Springer Tracts in Mechanical Engineering

Shinobu Yoshimura
Muneo Hori
Makoto Ohsaki *Editors*

High-Performance Computing for Structural Mechanics and Earthquake/Tsunami Engineering

Springer

Springer Tracts in Mechanical Engineering

Shinobu Yoshimura • Muneo Hori • Makoto Ohsaki
Editors

High-Performance Computing for Structural Mechanics and Earthquake/Tsunami Engineering

Springer

Editors
Shinobu Yoshimura
Department of Systems Innovation,
School of Engineering
The University of Tokyo
Tokyo
Japan

Muneo Hori
Earthquake Research Institute
The University of Tokyo
Tokyo
Japan

Makoto Ohsaki
Department of Architecture
and Architectural Engineering
Kyoto University
Kyoto
Japan

ISSN 2195-9862 ISSN 2195-9870 (electronic)
Springer Tracts in Mechanical Engineering
ISBN 978-3-319-21047-6 ISBN 978-3-319-21048-3 (eBook)
DOI 10.1007/978-3-319-21048-3

Library of Congress Control Number: 2015954713

Springer Cham Heidelberg New York Dordrecht London

Printed on acid-free paper

Springer International Publishing AG Switzerland is part of Springer Science+Business Media (www.springer.com)

Contents

Fundamentals of High-Performance Computing for Finite Element Analysis

Hiroshi Kawai, Masao Ogino, Ryuji Shioya, and Shinobu Yoshimura

Abstract As introduction, high-performance computing (HPC) technology for finite element analysis is explained. First, general notions, tips and techniques which are useful for the programming on modern HPC and supercomputing environments are introduced. Here, as a supercomputer, the authors assume mainly a distributed memory parallel computer with each computational node having one or more multi-core scalar processors. Both hardware and software aspects are covered, with important concepts such as MPI, OpenMP and other compiler directives, as well as network interconnect, cache, memory bandwidth, SIMD and peak performance ratio. There are also some tips and advices about the software development process specific for the supercomputers, such as software design, testing and debugging, and profiling. Then, some important ideas and techniques for the development of FE-based simulation code running on supercomputers, such as the selection of linear algebraic solvers, domain decomposition method (DDM), element-by-element (EBE), as well as mesh generation and visualization are explained. This chapter

11th World Congress on Computational Mechanics (WCCM XI)
5th European Conference on Computational Mechanics (ECCM V)
6th European Conference on Computational Fluid Dynamics (ECFD VI)
20–25 July 2014, Barcelona, Spain

H. Kawai (✉)
Tokyo University of Science-Suwa, 5000-1 Toyohira, Chino-shi, Nagano 391-0292, Japan
e-mail: kawai@rs.tus.ac.jp

M. Ogino
Nagoya University, Furo-cho, Chikusa-ku, Nagoya-shi, Aichi 464-8601, Japan
e-mail: masao.ogino@cc.nagoya-u.ac.jp

R. Shioya
Toyo University, 2100 Kujirai, Kawagoe-shi, Saitama 350-8585, Japan
e-mail: shioya@toyo.jp

S. Yoshimura
The University of Tokyo, 7-3-1 Hongo, Bunkyo, Tokyo 113-8656, Japan
e-mail: yoshi@sys.t.u-tokyo.ac.jp

S. Yoshimura et al. (eds.), *High-Performance Computing for Structural Mechanics and Earthquake/Tsunami Engineering*, Springer Tracts in Mechanical Engineering,
DOI 10.1007/978-3-319-21048-3_1

could also serve as the introduction of HPC and supercomputing to the following chapters dealing with more specific problems and schemes.

1 Introduction

Welcome to the supercomputing world! Here in this chapter, we explain various issues and techniques related to high-performance computing (HPC), and especially, how to write your finite element code running on a modern supercomputer.

Before understanding the current modern HPC technology, it is beneficial to know what had already happened in the past. Here we explain a brief history of supercomputing. By the way, the keyword "supercomputer" is generally defined as "a computer which is at least 1000 times faster than an ordinary computer like note PC and tablet."

About three decades ago, a supercomputer at that moment was a vector computer, which equips a vector-processing unit. To take advantage of the vector supercomputer, we had to vectorize our program. Although it forced us to modify some portions of our code, still it was a good old times because we needed to focus on just only one thing, "vectorization." Later on, shared memory parallel processing was added to the vector supercomputer. And compiler directive approaches for parallelization, which eventually evolved into OpenMP, also appeared at that time. On the other hand, although not successful, another type of supercomputers based on single instruction multiple data (SIMD) architecture was tried also. The usage of these SIMD machines was almost similar to that of the vector processor, based on the parallelization of loops.

