GWA-X : An Extensible GPU Accelerated Package for Permutation Testing in Genome-Wide Association Studies

Mulya Agung
MRC Human Genetics Unit, University of Edinburgh

Abstract

Genome-wide association studies (GWAS) aim to identify associations of genetic variants with a trait or disease. The scale of genomic datasets has increased to millions of genetic variants and hundreds of thousands of individuals, opening the possibilities for discoveries from GWAS. However, large-scale GWAS analyses are prone to high false positive rates because of the multiple testing problem. Permutation testing is the gold standard for maintaining false positive rates, yet it is impractical for large-scale GWAS because it requires vast computational resources.

This paper presents GWA-X, a software package that can fully benefit from GPUs and accelerate permutation testing in GWAS. In contrast to previous methods, GWA-X employs a novel whole-genome regression method to batch the computations of many genetic markers. It achieved a two-order magnitude speed-up compared with the CPU-based method and at least one-order magnitude speed-up compared with the current state-of-the-art GPU-based library. In addition, GWA-X provides an extensible framework for conducting permutation tests in GWAS.

1 Introduction

Genome-wide association studies (GWAS) scan a genome-wide set of genetic variants in a group of individuals to identify variants associated with a trait or disease. The growth of GWAS over the past decade has identified numerous associations for complex human traits and diseases. In recent years, the scale of genomic datasets has increased to millions of genetic variants and hundreds of thousands of individuals, opening the possibility for discoveries from GWAS and omics studies [1, 2].

However, large-scale GWAS demands vast computational resources due to the many variants or markers that need to be tested [3, 4]. In addition, conducting GWAS with millions of genetic variants or markers has a serious problem of the potentially high number of false positive results due to many variants tested [5, 6] and non-normal phenotypic distributions [7, 8].

Permutation tests are the gold standard for multiple testing correction in GWAS because they can reduce false positives while maintaining statistical power [9–11]. However, permutation tests are computationally demanding because they require many statistical tests to obtain a statistical distribution. The growth of scale in genomic datasets further increases the computational burden of permutation tests. The scale is expected to grow to billions of variants and millions of people in the future [1, 12]. Consequently, permutation testing is hardly practical in large-scale GWAS.

The end of Dennard scaling and the imminent end of Moore's law have led to the necessity of using specialized computing processors like graphics processing units (GPUs) to improve computational efficiency [13–15]. Because GPUs can perform numerical calculations faster than central processing units (CPUs) at comparable energy costs, they could significantly reduce GWAS's computational time and energy consumption. Although GPUs have been widely used in AI research [16], the use of GPUs in GWAS is still limited because existing GWAS tools that work well for CPUs cannot take advantage of GPUs due to hardware GPU constraints that are not present in CPUs. In addition, not all computations in GWAS permutation testing are suitable for GPUs. A collaboration between CPU and GPU is necessary to achieve high computational efficiency [17].

Several linear algebra libraries have been developed for GPUs [18, 19]. However, these libraries cannot efficiently parallelize GWAS' regressions because they aim to compute a regression of a large matrix problem in parallel to use all the GPU cores [20]. The regression in GWAS is computed for each variant or a group of variants, resulting in many regression computations of small matrices instead of one large matrix. These libraries will not fully utilize GPU cores in GWAS and cannot achieve significant performance improvements compared to CPUs.

Besides the computational demand, diverse permutation testing approaches have been developed for different appli-

cations in GWAS [10, 21, 22] simultaneously with novel regression methods to account for the population structure in the emerging genomic data [3, 23]. Meanwhile, the growth of genomic data size has imposed new computing system requirements, such as biobank platforms [2, 24]. GWAS permutation testing could benefit from the new methods and computing systems. However, existing GWAS and permutation testing tools cannot be extended easily due to their architecture limitations. Therefore, there is a demand for an extensible package for GWAS permutation testing to adapt to new developments.

