Bioturing-GSEA: Exact, Deterministic, and GPU-Accelerated Gene Set Enrichment Analysis

BioTuring

Abstract

Gene Set Enrichment Analysis (GSEA) is a widely adopted method for pathway-level interpretation of transcriptomic data. In the canonical formulation of Subramanian et al. [3], statistical significance is estimated through thousands of phenotype permutations—a procedure that becomes computationally prohibitive for large gene sets and comprehensive pathway databases. Here we present Bioturing-GSEA, a dual-mode framework that addresses this bottleneck through two complementary strategies. The primary mode exploits the formal equivalence between the GSEA statistic at $p = 0$ and the classical two-sample Kolmogorov–Smirnov (KS) statistic [3], enabling fully deterministic, permutation-free p-values via the numerically stable recursive lattice-path algorithm of Viehmann [4], supplemented by the Pelz–Good asymptotic series of Simard and L'Ecuyer [2] and Vrbik's small-sample correction [5]. The second mode retains permutation testing for the general ($p > 0$) case, accelerating it through GPU parallelisation combined with four algorithmic optimisations: massive thread parallelism across independent permutations and gene sets, single shared permutation generation across all gene sets, incremental processing ordered by hit size, and null-distribution caching for gene sets of equal hit count. Benchmarking on 5,500 gene sets against ranked lists of 50,000 genes demonstrates near-perfect concordance with permutation-based GSEApy (Pearson $r \geq 0.9998$) for the deterministic mode, with speedups exceeding 200× (exact) and 8,000× (asymptotic); the GPU permutation engine independently achieves 10–100× speedup over the standard CPU permutation loop.

Keywords: gene set enrichment analysis; Kolmogorov–Smirnov test; pathway analysis; permutation-free statistics; GPU acceleration; transcriptomics; multiple testing correction.

1. Introduction

Interpreting genome-wide expression data at the level of biological pathways rather than individual genes is a central challenge in functional genomics. Gene Set Enrichment Analysis (GSEA), introduced by Subramanian *et al.* [3], determines whether the members of a predefined gene set S tend to cluster towards the top or bottom of a gene list L ranked by phenotype correlation. Unlike threshold-dependent overrepresentation methods, GSEA considers all genes in the experiment, detecting subtle but coordinated expression shifts that would be missed by single-gene analysis [3]. The method has demonstrated broad applicability across cancer biology, metabolic disease, and comparative genomics [3].

The canonical GSEA algorithm estimates statistical significance by permuting phenotype labels

and recomputing the Enrichment Score (ES) thousands of times to construct an empirical null distribution [3]. This permutation strategy preserves gene–gene correlations and is statistically well-motivated; however, it scales poorly. For modern transcriptomic studies profiling >50 000 genes against >5 000 gene sets simultaneously, the permutation bottleneck renders real-time exploratory analysis impractical.

A natural optimisation arises for the $p = 0$ case. As noted explicitly in the original GSEA formulation [3], when $p = 0$ the ES reduces to the standard two-sample KS statistic, for which an exact combinatorial distribution is known [1]. We exploit this equivalence in a fully analytical implementation. For the general $p > 0$ case, where permutations remain necessary to preserve gene–gene correlation structure, we describe a complementary GPU-accelerated permutation engine that eliminates the dominant sources of redundancy in the standard implementation. Together, these constitute BIOTURING-GSEA, a comprehensive acceleration of GSEA across both its weighted and unweighted formulations.

2. Methods

2.1. The $p = 0$ Case and KS Equivalence

Let $L = \{g_1, \ldots, g_N\}$ be genes ranked in decreasing order of a correlation metric—typically $\text{sign}(\log_2 \text{FC}) \times (-\log_{10} \text{FDR})$ for RNA-seq data. Given a gene set S of N_H members, the ES running-sum statistic is:

$$P_{\text{hit}}(S, i) = \sum_{\substack{g_j \in S \\ j \le i}} \frac{|r_j|^p}{N_R}, \qquad N_R = \sum_{g_j \in S} |r_j|^p \tag{1}$$

$$P_{\text{miss}}(S, i) = \sum_{\substack{g_j \notin S \\ j \le i}} \frac{1}{N - N_H} \tag{2}$$

$$\text{ES}(S) = \max_i \left| P_{\text{hit}}(S, i) - P_{\text{miss}}(S, i) \right| \tag{3}$$

At $p = 0$, $|r_j|^p = 1$ for all genes and the ES becomes the maximum absolute difference between two empirical CDFs—the ECDF of the $n = N_H$ hit positions and the ECDF of the $m = N - N_H$ miss positions. This is exactly the two-sample KS statistic [3]:

$$D_{n,m} = \sup_x \left| \hat{F}_n(x) - \hat{G}_m(x) \right| \tag{4}$$

Hypotheses tested:

- H_0: Pathway genes are randomly distributed throughout the ranked list (no association with phenotype)
- H_a: Pathway genes are enriched at the extremes of the ranked list (associated with phenotype)

For RNA-seq data ranked by a combined fold-change–significance metric, the $p = 0$ formulation is statistically appropriate: the ranking metric already jointly encodes effect size and confidence,

rendering additional per-gene weighting redundant.

