Accelerated Genomics Analysis with NVIDIA Parabricks & RAPIDS

Huiwen (hoi – won) Ju, Solutions Architect, Higher Education & Research, hju@nvidia.com

5/1/2024 Mount Sinai

Agenda

- **NVIDIA Parabricks** for secondary analysis
- **RAPIDS** for tertiary analysis, single-cell RNAseq analysis and spacial genomics

Agenda

- **NVIDIA Parabricks** for secondary analysis
- **RAPIDS** for tertiary analysis, single-cell RNAseq analysis and spacial genomics

NVIDIA is Helping Across the Computational Genomics Workflow

From sequencing sensor to biological insights

TACGTG
TACATG
ATTCATGG
TACATG
ATTCATGG
TACATG
Single Cell
Spatial
TensorRT
NVIDIA CUDA
NVIDIA PARABRICKS
RAPIDS

DNA Basecalling

Error correction / polishing

Alignment / assembly

Variant calling

Interpretation / modelling

Primary

Secondary

Tertiary

NVIDIA Parabricks for Alignment & Variant Calling

Speed, Scale, Accuracy

FastQ

Alignment

BWA-MEM

Minimap2

STAR

Gold Standard Processing and Quality Control

Sort BAM

Mark duplicates

BQSR

BAM/CRAM

BAM Metrics

Collect multiple metrics

BAM2FQ

BAM/CRAM

High-Accuracy Variant Calling

DeepVariant

HaplotypeCaller

Mutect2

STAR-Fusion

GenotypeGVCF

IndexGVCF

VCF/gVCF

Universal Analysis

Industry-standard tools for all major sequencers, ported to GPU

Up to 100x Acceleration

Up to 100x faster for WGS compared to CPU-only

Up to 50% Lower Cost

Up to 50% lower compute cost for WGS compared to CPU-only

Higher Accuracy with AI

The power of deep learning for customized high accuracy analysis

Key Applications of NVIDIA Parabricks

Accelerated and Deep Learning Genomic Analysis

Population Genomics

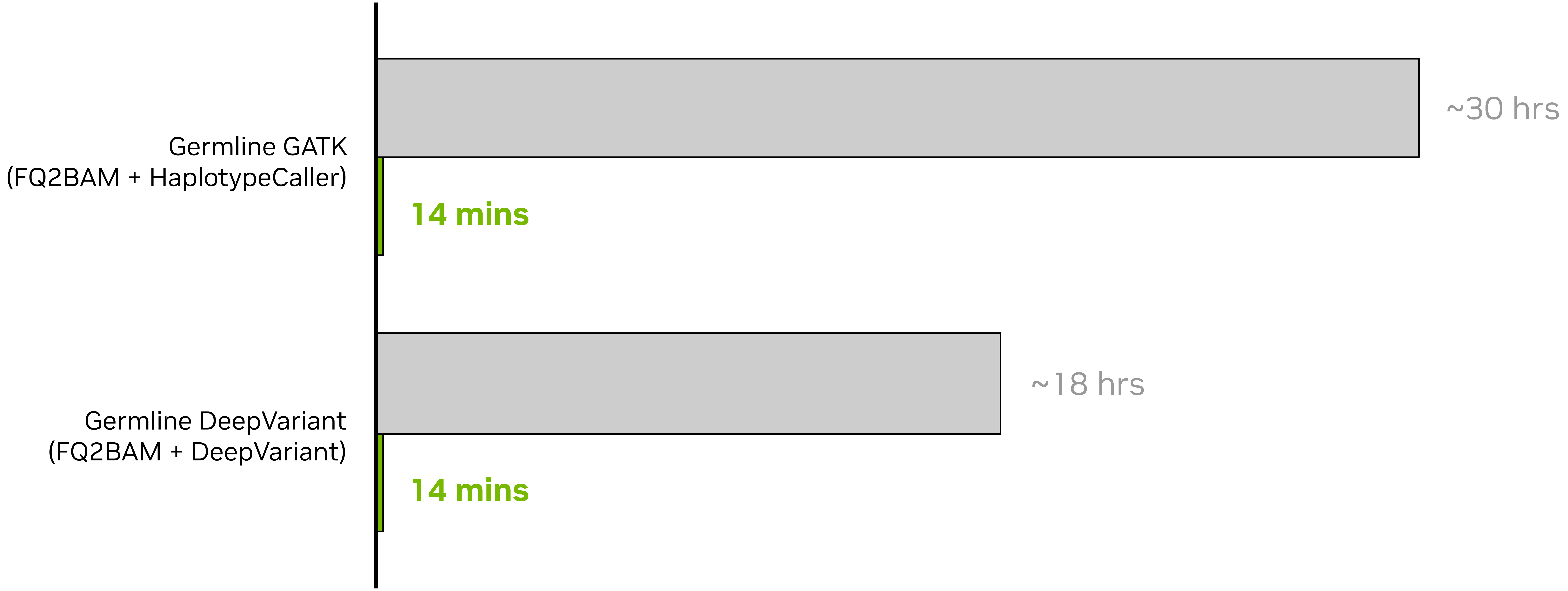
Cancer Genomics

RNA Sequencing

Higher Speed

From hours to minutes

End-to-end germline sample analysis with industry-standard tools in under 15 mins on the new NVIDIA H100 GPUs



Runtimes on CPU (m5.24xlarge)

Runtimes on NVIDIA GPU (8xH100)

Up to 80x Acceleration

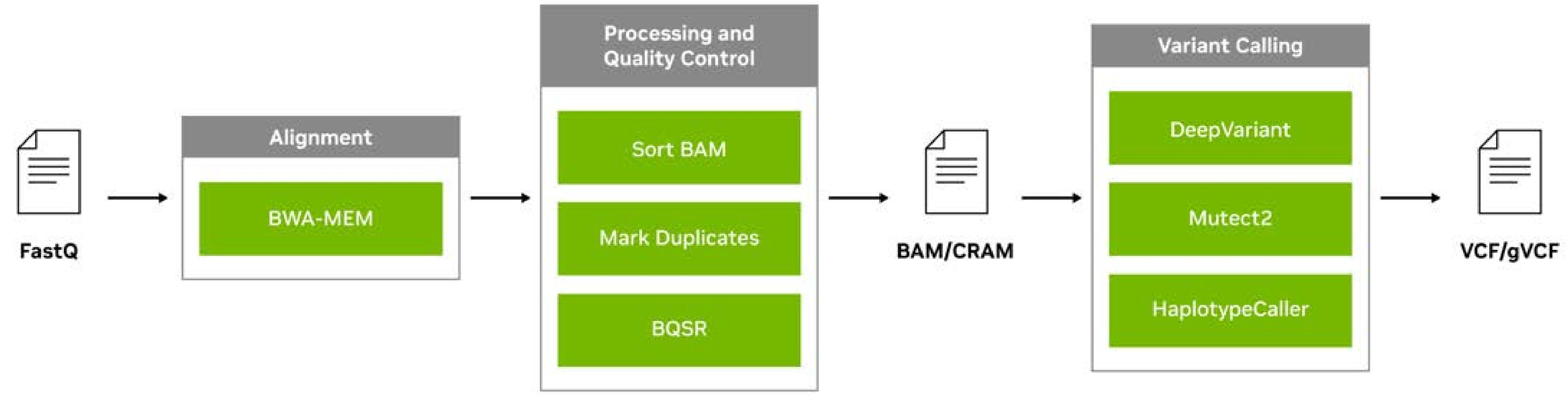
Gold-standard results, faster

Runtimes on CPU
Runtimes on NVIDIA GPU
BWA-MEM
11 mins
~4 hrs
Sort, MarkDups, Apply BQSR
6 mins
~9 hrs
HaplotypeCaller
4 mins
~16 hrs
DeepVariant
11 mins
~5 hrs
Mutect2
45 mins
~31 hrs

v3.8 Benchmarks
Dataset: HG002 30x WGS, except Mutect2 on SEQC2 50x WGS
CPU: m5.24xlarge; GPU: 8xA100, except DeepVariant & Mutect2 on 8xV100