This paper presents GWA-X, a software package that leverages GPUs to accelerate GWAS permutation testing. In contrast to previous methods, GWA-X batches regression computations of many genetic markers to achieve substantial acceleration on GPUs. The **contributions** of this paper are as follows.

- A whole-genome regression method for GPUs, including a genetic marker-based batch computation algorithm and a chunked data processing technique to handle large genotype data.
- A permutation testing strategy for GPUs that uses a memoization technique and a CPU-GPU collaboration strategy to reduce the computation time of permutation tests.
- An extensible and scalable software architecture that allows the addition of new regression algorithms, permutation testing approaches, and computing system requirements.

2 Related Work

Several libraries have been developed to accelerate batch matrix computations on GPUs [19, 20, 25], which focus mostly on square matrix problems or independent tall-skinny matrices. However, a GWAS analysis requires regressions of many matrices of genetic markers and the same matrices of the trait and covariates. These libraries cannot parallelize many regressions enough to maximize GPU occupancy. Hence, the computational performance of these libraries will not scale with many genetic markers in GWAS.

Most of the existing GWAS permutation testing methods focused on CPUs and cannot use GPUs [9, 26–28]. More recent methods use GPUs to accelerate the permutation tests in GWAS [10, 29]. However, these methods rely on existing GPU libraries that cannot fully utilize GPU's potential performance. In contrast with the GPU libraries and these permutation testing methods, GWA-X batches regression computations of many genetic markers. The evaluation of this paper demonstrates the superiority of GWA-X over an implementation with the current state-of-the-art GPU library. Furthermore, the previous methods are not extensible to more permutation testing approaches and computing system requirements.

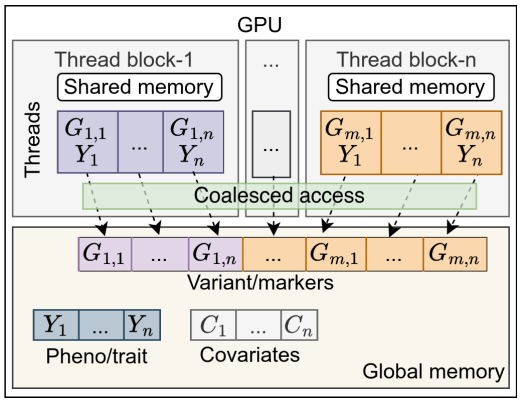

Figure 1: GWA-X's thread organization

3 Method

Typical GWAS approaches perform generalized linear models (GLM) (Equation 1) to test associations of genetic variants and traits depending on whether a continuous or binary trait is analyzed [5]. In GWAS permutation testing, the trait values are permuted while keeping the variants intact to account for linkage disequilibrium (LD), and the statistical tests are repeated for each permuted data set [9, 22, 27, 29]. The test results are then used to generate the distribution under the null hypothesis of no association. Therefore, the number of regression computations will increase linearly with the number of genetic variants and permutations. Regressions with many variants and large sample sizes are computationally demanding even without permutation. Therefore, accelerating the regressions is key to achieving a high computational efficiency in permutation tests.

$$Y = \alpha C + \beta G + \varepsilon, \tag{1}$$

where Y is a trait or disease of interest, g is a genetic marker, and C represents a matrix of covariates, and ε is the residual.

GWA-X accelerates the computational performance of GWAS permutation testing in two steps: it increases the degree of parallelization for regression computations on GPUs. Then, it employs a CPU-GPU collaboration strategy to accelerate permutation tests. The following subsections detail these steps.

