Quantification

Normalization

Mapping이 잘 되었는지 확인

```
$ cd ~/day1/star/example1
$ ls
example1Chimeric.out.sam
example1Chimeric.out.junction
example1Aligned.out.bam
example1SJ.out.tab
example1Log.progress.out
example1Log.out
example1Log.final.out
```

Samtools 준비

```
$ ln –s ~/../kjyi/bin/samtools ~/bin
$ samtools -h view example1Aligned.out.bam ¦ less
```

samtools 사용법 확인

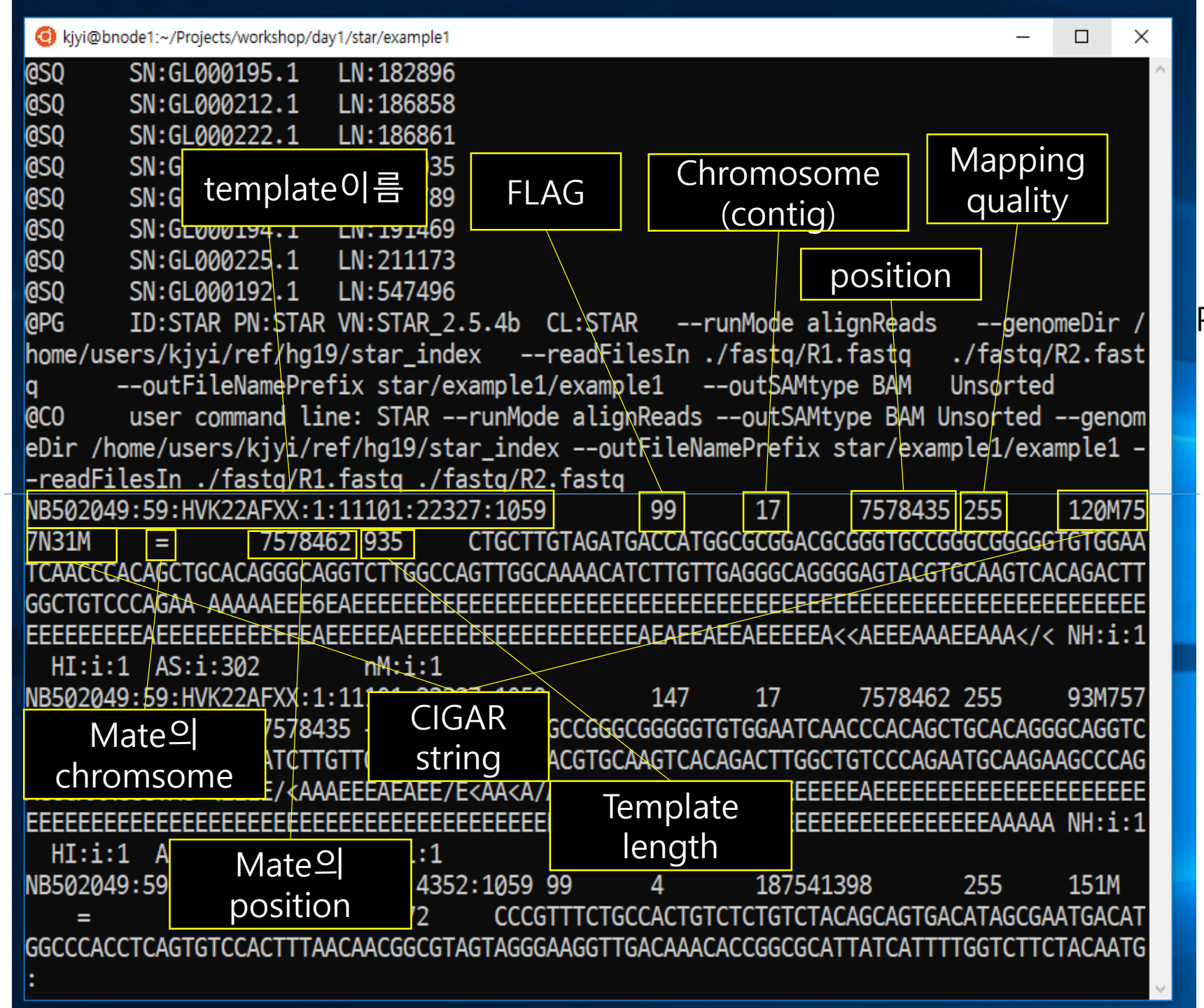
```
$ samtools ¦ less
$ samtools view ¦ less
```

HD = header, VN = format version

SQ = sequence dictionary, SN = reference sequence name, LN = Reference sequence length

