MP CBM-Z V1.0: design for a new CBM-Z gas-phase chemical mechanism architecture for next generation processors

Hui Wang[1], Junmin Lin[2], Qizhong Wu[1], Huansheng Chen[3], Xiao Tang[3], Zifa Wang[3], Xueshun Chen[3], Huaqiong Cheng[1], Lanning Wang[1]

[1]College of Global Change and Earth System Science, Joint Center for Global Changes Studies, Beijing Normal University, Beijing 100875, China
[2]Intel (China) Corporation, Beijing 100013, China
[3]State Key Laboratory of Atmospheric Boundary Layer Physics and Atmospheric Chemistry, Institute of Atmospheric Physics, Chinese Academy of Sciences, Beijing 100029, China

Correspondence to: Qizhong Wu (wqizhong@bnu.edu.cn) & Huansheng Chen(chenhuansheng@mail.iap.ac.cn)

Abstract. Precise and rapid air quality simulation and forecasting are limited by the computational performance of the air quality model used, and the gas-phase chemistry module is the most time-consuming function in the air quality model. In this study, we designed a new framework for the widely used Carbon Bond Mechanism Z (CBM-Z) gas-phase chemical kinetics kernel to adapt the Single Instruction Multiple Data (SIMD) technology in the next-generation processors for improving its calculation performance. The optimization implements the fine-grain level parallelization of CBM-Z by improving its vectorization ability. Through constructing loops and integrating the main branches, e.g. diverse chemistry sub-schemes, multiple spatial points in the model can be operated simultaneously on vector processing units (VPU). Two generation CPUs, Intel Xeon E5-2680 V4 CPU and Intel Xeon Gold 6132, and Intel Xeon Phi 7250 Knights Landing (KNL) are used as the benchmark processors. The validation of the CBM-Z module outputs indicates that the relative bias reaches maximum 0.025% after 10-h integration with -fp-model fast=1 compile flag. The results of module test show that the MP CBM-Z resulted in 5.16x and 8.97x speedup on a single core of Intel Xeon E5-2680 V4 and Intel Xeon Gold 6132 CPUs, respectively, and KNL got a speedup of 3.69x comparing with performance of old CBM-Z on Intel Xeon E5-2680 V4 platform. For the node, the speedup on the two generations CPUs can reach 104.63x and 198.50x using Message Passing Interface (MPI) and 101.02x and 194.60x using OpenMP, respectively, and the speedup on the KNL node can reach 194.60x using MPI and 167.45x using OpenMP. The speedup of the optimized CBM-Z is approximately 40% higher on a 1-socket KNL platform than on a 2-socket Broadwell platform and about 13~16% lower than on a 2-socket Skylake platform. We also tested a three-dimension chemistry transport model (CTM) named Nested Air Quality Prediction Model System (NAQPMS) equipped with MP CBM-Z. The tests illustrate an obvious improvement on performance for CTM after adopting the MP CBM-Z. The results show that MP CBM-Z leads to 3.32x and 1.96x for gas-phase chemistry module and the CTM on Intel Xeon E5-2680 platform. Moreover, on the new Intel Xeon Gold 6132 platform, the MP CBM-Z gains 4.90x and 2.22x speedups for gas-phase chemistry module and the whole CTM. For the KNL, MP CBM-Z enables a 3.52x speedup, but whole model lost 24.10% performance comparing with CPU platform due to bad performance of other modules. In addition, since this optimization seeks to improve the utilization

of the VPU, the model is more suitable for the new generation processors adopting the more advanced SIMD technology. The results of our tests already show that the profit of updating CPU improved about 47% by using MP CBM-Z since the optimized codes has better adaptability for new hardware. This work improves the performance of the CBM-Z chemical kinetics kernel as well as the calculation efficiency of the air quality model, which can directly improve the practical value of the air quality model in scientific simulation and routine forecasting.

1 Introduction

Air pollution and its impacts on human health have attracted widespread attention all over the world, especially in developing countries (Gurjar et al., 2016;Zhang et al., 2017). As a useful tool for air quality problems, the Chemistry Transport Models (CTM) is widely used in studies of air quality (Gao et al., 2016;Chen et al., 2015;Wu et al., 2014) and in establishing air quality forecasting (AQF) systems. As the core of the AQF system, a CTM requires a large number of computational resources to simulate the complex chemical and physical processes. To satisfy the demand of routine air quality forecasting in a timely manner, coarse spatial resolution and relatively simple processes are adopted in CTMs to minimize the use of computational resources. Meanwhile, the corresponding simulation studies with complicated processes are also limited by computational resources. Therefore, air quality simulation studies can benefit significantly by improving the performance of the CTM used. In the CTM, the most time-consuming module is the gas-phase chemistry module (Wang et al., 2017). The gas-phase chemistry module is described as a system of ordinary differential equations (ODEs) to simulate the chemical kinetics of trace gases in an atmosphere model (Seinfeld and Pandis, 2012). Linford (2013) reported that the Regional Acid Deposition Model version 2 (RADM2) (Zimmermann and Poppe, 1994;Chang et al., 1987), a chemical kinetics kernel, accounted for 90% of the computational time in the Weather Forecasting and Research/Chemistry (WRF-chem) model (Grell et al., 2005). Another widely used chemical kinetics kernel, Carbon Bond Mechanism version Z (CBM-Z) (Zaveri and Peters, 1999), accounts for approximately 68% of the computation time in the Global Nested Air Quality Prediction Model System (GNAQPMS) model(Chen et al., 2015;Wang et al., 2017). Therefore, accelerating the gas-phase chemistry module can directly improve the performance of the CTM as well as the whole AQF system. The AQF system can also benefit from the performance improvement by adopting a higher model resolution and improving the frequency of air quality forecasting.

The performance of models improves with updated hardware. However, reaching the bottleneck of power density and the thermal limitation of the silicon technology for a single-core design, frequent updating has not been an efficient way to improve the scientific model's performance. Additionally, multi-core architecture and a heterogeneous computing architecture such as a Many Integrated Core (MIC) and a Graphic Processing Unit (GPU) have become the hardware trend for high-performance computing (Xu et al., 2015;Lawrence et al., 2018). Meanwhile, to take full advantage of the advanced features of new processor architecture, the applications or the models must be redesigned or rewritten. Xu et al. (2015) rewrote the Princeton Ocean Model (POM) using CUDA-C to port it from a CPU to a GPU platform. Linford (2013) also tried to solve the computation bottleneck of RADM2 mentioned above by using a heterogeneous platform such as GPU-CPU. In addition, our previous work

showed the primary optimizations we performed to accelerate the GNAQPMS model on the new generation CPU and Intel MIC platforms (Knights Landing, KNL(Sodani et al., 2016)), and had a significant performance improvement on both platforms, a 2.77x speedup on CPU, and a 3.51x speedup on the KNL node (Wang et al., 2017). In this study, we redesign the code structure of the chemical kinetics kernel CBM-Z to improve its vectorization performance on the CPU and KNL platforms, which significantly improves its performance by fully utilizing the Single Instruction Multiple Data (SIMD) technology. We tested the performances of this optimized CBM-Z module as well as a regional CTM equipped with it, and the codes only contained this single module make it easier to let the CTM developers to reuse the codes.

Section 2.1 in this paper introduces the detail of CBM-Z scheme, and section 2.2 describes the new architecture we designed for CBM-Z. Since multiple spatial points were operated simultaneous in the optimized CBM-Z scheme, the optimized CBM-Z schemes were called Multiple-Points CBM-Z Version 1.0 (MP CBM-Z V1.0). In section 3.1, we present our benchmark platforms. In section 3.2 and 3.3, we introduced the test cases and the test results of single model tests and CTM tests separately. The conclusions and discussions are given in section 4.

2 Method Description

CBM-Z is a core module in CTMs that simulates the complicated gas-phase chemical processes in the atmosphere. In this module, too many branches and unbalanced calculations within the model grids make it a challenge to improve its performance on a vectorization level. This leads to the low performance of CBM-Z on the new generation processors that are highly dependent on powerful vector processing units (VPU). In our previous work, we conducted several optimizations on CBM-Z to enhance its vectorization and parallel performance(Wang et al., 2017). In this work, we attempt to further enhance its vector calculation ability by constructing a new structure, which makes the CBM-Z module suitable to be vectorized. The CBM-Z module was extracted as an individual box model to test its performance and improve code reusability.

2.1 Description of CBM-Z

CBM-Z is a lumped-structure photochemical mechanism that was developed to meet the needs of city-scale to global-scale tropospheric chemical simulation (Zaveri and Peters, 1999). The original scheme contains 67 species and 132 reactions. CBM-Z has been widely used in CTMs, e.g., the WRF-Chem (San José et al., 2015), the Nested Air Quality Prediction Model System (NAQPMS) (Wang et al., 2001) and the GNAQPMS model. In the NAQPMS and GNAQPMS model, CBM-Z was further modified by Li et al. (2012). It was updated to 76 species, and 28 heterogeneous reactions were added. The CBM-Z solver uses the Modified Backward Euler (MBE) solver developed by Feng et al. (2015), a faster and more robust algorithm which overcomes inflexibility and preserves the non-negativity.