3.1 Massively Parallel Regression on GPUs

GPUs have two memory types: on-chip and global memories. On-chip memory is faster but much smaller than global memory. In GPU programming, computation tasks are executed as units, called threads, organized into a grid of thread blocks. Each thread block can share data via an on-chip memory space known as shared memory, which is typically small and can only be accessed by threads within the same block. Before the computation, threads load the data from the global memory to

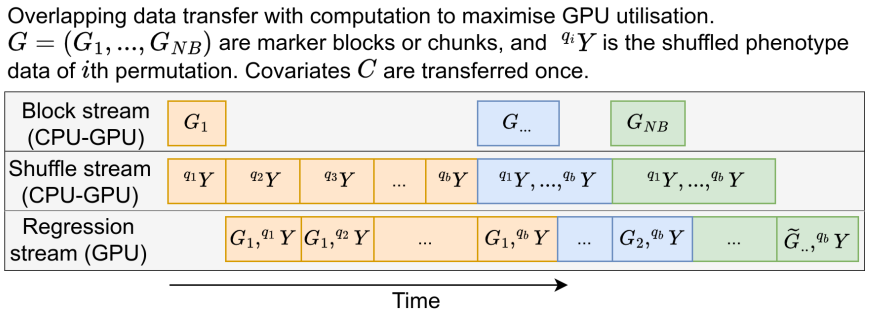

Figure 2: Overlapped computations with data transfers

the on-chip memory. Unlike CPUs, adjacent threads must access adjacent data elements in global memory to achieve high speeds on GPUs. That is why many programs implemented for CPUs often perform poorly on GPUs [30].

GPUs have thousands of processing units that can carry out many calculations in parallel, which is vital to accelerating regression computations in GWAS. To achieve high computational efficiency, a regression method must consider GPU memory constraints, whereas memory access is slower than computation. GWA-X increases the memory access throughput from/to GPU's global memory by coalesced access to maximize the use of GPU's memory bandwidth [20, 30].

Another key aspect is the organization of threads because it determines how many computation tasks can run in parallel and share data through shared memory. GWA-X uses the grid shown in Figure 1, which allows different markers and samples to be computed in parallel. Simultaneously, to use the fast shared memory, the threads of the same block compute the regression for the same marker.

In GLM, the genetic and fixed effects (β, α) can be calculated using Equation 2, where X is a concatenated matrix consisting of a genotype vector with a covariate matrix $(G|C)$ and Y is the vector of a trait. Thus, the regression requires a matrix inverse operation, which is computationally intensive. GWA-X accelerates regressions on GPUs by computing matrix inverse in parallel for different markers. It extends the previous method [31] to a batch matrix inverse computation of many markers.

$$\{\beta, \alpha\} = (X^T X)^{-1} X^T Y. \tag{2}$$

GPUs have limited memory capacity, and the data required might not fit into the GPU memory entirely, such as in whole-genome regression. GWA-X addresses this problem by chunking genotype data for data transfers and processing, including decompressing compressed genotypes on GPUs to increase GPU occupancy. However, multiple data transfers to process all the chunks may delay GPU computations. GWA-X reduces the delay by overlapping the data transfers with GPU computation (Figure 2).

3.2 Permutation Testing Strategy

CPUs and GPUs have unique features and strengths. Hence, dividing computational workloads among them can significantly improve computational efficiency [17]. For instance, data shuffling involves much more memory operations than numerical calculations. Hence, permuting trait values of large samples can be slower on GPUs than on CPUs due to the GPU's memory access latencies. In addition, transferring genotype data from/to CPUs and GPUs can delay GPU computations. GWA-X shuffles the trait data on CPUs and transfers them with chunked genotype data to GPUs that overlap with regression computations, as shown in Figure 2.

In permutation tests, trait values (Y) are shuffled for each permutation q, denoted as $({}^{q}Y)$. Then, the β coefficient is calculated for each permutation q using Equation 2. Then, permutation tests calculate test statistics for genetic markers using Equation 3, where t_q, β_q, and SE_q are the test statistic, coefficient, and standard error of the permutation q, respectively. Therefore, the covariance matrix, $(X^T X)^{-1}$, needs to be computed only once and can be reused for the whole permutation tests, including for the SE calculation. GWA-X applies a memoization technique to store and reuse these computation results across genetic markers to reduce computation time.

$$t_q = \beta_q / SE_q. \tag{3}$$

4 GWA-X Architecture

GWA-X's architecture, shown in Figure 3, comprises (1) a system layer containing computing system backends, (2) a library layer implementing regression algorithms and permutation testing approaches, (3) and an application layer to use the library.