Then, the 2nd wave of supercomputing, so-called massively parallel processors (MPP) arrived. It was a distributed memory parallel computer, and classified as multiple instruction multiple data (MIMD). Various kinds of message passing approaches had been appearing, but finally they were merged into the MPI standard. It actually had changed the fundamental design of our finite element code drastically. The notion of domain decomposition became MUST to exploit the potential power of MPP. Later on, MPP was taken over by so-called attack of killer micro, PC cluster. This means, a supercomputer became just a bunch of PC boxes connected each other by high-speed network. Each "box," a computational node of the supercomputer, contained one or a few scalar processors and their own memory space. Still, it was relatively easier than now to write code because we needed to think about MPI communication and domain decomposition only.

Currently, we are just in the midst of the 3rd on-going wave of supercomputing. Some people call it the era of accelerator in general, or GPU in more specific. Other people say multi-core or many-core age. In either way, while the number of computational nodes in one supercomputer system had gradually grown to hundreds of thousands, each computational node itself had become a kind of "beast."

So now, we are in the truly massively parallel era. Here, "massively parallel" means so many that it is virtually uncountable, while at the moment of the 2nd wave a typical MPP system had processors of only a thousand or less. The number of cores in a current peta-scale supercomputer is already reaching near 1 million, and it could increase to 1 billion in an exa-scale system which will soon appear in very near future.

On the other hand, readers should not forget that your desktop PC or workstation has also become very, very powerful. A typical workstation, which equips GPU or many-core accelerator, enjoys more than 1 TFLOPS. Welcome to Desktop Supercomputing! Unfortunately, something wrong about this machine is that a typical simulation code developed by a researcher runs only 1 GFLOPS on the machine, if without any performance tuning effort. It is simply because the clock frequency of these processors itself is only around a few GHz. Anyway, between Giga and Tera, 1000 times of difference! Is it a kind of cheat or FLOPS fraud? That's why we call it a "beast," which cannot be tamed easily. The same thing applies to supercomputing because currently the architecture of each computational node of a supercomputer is virtually the same as that of a desktop PC or a workstation. Thus, a bunch of "beasts" forms a "monster," the massively parallel many-core supercomputer.

Here in this chapter, first, supercomputing hardware and software technologies are explained from programmer's point of view. Then, the more specific topic, "how can I tune up my finite element code?" is explained.

Because of this limited space, each topic is explained only briefly, without enough illustrations and examples. Through this chapter, however, readers can find several important keywords, which are typically related to frequently occurring troubles and issues during the development of simulation code. Starting from these keywords, you can understand the challenge and also find the solution to them. Hopefully, this chapter could serve you as a survival guide in the supercomputing jungle.

2 Hardware

In this section, we start from hardware issues first. Here, we focus on three hardware elements: processor, memory, and network interconnect. To highlight the potential implication of these components to the performance design of simulation code, we try to explain them from programmer's point of view as much as possible.

A typical modern supercomputer consists of many computational nodes. They are connected each other by high-speed network interconnect. Each computational node holds one or a few processor chips and its associated memory chips, thus forming a distributed memory parallel computer. As a result, both intra-node and inter-node performance are important for the performance design of simulation code.

2.1 Processor

Nowadays, a processor can issue multiple instructions and execute multiple floating-point operations for each clock cycle. For example, a pair of multiplication and addition, or a set of multiply-add pairs packed in one SIMD instruction. It is called instruction-level parallelism (ILP). For example, if a processor can handle eight double precision (DP) floating-point operations per clock, and it runs with 2 GHz, its DP peak performance is $8 \times 2 = 16$ GFLOPS. Theoretically, it is nice, if it always works so. The reality, however, is not so simple. How can we write code which keeps handling four add and four multiply operations in every clock? Anywhere branch or procedure call appears, it doesn't work so. It is so-called ILP wall problem. This is one of the major issues to widen the gap between the ideal, peak performance of the hardware and the actual, sustained performance when running real application code on the hardware.

For example, let's think about SIMD instruction. The SIMD mechanism allows multiple operations of the same type to be packed in one instruction and executed at once. The operation can be addition, subtraction, multiplication or division, or even a pair of multiply and add/subtract, so-called fused multiply-add (FMA). If a processor can handle a set of four FMA operations, which is called 4-way SIMD, it sums up to eight operations per clock. A modern processor can handle 4, 8 or 16 operations in each clock. This number tends to grow up further as the density of transistors in semiconductor design increases (Moore's law).

Moreover, the above fact is merely for one core in a processor chip. In recent years, the number of cores in a chip had also increased. Here, "core" means "processor" in a traditional sense. Until recently, one processor corresponds to just one chip, or sometimes to multiple chips in case of very complicated processor design. But nowadays, because of Moore's law, one processor chip can hold multiple processors. To avoid ambiguity, the keyword "processor" is renamed to "core." Thus, it is called "multi-core." Currently, a typical scalar processor has 2, 4, 6 or 8 cores in one chip, although there are some "many-core" chips having more than ten cores. They can be regarded as a shared memory computer in one chip. Either MPI or OpenMP can be used for parallel programming. However, multi-core has more serious implication to memory access, which will be explained soon in the next section.

2.2 Memory

Even if the processor can perform multiple operations per clock, in reality, it is no meaning if data cannot be supplied from the memory system into the processor in enough speed.