The main control flow of CBM-Z is shown in Figure 1. The ***IntegrateChemistry*** function is treated as the core-function of the module. CBM-Z contains five chemistry sub-schemes. They are the Common Chemistry Scheme (COM), the Urban Chemistry Scheme (URB), the Biogenic Chemistry Scheme (BIO), the Marine Chemistry Scheme (MAR) and the Heterogeneous

Chemistry Scheme (HET). The integration of different sub-schemes is used to satisfy the simulation of diverse scenarios and scales. The combination of sub-schemes relies on the concentration and emission of each chemical species in the specific model grid, which is implemented in the ***SelectGasRegime*** function. The variable ***iregime*** stores the return-value of ***SelectGasRegime*** and controls the subsequent calculation processes of CBM-Z. The possible values and the sub-schemes represented are shown in Table 1. The combinations include the COM and HET schemes, while other schemes are added when the concentration or emission of a corresponding species in a certain scheme are greater than zero. Compared with the algorithm computing all chemical interactions, this algorithm is helpful in saving the computation resources on a simple core, while such irregular and unbalanced calculations lack well-structured loops and impede the vectorization of codes. Besides the chemistry sub-schemes mentioned above, CBM-Z uses the other functional branches, e.g. nocturnal and diurnal chemistry, and they impede the vectorization of the computation. The CBM-Z also contains multiple unconstructed scalar operations. We partially integrated the scalar operations by using indirect indexes to construct loops for vectorization (Wang et al., 2017). However, this method required significant efforts, and it only reconstructed a limited number of scalar operations. CBM-Z module still contains many scalar operations. With multi-level control flow divergences and many scalar calculations, it is not feasible to perform automatic vectorization with an Intel compiler.

Fortunately, contiguous model grids may have similar chemical processes in air quality simulations, which provides the opportunity to integrate the grids with similar or the same chemical processes to implement vectorization to calculate the processes of multiple grids simultaneously. The following section introduces the details about integrating the chemistry sub-schemes to implement the vectorization.

2.2 Algorithm Description

The new generation Intel CPU (e.g., Skylake) and Intel MIC chips are equipped with the AVX-512 or more advanced vectorization instructions, which supports a maximum of eight double precision and sixteen single precision operations with 512-bit-wide vector registers. It is critical to peak performance of the next generation CPUs and MICs to fully reach the potential of the AVX-512 (Mielikainen et al., 2014). As mentioned in section 2.1, the automatic vectorization using a compiler is impeded by the features of CBM-Z, and the common manual measures including constructing loops, avoiding the loop/data dependence and aligning the data with directives need to further vectorize CBM-Z. On the other hand, to implement the vectorization of the module, the general design allowed the CBM-Z module to handle multiple grids in one citing cycle, and the functions in CBM-Z were reconstructed by adding a regular loop for these grids. Subsequently, these loops can be vectorized to implement the fine-grained parallelization on a VPU.

All of the model grids are distributed to multiple cores using a Message Passing Interface (MPI) and OpenMP, which is a type of coarse-grain parallelization. Our goal is to implement fine-grained parallelization based on the SIMD and the grids that are distributed to a specific processor operate in parallel using the VPUs on each core. As shown in Figure 2, the calling method of the CBM-Z module changes from calculating one model grid calculation at a time to multiple model grids at the same time. The step length (VLEN in Figure 2) of the loops represents the number of the grids operated simultaneously, and it is

determined by the length of the vector register. The VLEN was set to 16 since the 512-bit-wide vector of AVX-512 can support 16 single-precision operations at the same time. Using this framework, the functions in CBM-Z construct an extra loop to manage the point number dimension, and the corresponding variables require an extra dimension to store the information of multiple grids. Using the structure with an extra loop, it was easier to implement the vectorization. Meanwhile, to avoid multiple remaining points which cannot satisfy the VLEN, we set a common variable array, ***pmask***(VLEN) as shown in Figure 2, to store the availability label of the model grids. When the number of remaining grids did not reach VLEN, the corresponding ***pmask*** value of excessive grids was set to False to mask these grids in the calculation. Furthermore, the latitude and longitude dimensions loop were merged, from nested loops to a single loop, to reduce the number of unavailable points as shown in Figure 2. Achieving such a large-scale vectorization also requires the balance of the calculation processes, but the calculation branches in CBM-Z are an obstacle to balancing the calculation. Therefore, the branches in CBM-Z should be taken into consideration in constructing the loops, especially the chemical schemes chosen in Table 1. As mentioned in section 2.1, the contiguous model grids may have similar chemical processes in the atmosphere. This provides an opportunity to integrate the sub-schemes by masking the heterogeneous model grids, and this type of masking operation can be used in the functions ***GasRateConstants*** and ***ODEsolver*** (Figure 1). Figure 3 shows the flowchart for masking the model grids to satisfy the vectorization of the grid array. The VLEN number grids contain an array to perform the operation simultaneously, and the variable ***pmask*** signed the valid grids. Meanwhile, the variable ***iregime*** described in Table 1 and representing the combination of sub-schemes, is used to determine whether the model grid must perform the subsequent operation or not. The grids with the same property or calculation are kept by setting the variable ***bmask*** to True. The COM and HET schemes are common for all grids, and the mask operation for COM and HET schemes only determine the availability of the grids. As shown in Figure 3, for the URB, BIO and MAR schemes, the ***iregime*** value and ***pmask*** are both used to filter the heterogeneous grids and the ***bmask*** stores the results. To improve the efficiency of vectorization, the ***bmask*** does not prevent the calculation of heterogeneous grids but prevents the calculation results from copying back to the return value. Thus, all computations are performed on all grids, but only the results of the valid grids are returned. This improves the utilization of data as well as the efficiency of vectorization. Because of the independence of the grids, the computation process of VLEN arrays are independent and satisfy the requirement of vectorization, and the corresponding directives were added to declare the independence of the arrays and force the compiler to perform the data alignment and vectorization after the reconstruction of the codes. Overall, by constructing the loops, the computations of the independent grids were integrated with the fine-level parallel implementation through the SIMD. In addition, the efficiency of such algorithms is linearly improved with the development of the width of the vector in the VPU.

3 Test Results

The validation and evaluation of the improvement of the new method were conducted using the box model of CBM-Z as well as a regional named CTM Nested Air Quality Prediction Model System (NAQPMS) model with optimized CBM-Z scheme.

We tested the theoretical performance of vectorization by using the box model, and the CTM tests illustrate its potential in the real scenes.

3.1 Benchmark Platform Description

The computation cluster for tests was provided by Institute of Atmospheric Physic (IAP), Chinese Academy of Sciences (CAS). The CPU and KNL platforms were used for testing the codes. The CPU platforms in this study include two generation CPUs, 2-socket CPU nodes with Broadwell architecture 2.4 GHz 14-core Intel Xeon E5-2680 V4 processors and 2-socket CPU nodes with 2.6 GHz Skylake architecture 14-core Intel Xeon Gold 6132. To the vector instructions, the previous generation of V4 adopted the AVX-2 vector instructions and the new generation used the AVX-512 vector instructions. The AVX-512 and AVX-2 instructions support 16 and 8 single precision floating-point calculations simultaneously, respectively. Comparing the two generation CPUs helped to present the potential of new MP CBM-Z to fully use the development of hardware. The KNL node contained one 1.40 GHz 68-core Intel Xeon Phi 7250 processor, which also adoptd the AVX512 vector instructions. The operating system was Red Hat release 4.8.5-16 for all platforms. The codes were all compiled using the Intel FORTRAN Compiler 2017 update 4, and the compile flags about vectorization and float-pointing accuracy control of CBM-Z module and NAQPMS model are shown in Table 2 and Table 3, respectively. The corresponding flags for vectorization (e.g., –xCORE-AVX2, -xCOMMON-AVX512, -xMIC-AVX512, -align array64byte) were adopted. We also tested the codes using diverse options of compile flag -fp-model, which control the balance between accuracy and performance of floating-point calculation, to investigate its impact on codes. We mainly discussed the two options of -fp-model ***precise*** and -fp-model ***fast=1***. The ***fast=1*** is the default option when -fp-model flag is not designated. Comparing with the option ***precise,*** the ***fast=1*** would improve the performance but decline the accuracy of floating-point calculation. Using ***precise*** would keep value safe and force the compiler avoid vectorization of some calculation to improve the accuracy. We compared the results of two options, including the outputs and the performance of models, to investigate its impact and discuss the suitable choice of compile flag.

3.2 Box Model Test

The box model of MP CBM-Z was used to validate the model outputs and investigate the ideal parallel performance of single module. We also tested the results under the different parallel frames, e.g. MPI and OpenMP. Each test was repeated for 10 times to isolate the impact of the random state of platforms.

3.2.1 Test Case Description

There are two cases were used for the box model of CBM-Z. One was a 10-h single point case with all species to validate the outputs of the model, and the other was a 1-hour simulation with 160*148*20 grids to test the performance of this module under the ideal scenario. The initial values of the single point case were showed in Table S1. The meteorological conditions were constant and emissions were set to zero to test the error of the algorithms. The time step-size was 5 seconds for two cases.

The case used for validating output the simulation results every 5 minutes, while the performance test did not open the output function to isolate the disturbance of I/O. The different compiling flags for precisions of floating-point calculation were presented in the Table 2. We tested the baseline and optimized model on two different platforms of CPU and KNL, and the time of the computation portion was counted using the ***system_clock*** function.