As the size of genomic data will increase in the future [2], the use of distributed computing systems is necessary. Accordingly, the permutation methods must be scalable to distributed CPU-GPU systems, imposing new system requirements. The system layer allows additional backends to use different systems, such as distributed systems, without changing the other

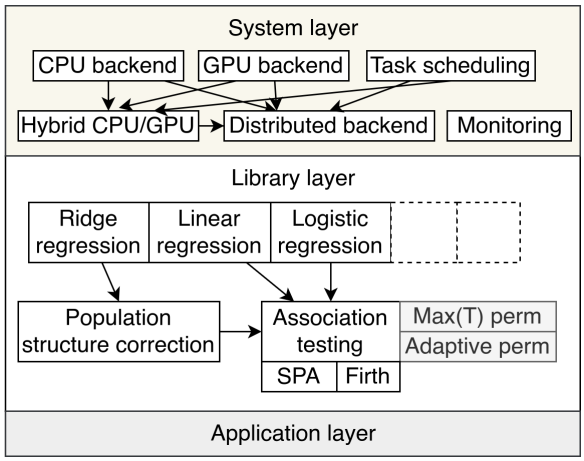

Figure 3: GWA-X's architecture

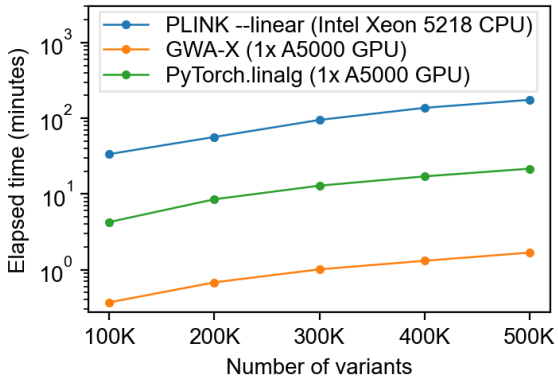

Figure 4: The computation time of GWAS regression

layers. This flexibility also broadens GWA-X applicability to various computing systems, such as biobank platforms [2, 24]. In addition, the system layer provides a task scheduler to optimize parallel executions on CPUs and GPUs, and a monitoring subsystem to track computational performance and energy usage.

The library layer contains implementations of regression algorithms and permutation testing approaches with a programming interface to reuse them. For instance, the regression and permutation testing routines can also be used for different analyses to benefit from GPUs, such as multi-omics [32] and rare variant analysis [22]. The application layer provides an easy-to-use user interface for conducting permutation tests.

4.1 Implementation

GWA-X is an active open-source project[†] developed at the University of Edinburgh, UK. It fully integrates with the Python environment and is easy to install using the Pip package manager. The regression method is built from scratch and compiled into machine code using the Numba compiler [33] to achieve comparable performance to a compiled programming language.

GWA-X is still at an initial stage. At the time of writing, a whole-genome regression method for quantitative traits and a standard permutation testing approach have been implemented. For accounting for population structure, GWA-X currently uses a matrix of covariates and the LOCO prediction file generated from Step 1 of the REGENIE tool [3]. As a future roadmap, GWA-X will support more regression algorithms, including logistic regression for binary traits, adaptive permutation approaches, and system backends to use distributed CPU/GPU systems. The extensible architecture will allow future development to be done in a modular way that does not require modifications to different layers.

[†]https://git.ecdf.ed.ac.uk/magung/gwa-x

5 Evaluation

The evaluation consists of computational efficiency and statistical performance comparisons. First, it compares the computational performance and energy consumption of GWA-X to a CPU-optimized software, PLINK [28], and an implementation using the GPU-accelerated PyTorch [19] library (Pytorch.linalg), which uses batch matrix computations on GPUs. The PyTorch-based implementation can be obtained from the GWA-X project repository.

