the **direct approach** involves comparing our samples to each other, and not to a reference. The removal of the reference lowers the variance, to about 0.67, although the statistical analysis is less intuitive.

## MA plots

We define:

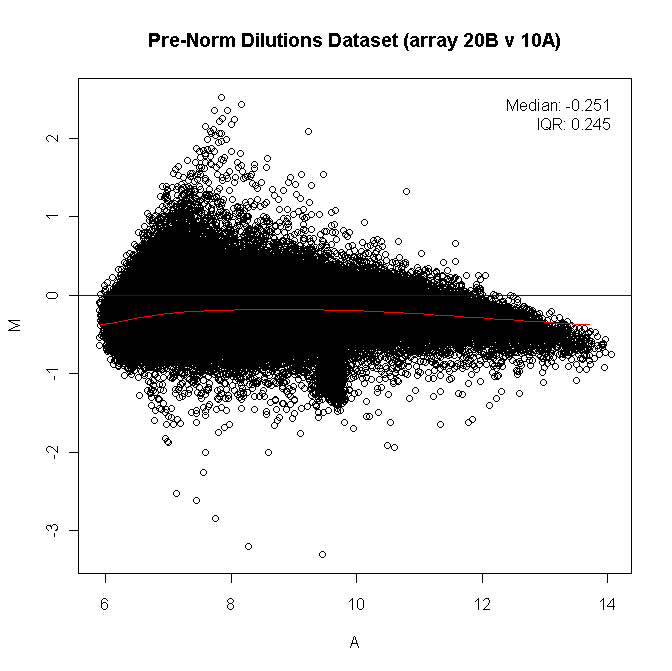
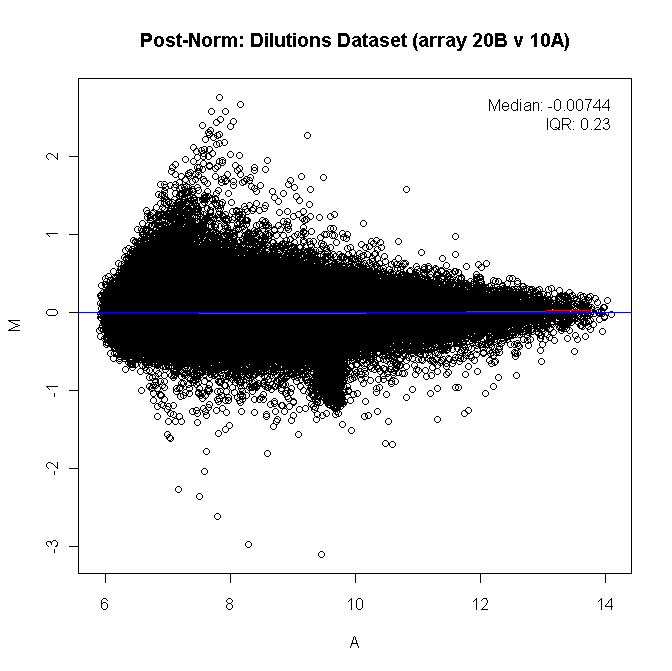
Where is the signal from the “treated” sample, and is the signal from the “untreated” sample. This plot is useful to check for biases introduced by the technology. We expect that the black cloud of dots (one for each gene) will be roughly linear and centred on 0. If it is not, we can correct each gene’s MA value by *normalization*. We will explore normalization options a little later.

Figure 2 – MA plots for some genomic dataset. To the left, raw data without normalization. Note how the shape of the distribution is skewed, and not equal to 0. These characteristics reflect some bias introduced by the technology but can mathematically be removed by normalization (right).

The M/A plot of the average M and A values of the replicates of several genes can be useful to appreciate the ***fold change*** of our genes. We can set 1 and -1 M (Fold Change) values as our thresholds, so we will consider genes that fall outside of these lines as differentially expressed in the samples, while genes that fall inside, not so. By increasing the number of biological replicates, thus decreasing variance, the black cloud of genes gets thinner and thinner. This means that some genes can disappear (by entering inside the boundary), and some genes can become interesting (by exiting the boundaries).

Why is M and A defined as Log\_2 values? It is to **normalize the distance between smaller values and larger values**. For example, if in one case a gene is 2x expressed, and another is 0.5x, the magnitude of the expression differences from 1 are the same, but the mathematical differences seem very small (1 in the first case, while only 0.5 in the second). By computing the logarithm in base 2, the same two numbers become 1 and -1; the same magnitude. This step is done for almost all types of analysis.

**NCBI created the Gene Expression Omnibus** (GEO), a repository of the results of these studies. Each study has an ID, and the samples used get recorded with an additional ID (to characterize the sample, the probe, etc...). The raw data of these studies can be accessed as a spreadsheet; with several header rows (usually marked by a “!”) and a reference ID / probe grayscale result ordered in a table. Each column of the table is a sample (with its reference ID), and each row is a grayscale value for each probe (so, for each gene). These grayscale values can be used to compare the gene expression profiles between samples, as seen previously.

## Data analysis for Microarrays - Bioconductor

To analyse the data from microarrays we can use **Bioconductor**, a free project based on a set of libraries containing several R functions to analyse microarray (among other) data. The steps followed during data analysis are usually:

1) **Quality control**

**We check if there are some chips that we cannot use for statistical analysis**. For example, we can visualize the quality of each sample used in our experiment by **box plotting the mean gene expression levels** of each chip, which should be almost all the same (as the small differences of gene expression between samples are removed by the averaging all the genes). Other metrics that need to be quality-controlled are the scale factor, the number of genes that are “called present”, the 3’ to 5’ rations and the expression of housekeepers (such as GAPDH), and check for RNA degradation.

2) **Normalization**

We want to **minimize the bias introduced by the technology**. If we see an M/A plot without a mean of 0, we should centre the cloud to 0. Plus, if we see biased skewness in some portion of the cloud, we should restore linearity to that portion. After normalization, we should see a linear cloud distribution centred on 0 (Figure 2). How do we perform normalization? One of the most famous mathematical methods is the **RMA (Robust Multi-array Average) method**, created by Irizarry et al in 2003, and it accounts for background correction, computes normalization, etc. Another methodology based on RMA is GCRMA, which takes into account for GC content and other parameters. Anyhow, all these corrections are performed automatically.

3) **Data Analysis/Filtering**

We create one Gaussian distribution for each gene, based on the variance of expression results between controls and treatments. A wide gaussian means very different median values in the two conditions ( is high), while narrow gaussians means smaller variations between the two sets of data. By placing a threshold, we can filter out the very similar (steep) gaussian curves and only keep the significantly wide curves.

4) **Statistics**

Now we only want to keep the very different sets of data. How different are these data, from a statistical point of view? We can calculate p-values to give statistical significance to each curve. This is why we want replicates in these kinds of experiments, without them, we cannot calculate at an enough power the significance. As a reminder, the replicates are both technical and biological: technical replicates are mandatory for Agilent, and strongly suggested for Affymetrix, but not for NGS. Biological replicates are essential in all studies (really, any -omics analysis), and the number of required replicates **must** be determined **before** an experiment is run.

We then consider both the Fold Change value of each gene and the statistical significance of the difference between the two samples. Each are important to select which genes can actually be used for subsequent steps. Indeed, if very low FC is associated with significant P-value, we will not be able to validate the signal in another (more reliable) technology, such as PCR, which might have lower sensibility than microarrays. We place arbitrary thresholds of both P-value and FC value, and only keep data point that fall inside these thresholds.

Statistical validation can either be parametric or non-parametric. The parametric approaches usually have the requirement of normality. The most common type of parametric regression is the linear model. The Bioconductor package for linear models is called “limma”.

The response parameter is the realization of the variable Y (the expression level) for the gene in sample . In a linear model:

Where:

is considered to be normally distributed, independent for each gene and sample, and . In this model, our Null hypothesis then becomes:

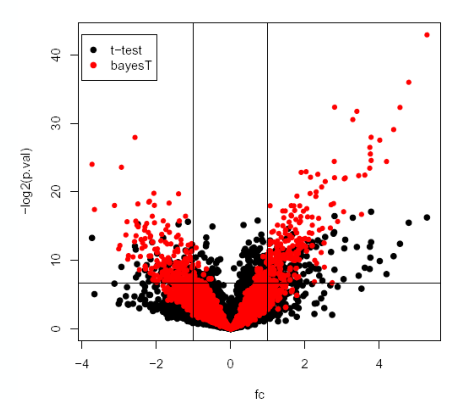
In other words, we wish to check whether or not there are differences between control sample and test sample. We can use a t-test to check this hypothesis and calculate a p-value.

Figure 3 – Volcano Plot of some NGS data. Notice the thresholds both for FC (>1 and <-1) and for P-value (<0.05)

If we plot the in the “Y” axis and Fold Change in the X axis, we get the so-called “**volcano plot**”. We can place the thresholds for both FC and p-value, usually |> 1| and < 0.05, respectively, that we discussed before onto the plot. The thresholds can be more stringent by increasing the one for Fc and lowering the one for p. Note that Bayesian tests are more suited for this purpose than t-tests, and are the ones used by statistical packages.

The number of genes we obtain with this technique are very high, so we would like to overview what pathways are actually involved. This process is called **annotation and metadata analysis** (for example, GO term analysis, explained later).

# 3 – Introduction to NGS

Several platforms are now available for NGS, with different coverage: from a few megabases to tens of gigabases.

The latest Illumina machine is the NovaSeq 6000, which can output the sequence of multiple genomes at once, with outputs of upwards of 6 Tb of data in less than two days. The cost of the machine itself is very high, about 1 million euro per machine.

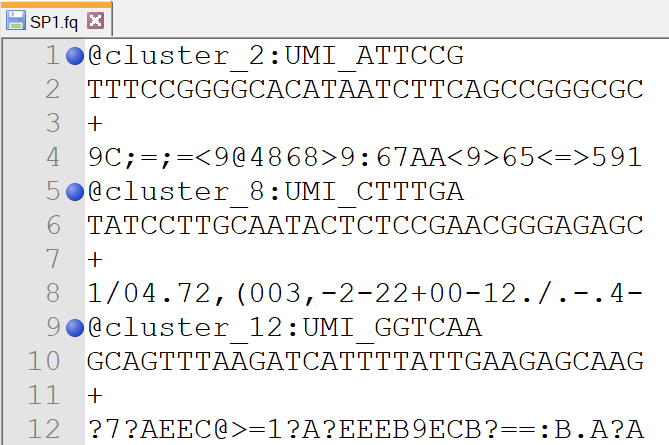
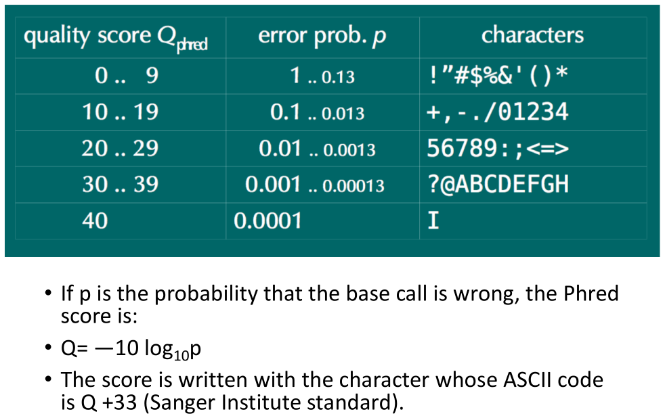
The output of these machines is always the same, a **FASTQ file** (FASTA + Quality). In FASTQ files, each read is identified with an ID, (@ID) the nucleotide sequence of the read (for Illumina, short reads of 50-350 Base pairs), an empty line and the quality.

Figure 4 – Example FASTQ file. The ID is also highlighted with a blue dot for simplicity. Note the format of the data.

The quality value is equal to , where is the probability that the nucleotide written in that position is wrong. The quality score increases as the probability decreases, up to 40 (which is the maximum quality, with a probability of 0.0001). The quality score is then converted into single characters through ASCII code, with, in order from worst to best, these symbols: **(space) ! " # $ % & ' ( ) \* + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ? @ A B C D E F G H I**

The encoding is to optimize file size and subsequent manipulations.

Short reads are easier to keep stable in high-throughput environments, so high quality can be maintained for the whole read. Longer reads (~1000 bp) have bad quality if sequenced with Illumina technology, especially in the final bases.

Examples of long read-producing platforms use very different reactions in respect of Illumina:

* Pacific Biosciences
  + A very tiny well is created on a wafer, where a DNA polymerase is anchored on the bottom of the well. A DNA molecule is fed into the polymerase, and a beam of light is shined on it, and the light re-emission changes based on the introduced fluorophore (linked to a nucleotide), so the sequence can be read.
* Nanopore
  + Uses a similar idea, to use something present in nature to sequence. No well is used, but a lipid membrane. In the membrane there are some modified bacterial porin protein (α-haemolysin, a S. aureus toxic protein) together with an exonuclease. The exonuclease cuts each nucleotide, which falls through the pore: their movement through it changes ionic flow that is passing from the pore in a nucleotide-specific way, which can be detected so the sequence is recorded.
  + The nanopore machine is very small, akin to a smartphone. Their idea is for small, disposable machines for NGS.
* ABI
  + Again, based on a completely different type of sequencing, for short reads.
  + They obtain the sequence of the read, with an id, but not expressed in a nucleotide alphabet (A, C, T, G), but as colours (0, 1, 2, 3). However, we won't talk about this technique, as it is pretty old and not popular.

## Illumina library prep

We will now see the Illumina library prep. It is important to note that all platforms have different library preparation methodologies, but since Illumina machines are the most used, we will focus on Illumina library preparation.