Let's start from a simple example of adding two vectors, like "$c(i) = a(i) + b(i)$." Also assume that a processor chip has 8 cores, and each core equips 4-way SIMD and perform four add operations per clock. If the chip runs in 3 GHz, it can perform 4 operations × 8 cores × 3 G times per second = 96 GFLOPS. Not so bad. Then, think about data also. Data I/O required to read/write is, assuming double precision, 8 bytes per number, 8 bytes × 3 I/O (2 read, 1 write) × 96 G FP operations per second = 2304 GB/s. Currently, no memory system can supply data in the speed of such vast amount. Usually, only from 30 to 300 GB/s at most. It is so-called memory wall problem.

Memory access pattern is the important keyword to understand the performance characteristics of the memory system. Let's imagine the memory access patterns in a loop. It can be sequential, stride or random. In case of random access, actually it may be indirect index access. For example, in the context of finite element code, touching nodal data in each element loop is a typical example of this case. For each finite element of the loop, its associated nodal data must be accessed through the element connectivity data, in a form like "data(index(i))." Of these access patterns, typically, the sequential access pattern is the fastest. On the other hand, stride and indirect index accesses have some problems. Therefore, if possible, data should be rearranged so that the sequential access pattern dominates.

For example, let's consider an element-by-element (EBE) loop. In this loop, many floating-point operations have to be performed for each finite element. However, its dominant memory access pattern is indirect index access. This memory access pattern may prohibit efficient execution of the loop body. To overcome the issue, an extra element loop is added just before the main loop, and in this loop, nodal data are first copied into another element-wise array. Then, in the main element loop, using this extra element-wise array, all the data access can be done in element-wise manner, thus, sequentially. It can allow compiler to vectorize the main loop. It is a useful technique if EBE involves relatively heavy calculation.

Cache memory is very important in the design of modern scalar processors, and it is related to a memory access pattern called data locality. In short, the cache memory is a fast but small special memory region, and it is separated from the main memory. When the processor tries to read a small amount of data from the main memory, the fetched data is once copied into the cache memory automatically. Then, suppose this small data becomes necessary again. If the data still remains on the cache memory, instead of reading the original data again from the slow main memory, it is sufficient to access this copy from the fast cache. This means, it is better to keep using this small amount of data as much as possible. If the size of frequently used data is larger than the size of the cache, however, it doesn't work in this way. While reading it, most of data are kicked out of the cache, and it ends up virtually accessing to the main memory. Thus, the cache mechanism works only if "a relatively small amount of data is accessed repeatedly, repeatedly and repeatedly."

Then, how can we utilize such tricky cache mechanism? There is a well-known technique called blocking or tiling. In cache blocking, a big data region is divided into many small data blocks first. For example, in case of a matrix, it is decomposed into multiple sub-matrices. They look like tiles. Using associated blocking algorithm, once a small block is read from the main memory, it is utilized repeatedly before the next one is needed.

From programmer's point of view, there is one easy way to implement this cache blocking. First, prepare explicitly small arrays as working area. The total size of these arrays must fit on the cache memory. Then, copy data subset from the big array into these small arrays. Using only this working area, perform relatively heavy calculation. And finally, move the result back to the big array. Was it easy? If this way of coding style works fine with your case, you are lucky to obtain a blocking version of your algorithm.

However, if any blocking algorithm cannot be devised, how come? In this case, sadly, you have to directly tackle against the slowness of the main memory. Now, memory bandwidth becomes a serious issue.

To consider the memory bandwidth problem, it is useful to understand the keyword, B/F ratio. B/F ratio measures how much data can be read/write from/to the main memory for each execution of floating-point operation. This ratio typically assumes a specific numerical algorithm. If the actual B/F ratio value is no less than the value originally required by the algorithm, it is OK. Otherwise, there is memory bandwidth bottleneck. You may switch to much better supercomputer, if possible. Or you might be forced to modify your algorithm.

For example, let's think about matrix–vector product. Suppose the size of matrix and vector as N. Code can be written as "$c(i) = c(i) + a(i, j) * b(j)$," with "$i$" as outer loop and "$j$" as inner loop. Assuming "$c(i)$" fits on processor register, there are two read accesses, "$a(i, j)$" and "$b(j)$." Therefore, assuming double precision, 8 bytes per number, the total amount of data I/O is 16 N^2 bytes, while that of floating-point operations is 2 N^2 operations. In this case, B/F ratio becomes 8. Furthermore, a blocking algorithm can be applied to the case of this matrix–vector product. Because vector "$b(j)$" can be moved on cache, so B/F ratio drops to 4. Still, a supercomputer with a very strong memory system, having B/F ratio 4, is required to perform this algorithm efficiently. Otherwise, the plenty of extra floating-point hardware resources available in processor side is merely wasted.

In reality, the current situation is very bad. Most of scalar processors can supply B/F ratio of only from 0.1 to 0.5 at most. This means, when the matrix–vector product is performed, peak performance ratio becomes only a few percent! Is it a kind of cheat or FLOPS fraud? That's why it is so-called the issue of "memory wall."