3.2.2 Results Validation of box model

The test case for validation was a one-point simulation including all reactions and species. The simulation results of all species were output every five minutes with the meteorological conditions updated. We validated the chemical species including ozone (O_3), nitrogen dioxide (NO_2), nitrogen monoxide (NO), hydrogen peroxide (H_2O_2), sulfur dioxide (SO_2), sulfuric acid (H_2SO_4), hydroxyl (OH) radical, hydroperoxyl (HO_2) radical, and alkyl peroxy (RO_2) radical. These species are relatively representative and suitable for validating whether the optimization significantly changed the simulated results or not. Figure 4 show the time-series of the simulated concentrations of the species above from the baseline (base) and optimized (opt) version model with ***precise*** and ***fast=1*** compile flags. The results by the baseline codes with ***precise*** compile flag is as the benchmark, and there is no difference between the results from baseline and optimized codes with same ***precise*** compile flag. The ***precise*** compile flag is a relative safe compile flag and would prohibit the optimizations that would affect the accuracy. The ***fast=1*** compile flag would lead to error even with the same codes, but the relative error (RE) of the baseline codes with ***fast=1*** compile flag is extremely small (<0.0002%). As stated in Figure 4, with the optimized codes, the ***fast=1*** compile flag induced about maximum 0.025% RE for NO and NO_2 at the end of simulation. We can find that the error brought by ***fast=1*** compile flag did not become obvious for the species with small magnitude like OH and RO_2. We would further discuss the impact of ***fast=1*** compile flag in the Section 3.3.2 under a real simulation scenario.

3.2.2 Performance Test of box model

The case with 160*148*20 grids was used to test the ideal performance. Both the baseline and the optimized version of CBM-Z contained the same 76 species. The computational time of the baseline version on a single core of E5-2680 V4 CPU with ***precise*** compile flag was considered as the benchmark time to evaluate the speedups. The tests were done with two generations CPUs and KNL.

The option of -fp-model could directly affect the performance. As shown in Table 4, the benchmark performance was 1014.67 seconds on the E5-2680 V4 platform. By using the new platform with Intel Gold 6132, the baseline version codes gets 1.52x speedup with ***precise*** compile flag. The ***fast=1*** compile flag leads to 1.28x and 2.04x speedups for baseline codes on two generation CPUs. Meanwhile, updating CPU enable the original CBM-Z module gains about 1.52 and 1.59 times acceleration with ***precise*** and ***fast=1*** compile flags, respectively.

MP CBM-Z module shows good performance on both two generation CPUs. On the E5-2680 V4 CPU with old architecture of Broadwell, the optimized codes with two different compile flags consumed 581.14 seconds and 153.32 seconds, respectively; meanwhile, the speedups reach 1.75x and 6.62x compared with the benchmark performance, respectively. Regarding to Intel

Gold 6132 platform, the optimized version CBM-Z consumed 352.00 seconds and 55.42 seconds with ***precise*** and ***fast=1*** compile flags, respectively. Compared with the benchmark time, the speedups could reach 2.88x and 18.31x with impact of diverse compile flags. By using the same ***fast=1*** compile option, the MP CBM-Z shows 5.16x and 8.97x speedups on two generations of CPU compared with original CBM-Z codes.

The results also illustrate that the optimized codes could better utilize the updating of cores through good vectorization ability compared with old codes. Comparing the performance of optimized codes on two generations codes, we can find that updating CPU generation could lead to about 1.65 times and 2.76 times acceleration with ***precise*** and ***fast=1*** compile flags, respectively, which is higher than the 1.5 times of baselines codes.

Compile flags largely affect the performance of codes on KNL. On the Xeon Phi 7250 platform, the optimized codes consumed 3454.9 seconds to finish the single core with ***precise*** compile flag since majority of vectorizations were forbidden, and it's even slower than the benchmark performance; while it only consumed 214.09 seconds and gains 4.74x speedup with ***fast=1*** compile flag. Comparing with old CBM-Z on Intel Xeon E5-2680 V4, KNL gains a 3.69x speedup with MP CBM-Z.

In addition, baseline and optimized codes with ***fast=1*** were also analyzed by using the HPC Performance Characterization from the Intel VTune tools on the CPU platform. On Intel Gold 6132 platform, the single precision giga-floating point operations calculated per second (GFLOPS) increased from 4.81 to 21.37 comparing with original CBM-Z module, and the vector capacity usage improved from 14.3% in the baseline CBM-Z to 89.4% in MP CBM-Z, which implies that majority floating-point instructions in CBM-Z were vectorized.

In addition, we also tested the parallel version of MP CBM-Z compiling by ***fast=1*** option with the MPI and OpenMP separately. We valued the speedups based on the performance of old CBM-Z on Intel Xeon E5-2680 V4 platform with ***fast=1*** option. The results were showed in Table 5, and the MPI and OpenMP version of CBM-Z had 104.63x speedup and 101.02x speedup on the Intel Xeon E5-2680 V4 platform. On the new Intel Xeon Gold 6132, the MP CBM-Z got a speedup of 198.50x and 194.6x for MPI and OpenMP. For the KNL, the speedup reached 175.23x by using MPI and 167.45x by using OpenMP, which was approximately 40% faster than those on the 2-socket Broadwell platform with AVX2 vectorization instruction and about 13~16% slower than those on the 2-socket Skylake platform with same AVX512 vectorization instruction. The combination of the fine-grain vectorization and the coarse-grain parallelization of OpenMP/MPI resulted in a significant performance improvement on the new generation processors. The enhancement of the vectorization performance may be the key to fully using the new generation processors equipped with advanced and wider vectors, which can be significant to fully use the new MIC architecture processors such as KNL.

3.3 CTM Test

The regional CTM, Nested Air Quality Prediction Model System (NAQPMS) model (Wang et al., 2001;ZiFa et al., 2006), was used to test the MP CBM-Z module in real scenarios. The NAQPMS model with diverse version CBM-Z codes were compiled and tested to get the insight of pragmatic value of MP CBM-Z.

3.3.1 CTM and Test Case Description

NAQPMS is a regional CTM developed by IAP, CAS (Li et al., 2011;Li et al., 2013), and has been widely used in air quality research (Wang et al., 2018) and routine air quality forecasting (Wu et al., 2010;Chen et al., 2013). NAQPMS involves all essential processes including diffusion, advection, dry and wet deposition, multi-phase chemistry reactions. More details can be found in Li et al. (2013). Same as the box model, NAQPMS model with baseline and optimized CBM-Z were compiled with diverse compiling flags shown in Table 3.

The test case is a 72-h simulation covered East Asia region. The horizontal resolution of grids is 15 km. The model adopted 20 vertical layers. The meteorological fields driven the NAQPMS model were provided by the Weather Research and Forecasting (WRF) model (Skamarock et al., 2008). The anthropogenic emission inventory was from the Hemispheric Transport of Air Pollution (HTAP) V2 and the organic emission inventory was provided by results from Sindelarova et al. (2014) using the Model of Emissions of Gases and Aerosols from Nature (MEGAN) (Guenther et al., 2006;Guenther et al., 2012). The simulation started at 00:00 UTC, August 17, 2015 and ended in 00:00 UTC, August 20, 2015. We only used one node for testing to exclude the interference of new-work communication. Each experiment was repeated 5 times and the calculation of performance was based on the average value of experiments.

3.3.2 Results Validation of CTM

We chose four major gas pollutants includes NO_2, O_3, SO_2 and CO after 72h integration or simulation at 00:00 UTC, August 20, 2015 to valid the optimized codes. The simulation results of old NAQPMS codes compiled by ***precise*** flag were as the benchmark results, and we mainly compared the simulation results of old NAQPMS codes with ***fast=1*** flag and optimized NAQPMS with ***precise*** and ***fast=1***, respectively.

Figure 5 and Figure 6 present the spatial distributions of NO_2, O_3, SO_2 and CO as well as the absolute errors (AEs) of concentration of these species from other experiments. We can find that all model results show same reasonable spatial distribution of pollutants. In general, for NO_2, O_3 and SO_2, the AEs of majority grids are in the range of ± 0.02 ppbv of other three experiments; for CO, the AEs of old and optimized NAQPMS with same ***fast=1*** out that range shows the more obvious AEs.

The ***precise*** options enable the results of two version codes to be more consistent. Figure 7 shows the distribution of AEs and relative error (REs) for four species in near surface model layer. For majority points, the AEs and REs are in a relatively small range. However, some points existed the exceptional obvious errors. The maximum AEs for NO_2, O_3, SO_2 and CO are 0.166, 0.197, 0.001 and 0.03 ppb over the whole map after 72 integration, and ***fast=1*** option shows more obvious error for both version codes. For old NAQPMS codes, using ***fast=1*** leads to maximum 0.23, 4.5, 0.17 and 2.6 ppbv AEs for NO_2, O_3, SO_2 and CO. To NAQPMS with MP CBM-Z, using the ***fast=1*** option leads to maximum 0.13, 0.93, 0.76 and 0.64 ppbv AEs for NO_2, O_3, SO_2 and CO over the whole map, which is relative better than the old NAQPMS.