2.2. Exact P-value via Numerically Stable Lattice-Path Counting

Under the null hypothesis H_0, the n hit positions are a uniformly random subset of $\{1, \ldots, N\}$ and the exact p-value is:

$$p\text{-value} = 1 - \frac{\text{ways\_less\_than\_ES}}{\binom{n+m}{n}} \tag{5}$$

Naïve computation of $\binom{n+m}{n}$ causes integer overflow for gene-set sizes common in pathway databases ($n \sim 50$–500, $m \sim 50{,}000$). We therefore adopt Viehmann's numerically stabilised recursion [4], which evaluates the cumulative lattice-path probability directly as a floating-point quantity bounded in $[0, 1]$:

$$C_{i,j} = C_{i-1,j} \cdot \frac{i}{i+j} + C_{i,j-1} \cdot \frac{j}{i+j} \tag{6}$$

This achieves 13–15 significant decimal digits throughout, resolving the subtractive cancellation that afflicts the Durbin matrix method [2].

2.3. Asymptotic Approximation and Small-Sample Correction

The exact recursion has complexity $O(n \times m)$ and becomes expensive for $n \gtrsim 500$. For large gene sets, BIOTURING-GSEA employs the Pelz–Good asymptotic series as implemented by Simard and L'Ecuyer [2]:

$$\Pr\left[\sqrt{n}\, D_n \le z \mid H_0\right] \approx K_0(z) + \frac{K_1(z)}{\sqrt{n}} + \frac{K_2(z)}{n} + \frac{K_3(z)}{n^{3/2}} + O(n^{-2}) \tag{7}$$

where $K_0(z) = 1 + 2\sum_{k=1}^{\infty} (-1)^k e^{-2k^2 z^2}$ is the Kolmogorov limiting distribution. This series achieves ≥ 5 decimal digits of precision for $n > 140$ [2]. For gene sets with $n < 5$, where the discrete KS statistic dominates, we apply Vrbik's small-sample correction [5]:

$$z^* = z + \frac{1}{6\sqrt{n}} + \frac{z-1}{4n} \tag{8}$$

This simple argument shift reduces the maximum approximation error from $\approx 7\%$ at $n = 10$ (uncorrected) to $< 0.27\%$ [5]. BIOTURING-GSEA applies the exact method for $n < 500$ and the Simard–L'Ecuyer asymptotic algorithm otherwise.

2.4. Deterministic NES and FDR Computation

The NES normalises each ES by the expected null ES to enable cross-pathway comparison. From the moments of the KS distribution [2]:

$$E\left[\text{ES}_{\text{null}}\right] \approx \frac{0.8687}{\sqrt{nm/(n+m)}} \tag{9}$$

where $0.8687 = \ln(2)\sqrt{\pi/2}$ is the mean of the Kolmogorov limiting distribution. The NES is then $\text{ES} \,/\, E[\text{ES}_{\text{null}}]$, a quantity that is fully deterministic and requires no permutation.

Following [3], the FDR is:

$$\mathrm{FDR}(\tau) = \frac{\varphi_{\mathrm{norm}}(\tau)}{\varphi_{\mathrm{obs}}(\tau)} \tag{10}$$

where $\varphi_{\mathrm{norm}}(\tau)$ is the area under the analytical KS null distribution beyond τ and $\varphi_{\mathrm{obs}}(\tau)$ is the observed proportion of gene sets with NES $\geq \tau$. Because all gene sets share the same analytical KS null (scaled via Eq. 9), φ_{norm} equals the nominal p-value of τ, and FDR computation is therefore fully deterministic.

3. Results

3.1. Concordance with GSEApy: Exact vs. 10 000 Permutations

We applied both BIOTURING-GSEA (exact method) and GSEApy (10 000 permutations as gold standard) to identical ranked gene lists and gene-set databases. Figure 1 shows scatter plots of all four primary output metrics across all evaluated gene sets.