A Universal Analysis Solution

Short-Read



Element Biosciences

illumina

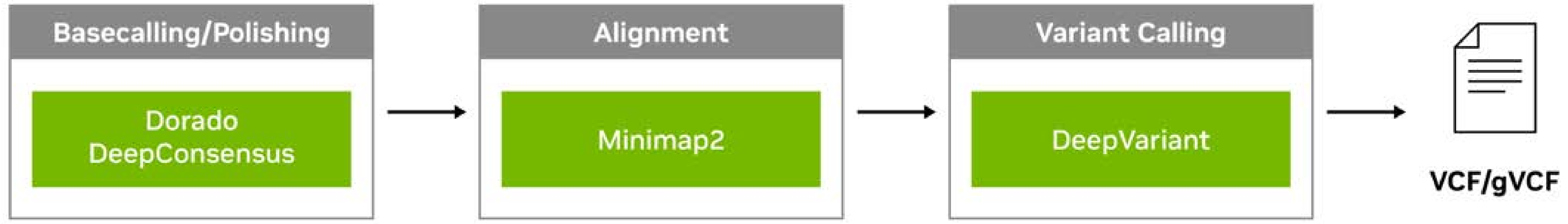
MGI

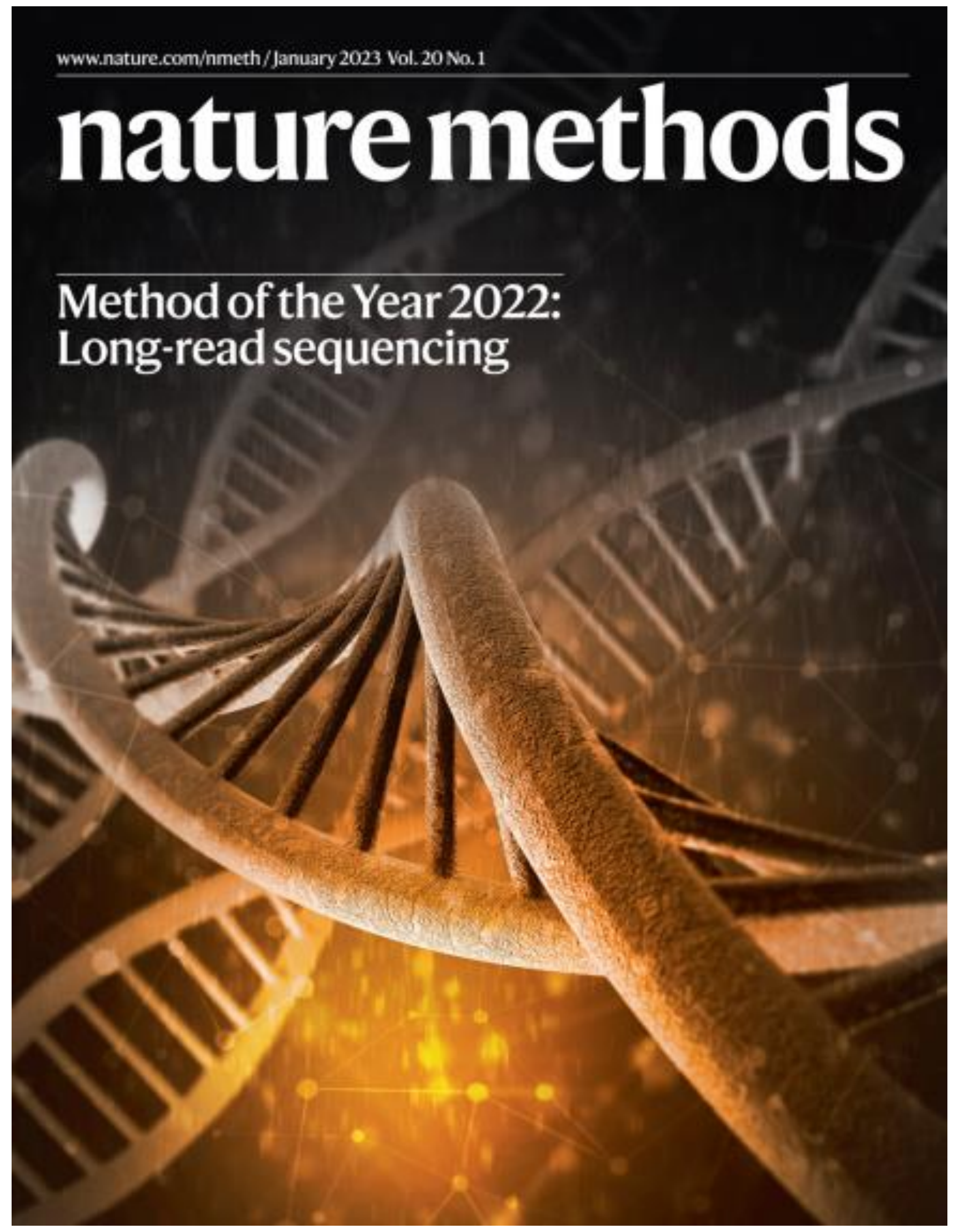
SINGULAR GENOMICS

ThermoFisher SCIENTIFIC

ULTIMA GENOMICS

Long-Read





www.nature.com/nmeth / January 2023 Vol. 20 No. 1

nature methods

Method of the Year 2022:
Long-read sequencing

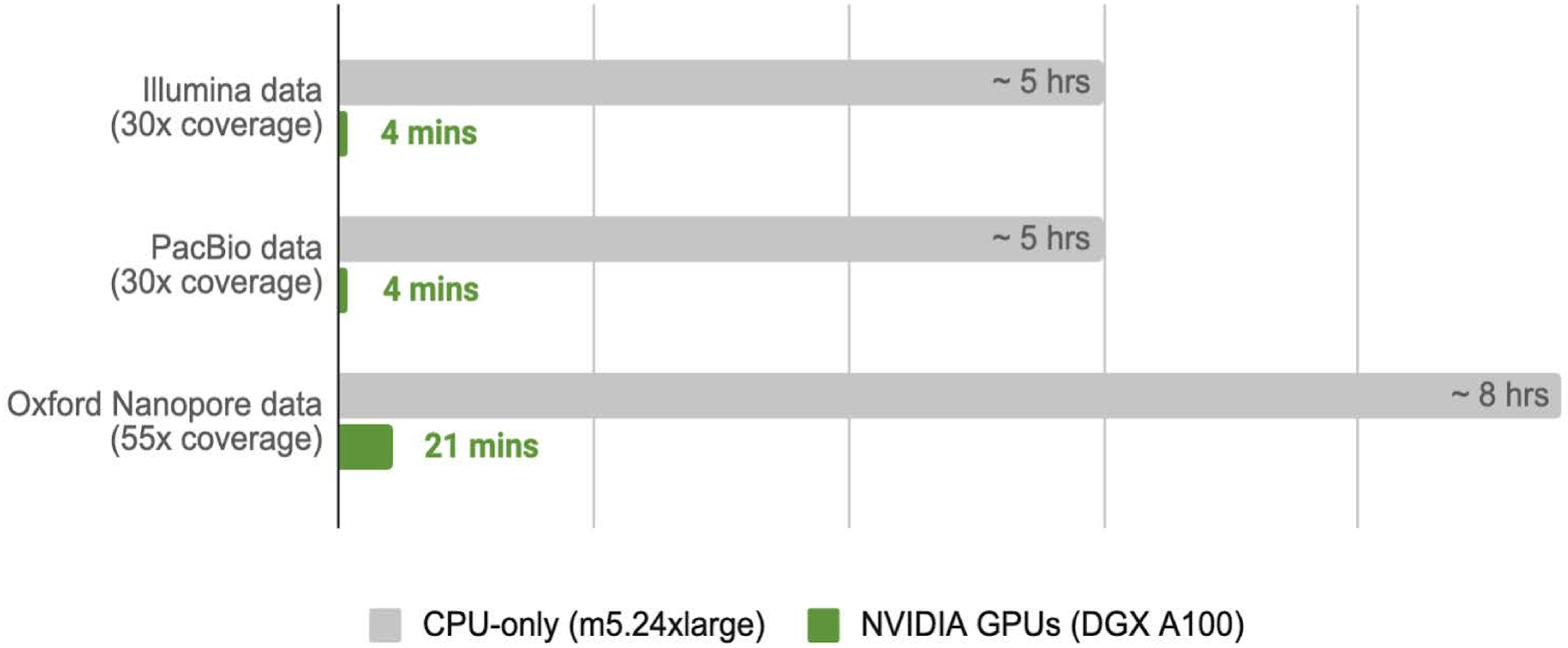
Oxford NANOPORE Technologies

PacBio

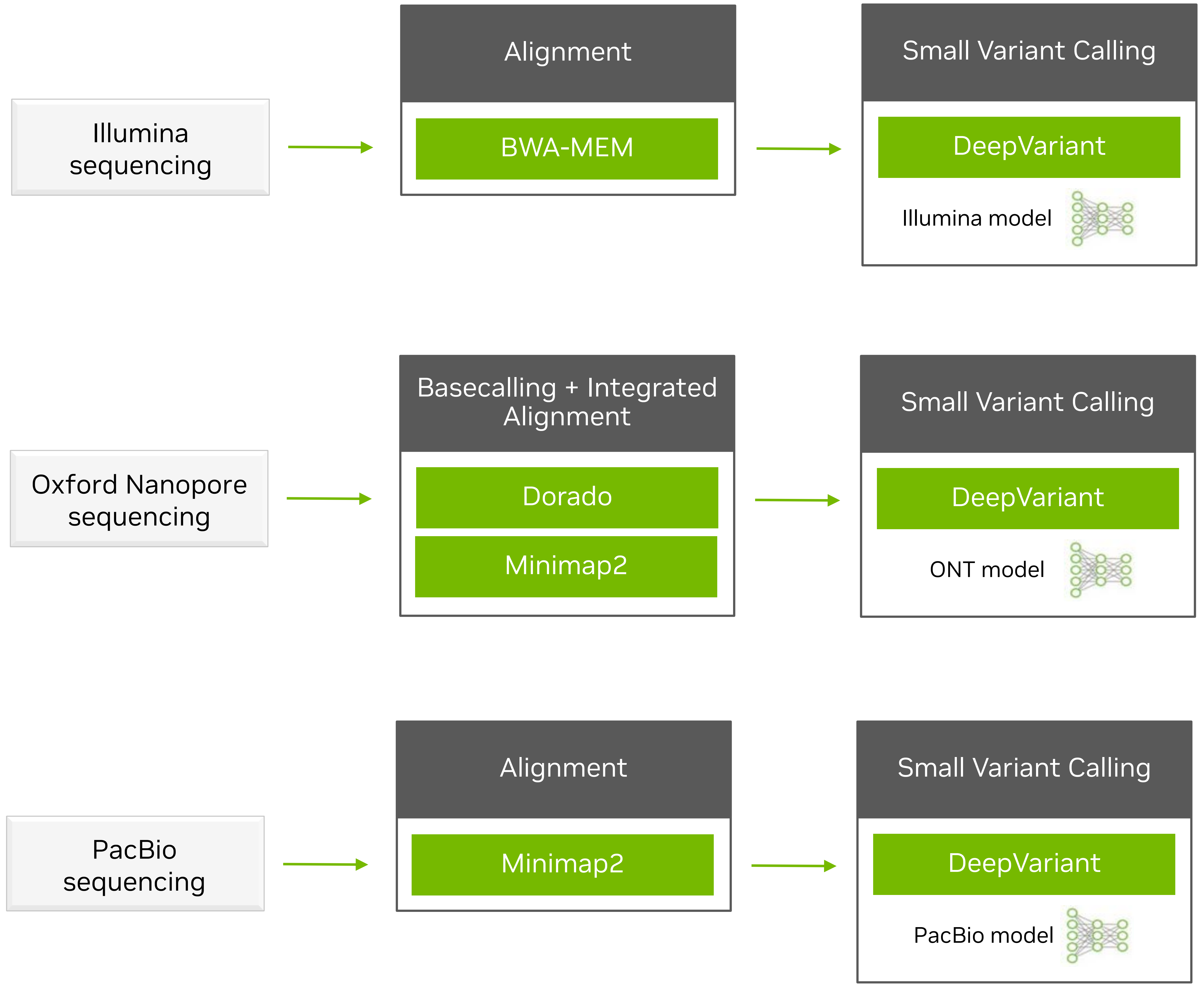
Universal Sequencing Analysis

The only hardware-accelerated solution for multiple sequencers

Accelerated DeepVariant models, for up to 75x faster variant calling



Reference WDL/NextFlow workflows for multiple sequencers

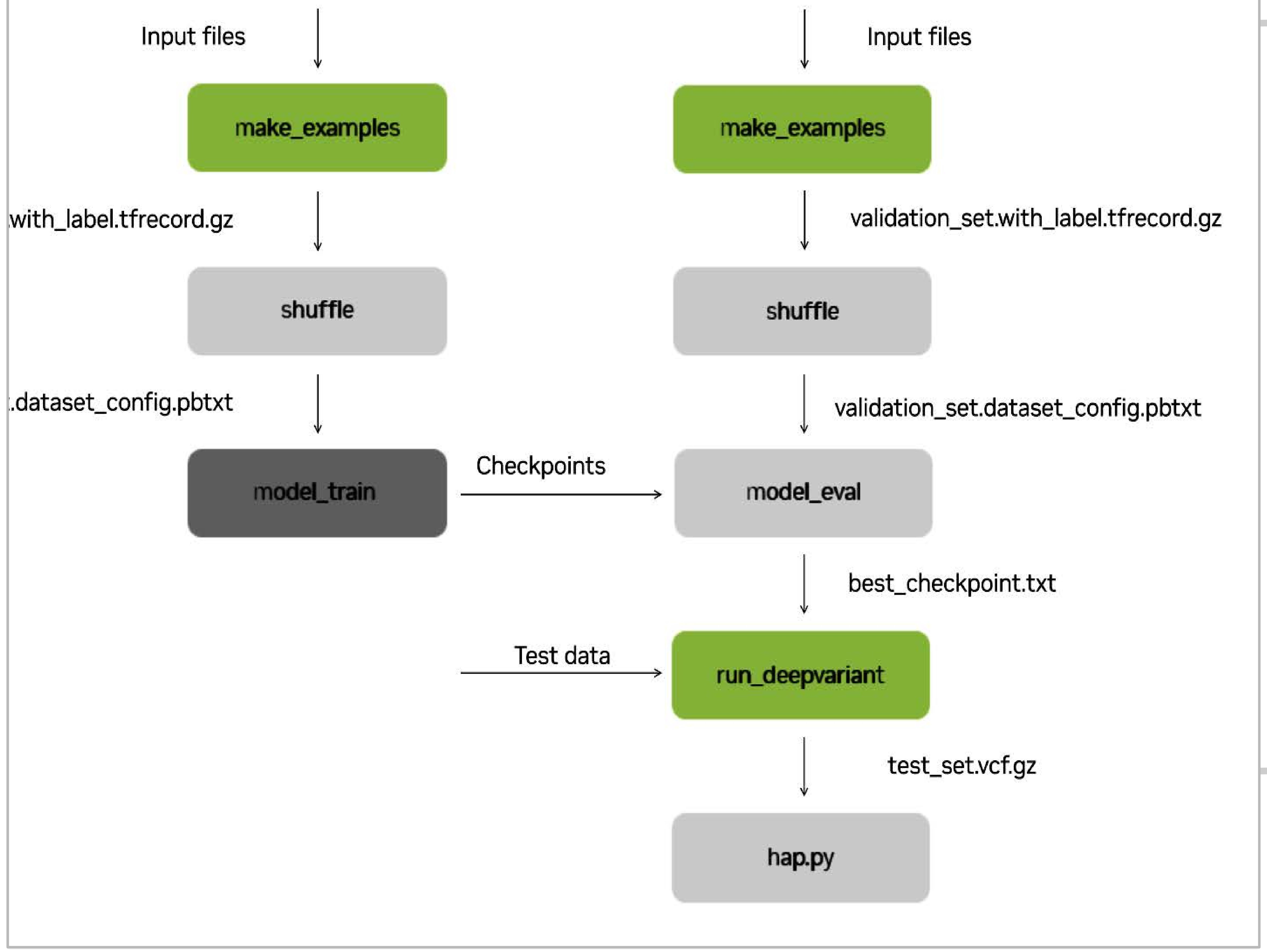


Parabricks DeepVariant Training Framework

Easily Train a Custom Model for Optimal Accuracy

Training

Custom deep learning models optimized for very specific data



Input files

make_examples

with_label.tfrecord.gz

shuffle

.dataset_config.pbtxt

model_train

Checkpoints

Input files

make_examples

validation_set.with_label.tfrecord.gz

shuffle

validation_set.dataset_config.pbtxt

model_eval

best_checkpoint.txt

Test data

run_deepvariant

test_set.vcf.gz

hap.py

For re-training on different sequencers / data types

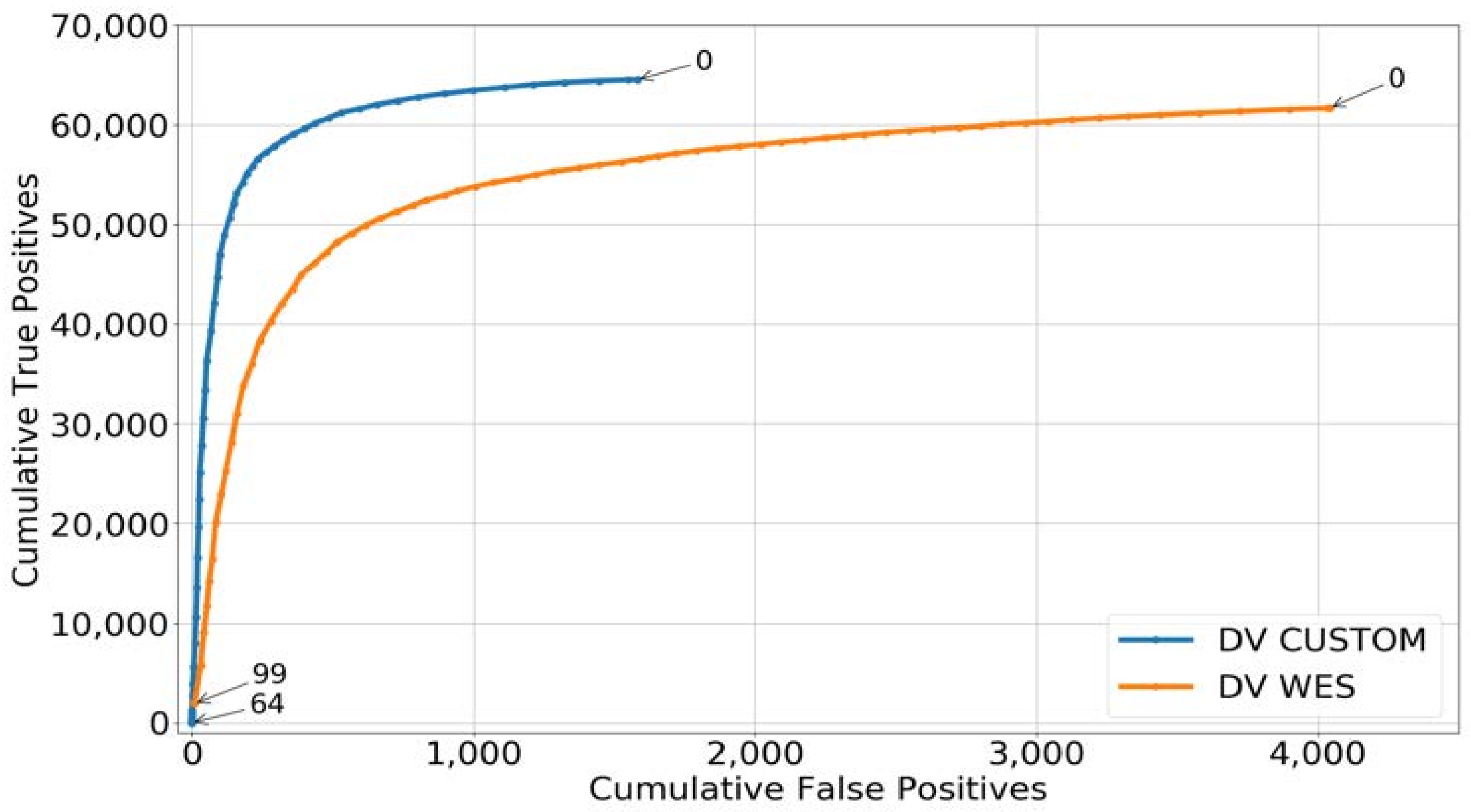
SINGULAR GENOMICS

ULTIMA GENOMICS

For finetuning to individual labs protocols

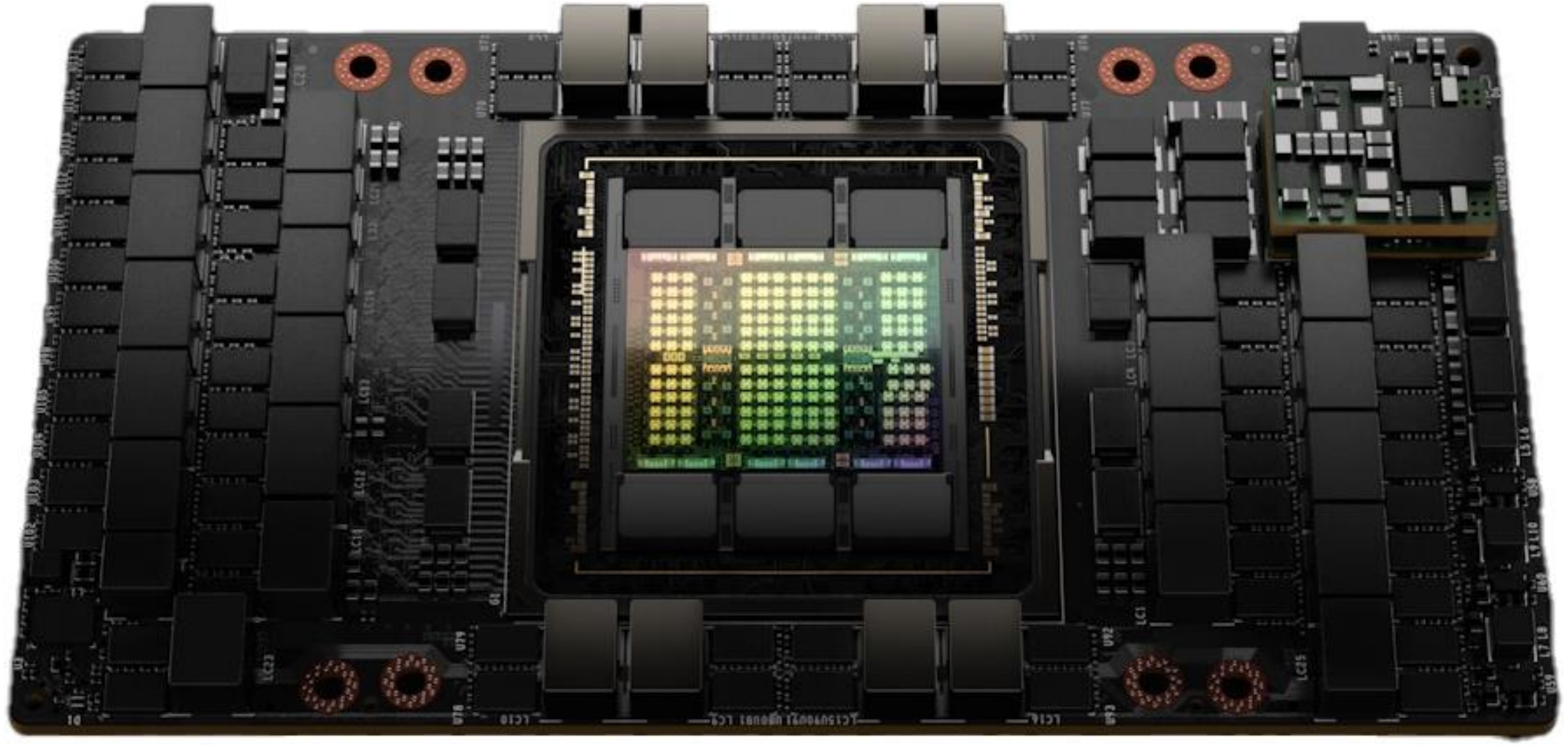
RGC™

Regeneron Genetics Center



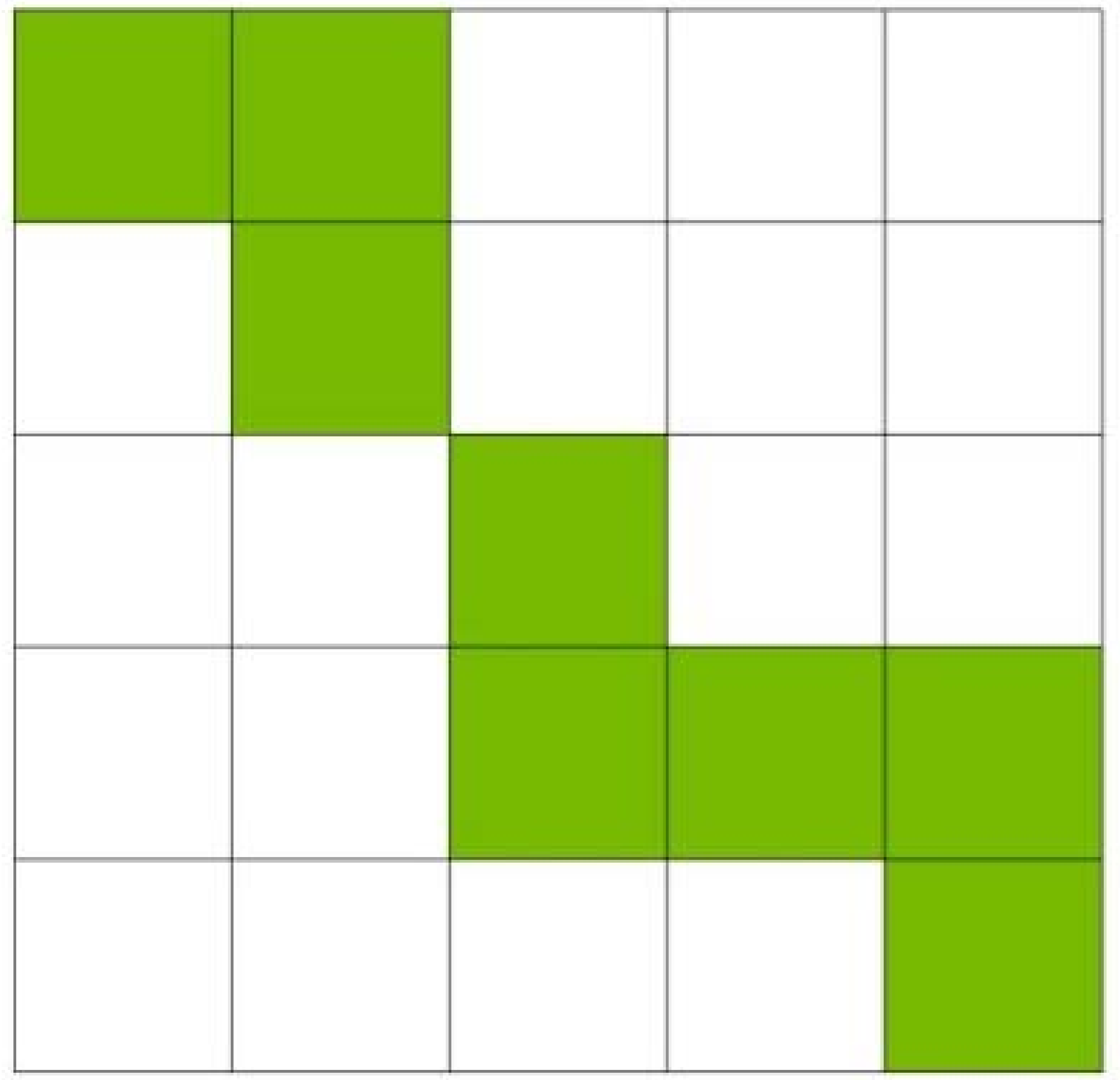
NVIDIA GPUs for Secondary Analysis

The H100 Dynamic Programming Core



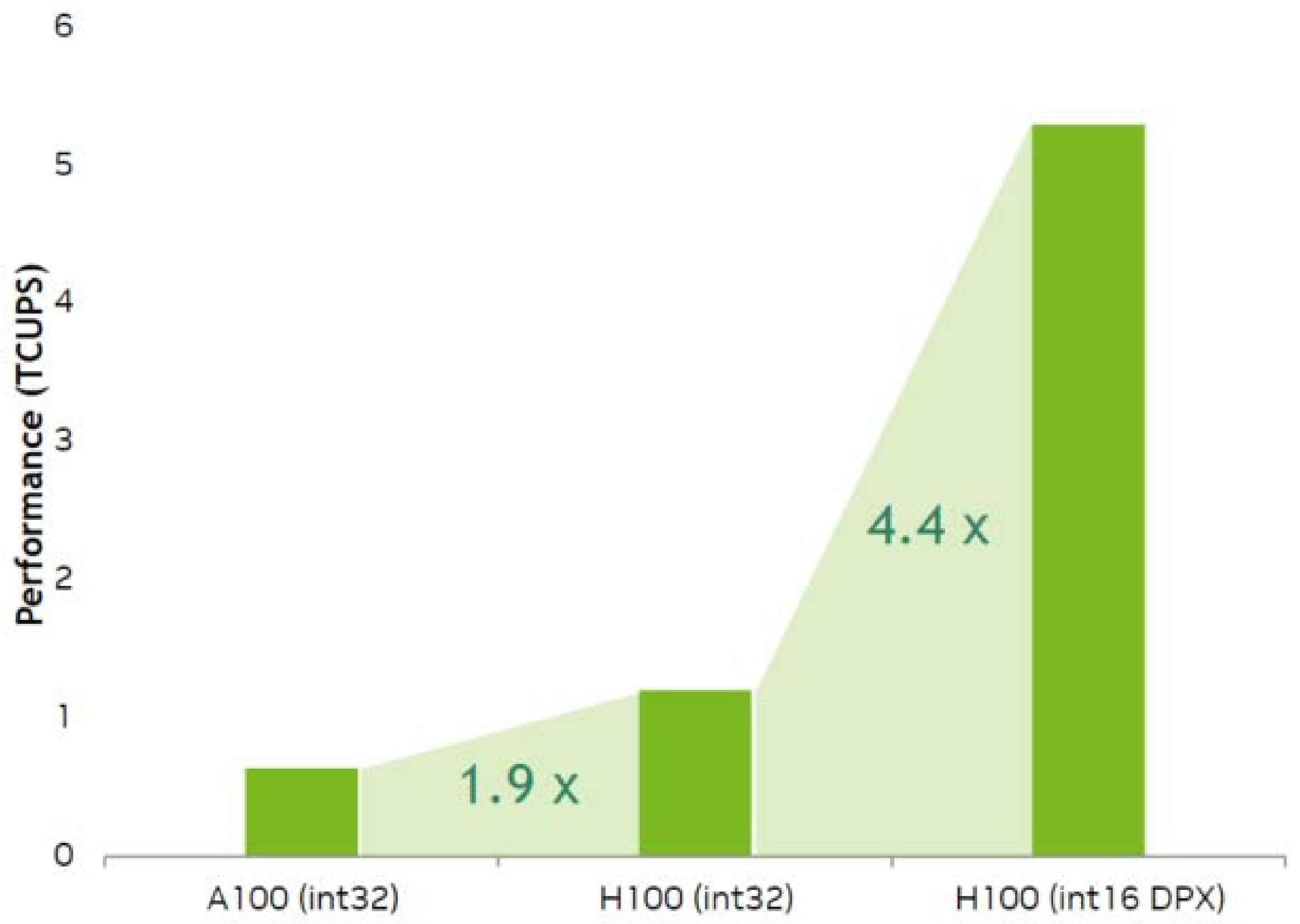
Dynamic Programming

Exponential to polynomial time problem solving



Supercharged Smith-Waterman

35x acceleration over CPU
7x acceleration over Ampere



Affine gap alignment (score calculation)
Weights used from BWA.
Data: HG002 (NA24385) paired-end protocol using Illumina Sequencers.

