Introduction to Accelerated Genomics

Lesson 01 - Introduction to Sequencing

DNA Sequencing process

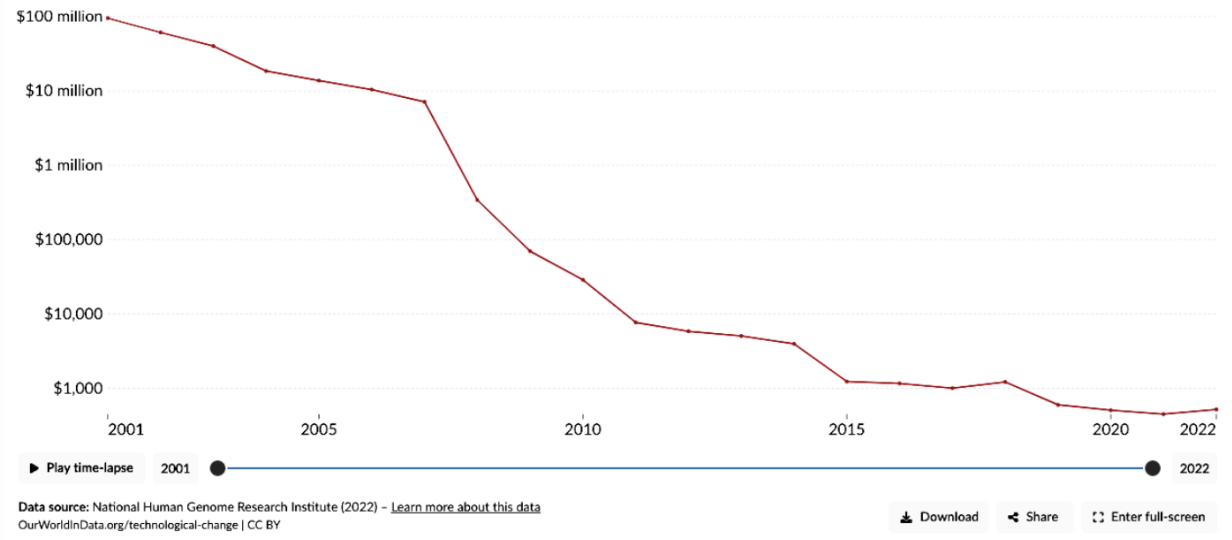
- Process of determining the order of nucleotides in a genome
- Sanger Sequencing - first-generation sequencing methods
  - Produces DNA fragments of varying lengths terminated at specific bases
  - Fragments separated by size and read via fluorescent tags

Next Generation Sequencing (NGS)

- Increasing the throughput by massively parallelising the sequencing process

NGS advantages over Sanger

- High throughput: Faster turnaround times
- Cost-effective: Reduced human genome sequencing from millions of dollars to affordable levels