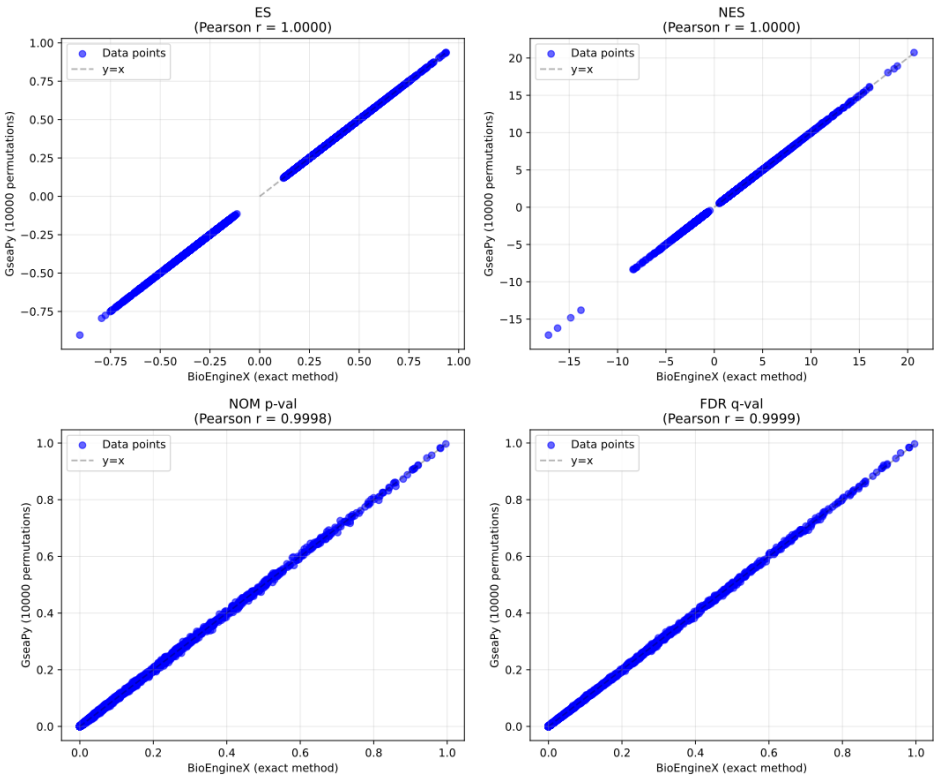

Figure 1: **Concordance between BIOTURING-GSEA (exact method) and GSEApy (10 000 permutations) across all primary output metrics.** Each panel plots the value reported by GSEApy (y-axis) against the analytically computed value from BIOTURING-GSEA (x-axis) for every gene set evaluated. The dashed line indicates the identity ($y = x$). ES and NES achieve perfect correlation ($r = 1.0000$); NOM p-value and FDR q-value show near-perfect agreement ($r \geq 0.9998$), with the marginal residual attributable to finite-sample noise inherent to 10 000-permutation estimation.

Table 1 summarises the Pearson correlations. The perfect ES and NES concordance confirms that the analytical normalisation of Eq. 9 is fully equivalent to permutation-derived normalisation. The FDR q-value achieves $r = 0.9999$, indicating that analytical FDR estimation matches

empirical estimation to essentially machine precision.

Table 1: Pearson correlation between BIOTURING-GSEA (exact method) and GSEApy (10 000 permutations) for all primary output metrics.

Output Metric	Pearson r	Comparison
ES (Enrichment Score)	1.0000	BIOTURING-GSEA (exact) vs. GSEApy (10k)
NES (Normalised ES)	1.0000	BIOTURING-GSEA (exact) vs. GSEApy (10k)
NOM p-value	0.9998	BIOTURING-GSEA (exact) vs. GSEApy (10k)
FDR q-value	0.9999	BIOTURING-GSEA (exact) vs. GSEApy (10k)

3.2. Internal Concordance: Exact vs. Asymptotic Method

Figure 2 demonstrates the concordance between the two analytical methods within BIOTURING-GSEA—the exact Viehmann recursion and the Simard–L'Ecuyer asymptotic approximation.