kjyi@bnode1:~/Projects/workshop/day1/star/example1

```
@HD     VN:1.4
@SQ     SN:1    LN:249250621
@SQ     SN:2    LN:243199373
@SQ     SN:3    LN:198022430
@SQ     SN:4    LN:191154276
@SQ     SN:5    LN:180915260
@SQ     SN:6    LN:171115067
@SQ     SN:7    LN:159138663
@SQ     SN:8    LN:146364022
@SQ     SN:9    LN:141213431
@SQ     SN:10   LN:135534747
@SQ     SN:11   LN:135006516
@SQ     SN:12   LN:133851895
@SQ     SN:13   LN:115169878
@SQ     SN:14   LN:107349540
@SQ     SN:15   LN:102531392
@SQ     SN:16   LN:90354753
@SQ     SN:17   LN:81195210
@SQ     SN:18   LN:78077248
@SQ     SN:19   LN:59128983
@SQ     SN:20   LN:63025520
@SQ     SN:21   LN:48129895
@SQ     SN:22   LN:51304566
@SQ     SN:X    LN:155270560
@SQ     SN:Y    LN:59373566
:
```



```
@SQ     SN:GL000195.1   LN:182896
@SQ     SN:GL000212.1   LN:186858
@SQ     SN:GL000222.1   LN:186861
@SQ     SN:G                    35
@SQ     SN:G                    89
@SQ     SN:GL000194.1   LN:191469
@SQ     SN:GL000225.1   LN:211173
@SQ     SN:GL000192.1   LN:547496
@PG     ID:STAR PN:STAR VN:STAR_2.5.4b  CL:STAR   --runMode alignReads   --genomeDir /
home/users/kjyi/ref/hg19/star_index   --readFilesIn ./fastq/R1.fastq   ./fastq/R2.fast
q      --outFileNamePrefix star/example1/example1   --outSAMtype BAM   Unsorted
@CO     user command line: STAR --runMode alignReads --outSAMtype BAM Unsorted --genom
eDir /home/users/kjyi/ref/hg19/star_index --outFileNamePrefix star/example1/example1 -
-readFilesIn ./fastq/R1.fastq ./fastq/R2.fastq
NB502049:59:HVK22AFXX:1:11101:22327:1059        99      17      7578435 255     120M75
7N31M   =       7578462 935     CTGCTTGTAGATGACCATGGCGCGGACGCGGGTGCCGGGCGGGGGTGTGGAA
TCAACCCACAGCTGCACAGGGCAGGTCTTGGCCAGTTGGCAAAACATCTTGTTGAGGGCAGGGGAGTACGTGCAAGTCACAGACTT
GGCTGTCCCAGAA AAAAAEEE6EAEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
EEEEEEEEEAEEEEEEEEEEEEAEEEEEAEEEEEEEEEEEEEEEEEEEEEAEAEEAEEAEEEEEA<<AEEEAAAEEAAA</< NH:i:1
  HI:i:1  AS:i:302      nM:i:1
NB502049:59:HVK22AFXX:1:11                      147     17      7578462 255     93M757
                    7578435 -                 GCCGGGCGGGGGTGTGGAATCAACCCACAGCTGCACAGGGCAGGTC
                    ATCTTGTT                  ACGTGCAAGTCACAGACTTGGCTGTCCCAGAATGCAAGAAGCCCAG
                    E/<AAAEEEAEAEE/E<AA<A/                  EEEEEAEEEEEEEEEEEEEEEEEEEEEEE
EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE                  EEEEEEEEEEEEEEEEEAAAAA NH:i:1
  HI:i:1  A                    :1
NB502049:59                    4352:1059 99      4       187541398       255     151M
    =                          72      CCCGTTTCTGCCACTGTCTCTGTCTACAGCAGTGACATAGCGAATGACAT
GGCCCACCTCAGTGTCCACTTTAACAACGGCGTAGTAGGGAAGGTTGACAAACACCGGCGCATTATCATTTTGGTCTTCTACAATG
:
```

PG = 프로그램 (bam 파일을 생성한)

CO = 코멘트 (STAR가 생성함)

여기부터 alignment section

https://samtools.github.io/hts-specs/SAMv1.pdf

FLAG: Combination of bitwise FLAGs.[7] Each bit is explained in the following table:

Bit		Description
1	0x1	template having multiple segments in sequencing
2	0x2	each segment properly aligned according to the aligner
4	0x4	segment unmapped
8	0x8	next segment in the template unmapped
16	0x10	SEQ being reverse complemented
32	0x20	SEQ of the next segment in the template being reverse complemented
64	0x40	the first segment in the template
128	0x80	the last segment in the template
256	0x100	secondary alignment
512	0x200	not passing filters, such as platform/vendor quality controls
1024	0x400	PCR or optical duplicate
2048	0x800	supplementary alignment

Paired-end sequencin이면서 (0x1 = 1) (0x2 = 2)
두 pair가 proper mappin이 되고 (방향과 거리가 STAR에서 설정한 대로 적절함)
둘다 primary alignmen이면서, 이 read가 First in pair인 경우 FLAG는? 1+2+64=67
(0x40 = 4*16 = 64),

Mate의 FLAG는? 0x1 + 0x2 + 0x80 = 131

Sort, index

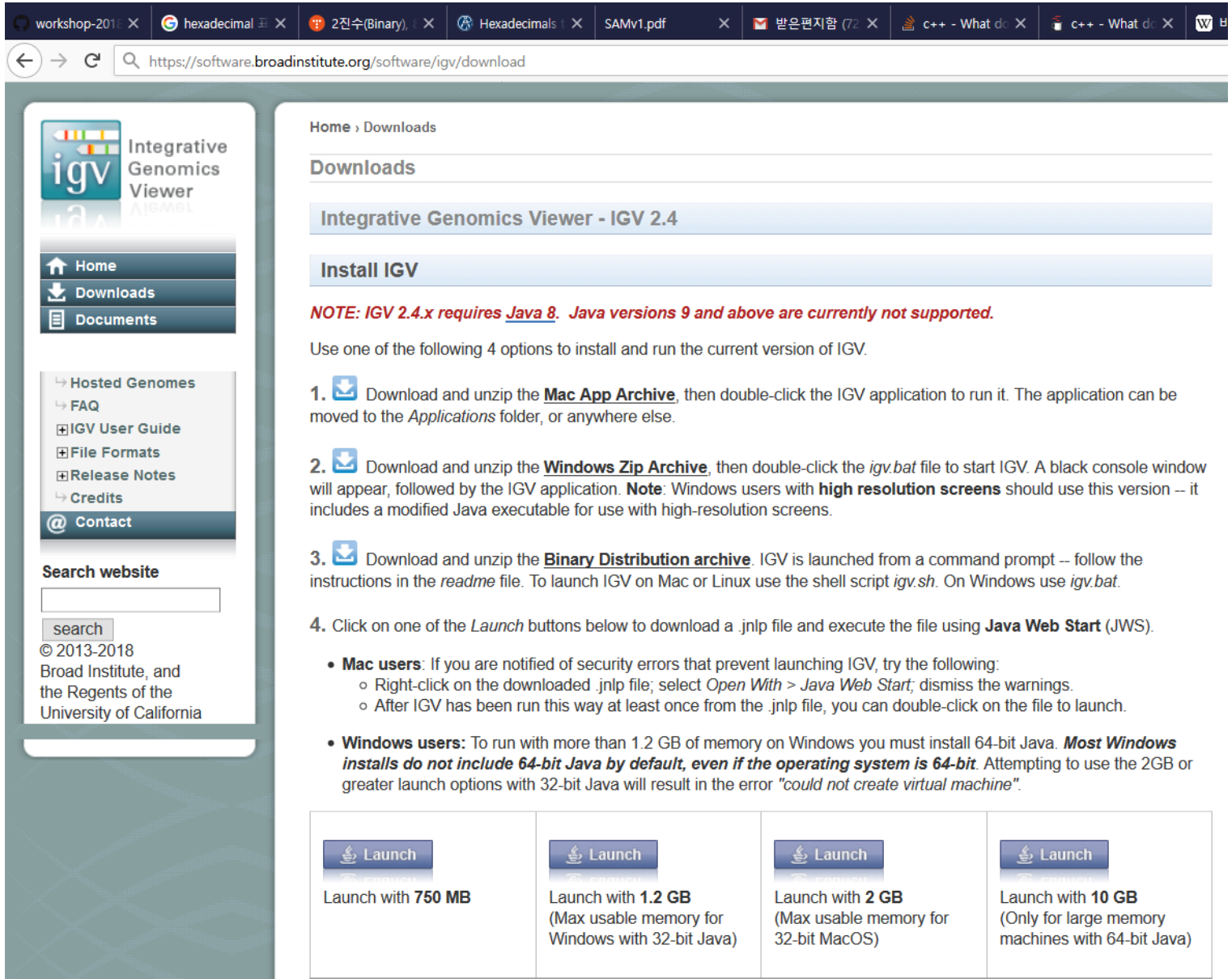
```
$ samtools sort example1Aligned.out.bam > example1Aligned.sort.bam

$ samtools index example1Aligned.sort.bam
```