Some initial remarks:

1. We work with a **set of cells**. Some other preps can derive information from a single cell, but we will discuss methods that use several cells.
2. We can work with **either DNA or RNA** (and the library prep steps will differ).
3. We will obtain **substrings of either DNA or RNA**, meaning short stretches of sequences (reads). All NGS technology creates a **set of reads**.

There are four steps for library prep. Sample preparation, Cluster amplification, Alignment and Data Analysis.

1. Sample generation
   1. Our biological DNA sample is fragmented and linked with adapters. This step is called *tagmentation*.
   2. Specific enzymes (transposases) are used to both insert the adapter sequence and cut the DNA.
   3. The adapter has a particular structure: **a sequence primer both at 5' and 3'**, followed by an *INDEX* nucleotide sequence (index 1 and index 2) and at the end, a region complementary to the anchor in the flow cell.
   4. The flow cell is the "Affymetrix chip" of the Illumina technology, essentially a wafer with probes attached in a pretty random fashion. The flow cell is split in 7 lanes. On each lane, complementary anchors with the end part of the adapter are linked with the wafer and can capture the DNA molecules of our library thanks to the anchor sequences.
2. Cluster amplification
   1. We need amplification of our sample. This is a critical step that can go wrong and invalidate our data. Having several copies of our molecule will allow the machine to both get a strong enough signal to detect and provide safeguards to technical errors (such as the incorrect base being incorporated).
   2. The flow cell links with the sample using the anchors. After amplification, we will obtain **clusters of clones of the DNA fragments close to where the sample was originally anchored**. The cluster allows for quality detection of the sequencing step: if a strong, clear signal from only one fluorophore comes from the cluster, we are pretty sure that is the correct nucleotide in that position. On the other hand, if a “dirty” signal, made of multiple wavelengths comes from a cluster, that position will be associated with a lower quality.
   3. The amplification starts with the creation of complementary sequences, using the anchor attached to the wafer as primer. The original sample is washed away. The other adapter now will bind to another anchor, next to it. The polymerase now elongates again, and when the denaturation step opens up the bridge, the cycle then repeats and ultimately creates a cluster of forward and reverse strands.
   4. The reverse strands are removed by cleavage with a specific enzyme.
   5. The 3’ ends of the reads are blocked to prevent unwanted elongation of these portions.
3. Sequencing by synthesis occurs. As sequencing by synthesis is a proprietary technique, we can only speculate how it is carried out. A sequencing primer adapts to the top of the wafer, and DNA polymerase is added (maybe). Sequencing following a Sanger-like method then follows:
   1. Add the four terminally-blocked, fluorophore-tagged nucleotides and wait for elongation.
   2. Wash away free nucleotides.
   3. *Snapshot* the fluorescent dots now created for each cluster. This fluorescence is recoded, and the corresponding nucleotide and quality is noted.
   4. Unblock the nucleotide, to allow for elongation.
   5. Wash the unblocking agents.
   6. Repeat steps I-VI n-1 times, where n is the read length.
   7. After n cycles, we obtain reads of length n.
4. The read product is washed off. The index primer is then added, and the index is sequenced by synthesis. This is important in multiplexing (see later). The products are washed off.
5. For paired-end reads, another cycle of cluster amplification follows, to create the reverse strands. The 3’ ends of the reads are uncapped, and the molecules bind as a bridge to the reverse anchors.
6. The index of the reverse strand is read, using the anchor as a primer.
7. Polymerase action elongates the molecules, creating the reverse strand.
8. Again, sequencing by synthesis occurs, to sequence the reverse reads.
9. Alignment and data analysis
   1. We have our FASTQ file. The reads are associated with the quality of each base (encoded in hexadecimal characters). The quality is how confident the machine is of that base read.
   2. How do we obtain the quality value? For each cluster, we have several copies of the reads, associated with a fluorescent signal each; Clean, strong signals for only one base will be associated with high quality, while “dirty” signals will be marked as low-quality. The maximum quality is 40, and the values will decrease as more “mismatches” are detected for that base.

How can we use indexes?

The flow cell has seven lanes: we could put one sample in each lane, so, 7 samples total for one flow cell. This is highly wasteful: a single lane has the capacity of sequencing far more than a single sample. Using indexes, we can put different indexes for several cell samples, and put all of them in a single lane. This allows for sequencing multiple samples in the same flow cell lane (multiplexing).

After sequencing, the reads are *demultiplexed*, by clustering together reads with the same index, so we “reconstruct” the samples. We can then treat the two clusters as different samples, as if they were sequenced by themselves. We can analyse 14 to 20 samples per lane using multiplexing.

We can obtain two types of (short) reads:

* **Single end reads**
  + What we discussed until now. The sequence of interest is short, 50/100 base pairs, set by the experimenters. The sequence created synthesis is generated starting from a single primer, and we read all the bases of the sequence of interest, from the first base to the last, in one set of sequencing cycles.
* **Paired end reads**
  + We are able to read two different sequences from a single fragment. The region of interest is very long, but quality decreases over time as more bases are introduced. So, we can use two primers to create two reads (read/1 and read/2) of about 50/100 bp long, one at the 5’ and one at the 3’. We will then have a gap of DNA uncovered by the reads (DNAi), which is usually ~100 bp long (Short-insert paired end).
  + To follow the DNA synthesis direction (5’ → 3’), one read is done from the FW strand, while the other from the RW.
  + Why can we do this? We would have gaps in our reference, wouldn't we? No! Since we use different cells, there will be a lot of overlaps between our reads, so we can correctly align the sequences, with no gaps.
  + In reality, the information that the two regions are connected, with some fixed distance, gives us powerful new data for alignment: if one of the two reads aligns, but the linked one doesn't, we will be in the wrong position in the genome.
  + This is also highly useful for exon mapping: instead of having to have exon-overlaps in the reads themselves (which are short), we can just have read/1 on one exon and read/2 inside another (and we will have also reads spanning the region themselves, plus the other linked read on the exon).
  + The DNAi “gap” can be much longer than 50/100 bp (long-insert paired end).

With **De Novo sequencing** we refer to the sequencing of reads that have no reference genome, so the goal of sequencing is usually to reconstruct it. Here, we want to have both long-insert and short-insert pair end reads, and with all of these data we can reassemble the genome much more easily than with non-paired short reads.

## Target sequencing

Another sequencing method is target sequencing. If we don't wish to have the global information of whole DNA/RNA preparations, but only of a set of genes/transcripts, we will have a new step to select only the molecules of interest before sequencing.

For example, consider ChIP-seq. Here, the sequence of interest is bound to a protein: we can select them with an antibody against that protein. We can also select sequences using other types of enrichment techniques (such as DNA/RNA probes). Another method for target sequencing is using PCR. If we know exactly the sequence of our target, we can amplify the target region and sequence it through NGS. This is useful for highly variable regions, such as the immunoglobulin locus after maturation (which can be monitored to follow leukaemia progression). The primers are what will select what region will then be sequenced. Note that the number of reads needed for this technique is very different than before (so the throughput is much different than whole-genomics analysis).

To sum up:

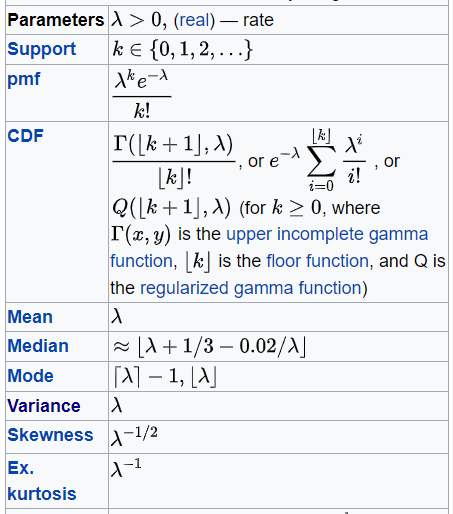
* Each step of the technique is selected in basis of the biological question. How do we enrich the library? We do pair-end or single-end? How many reads do I need to detect what I want?

## The concept of coverage

We can define coverage using the Lander/Waterman equation:

Where is the coverage, is the length of the read, is the number of reads and is the genome size. The coverage can be interpreted as the number of times a single base in the genome is read by the sequencer. The larger the coverage, the “surer” we are that that base is actually the base read.

The Lander/waterman equation uses the Poisson distribution to compute coverage. This distribution has two major assumptions:

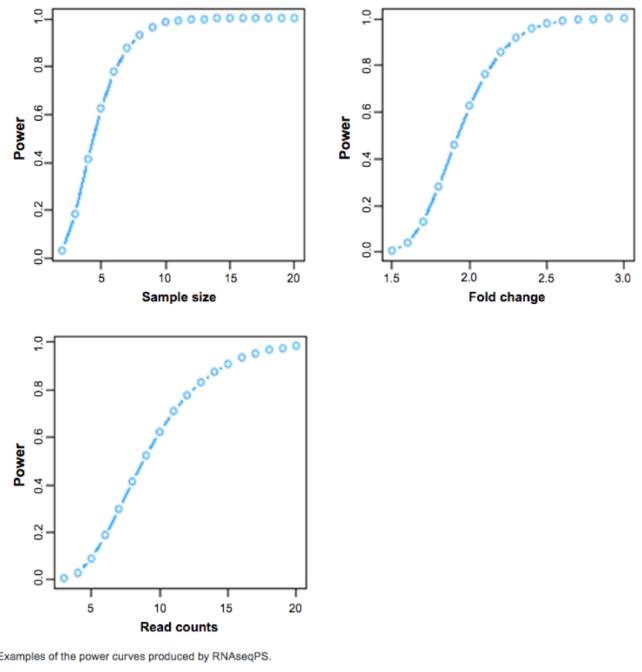
* Reads are distributed randomly across the genome
* Overlaps don’t vary between reads, meaning each read is independent from the others.

The distribution’s formula is:

Where is the coverage and is a natural number representing the probability that a base is sequenced exactly that number of times.

To calculate , so the probability that a base is read or fewer times, the Poisson distribution can be summed. So, we calculate each whole number lower and equal than and sum the probabilities:

## Statistical power

The power of any test of statistical significance is defined as **the probability that it will reject a false null hypothesis**. **Statistical power** is inversely related to beta or the probability of making a Type II error. In short, power = 1 – *β*.

In plain English, statistical power is **the likelihood that a study will detect an effect when there is an effect there to be detected**. If statistical power is high, the probability of making a Type II error, or concluding there is no effect when, in fact, there is one, goes down.

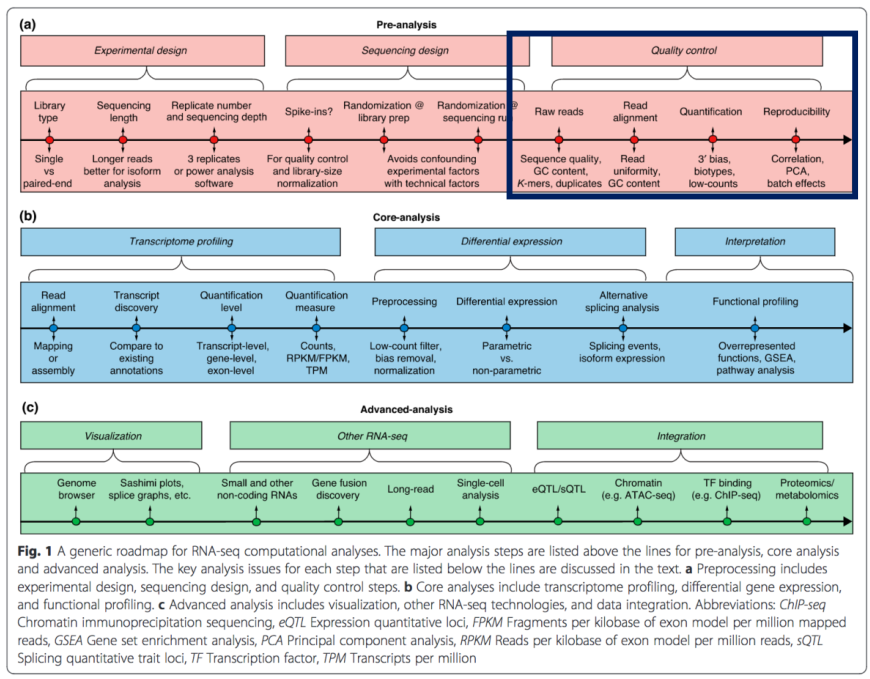
Statistical power is affected chiefly by **the size of the effect** and **the size of the sample used to detect it**. Bigger effects are easier to detect than smaller effects, while large samples offer greater test sensitivity than small samples.

The number of (biological) replicates *per group* is essential for determining the statistical power of NGS. It increases as both the FC considered increases, and the number of replicates increases. It is also influenced by sequencing depth: more coverage means increased power. Higher statistical power is also translated to better extrapolation to the population, and indirectly, to other cohorts. The required power for any analysis should be computed *a priori*.

## Concluding remarks

* **Types of targets**
  + Poly-A selected (mature mRNA)
  + Exon-capture
  + Ribosomal-depleted miRNAs
  + Other capture methods such as BrdU (GRO-seq)
* **Sequencing length**
  + From 50 to 75, 100, usually. Long read technology can sequence up to thousands of bps.
  + For a simple list of Differentially Expressed Genes (DEGs), single end of 50, 75 bp
  + For detection of isoforms, paired end, as long as possible reads (75, 100)
  + Exon capture, Paired ends 50-75 bp
  + miRNA, Single End 50 bp
* **Sequencing dept**
  + Gene-level 20-30 million reads
  + Isoform-level 80-100 million reads
  + rRNA depletion 80-100 million reads
  + miRNAs 1-3 million reads

# 4 – Data Analysis

**We will focus on data analysis for RNA-seq.**

## Quality Control

NGS provides us with FASTQ files, containing both the reads and the associated quality information. The first step in the data analysis pipeline is quality controls. Reads are trimmed or discarded to remove technical errors, assuring that only high-quality reads are analysed in successive steps.