Boosting Dynamic Programming Performance Using NVIDIA Hopper GPU DPX Instructions

User Success Stories

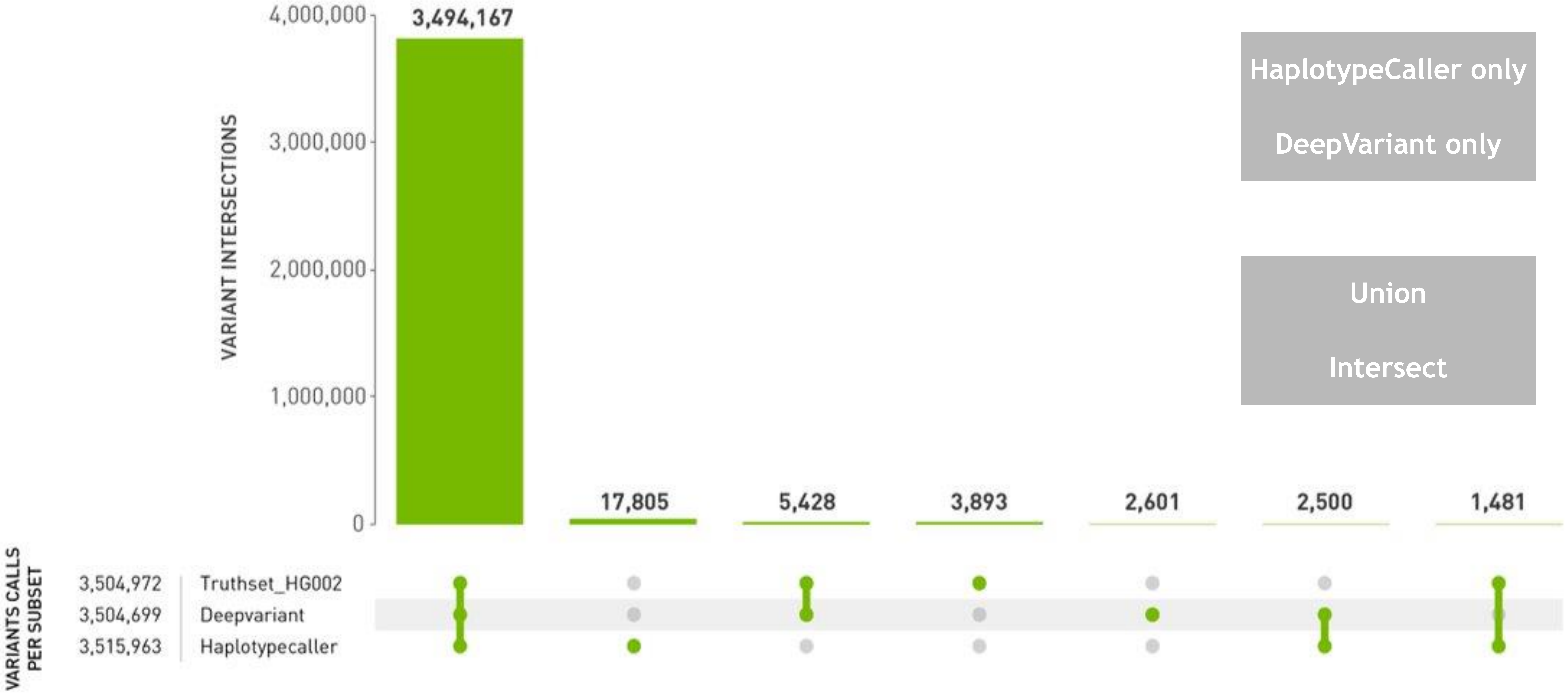
Large Population Study

UK Biobank's 470,000+ exomes analyzed by Regeneron as of July 2022

- Regeneron Genetics Center Sequencing close to 500,000 Exomes
- Exomes were analyzed in 5 minutes with Clara Parabricks versus 1 hour in CPUs
- The cost went down 60% on GPUs
- DeepVariant optimized for RGC outputs, especially alignment.
- Analysis of DNA given to researchers on UK Biobank which span academia, pharma and other scientists interested in genomic variants for diseases

MULTI-TOOL IMPLEMENTATION PROVIDES THE BEST RESULTS

Comparing germline calling between GATK Haplotype and Google's DeepVariant



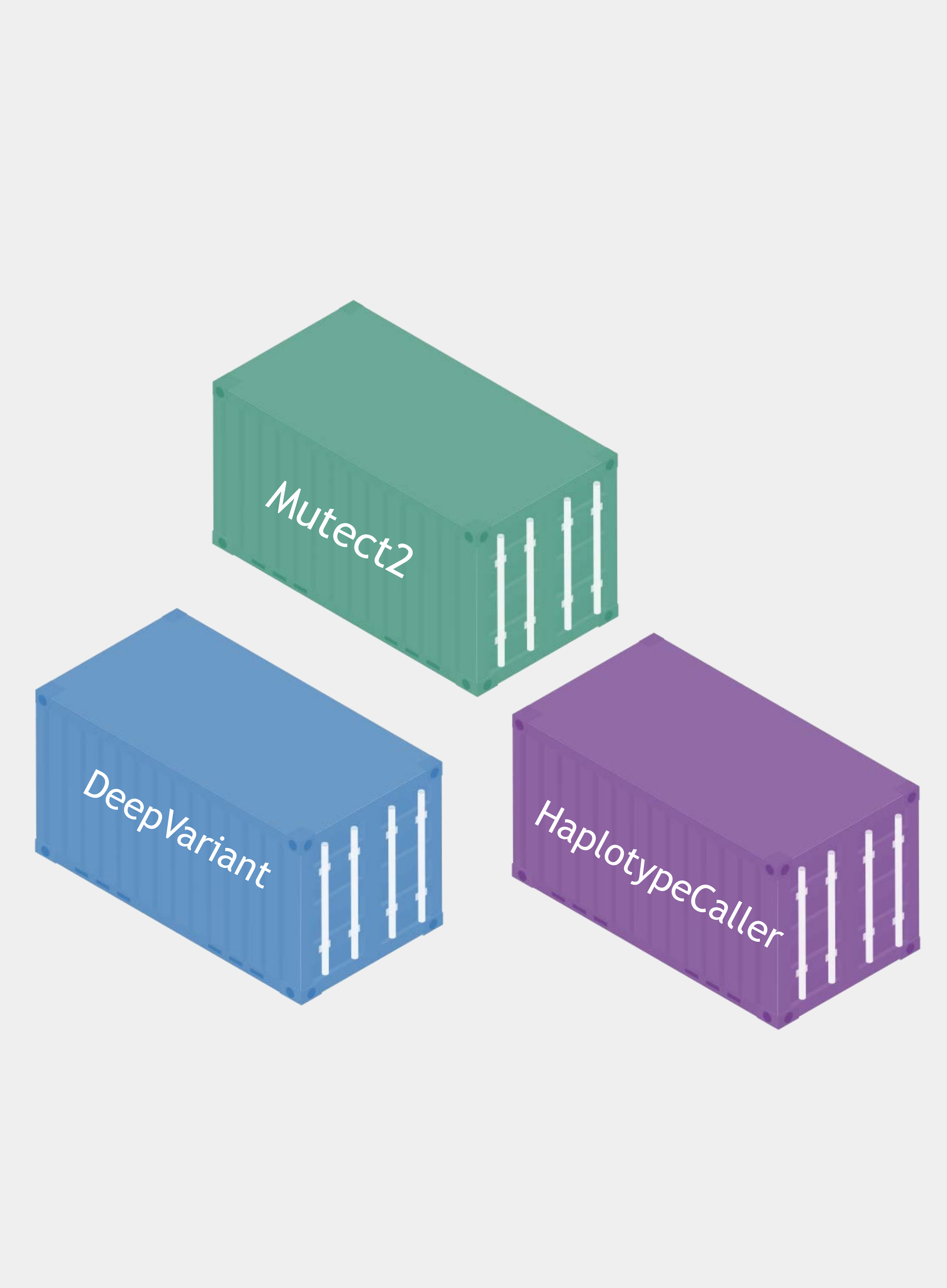
	False Positives	False Negatives
HaplotypeCaller only	20,305	9,321
DeepVariant only	5,101	5,374
Union	22,906	3,893
Intersect	2,500	10,802

HG002 Target genome, analyzing only high confidence regions

Getting Started with NVIDIA Parabricks

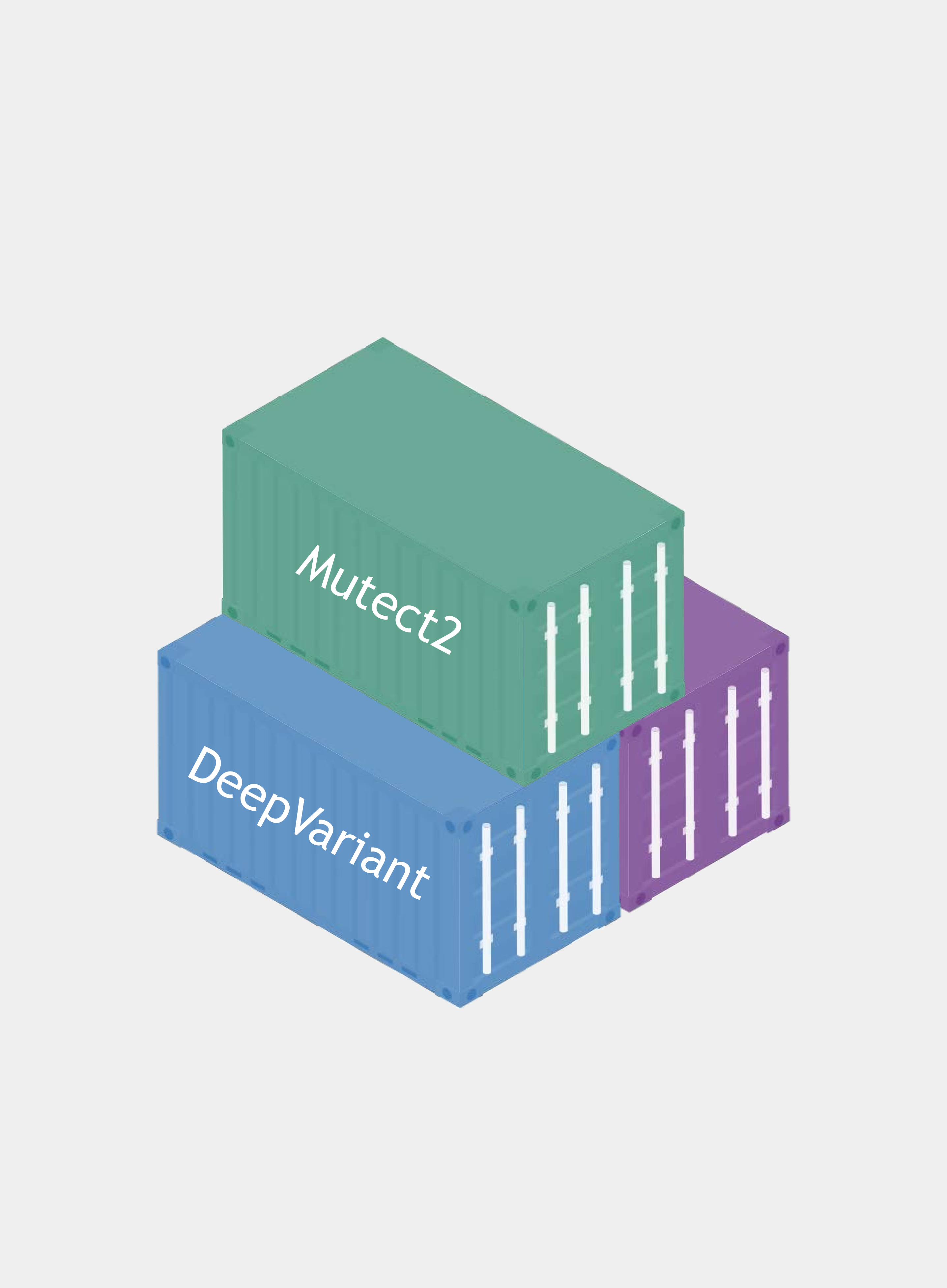
Modular Tools for Flexible Deployment

With NVIDIA AI Enterprise for Production



- Individual Containers for Each Tool
- Agile Releases as and when required
- Lean Deployment on Sequencers or as part of custom workflows

Top of Tree Individual Containers
Available on NGC



- Includes every Parabricks tool
- Regular Releases timed with NVIDIA AI Enterprise
- Option of Purchasing Enterprise Support thru NVIDIA AI Enterprise

Stable Unified Container
Available on NGC
Enterprise Support Available

All NVIDIA Parabricks Containers are available publicly in the NGC Catalog

Workflow Manager Compatible

Customize and deploy Parabricks at scale

GPU-based tool

singularity: parabricks:4.0.0
gpuType : nvidia-tesla-t4
gpuCount : 2

GPU-based tool

singularity: parabricks:4.0.0
gpuType : nvidia-tesla-v100
gpuCount : 4

Tasks deployed on cloud batch service or local cluster (e.g. Minerva)

CPU-based tool

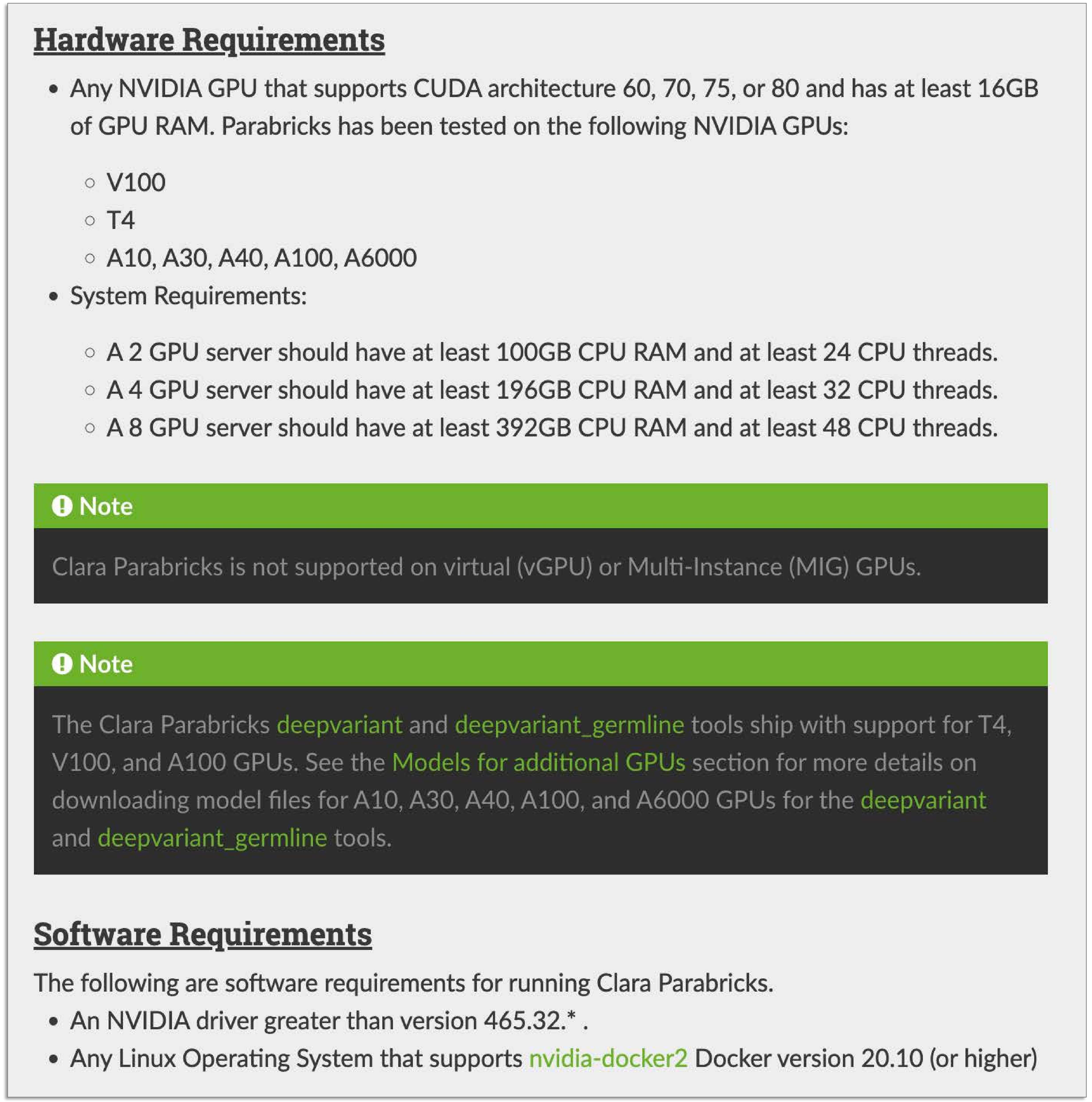
singularity: ubuntu:latest
cpu : 96
memory : 16G

Parabricks is fully compatible with common workflow managers WDL and NextFlow for deploying at scale

- Intertwine GPU and CPU powered tasks with different compute requirements
- Reference workflows and recommended compute configs at: github.com/clara-parabricks-workflows

Running NVIDIA Parabricks

Requirements



Hardware Requirements

- Any NVIDIA GPU that supports CUDA architecture 60, 70, 75, or 80 and has at least 16GB of GPU RAM. Parabricks has been tested on the following NVIDIA GPUs:
  - V100
  - T4
  - A10, A30, A40, A100, A6000
- System Requirements:
  - A 2 GPU server should have at least 100GB CPU RAM and at least 24 CPU threads.
  - A 4 GPU server should have at least 196GB CPU RAM and at least 32 CPU threads.
  - A 8 GPU server should have at least 392GB CPU RAM and at least 48 CPU threads.