(sort된 bam file 만 indexing할 수 있습니다.)

IGV 실습을 위해 example1Aligned.sort.bam file and example1Aligned.sort.bam.bai 파일을 다운로드하세요

https://software.broadinstitute.org/software/igv/download



Home › Downloads

Downloads

Integrative Genomics Viewer - IGV 2.4

Install IGV

NOTE: IGV 2.4.x requires Java 8. Java versions 9 and above are currently not supported.

Use one of the following 4 options to install and run the current version of IGV.

1. Download and unzip the **Mac App Archive**, then double-click the IGV application to run it. The application can be moved to the *Applications* folder, or anywhere else.

2. Download and unzip the **Windows Zip Archive**, then double-click the *igv.bat* file to start IGV. A black console window will appear, followed by the IGV application. **Note**: Windows users with **high resolution screens** should use this version -- it includes a modified Java executable for use with high-resolution screens.

3. Download and unzip the **Binary Distribution archive**. IGV is launched from a command prompt -- follow the instructions in the *readme* file. To launch IGV on Mac or Linux use the shell script *igv.sh*. On Windows use *igv.bat*.

4. Click on one of the *Launch* buttons below to download a .jnlp file and execute the file using **Java Web Start** (JWS).

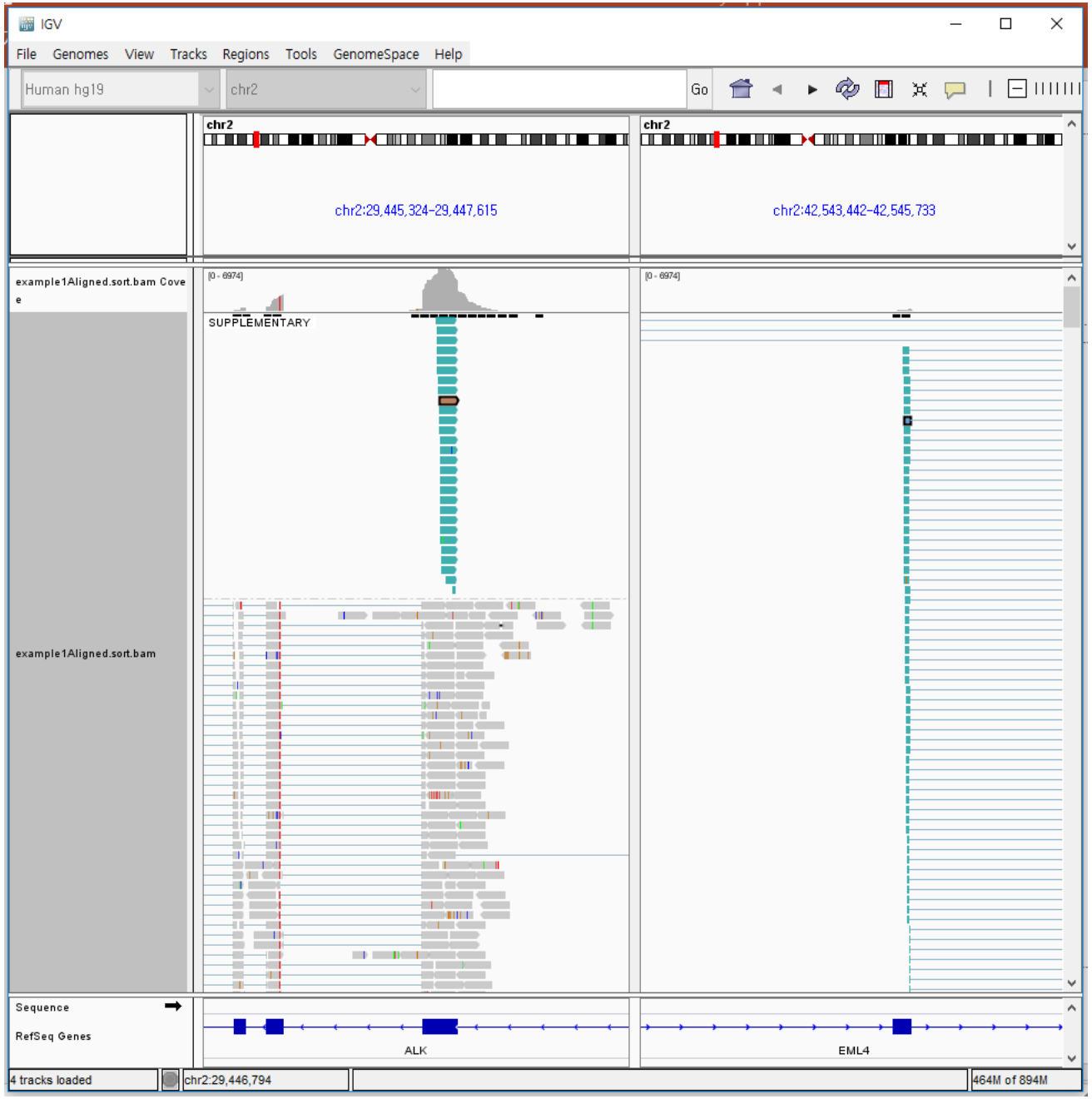
- **Mac users**: If you are notified of security errors that prevent launching IGV, try the following:
  - Right-click on the downloaded .jnlp file; select *Open With > Java Web Start;* dismiss the warnings.
  - After IGV has been run this way at least once from the .jnlp file, you can double-click on the file to launch.
- **Windows users:** To run with more than 1.2 GB of memory on Windows you must install 64-bit Java. ***Most Windows installs do not include 64-bit Java by default, even if the operating system is 64-bit***. Attempting to use the 2GB or greater launch options with 32-bit Java will result in the error *"could not create virtual machine"*.