The recent rapid growth of the raw computational power of a processor chip in terms of floating-point operations, caused by the increase of both the number of processor cores and the number of SIMD ways, has made this memory bandwidth issue more serious and desperate. Thanks to Moore's law, the growth pace of the FP capability is far exceeding that of the memory bandwidth. While each core has its own cache memory (roughly speaking), the path to the main memory is basically shared among all the cores, because it is "shared memory." As a result, only the algorithms which can take advantage of cache mechanism can keep up with the growth of the computational power, while B/F ratio will drop further.

2.3 Network Interconnect

Of the three key hardware components, processor, memory and network interconnect, the former two, explained already, are also related to the performance design of simulation software running on a desktop PC and a workstation. In this sense, the last one, network interconnect, is the only key component unique in supercomputing. In modern supercomputing, distributed memory parallel architecture is the primary hardware architecture. A supercomputer is composed of multiple computational nodes. Each of them has its own memory space. A computational node cannot directly access memory owned by other nodes. Instead, it has to send/receive data to/from others through this high-speed network interconnect.

As the architecture of the network interconnect, nowadays torus or fat tree is utilized for a high-end supercomputer, while for low-end HPC environment like a PC cluster, switching hardware is used. The fat tree can be said to be cascade of switches. On the other hand, in case of the torus architecture, each node has very high bandwidth connection directly, but to only a few neighbouring nodes. To communicate to any node other than these neighbour nodes, data has to be relayed through one or a few intermediate nodes. It is typically used to connect a relatively large number of nodes.

To use such a distributed memory parallel computer, care must be taken about communication patterns. Because some of the communication patterns are either inherently efficient or strengthened by additional hardware mechanism, the use of these special patterns should be considered first in the design of parallel software.

Global communication, such as barrier, broadcast and reduce operations are frequently used in various kinds of parallel programs. High-speed interconnect supports those patterns directly in hardware level. Therefore, if these communication patterns are recognized in your code, instead of ordinary send/receive protocol, the corresponding special communication API or directives should be used. They are the only routes to take advantage of the special network hardware mechanism.

On the other hand, neighbour communication is another important pattern. To understand it, let's consider the difference between bus and switch. Suppose there are four nodes, A, B, C and D. In case of the bus architecture, nodes C and D cannot communicate while nodes A and B are talking. In case of the switching architecture, however, communication of A–B and C–D can work simultaneously. It can be easily extended to the case of many, many nodes: A–B, C–D, E–F, G–H, and so on. All of them can be invoked at once. Instead, if node A wants to receive data from more than one, for example, nodes B and C, there will be conflict.

The neighbour communication pattern is very important in case of macro-scale, continuum mechanics-based simulation. In this type of simulation, a whole analysis domain can be decomposed into multiple subdomains. Typically, communication between neighbouring subdomains frequently occurs. It can be represented as the neighbour communication pattern. Thus, well-designed code for the continuum mechanics field can easily scale on a supercomputer.

As for the neighbour communication pattern, it is better to understand the keyword, volume to area ratio. This means the ratio between amount of calculation per subdomain as a volume, and that of the associated interface boundaries as an area. As the problem size grows, this ratio is expected to grow as well, because of comparison between volume N^3 and area N^2. That means, the bigger the problem size is, the easier to parallelize. You can always enjoy good parallel efficiency, if you specify the problem size big enough. Of course, there are some issues, too long execution time and lack of memory.

In reality, however, you cannot wait so long. And more often, you may want to solve the problem just faster, with the problem size fixed. In this case, further intensive performance optimization of the communication routines will be required, such as overlap of communication and computation. There is a notorious saying that, "Using a supercomputer, you cannot shrink time. Instead, you can grow problem size."

3 Software

In the previous section, the hardware architecture of modern supercomputers is described. It is also necessary to mention about more software-related topics, especially, the software development process and environment of supercomputing applications. In this section, starting from software design principle, basic programming models, programming languages, compilers, libraries and tools are explained.

3.1 Performance-Centric Software Design

Anyway, why using a supercomputer? Of course, you want to run your simulation code faster, and also you may want to make the problem size much larger, with a finer mesh, more detailed geometry, much wider coverage of the analysis domain, and more accurate modelling. If you can just use your original, un-tuned program as is without modification, you will be very happy. The reality is much harder, however, and you will be forced to optimize your code, to take advantage of the potential power of the supercomputer.

Usually, the performance optimization task is only a small portion of the whole software development cycle. Testing, debugging, trying to find much better design for maintenance and extension in the future, and moreover, capturing true user requirement (for example, in case of simulation, appropriate modelling of target physical phenomena) are more important and essential tasks. As a result, the performance tuning is often ignored. Once you have decided to use a supercomputer, however, this task suddenly becomes more than essential, and the first priority. It is simply because, if you failed to gain performance boost from the use of the supercomputer, it would be just a waste of time and money. The sole purpose of using the supercomputer is, to make your code faster.