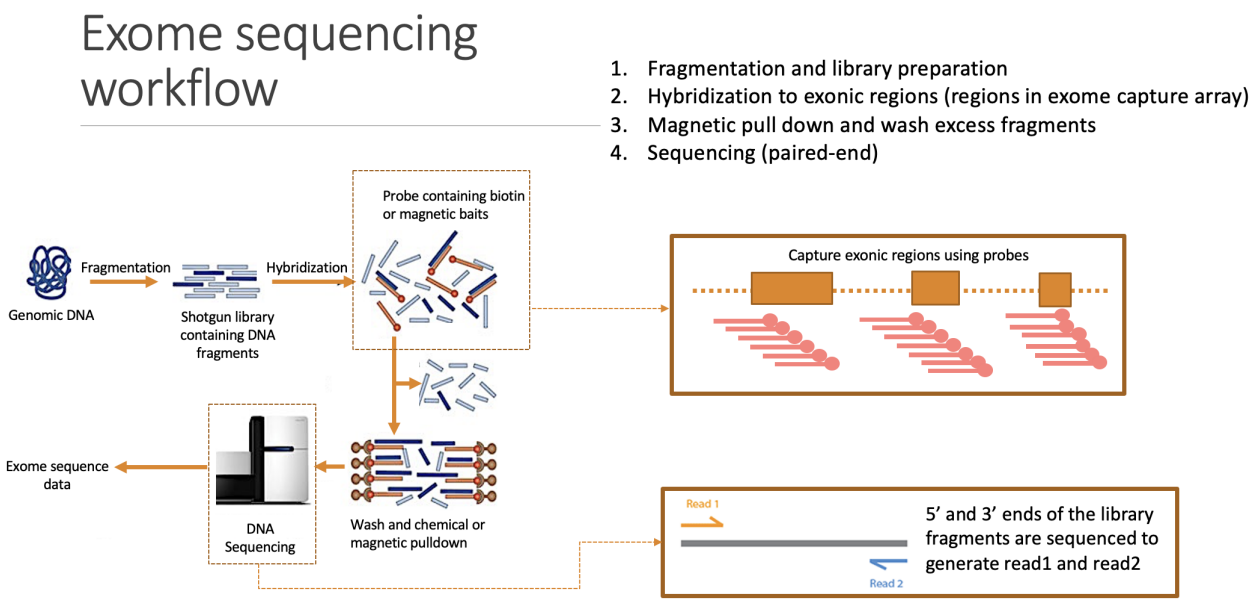
https://ourworldindata.org/grapher/cost-of-sequencing-a-full-human-genome

-
- Large-scale applications: Enables population-scale studies and generates terabytes of data
- Modern accuracy: >99% variant detection accuracy (improved from early NGS limitations)
- Versatile read lengths: Both short-read and long-read platforms available

NGS - Exome vs Whole-Genome Sequencing

Exome Sequencing (WES)

- Focuses on protein-coding regions (~2% of genome)
- Contains ~85% of disease-causing mutations in Mendelian disorders
- Cost-effective tool for genetic disease analysis



Output - Raw sequence data (FASTQ)

FASTQ files from Illumina sequencing technology list "reads" from the sequencing machine - Read-1 and Read-2 FASTQ files. Each entry (i.e., single read) in a FASTQ file contains 4 lines

- First line: A sequence identifier with metadata on the sequencing run the read
- The sequence (i.e., the base calls)
- A separator
- Quality scores of each base call.

Single entry in a FASTQ file

@ERR059938.60 HS9_6783:8:2304:19291:186369#7/1
GTCTCCGGGGGCTGGGGGAACCAGGGGTTCCCACCAACCACCCTCACTCAGCCTTTTCC
CTCCAGGCATCTCTGGGAAAGGACATGGGGCTGGTGCGGGG
+
7?CIGJB:D:-F7LA:GI9FDHBIJ7,GHGJBKHNI7IN,EML8IFIA7HN7J6,L6686LCJE?
JKA6G7AK6GK5C6@6IK+++?5+=<;227*6054

Additional reading materials

- FASTQ files explained from knowledge.illumina.com
- About FASTQ sequence read files from IGSR resources

Key Challenges of WES

- Limited to protein-coding regions (disease-relevant non-coding variants)

Whole-Genome Sequencing

- Analyze the whole genome, including coding, non-coding, and mitochondrial DNA
- Discover a broad range of variants