To maintain high quality data, we want to discard two different types of reads: reads with **low base quality** and reads that are clearly **artefacts of sequencing**. To filter out low quality bases we can use the Java tool FastQC (Fasta quality control). It takes the FASTQ file and creates several boxplots of the quality score, one for each read position. Bases that have low quality scores can then be removed, either by discarding the whole read (not commonly done, as almost all the reads would be discarded by this), or by trimming, i.e. deleting, the incriminating nucleotides. As sequencing becomes harder towards the end of the reads, due to multiple factors such as availability of nucleotides and polymerase stability, the last few bases are usually trimmed as they show low quality.

Another read to be removed are those that contain only adapter sequences. Sometimes, when ligating adapters, they fuse to each other. These reads are obviously garbage. FASTQC can remove also these kinds of reads.

FASTQC also shows the base content per position. The first positions usually show artefacts of incorporation, showing very different base composition than the expected 25%. They can be trimmed away. However, not all analyses are deep enough to actually show the expected 25% incorporation per base: for example, miRNA analysis can show biased incorporation.

In some sequencing settings, highly similar or identical reads can be filtered out as they are considered to be artefacts of PCR amplification. However, somewhat high duplication is to be expected in RNA-seq.

## Alignment and Count tables

Reads are then aligned to the reference genome. The reference genome is annotated with genes, and every read that falls on a gene body is recorded in a *gene table*. A gene table contains every known gene and a score (how many reads fall on that gene’s body) per sample. We can then map this expression on a plot.

This plot will contain an axis for every gene taken in consideration, together with a point for each sample. Consider that the sample number can be as low as 6 for “bulk” RNA-seq, and as high as thousands for single-cell technology. The number of genes is always very high, around all the genes annotated in humans, 20000.

How can we represent such a graph? It would be gene-dimensional! So, we have to **reduce the number of dimensions**. There are several techniques for this, the most common one is PCA, or Principal Component Analysis.

PCA reduces the number of dimensions by discarding some information. To discard as little information as possible, we look at the **variance** added to the data by each dimension. First, a line is fitted in the data, in such a way that the points, if they were projected to the line, showed the maximum possible variation. This is called Principal Component 1, or PCA1. We can then add a second PC by fitting a second, perpendicular line to PCA1 again in order to maximise the variance of the data when projected on this line. We can continue adding PCs for as long as we have dimensions.

To measure the variance contributed to by each PC, we determine an Eigenvector (a vector that has a scalar module and which direction is unchanged after the transformation). When projected onto each PCA, the module of the Eigenvector gives the Eigenvalue (EV), or loading score, for that PCA. The larger the Eigenvalue, the more variance the PC accounts for in the data.

We can the sort and display each EV for each PC and see them in a plot such as a histogram. If few PCs show large EVs, we can discard the other PCs and still conserve a meaningful amount of information but reduce the dimensionality of the data. If all the EVs are similar, then losing one dimension will cause a high loss of “meaningfulness”. If this is the case, other, more complex methods of reduction of dimensionality are available, such as t-distributed stochastic neighbour embedding.

Aligning the reads is a difficult task. Why?

* The reads are short, and the isoforms are many.
  + How do you reconstruct different splicing alternatives if they all overlap? Reads that fall entirely in a single exon, which can be included in many different isoforms, are not informative as we cannot know where they originated from. There are specific techniques that take into account exon overlapping reads and return quantification. See later steps.
* Error rates are considerable
  + New machines give high quality reads throughout the read, but the error rates must always be considered.
* Many reads span exon-exon junctions, and are difficultly mapped locally, due to the massive gap corresponding to the introns.
  + Reads completely inside an exon are called "encompassing", while reads between them are "spanning". Spanning reads are the key to solve the problem of finding different types of transcripts and to quantify their expression.
  + 10M reads are enough to quantify gene, but not isoform, expression, but 80-100M reads are required to discern isoform types and quantity.

## Algorithms for alignment

We have two main types of alignment algorithms, whole-read (unspliced) aligners and spliced aligners. As the name suggests, the two types of aligners deal with different problems.

* **Type I: Short, unspliced aligners:**
  + Based on two main methods, Seed-and-extend and Burrows-Wheeler Transformation.
* Seed methods
  + The seeding methods are similar to the sliding approaches seen with Beccuti.
  + We define as the number of mismatches allowed in a read, and is a very small value. For example, k=2 and m=1
  + We **segment the reads** (not overlapping, like a sliding window, but a simple “cut”) in segments, and we produce, considering the whole reads, all possible “subreads” containing at maximum k mismatches. The segments that contain the mismatches are then masked. The remaining fragments are “seeds”.
  + We anchor each seed where it **perfectly** matches in the genome and then use more specific algorithms, such as Smith-Waterman, to extend the alignment region in the masked portions. We then select the positions where the reads better align.
  + The main drawback of this method is that it is time consuming, especially if we want more sensitivity by increasing .
* Burrows-Wheeler transformation
  + It is at the base of Bowtie, one of the most used aligners, and the Burrows-Wheeler Aligner (BWA).
  + The BWT is a lossless method of compressing text. In this setting, it compacts the genome into a smaller structure, but without any information loss.
  + Aligners based on this method are very fast but are usually bad at handling mismatches. While Seed methods can work with upwards to 10 mms, the BWT-based methods can handle (and still pretty badly) at maximum 2 mismatches. However, the reads that show high mms are usually spanning reads and can be discarded at this time and analysed with other methods.
  + If we use a transcriptome instead of the genome, we can easily map even spanning junctions using these aligners, however, missing information in the transcriptome can lead to huge biases and errors, so the proper handling methods must be used.
* **Type II: Spliced Aligners**
  + They deal with exon-exon junctions, mainly by splitting the read in smaller pieces and aligning those instead of the whole read at once. They can either use an annotated genome to help with the alignment task, or they can perform it *De Novo*.
  + **TopHat** is one of the most famous spliced aligners and can work both using an annotated genome or *De Novo*. First, the reads are **mapped to the exons**, with very low (usually zero) mismatches by using bowtie. We then create a reference of exons that are mapped in the encompassing reads.
  + Second, all the reads that are not perfectly aligned are considered to be exon-skipping, are separated in **small k-mers** and aligned to the end of the exons perfectly.
  + We then reconstruct the junctions between them using the links between the k-meres.
  + This method removes the bias of using algorithms such as bowtie with a transcriptome, as the transcriptome itself is constructed by the aligner.
  + A potential pitfall is pseudogenes, as they seem exons stitched together, so that actual spanning reads seem to be encompassing.
* Seed-extend methods
  + Same idea of seed methods, but we first split the read into k-meres or other small pieces and then align them heuristically and then reconstruct the junctions using this new information. The seeds are allowed 0 mismatches, and the mismatches are handled by the high repetition of seeds.
  + These algorithms can directly find splicing and detect more isoforms than the exon-first methods (but we are not sure they are real isoforms), but take an enormous amount of time, as the reads need to be divided into smaller fragments.
  + Additionally, the biological meaning of these extra transcripts has not been confirmed.

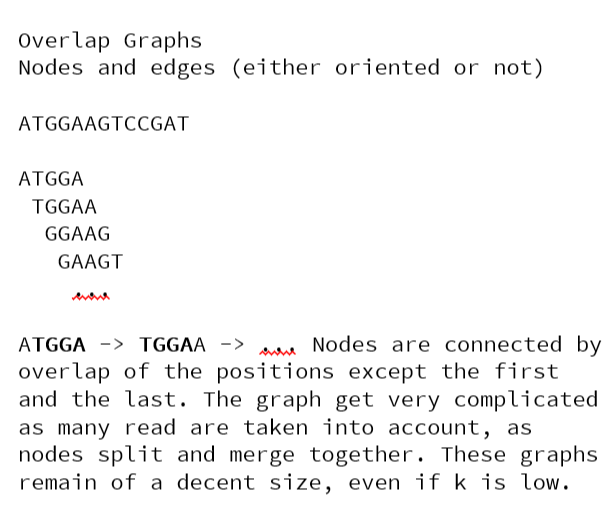
## Transcriptome Assembly

This problem is again very complex, much like the alignment. The main problems are:

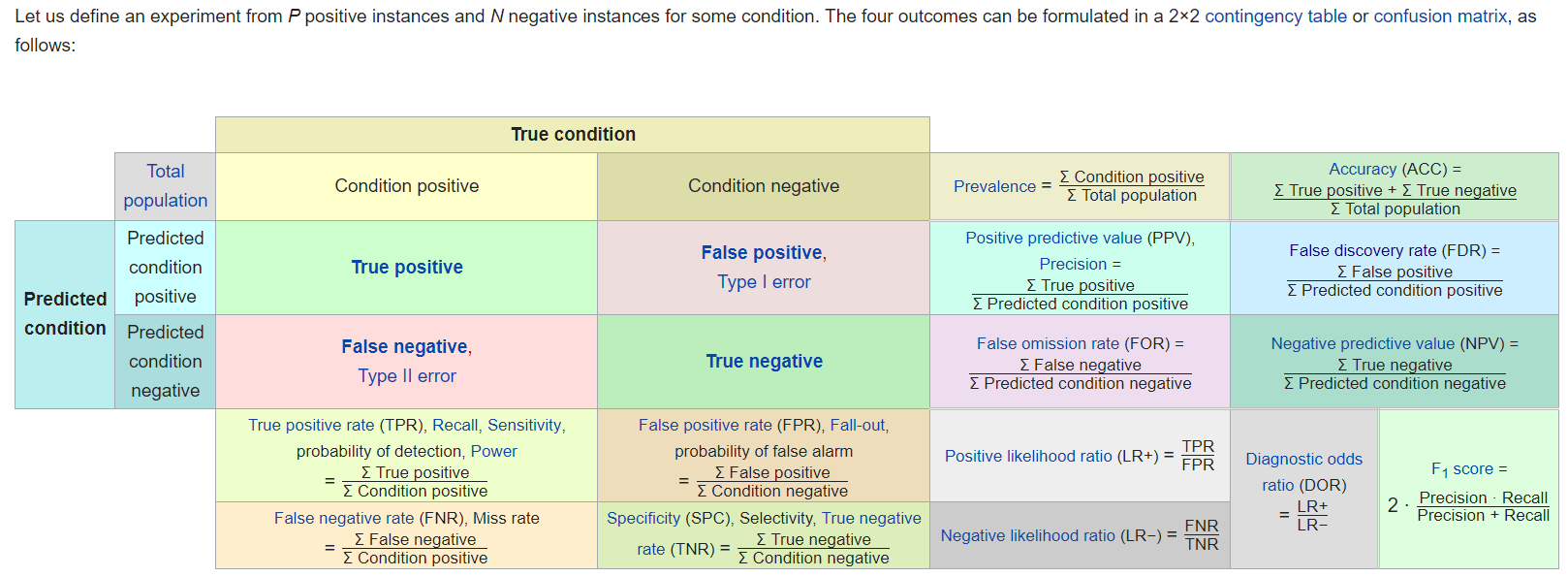
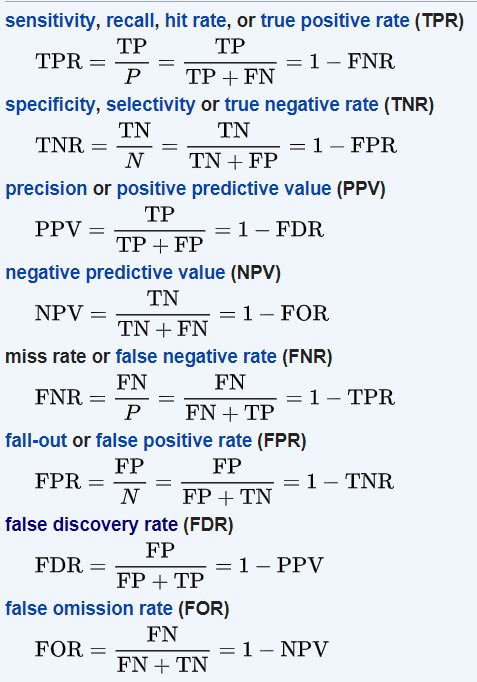
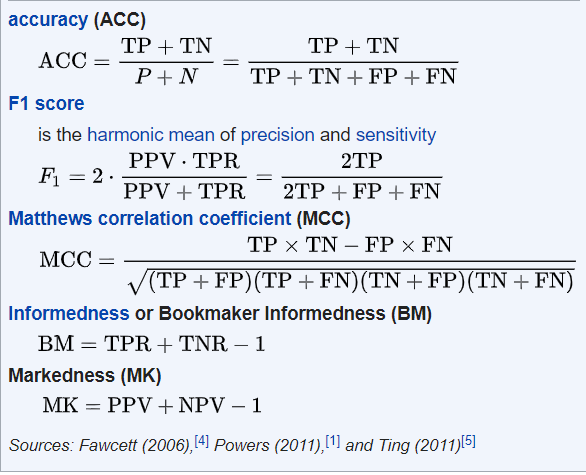
* Genes are differentially expressed. Even with a smooth coverage, some genes are highly expressed, and some are very low expressed, and most data (even when depth is increased) is absorbed by highly expressed genes.
* The changes in expression span several orders of magnitude.
* Reads originate from both mature mRNA (with Exon-exon junctions) and immature RNA (still with introns).
* Reads are short and there are many isoforms, so they overlap.