In the good old days, money could buy everything. All you need to do was, just to prepare budget for renting a supercomputer. With the same, original code, you could have enjoyed much faster simulation. Nowadays, however, money alone is not enough. You need also time to modify your code. To do so, the design of the code should be re-considered. "Why does this code run so slowly?" Then, here comes the performance-centric software design.

To incorporate the performance tuning tasks into the software design process up-front, there is an established systematic approach. It consists of three tasks, finding hot spot, scaling test and unit node test.

First, hot spot has to be identified. Here, the hot spot means routines or portion of the code which consume the majority of execution time. In most of simulation programs, it could easily happen that only a few routines or loops spend more than 99 % of time. Then, it is good enough to focus your effort only on these hot spot routines and loops. They may also be called as performance kernels.

Next, in case of parallel processing, scaling test is performed to check the scalability of the code and also to identify the performance bottleneck if there exists. There are two types of tests, strong scaling and weak scaling. The former test keeps the problem size unchanged, and it can be used to make sure the computational time shrinks as the number of processors increases. On the other hand, in case of the latter one, both the number of processors and the problem size are increased. In an ideal case, the execution time does not change. Instead, if the execution time also increases, there would be the bottleneck against parallelization.

Recently, not only the scalability, but also the concept of peak performance ratio becomes more and more important. This is the ratio between the peak performance of the supercomputer as hardware and the actual speed of application code, both measured in FLOPS. For example, if the simulation code runs with the speed of 100 TFLOPS on the machine with the peak performance 1 PFLOPS, the ratio is 10 %. Usually, the scalability is pre-requisite for the high peak performance ratio, but more is required to obtain good peak ratio. The unit node performance, the performance measured on a single computational node, cannot be ignored as well. If the peak ratio on a single node is less than 5 %, the ratio value using the whole system can be also 5 % in the best case, and it is usually less than this value. Thus, the unit node performance is another important measure, and the unit node test has to be carried out.

Anyway, what the unit node performance actually means? In case of a modern supercomputer, the hardware architecture of each computational node is almost the same as that of a brand new PC on your desktop. Therefore, before porting your code onto the supercomputer, it is a good idea to tune the code on your desktop PC first. However, this task itself is getting harder and harder nowadays.

3.2 Programming Model

To consider the software design for modern supercomputing applications, there are some special programming models. Each model covers a specific HPC hardware architecture. Here, we introduce two important programming models, hybrid parallel and data parallel.

Suppose if a supercomputer is based on multi-core scalar processors, there are two choices in programming models, flat MPI and hybrid parallel. In case of flat MPI, it is just sufficient to re-use your MPI-based code without modification, but you need to set the number of MPI ranks as same as the number of total cores using. On the other hand, the hybrid parallel programming model is needed mainly for two reasons. One is, to obtain the maximum parallel efficiency. The other one is, to utilize processor cores of sub-million order or more.

Theoretically, it is better to utilize OpenMP for intra-node communication, while MPI is used for inter-node one. In reality, it depends on applications and problems. Typical trend is, when the problem size is relatively small while the number of cores used is many, hybrid parallel programming model works well. If the parallel efficiency is already good enough with the current flat MPI implementation, however, the additional gain obtained from the hybrid parallel approach is marginal. In case of domain decomposition-based approaches, because of the volume to area ratio, explained before, the bigger the problem size is, the better the parallel efficiency is.

On the other hand, if luckily you have a chance to utilize a world-class supercomputer, then unluckily you will be forced to adopt the hybrid parallel programming model, simply because the flat MPI model does not work in this environment. Currently, no MPI implementation seems to work well more than ten thousand MPI processes. The combination of MPI and OpenMP is the only way to utilize millions of cores available on such a top-end supercomputer.

Data parallel programming model, or SIMD, is the programming model useful for GPU, and also for SIMD instruction in modern scalar processors. SIMD is single instruction multiple data. It usually involves the use of either compiler directives, such as SIMD vectorization and OpenACC, or extension to existing programming languages, such as CUDA and OpenCL. In the former cases, a loop is annotated by compiler directives with additional information necessary to parallelize. In case of CUDA and OpenCL, the loop body of a loop is extracted as a GPU-local routine, which is associated to a thread when running on the GPU.

3.3 Programming Language and Compiler

To port your simulation code onto a supercomputer, more or less some amount of code modification effort will be required. It typically involves inserting special compiler directives such as vectorization, OpenMP and OpenACC into the code, and calling functions/subroutines of HPC-related libraries such as MPI and BLAS. In the worst case, instead of keep using ordinary programming languages such as Fortran, C and C++, special programming languages dedicated for specific HPC environments might have to be employed, and the code would be re-written completely from scratch.

If the hybrid parallel programming model, explained before, is employed, OpenMP is the primary choice for intra-node parallelization. OpenMP is mainly a set of compiler directives designed for thread-level parallelization on shared memory parallel environment. It fits well on modern multi-core or many-core scalar processors. Programmer specifies OpenMP compiler directives on each loop which can be parallelized. Instead, if your code is relatively simple and you are lucky, the automatic parallelization capability of compiler may also work well.