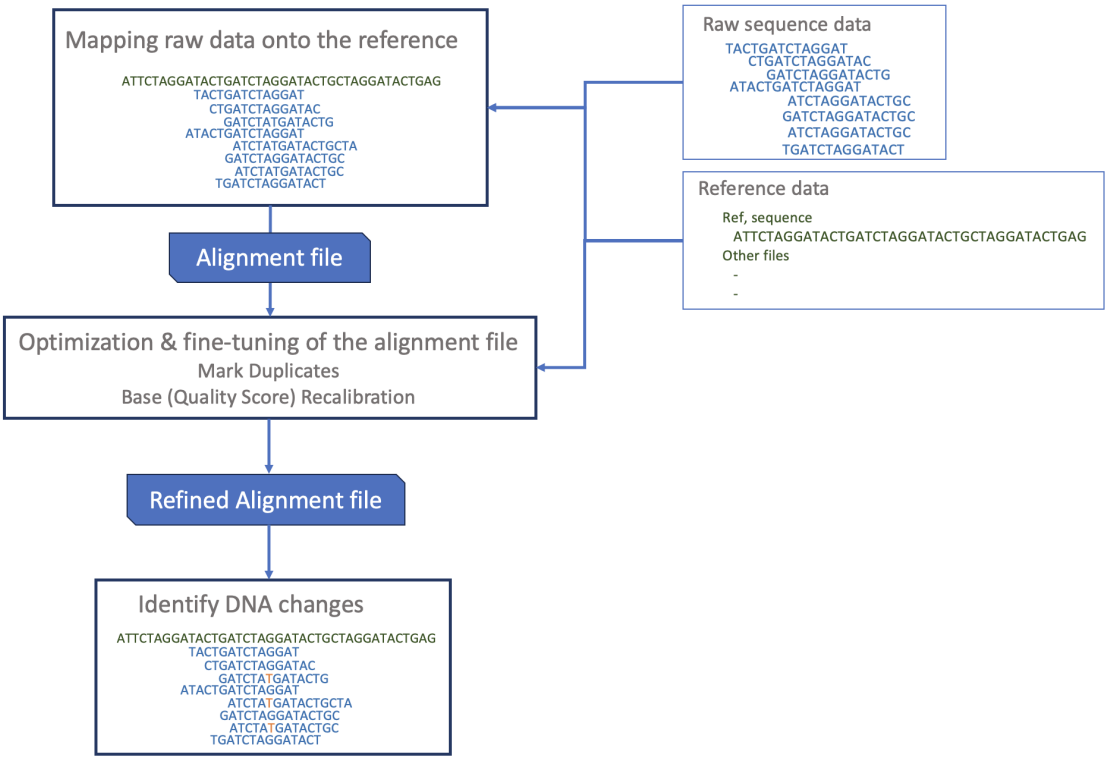
Whole Genome Sequencing (WGS) Challenges

- Data Volume & Computational Burden
  - Massive data: Individual samples: dozens to 250 GB; daily output: terabytes
  - CPU limitations: Traditional processors can't keep pace with data generation
    - Inefficient parallelization: Scatter-and-gather overheads, only 2-3x speedup despite increased resources
- Speed & Turnaround Time
  - Slow processing: Standard WGS takes days (over 24 hours) per sample on CPU-only systems
  - Clinical urgency: this turnaround is insufficient for critically ill patients

Lesson 02 - Basic NGS data processing workflow

NGS data processing workflow

A basic NGS processing workflow comprises several key steps.



- Mapping raw data onto the reference
- Optimization & fine-tuning of the alignment file
    - Mark Duplicates
    - Base (Quality Score) Recalibration
- Identify DNA changes

Main steps in a germline variant calling (short variants - InDel and SNP calling)

Basic best-practices NGS processing workflow (CPU-native)

A GATK best-practices workflow runs several GATK tools to call SNVs and InDels from sequence datasets

- Read mapping and alignment refinement stages:
    - Step 1: BWA read mapping and sorting
    - Step 2: GATK MarkDuplicates
    - Step 3: GATK BaseRecalibrator
    - Step 4: GATK ApplyBQSR
- SNVs and InDels calling can be performed by using a caller like HaplotypeCaller (Step 5)

Lesson 03 - NVIDIA Parabricks

NVIDIA Parabricks

- Specialized software suite for accelerating NGS data analysis using GPUs
- Originally developed at University of Michigan, later acquired by NVIDIA
- Leverages GPUs to optimize computationally intensive NGS processing
  - Adapts existing GATK-based algorithms to GPU-enabled CUDA code
  - Parallelizable tasks on GPUs, sequential functions on CPUs
- Delivers improved computing time: i.e., low turnaround time
- Enable scalable genomic analysis
- Maintains high accuracy and compliance with standard genomic formats
- Parabricks documentation:
- "Parabricks was built from the ground up by GPU computing and Deep Learning experts who wanted to develop the fastest and most efficient possible implementation of common genomics algorithms."