We have two main approaches in the reconstruction of the transcriptome: If we have access to a (annotated) genome of reference, or we if we don’t.

The methods that rely on a reference genome in order to reconstruct the transcriptome are called “Genome-guided reconstruction methods”. There are essentially two approaches in this category, Exon identification:

* **Exon identification**
  + Developed for early short fragments, where the spanning reads were few.
  + We align the reads on the exons and fix them as islands. We then use the information from the spanning reads to reconstruct the actual map of isoforms.
  + They are underpowered when dealing with full-length structures of low-expression, long or alternatively-spliced genes.
* **Genome-guided assembly** (so it is called Genome-guided-genome-guided assembly)
  + Both algorithms use the idea of *overlap graph*s. A graph is a computational structure of nodes and edges, the latter being either direct or indirect. Here, each node bears a small sequence, extrapolated from the genome of reference.
  + **Scripture**
    - It transforms the genome into an overlap graph where each base is linked with other bases either because they are next to each other, or when they are linked by a spanning read.
    - The nodes that are overlapping a junction end up with very low edges, so we can extrapolate a map containing all possible exons splicing isoforms.
    - The isoforms of the gene present in the cell must be included in a path in the graph, so this algorithm has the maximum sensitivity.
    - We can then separate each graph into the maximum number of single isoforms (maximal set), and all of these isoforms are compatible with the data.
    - Scripture considers even a single spanning read enough to create a breach point for different isoforms.
    - After the graph is built, scripture returns all possible paths that are compatible with the read data (that is, that could have originated the reads). Therefore, it has the highest sensitivity, meaning that all actual transcripts are included, but not all included transcripts are actually real.
  + **Cufflinks** 
    - It also uses an overlap graph. It connects two string because there is an overlap between them. We split the reads into k-mers, put them into nodes, and connect them if they overlap. Such a graph is constructed from the reads.
    - In the graph, all possible isoforms of genes are included, so it has the highest precision.
    - These graphs explode in size as k diminishes.
    - We then choose a first node and move along the graph to create paths. We then use the reads to create paths from point to point to recreate isoforms, reporting the minimal number of compatible isoforms. So, we report the minimum number of paths that are compatible with the data. Therefore, cufflinks has the maximum precision: all the included isoforms are actual isoforms, but not actual isoforms are included.
    - Cufflinks also uses the read coverage as a method of discarding paths that are not supported by the data.

So, cufflinks has the maximum ***precision*** while Scripture has the highest ***sensitivity***.



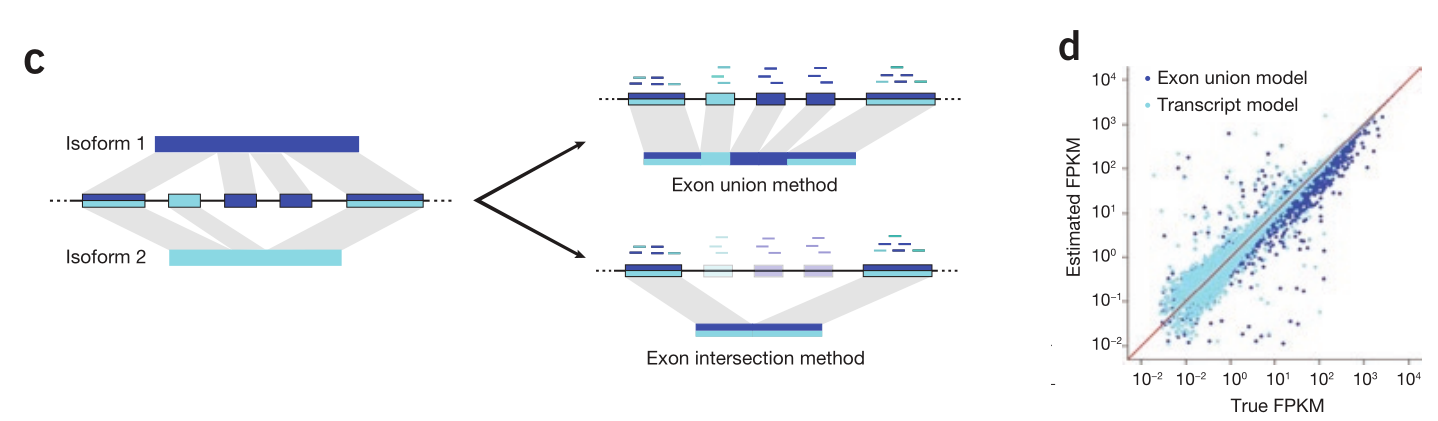
Genome-independent:

* Use a *de Brujin* graph. We split the reads into k-mers and we use them to produce a de Brujin graph, very similar to an overlap graph, but built specifically for pair-end reads, and reads in general.
* The lower the k, the higher the graph complexity. To balance the choice of k, we can take into account the coverage: smaller k-s are fit for low coverage, as we "increase artificially" the sensitivity. For deep coverage, low k-s can create huge "complete" graphs (where every node connects to every other node, and we lose all significance of the graph), so a high k is preferred.
* When we have a graph, we recreate the possible isoforms which can then be mapped to a genome of reference. We first use the reads, especially spanning reads, and the coverage in order to prune connections which are false. Each remaining path is then reported as a possible transcript.
* The limitations of these algorithms are that they cannot distinguish sequencing errors from variations (since they don’t use a genome), and the balance between k and graph complexity is very difficult to find. How do we choose k? in general, if the coverage is low, k should be low, as more k-meres will be generated in order to better find links between the isoforms (a sort of “artificial” increase in read number). When the coverage is large, k should be large, as a small k with lots of reads will create an overly complex, and therefore useless, graph.

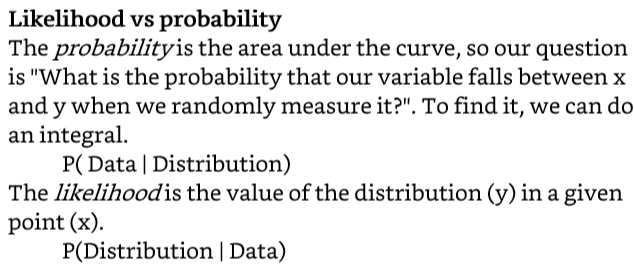
## Quantification

We now wish to determine and quantify the *expression level* of a gene. We can either extract from the reads the whole-gene expression, or the expression for each transcript isoform. Essentially, we need to assign the reads to a gene or a transcript.

We might wish to simply know the expression of a certain gene, without considering the different expression of the gene’s isoforms (if any). There are two main methods to translate reads mapping on genes into quantities: The first is to just count all the reads among all of the gene’s exon (exon union method); another method is to only consider the regions in common between isoforms (exon intersection method), and only count those reads for quantification. However, simplicity comes at a cost: the union model usually underestimates the expression, especially for highly expressed genes, while the intersection method, since it discards many of the genes, reduces statistical power.

If isoform-level data is available, the *true expression* measure or transcript model, or isoform deconvolution method, can be used to determine the expression of the gene by summing up the normalizes read counts for every isoform. This method, accounting for all isoforms, is better suited at estimating gene expression than the simple exon union/intersection models.

The quantification of reads to single transcripts is more difficult as reads that map to the same position in the genome can overlap different isoforms. However, we have several methods of determining the number of transcripts for each isoform, one of the most used is the **Expectation Maximisation** method (EM): we use the likelihood that a read belongs to a certain transcript to reconstruct the expression levels:

Cufflinks, for example, uses this “likelihood function”: it computes the probability distribution of an exon read to fall in one of the isoforms, and then uses the maximum **likelihood** to quantify the expression. To create this function, it uses the number of reads in common with more isoforms, and the distribution of fragment lengths, among other parameters extrapolated from the data, in order to assign reads to their respective isoforms.

The EM method, at its most bare, is very easy: We begin by assuming that each transcript’s expression is equal to the others. We then proportionally assign the reads based on this distribution. We then recalculate the original expression levels of the isoforms based on the quantity of reads and repeat these steps until they converge.

CuffDiff2 takes it a step further: it ignores normalization (?) and immediately finds Differentially expressed genes by comparing the reads generated from them.

### Normalization

* Fragmentation creates more reads from longer reads. The fragmentation procedure is random, and the fragment length is random.
* The number of reads created at each run can fluctuate due to technical and other errors.

We have three different normalization:

* **RPKM**, reads per Kilobase per Million reads
  + We divide the total number of reads by 1e6
  + We divide each read number by this number
  + We divide by the length of the gene
* **FPKM**, Fragments per Kilobase per Million reads
  + Used for paired ends, as each fragment can either map only once, or in two positions. We count the fragments and not the reads that map, thus we don’t double the read
* **TPM,** Transcript per million
  + Normalize by gene length
  + We compute the sum and divide by the scaling factor
  + We divide each number by this number.
  + Note that the steps are the same as RPKM but switched. In TPM, the total number of reads per sample is the same, so we can compare each sample with the others. This doesn't occur if we compute RPKM.

Another, more fancy type of normalization handles the variability introduced by highly expressed genes and is explained below.

# 5. Differential Expression analysis

Differential expression analysis (DEA) refers to the **identification of genes** (or other types of genomic features such as transcripts or exons) **that are expressed in significantly different quantities in distinct** **groups of samples** (differentially expressed genes, DEGs), be it biological conditions (drug-treated vs. controls), diseased vs. healthy individuals, different tissues, different stages of development, etc…

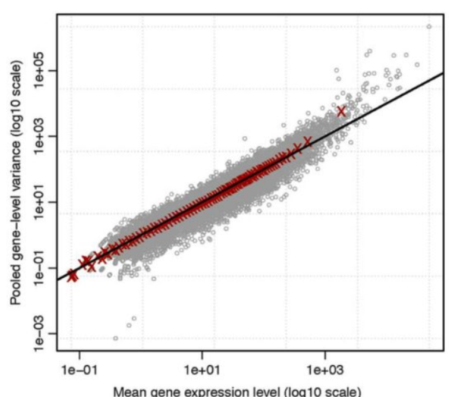
Genes are usually analysed in a **univariate way**, assuming that the expression of one gene is not related to the expression of the other genes. This is obviously not the case, but fitting a statistical model considering all the interactions between genes is often not possible. Usually, methods to reduce dimensionality are applied for data visualization and statistical modelling of the data beforehand.

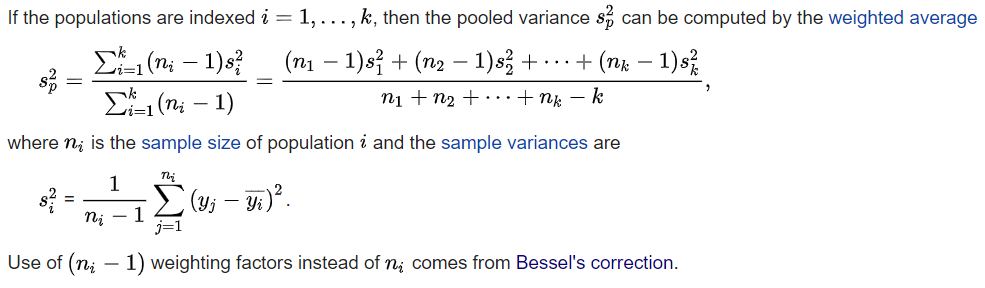
## Variance, replication, variability, normalization

Replication is essential in order to discover statistical significance. A difference between two groups, to be deemed significant, must be larger than the variance inside the groups themselves. Increasing the sample size and the replicates decreases variability, thus it is easier to detect differences between groups. A minimum of three biological replicates per biological condition must be used in every experiment.

Many sequencing core facilities require or suggest using at least three or four replicates per group to be compared; two is almost always too few. With three, there is the risk that at least one sample will fail in library preparation or sequencing and you still end up with only two replicates in one of the groups.

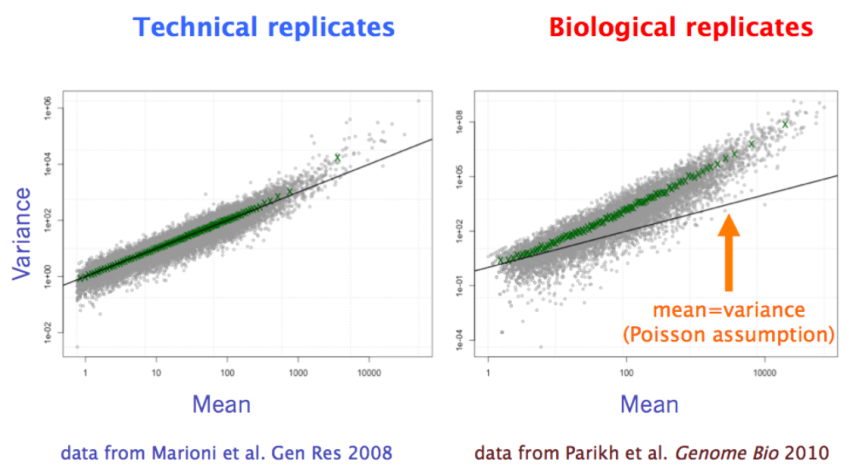
Human blood and some tissue samples used for clinical case–control transcriptomics studies seem to exhibit considerable variation between individuals. Particularly for complex diseases, very large numbers of replicates (perhaps hundreds or thousands) may be needed to observe differential expression between cases and controls. For cell lines or samples from distinct tissues, only a few replicates may be needed.