OpenMP can be utilized if a computational node contains multiple processor chips, each of which may further be multi-core. In this case, non-uniform memory access (NUMA) memory architecture should be considered. Each processor chip has direct connection to memory chips. That means, each processor has its own memory. Access to own memory is fast, while access to other processor's memory tends to be much slower. Actually, it can be considered as distributed memory. From programmer's point of view, first touch principle can be applied. Data region "touched" first by a thread running on a processor is owned and managed by the processor, and it is allocated on its memory. Through the capability of OpenMP, thread ID information is required to handle this situation.

Recently, the impact of SIMD instructions is becoming more and more important for the performance design in supercomputing. To utilize the SIMD instructions, there are three ways. One is, of course, directly use assembly language. Most of people, including us, would want to ignore the first choice. Then, the second option is, to use SIMD intrinsic functions/subroutines. Still, this option is very tedious. Therefore, the third option, compiler-driven vectorization, is more practical for the most of programmers.

Compiler-driven vectorization technique for SIMD is very similar to the so-called "vectorization" in the vector supercomputer age. The idea itself has been very simple, inserting compiler directives just before loops which can be vectorized. However, there is one big difference between the current SIMD vectorization (also called, short vectorization) and the old predecessor. While the vector processor can aggressively read/write data from/to main memory, the SIMD mechanism of the modern scalar processor is effective only to data on cache memory. That means, before considering the use of SIMD instructions, first, you need to put your data on cache.

So, how to put your data on cache? It depends on code design and numerical algorithm itself. We have already explained that small amount of data with the size fitting on the cache has to be accessed repeatedly. Some algorithms fit well on cache architecture using the tiling/blocking technique, while others do not. For example, addition of vector and vector cannot utilize cache at all. Then, in case of matrix–vector product, at least the vector data can be placed on cache, while the matrix side cannot be. Still by applying these cache-aware techniques, the code can be significantly accelerated on a modern cache-based scalar processor, because the amount of data to be read from the main memory can be cut by half. Finally, in case of matrix–matrix product, with cache blocking, the majority of the working data set can be placed on cache, which makes further application of SIMD vectorization very effective. Thus, whether SIMD and cache blocking are effective or not heavily depends on the nature of the numerical algorithm employed and higher level design.

We need to mention about accelerators. The use of accelerators such as GPU and many-core chip is gradually starting. On these environments, special programming languages or extension to existing languages, such as CUDA and OpenCL, may have to be utilized. Directive-based programming models such as OpenACC, which require much less work, are also becoming available.

3.4 Library

Some HPC-related libraries are available. They may have to be employed into your code if necessary.

Assuming the use of a modern distributed memory parallel supercomputer, for inter-node parallelization, message passing interface (MPI) is the primary choice. It is a message passing-based communication library among computational nodes. In case of macro-scale simulation, domain decomposition is required. Especially in case of unstructured grid or mesh-free/particle, identifying the boundary region between subdomains is a bit complicated task. Instead of this really tedious approach, some people have started using PGAS languages. Using these special languages, programmer can parallelize code in much similar way as shared memory environments like OpenMP.

In addition, the knowledge of linear algebra is often required. It is very useful if matrix and vector-related libraries are available. Linear algebraic solver and eigenvalue solver libraries are also important.

Basic linear algebraic subroutines (BLAS) is one of the famous libraries for matrix and vector operations in the basic and fundamental level, such as various kinds of vector–vector, matrix–vector and matrix–matrix operations. The scene behind the fact that recently this library becomes very important is, however, because nowadays highly tuned versions of this library prepared by hardware vendors themselves are available. Especially, level-3 BLAS routines, which are related to matrix–matrix operations, are really important. Usually, inside the vendor-provided library, these routines are implemented as cache-aware and SIMD-vectorized. And, by using these optimized routines, high peak ratio can be easily obtained. Although BLAS is designed for dense matrix, recently, its sparse matrix version becomes available.

As for the linear algebraic solver, a variety of libraries are available. LAPACK contains direct solvers for dense and banded matrices [1]. ScaLAPACK is also available for MPI-parallel environments. In case of sparse solvers, some are direct and others are iterative. SuperLU, MUMPS, PARDISO and WSMP are examples of sparse direct solvers, while PETSc is an example of iterative solver libraries.

3.5 Supporting Environment and Tools

Other than compiler, the most important programming tool for supercomputing is profiler. This tool can be used to identify hot spot of the code. Finding the hot spot is vital in performance-centric software design, described before. Moreover, the profiler can measure how fast your code runs on the supercomputer. That means, measuring FLOPS values.

They say that, "without measurement, no improvement." This is absolutely right. On a supercomputer, the profiler is a survival tool. The concept of peak performance ratio has no meaning if one cannot measure FLOPS accurately. But if there is no such profiler available in your environment, how can you survive? In theory, it is very simple. First, have your program count the number of total floating-point operations executed inside your code. Then, divide this number by the execution time measured in seconds. That's it. Yes, this is floating-point operation per second (FLOPS)! Actually, this task is very tedious as you imagine. That's why the profiler is essential in HPC.