Tools in Parabricks

Software Tools in v4.6.0

The Parabricks somatic (Somatic Variant Caller), germline (GATK Germline Pipeline) and deepvariant_germline tools are collections of several other individual tools that are commonly used together, all wrapped up as a single tool. For example, the deepvariant_germline takes FASTA and FASTQ files as input and produces a VCF and BAM file as output. Internally, it runs BWA mem alignment, performs coordinate sorting, marks duplicates, and then runs DeepVariant.

The following tools are available in the NVIDIA Parabricks software. Click on one of the tool names below for tool-specific options.

Parabricks Tool	Details
applybqsr	Apply BQSR report to a BAM file and generate a new BAM file
bam2fq	Convert a BAM file to FASTQ
bammetrics	Collect WGS Metrics on a BAM file
bamsort	Sort a BAM file
bqsr	Collect BQSR report on a BAM file
collectmultiplemetrics	Collect multiple classes of metrics on a BAM file
dbsnp	Annotate variants based on a dbsnp
deepsomatic	Run GPU-DeepSomatic for calling somatic variants
deepvariant	Run GPU-DeepVariant for calling germline variants
deepvariant_germline	Run the germline pipeline from FASTQ to VCF using a deep neural network analysis
fq2bam (BWA-MEM + GATK)	Run bwa mem, co-ordinate sorting, marking duplicates, and Base Quality Score Recalibration
fq2bam_meth	Run GPU-accelerated bwa-meth compatible alignment, co-ordinate sorting, marking duplicates, and Base Quality Score Recalibration

genotypegvcf	Convert a GVCF to VCF
germline (GATK Germline Pipeline)	Run the germline pipeline from FASTQ to VCF
giraffe (vg giraffe + GATK)	Run GPU-accelerated VG-giraffe compatible pangenome alignment and co-ordinate sorting
haplotypecaller	Run GPU-HaplotypeCaller for calling germline variants
indexgvcf	Index a GVCF file
markdup	Identifies duplicate reads
minimap2	Align long read sequences against a large reference database to convert FASTQ to BAM/
mutectcaller	Run GPU-Mutect2 for tumor-normal analysis
ont_germline	Run the germline pipeline from FASTQ/BAM to VCF by aligning long read ONT sequences with minimap2 and using a deep neural network
pacbio_germline	Run the germline pipeline from FASTQ/BAM to VCF by aligning long read sequences with minimap2 and using a deep neural network analysis
pangenome_aware_deepvariant	Run GPU-Pangenome-aware-deepvariant that uses a pangenome reference for calling germline
postpon	Generate the final VCF output of doing mutect pon
prepon	Build an index for PON file, which is the prerequisite to performing mutect pon
rna_fq2bam	Run RNA-seq data through the fq2bam pipeline
somatic (Somatic Variant Caller)	Run the somatic pipeline from FASTQ to VCF
starfusion	Identify candidate fusion transcripts supported by Illumina reads

Lesson 04 - Key Advantages of Parabricks

The advantages of using Parabricks for NGS processing can be discussed in four main categories

- Exceptional Speed & Efficiency
- Accuracy comparable to best-practices workflows
- Cost-Effectiveness

Exceptional Speedup & Efficiency

- *Speedup: Quantify performance improvements*
  - *Speedup = CPU-only runtime ÷ Accelerated runtime*
- Pipeline speedup:
  - 20-30 speedup for Mapping and 80-130 speedup for Variant calling
  - Drastically reduce the turnaround times (FASTQ to VCF)
- *E.g. 50x WGS processing: Under 1 hours (on H100 GPUs) vs. nearly 24 hours with traditional (CPU-native) GATK v4.1.0*