Besides of considering the accuracy mentioned above, the determination of -fp-model option should also takes its impact on the performance in to account. In some pragmatic applications like routine air quality prediction, in an acceptable range, sacrificing the accuracy to gain is more reasonable. Conversely, the applications like long term climate simulation, choosing more value-safe compile flags or adopting double-precision for calculation should be required to avoid accumulation of errors.

3.3.3 Performance tests of CTM

The performance tests of old NAQPMS illustrate the improvement of performance led by the platform update and changing the compile flags. The performance of old NAQPMS with ***precise*** was as the benchmark for comparison with other tests. As showed in Table 6, in the original version NAQPMS, the CBM-Z module accounts for 72.26% wall-time of whole simulation. Changing the compile option of -fp-model to trade performance by sacrificing accuracy would lead to 1.34x and 1.25x speedups for the module CBM-Z and the whole model on Intel Xeon E5-2680 platform, respectively. By updating the CPU from Intel Xeon E5-2680 to Intel Xeon Gold 6132, the module CBM-Z and whole model gains 1.28x and 1.29x speedups, respectively. The speedups incline to 1.68x and 1.58x for CBM-Z and the whole model, respectively, by using ***fast=1*** compile flag on Xeon Gold 6132 platform. The benefit from updating hardware is limited with the old codes, which implying the need of optimizing codes to adapt to the new hardware features.

The performances of gas-phase chemistry module and the NAQPMS model are largely improved after adopting MP CBM-Z described in this paper. As showed in Table 6, the CBM-Z model and the whole NAQPMS model gets 1.59x and 1.40x speedups on the old Xeon E5-2680 platform with same ***precise*** compile flag, and the speedups are improved to 4.45x and 2.44x by using the ***fast=1*** compile flag. With same ***fast=1*** flag, MP CBM-Z showed 3.32 and 1.96 times acceleration compared with old CBM-Z for gas-phase chemistry module and whole NAQPMS model. Such results illustrate that the optimization for vectorization releases the potential of existing hardware, the performance is highly improved even with the relative strict ***precise*** compile flag, which prevents most vectorizations.

The new generation CPU further enforced the performance of MP CBM-Z. Using the platform with new generation processor Xeon Gold 6132, the speedups could reach 1.84x and 1.66x for the CBM-Z and the NAQPMS model with ***precise*** compile flag, respectively, and adopting ***fast=1*** compile flag improves the speedups to 8.22x and 3.50x compared with benchmark performance. On the same Xeon Gold 6132 platform with ***fast=1*** compile flag, MP CBM-Z gains 3.32 and 2.22 times acceleration compared with old CBM-Z for gas-phase chemistry module and whole NAQPMS model. Moreover, the time-consuming proportion of gas-phase chemistry declined to 30.74%, which is largely lower than that of 72.26% in baseline version model.

In addition, MP CBM-Z extended the benefit gained from advanced hardware. Using the same fast=1 compile option, the performance of old CBM-Z on AVX-512 platform is about 1.25 times of that on AVX-2 platform, and the performance of MP CBM-Z is about 1.84 times of that on AVX-2 platform. The efficiency of using new CPUs improved about 47% by adopting MP CBM-Z. Therefore, enhancing the vectorization ability of codes ensures that the applications, like CTM in this paper, could further utilize the improvement of processors on vectorization in the future.

KNL are more relied on SIMD to improve its performance according to the test results. The CBM-Z module is accelerated on KNL with a speedup of 5.9x, but the whole model only got a 1.27 times acceleration compared with benchmark performance. Comparing old CBM-Z on Intel Xeon Gold 6132 platform, MP CBM-Z gets a speedup of 3.52x for gas-phase chemistry on KNL, however, the performance of whole model declined 24%. Therefore, MP CBM-Z large improved the efficiency of CBM-Z on KNL by improving its vectorization ability, but further optimizations were required to let the whole CTM adapt to the architecture of KNL.

4 Conclusion and Discussion

A new framework was designed for helping the chemical kinetics kernel CBM-Z to adapt to the next generation processes by improving its vectorization. Through packing the multiple spatial points, the optimized CBM-Z module handled these grids simultaneously. The functions in the original CBM-Z were reconstructed with loops, which provided the opportunity to implement the fine-grain level parallelization of vectorization. Meanwhile, we masked the heterogeneous grids to integrate the chemistry sub-schemes in the CBM-Z to perform the calculation of multiple grids simultaneously. Since the contiguous grids had similar chemistry processes, the impact of these process on performance was largely limited, and the codes were highly vectorized.

The computation cluster equipped with the two generations CPUs (Intel Xeon E5-2680 V4 and Intel Xeon Gold 6132) and KNL (Intel Xeon Phi 7250) provided by IAP, CAS, were used to test the performance respectively. We tested the codes with two different compile options of -fp-model ***precise*** and -fp-model ***fast=1*** to present its impact on accuracy of single-precision computation and performance. The validation test ensured the reliability of our optimization on the model results, and the discrepancies of all diagnostic chemical species caused by single float were lower than about 0.025% after 10-h integration with ***fast=1*** option. Based on the HPC Performance Characteristic from the Intel Vtune tools on the Intel Xeon Gold 6132, the GFLOPS of CBM-Z increased from 4.81 to 21.37, and the vector capacity usage improved from 14.13% in baseline CBM-Z to 89.4% in the optimized CBM-Z.

The tests using the single core showed that the vectorization optimization led to 5.16x and 8.97x speedups on Intel Xeon E5-2680 V4 and Intel Xeon Gold 6132 CPUs, respectively, and KNL gets a speedup of 3.69x comparing with performance of old CBM-Z on Intel Xeon E5-2680 V4 platform. It highlights the importance of vectorization to the KNL platform. Meanwhile, we also tested the MPI and OpenMP version CBM-Z. For the node, the speedup on the two generations CPUs can reach 104.63x and 198.50x using Message Passing Interface (MPI) and 101.02x and 194.60x using OpenMP, respectively, and the speedup on the KNL node can reach 194.60x using MPI and 167.45x using OpenMP. The speedup on the KNL node can reach 194.60x using MPI and 167.45x using OpenMP. The speedup of the optimized CBM-Z is approximately 40% higher on a 1-socket KNL platform than on a 2-socket Broadwell platform and about 13~16% lower than on a 2-socket Skylake platform.

The regional CTM NAQPMS were used to test the practical improvement of the MP CBM-Z in the real situation. The old and optimized codes of NAQPMS compiled with ***precise*** and ***fast=1*** options, respectively, were tested the on diverse platforms.

The outputs of models after 72-h simulation were used to validate and present the error by the codes as well as compile flags. The difference between old and optimized codes are generally in the range of ± 0.02 ppbv using ***precise*** option. The maximum over the whole map is about 0.166, 0.197, 0.001 and 0.03 ppbv for NO_2, O_3, SO_2 and CO. The ***fast=1*** option leads to more obvious error; however, performance could benefit a lot through adopting this option.

The choice of -fp-model compile flag decides the balance between the accuracy and performance. According to our test, after using the fast=1 option, the performance of codes would largely be improved but sacrificing some accuracy even using same codes. However, the loss of accuracy is relatively small, and in some practical applications that do not require high accuracy of floating-point calculation, it's acceptable to using ***fast=1*** option to using the codes.

Besides the CBM-Z chemical scheme, this algorithm is also suitable for models with a similar code structure to improve its vectorization. In addition, in this study, CBM-Z was treated as an example to describe this simple optimization strategy to implement the optimization on new generation processors, which emphasizes the importance of vectorization. However, some specific strategies should also be considered before adoption. The optimizing methods such as constructing loops from the discrete scalar calculations as described in Wang et al. (2017), would diminish the readability of the source code by using a mediate or indirect index and could cause problems to the following developers. Therefore, it is essential to write annotations when the developers are doing the optimization. Maintaining the original code and controlling the compile process allow the next developer to understand the code.

Code Availability.

The source code of the baseline and optimized version CBM-Z box model, including OpenMP and MPI versions, is available online via ZENODO (https://doi.org/10.5281/zenodo.1161576).

Acknowledgements.

The National Key R&D Program of China (2017YFC0209805 and 2016YFB0200800), the CAS Information Technology Program (XXH13506-302), the National Natural Science Foundation of China (41305121) and the Fundamental Research Funds for the Central Universities funded this work. The authors would like to thank Institute of Atmospheric Physics and Intel Corporation's Software Support Group (SSG) for providing the high-performance computing (HPC) environment and technical supports.

References

Chang, J. S., Brost, R. A., Isaksen, I. S. A., Madronich, S., Middleton, P., Stockwell, W. R., and Walcek, C. J.: A three-dimensional Eulerian acid deposition model: Physical concepts and formulation, Journal of Geophysical Research Atmospheres, 92, 14681-14700, 1987.

Chen, H., Wang, Z., Qizhong, W. U., Jianbin, W. U., Yan, P., Tang, X., Zhe, and Wang: Application of Air Quality Multi-Model Forecast System in Guangzhou: Model Description and Evaluation of PM10 Forecast Performance, Climatic & Environmental Research, 18, 427-435, 2013.