Launch	Launch	Launch	Launch
Launch with **750 MB**	Launch with **1.2 GB** (Max usable memory for Windows with 32-bit Java)	Launch with **2 GB** (Max usable memory for 32-bit MacOS)	Launch with **10 GB** (Only for large memory machines with 64-bit Java)

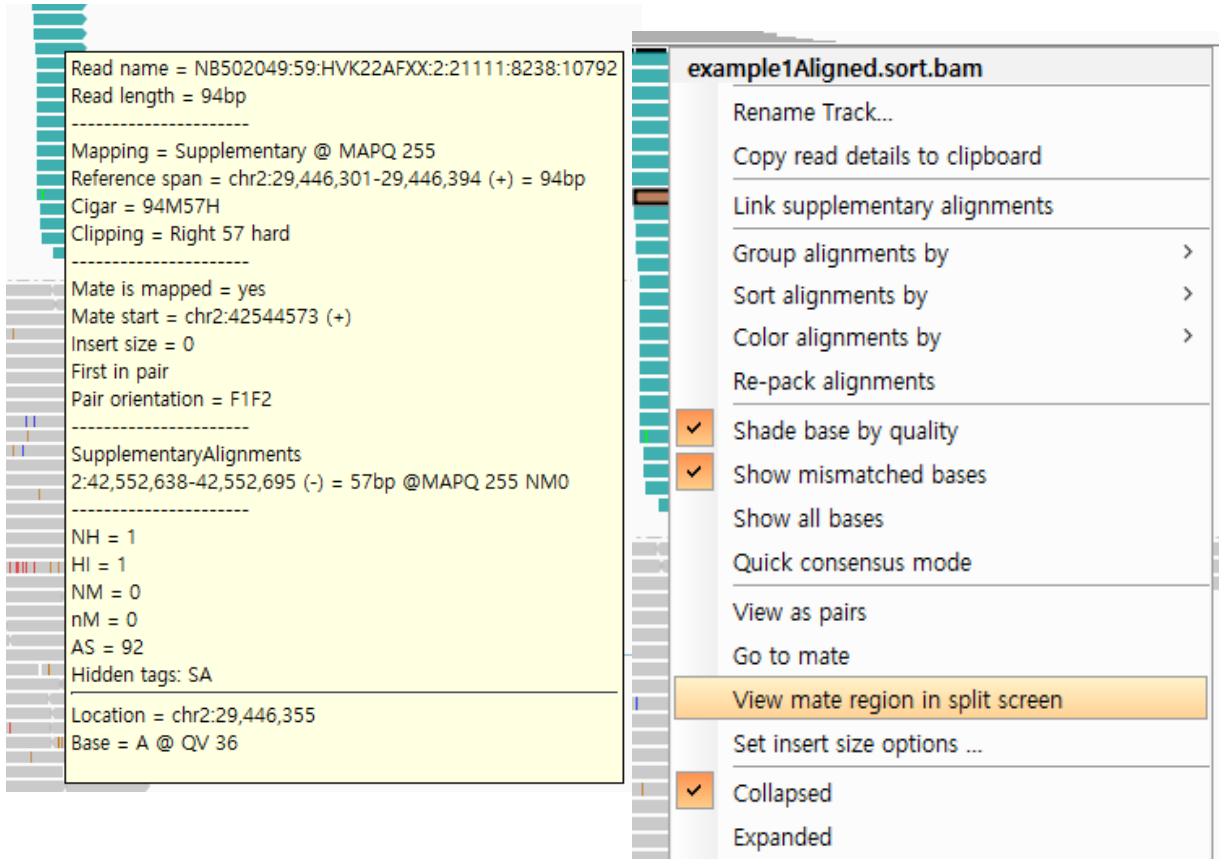
맥사용자

윈도우 사용자

리눅스, 기타 사용자



1. open bam file (sorted, indexed)
2. Go to the region
   chr2:29,445,324-29,447,615
3. right click alignment tract ->
   group by supplementary flag
4. Right click one of improperly mapped reads, Select View mate region in split screen



Quantification

Normalization

Advanced bash commands

(for, while, if, md5sum)

Introduction to R

Quantification of Gene Expression

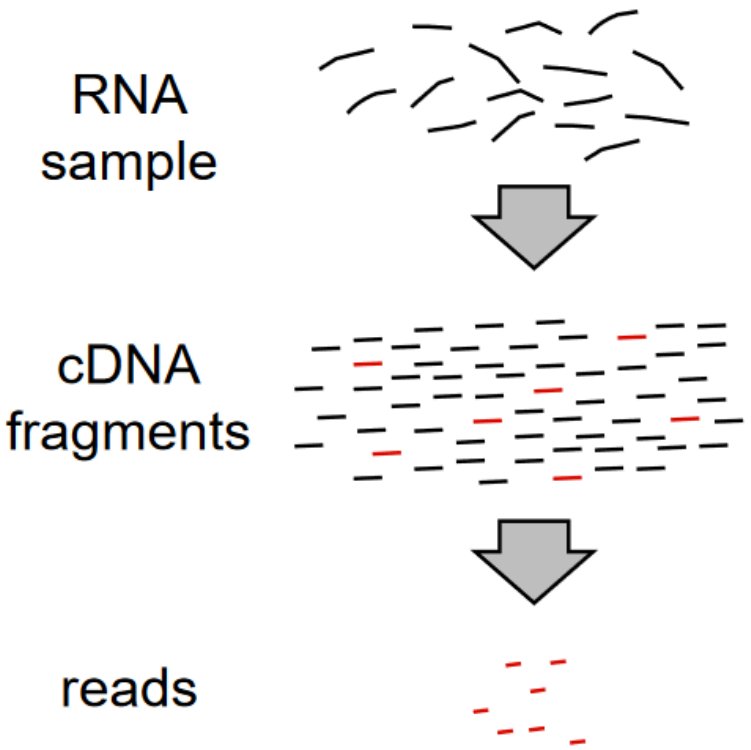
- Estimate relative abundance of transcripts
- Count reads, fetch depth/coverage