! Note

Clara Parabricks is not supported on virtual (vGPU) or Multi-Instance (MIG) GPUs.

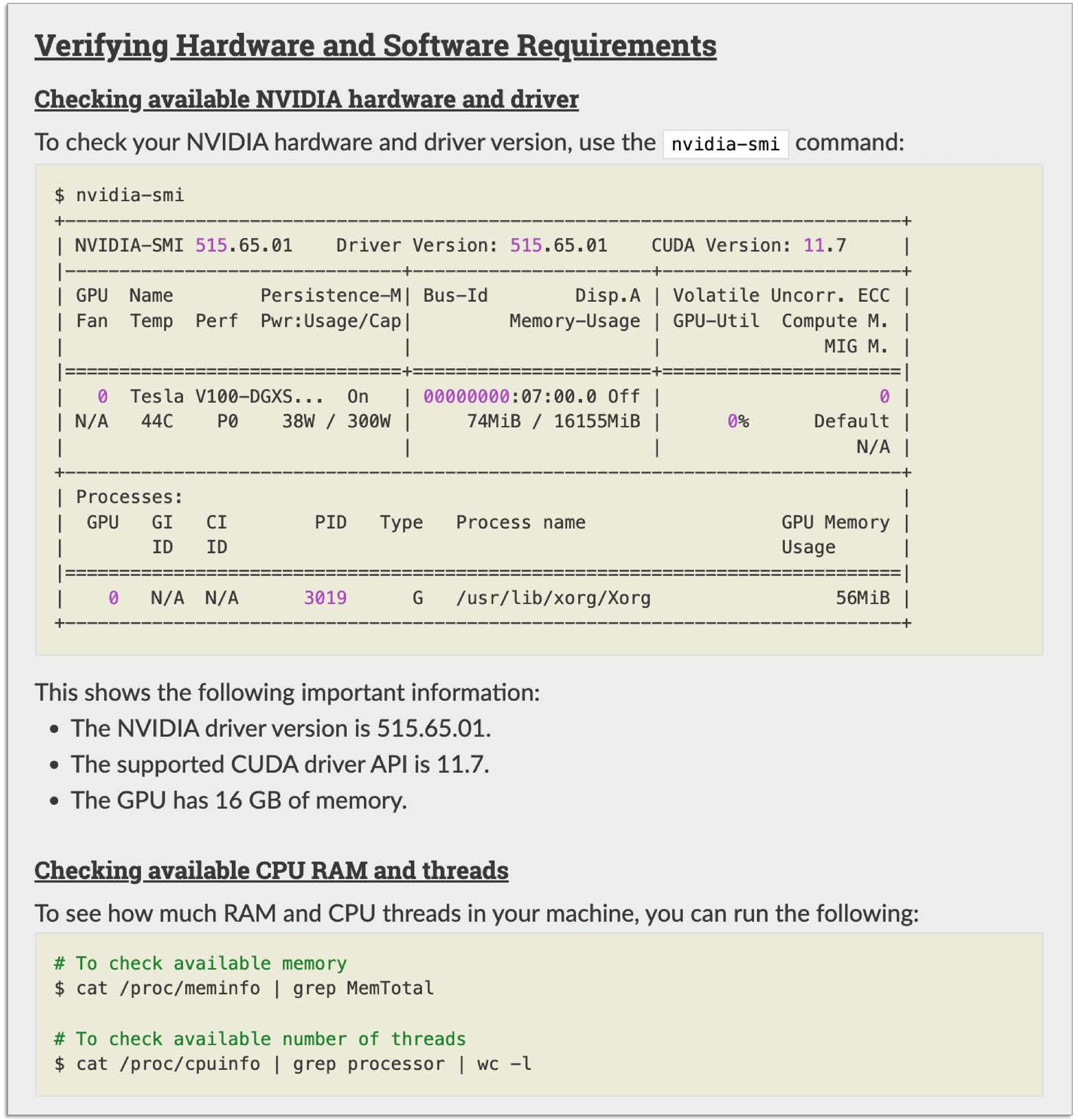
! Note

The Clara Parabricks deepvariant and deepvariant_germline tools ship with support for T4, V100, and A100 GPUs. See the Models for additional GPUs section for more details on downloading model files for A10, A30, A40, A100, and A6000 GPUs for the deepvariant and deepvariant_germline tools.

Software Requirements

The following are software requirements for running Clara Parabricks.

- An NVIDIA driver greater than version 465.32.* .
- Any Linux Operating System that supports nvidia-docker2 Docker version 20.10 (or higher)



Verifying Hardware and Software Requirements

Checking available NVIDIA hardware and driver

To check your NVIDIA hardware and driver version, use the `nvidia-smi` command:

```
$ nvidia-smi
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 515.65.01    Driver Version: 515.65.01    CUDA Version: 11.7     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  Tesla V100-DGXS...  On   | 00000000:07:00.0 Off |                    0 |
| N/A   44C    P0    38W / 300W |     74MiB / 16155MiB |      0%      Default |
|                               |                      |                  N/A |
+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|    0   N/A  N/A      3019      G   /usr/lib/xorg/Xorg                 56MiB |
+-----------------------------------------------------------------------------+
```

This shows the following important information:

- The NVIDIA driver version is 515.65.01.
- The supported CUDA driver API is 11.7.
- The GPU has 16 GB of memory.

Checking available CPU RAM and threads

To see how much RAM and CPU threads in your machine, you can run the following:

```
# To check available memory
$ cat /proc/meminfo | grep MemTotal

# To check available number of threads
$ cat /proc/cpuinfo | grep processor | wc -l
```

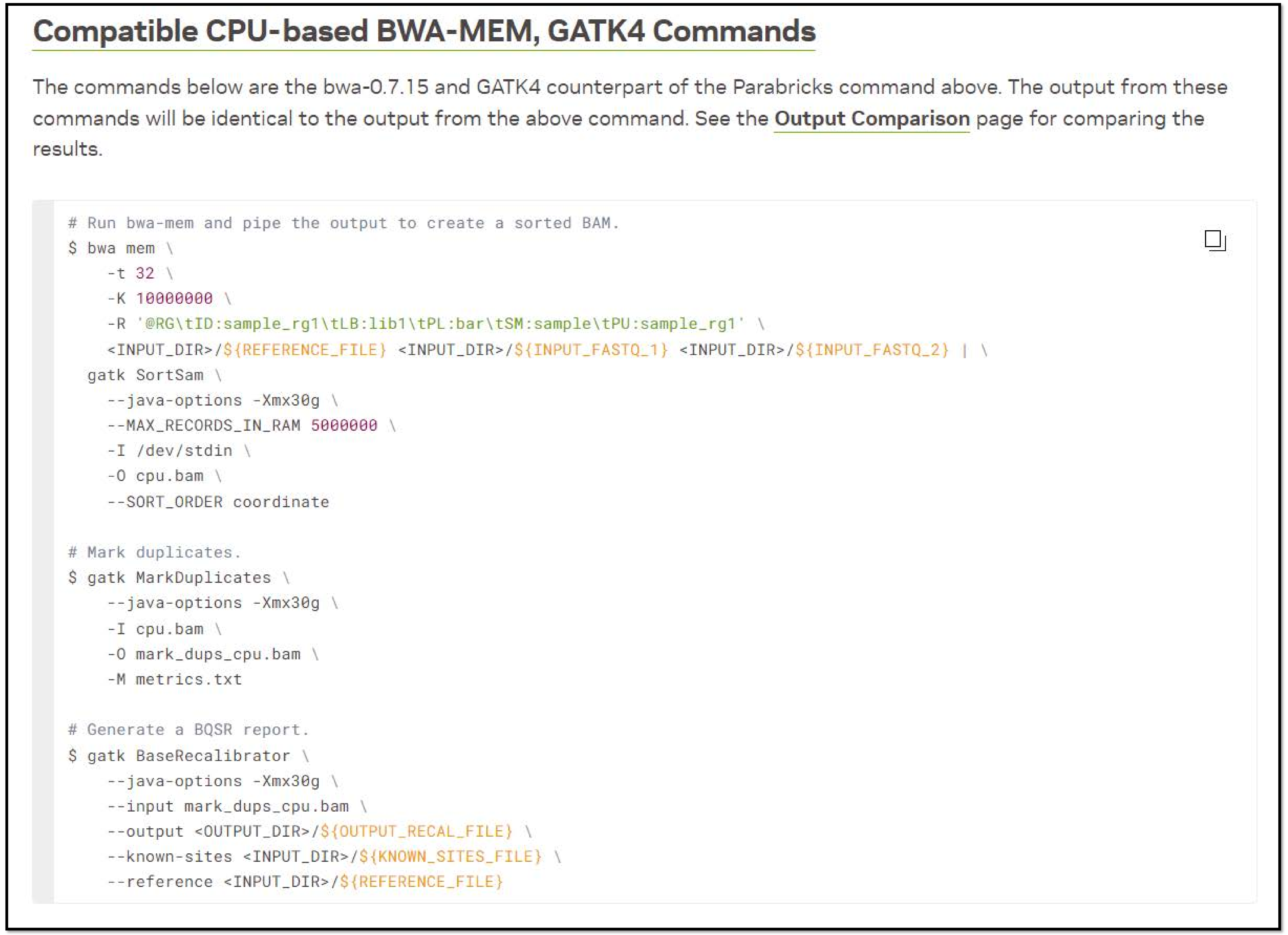
Doc for NVIDIA Parabricks 4.3.0

Running NVIDIA Parabricks

Drop-in Command Line Replacements

Run Parabricks on Minerva

```
singularity exec \
    --nv \
    --bind ${DATA_DIR}:${DATA_DIR} \
    ${SINGULARITY_IMAGE} \
    pbrun fq2bam \
    --ref /workdir/${REFERENCE_FILE} \
    --in-fq /workdir/${INPUT_FASTQ_1} /workdir/${INPUT_FASTQ_2}  \
    --knownSites /workdir/${KNOWN_SITES_FILE} \
    --out-bam /outputdir/${OUTPUT_BAM} \
    --out-recal-file /outputdir/${OUTPUT_RECAL_FILE}
```



Compatible CPU-based BWA-MEM, GATK4 Commands

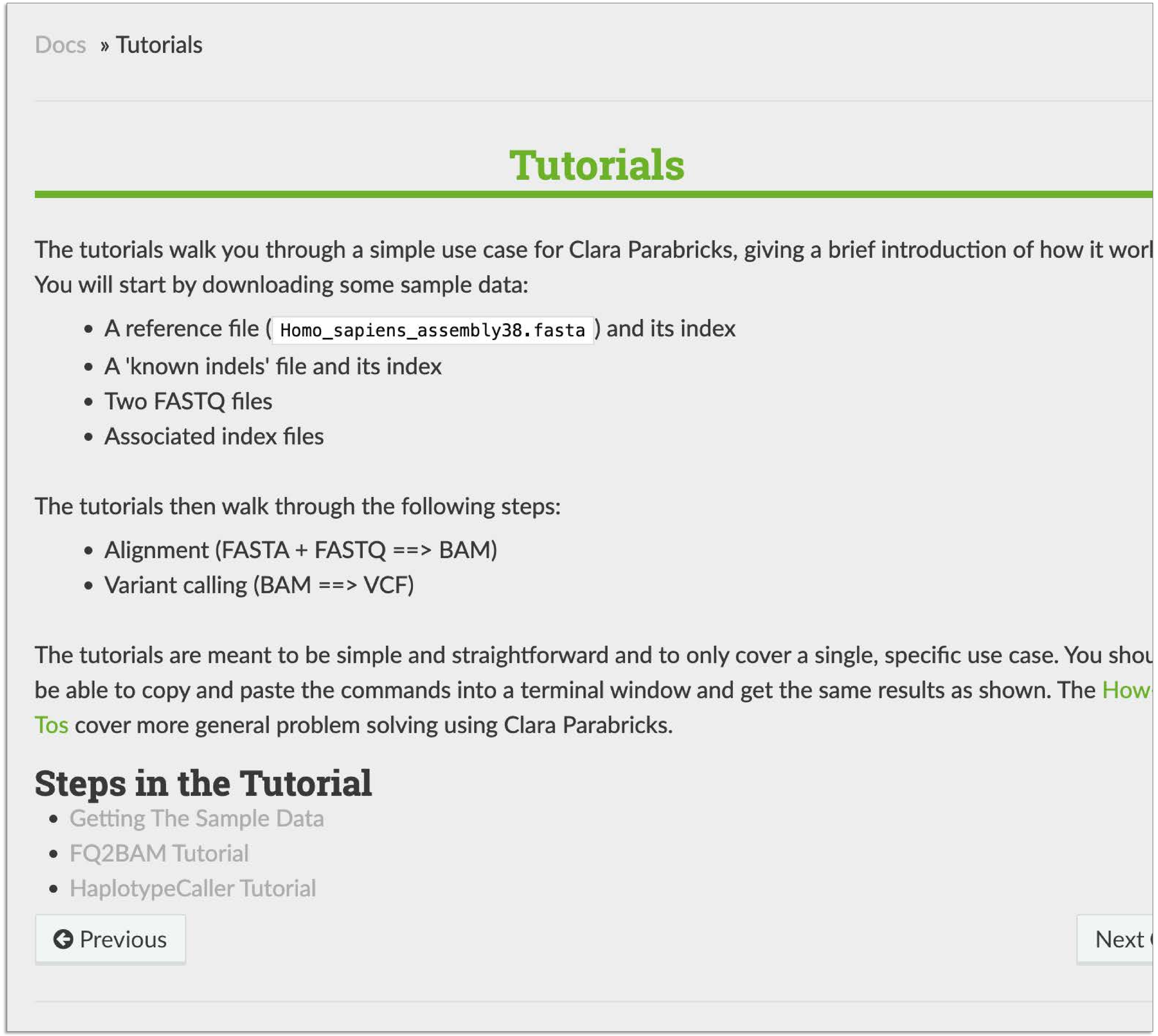
The commands below are the bwa-0.7.15 and GATK4 counterpart of the Parabricks command above. The output from these commands will be identical to the output from the above command. See the **Output Comparison** page for comparing the results.

```
# Run bwa-mem and pipe the output to create a sorted BAM.
$ bwa mem \
    -t 32 \
    -K 10000000 \
    -R '@RG\tID:sample_rg1\tLB:lib1\tPL:bar\tSM:sample\tPU:sample_rg1' \
    <INPUT_DIR>/${REFERENCE_FILE} <INPUT_DIR>/${INPUT_FASTQ_1} <INPUT_DIR>/${INPUT_FASTQ_2} | \
  gatk SortSam \
    --java-options -Xmx30g \
    --MAX_RECORDS_IN_RAM 5000000 \
    -I /dev/stdin \
    -O cpu.bam \
    --SORT_ORDER coordinate

# Mark duplicates.
$ gatk MarkDuplicates \
    --java-options -Xmx30g \
    -I cpu.bam \
    -O mark_dups_cpu.bam \
    -M metrics.txt

# Generate a BQSR report.
$ gatk BaseRecalibrator \
    --java-options -Xmx30g \
    --input mark_dups_cpu.bam \
    --output <OUTPUT_DIR>/${OUTPUT_RECAL_FILE} \
    --known-sites <INPUT_DIR>/${KNOWN_SITES_FILE} \
    --reference <INPUT_DIR>/${REFERENCE_FILE}
```

Doc for NVIDIA Parabricks 4.3.0

Running NVIDIA Parabricks

Demo: run tutorials on Minerva



Docs » Tutorials

Tutorials

The tutorials walk you through a simple use case for Clara Parabricks, giving a brief introduction of how it worl You will start by downloading some sample data:

- A reference file (`Homo_sapiens_assembly38.fasta`) and its index
- A 'known indels' file and its index
- Two FASTQ files
- Associated index files

The tutorials then walk through the following steps:

- Alignment (FASTA + FASTQ ==> BAM)
- Variant calling (BAM ==> VCF)

The tutorials are meant to be simple and straightforward and to only cover a single, specific use case. You shou be able to copy and paste the commands into a terminal window and get the same results as shown. The How Tos cover more general problem solving using Clara Parabricks.