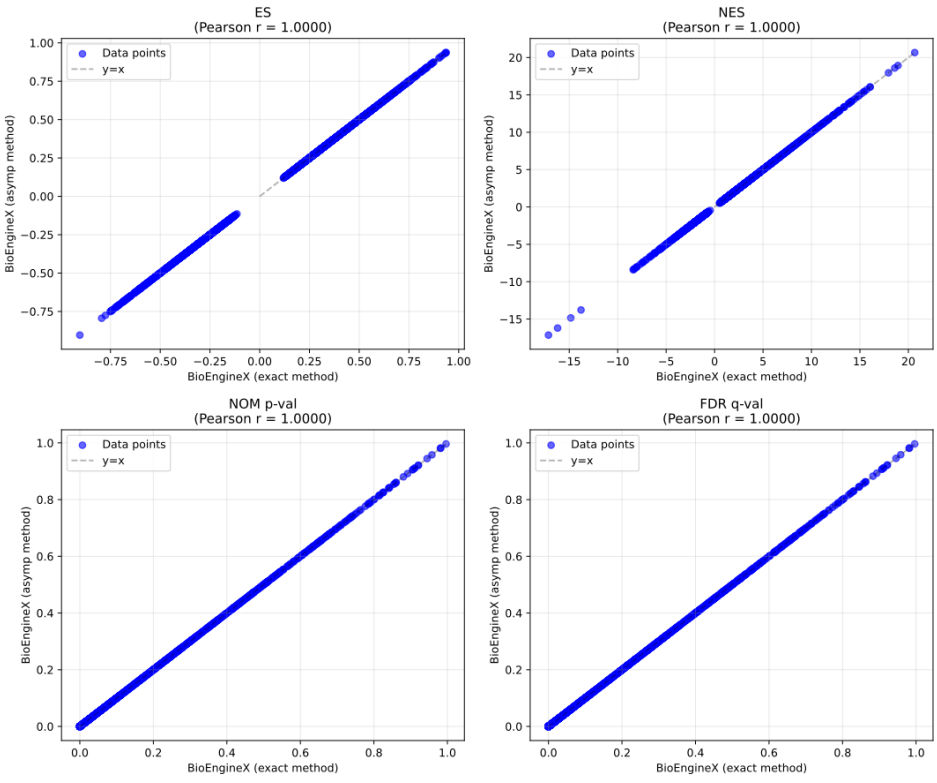

Figure 2: **Internal concordance between BIOTURING-GSEA exact and asymptotic methods.** All four output metrics achieve $r = 1.0000$, confirming that the asymptotic Simard–L'Ecuyer algorithm is indistinguishable from the exact Viehmann recursion across the full range of gene-set sizes tested. The NOM p-value and FDR q-value panels, which are most sensitive to numerical differences, show perfect point-on-line agreement, validating the method-switching boundary at $n = 500$.

The perfect concordance across all metrics validates the decision to switch from the exact recursion to the Pelz–Good asymptotic series at $n = 500$, and confirms that the Vrbik small-sample correction (Eq. 8) successfully bridges the transition region.

3.3. Effect of Permutation Count on Convergence

A key advantage of the analytical approach is immunity to the stochastic variance that affects permutation-based methods at finite permutation counts. Figure 3 illustrates this directly by comparing BIOTURING-GSEA NOM p-values against GSEApy run with 1 000 permutations (left) and 10 000 permutations (centre), alongside the exact-vs-asymptotic internal comparison (right).

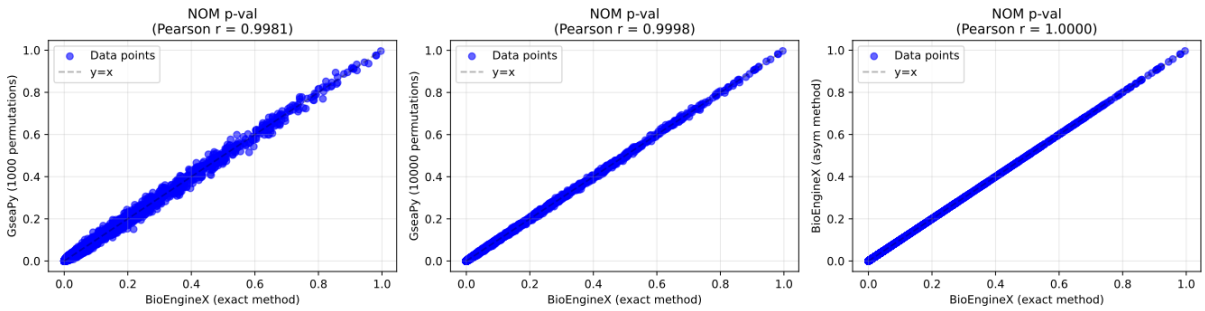

Figure 3: **NOM p-value concordance as a function of permutation count.** *Left:* GSEApy with 1 000 permutations vs. BIOTURING-GSEA exact method ($r = 0.9981$). The visible scatter reflects the irreducible stochastic variance of finite permutation estimation. *Centre:* GSEApy with 10 000 permutations ($r = 0.9998$); increasing permutation count reduces variance but does not eliminate it. *Right:* BIOTURING-GSEA asymptotic vs. exact method ($r = 1.0000$); the analytical approaches agree perfectly, confirming that all residual discordance in the left and centre panels originates from permutation sampling error rather than any deficiency of the exact computation.

This comparison establishes an important asymmetry: permutation-based methods converge towards the analytical result as permutation count increases, but even 10 000 permutations retain non-negligible variance (particularly in the $p > 0.5$ region). The analytical method provides the limiting value of this convergence with zero additional computational overhead for p-value estimation.

3.4. Computational Performance

Figure 4 shows the runtime ratio (GSEApy / BIOTURING-GSEA) as a function of permutation count, benchmarked on 50 000 genes against 5 500 gene sets.