- Differential expression
  - 두 조건에서 얻은 gene expression profile을 비교하여, 어떤 transcript가 두 조건 사이에서 발현에 차이를 보이는지를 찾는 것

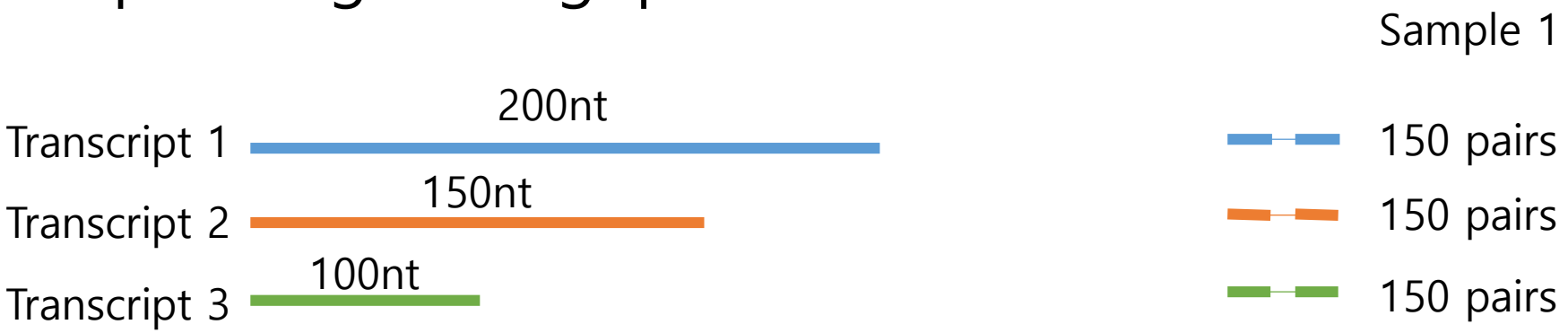
Quantification of Gene Expression (2)

고려해야 할 것들

- Absolute quantification
- Sequencing throughput (depth)
- Gene length
- Transcript variant (different exon usage)



- Gene length
- Sequencing throughput



$$\hat{f} \propto \frac{\frac{150}{450}}{200} = \frac{1}{600}$$

Number of reads mapped in transcript 1

Total number of reads

Length of transcript 1

Relative abundance of Transcript 1 in sample 1

RPKM – Reads Per Kilobase per Million mapped reads

1. 샘플의 total read에 1,000,000을 나눈다. 이것이 per million scaling factor이다.
2. Read counts를 per million scaling factor로 나눈다. 이것이 reads per million (RPM)
3. RPM을 유전자(transcript)의 (평균)길이(kb 단위)로 나눈다. 이것이 RPKM이다.

FPKM은 read pair 수를 이용하면 된다. Pair 중 하나만 mapping이 된 경우, 1로 취급한다.

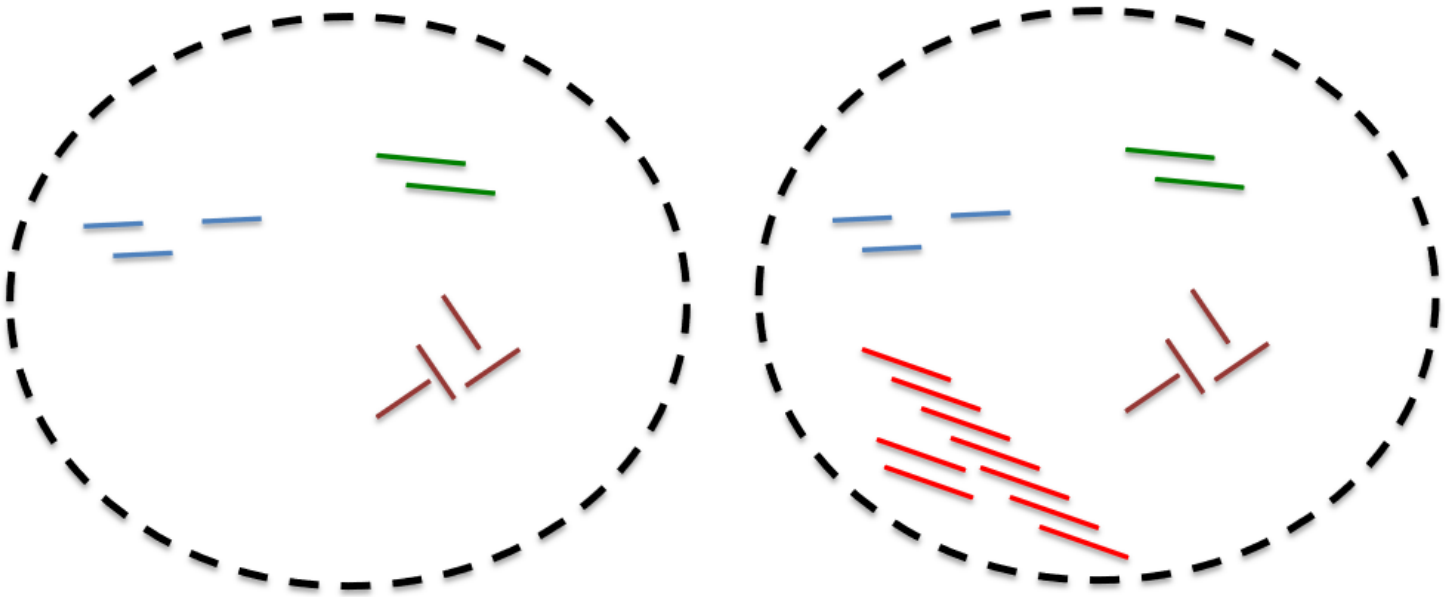
TPM – transcript per million

1. Read count를 각 유전자의 길이(kb 단위)로 나눈다. 이것이 reads per kilobase(RPK)
2. 한 샘플의 모든 transcript의 RPK를 다 더해서, 1,000,000로 나눈다.
   이것이 per million scaling factor이다.
3. RPK를 per million scaling factor로 나눈다. 이것이 TPM이다.