We can assume that the variance of the distribution of the variance of genes would follow a Poisson distribution, as it is based on randomly sampled, independent reads. Indeed, plotting the variance in a variance/mean plot we can observe that the cloud of dots closely follows the expected variation in a Poisson distribution, where

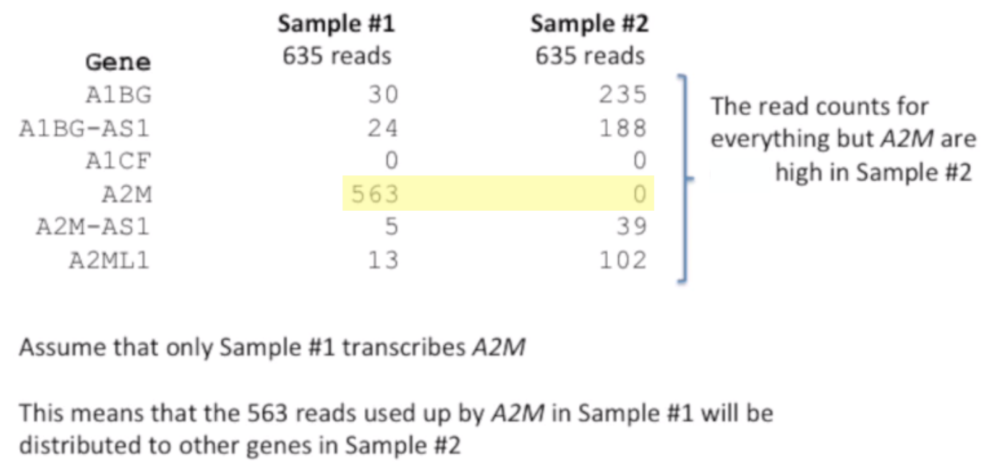
The plot to the side shows mean expression vs pooled gene variance (see box under). We can see that it closely (but not completely) follows the estimated Poisson variance, represented as the black line. The pooled variance is an estimate of the fixed common variance {\displaystyle \sigma ^{2}}underlying various populations that have different means.

The sequencing steps introduce noise, or additional variability. We can distinguish a few types of noise:

* **Shot noise**
  + The inherited variance arising from the sequencing. This is unavoidable, and accounts for the Poisson-distributed variance that is seen in technical replicates.
  + Although unavoidable, can be accounted for mathematically.
  + It is the main source of variance for the low-expressed genes.
* **Technical noise**
  + From sample preparation and sequencing. This is minimal if the methodology is executed well.
* **Biological noise**
  + Arising from the biological variability of the samples. It cannot be minimized technically but can be accounted for from the datapoints.
  + It is the strongest source of variance for highly-expressed genes.

When samples are taken from biologically distinct sources, such as different individuals, the variability between them has often been modelled by a negative binomial distribution (sometimes called gamma-Poisson distribution). This distribution can be described as an over dispersed Poisson distribution. It exhibits higher values for variance with higher means, according to the Biological noise concept, where highly expressed genes would show higher biological variability between samples.

We can see in the graphs to the right the different distributions of variance for technical and biological replicates.

This increased variability for strong expressed genes adds a problem in the normalization step when looking at different biological samples. Genes that are highly differentially expressed can distort the respective normalization if only total reads are considered for normalization (like RPKM, TPM, etc… do). So, we need to find a better way to normalize read counts and sequencing depth but removing this distortion.

We calculate the geometric variance of each gene. The geometric variance is calculated as such:

Let be the expression (read count) of the gene in sample and the number of samples. The geometric mean of gene across all samples is:

Consider that in R, is considered to be . Therefore, all genes with at least one samples with 0 read counts will have has geometrical mean. They are usually filtered out for subsequent steps. This helps focus the normalization to only use housekeeping genes similarly expressed in all samples.

To normalize expression between all libraries, the mean log value for a certain gene, , is subtracted to all log values of that gene. Then, the median log expression value is computed for each sample, and the scaling factor is calculated. Then, each original expression value in each sample is multiplied by the scaling factor for that sample. The new read counts are normalized.

In summary:

* Calculate the geometrical mean for each gene considering all the samples. Discard genes that have in at least one sample.
* Calculate the median value of the ln gene expression in each sample considering all not-discarded genes from before.
* Calculate the scaling factor .
* Multiply all original expression levels by the scaling factor.

This normalization is used by tools for differential expression analysis such as DeseQ2.

## Using linear models for DEA

As an example, we can utilize linear models in order to statistically infer difference between gene expression in the different samples. First, we make some assumptions:

* The count value for a gene in sample is generated by a negative binomial distribution with mean and dispersion .

If we use a linear function to describe our data, our hypothesis become:

To test our hypothesis, we first have to fit our linear models with the data.

### Method of least squares

The model is trained to reduce at a minimum the squared differences between predicted value for a certain data point and the train value. This method is called the method of least squares.

Once a line is fitted by the data, it will have (assuming only two variables) the following form:

To evaluate how good this fit is on the training data, we can calculate . Let be the predicted value of for point , and the number of training data points:

In other words, the least squared variance (defined as the mean sum of squared differences) of the model, in contrast with the same variance but computed simply using the mean of the test variable (usually in the axis), “explains” of the variance. For example, if , the model accounting for the test variable explains 60% more variance than the mean value alone.

How do we translate an score to an interpretable p-value? We can move through the F statistic. We define the F statistic as:

Where is the number of parameters for the linear model, and is the number of parameters for the mean line. To move from an F statistic to a p-value we need a distribution for the values of the F statistics. To do this for a single model, we generate random data points and we fit the same model shape to them. We then calculate the F statistics of these models and generate a frequency table. This frequency table can be fitted using an F distribution, and the p-value of this distribution (using the F statistic of our model) can be easily computed as the area under the curve from F ≥ F statistic.

Linear models can be generalized in several ways. We have already seen the increase in variable number, but we can also include polynomials and other functions (although their effect on the model must be linear, so for example a logarithm of an exponential parameter can be included, but not the parameter itself). Generalized linear model give us a strong statistical tool to calculate statistical differences.

The Bioconductor package DeseQ2 uses the same principle but applied to the negative binomial distribution (that best fits biological replicates of NGS data). The p-values of this distribution can be computed using the Wald test or the Likelihood ratio test (LTR). The LTR test can be run on two models, L1 and L2, where L2 has one less predicting variable than L1. The LTR test assumes as H0 that the two models are actually the same, and computes

Where L1 and L2 are the *likelihood* that the model is actually correct, based on the data points. The LTR score can then be applied to the likelihood distribution to obtain a p-value. Small p-values indicate that the additional variable included in the model actually increased the overall likelihood of the model more than it would be expected if the variable had a true coefficient 0 (so, it not being there).

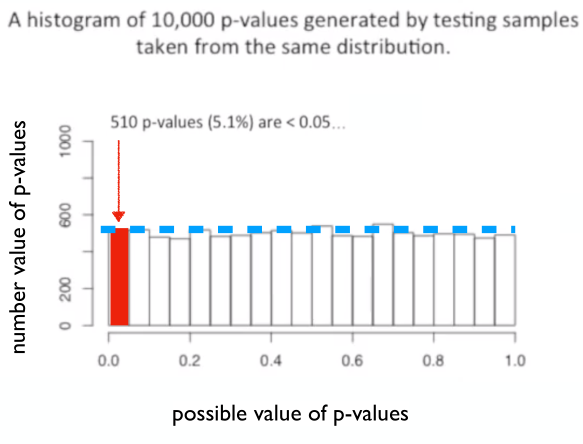
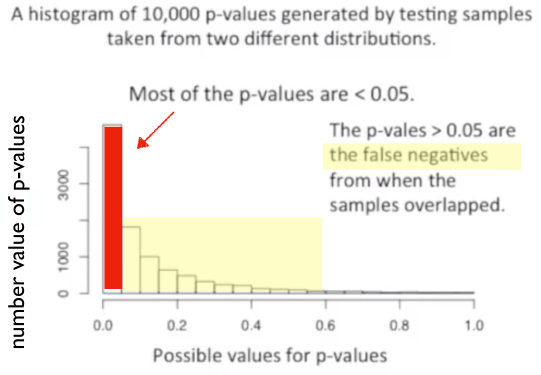
## P-value adjustment – Multiple Comparisons

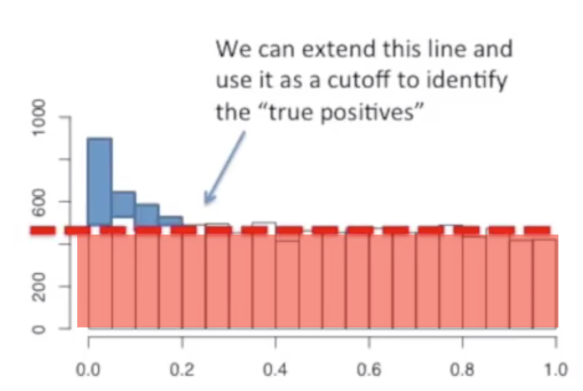
Since our gene counts is extremely large, usually in the order of thousands, even accepting a low p-value, which remember is the representation of the error (false positives), we would get a very large number of false positives. So, we must calculate adjusted p-values in order to control (limit) the number of false positives (genes that seem to be statistically significant but are not).

For example, imagine we have several samples all deriving from the same normal distribution. In most cases, if we calculate the p-value of the two samples coming from the same distribution (such as using a t-test or the generalized linear model), we would get a very large test score, > 0.05, it being correctly identifying the shared origin of the data points. Rarely, though, we would get a p-value < 0.05, due to the nature of randomness. If we are testing a few samples, this wouldn’t be a problem. However, as we said before, testing a large number of samples will give us many false positives, more specifically, 5% of all “detected positives” will be actually false positive (as we accept an error of 0.05).

If we consider 10000 genes, 10000 \* 0.05 = 500; so, 500 genes will be false positives. How can we limit this number? False Discovery Rate, or more specifically the Benjamini-Hochberg procedure can limit the number of false positives.

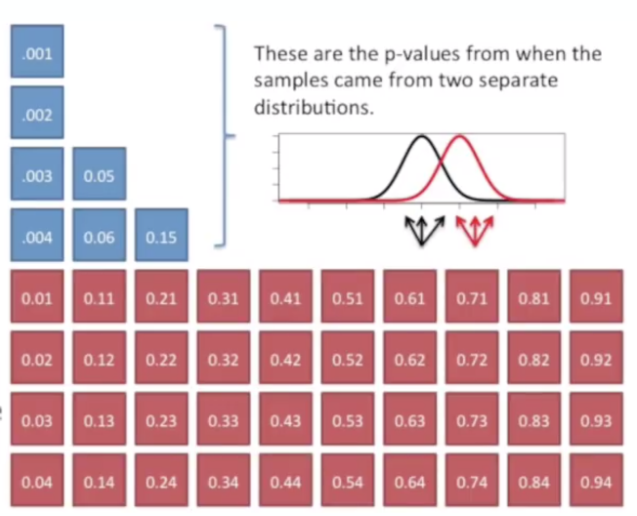
Imagine we generate many samples from the same distribution and calculate the p-value. By the nature of randomness, we expect that 5% () values will have p-values < 0.05. See the histogram to the left.

On the other hand, generating several samples from two different distributions and calculating the p-value between them will most often give small (<0.05) p-values, and some false negatives, where p-value > 0.05 (β error). See the histogram to the right. Consider that the β error can be reduced by increasing sample size.

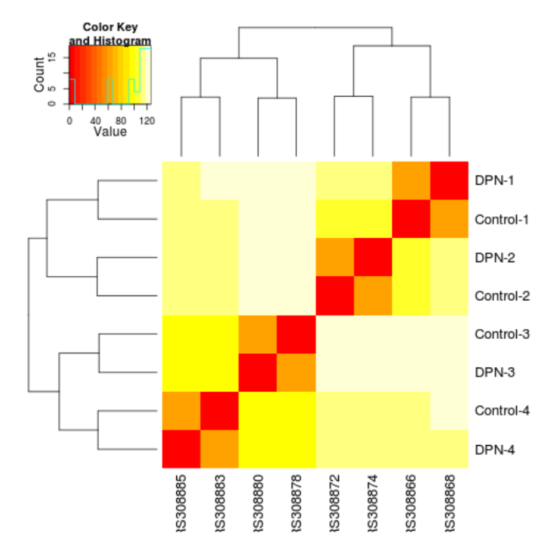
In a true experiment, we would imagine that most samples (genes) come from the same distribution while only a minority come from two different distributions. So, we can image that the bulk of our p-values will follow the distribution to the left, while a minority the one on the right. So, we can create a histogram like the one below, where most p-values arise from the same distribution and floating on top the p-values coming from two different distributions. See the histogram to the right. The red p-values come from the same distribution while the blue p-values from different distributions.

If we imagine to “cut” the last histogram in red and blue portions, we can see that we are not that far off the real histograms from before. The B-H procedure essentially translates this idea into mathematical practice: it increases p-values in such a way that only the very small p-values (the blue ones deriving from actually different distributions) will fall in the <0.05 range, while all the others (red) will shift into > 0.05 bins, so they stop being significant.