Chen, H. S., Wang, Z. F., Li, J., Tang, X., Ge, B. Z., Wu, X. L., Wild, O., and Carmichael, G. R.: GNAQPMS-Hg v1.0, a global nested atmospheric mercury transport model: model description, evaluation and application to trans-boundary transport of Chinese anthropogenic emissions, Geosci. Model Dev., 8, 2857-2876, 10.5194/gmd-8-2857-2015, 2015.

Feng, F., Wang, Z., Li, J., and Carmichael, G. R.: A nonnegativity preserved efficient algorithm for atmospheric chemical kinetic equations, Applied Mathematics & Computation, 271, 519-531, 2015.

Gao, M., Carmichael, G. R., Wang, Y., Saide, P. E., Yu, M., Xin, J., Liu, Z., and Wang, Z.: Modeling study of the 2010 regional haze event in the North China Plain, Atmos. Chem. Phys., 16, 1673-1691, 10.5194/acp-16-1673-2016, 2016.

Grell, G. A., Peckham, S. E., Schmitz, R., McKeen, S. A., Frost, G., Skamarock, W. C., and Eder, B.: Fully coupled "online" chemistry within the WRF model, Atmospheric Environment, 39, 6957-6975, https://doi.org/10.1016/j.atmosenv.2005.04.027, 2005.

Guenther, A., Karl, T., Harley, P., Wiedinmyer, C., Palmer, P., and Geron, C.: Estimates of global terrestrial isoprene emissions using MEGAN (Model of Emissions of Gases and Aerosols from Nature), Atmos. Chem. Phys, 6, 3181-3210, 2006.

Guenther, A. B., Jiang, X., Heald, C. L., Sakulyanontvittaya, T., Duhl, T., Emmons, L. K., and Wang, X.: The Model of Emissions of Gases and Aerosols from Nature version 2.1 (MEGAN2.1): an extended and updated framework for modeling biogenic emissions, Geoscientific Model Development, 5, 1471-1492, 10.5194/gmd-5-1471-2012, 2012.

Gurjar, B. R., Ravindra, K., and Nagpure, A. S.: Air pollution trends over Indian megacities and their local-to-global implications, Atmospheric Environment, 142, 475-495, https://doi.org/10.1016/j.atmosenv.2016.06.030, 2016.

Lawrence, B. N., Rezny, M., Budich, R., Bauer, P., Behrens, J., Carter, M., Deconinck, W., Ford, R., Maynard, C., Mullerworth, S., Osuna, C., Porter, A., Serradell, K., Valcke, S., Wedi, N., and Wilson, S.: Crossing the chasm: how to develop weather and climate models for next generation computers?, Geosci. Model Dev., 11, 1799-1821, 10.5194/gmd-11-1799-2018, 2018.

Li, J., Wang, Z., Wang, X., Yamaji, K., Takigawa, M., Kanaya, Y., Pochanart, P., Liu, Y., Irie, H., Hu, B., Tanimoto, H., and Akimoto, H.: Impacts of aerosols on summertime tropospheric photolysis frequencies and photochemistry over Central Eastern China, Atmospheric Environment, 45, 1817-1829, https://doi.org/10.1016/j.atmosenv.2011.01.016, 2011.

Li, J., Wang, Z., Zhuang, G., Luo, G., Sun, Y., and Wang, Q.: Mixing of Asian mineral dust with anthropogenic pollutants over East Asia: a model case study of a super-duststorm in March 2010, Atmospheric Chemistry & Physics, 12, 7591-7607, 2012.

Li, J., Wang, Z., Huang, H., Hu, M., Meng, F., Sun, Y., Wang, X., Wang, Y., and Wang, Q.: Assessing the effects of trans-boundary aerosol transport between various city clusters on regional haze episodes in spring over East China, Tellus B: Chemical and Physical Meteorology, 65, 20052, 10.3402/tellusb.v65i0.20052, 2013.

Linford, J. C.: Multi-core acceleration of chemical kinetics for simulation and prediction, High Performance Computing Networking, Storage and Analysis, Proceedings of the Conference on, 2013, 7,

Mielikainen, J., Huang, B., and Huang, A. H. L.: Intel Xeon Phi accelerated Weather Research and Forecasting (WRF) Goddard microphysics scheme, Geosci. Model Dev. Discuss., 2014, 8941-8973, 10.5194/gmdd-7-8941-2014, 2014.

San José, R., Pérez, J. L., Balzarini, A., Baró, R., Curci, G., Forkel, R., Galmarini, S., Grell, G., Hirtl, M., Honzak, L., Im, U., Jiménez-Guerrero, P., Langer, M., Pirovano, G., Tuccella, P., Werhahn, J., and Žabkar, R.: Sensitivity of feedback effects in CBMZ/MOSAIC chemical mechanism, Atmospheric Environment, 115, 646-656, https://doi.org/10.1016/j.atmosenv.2015.04.030, 2015.

Seinfeld, J. H., and Pandis, S. N.: Atmospheric Chemistry and Physics: From Air Pollution to Climate Change, 2nd Edition, 2012.

Sindelarova, K., Granier, C., Bouarar, I., Guenther, A., Tilmes, S., Stavrakou, T., Müller, J. F., Kuhn, U., Stefani, P., and Knorr, W.: Global data set of biogenic VOC emissions calculated by the MEGAN model over the last 30 years, Atmos. Chem. Phys., 14, 9317-9341, 10.5194/acp-14-9317-2014, 2014.

Skamarock, W. C., Klemp, J. B., Dudhia, J., Gill, D. O., Barker, D. M., Duda, M. G., Huang, X.-y., Wang, W., and Powers, J. G.: A description of the advanced research WRF version 3, NCAR Technical Note NCAR/TN-475+STR, 2008.

Sodani, A., Gramunt, R., Corbal, J., Kim, H. S., Vinod, K., Chinthamani, S., Hutsell, S., Agarwal, R., and Liu, Y. C.: Knights Landing: Second-Generation Intel Xeon Phi Product, IEEE Micro, 36, 34-46, 2016.

Wang, H., Chen, H., Wu, Q., Lin, J., Chen, X., Xie, X., Wang, R., Tang, X., and Wang, Z.: GNAQPMS v1.1: accelerating the Global Nested Air Quality Prediction Modeling System (GNAQPMS) on Intel Xeon Phi processors, Geosci. Model Dev., 10, 2891-2904, 10.5194/gmd-10-2891-2017, 2017.

Wang, Y., Chen, H., Wu, Q., Chen, X., Wang, H., Gbaguidi, A., Wang, W., and Wang, Z.: Three-year, 5 km resolution China PM2.5 simulation: Model performance evaluation, Atmospheric Research, 207, 1-13, https://doi.org/10.1016/j.atmosres.2018.02.016, 2018.

Wang, Z., Maeda, T., Hayashi, M., Hsiao, L. F., and Liu, K. Y.: A Nested Air Quality Prediction Modeling System for Urban and Regional Scales: Application for High-Ozone Episode in Taiwan, Water, Air, and Soil Pollution, 130, 391-396, 10.1023/A:1013833217916, 2001.

Wu, Q., Wang, Z., Gbaguidi, A., Tang, X., and Zhou, W.: Numerical Study of The Effect of Traffic Restriction on Air Quality in Beijing, Sola, 6, 17-20, 2010.
Wu, Q. Z., Xu, W. S., Shi, A. J., Li, Y. T., Zhao, X. J., Wang, Z. F., Li, J. X., and Wang, L. N.: Air quality forecast of PM10 in Beijing with Community Multi-scale Air Quality Modeling (CMAQ) system: emission and improvement, Geosci. Model Dev., 7, 2243-2259, 10.5194/gmd-7-2243-2014, 2014.
Xu, S., Huang, X., Oey, L. Y., Xu, F., Fu, H., Zhang, Y., and Yang, G.: POM.gpu-v1.0: a GPU-based Princeton Ocean Model, Geosci. Model Dev., 8, 2815-2827, 10.5194/gmd-8-2815-2015, 2015.
Zaveri, R. A., and Peters, L. K.: A new lumped structure photochemical mechanism for long-scale applications, Journal of Geophysical Research Atmospheres, 104, 30387-30415, 1999.
Zhang, Q., Jiang, X., Tong, D., Davis, S. J., Zhao, H., Geng, G., Feng, T., Zheng, B., Lu, Z., Streets, D. G., Ni, R., Brauer, M., van Donkelaar, A., Martin, R. V., Huo, H., Liu, Z., Pan, D., Kan, H., Yan, Y., Lin, J., He, K., and Guan, D.: Transboundary health impacts of transported global air pollution and international trade, Nature, 543, 705-709, 10.1038/nature21712 http://www.nature.com/nature/journal/v543/n7647/abs/nature21712.html#supplementary-information, 2017.
ZiFa, W., FuYing, X., XiQuan, W., JunLing, A., and Jiang, Z.: Development and Application of Nested Air Quality Prediction Modeling System, Chin. J. Atmos. Sci., 30, 778-790, 2006.
Zimmermann, J., and Poppe, D.: A Supplement for the RADM2 Chemical Mechanism: The Photooxidation of Isoprene, Atmospheric Environment, 30, 1255–1269, 1994.

Table 1. The possible values of iregime and the combination of chemical schemes.

iregime	1	2	3	4	5	6
Sub-schemes	COM	COM	COM	COM	COM	COM
	HET	HET	HET	HET	HET	HET
		URB	URB		URB	URB
			BIO			BIO
				MAR	MAR	MAR

Table 2. Compile flags of the different versions of CBM-Z.