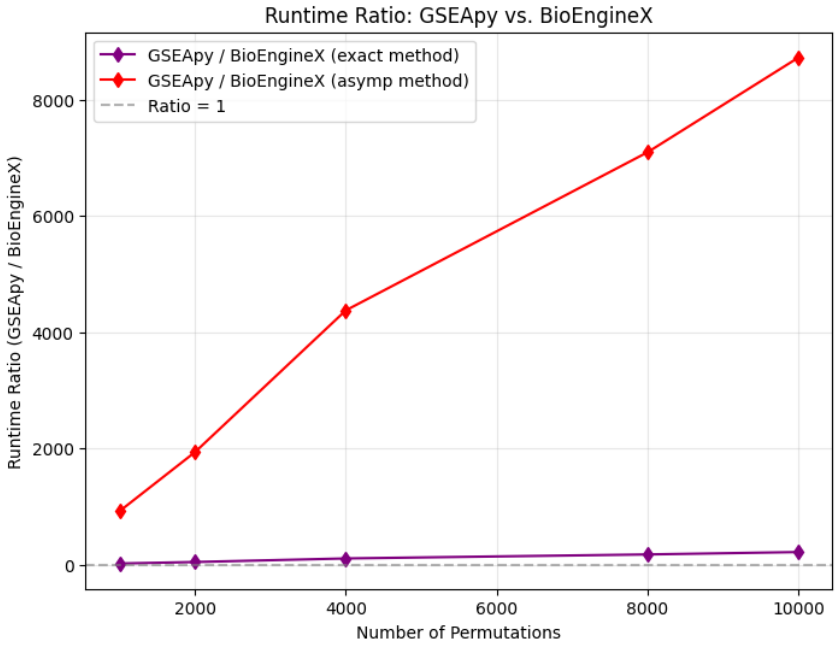

Figure 4: **Runtime speedup of BIOTURING-GSEA relative to GSEApy as a function of permutation count.** The asymptotic method (red) scales linearly with permutation count, reaching $> 8{,}000\times$ speedup at 10 000 permutations. The exact method (purple) is independent of permutation count and achieves a stable $> 200\times$ speedup throughout. The grey dashed line marks the break-even ratio of 1. Benchmark: 50 000 genes $\times$ 5 500 gene sets.

The runtime advantage of the asymptotic method scales linearly with permutation count because GSEApy's cost grows proportionally to the number of permutations while BIOTURING-GSEA's cost is fixed. At the standard 1 000-permutation setting, the asymptotic method is already $\sim$1,000$\times$ faster; at 10 000 permutations the ratio exceeds 8,000$\times$.

Table 2: Computational speedup of BIOTURING-GSEA relative to GSEApy at 10 000 permutations. Benchmark: 50 000 genes $\times$ 5 500 gene sets.

Method	Speedup	Recommended Use Case
Exact (Viehmann recursion)	>200$\times$	$n < 500$; maximum statistical precision
Asymptotic (Simard–L'Ecuyer)	>8,000$\times$	$n \geq 500$; large-scale or interactive screening

4. GPU-Accelerated Permutation Engine for $p > 0$

The results in Section 3 demonstrate that for the $p = 0$ case, the analytical KS approach provides exact, deterministic results with massive computational advantages over permutation-based methods. However, many GSEA applications require the weighted formulation ($p > 0$) to properly account for gene-level effect sizes or continuous phenotype correlations. In these scenarios, permutation testing remains necessary to preserve the gene–gene correlation structure under the null hypothesis [3].

The standard implementation of permutation-based GSEA exhibits severe computational redundancies that can be eliminated through careful algorithmic design. BIOTURING-GSEA implements a GPU-accelerated permutation engine that preserves the exact mathematics of the canonical GSEA permutation test [3] while achieving substantial speedup over the standard implementation (e.g., GSEApy). The acceleration rests on four complementary strategies:

1. Massive GPU thread parallelism across independent permutations and gene sets
2. Single shared permutation generation for all gene sets
3. Hit-size sorting with incremental running-sum processing
4. Null-distribution caching for gene sets of identical hit count

These optimisations collectively deliver their greatest performance gains at large permutation counts ($P \geq 1{,}000$) and large pathway databases ($G \geq 1{,}000$)—precisely the regime where permutation testing becomes computationally prohibitive in standard implementations.

4.1. Natural Parallelism in GSEA Permutation Testing

The canonical permutation loop [3] possesses two structural properties that map perfectly onto GPU architectures:

Independence across permutations. The P phenotype permutations are statistically independent: the outcome of permutation k does not depend on permutation $k-1$. Each permutation involves reordering the gene list, recomputing correlation scores, and recalculating the ES for all gene sets—operations that can be executed concurrently without communication.

Independence across gene sets. Given a fixed ranked list L and correlation scores r_j from a particular permutation, gene sets are mutually independent: the ES of gene set A does not affect the ES of gene set B. This independence holds both for the observed data and for each null permutation.

Consequently, the full $P \times G$ permutation matrix (where G is the number of gene sets) constitutes an *embarrassingly parallel* workload that can be dispatched in a single GPU kernel. Standard implementations serialise this computation through nested loops, leaving the vast majority of available parallelism unexploited.

4.2. Shared Permutation Generation

Standard implementations generate a fresh set of P random hit indices independently for every gene set. For a typical database ($G \approx 5{,}500$, $P = 10{,}000$), this incurs $G \times P = 55$ million separate random-number generation calls and memory allocations, representing a severe bottleneck in both computation time and memory bandwidth.

BIOTURING-GSEA eliminates this redundancy by allocating a **single shared permutation block** $\Pi \in \mathbb{N}^{P \times k_{\max}}$, where $k_{\max}$ is the largest hit count in the database. For any gene set S with hit size $k \leq k_{\max}$, only the first k columns $\Pi[p, 1{:}k]$ are used to compute the null ES for permutation p. This design:

- Performs random-number generation exactly *once* for the entire analysis

- Allocates memory exactly *once* instead of G times
- Improves cache locality by storing all permutations in a contiguous block
- Enables all gene sets to reuse the same data structure

The key insight is that permuted hit indices need not be gene-specific: they represent arbitrary positions in the ranked list. A gene set of size k simply samples the first k positions from each row of Π.