The actual B-H procedure is very simple:

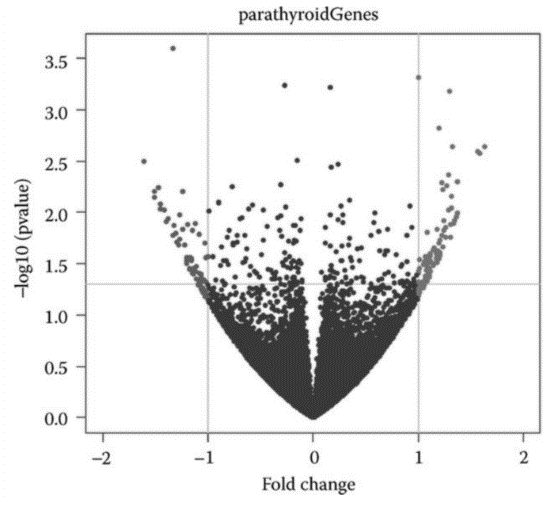
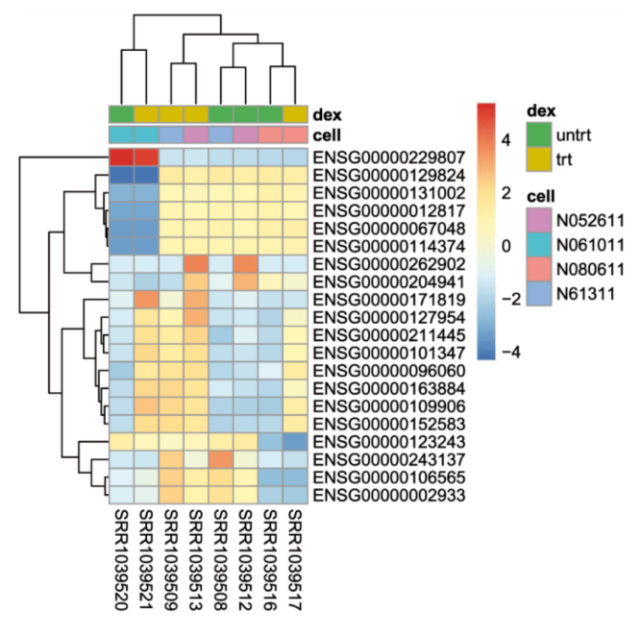
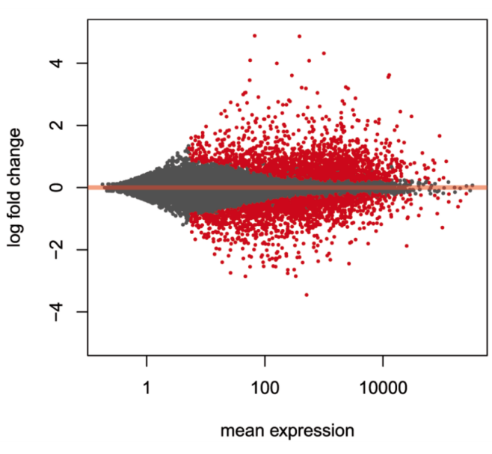
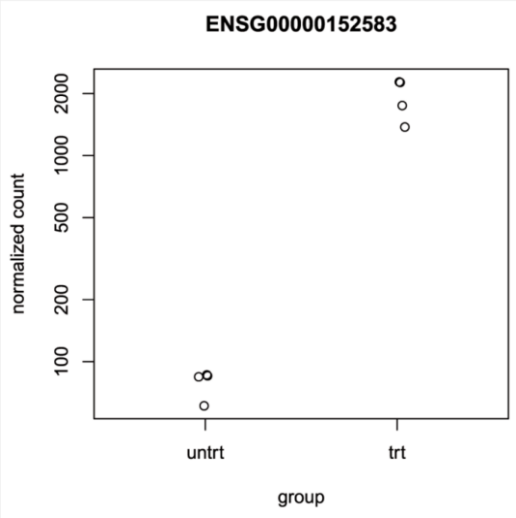
* Order all p-values from smallest to largest;
* Rank them using integers, from 1 to n, 1 being the smallest and n being the largest.
* The n-th p-value’s adjusted p-value is equal to itself;
* The n-1 and all subsequent p-value’s adjusted selves is the minimum value between two options:
  + The previous adjusted p-value;
  + The current p-value \*

## Plotting differential expression

There are several kinds of plots that can aid us in showing differential expression. Here are the main kinds:

Heatmap plots

We can plot the expression of our genes (x axis) in the different samples (y axis). In the heatmap, the stronger the colour, the higher is the expression.



Normalized counts

Even a simple plot showing the different samples (x) and the normalized count values (y) for a specific gene can be informative

Heatmap 2.0

This more complex heatmap shows log\_2 FC of different gens in different samples, but also encodes with colours the different treatments and cell lines used in the experiment.

Volcano Plot

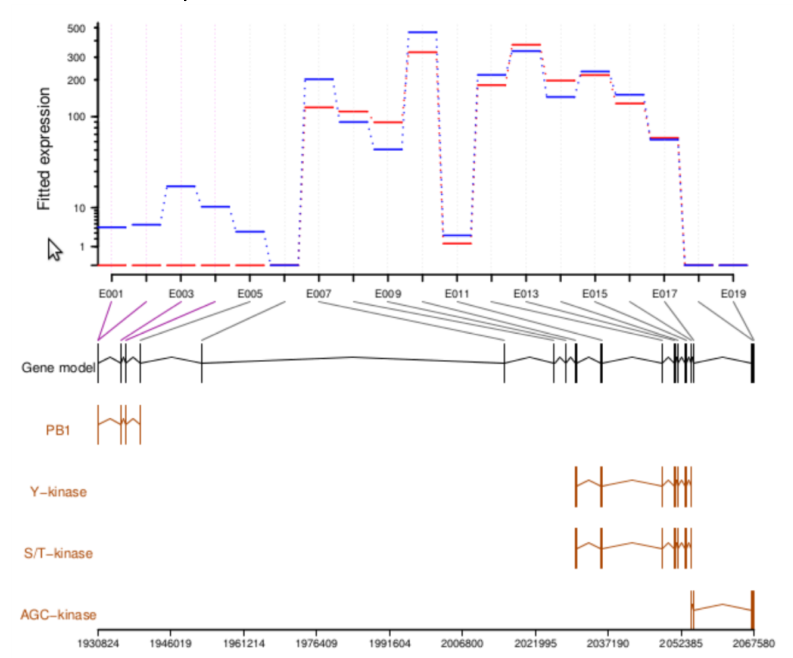
It shows on the x axis the log\_2 FC and on the Y axis the -log\_10 (p-value). The two lines show thresholds for both FC and p-value.

MA plots

M → “minus”, as the log FC is actually log [CT] – log [TR]

A → Average expression

Genes with an adjusted p value below a threshold (here 0.1, the default) are shown in red.



## Data Interpretation – GO terms

Complex DEXseq plot

This is an expression graph that shows the computed expression levels of each exon for each different isoform. It is created by DEXseq.

**Ontology**: ***The representation of concepts using specific vocabulary***. It can be represented by a hierarchical DAG, more specifically a tree, with a root, levels, and leaves. The **gene ontology** (GO) **consortium** was founded in 2000 after the human genome project. It's “open”, meaning anyone can insert new genes inside the terms, or completely new terms. Currently, the GO consortium maintains three trees: the **GO biological process**, the **GO molecular function** and the **GO cellular component**.

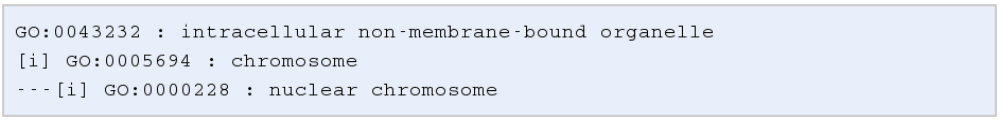
In a GO, the root contains all the genes, then we move down to increasingly specific functions, and thus less genes. When one wants to add a new node, we need to also choose the level in where to put the node.

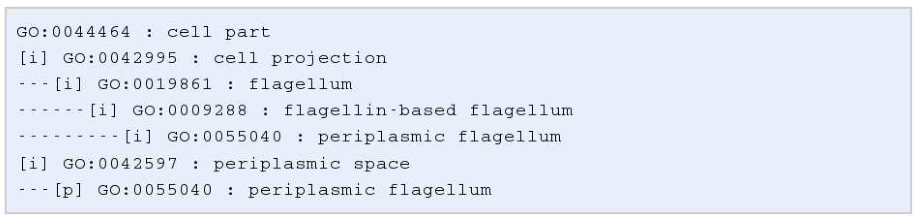
*Gene node* = *Gene term*, and the node can be a **parent** or a **child** (but, except for the root and the leaves, are both, depending on the point of view). Associated to a node is a specific term, the **GO term**, and on the arcs we have three types of relationships: “**is\_a”**, “**has\_a”** (aka part of), and **positive/negative regulation** (only in the “biological process GO”).

For example: “Mitotic chromosome” is\_a “chromosome”, so we have parent(“chromosome”) – is\_a -> child(“mitotic chromosome”)

Each term, each node, has a **7 digit ID number**, such as GO:0000000

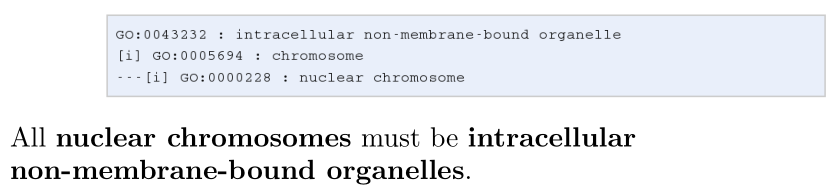
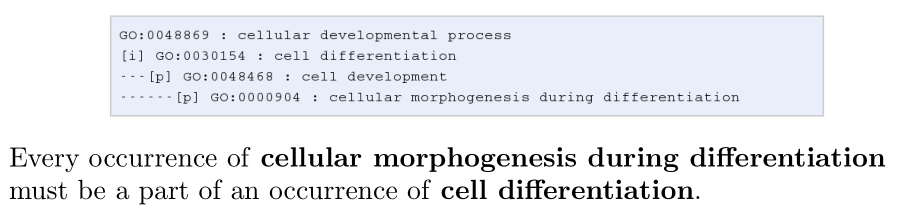
For example:

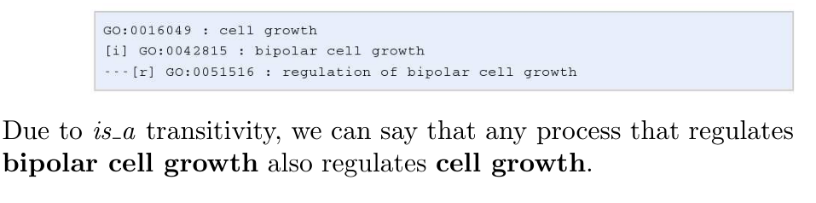
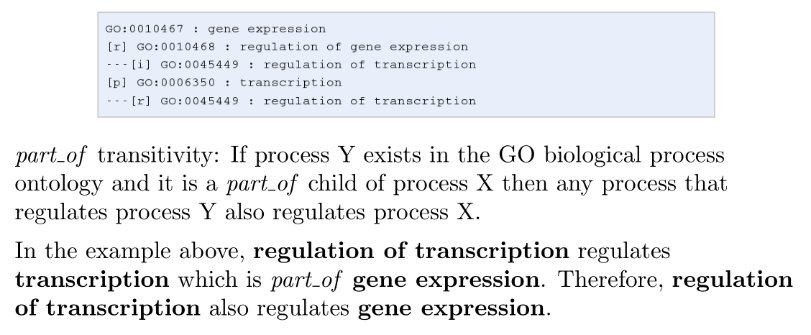
In this graph, the relationship between the levels is "is a", identified by **[i]**. Therefore, “nuclear chromosome” is (always) a chromosome.

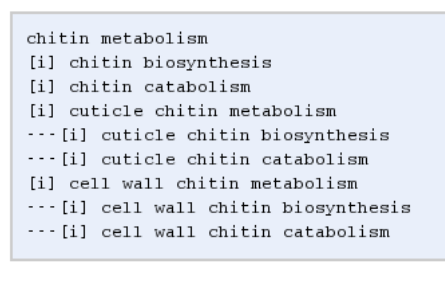
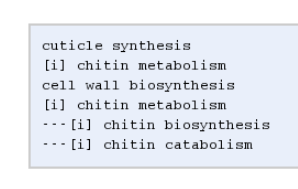
The "part of" keyword is identified by **[p]**:

Here, “periplasmic flagellum” is always a part of periplasmic space, but this doesn’t mean that every periplasmic space has a periplasmic flagellum.

In order to maintain the possibility of automated search of a GO tree, we have some rules to follow when adding new nodes and genes:

* **Transitivity** of is\_a and part\_of:
  + If Chromosome is\_a intracellular non-membrane bound organelle, so nuclear chromosome (a child of chromosome) is\_a intracellular non-membrane bound organelle.



* Every GO term must obey the **true path rule**: if the child term describes the gene product, then all its parent terms must also apply to that gene product.

The GO is organism-independent, meaning we have information for each gene in each organism.

We have info about how the information was acquired, coded inside the gene name in a GO node. For example, inferred from experiment is EXP, inferred from Direct Assay is IDA…

### GO enrichment analysis

If I have 100 differentially expressed genes, all falling into a specific leaf, we can conclude that there is a very high chance that the differential expression has caused a difference in that process. To calculate this formally, we have to use a **hypergeometric distribution** (HGD) and test our hypothesis with **Fisher's exact test**.

The HGD is a discrete probability distribution that describes the probability of a success in n draws, without replacement, from a finite population of size N, that contains exactly K objects with the feature of interest, where each draw is either a success or a failure.

We can think of this as a bag of balls with N balls. We draw 7 balls from the bag, 6 blue and 1 red, and we know that there are 5 red, 8 blue, 8 orange, 5 yellow, 8 green and 5 brown balls (we know the population). The order of balls is not important, so extracting one red and 6 blues, or 6 blues and 1 red, or any other permutation, has no difference. So, we need to consider all possible permutations of the sequence “B B B B B B B R”:

B1 = 8 / 40 (prob of extracting a blue ball is 8 blue over 40 total)

B2 = 7 / 39

...