https://www.rna-seqblog.com/rpkm-fpkm-and-tpm-clearly-explained/

Normalization for comparing a gene across samples

1. Technical spike-in
2. Cross-sample normalization



$$\hat{s}_j = \underset{i}{\text{median}} \frac{k_{ij}}{\left(\prod_{v=1}^{m} k_{iv}\right)^{1/m}}$$

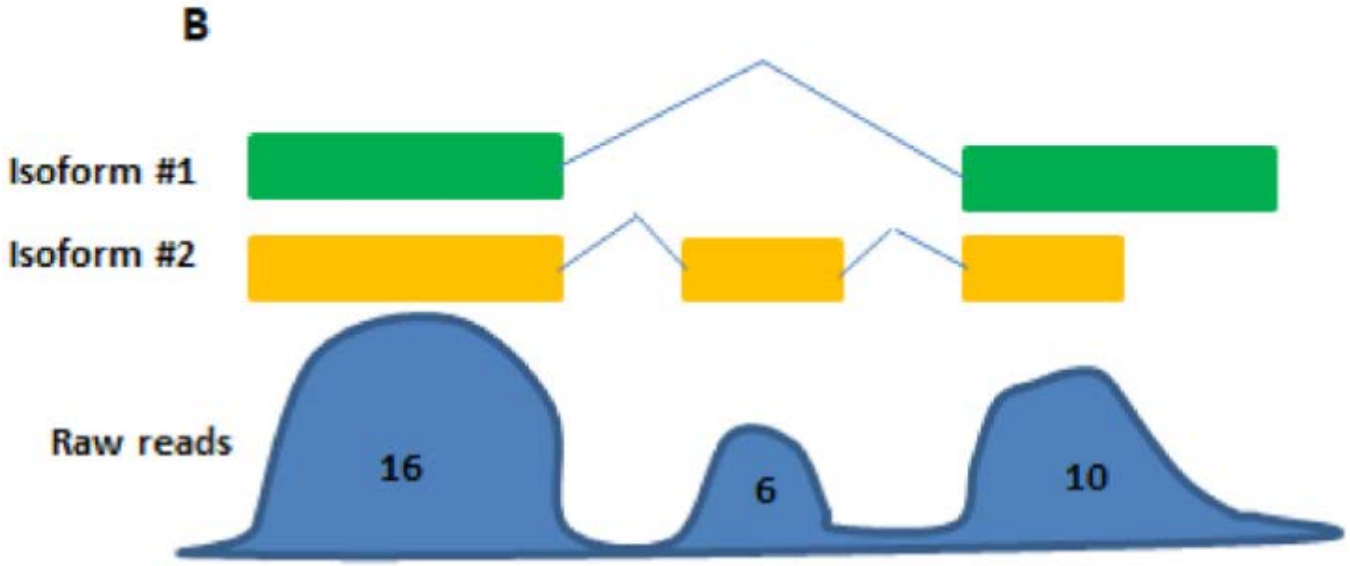
Gene i의 여러 샘플에서의 read count의 geometric mean

Size factor of sample j

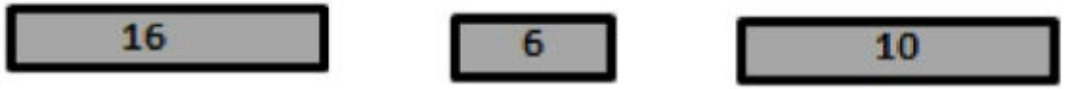
$$\hat{q}_{i\rho} = \frac{1}{m_\rho} \sum_{j:\rho(j)=\rho} \frac{k_{ij}}{\hat{s}_j}$$

http://bioinfo.umassmed.edu/content/pdf2014fall/Week4.pdf

Zhao, Shanrong, Li Xi, and Baohong Zhang. "Union exon based approach for RNA-seq gene quantification: To be or not to be?." *PLoS One* 10.11 (2015): e0141910.

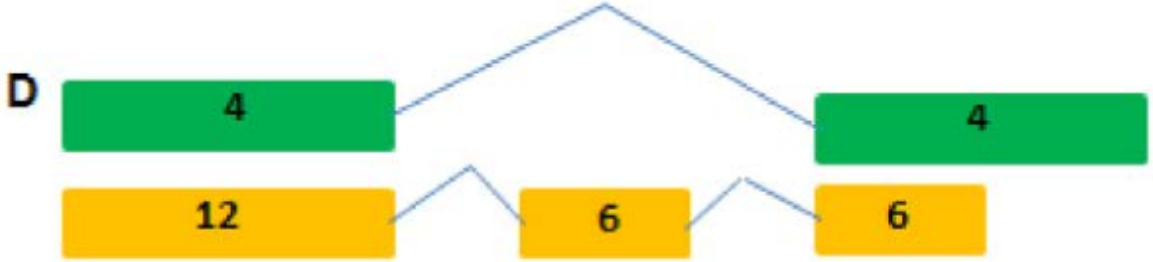


union-exon based approach



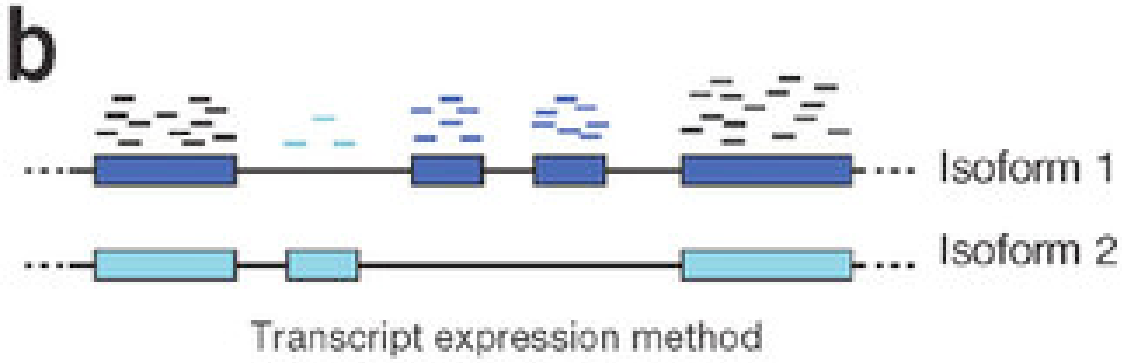
Total gene length after exon flattening: **5kb**
Total reads: **32**
RPKM for gene: **6.4 (=32/5)**