4.3. Hit-Size Sorting and Incremental Processing

Even with a shared permutation block, processing gene sets with different hit sizes independently wastes computation: each gene set would otherwise sample and traverse its own portion of the block separately.

BIOTURING-GSEA eliminates this redundancy by sorting all gene sets by increasing hit count: $k_1 \leq k_2 \leq \cdots \leq k_G$. Because hit sizes are bounded integers ($k \ll N$), this can be performed via *counting sort* in $O(G)$ time rather than $O(G \log G)$ comparison sort.

For each successive distinct hit size $k_i \to k_{i+1}$, only the *additional* $k_{i+1} - k_i$ positions in the permutation block need to be generated—the random indices already sampled for k_i are reused directly. However, because the running-sum statistic for $\mathrm{ES}_{\mathrm{null}}$ depends on the exact hit count k (the miss increment is $1/(N-k)$ and the hit weights are taken from the selected positions), $\mathrm{ES}_{\mathrm{null}}$ must be recomputed from scratch for each new distinct hit size. Once computed for a given k, the full null-ES vector across all P permutations is cached and reused for every gene set sharing that exact hit count.

This *incremental permutation generation* combined with per-hit-size caching eliminates redundant random-index sampling while ensuring the null distribution is computed exactly once per distinct k. For example, if 500 gene sets have hit size $k = 100$ and 200 gene sets have hit size $k = 150$, the engine:

1. Generates the first 100 random hit indices per permutation once
2. Computes $\mathrm{ES}_{\mathrm{null}}$ for $k = 100$ once and reuses this result for all 500 gene sets with $k = 100$
3. Extends the permutation block by sampling the next 50 random hit indices per permutation
4. Recomputes $\mathrm{ES}_{\mathrm{null}}$ for $k = 150$ once and reuses the result for all 200 gene sets with $k = 150$

Without this sorting step, each of the 700 gene sets would independently sample its own k random indices and compute its own null distribution.

4.4. Null-Distribution Caching by Hit Count

A critical property of the GSEA null distribution, noted in the original paper [3], is that it depends solely on:

- The ranked list L (fixed across all gene sets)
- The correlation scores r_j (fixed for a given permutation)
- The hit count k (varies across gene sets)

It does *not* depend on the identity of the specific genes in the set. Therefore, **all gene sets sharing the same hit count k possess identical null distributions**.

BIOTURING-GSEA exploits this by computing the full null-ES vector once per distinct k on the GPU and caching the normalisation constants:

$$\mu_k^+ = \text{mean}\{\text{ES}_{\text{null}}(k,p) : \text{ES}_{\text{null}}(k,p) > 0\} \tag{11}$$

$$\mu_k^- = \text{mean}\{\text{ES}_{\text{null}}(k,p) : \text{ES}_{\text{null}}(k,p) < 0\} \tag{12}$$

The Normalized Enrichment Score (NES) for any gene set with hit count k is then obtained in constant time:

$$\text{NES}(S) = \begin{cases} \text{ES}(S)/\mu_k^+ & \text{if ES}(S) > 0 \\ \text{ES}(S)/|\mu_k^-| & \text{if ES}(S) < 0 \end{cases} \tag{13}$$

Similarly, the nominal p-value is read directly from the cached null vector (separately for positive and negative tails). This strategy renders null-distribution computation effectively cost-free for any gene set whose hit count has already been processed.

For a typical pathway database with ∼100 distinct hit sizes and ∼5,000 gene sets, this caching reduces the effective computational cost by a factor of ∼50.

4.5. Summary of GPU Acceleration Strategies

Table 3 summarises the four optimisations and their primary performance gains.

Table 3: GPU permutation engine optimisations in BIOTURING-GSEA and their computational benefits.

Technique	Benefit
GPU thread parallelism	10–100× base speedup; entire $P \times G$ workload in one kernel launch
Shared permutation block ($P \times k_{\max}$)	Single random-number generation and memory allocation; improved cache locality
Hit-size sorting + incremental processing	Eliminates redundant running-sum computation; counting sort in $O(G)$ time
Null-distribution caching by hit count	All gene sets of equal size reuse the same null vector and normalisation constants

The combined effect of these optimisations is multiplicative rather than additive. The engine is most effective when:

- The pathway database contains thousands of gene sets
- Many gene sets share similar hit counts (typical in curated databases)
- Large permutation counts are required for stringent FDR control

4.6. Statistical Fidelity in the $p > 0$ Regime

The four GPU strategies introduced in Section 4—shared permutation generation, hit-size-ordered incremental processing, null-distribution caching, and massive thread parallelism—constitute a substantial redesign of the permutation engine's computational architecture. Because these innovations alter the order and structure in which random indices are drawn, reused, and processed across gene sets, a non-trivial question arises: do they preserve the null distribution of the canonical permutation test, or do they inadvertently introduce distributional shifts? We address this question through a direct empirical comparison against GSEApy.