Steps in the Tutorial

- Getting The Sample Data
- FQ2BAM Tutorial
- HaplotypeCaller Tutorial

Doc for NVIDIA Parabricks 4.3.0

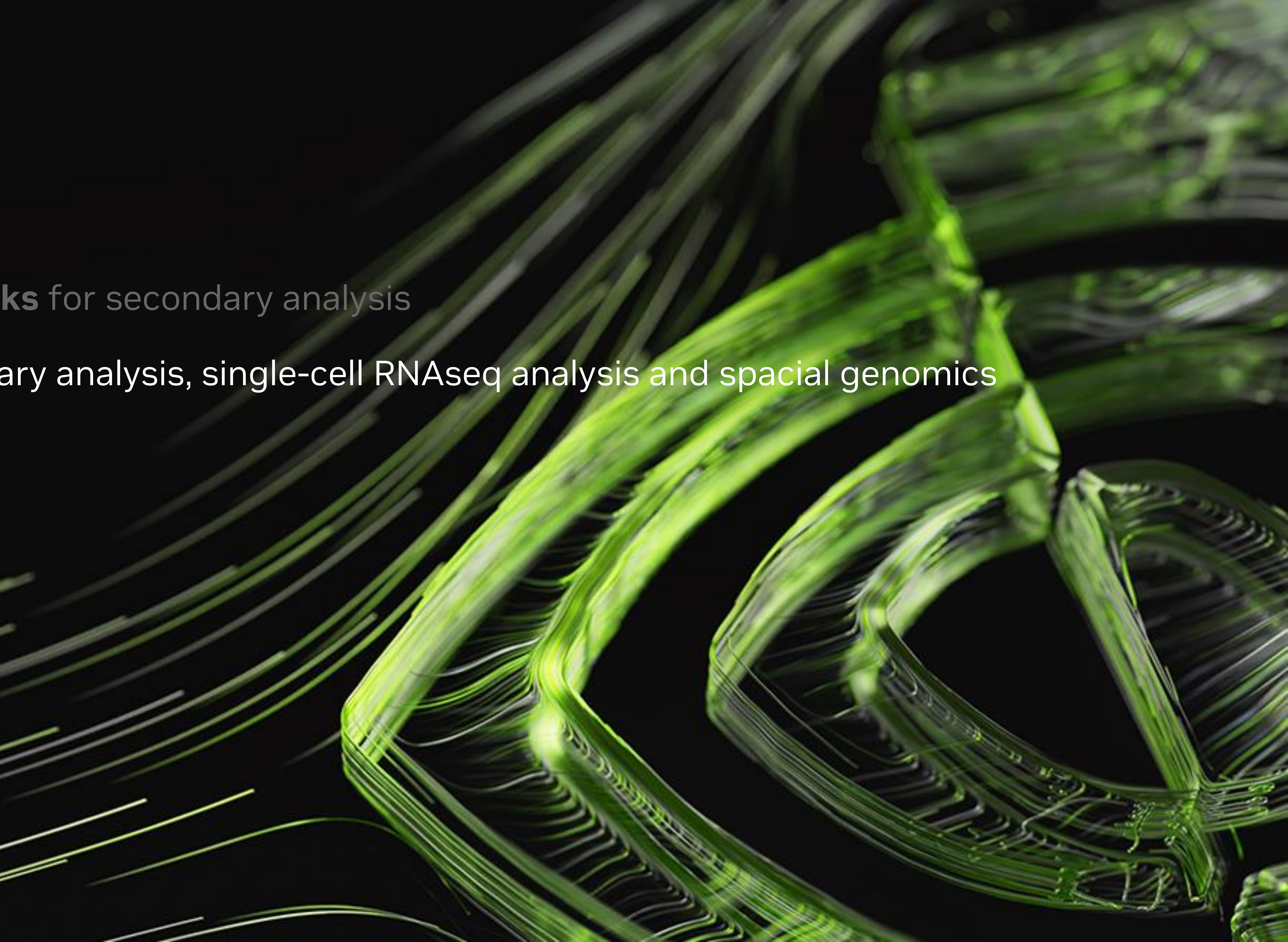
Resources

GTC DLI Workshop – Training DeepVariant Models using Parabricks

- Recording: https://www.nvidia.com/en-us/on-demand/session/gtc24-dlit61813/
  or search for the title on NVIDIA On-Demand

Agenda

- **NVIDIA Parabricks** for secondary analysis
- **RAPIDS** for tertiary analysis, single-cell RNAseq analysis and spacial genomics



NVIDIA is Helping Across the Computational Genomics Workflow

From sequencing sensor to biological insights

TACGTG
TACATG
ATTCATGG
TACATG
ATTCATGG
TACATG
Single Cell
Spatial
TensorRT

DNA Basecalling

Error correction / polishing

Alignment / assembly

Variant calling

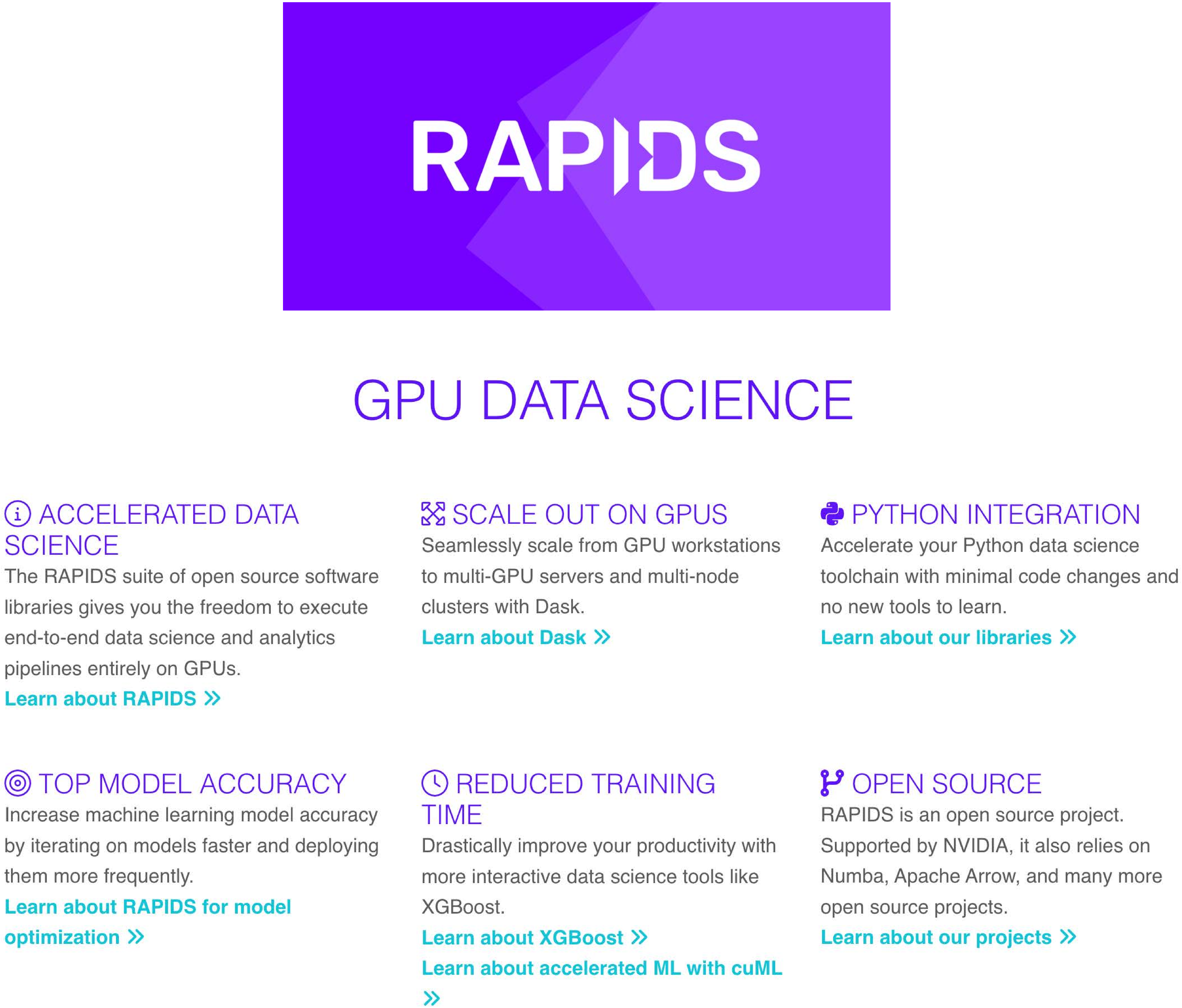
Interpretation / modelling

Primary — Secondary — Tertiary

NVIDIA CUDA

NVIDIA PARABRICKS

RAPIDS



GPU DATA SCIENCE

ACCELERATED DATA SCIENCE

The RAPIDS suite of open source software libraries gives you the freedom to execute end-to-end data science and analytics pipelines entirely on GPUs.

Learn about RAPIDS »

SCALE OUT ON GPUS

Seamlessly scale from GPU workstations to multi-GPU servers and multi-node clusters with Dask.

Learn about Dask »

PYTHON INTEGRATION

Accelerate your Python data science toolchain with minimal code changes and no new tools to learn.

Learn about our libraries »

TOP MODEL ACCURACY

Increase machine learning model accuracy by iterating on models faster and deploying them more frequently.

Learn about RAPIDS for model optimization »

REDUCED TRAINING TIME

Drastically improve your productivity with more interactive data science tools like XGBoost.

Learn about XGBoost »

Learn about accelerated ML with cuML »

OPEN SOURCE

RAPIDS is an open source project. Supported by NVIDIA, it also relies on Numba, Apache Arrow, and many more open source projects.

Learn about our projects »

DATA SCIENCE TOOLSETS

	CPU	GPU/RAPIDS
Data handling	pandas	cuDF
Machine learning	scikit-learn	cuML
Graph analytics	NetworkX	cuGraph

	CPU	GPU/RAPIDS
Viz	Bokeh/ Datashader	cuXfilter
Geospatial	GeoPandas/ SciPy.spatial	cuSpatial
Signals	SciPy.signal	cuSignal
Cyber	cyberpandas	CLX