To evaluate regression performance evaluation, GWA-X and the PyTorch-based implementation were run on a single NVIDIA A5000 GPU, while PLINK2 was run with the `-linear` option on a 16-core Intel Xeon 5128 CPU @2.3 GHz. For permutation tests, GWA-X performance is compared with PLINK `-mperm` on a single node of the ARCHER2 supercomputer [34], equipped with 128-core AMD EPYC 7742 processors. The evaluation uses a simulated data set of genotypes, a quantitative trait, and a covariate matrix with 430,000 samples and up to 500,000 genetic markers.

Figure 4 shows the computational performance of regression. GWA-X achieved a faster regression by a hundredfold compared with PLINK2 (95.5x on average) using only one GPU card, more than tenfold faster than PyTorch (12.5x on average). Figure 5 shows the energy consumptions and carbon emissions of GWA-X and PyTorch, measured using CodeCarbon [35] that is implemented in GWA-X's system layer. GWA-X reduced energy and CO2 emissions by 88.4% from faster regressions.

For permutation tests, GWA-X achieved a speed-up ratio of more than two orders of magnitude (99.5x on average) compared with PLINK on ARCHER2, as shown in Figure 6. The results suggest that GWA-X can save substantial cost and energy by replacing hundreds of CPU nodes with a GPU card. The GWA-X's speed-up ratio is much faster than the speed-up reported by the previous GPU-based methods [10, 29], demonstrating the effectiveness of GWA-X's regression method and permutation testing strategy.

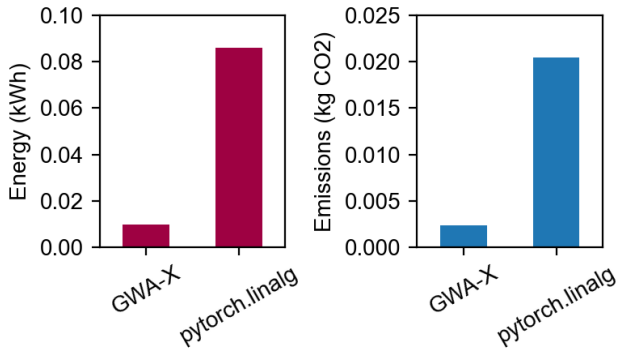

Figure 5: The energy usage and carbon emission of GWAS regression

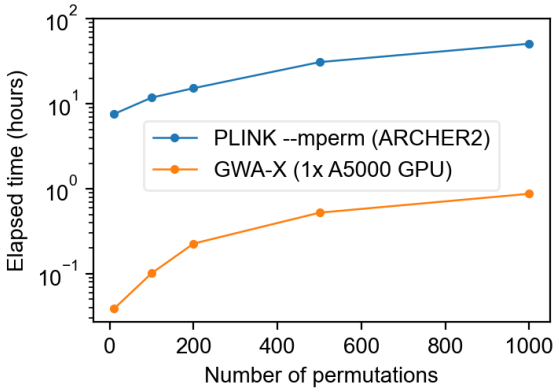

Figure 6: The computation time of permutation tests

5.1 Statistical Performance Evaluation

GWA-X statistical performance is evaluated by comparing the empirical power and false positive rates (FPR) of the standard genome-wide thresholds ($5e-8$) and two permutation-based thresholds obtained from 1,000 and 10,000 permutations (m). For the comparison, 10,000 causal variants or markers were simulated using the GCTA software ($h^2 = 0.2$) from the same dataset used in the computational performance evaluation. The significant variants are the variants with P values less than the threshold, while the empirical power is the proportion of causal variants detected as significant, and FPR is the proportion of significant variants that are non-causal.