High Accuracy

- Overall accuracy:
  - >99% accuracy and precision (comparable to CPU-native GATK)
- Clinical validation:
  - 99.5% variant call concordance with validated CPU workflows

Cost-Effectiveness

- Malaria genomics: 4.6-5x cost reduction despite higher hourly GPU costs on cloud solutions
- 1000 P. falciparum genomes: $290 (4-GPU) vs. $1330 (CPU) on local HPC

Lesson 05 - Parabricks workflow (basic workflow)

Step 1: Read mapping and alignment refinement

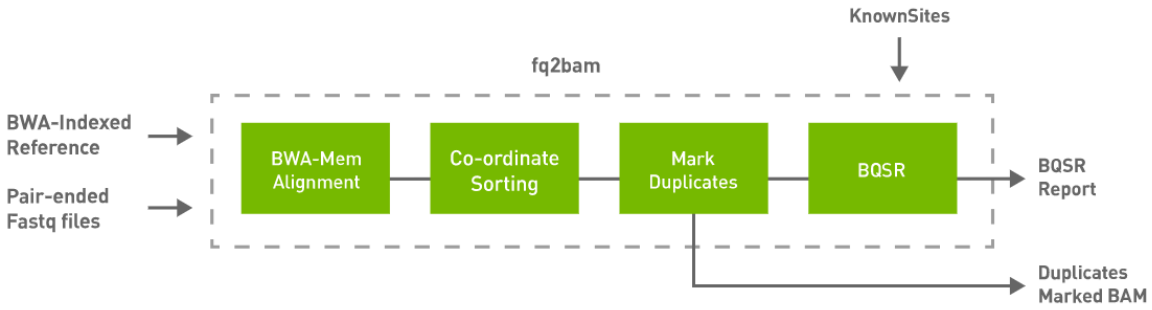
Parabricks Command fq2bam

Perform the following 4 best-practice processes

- BWA read mapping and sorting
- GATK MarkDuplicates
- GATK BaseRecalibrator

Output files:

- Alignment file
- BQSR report



Command fq2bam

Step 2: Apply BQSR report

`Parabricks Command applybqsr`

- Generates refined alignment file
- Compatible GATK4 Command: `gatk ApplyBQSR`

Step 3: Parabricks Variant calling - HaplotypeCaller

- Accelerated SNP and InDel calling

Lesson 06 - Real-world applications of Parabricks

Broader Impact

- Addresses the computational burden of analysing terabytes of genomic datasets generated in the rapid phase
- Enables researchers/clinicians to keep pace with sequencing throughput as the technology advances

Clinical Genomics & Precision Medicine

- Faster diagnosis for critically ill patients (e.g., neonatal intensive care)
- Personalized therapeutic approaches

Pathogen Biosurveillance & Epidemiology

- Rapid processing of pathogen genomic data (P. falciparum) and viral genome data
- Monitor transmission patterns and guide public health responses

Cancer Research

- Cancer mutation signature identification
- Accelerated somatic variant detection

Population-scale Genomic Research

- Scale-up NGS data processing in large-scale population analyses
- Enhanced understanding of genetic diseases and diversity

Lesson 07 - Challenges and Considerations of implementing Parabricks

Despite its numerous advantages, the use of Parabricks and GPU acceleration comes with certain considerations:

Integrated Quality Control (QC):

- Parabrisk may initially lack comprehensive built-in quality control and quality assurance steps
- This necessitates the integration of additional CPU-based QC measures to ensure clinical efficacy and trustworthiness of variants.
    - Parabricks outputs are in standard formats, allowing easy integration with CPU-based QC tools

Benchmarking Requirement:

- Parabricks is closed-source (i.e., source-code is not publicly available). Therefore, independent benchmark studies are important to ensure the trustworthiness of these workflows
- Parabricks can be implemented on different platforms. Therefore benchmarking specific algorithms on these platforms and with representative datasets is crucial