RAPIDS PLATFORM

Dask

cuDF
Data Prep/Handling

cuML
Machine Learning

cuGraph
Graph Analytics

DL Frameworks
Deep Learning

cuXfilter
Visualization

GPU Memory

Apache Arrow

Specialized package examples

cuSpatial
Geospatial Analytics

cuSignal
Signal Processing

CLX
Cyber Analytics

SINGLE-CELL RNA-SEQ ANALYSIS USING RAPIDS

Cell x Gene matrix

Gene 1……. Gene n

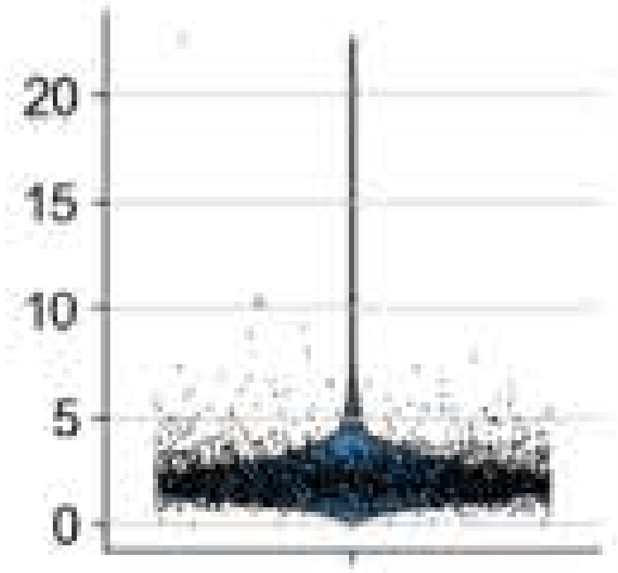
Cell 1 ….. Cell n

Data processing

Filter

Linear Regression

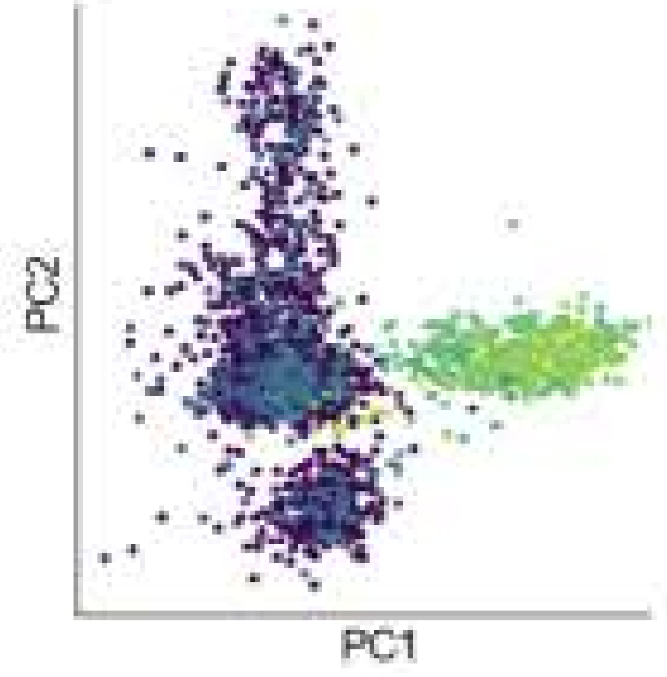
Normalize



cuDF
cuML

Dimensionality Reduction

PCA

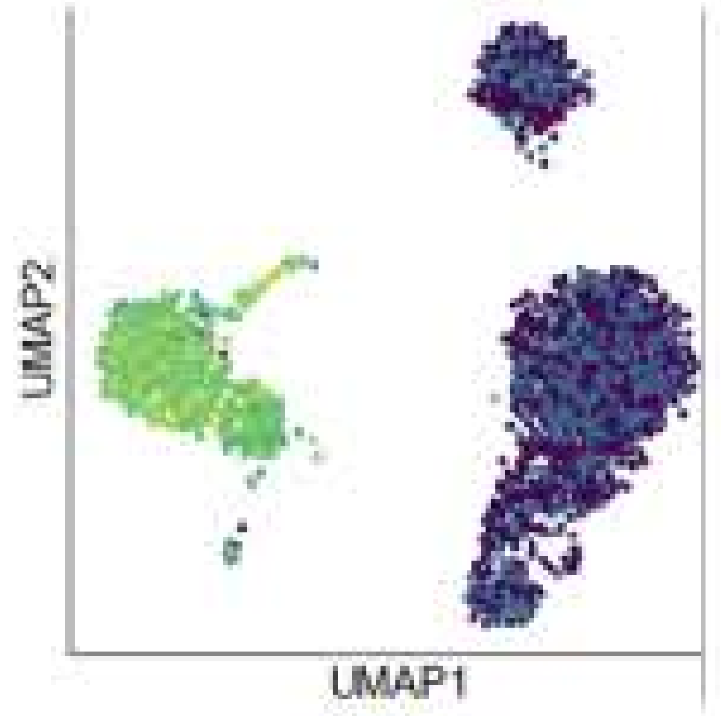


cuML

Visualization

UMAP

t-SNE



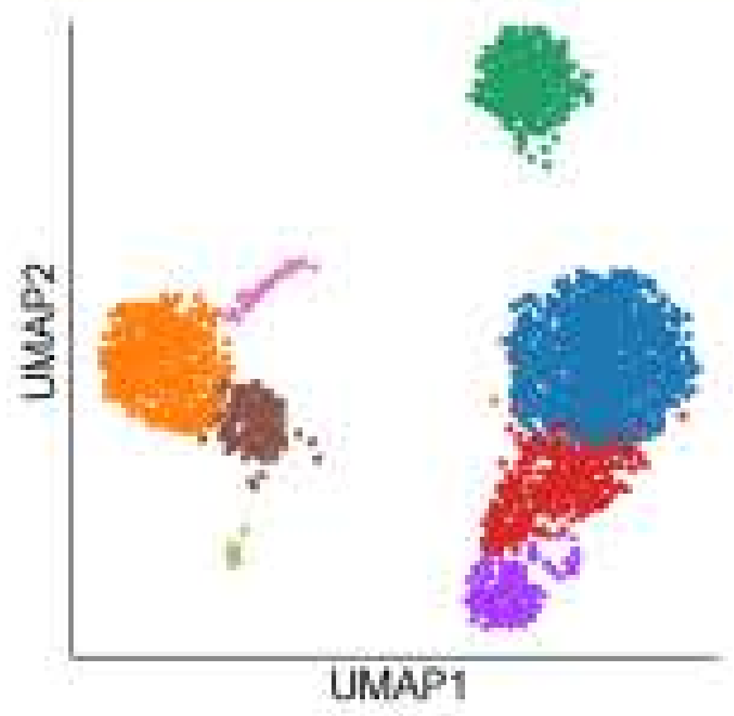
cuML
cuGraph

Clustering

k-Means

Louvain

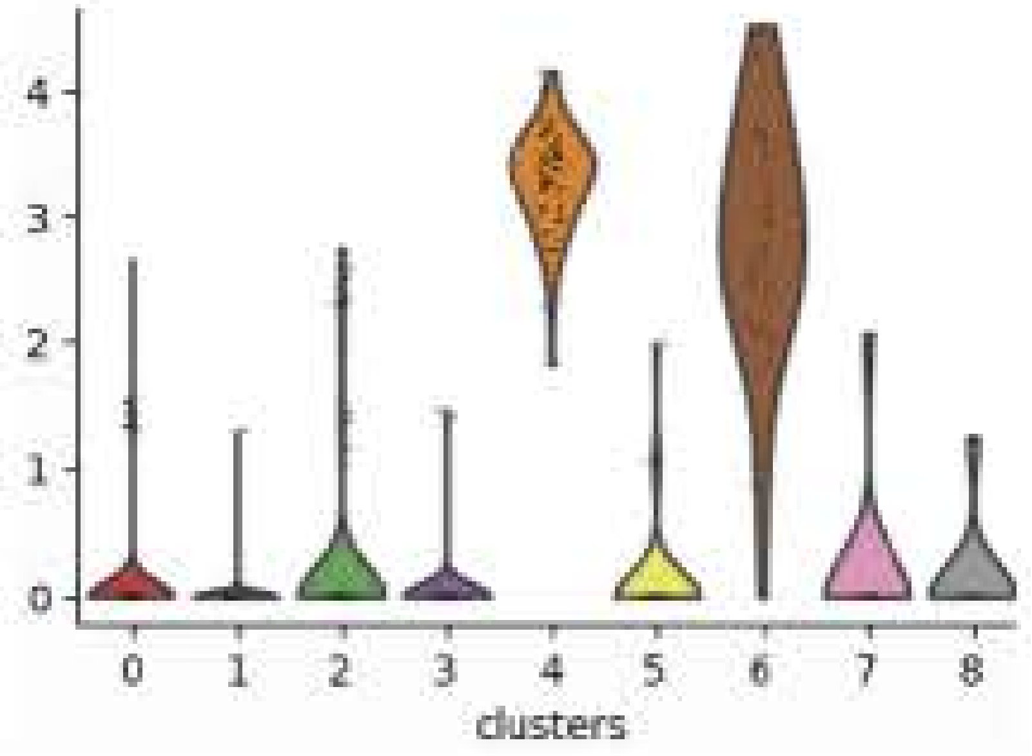
Leiden



cuML

Differential Gene Activity

Logistic Regression



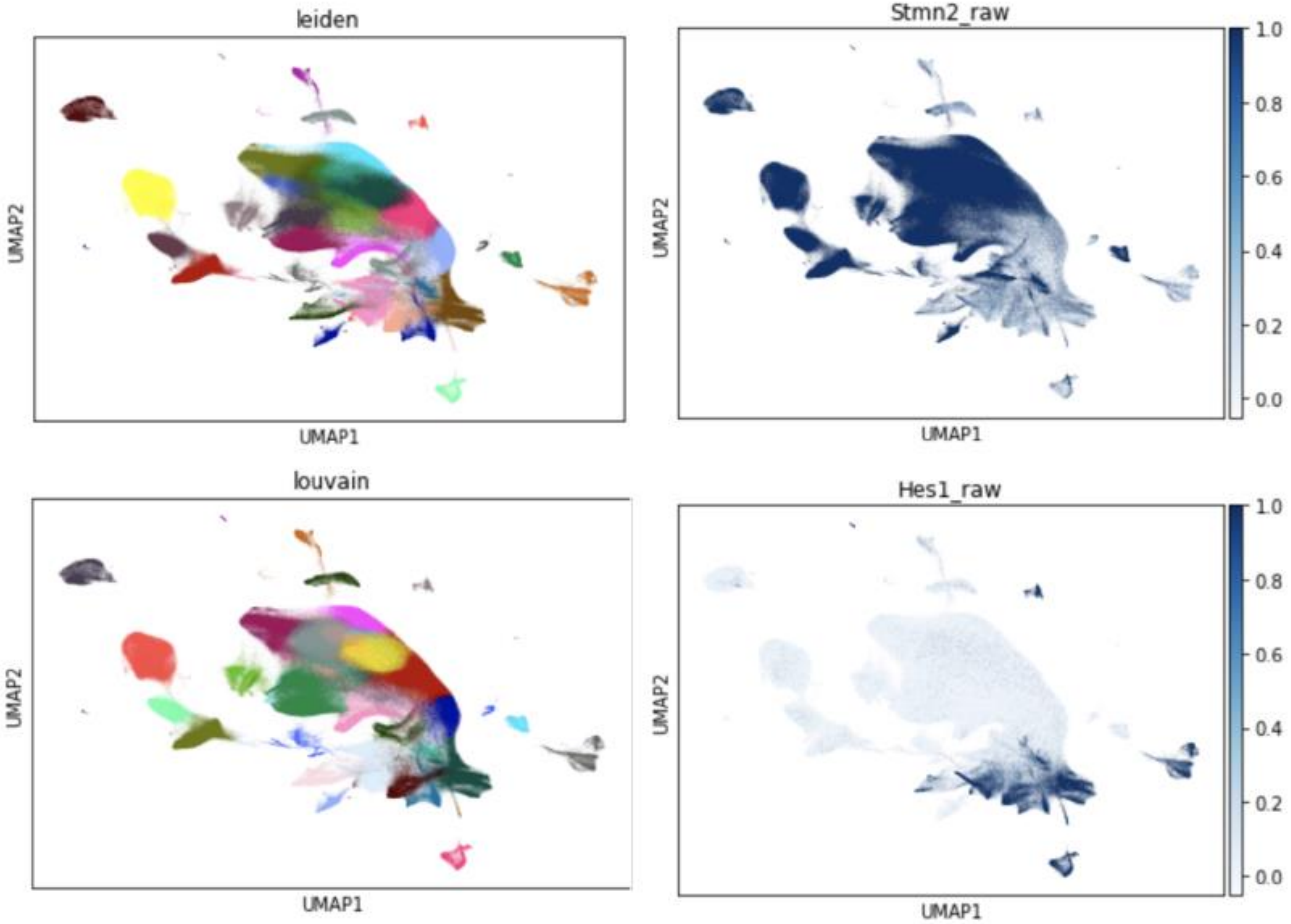
cuML

Iterate

GPU ANALYSIS OF 1 MILLION CELLS

From 3.5 hours to 8 minutes

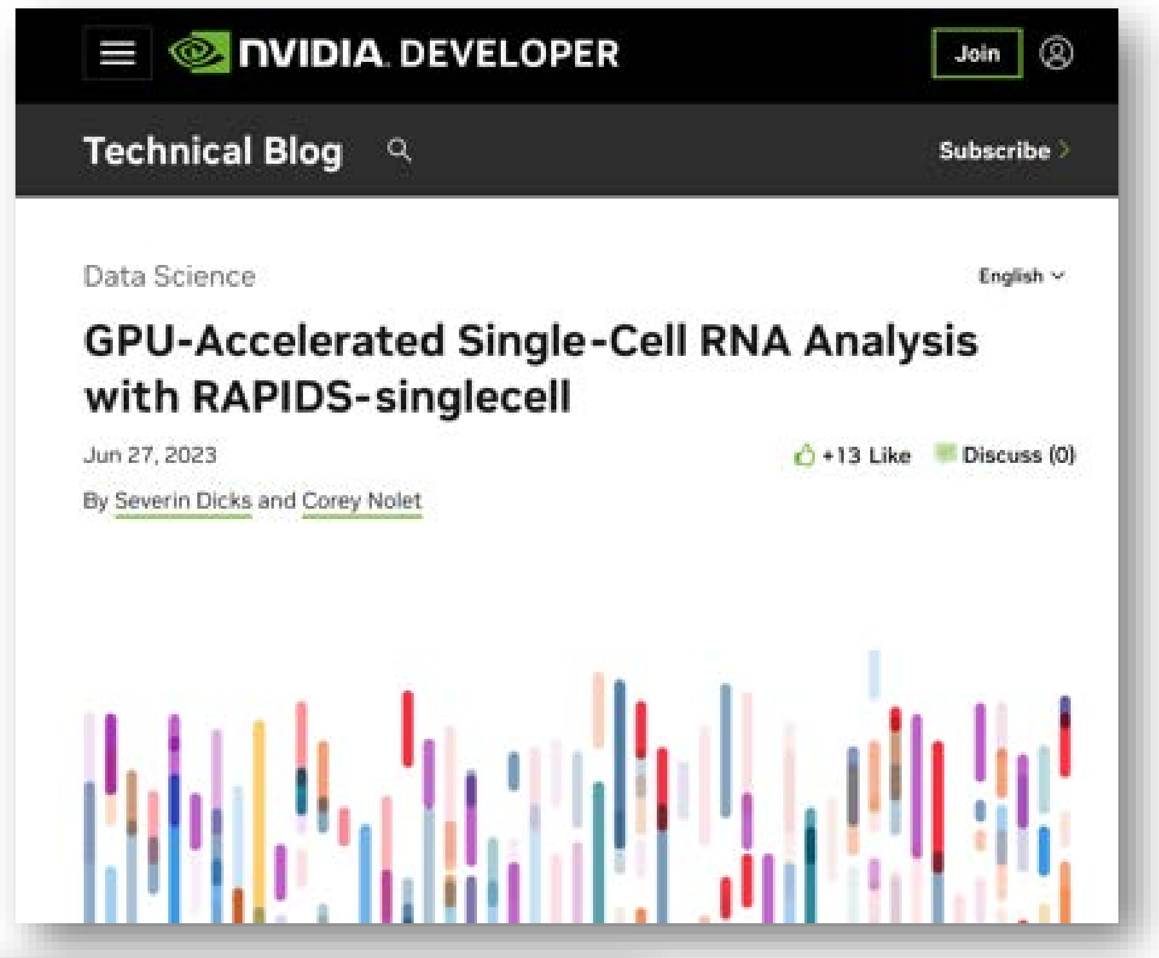
	CPU Runtime n1-highmem-32 32 vCPUs	GPU runtime a2-highgpu-1g Tesla A100 40GB GPU	GPU acceleration
Preprocessing	28m35s	3m21s	9x
PCA	29.2s	11.4s	2.6x
t-SNE	1hr23m10s	28s	178x
KNN	3m5s	46s	4x
UMAP	21m47s	13.4s	98x
k-means clustering	2m6s	1.9s	66x
Louvain clustering	15m5s	1.9s	476x
Leiden clustering	51m1s	1.4s	2186x
End-to-end runtime	3hr31m48s	8m22s	25x
End-to-end cost	$6.682	$0.553	



Repository for example jupyter notebooks: https://github.com/NVIDIA-Genomics-Research/rapids-single-cell-examples

RAPIDS-SingleCell

Part of the Scverse ecosystem



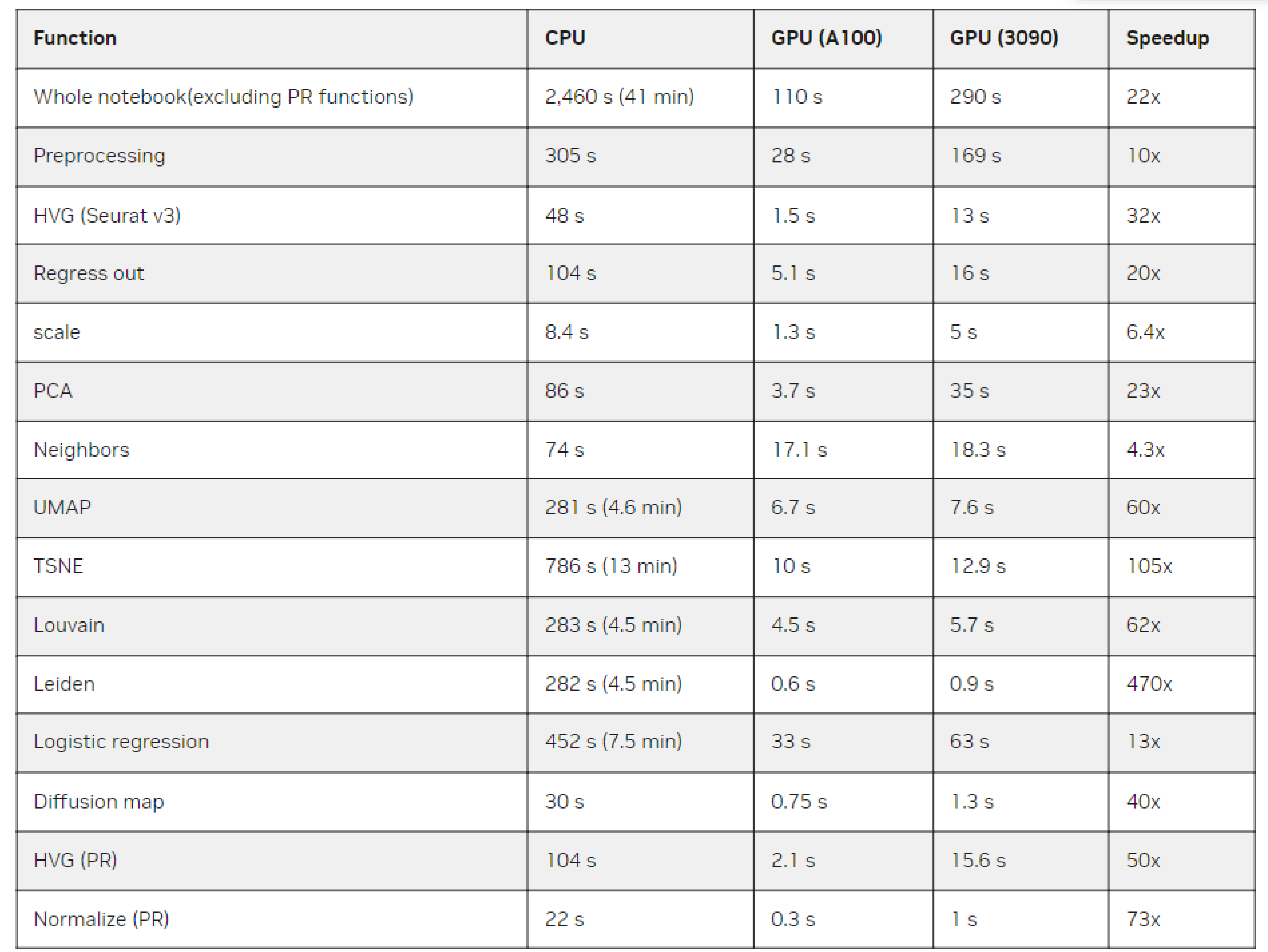
Data Science

GPU-Accelerated Single-Cell RNA Analysis with RAPIDS-singlecell

Jun 27, 2023

By Severin Dicks and Corey Nolet

- A new library drawing inspiration from the rapids-single-cell-examples library and the ScanPy library
- Introduces GPU-optimized versions of the ScanPy (single cell) and SquidPy (spatial) functions
- The library's primary objective is to blend the computational strength of GPUs with the user-friendly nature of the scverse ecosystem.



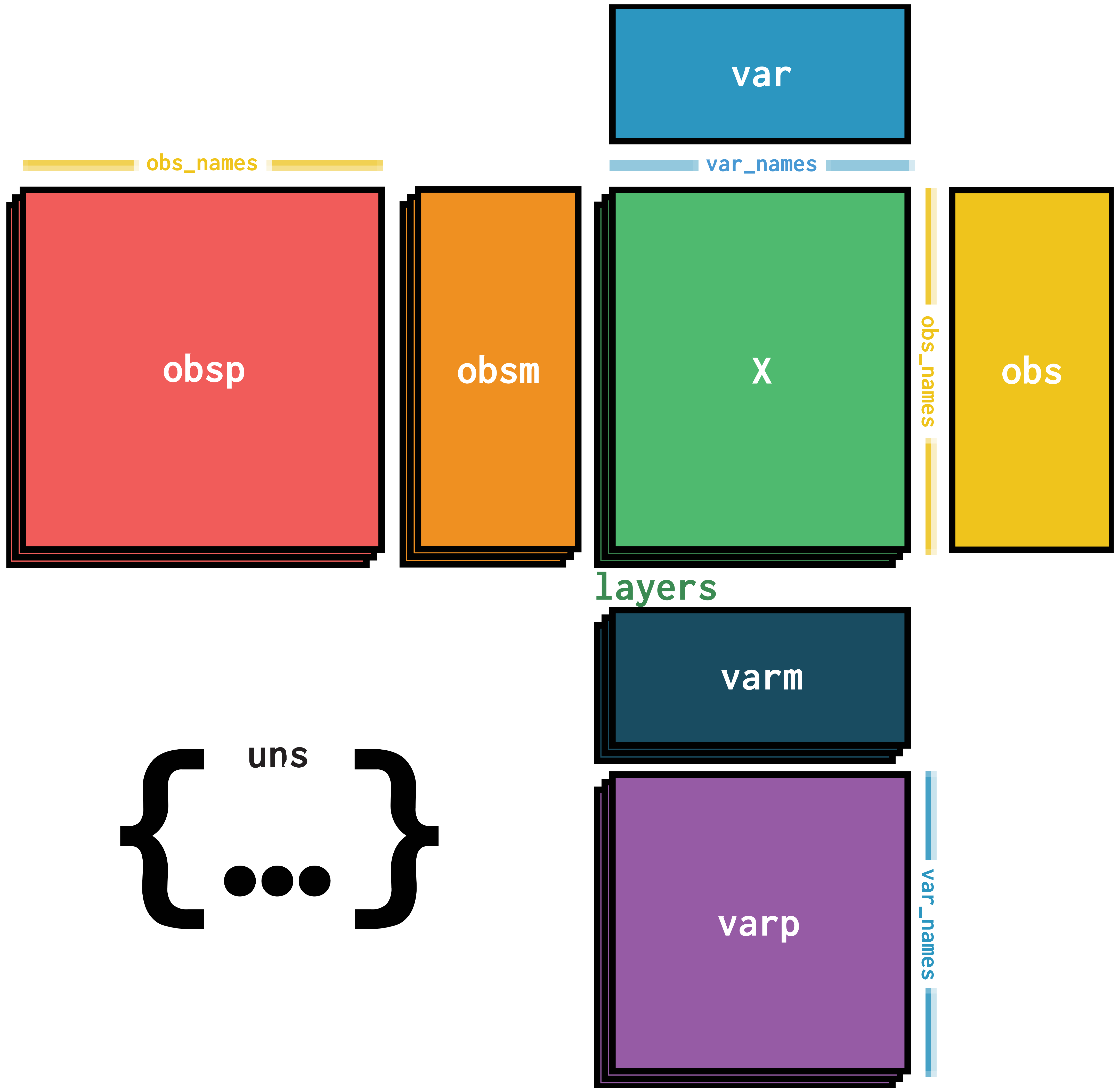
Function	CPU	GPU (A100)	GPU (3090)	Speedup
Whole notebook(excluding PR functions)	2,460 s (41 min)	110 s	290 s	22x
Preprocessing	305 s	28 s	169 s	10x
HVG (Seurat v3)	48 s	1.5 s	13 s	32x
Regress out	104 s	5.1 s	16 s	20x
scale	8.4 s	1.3 s	5 s	6.4x
PCA	86 s	3.7 s	35 s	23x
Neighbors	74 s	17.1 s	18.3 s	4.3x
UMAP	281 s (4.6 min)	6.7 s	7.6 s	60x
TSNE	786 s (13 min)	10 s	12.9 s	105x
Louvain	283 s (4.5 min)	4.5 s	5.7 s	62x
Leiden	282 s (4.5 min)	0.6 s	0.9 s	470x
Logistic regression	452 s (7.5 min)	33 s	63 s	13x
Diffusion map	30 s	0.75 s	1.3 s	40x
HVG (PR)	104 s	2.1 s	15.6 s	50x
Normalize (PR)	22 s	0.3 s	1 s	73x

Table 2. Server node and consumer system benchmark for a dataset of 500,000 cells

https://developer.nvidia.com/blog/gpu-accelerated-single-cell-rna-analysis-with-rapids-singlecell/