Figure 7 shows the empirical power, false positive rate (FPR), and the number of significant variants detected using the standard and permutation-based significant thresholds. Compared with the standard threshold, the permutation-based thresholds detected more significant variants and achieved higher statistical power while maintaining FPR. The power increased with m, showing that the accuracy increased with the number of permutations. These results also show the benefit of GWA-X's acceleration, which allows more permutations

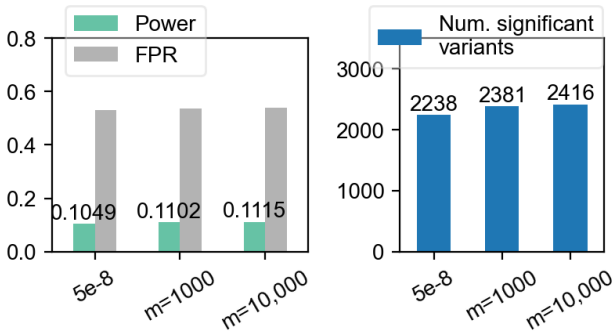

Figure 7: The empirical power, FPR, and number of significant variants of permutation-based thresholds

to achieve higher accuracy.

6 Conclusion

Currently, there is no GPU-accelerated software for GWAS permutation testing that is both efficient and extensible. GWA-X employs a novel whole-genome regression method and a CPU-GPU collaboration strategy to benefit fully from GPU computational capabilities. The computational efficiency and extensibility simultaneously achieved by its architecture are essential for emerging data scales and analyses in GWAS. The evaluation demonstrates unprecedented acceleration and energy saving on regression and permutation tests. Thus, GWA-X provides computational efficiency and flexibility for permutation testing in future GWAS.

References

[1] Backman, J. D., Li, A. H., Marcketta, A., Sun, D., Mbatchou, J., Kessler, M. D., Benner, C., Liu, D., Locke, A. E., Balasubramanian, S., et al. Exome sequencing and analysis of 454,787 uk biobank participants. *Nature 599*, 7886 (2021), 628–634.

[2] Biobank, M. B. A. L. . W. S. J. . W. V. A. . M. J. G. . C. M. S. . ., and of Us Research Demonstration Project Teams Choi Seung Hoan 14 http://orcid. org/0000-0002-0322-8970 Wang Xin 14 http://orcid. org/0000 0001-6042-4487 Rosenthal Elisabeth A. 15, A. Genomic data in the all of us research program. *Nature 627*, 8003 (2024), 340–346.

[3] Mbatchou, J., Barnard, L., Backman, J., Marcketta, A., Kosmicki, J. A., Ziyatdinov, A., Benner, C., O'Dushlaine, C., Barber, M., Boutkov, B., et al. Computationally efficient whole-genome regression for quantitative and binary traits. *Nature genetics 53*, 7 (2021), 1097–1103.

[4] Jiang, L., Zheng, Z., Fang, H., and Yang, J. A generalized linear mixed model association tool for biobank-scale data. *Nature genetics 53*, 11 (2021), 1616–1621.

[5] Uffelmann, E., Huang, Q. Q., Munung, N. S., De Vries, J., Okada, Y., Martin, A. R., Martin, H. C., Lappalainen, T., and Posthuma, D. Genome-wide association studies. *Nature Reviews Methods Primers 1*, 1 (2021), 59.

[6] PULIT, S. L., DE WITH, S. A., AND DE BAKKER, P. I. Resetting the bar: Statistical significance in whole-genome sequencing-based association studies of global populations. *Genetic epidemiology 41*, 2 (2017), 145–151.

[7] NEALE, B. M., RIVAS, M. A., VOIGHT, B. F., ALTSHULER, D., DEVLIN, B., ORHO-MELANDER, M., KATHIRESAN, S., PURCELL, S. M., ROEDER, K., AND DALY, M. J. Testing for an unusual distribution of rare variants. *PLoS genetics 7*, 3 (2011), e1001322.