Version of CBM-Z	Processor	Intel Compiler Flags	
		Flags for Vectorization	Flags for Floating-point Accuracy
Baseline CBM-Z	**Xeon E5-2680 V4**	–xCORE-AVX2	-fp-model precise
		–xCORE-AVX2	-fp-model fast=1
	Xeon Gold 6132	–xCOMMON-AVX512	-fp-model precise

		–xCOMMON-AVX512	-fp-model fast=1
MP CBM-Z	**Xeon E5-2680 V4**	–xCORE-AVX2	-fp-model precise
		–xCORE-AVX2	–fp-model fast=1
	Xeon Gold 6132	–xCOMMON-AVX512	–fp-model precise
		–xCOMMON-AVX512	–fp-model fast=1
	Xeon Phi 7250	–xMIC-AVX512	–fp-model fast=1

Table 3. Compile flags of the different versions of NAQPMS

Version of NAQPMS	Processor	Intel Compiler Flags	
		Flags for Vectorization	Flags for Floating-point Accuracy
Baseline NAQPMS	**Xeon E5-2680 V4**	–xCORE-AVX2	-fp-model precise
		–xCORE-AVX2	-fp-model fast=1
	Xeon Gold 6132	–xCOMMON-AVX512	-fp-model precise
		–xCOMMON-AVX512	-fp-model fast=1
NAQPMS with MP CBM-Z	**Xeon E5-2680 V4**	–xCORE-AVX2	-fp-model precise
		–xCORE-AVX2	–fp-model fast=1
	Xeon Gold 6132	–xCOMMON-AVX512	–fp-model precise
		–xCOMMON-AVX512	–fp-model fast=1

	Xeon Phi 7250	–xMIC-AVX512	–fp-model fast=1

Table 4. The performance tests of the baseline and optimized codes on different CPUs and KNL platforms with one physical cores. The unit of the wall-times for the tests is second (s).

	Processor	Vector Instruction	-fp-model	Wall-Time	Speedup
Baseline CBM-Z	**Xeon E5-2680 V4**	AVX2	precise	1014.67	1.00
			fast=1	792.03	1.28
	Xeon Gold 6132	AVX512	precise	665.44	1.52
			fast=1	497.64	2.04
MP CBM-Z	**Xeon E5-2680 V4**	AVX512	precise	581.14	1.75
			fast=1	153.32	6.62
	Xeon Gold 6132	AVX512	precise	352.00	2.88
			fast=1	55.42	18.31
	Xeon Phi 7250	AVX512	precise	3454.90	0.29
			fast=1	214.09	4.74

Table 5. The performance tests of the optimized codes on different CPUs and KNL platforms with MPI and OpenMP. The unit of the wall-times for the tests is second (s).

	Single Core Test				
	Processor	Vector Instruction	Number of Cores	Wall-Time	Speedup
	Xeon E5-2680 V4	AVX2	1	792.03	1.00
	MPI with Vectorization				
MP CBM-Z	**Xeon E5-2680 V4**	AVX2	28	7.57	104.63
	Xeon Gold 6132	AVX512	28	3.99	198.50
	Xeon Phi 7250	AVX512	68	4.52	175.23
	OpenMP with Vectorization				
	Xeon E5-2680 V4	AVX2	28	7.84	101.02
	Xeon Gold 6132	AVX512	28	4.07	194.60
	Xeon Phi 7250	AVX512	68	4.73	167.45

Table 6. The performance tests of the baseline and optimized codes on the diverse platforms with different compile flags. The unit of the wall-times for the tests is second (s).

	Processor	Vector Instruction	-fp-model	Wall-Time (CBMZ)	Wall-Time (Total)	Speedup (CBMZ)	Speedup (Total)
Baseline NAQPMS	**Xeon E5-2680 V4**	AVX2	precise	17675.86	24460.54	1.00	1.00
			fast=1	13201.56	19619.20	1.34	1.25
	Xeon Gold 6132	AVX512	precise	13817.24	18950.95	1.28	1.29
			fast=1	10544.60	15502.39	1.68	1.58
NAQPMS with MP CBMZ	**Xeon E5-2680 V4**	AVX2	precise	11127.90	17454.95	1.59	1.40
			fast=1	3971.48	10019.21	4.45	2.44
	Xeon Gold 6132	AVX512	precise	9584.59	14698.38	1.84	1.66
			fast=1	2150.20	6994.43	8.22	3.50
	Xeon Phi 7250	AVX512	fast=1	2997.96	19239.20	5.90	1.27

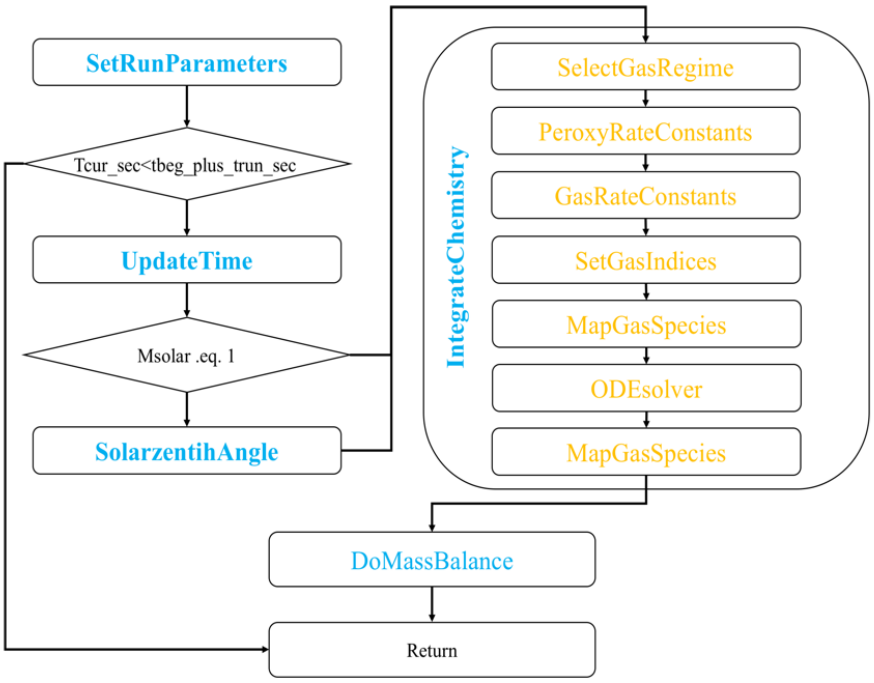

Figure 1. The framework of the CBM-Z gas-phase chemistry module. The functions in the yellow font represent the inner function of IntegrateChemistry.

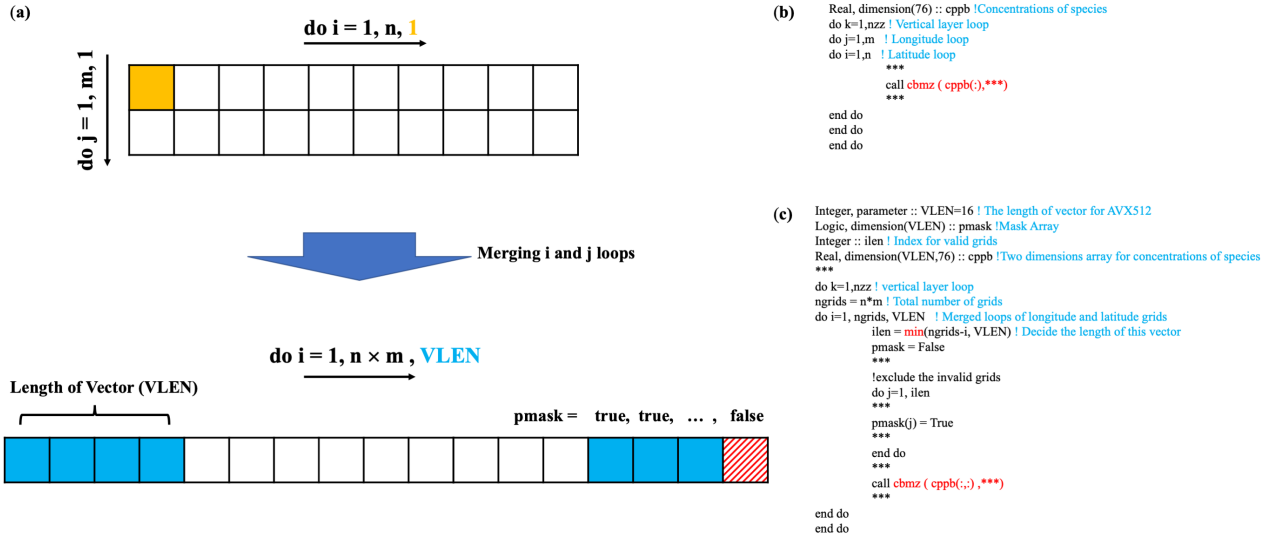

Figure 2. A schematic diagram of the changes of the calling method of CBM-Z. The calling method of the CBM-Z module changes from calculating one model grid calculation at a time to multiple model grids at the same time. The VLEN represents the number of points operated simultaneously, which is determined by the length of the register in the Vector Processing Unit (VPU). The i and

j loops, equaled latitude and longitude loops, were merged to construct one vector to reduce the number of unfilled vectors. (b) and (c) illustrate the sample codes before and after integrating grids.