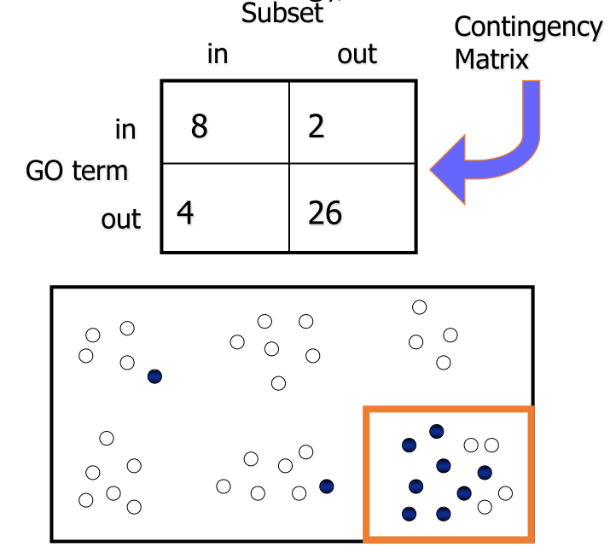
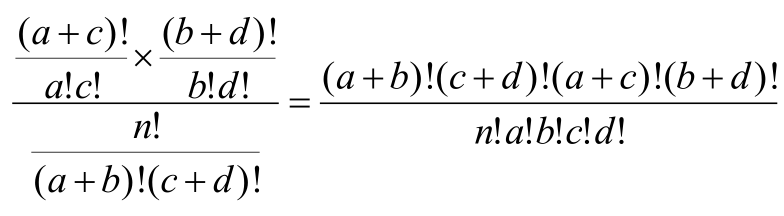
B7 = 2 / 34

R = 5 / 33

We multiply all of these values and get the probability of that specific permutation. We compute this probability for all permutations and combine them together to create a new probability value. Can we compute a p-value from this probability value?

The probability computed in this way is ~0.00000053.

Since we know the population, we can use Fisher's Exact Test to compute an **exact p-value**. This is an exact p-value as we know all the possible outcomes, so we can calculate the p-value based simply on its definition, the probability of getting the outcome or any outcome rarer than this one. Continuing our example, the p-value computed by Fisher’s exact test is 0.01. Fisher’s exact test is based on the Hypergeometric distribution.

Applying the same principle for a list of differentially expressed genes (knowing the population of all genes in all go terms), we can calculate Fisher’s exact p-value: if the p-value is significant, the list of gene is more “involved” than normal in that specific term than randomness.

Here we can see the concept of HGD applied to GO terms. We can see each Go term as a box (orange) containing terms (dots). The terms of interest (blue) are the ones DE in our analysis. We can then construct a contingency matrix containing this information and compare it to a “random” contingency matrix expected to arise from the same tree. The probability of any particular matrix occurring by random selection, given no association between the two variables, is given by the hypergeometric rule, shown above. There is a statistical test, called Hyper Geometric Test, that can use this value in order to produce a p-value. If the p-value is small, there is a very high probability that the score matrix seen in the experiment is not due to randomness, but the samples are actually different.

GO-rilla is a tool to search the GO terms starting from a list of genes. Giving it a list of ranked (such as with FC scores) or unranked genes it can calculate enrichment for certain GO terms and show the relevant branch of the trees involved. It can also take a “background” list of genes, meaning all genes that appear with at least 1 read in at least 1 sample in the analysis. It uses this list of background genes to reduce the “noise” of the experiment.

The GO is however marked by flaws due to a failure to address basic ontological principles.

* the existing annotation databases are **incomplete**;
* the quality of an association among GO terms and genes depends upon the source of the annotation, and some data are imprecise or incorrect;
* the GO is an ongoing project in which new GO terms are added continuously and this can lead to a re-classification of already-tagged gene products;
* genes involved in several biological process, all the biological process is weighted equally, and it is not possible single out the more relevant one.
  + How can you say that “Cell mitosis” is more general than “Nucleotide synthesis”?

In summary, here are the criteria for detecting a set of differentially expressed genes:

* Statistically significant differential expression
  + After having a gene expression table, detected using t-tests, multi-way ANOVA, etc.
  + Set a p-value cut-off, such as 0.01, but consider FDR, as it is more stringent
* Satisfactory FDR
  + Apply BH to the p-values, require FDR ≤ 0.20 or 0.10.
* Require minimum fold change
  + Require |FC| ≥ 1.5 or 2; where FC = expression sample A / expression sample B

## Gene set enrichment analysis (GSEA)

Inferring biological significance to a list of DEGs can be a difficult task. We might end up with a long list of genes with no unified biological meaning, or, on the opposite, only a handful of genes.

Gene Set Enrichment Analysis, GSEA, attempts to fix these problems in a few ways:

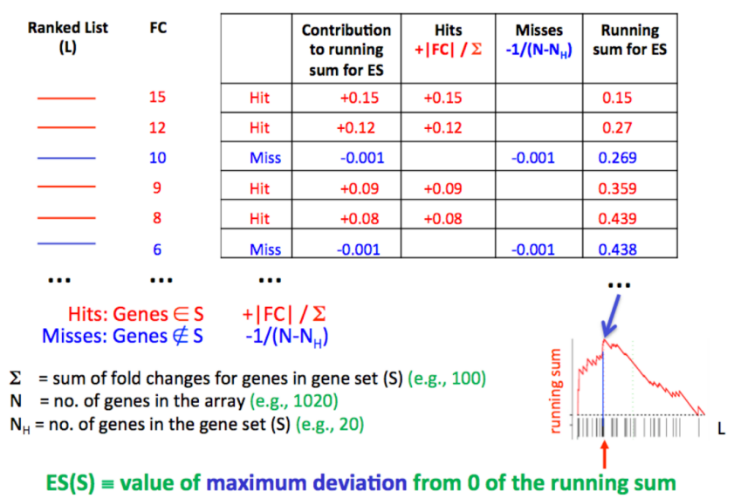
* We combine together information from several samples, such as all controls and treated, and find differences between groups of samples and not single samples alone.
  + This “fusion” increases statistical power, i.e. increasing the change of detecting a difference when, in fact, there is one.
* The expression of all genes in a dataset is used, and the gene set can derive from different studies.

From GSEA’s website: “**Gene Set Enrichment Analysis** (GSEA) is a computational method that determines whether an *a priori* defined set of genes shows statistically significant, concordant differences between two biological states”

GSEA, broadly, does the following steps when analysing sets of data:

* As input, it takes a gene count table, and a list, or more, of genes (S).
  + The S gene table is decided a priori. This means that its biological significance can be set before the analysis begins, and thus the problem of finding biological significance in a simple list of DEGs is surpassed.
* It sorts the gene count table based on some parameter, usually the FC between the mean expression of the control samples and the mean expression of the treatment samples.
* For each gene in the set S, find its position in the ranked list.
* Generate a running sum of gene S, see below. The maximum value of this sum is the Enrichment Score, ES.
* Use permutation testing to re-run the test and test for significance against a background. The statistical test used is Kolmogorov-Smirnov, with H0 that the two sets of numbers were drawn from the same continuous distribution.
  + Consider that the gene sets (samples) used with GSEA are a lot, so this scrambling can generate many different patterns.

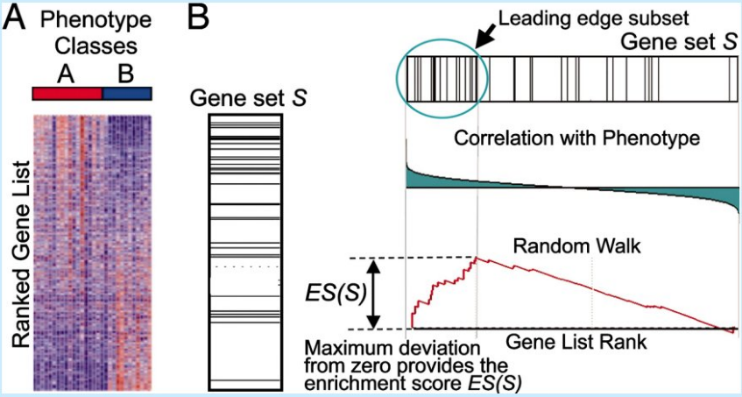
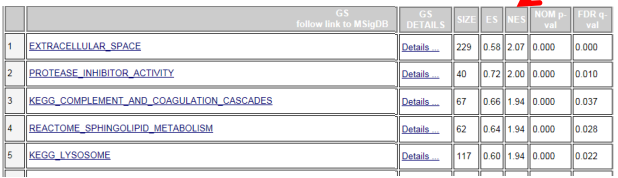
How do we calculate ES?

We begin with the ranked list of genes, and their associate FC. We mark genes as “hit” if they are in the gene list S or miss otherwise. We then calculate scores for hits and misses: a hit score is , while misses are a fixed value, such as , where N is the number of genes in the ranked list and Nh is the number of genes in S. We then compute the running sum for each row in the table, and the maximum value is the ES.

To check for significance, the labels of the provided gene set are swapped randomly, and the test is re-run. This happens several times, such as 1000, and an ES distribution is computed. Then, the new distribution is used to compute statistical significance in the form of a p-value.

However, ES might be biased due to sample size and multiple comparisons. To correct them, Normalized Enrichment Scores are calculated:

* To normalize for the size of the S list, GSEA divides the ES “test” value with the mean of the ES “bootstrapped” values. From the manual: “The normalized enrichment score (NES) is the primary statistic for examining gene set enrichment results. By normalizing the enrichment score, **GSEA accounts for differences in gene set size and in correlations between gene sets and the expression dataset**; therefore, the normalized enrichment scores (NES) can be used to compare analysis results across gene sets (experiments)”
* To account for multiple comparisons, we calculate an FDR score. From the manual: The FDR is a ratio of two distributions: (1) the actual enrichment score versus the enrichment scores for all gene sets against all permutations of the dataset and (2) the actual enrichment score versus the enrichment scores of all gene sets against the actual dataset.
  + From the publication: When an entire database of gene sets is evaluated, we adjust the estimated significance level to account for multiple hypothesis testing. We first normalize the *ES* for each gene set to account for the size of the set, yielding a normalized enrichment score (*NES*). We then control the proportion of false positives by calculating the false discovery rate (FDR) corresponding to each *NES*. The FDR is the estimated probability that a set with a given *NES* represents a false positive finding; it is computed by comparing the tails of the observed and null distributions for the *NES*.

Concluding, GSEA takes a new approach in DGE analysis, by not considering the single genes but groups of genes with a specific biological function. After we compute mathematical enrichments (such as NES) we can further think of what this difference means for our samples. Down here is an example of the result from a GSEA.

## Fusion gene detection

RNA-seq can detect newly formed *fusion genes*. Fusion genes are created by important chromosomal events, such as chromosomal breakage. They are implicated in the development of cancer. We can have both intergenic splicing, where two genes on the same chromosome are stitched together, or transgenic splicing, where two genes from different chromosomes get fused together. The fusion patterns change between individuals and between cancers, and some cancers have several fusion genes while other have none. It was also discovered that fusion genes have a specific function in cancer and that there are patters of fusion which are cancer specific: it seems that some genes preferentially fuse with specific genes.

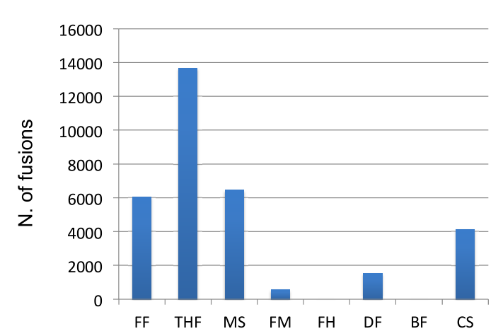
How is this different from transcriptome assembly? It is not significantly different, but we must consider that the junctions can happen outside of exon boundaries, and that two exons not necessarily fuse as a whole.

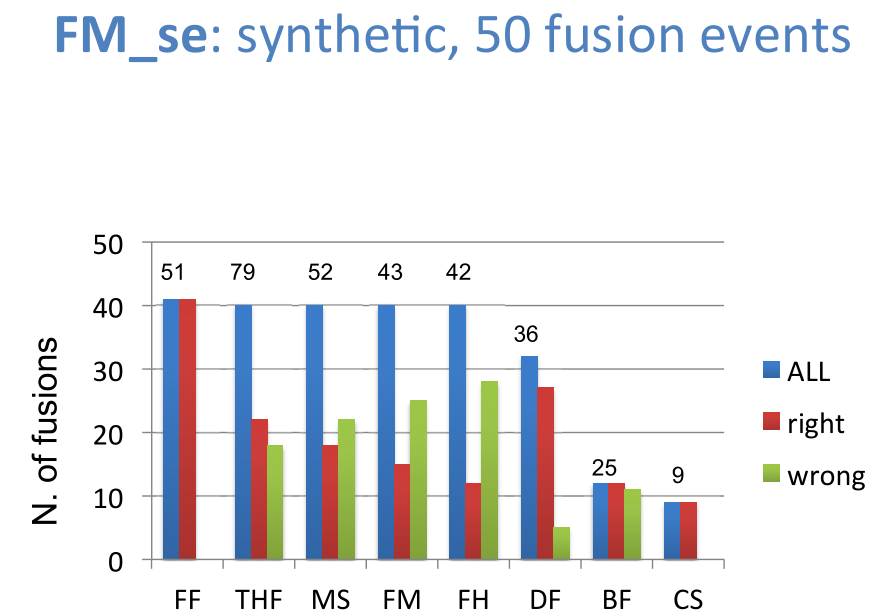
Aligners, and even spliced aligners, will discard all spanning reads that overlap a junction between genes, simply because the intron length is too large for a normal splicing event. However, paired-end reads that fall into exons of different genes are not discarded, and are said to be *encompassing* the fusion junction. Three types of algorithms use these reads in order to detect fusion genes:

* Whole Paired End
  + Algorithms such as DeFuse and FusionHunter
  + They align the paired-ends reads on the reference and use discordant alignments (where the intron length is too long) in order to detect putative fusion junctions. Then, these events are filtered based on several additional characteristics and the surviving ones are given as fusion events.
* Paired End + Fragmentation
  + For example, TopHat-Fusion, ChimeraScan and Ballerophontes
  + We follow the same steps as above. The encompassing reads are selected as putative junctions, but then a new pseudoreference is generated by actually fusing the genes involved. Afterwards, reads unaligned (discarded) in the first step (i.e. spanning reads) are fragmented and aligned to the new reference. Only putative junctions that are supported by several junction-spanning reads are sent to filtering.
* Direct Fragmentation
  + For example, MapSplice, FusionMap and FusionFinder
  + We first fragment the input reads and align the fragments. Junctions are discovered when two fragments are aligned with a large intron in between.
  + The putative junctions are then filtered.