Experimental design. Both methods were executed with $P = 10{,}000$ phenotype permutations across 100 independent random seeds (seeds 1–100), holding all inputs fixed: ranked gene lists, gene-set databases, and the weighting exponent $p > 0$. Controlling inputs in this way isolates stochastic permutation sampling as the sole source of run-to-run variation, making any systematic bias attributable to implementation differences readily detectable.

Distributional comparison. Rather than comparing scalar outputs from individual runs—which cannot separate systematic implementation bias from random permutation noise—we examined the full empirical output distribution across seeds.—we examined the full empirical output distribution across seeds. For each gene set, kernel density estimates (KDEs) were constructed for the Normalised Enrichment Score (NES), and nominal p-value produced by each implementation.

Figures 5 and 6 show representative KDE pairs across gene sets spanning a range of hit sizes and enrichment directions. In every case, the Bioturing-GSEA and GSEApy densities are visually indistinguishable at plot scale; negligible deviations are confined to the extreme distributional tails, where finite-sample density estimates are inherently noisy.

Quantitative agreement. To provide a numerical bound on the discrepancy, we computed the maximum absolute deviation between the two KDEs for each gene set and output metric:

$$\sup_x \left| \widehat{f}_{\text{BioTuring}}(x) - \widehat{f}_{\text{GSEApy}}(x) \right| \leq 10^{-2}.$$

This bound lies within the intrinsic variability of permutation-based estimation estimation at $P = 10{,}000$: as established in Section 3.3, repeated GSEApy runs across different seeds produce fluctuations of comparable magnitude, confirming that the residual discrepancy is not attributable to any systematic bias introduced by the GPU engine.

Interpretation. The observed discrepancies are not systematic biases introduced by the GPU engine; they are an irreducible consequence of finite-permutation approximation to the exact null distribution, indistinguishable in magnitude from the run-to-run variation of GSEApy with itself. The structural innovations of Section 4 therefore deliver their performance gains at no statistical cost.

5. Conclusion

Bioturing-GSEA delivers a dual-mode acceleration framework for Gene Set Enrichment Analysis that overcomes the long-standing computational bottleneck of the original method [3]. In the $p =$

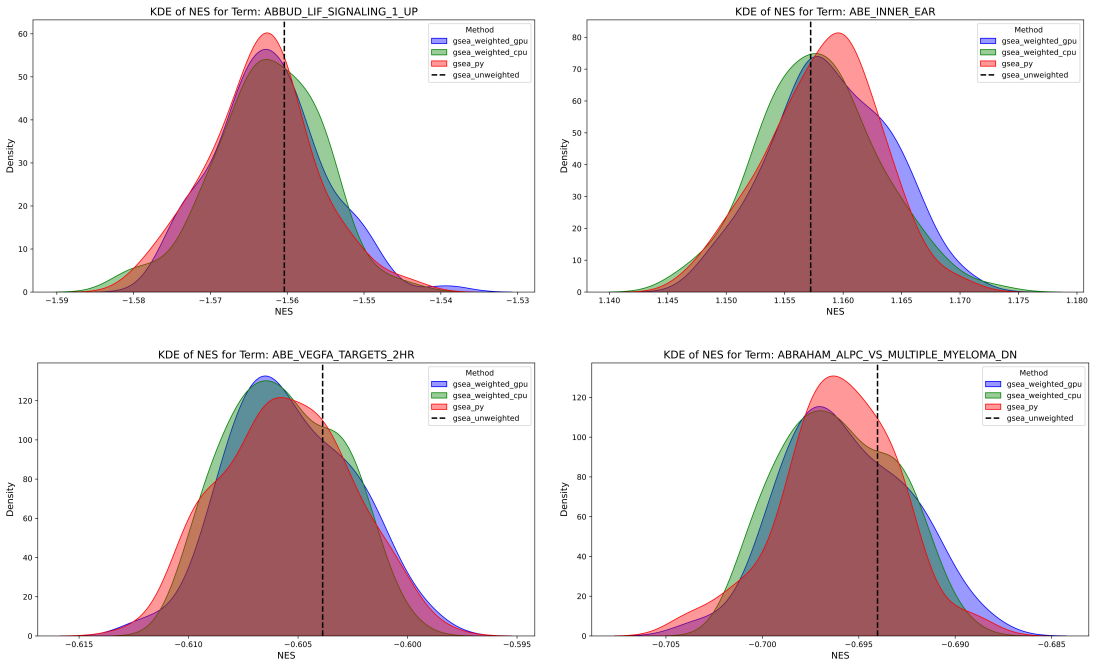

Figure 5: **KDE comparison of enrichment statistics (NES) across random seeds.** Kernel density estimates of NES computed by Bioturing-GSEA (GPU permutation mode) and GSEApy over 100 independent random seeds. Distributions overlap near-perfectly across all gene sets shown, indicating statistical equivalence up to unavoidable finite-permutation approximation

0 regime, it replaces thousands of stochastic phenotype permutations with a fully deterministic, analytically exact computation by exploiting the formal equivalence between the GSEA statistic and the two-sample Kolmogorov–Smirnov test. Through the numerically stable recursive lattice-path algorithm of Viehmann [4], the Pelz–Good asymptotic series of Simard and L'Ecuyer [2], and Vrbik's small-sample correction [5], Bioturing-GSEA achieves near-perfect concordance with 10,000-permutation GSEApy (Pearson $r \geq 0.9998$) while delivering speedups exceeding 8,000×.