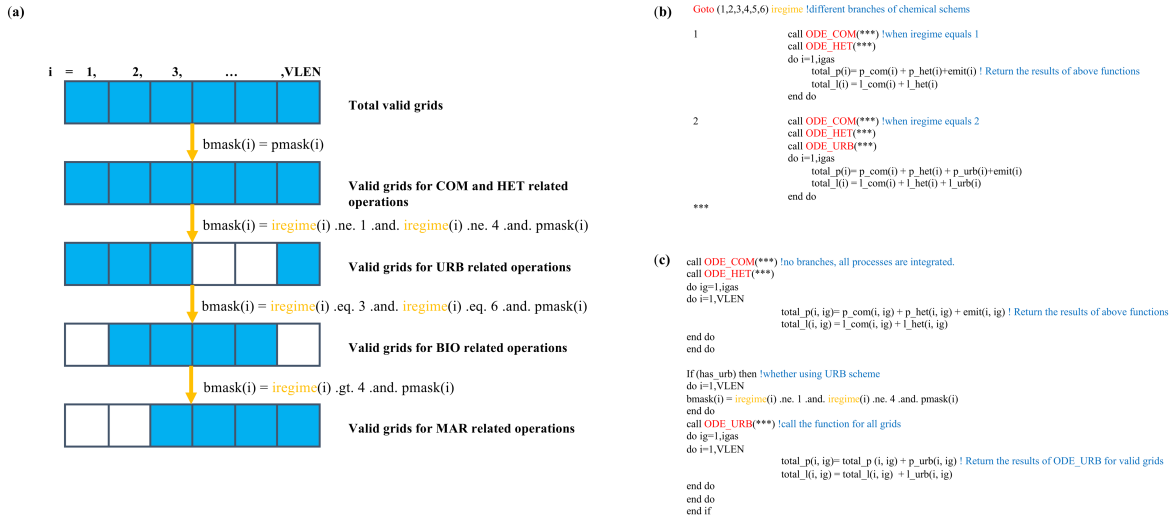

Figure 3. The flowchart (a) shows the way to mask the heterogeneous girds to integrate grids to perform the vectorization operations according to the iregime values. (b) and (c) illustrate the sample codes before and after integrating grids. In figure (b), iregime leads different calling processes; in figure (c), the calling processes are integrated in one flow, and the function are called for all grids but only the values of valid grids would be returned.

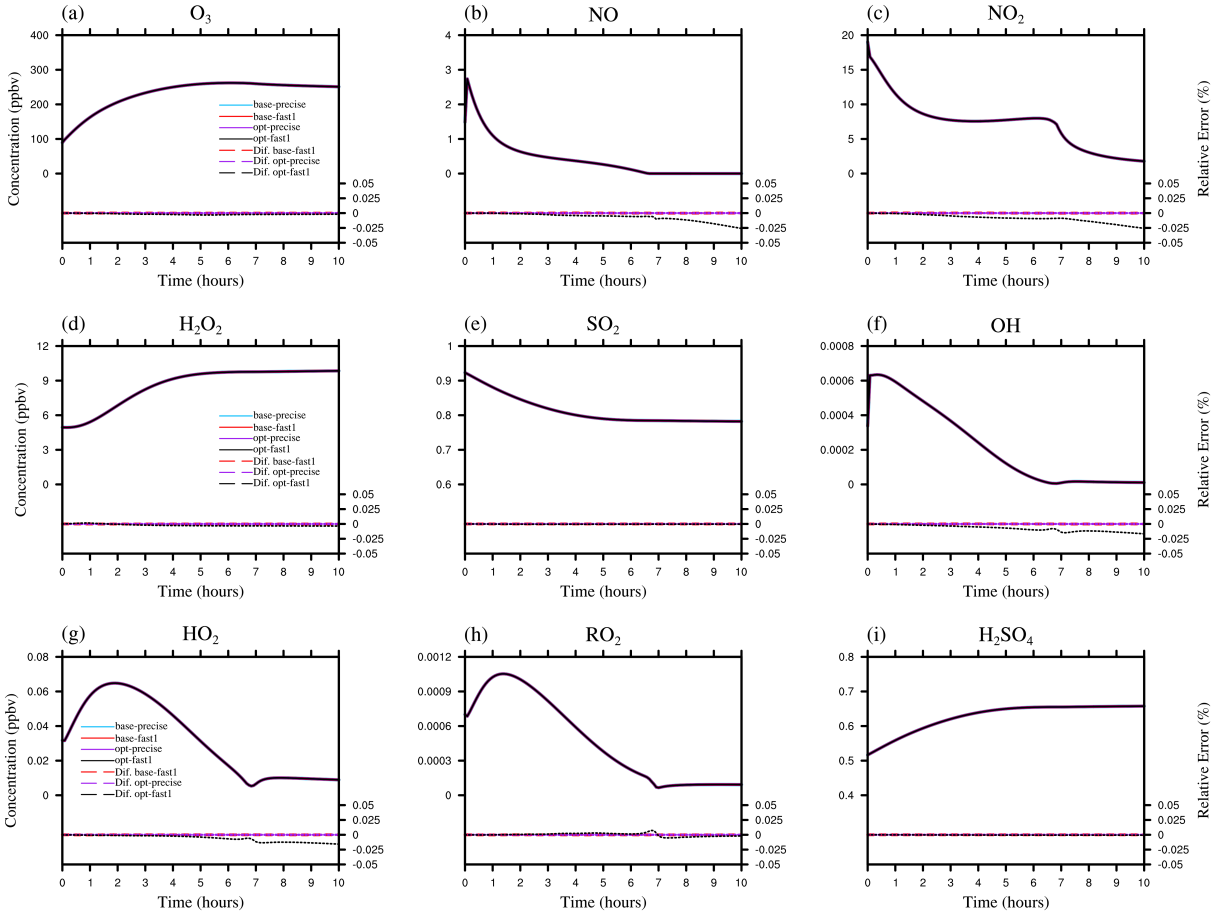

Figure 4. Comparison of the time-series concentrations of O_3, NO, NO_2, H_2O_2, SO_2, OH, HO_2, RO_2 and H_2SO_4 ((a)-(i)) from the baseline and optimized CBM-Z simulation with diverse -fp-model options.

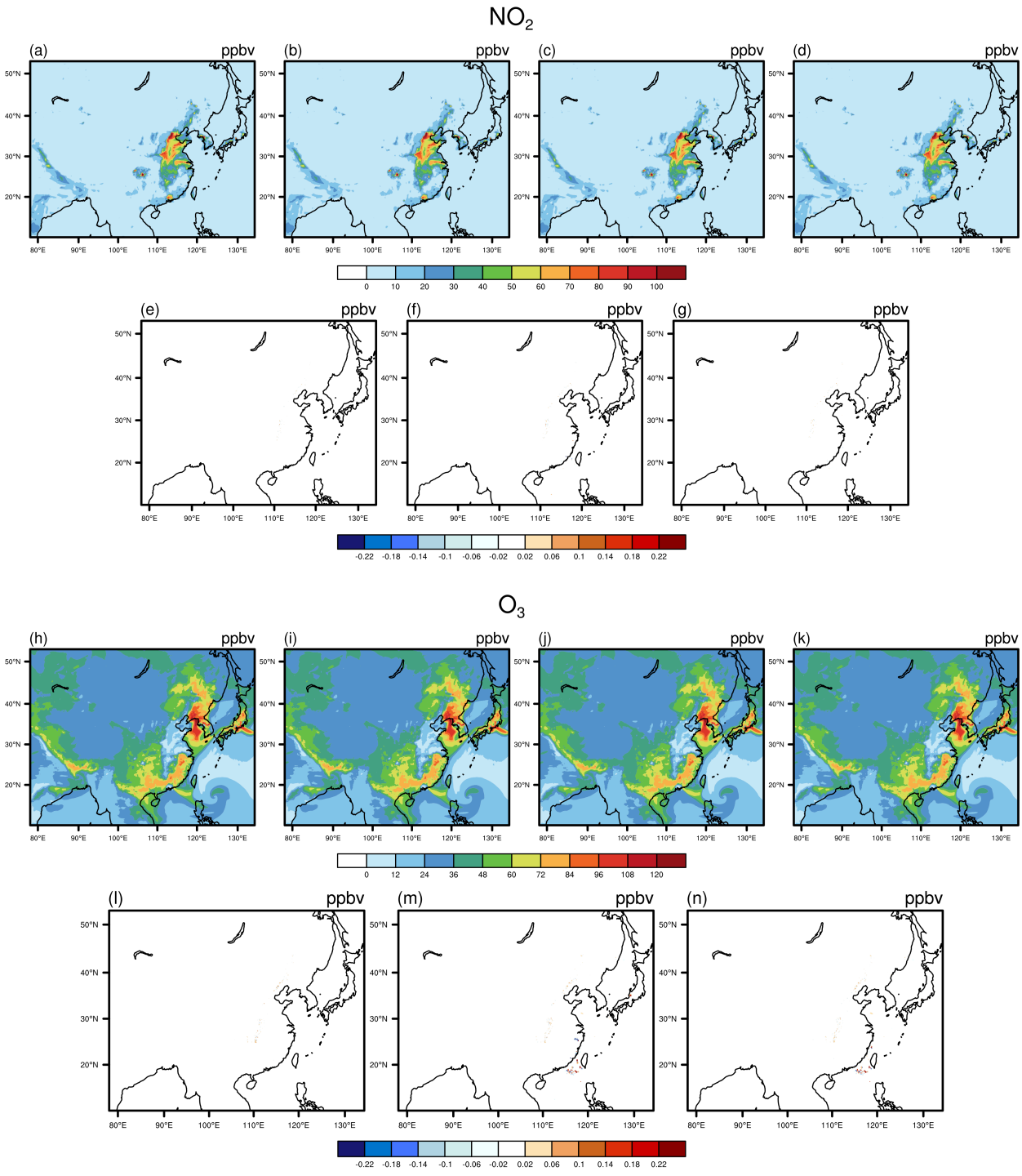

Figure 5. NO_2 and O_3 concentrations outputted by baseline and optimized codes with different accuracy compile flags. (a) and (h) are from baselines codes compiled by *precise* option, which are treated as benchmark for comparison. (b) and (i) are from