How do we perform filtering? We can apply several types of filters on our putative junctions:

* **Paired-End Information Filters** verify the correct distance between the tags of a pair to validate the alignment on a fusion. This distance depends on the protocols used for the library preparation, and the tools can either take this information as input or infer it from the first alignment step. In both cases, reads mapping on the putative fusions at an excessive distance are filtered out.
* **Anchor Length Filters** use the concept of “Anchor length” (i.e. the number of nucleotides overlapping each side of a fusion junction) to evaluate the quality of junction-spanning reads associated with a fusion junction. Junction-spanning reads having at least one of the two anchor lengths below a threshold are interpreted as possible artefacts caused by mismatches or sequence similarity and are removed.
* **Read Through Transcripts Filters** try to discover and remove RNA molecules formed by exons of adjacent genes, usually generated when the gene end is not recognized during the RNA elongation phase (read-through).
* **Junction Spanning Reads Filters** remove all the fusion products not supported by a number of spanning reads greater than a specified threshold.
* **PCR Artefact Filters** try to discover and remove all duplicated reads generated by the PCR amplification pro-cess, by the identification of clusters of reads of the same length with an identical alignment on the reference.
* **Homology Filters** remove all the putative fusions having a high number of reads on homologous or repetitive regions which can lead to multiple alignments.
* Other filters are available, taking in consideration read quality, encompassing reads, statistics, entropy, etc…

One of the main problems with these tools is the very high number of false discoveries (false positives). Indeed, many of these tools will detect fusion genes even in synthetic datasets which contain no actual fusion genes (right).

Another point to consider is the sensitivity of these tools. When applied to a synthetic dataset containing 50 fusion events, while most tools detected around 50 mutations, in many the percentage of correct guesses is around at, or slightly higher than 50%. However, running the same comparison over real datasets with experimentally-validated fusion genes yields different results.

## Detection of miRNA

To sequence miRNA, which are very short, we need a peculiar library preparation. After miRNA are extracted from the cells (using a peculiar protocol in order to precipitate longer RNAs), adaptor sequences are ligated to the ends. These adaptors are the same used for the other library preparations, but, since the miRNA insert is very short, most of the template molecule will be made up of adapters. Therefore, during sequencing, the reads will contain not only the sequence of the miRNA, but very often parts of the read will be composed of the adaptor sequence.

Several tools are available to trim away the adaptor sequences from the reads, such as Adapter\_trim and Cutadapt. Cutadapt is written in Python with the alignment algorithm written in C; it trims the adapter sequences and all parts after that. It also works for the start of the reads.

To analyse miRNA, the discontinued tool SHRiMP can be used to align the reads to the genome using the Smith/waterman algorythm. Afterwards, the countOverlaps() (or findOverlaps()) method can be used to find where miRNAs overlap with the target sequence. Enriched portions are then normalized and can be further analised with tools such as DEseq2 or RankProd.

Another option is to use the comprehensive tool miRDeep2, which handles all aspects of the analysis steps, from alignment, to annotation, to enrichment detection.

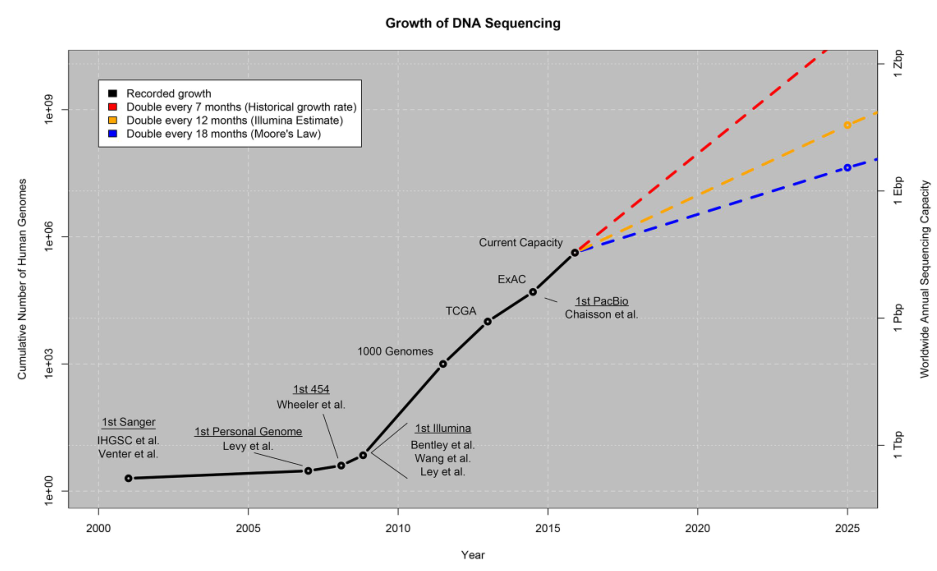
Other than miRNAs themselves, other short RNA molecules can be analysed, such as circular RNAs. Note that, for circRNAs, since their formation is Exon-start Exon-end, they are treated much like fusion genes, in the sense that large “intron” distances are used to detect them. We can also discriminate the circRNAs with their linear counterparts.

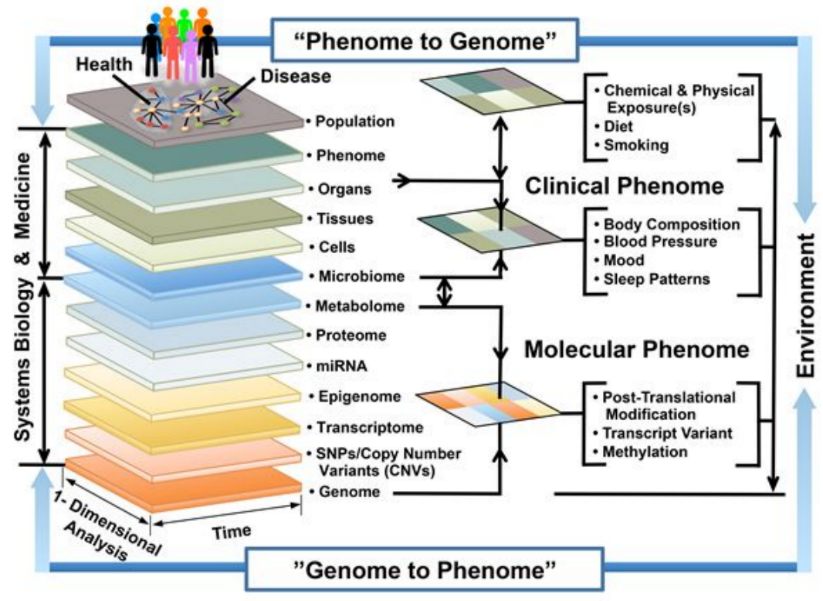
## Other Applications for Deep Sequencing

NGS can be used for many more applications than what we discussed here. For example, to detect epigenome modifications that distinguish various cell types in an organism and between organisms. Consider that microarrays have been extensively used for methylation patterns. Several ongoing projects, such as ENCODE and modENCODE are attempting to map the 100+ histone modifications together with DNA methylation, nucleosome positioning and DNA accessibility patters genome-wide in multiple cell types, individuals, species and conditions.

ChIP and ChIP-seq are extensively used to determine protein-DNA interactions, such as the detection of Transcription factors, and the complex landscape of gene regulation as a whole.

## Using public resource databases

The amount of genomic data available today is massive. It is thus increasingly important to know how to best use this increased knowledge in our own research. Where can we find this data? How can we use it? How can we use it *easily*? And most importantly, how can we integrate this knowledge with other biological information to extract meaning from the data?

The genome and the phenome are tightly linked, and all layers of this interaction can be explored through NGS. Additionally, a lot of experimental data is available online for free for further mining.

### Types of data

* Raw Data: FASTq files, microarray signals (CEL files), proteomic spectra (mzML files), …
* Processed: Expression tables, ChIP-seq peaks (BED), genomic variations (VCF), etc…
* Integrated: High-level platforms such as databases or programs with an interface.

Here are some of the most prominent NGS projects, and where to find their data:

* 1000 genomes project
  + Genome study
  + Aim: Create a catalogue of human genetic variations
  + Samples: Blood, primary lymphoblastoid cell lines
  + Technologies: WES, WGS
  + Available Data: Integrated, Raw
  + Data at: <http://www.internationalgenome.org/data>
    - VCF
    - SAM (alignments)
  + Data visualization available @ ENSEMBL genome browser, under “Population genetics”
* Roadmap Epigenomics Project
  + Epigenome study
  + Aim: Characterization of human healthy tissue epigenomes
  + Samples: Tissues and cell types
  + Technologies: DNA methylation, Histone PTMs ChIP-seq, DNAse-seq, RNA-seq
  + Available Data: Raw, Processed, Integrated
  + Data at: <http://egg2.wustl.edu/roadmap/web_portal/>
    - Raw, single dataset data:
      * Cell line, dataset Metadata
      * Quality control results
      * Read Alignments
      * Signal peaks
      * Genome-wide signal coverage
      * Expression quantification
    - Integrated data from multiple experiments:
      * Chromatin states
      * Comparison between epigenomes
      * Predicted promoters and enhancers
      * Tissue-specific regulators
      * Prediction of disease variant functional effects
* GTEX – Genotype-Tissue Expression project
  + Level: Transcriptome
  + Aim: Large scale analysis of expression Quantitative Traits Loci (eQTL, the normalized quantity of SNPs in a certain locus)
  + Samples: Post-mortem human tissues
  + Technologies: WES, WGS, SNP microarray, RNA-Seq
  + Available Data: Processed and integrated
  + Data at: <https://www.gtexportal.org/home/>
* The Human Protein Atlas
  + Level: Proteome
  + Aim: Analysis of protein tissue expression and localization
  + Samples: Tissues, cell lines
  + Technologies: Protein arrays, IHC (Immunohistochemistry), Immunofluorescence, Western blot, RNA-Seq
  + Available Data: Processed and integrated
  + Data at: <https://www.proteinatlas.org/>
* ENCODE
  + Level: Multi omics - cellular level
  + Aim: Functional annotation of the human genome
  + Samples: Cell lines, primary cells, tissue (non-human), whole organism
  + Technologies: ChIP-Seq, RNA-Seq, CLIP/RIP-Seq, WGBS, RRBS, DNase-seq, ChIA-PET, 5C, FAIRE-Seq, Repli-Seq, WGS, Hi-C
  + Available Data: Raw, processed, integrated
  + Data at: <https://www.encodeproject.org/>, <https://www.encodeproject.org/matrix/?type=Experiment>
  + ENCODE has several data visualization services:
    - WashU gene browser for ChIP: <https://epigenomegateway.wustl.edu/>
    - <https://genome.ucsc.edu/encode/>
* TCGA (The Cancer Genome Atlas)
  + Level: Multi omics - tissue level
  + Aim: Molecular characterization and classification of tumors and their vulnerability.
  + Samples: Healthy and primary cancer tissue samples
  + Technologies: WES, WGS, RNA-Seq, protein array, DNA Methylation array
  + Available Data: Processed and integrated
  + Data at: <https://portal.gdc.cancer.gov/>
    - Gene / miRNA expression quantification
    - VCF and MAF file
    - CNV (Copy Number Variation) data
    - Methylation data
    - Protein level data
    - Clinical data of samples
  + Data visualization available at <http://www.cbioportal.org/>
* Allen Brain Atlas
  + Level: Multi omics - organ level (brain)
  + Aim: Integrated analysis of cell line drug responsiveness
  + Samples: Human and mouse brain biopsies
  + Technologies: Microarray, ISH, RNA-Seq, MRI
  + Available Data: Raw, processed and integrated
  + Data at: <http://www.brain-map.org/>
    - Mouse Brain
      * Mouse Brain: ISH data
      * Developing Mouse Brain: ISH data
      * Mouse Spinal Cord: ISH data
      * Mouse Brain Connectivity: Reconstruction using transgenic mice data
      * Cell Type: RNA-Seq data, Morphology and electrophysiology data
      * Brain observatory: Physiological data on transgenic mice
    - Human / Primate
      * Human Brain: Microarray and RNA-Seq data, MRI scan
      * Developing Human Brain: Microarray, RNA-Seq, Exon microarray data
      * Glioblastoma Atlas: RNA-Seq, Clinical data
      * Aging, Dementia, TBI: RNA-Seq, Clinical data, IHC
      * Non-human primate data: Microarray data
* Other than these large curated projects, many more raw data can be found in less refined databases such as the Gene Expression Omnibus (GEO), found at <https://www.ncbi.nlm.nih.gov/geo/>
* A plethora of data can also be found as supplementary material in PubMed publications.