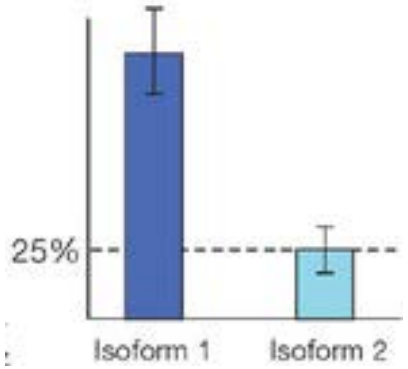
Transcript based approach

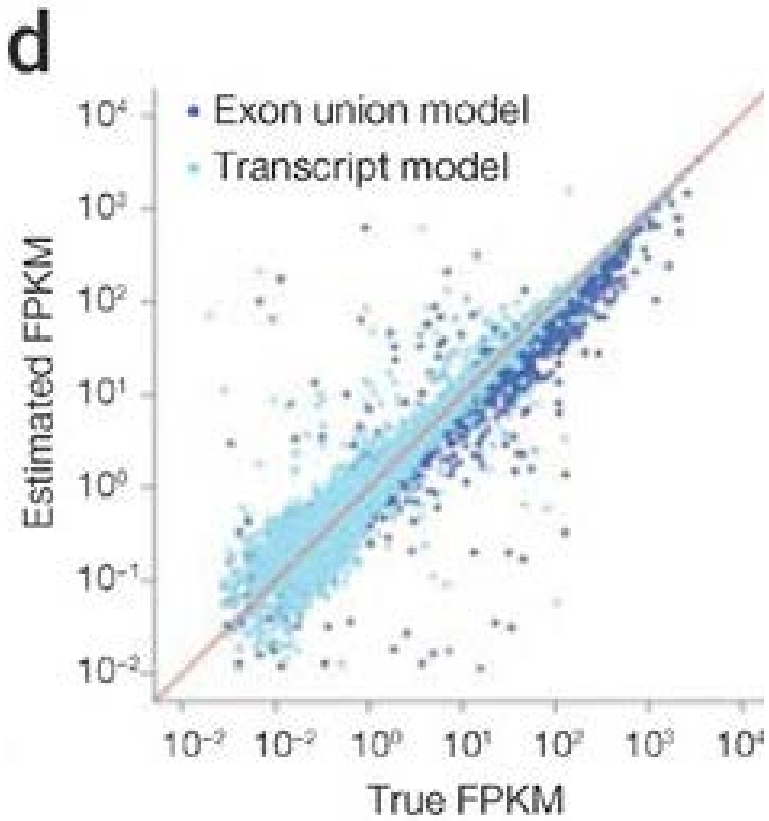


Relative isoform abundance (#1/#2): **25% / 75%**
RPKM for isoform #1 and #2: **2 and 6**
RPKM for gene (=sum of isoforms): **8 (=2+6)**

Garber et al. Nature Methods 8, 469–477 (2011)





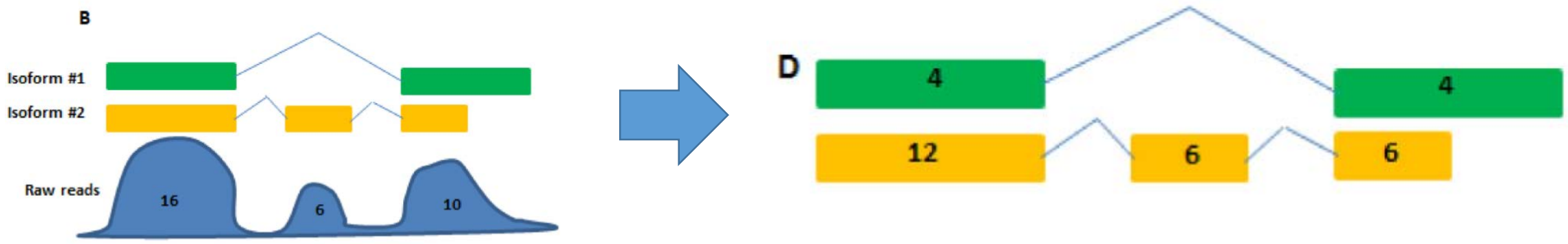


Exon union model underestimate gene expression

Transcript model을 이용하는 counting을 해야 한다

RSEM

- 주어진 read들 (sequencing 결과)를 가장 잘 설명해주는 transcript abundance를 확률 모델과 expectation maximization 기법으로 추정함



https://bmcbioinformatics.biomedcentral.com/articles/10.1186/1471-2105-12-323

RSEM 실습

준비

```
$ mkdir -p ~/day2/star
$ cd ~/day2/star
$ ln -s ~/../kjyi/day2/star/* .
$ cd ..
```

```
NSCLC_01_NTIL._STARpass1
NSCLC_01_NTIL.Aligned.sortedByCoord.out.bam
NSCLC_01_NTIL.Aligned.sortedByCoord.out.bam.bai
NSCLC_01_NTIL.Aligned.toTranscriptome.out.bam
NSCLC_01_NTIL.Chimeric.out.junction
NSCLC_01_NTIL.Chimeric.out.sorted.bam
NSCLC_01_NTIL.Chimeric.out.sorted.bam.bai
NSCLC_01_NTIL.Log.final.out
NSCLC_01_NTIL.Log.out
NSCLC_01_NTIL.Log.progress.out
NSCLC_01_NTIL.ReadsPerGene.out.tab
NSCLC_01_NTIL.SJ.out.tab
```