optimized codes compiled by *precise* option. (c) and (j) are from basline codse compiled by *fast=1* flag. (d) and (k) are from optimized codes compiled by *fast=1* flag. (e)-(g) and (l)-(m) are the output concentration differences of optimized codes (precise), baseline codes (fast=1) and optimized codes (fast=1) compared with baseline codes (precise).

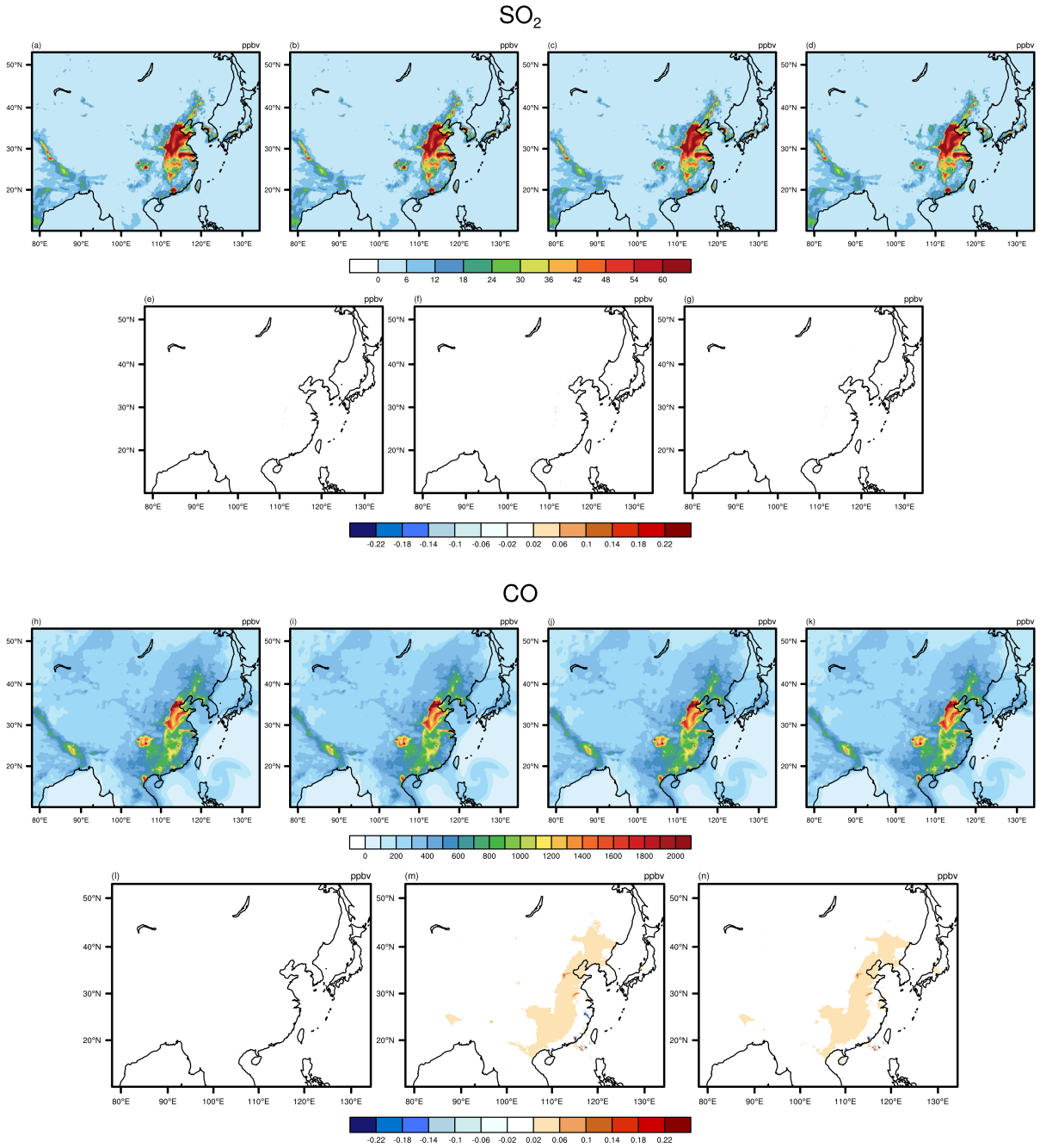

Figure 6. SO_2 and CO concentrations outputted by baseline and optimized codes with different accuracy compile flags. (a) and (h) are from baselines codes compiled by *precise* option, which are treated as benchmark for comparison. (b) and (i) are from

optimized codes compiled by *precise* option. (c) and (j) are from basline codse compiled by *fast=1* flag. (d) and (k) are from optimized codes compiled by *fast=1* flag. (e)-(g) and (l)-(m) are the output concentration differences of optimized codes (precise), baseline codes (fast=1) and optimized codes (fast=1) compared with baseline codes (precise).

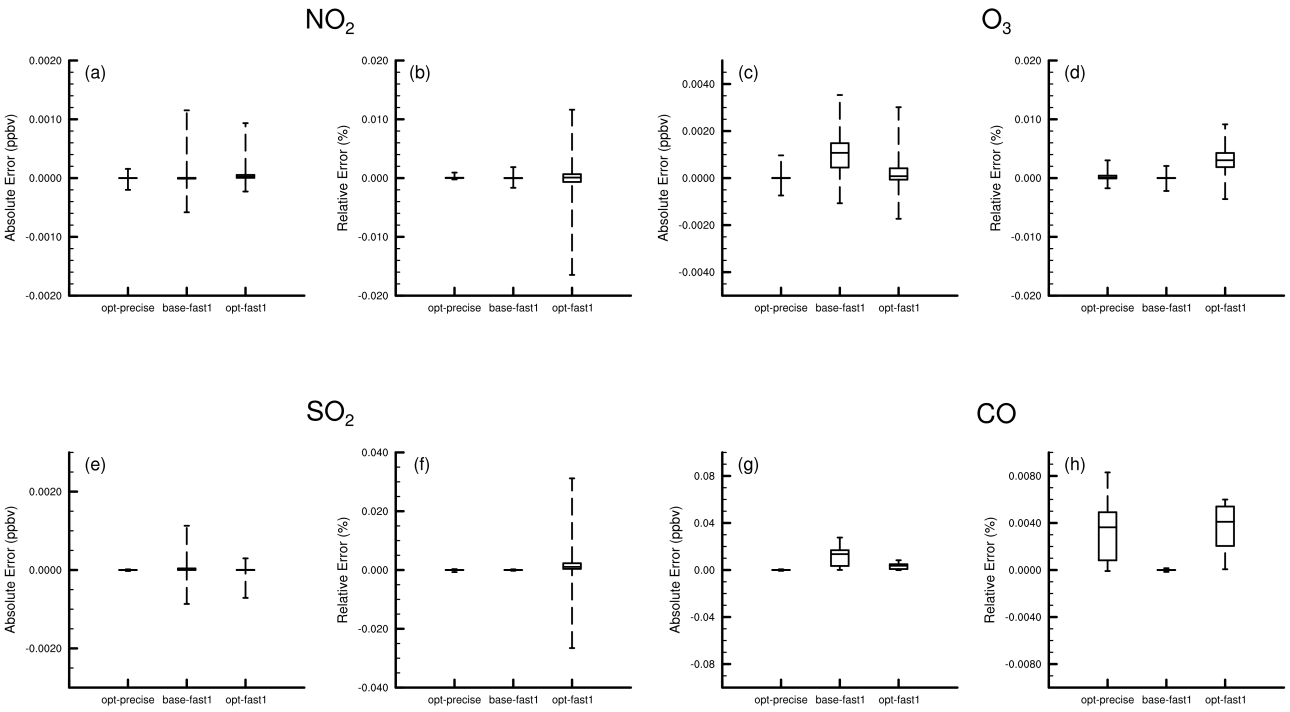

Figure 7. The distributions of Absolute Errors and Relative Errors for O_3, NO_2, SO_2 and CO in near surface model layer. The reference points are 1%, 25%, 50%, 75% and 99%.