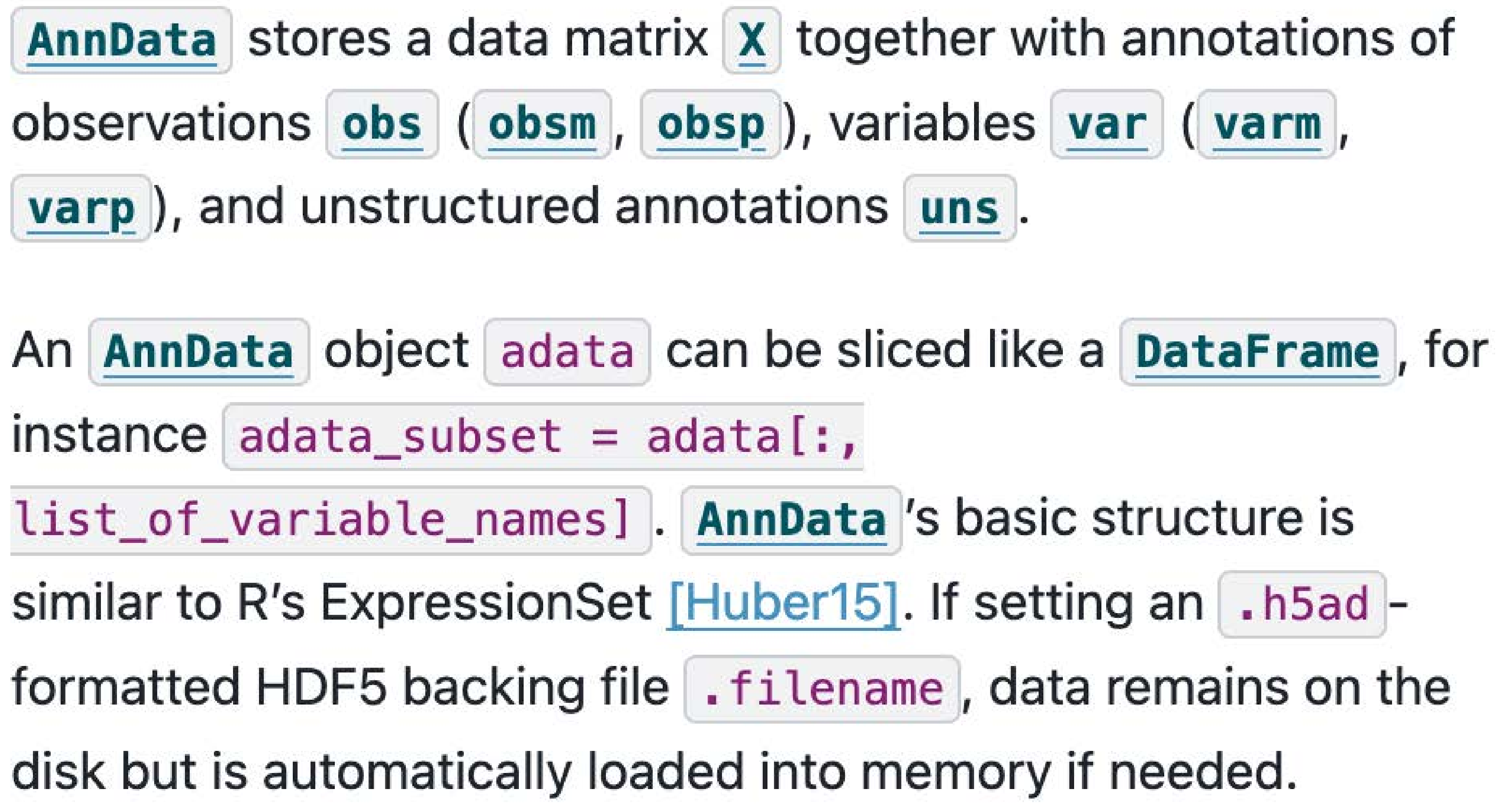
RAPIDS-SingleCell

The AnnData framework now supports CuPy arrays



https://anndata.readthedocs.io/en/latest/#

Rapids-singlecell utilizes the scverse AnnData data framework, which supports dense and sparse CuPy arrays

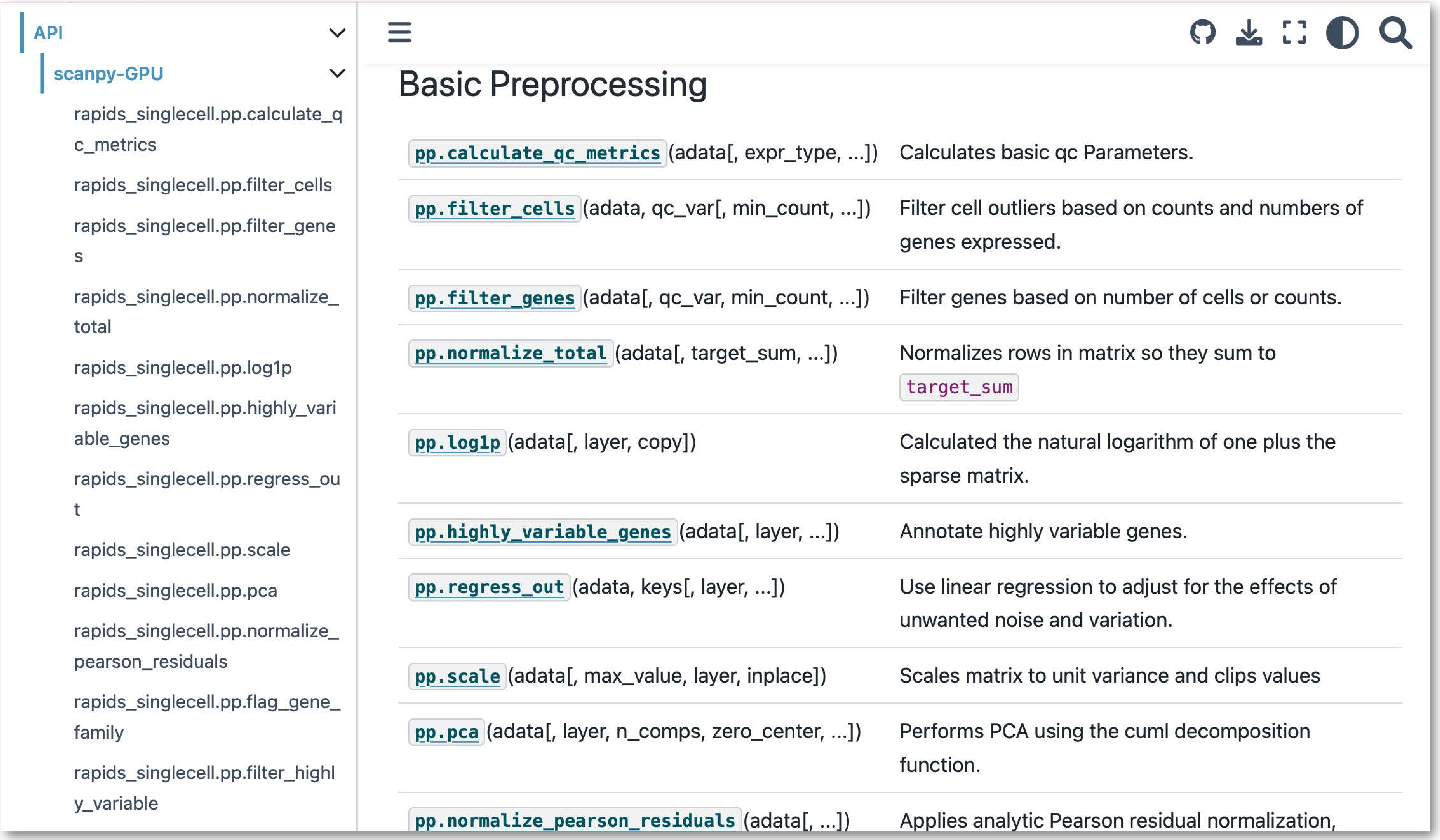


`AnnData` stores a data matrix `X` together with annotations of observations `obs` (`obsm`, `obsp`), variables `var` (`varm`, `varp`), and unstructured annotations `uns`.

An `AnnData` object `adata` can be sliced like a `DataFrame`, for instance `adata_subset = adata[:, list_of_variable_names]`. `AnnData`'s basic structure is similar to R's ExpressionSet [Huber15]. If setting an `.h5ad`-formatted HDF5 backing file `.filename`, data remains on the disk but is automatically loaded into memory if needed.

RAPIDS-SingleCell

API based on ScanPy and SquidPy



API

scanpy-GPU

- rapids_singlecell.pp.calculate_qc_metrics
- rapids_singlecell.pp.filter_cells
- rapids_singlecell.pp.filter_genes
- rapids_singlecell.pp.normalize_total
- rapids_singlecell.pp.log1p
- rapids_singlecell.pp.highly_variable_genes
- rapids_singlecell.pp.regress_out
- rapids_singlecell.pp.scale
- rapids_singlecell.pp.pca
- rapids_singlecell.pp.normalize_pearson_residuals
- rapids_singlecell.pp.flag_gene_family
- rapids_singlecell.pp.filter_highly_variable

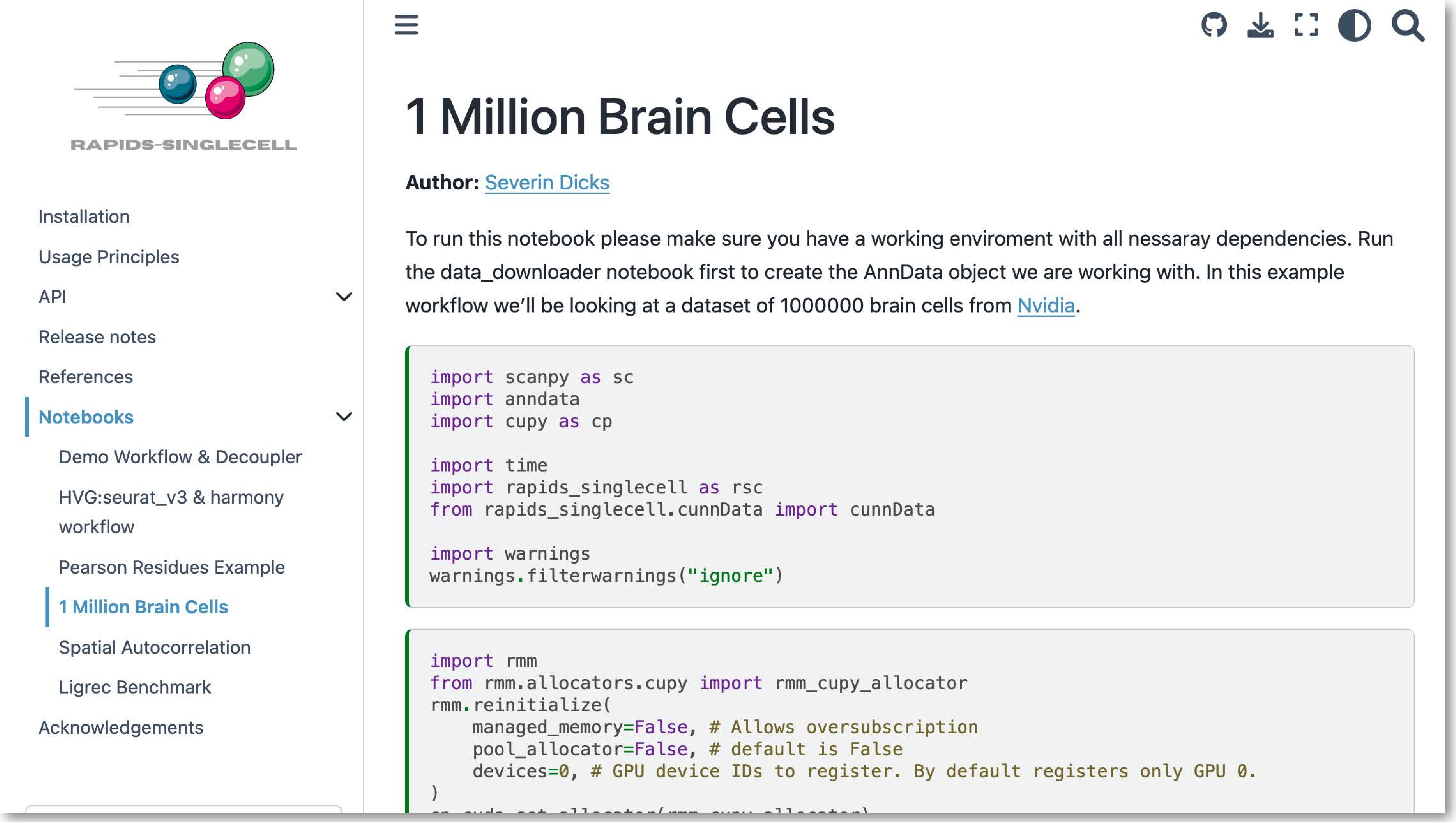
Basic Preprocessing

`pp.calculate_qc_metrics`(adata[, expr_type, ...])	Calculates basic qc Parameters.
`pp.filter_cells`(adata, qc_var[, min_count, ...])	Filter cell outliers based on counts and numbers of genes expressed.
`pp.filter_genes`(adata[, qc_var, min_count, ...])	Filter genes based on number of cells or counts.
`pp.normalize_total`(adata[, target_sum, ...])	Normalizes rows in matrix so they sum to `target_sum`
`pp.log1p`(adata[, layer, copy])	Calculated the natural logarithm of one plus the sparse matrix.
`pp.highly_variable_genes`(adata[, layer, ...])	Annotate highly variable genes.
`pp.regress_out`(adata, keys[, layer, ...])	Use linear regression to adjust for the effects of unwanted noise and variation.
`pp.scale`(adata[, max_value, layer, inplace])	Scales matrix to unit variance and clips values
`pp.pca`(adata[, layer, n_comps, zero_center, ...])	Performs PCA using the cuml decomposition function.
`pp.normalize_pearson_residuals`(adata[, ...])	Applies analytic Pearson residual normalization,

https://rapids-singlecell.readthedocs.io/en/latest/index.html

RAPIDS-SingleCell

Get Started Now with Example Notebooks



1 Million Brain Cells

Author: Severin Dicks

To run this notebook please make sure you have a working enviroment with all nessaray dependencies. Run the data_downloader notebook first to create the AnnData object we are working with. In this example workflow we'll be looking at a dataset of 1000000 brain cells from Nvidia.

```
import scanpy as sc
import anndata
import cupy as cp

import time
import rapids_singlecell as rsc
from rapids_singlecell.cunnData import cunnData

import warnings
warnings.filterwarnings("ignore")
```

```
import rmm
from rmm.allocators.cupy import rmm_cupy_allocator
rmm.reinitialize(
    managed_memory=False, # Allows oversubscription
    pool_allocator=False, # default is False
    devices=0, # GPU device IDs to register. By default registers only GPU 0.
)
cp.cuda.set_allocator(rmm_cupy_allocator)
```

https://rapids-singlecell.readthedocs.io/en/latest/notebooks/demo_gpu-seuratv3-brain-1M.html

Spatial Genomics

GTC DLI Workshop - Everything, All at Once: Processing Spatial Transcriptomics Data using Accelerated Computing
- Recording: https://www.nvidia.com/en-us/on-demand/session/gtc24-dlit61337/
  or search for the title on NVIDIA On-Demand

Everything, All at Once: Processing Spatial Transcriptomics Data using Accelerated Computing

Share
Favorite
Add to list
Rate Now

Jonny Hancox, Senior Solution Architect, NVIDIA

The potential clinical benefits of combining the power of mRNA expression with the rich context of tissue morphology are considerable. Equally impressive is the amount of data preparation and processing needed to perform this analysis. In the absence of well-established pipelines dedicated to this analysis, scientists often have to make do with their own curated set of tools and techniques. Learn how RAPIDS and compatible tools can be used to address the key steps in the process, reducing the time to turn biological mysteries into actionable insights.

You'll learn to use GPU tools in python environment - RAPIDS, CuPy, Numba

MONAI Core

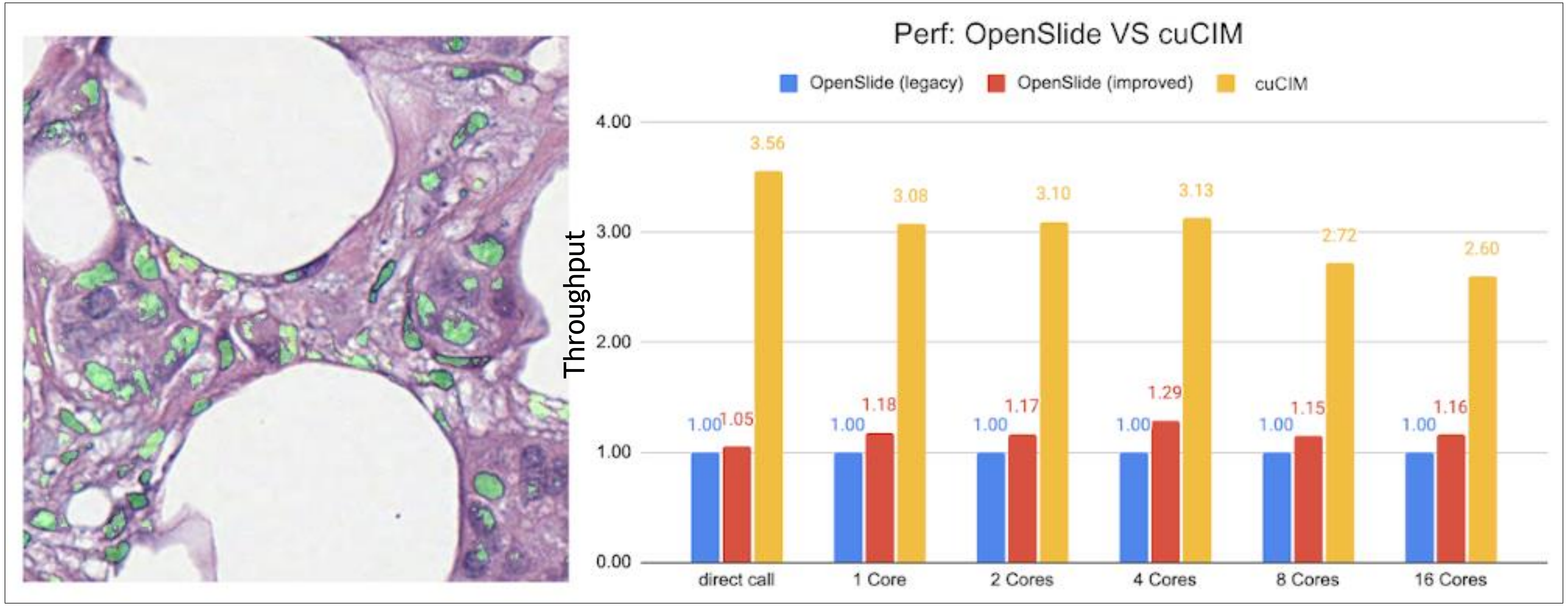
1. Medical imaging specifi

2. Superior performance

3. Friendly community

Optimize data loading

cuCIM – Whole Slide Imaging (digital pathology)



cuCIM - a library within RAPIDS

MONAI Core

1. Medical imaging specif

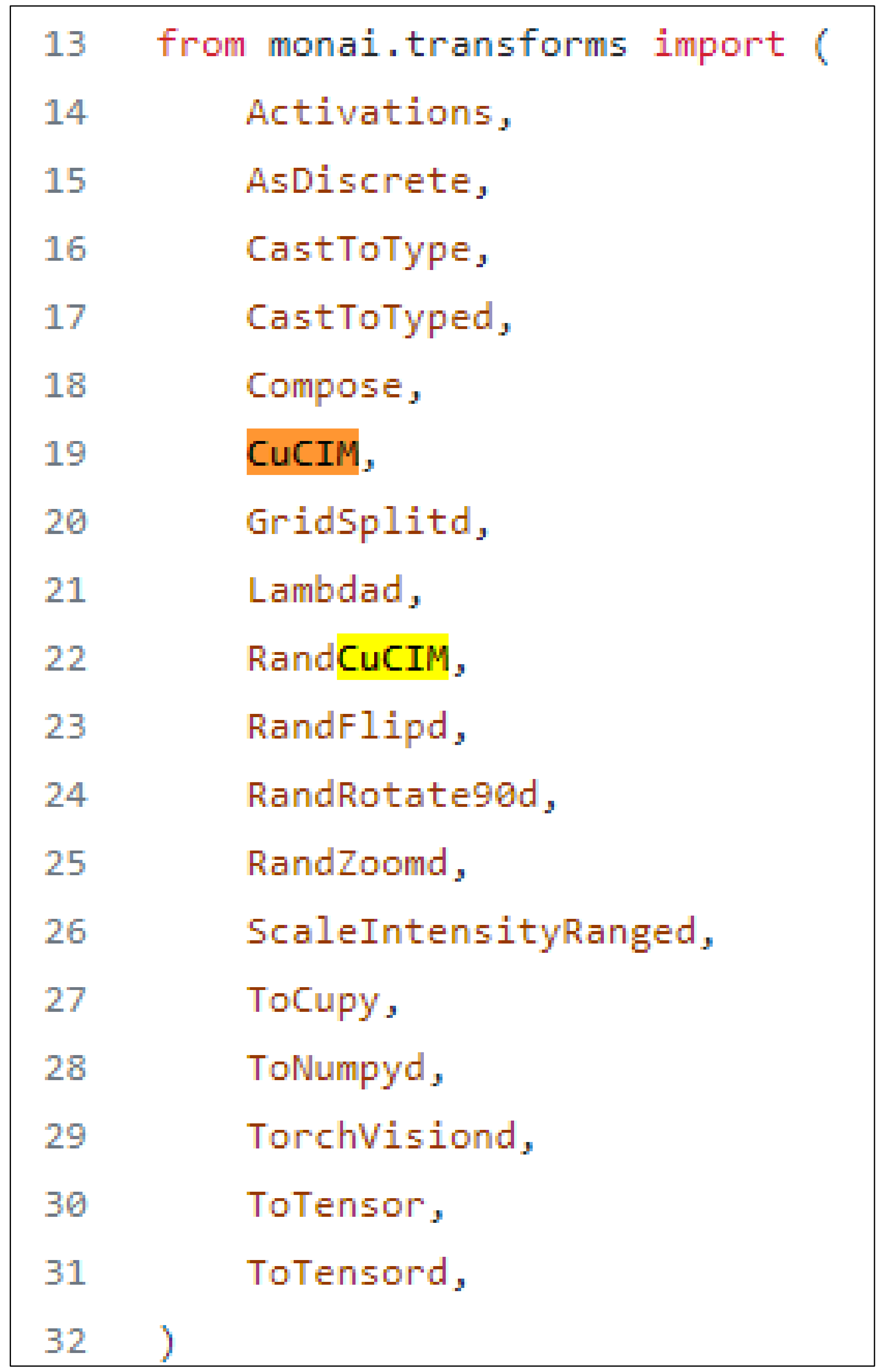
2. Superior performance

3. Friendly community

Optimize GPU utilization

Do transforms on GPU

cuCIM -> common transforms in digital pathology



```
from monai.transforms import (
    Activations,
    AsDiscrete,
    CastToType,
    CastToTyped,
    Compose,
    CuCIM,
    GridSplitd,
    Lambdad,
    RandCuCIM,
    RandFlipd,
    RandRotate90d,
    RandZoomd,
    ScaleIntensityRanged,
    ToCupy,
    ToNumpyd,
    TorchVisiond,
    ToTensor,
    ToTensord,
)
```