Compared to the profiler, debugger, which is usually another essential tool in more general software development environment, seems to be of little use, at least for us. We mean, although we can often find the situation where the debugger is useful in this environment, it is far from enough. To supplement the debugger, we can suggest two ways for debugging.

One is, to test/debug your code on a single processor and single core environment first, and keep doing it until all the bugs disappear. Virtually, invoking the debugger on the supercomputer is too late to find a bug.

The other is, to prepare additional debugging mechanism embedded in your code explicitly. For example, it is convenient to add the functionality to compare internal data set between single version and parallel one. While it is also a tedious task and even the meaning of data comparison itself depends heavily on each problem, still it is a great help if bugs are somewhere in your code and also such functionality is ready. A cool and fancy parallel debugger would not help you so much.

In addition to these headaches, you might face more mysterious bugs on the supercomputer. Although thread-level parallelization like OpenMP and CUDA helps you write parallel code much easier, it could be often the source of serious bugs, such as synchronization and subtle timing issues. You might also encounter the bug of compiler, profiler, libraries and OS itself. For the most of the cases, however, it will end up just caused by your mere misunderstanding about the actual behaviour of these tools. But, based on our experiences, in very rare occasions, you might actually see these bugs!

Anyway, even on ordinary, single core desktop PC environment, just using the debugger, can you actually find the bug of a minus operator, which is wrong and should be plus instead, from a bunch of formulas in your code? Probably, intensive and thorough code review, and verification test of the code against theoretical or experimental data are required. Thus, in the development of simulation code, the debugging and testing task itself has been very important and critical from long, long time ago. It will be so in near future also, and perhaps forever. It is simply because the debugger does not work well. It is noticed that, on the supercomputer, without thorough and deliberate preparation for debugging, you could see the real hell of debugging. Please be aware of this fact.

3.6 Other Issues

Here, we mention about some issues, advices and programming tips to take advantage of these modern technologies in HPC and supercomputing.

3.6.1 Estimation and Measurement

For any scientific activity, measurement is an important task. Also, prediction and estimation of the behaviour of the target object is useful to establish background theory behind the phenomena. Such kind of so-called Plan (prediction and estimation)—Do (experiment)—See (measurement) cycle is also effective for the performance optimization process of supercomputing applications.

For example, have you ever estimate and measure actually about the following things? How much is the calculation cost and how much is the communication cost? How much data is sent to/received from neighbour nodes? In this loop, how much data is read/written from main memory, while how many FP operations are carried out? Without estimation, there can be no reasoning and theory. And also, without measurement, no justification or correction of them.

In HPC and supercomputing, “how many” is very important. How many times the code becomes faster than before? How large problem the code can handle in this supercomputer? And, in the end, how long does it take to run this job? It is quite different from using other hardware devices and computer environments. For example, to utilize a 3-D graphics device, it is anyway fine if you can draw lines and triangles. The performance is a secondary issue, and it should be handled only if necessary. That means, in 3-D graphics, “can do” is more important than “how many.” However, in case of a supercomputer, merely using this expensive machine has virtually no meaning. As we mentioned before, the sole purpose of using the supercomputer is, making your code faster.

3.6.2 Memory Access Patterns

In modern HPC environments, memory is one of the slowest components, and it can easily become the performance bottleneck. Therefore, it is useful to design your code considering memory access pattern. Nowadays, the execution time of the code can be roughly estimated from memory access cost, rather than the number of FP operations.

Whether to main memory, cache, SSD or hard disk, data access pattern to these memory devices virtually defines the performance. If possible, sequential access is the best. Otherwise, certain amount of penalty should be expected, depending on the type of devices.

Because of memory hierarchy mechanism, automatic data movement between a slow device and a fast one may occur also. Data access patterns considering data locality control the actual behaviour of this mechanism, and you can identify "where is this data actually placed now?"

3.6.3 Implicit or Explicit?

When discussing about issues of any programming language and compiler in HPC and supercomputing, there is the argument of so-called implicit or explicit. In case of implicit, the behaviour of the code is implicitly defined in a certain, hopefully convenient for the programmer, way. On the other hand, in case of explicit, the programmer has to explicitly define the behaviour of the code one by one, line by line.

At a glance, the former one is easier and better. The latter one is usually so tedious and error-prone, people tend to choose the former implicit or automatic approach. The question is, "in the end, did your code actually get faster?" As we mentioned before, using a supercomputer is not just using it, but also, "how fast" your code runs on the supercomputer. In this sense, the implicit approach is something in the middle. Your code may become faster, or not so much. It actually depends. It depends on the design of your code.

It is our opinion that, whether implicit or explicit, the code performance is virtually defined by the design of the code. Sometimes, it heavily depends on the type of numerical algorithm adopted in the code. And the difference between implicit and explicit is, while deep understanding about the implication of the code design is required "before" using the explicit approach, in case of implicit, you can easily jump start into the coding phase without thinking about these design issues. As a result, if lucky, it works fine. Otherwise, and for the most of the cases in our experiences, you are forced to re-consider the design of your code thoroughly from scratch, or just give up and retreat from supercomputing.