[8] CHEN, W., COOMBES, B. J., AND LARSON, N. B. Recent advances and challenges of rare variant association analysis in the biobank sequencing era. *Frontiers in genetics 13* (2022), 1014947.

[9] PAHL, R., AND SCHÄFER, H. Permory: an ld-exploiting permutation test algorithm for powerful genome-wide association testing. *Bioinformatics 26*, 17 (2010), 2093–2100.

[10] JOHN, M., ANKENBRAND, M. J., ARTMANN, C., FREUDENTHAL, J. A., KORTE, A., AND GRIMM, D. G. Efficient permutation-based genome-wide association studies for normal and skewed phenotypic distributions. *Bioinformatics 38*, Supplement_2 (2022), ii5–ii12.

[11] JOO, J. W. J., HORMOZDIARI, F., HAN, B., AND ESKIN, E. Multiple testing correction in linear mixed models. *Genome biology 17* (2016), 1–18.

[12] WOOD, A., DENHOLM, R., HOLLINGS, S., COOPER, J., IP, S., WALKER, V., DENAXAS, S., AKBARI, A., BANERJEE, A., WHITELEY, W., ET AL. Linked electronic health records for research on a nationwide cohort of more than 54 million people in england: data resource. *bmj 373* (2021).

[13] DEAN, J., PATTERSON, D., AND YOUNG, C. A new golden age in computer architecture: Empowering the machine-learning revolution. *IEEE Micro 38*, 2 (2018), 21–29.

[14] VILLA, O., JOHNSON, D. R., OCONNOR, M., BOLOTIN, E., NELLANS, D., LUITJENS, J., SAKHARNYKH, N., WANG, P., MICIKEVICIUS, P., SCUDIERO, A., ET AL. Scaling the power wall: a path to exascale. In *SC'14: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis* (2014), IEEE, pp. 830–841.

[15] REED, D., GANNON, D., AND DONGARRA, J. Reinventing high performance computing: challenges and opportunities. *arXiv preprint arXiv:2203.02544* (2022).

[16] TALIB, M. A., MAJZOUB, S., NASIR, Q., AND JAMAL, D. A systematic literature review on hardware implementation of artificial intelligence algorithms. *The Journal of Supercomputing 77*, 2 (2021), 1897–1938.

[17] MITTAL, S., AND VETTER, J. S. A survey of cpu-gpu heterogeneous computing techniques. *ACM Computing Surveys (CSUR) 47*, 4 (2015), 1–35.

[18] DONGARRA, J., GATES, M., HAIDAR, A., KURZAK, J., LUSZCZEK, P., TOMOV, S., AND YAMAZAKI, I. Accelerating numerical dense linear algebra calculations with gpus. *Numerical Computations with GPUs* (2014), 1–26.

[19] ANSEL, J., YANG, E., HE, H., GIMELSHEIN, N., JAIN, A., VOZNESENSKY, M., BAO, B., BELL, P., BERARD, D., BUROVSKI, E., ET AL. Pytorch 2: Faster machine learning through dynamic python bytecode transformation and graph compilation. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2* (2024), pp. 929–947.

[20] DONGARRA, J., HAMMARLING, S., HIGHAM, N. J., RELTON, S. D., VALERO-LARA, P., AND ZOUNON, M. The design and performance of batched blas on modern high-performance computing systems. *Procedia Computer Science 108* (2017), 495–504.

[21] CHE, R., JACK, J. R., MOTSINGER-REIF, A. A., AND BROWN, C. C. An adaptive permutation approach for genome-wide association study: evaluation and recommendations for use. *BioData mining 7* (2014), 1–13.

[22] RAJABLI, F., AND KUNKLE, B. W. Strategies in aggregation tests for rare variants. *Current protocols 3*, 11 (2023), e931.

