# ICPP23-AMPStencil-rebuttal-full-length

We appreciate all reviewers for their valuable comments and will consider all suggestions to revise the final paper.

### Reviewer 2 & 3

Q1: Further GPU optimizations

A1: AMPStencil can be further accelerated by using shared memory for mesh, overlapping computation and precision conversion, and using different threads for high-precision cells and low-precision cells to achieve load balance between SM. AMPStencil can also use tensor/matrix cores [A] to accelerate low-precision stencil computation.

[A] Toward accelerated stencil computation by adapting tensor core unit on GPU, ICS22

---

### Reviewer 2 & 4

Q1: Limited performance improvement using FP64-FP32

A1: AMPStencil is proposed to target stencil-based applications that are generally memory-bound on GPU. The limited performance speedup when replacing FP64 with

FP32 can hardly improve the performance of stencil-based applications evaluated in our paper (speedup less than 5%), which is proved by the experiment results shown in Table 1. For FP16, the memory access transactions can be merged and stencil's compute intensity is increased. Thus, stencil can obtain significant benefits with FP16.

Q2: Applicable for using average error

A2: For dominance-shifting applications, the state of the simulation result is more important and can be reflected by mean error. For example, grain growth application focus on simulating the interface between two grains[3], which can be reflected by the concentration of each grain.

---

### Reviewer 4 & 5

Q1: Applicability of AMPStencil

A1: The dominance-shifting application is a wide category of applications whose accuracy can be satisfied without calculating the whole mesh in high precision, such as gravitational wave propagation[7], meteorological simulation[14], and materials manufacture[3,23].

---

### Reviewer 1

Q1: Meaning of "Towards" and "AMP"

A1: "Towards" is used to emphasize the process for elaborating our method comprehensively. "AMP" is short for Adaptive Mixed Precision.

Q2: Memory for low-precision data

A2: We keep the data in both high precision (store in *highMesh*) and low precision (store in *lowMesh* in half2). Although this introduces additional storage overhead, it brings performance improvement for precision conversion.

Q3: GRAM number

A3: Table 3 shows the speedup of GRAM to high-precision implementation and precision configuration ($\alpha$) of GRAM.

Q4: Experimental setup for error

A4: We design different magnitudes (from 1E-5 to 1E-10) of error to demonstrate AMPStencil's ability to get better mixed-precision configurations, similar to [4, 10]. In Section 4.4, 1E-3 is the error of FP16 implementation, not for AMPStencil.

Q5: Overhead in Table 4

A5: The overhead is compared to mixed-precision implementation, not high-precision implementation. Although mixed precision introduces extra overhead, AMPStencil can still accelerate the applications (demonstrated by Table 3).

## Reviewer 2

Q1: Overhead of data shuffling

A1: We keep the low-precision data in half2 with data reorganization in the entire execution of the application. We do not re-organize it during runtime and there is no overhead for it. The performance results shown in Table 3 include the overhead of monitor detection and precision conversion. Even with the overhead included, AMPStencil can still achieve higher performance than FP32.

Q2: Error metric for other stencil problems

A2: We use average error for dominance-shifting applications in experiments. Other stencil applications can also use average error if they focus on mesh status, such as thermodynamic simulation.

---

## Reviewer 3

Q1: Comparison with GRAM

A1: GRAM is the SOTA framework that supports data-wise mixed-precision strategy. We have also tried to compare AMPStencil with the variable-based framework ADAPT [16]. However, ADAPT fails to obtain the mixed-precision configurations of the three applications, because it requires too much GPU memory (> 256 GB).

Q2: AMPStencil tailored for memory-bound

A2: AMPStencil targets dominance-shifting applications based on memory-bound stencil. Considering that, AMPStencil does not explore FP64-FP32 and proposes half2 optimization for stencil. AMPStencil can also benefit other applications with computation-bound kernels.

---

### Reviewer 4

Q1: Trade-off between performance and accuracy

A1: AMPStencil search for precision configuration with highest performance with the accuracy constraint given by user. AMPStencil trade-off performance and accuarcy by Monitor.  With lower *cha*, lower *check_duration*, and higher prefetching distance, AMPStecnil can achieve higher accuracy  but lower performance.

Q2: Comparion with APE[ICS22]

A2: APE selects precision strategy only for GEMM operation, while AMPStencil determines precision configuration for whole program with a exponentially large search space.

Q3: Discussion of accuracy and numerical stability

A3: AMPStencil uses the monitor to detect the dominating cells during runtime and applies high precision to the dominating cells, which are the most fundamental factors that affect the numerical stability and accuracy of the dominance-shifting applications. Therefore, if one cell is important to numerical stability and accuracy,

AMPStencil will keep the cell in high precision adaptively during simulation. Similar to existing mixed-precision frameworks [9,10,12,13,16], the numerical stability and accuracy can be guaranteed by the threshold determined offline through empirical study. Since the threshold is determined using representative input, it can be reused in production to provide expected numerical stability and accuracy. However, to study the rigorous mathematical proof is beyond the scope of this paper.

Q4: Theoretical study of mixed precision

A4: We agree with the reviewer that there are orthogonal research directions (e.g., theoretical study on numerical properties) to AMPStencil that altogether make mixed precision techniques both sound and practical when adopted in scientific computing. Particularly, AMPStencil focuses on the practical aspect that proposes computation methods to achieve performance improvement when mixed precision is adopted. The contribution of AMPStencil can complement the theoretical study, because even with theoretical methods available it cannot automatically translate into performance speedup without efficient computation methods such as AMPStencil. On the other hand, the theoretical study when available can guide the mixed precision adjustment of AMPStencil to achieve performance speedup under expected error bound. We believe the importance of the computation methods such as AMPStencil is on par with theoretical study in the field of mixed precision, which should not be overlooked.

Q5: Maximum absolute error and maximum relative error.

A5: When threshold is 0.001, the maximum absolute error is 1E-07 and maximum relative error is 1 for FP64-FP16. The relative error is high because some extremely tiny value (under 1E-07) is zero in mixed-precision result, which does not affect the final simulation results essentially.

---

## Reviewer 5

Q1: Relevance to the conference and multi-GPUs

A1: The topic of Algorithms is "Parallel and Distributed Algorithms". AMPStencil is a mixed precision technique for parallel algorithm 0n GPU. AMPStencil can scale to multiple GPUs by supporting precision conversion of halo across GPUs.

Q2: Applicability of non-stencil application

A2: The key idea of AMPStencil can also extend to non-stencil application whose accuracy can be satisfied without calculating the whole program in high precision (such as unstructure grid method) with customized Monitor and precision conversion.