For the general weighted case ($p > 0$), the GPU-accelerated permutation engine exploits four complementary strategies to eliminate computational redundancy—massive thread parallelism across independent permutations and gene sets, single shared permutation generation, hit-size-ordered incremental processing, and null-distribution caching by hit count—achieving 10–100× speedup over standard CPU implementations while preserving the exact mathematics of the canonical permutation test [3].

Together, these advances transform GSEA from a batch-oriented offline analysis into a practical, real-time tool. Large-scale, reproducible pathway interpretation is now feasible even on comprehensive databases with tens of thousands of genes and thousands of gene sets, enabling seamless integration into modern single-cell, spatial transcriptomics, and interactive exploratory workflows.

Bioturing-GSEA is openly available and provides exact deterministic results alongside high-performance permutation testing within a single, unified framework.

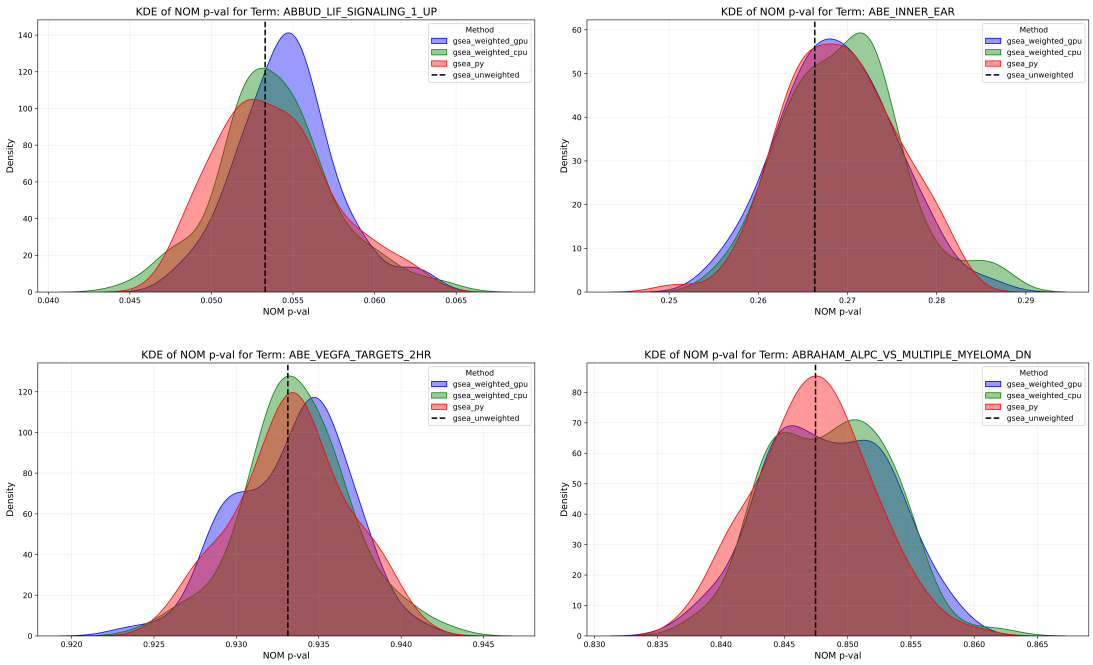

Figure 6: **KDE comparison of nominal p-values across random seeds.** Kernel density estimates of permutation-based nominal p-values from Bioturing-GSEA and GSEApy. Near-perfect overlap across all panels confirms that both implementations draw from the same null distribution; residual discrepancies are attributable solely to finite permutation noise.

References

[1] Hodges, J. L. (1958). The significance probability of the Smirnov two-sample test. *Arkiv för Matematik*, 3(43):469–486.

[2] Simard, R. and L'Ecuyer, P. (2011). Computing the two-sided Kolmogorov–Smirnov distribution. *Journal of Statistical Software*, 39(11):1–18.

[3] Subramanian, A., Tamayo, P., Mootha, V. K., Mukherjee, S., Ebert, B. L., Gillette, M. A., Paulovich, A., Pomeroy, S. L., Golub, T. R., Lander, E. S., and Mesirov, J. P. (2005). Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Sciences USA*, 102(43):15545–15550.

[4] Viehmann, T. (2021). Numerically more stable computation of the p-values for the two-sample Kolmogorov–Smirnov test. *arXiv*:2102.08037.

[5] Vrbik, J. (2018). Small-sample corrections to the Kolmogorov–Smirnov test statistic. *Pioneer Journal of Theoretical and Applied Statistics*, 15(1–2):15–23.