[23] GURINOVICH, A., LI, M., LESHCHYK, A., BAE, H., SONG, Z., ARBEEV, K. G., NYGAARD, M., FEITOSA, M. F., PERLS, T. T., AND SEBASTIANI, P. Evaluation of genesis, saige, regenie and fastgwa-glmm for genome-wide association studies of binary traits in correlated data. *Frontiers in Genetics 13* (2022), 897210.

[24] CONSORTIUM, U. B. W.-G. S., LI, S., CARSS, K. J., HALLDORSSON, B. V., AND CORTES, A. Whole-genome sequencing of half-a-million uk biobank participants. *medRxiv* (2023), 2023–12.

[25] ABDELFATTAH, A., BEAMS, N., CARSON, R., GHYSELS, P., KOLEV, T., STITT, T., VARGAS, A., TOMOV, S., AND DONGARRA, J. Magma: Enabling exascale performance with accelerated blas and lapack for diverse gpu architectures. *The International Journal of High Performance Computing Applications* (2024), 10943420241261960.

[26] STEISS, V., LETSCHERT, T., SCHÄFER, H., AND PAHL, R. Permory-mpi: a program for high-speed parallel permutation testing in genome-wide association studies. *Bioinformatics 28*, 8 (2012), 1168–1169.

[27] PAIK, H., CHO, Y., CHO, S. B., AND KWON, O.-K. Mpi-gwas: a supercomputing-aided permutation approach for genome-wide association studies. *Genomics & Informatics 20*, 1 (2022).

[28] CHANG, C. C., CHOW, C. C., TELLIER, L. C., VATTIKUTI, S., PURCELL, S. M., AND LEE, J. J. Second-generation plink: rising to the challenge of larger and richer datasets. *Gigascience 4*, 1 (2015), s13742–015.

[29] FREUDENTHAL, J. A., ANKENBRAND, M. J., GRIMM, D. G., AND KORTE, A. Gwas-flow: A gpu accelerated framework for efficient permutation based genome-wide association studies. *BioRxiv* (2019), 783100.

[30] HIJMA, P., HELDENS, S., SCLOCCO, A., VAN WERKHOVEN, B., AND BAL, H. E. Optimization techniques for gpu programming. *ACM Computing Surveys 55*, 11 (2023), 1–81.

[31] SHARMA, G., AGARWALA, A., AND BHATTACHARYA, B. A fast parallel gauss jordan algorithm for matrix inversion using cuda. *Computers & Structures 128* (2013), 31–37.

[32] DOMÍNGUEZ CONDE, C., XU, C., JARVIS, L., RAINBOW, D., WELLS, S., GOMES, T., HOWLETT, S., SUCHANEK, O., POLANSKI, K., KING, H., ET AL. Cross-tissue immune cell analysis reveals tissue-specific features in humans. *Science 376*, 6594 (2022), eabl5197.

[33] LAM, S. K., PITROU, A., AND SEIBERT, S. Numba: a llvm-based python jit compiler. In *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC* (New York, NY, USA, 2015), LLVM '15, Association for Computing Machinery.

[34] ARCHER2. Hardware - ARCHER2 user documentation, 2024. https://https://docs.archer2.ac.uk/user-guide/hardware [Accessed: 13 Sept 2024].

[35] COURTY, B., SCHMIDT, V., LUCCIONI, S., GOYAL-KAMAL, MARIONCOUTAREL, FELD, B., LECOURT, J., LIAMCONNELL, SABONI, A., INIMAZ, SUPATOMIC, LÉVAL, M., BLANCHE, L., CRUVEILLER, A., OUMINASARA, ZHAO, F., JOSHI, A., BOGROFF, A., DE LAVOREILLE, H., LASKARIS, N., ABATI, E., BLANK, D., WANG, Z., CATOVIC, A., ALENCON, M., STĘCHŁY, M., BAUER, C., DE ARAÚJO, L. O. N., JPW, AND MINERVABOOKS. mlco2/codecarbon: v2.4.1, May 2024.