MONAI Core pathology tutorials

Agenda

- **NVIDIA Parabricks** for secondary analysis
- **RAPIDS** for tertiary analysis, single-cell RNAseq analysis and spacial genomics

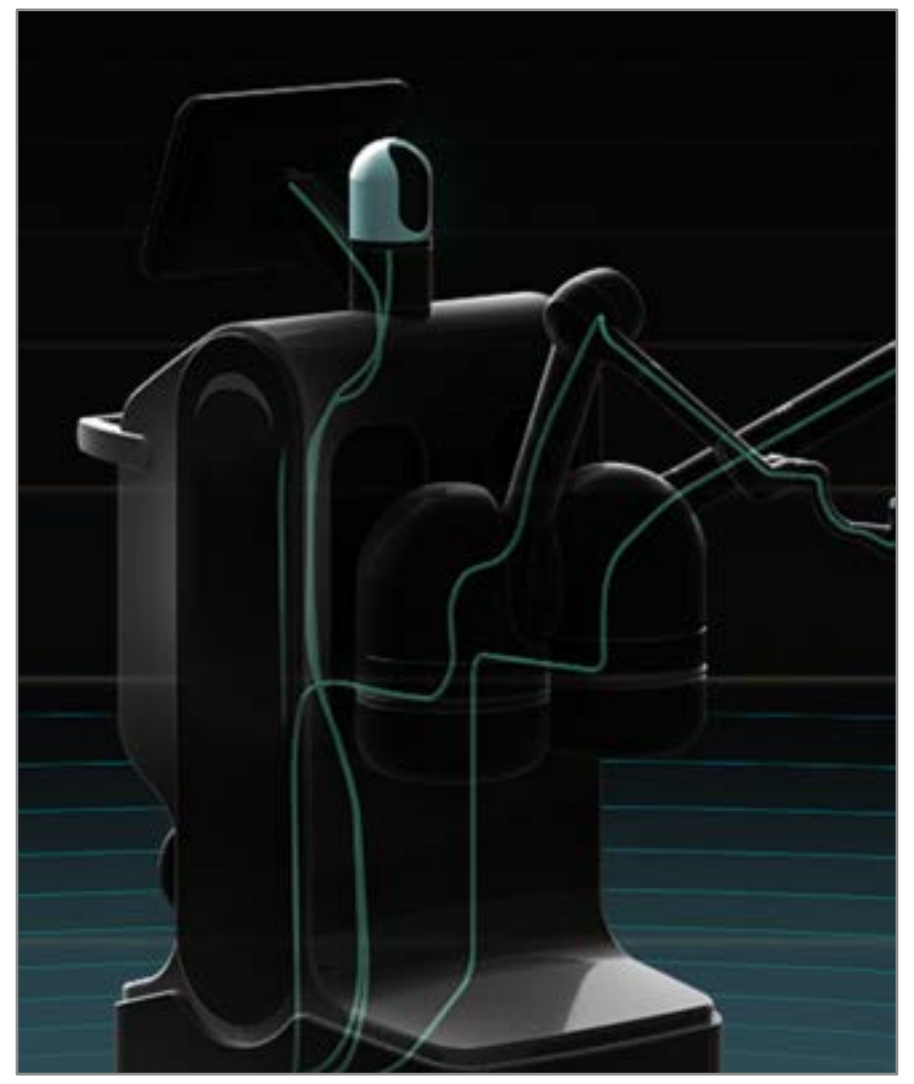
NVIDIA Clara

Accelerated Computing Platform for Healthcare & Life Sciences

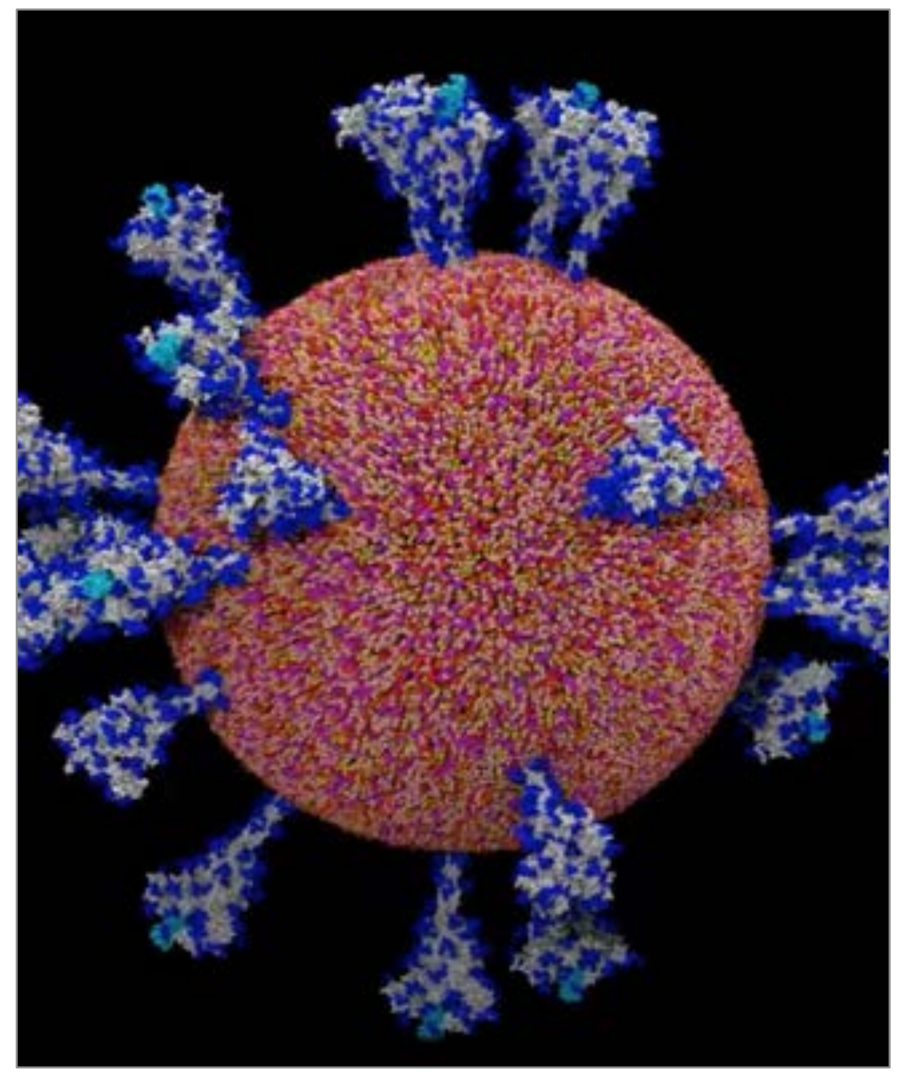
NVIDIA CLARA

PARABRICKS
Genomics

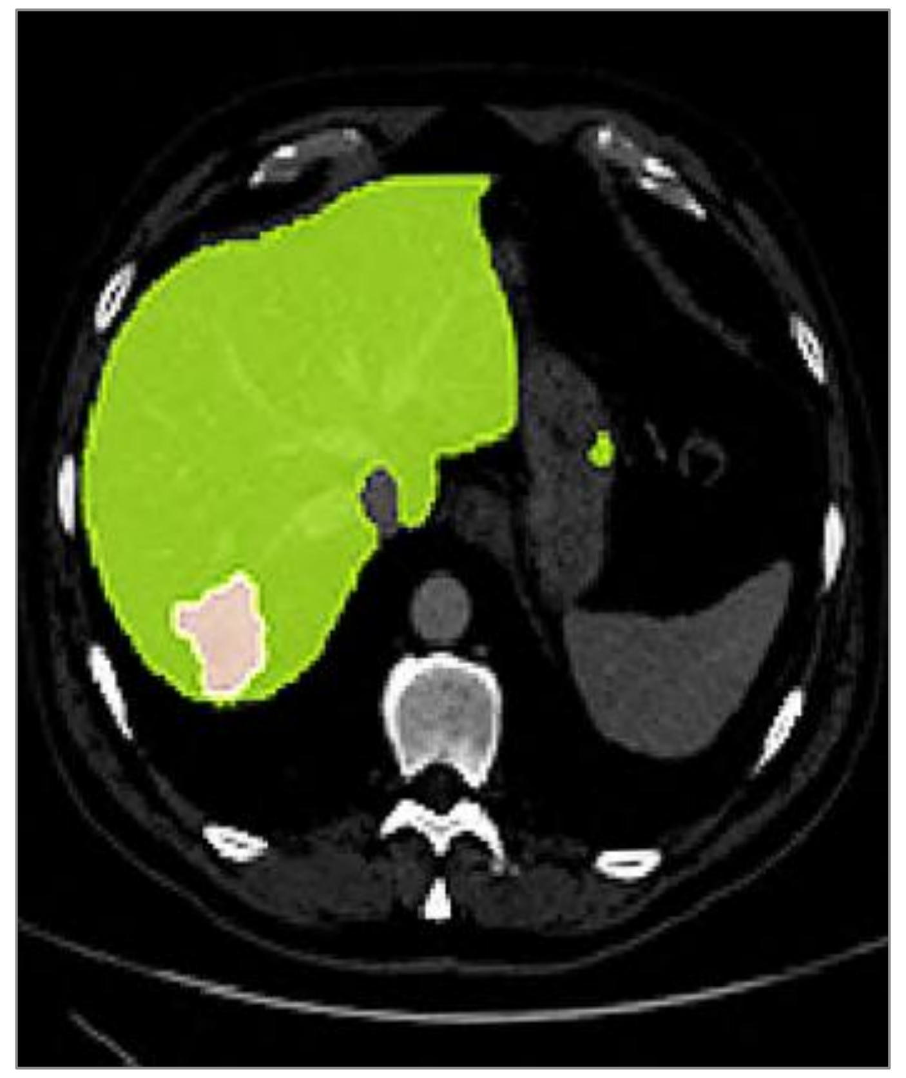
ISAAC
Robotics



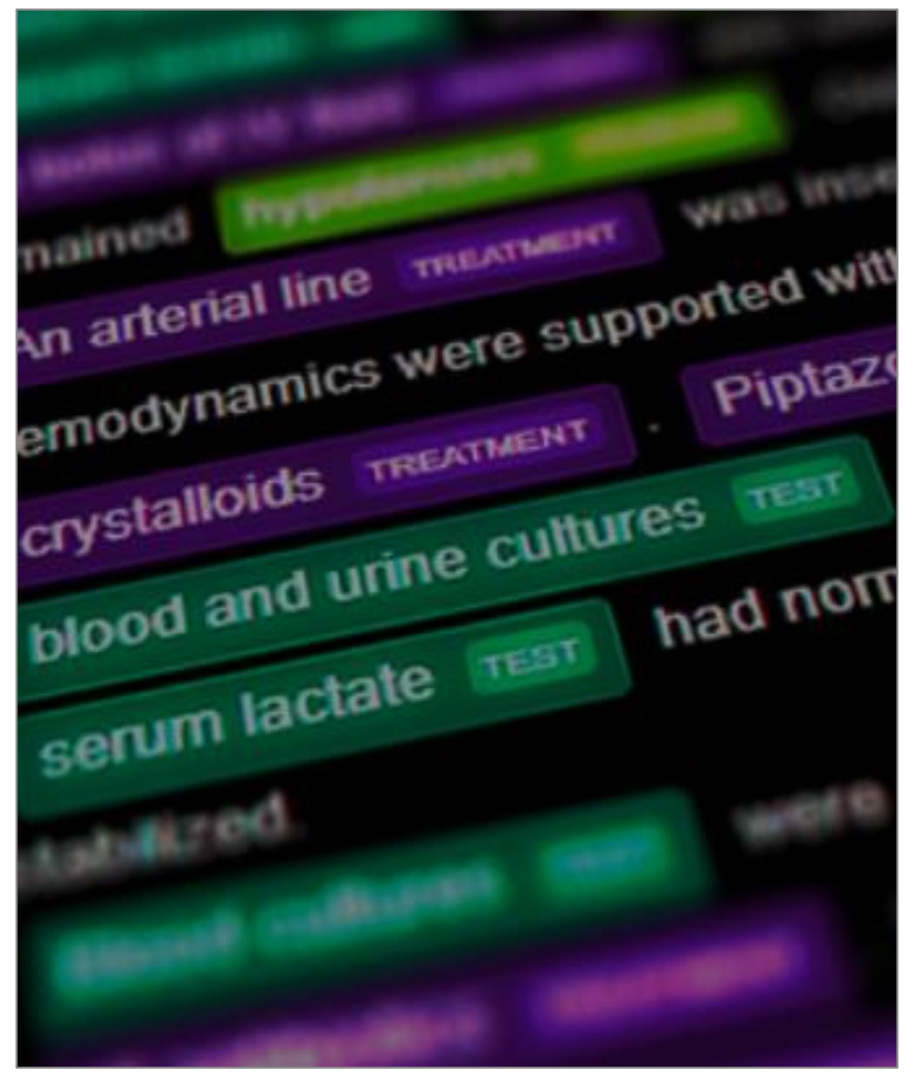
HOLOSCAN
Instruments



BIONEMO
Biomolecules



MONAI
Imaging



An arterial line TREATMENT
emodynamics were supported wit
crystalloids TREATMENT
blood and urine cultures TEST
serum lactate TEST
had nor

NEMO
Natural Language

Federated Learning

NVIDIA BioNeMo Framework

Enables Data Scientists and Researchers Train on DNA, Protein, Chemistry Data

Model Pre-Training

Foundation Model

Data Loaders

Optimized Training

Optimize

Model Fine-Tuning

Genomic	Geometric	Structure
DNABert$^{\text{NEW}}$	Equidock$^{\text{NEW}}$	ESM-2
scGPT	DiffDock$^{\text{NEW}}$	ProtT5
	MegaMolBART	OpenFold$^{\text{NEW}}$
	MolMIIM	

Training Framework

Pre-Trained Models

Microsoft Azure aws Google Cloud ORACLE

search for the title on NVIDIA On-Demand

Genomics Recommended Sessions to Attend

FEATURED TALKS

Kimberly Powell Special Address: Generative AI is Accelerating Healthcare into One of the Largest Technology Industries [S62604]
Kimberly Powell | General Manager and VP, Healthcare and Life Sciences
Tuesday, Mar 19 | 8:00 AM PDT

The Role of Generative AI in Modern Medicine [S62777]
Kimberly Powell | General Manager and VP, Healthcare and Life Sciences
Eric Topol | Professor and Executive Vice President
Catherine D. Wood | Chief Executive Officer/Chief Investment Officer
Peter Lee | Corporate Vice President of Research and Incubations
Tuesday, Mar 19 | 11:00 AM - 11:50 AM PDT

How Artificial Intelligence is Powering the Future of Biomedicine [S62283]
Priscilla Chan | Co-Founder and co-CEO, Chan Zuckerberg Initiative
Mona Flores | Global Head of Medical AI, NVIDIA
Tuesday, Mar 19 | 10:00 AM - 10:25 AM PDT

First-Ever Whole Transcriptome Imaging of Tissues using CosMx-SMI: Highest-Density Dataset Ever Collected [S61995]
Joseph Beechem | Senior Vice President of Research and Development, Nanostring
Wednesday, Mar 208:00 AM - 8:25 AM PDT

Nucleotide Transformer: Advancing Genomic Analysis with Large Language Models [S62438]
Karim Beguir | CEO and Co-founder, InstaDeep
Tuesday, Mar 19 | 9:00 AM - 9:25 AM PDT

Computer Vision for Rare Disease Genomic Medicine [S62535]
Wolfgang Pernice | Assistant Professor of Neurological Sciences, Columbia University In The City Of New York
Thursday, Mar 21 | 9:00 AM - 9:25 AM PDT

Introduction to GPU-Accelerated Genomics with Parabricks [S62322]
Harry Clifford | Genomics Product Lead, NVIDIA
Wednesday, Mar 20 | 4:30 PM - 4:55 PM PDT

WORKSHOPS & TRAININGS

Training DeepVariant Models using Parabricks [DLIT61813]
Thursday, Mar 21 | 2:00 PM - 3:40 PM PDT

Thanks for your attention!

Q&A