~/day2/run_rsem.sh를 아래와 같이 작성

```
/usr/local/bin/rsem-calculate-expression \
    --num-threads 2 \
    --no-bam-output \
    --estimate-rspd \
    --bam ./star/NSCLC_01_NTIL.Aligned.toTranscriptome.out.bam \
    /home/users/kjyi/ref/hg19/rsem_reference/rsem_reference \
    NSCLC_01_NTIL.rsem
```

30분 정도 소요

사용법 확인

```
$ rsem-calculate-expression 2>&1 | less
```

Quantification

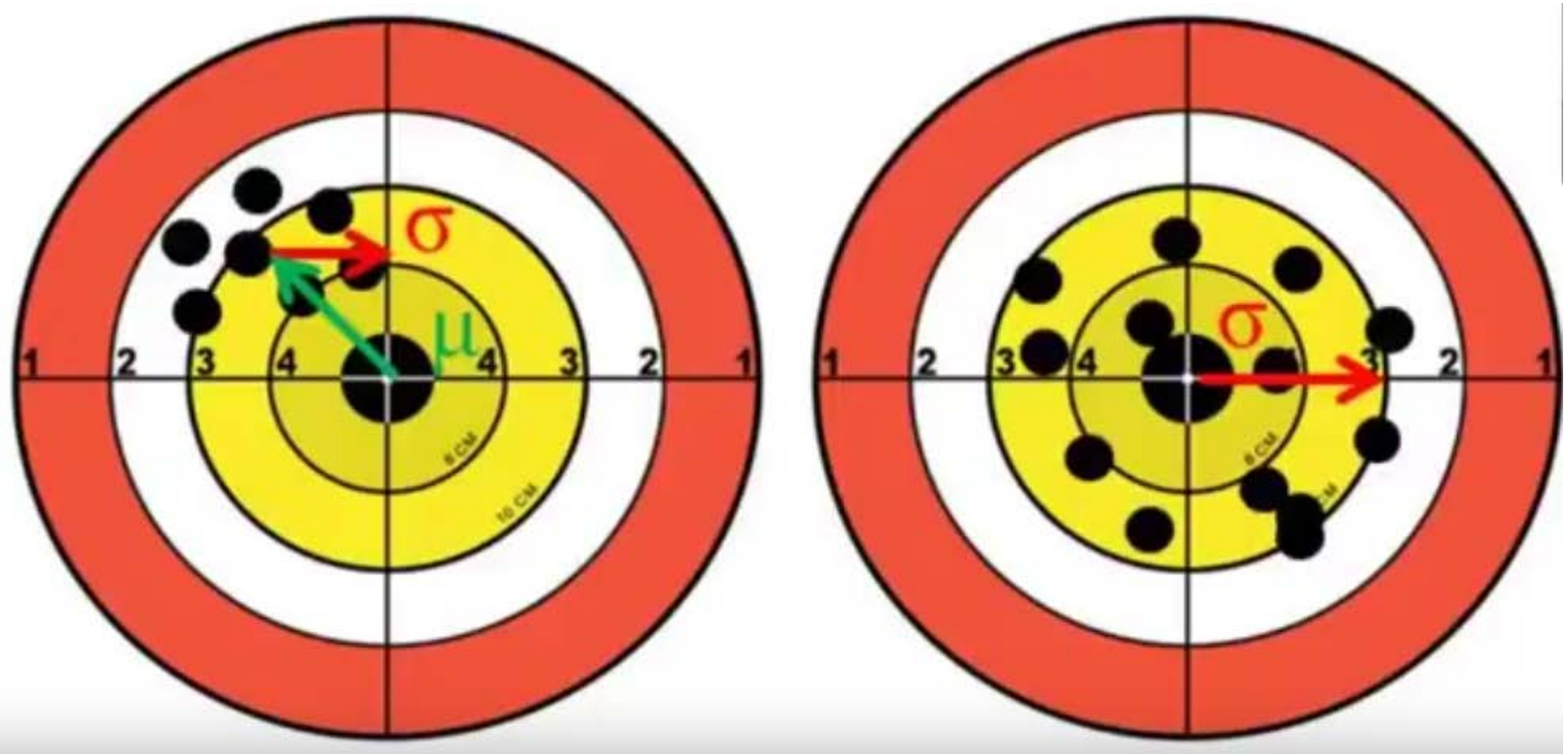
Normalization

Advanced bash commands

(for, while, if, md5sum)

Introduction to R

Normalization



Systematic measurement의 error를 보정하는 목적

Mean and Standard deviation
Z-score normalization

Mean

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i$$

Standard Deviation

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2}$$

Z-score normalization

			mean	sd	(x-mean)/sd		
2	4	4	3.33	1.15	-1.2	0.6	0.6
5	4	14	7.67	5.51	-0.48	-0.67	1.15
4	6	8	6	2	-1	0	1
3	5	8	5.33	2.52	-0.93	-0.13	1.05
3	3	9	5	3.46	-0.58	-0.58	1.15

Quantile normalization example

Original

2	4	4
5	4	14
4	6	8
3	5	8
3	3	9

Ranked

2	3	4
3	4	8
3	4	8
4	5	9
5	6	14

Averaged

3	3	3
5	5	5
5	5	5
6	6	6
8	8	8

Re-ordered

3	5	3
8	5	8
6	8	5
5	6	5
5	3	6

https://www.youtube.com/watch?v=v0j4guy_z30

Quantile normalization 방법 자세히 소개 (쉬운 강의)

https://www.youtube.com/watch?v=v0j4guy_z30

다양한 normalization 방법을 자세히 소개 (어려움)

Anders, Simon, and Wolfgang Huber. "Differential expression analysis for sequence count data." *Genome biology* 11.10 (2010): R106.

Take home message

- Gene expression profile은 cell 내의 여러 transcript의 relative abundance를 조사하는 것이다.
- Absolute quantification을 위해 technical spike-in이 도움이 될 수 있다.
- Normalization은 technical bias를 compensation하는 주 목적이다.
- Different exon usage로 인해 Quantification 방법에 따라 결과가 달라진다
  - Union-exon based approach < Transcript based approach (RSEM)