Actually, in terms of the productivity, implicit approaches, such as OpenMP and OpenACC, are far more superior than corresponding explicit approaches of much lower level, such as p-thread and CUDA. Same is applied to SIMD vectorization driven by compiler directives, over assembly languages. As for the discussion about MPI versus PGAS languages, we cannot say here, because we don't have so much experience of the latter approaches. In some HPC environments, only implicit approaches are available. In this sense, virtual memory, cache mechanism and NUMA are also classified into implicit approaches.

When an implicit approach is appropriate or the only one available, how to use it? The key point is the thorough understanding of the actual behaviour of this implicit approach, until the level of hardware layer. In case of NUMA, even if it looks like shared memory, it is actually a kind of distributed memory, with each processor having its own memory. In case of SIMD instruction, what kind of assembly code is actually generated by compiler through SIMD vectorization? And also, what kind of memory access pattern dominates the loop? Such level of understanding might be required, if you are unlucky and something is not going well.

3.6.4 Software Design and Performance Optimization

As already mentioned, through this chapter, the concept of performance-centric software design is introduced. The idea is, let's think about the performance tuning task up-front in the design phase of software, rather than mere hacking-like activity after everything finished. And also, we have mentioned just before that, it is necessary to consider some aspects of HPC hardware and software architecture in the design phase. Here in this section, some more know-hows and tips are shown for seeking the performance-centric design.

For example, portability. How to handle this issue in the code development for HPC? In the extreme case, it might be the best to write code in assembly language for each different HPC platform. In terms of portability, this is obviously the worst case. In the modern supercomputing environments, however, it seems to be unavoidable to modify the code for each specific hardware platform and its associated special software development environment. Even programming languages might have to be chosen.

Let's follow a well-known software design principle. The solution is modularization, as always. Identification of hot spot helps to isolate performance-sensitive portions of the code from the rest. Let's move them into the device-dependent part, and optimize only these portions thoroughly for each specific HPC environment. Whatever tools and programming languages, even assembly language can be used in these isolated portions. It is very similar to just maintaining portability among various kinds of OS, network and GUI environment, found usually in the scene of modern software development in other fields. Libraries such as MPI and BLAS can be considered as examples of this case. In the similar way, each application can prepare its own portability layer against platform diversity and future change of hardware.

4 Design and Implementation of Finite Element Code

The finite element method (FEM) is one of the famous approximation methods for solving partial differential equations. Starting from structural mechanics, it has mainly been applied to macro-scale problems in the continuum mechanics field.

As for the history of FEM on supercomputers, until recently, it used to be just enough to tune only the direct solver part of the FE code. With sufficiently large problem size, either band or skyline solver can be easily vectorized on vector processor, or parallelized on shared memory environment.

With the emergence of distributed memory parallel supercomputers, however, things have been drastically changed. Numerical schemes adopted in the applications running on PC cluster and MPP are mainly dominated by either static/implicit time marching schemes using iterative solver as the linear equation solver, or explicit time marching schemes. The whole analysis domain has to be domain-decomposed into multiple subdomains. The use of sparse direct solvers on such a distributed memory parallel environment has just begun recently with a relatively limited number of computational nodes.

Here in this section, related to the implementation of FE code on supercomputers, four topics are explained. Starting from EBE approaches, some issues about linear algebraic solver are described, followed by the domain decomposition method (DDM) as a parallel processing scheme. Finally, the issue of pre- and post-processing in supercomputing applications is briefly described.

4.1 Element-by-Element

Although the typical performance bottleneck, or the hot spot, of a FE code is its linear algebraic solver, the part of forming element-wise matrices and vectors can also be another weak point. In case of an explicit code, this part typically dominates. Even in an implicit or a static code, it can occupy a significant portion of execution time, if non-linearity is strong and stress integration involves relatively heavy calculation, or some of the terms in the weak form are explicitly evaluated. The performance tuning effort of EBE operations is at least not negligible.

As for the performance optimization of the EBE routines, it is easy to parallelize them. Because they are namely EBE operations, each of them can be done independently. The granularity of parallelization can be any, because there will be millions, or perhaps billions of elements to be processed.

Care should be taken for the assembly part, however. To assemble element-wise matrices into a global matrix, and element-wise vectors into a global vector, conflict occurs. Sorting of elements is required to avoid this conflict. In more general, this assembly process corresponds to so-called scatter operation, from elements to nodes. It distributes element data onto the corresponding nodes. The reverse one is "gather" operation, from nodes to elements. It collects nodal data for each element.

While the gather operation is read only operation from a nodal vector, the scatter operation updates a nodal vector. It involves both read and write. When multiple elements try to write their own data into the same nodal vector simultaneously, it easily causes data update conflict. In case of shared memory environment, this conflict must be suppressed by element reordering. Instead, if it is performed to domain-decomposed data structure in distributed memory parallel environment, data exchange occurs along the interface boundary between adjacent subdomains.

In addition to parallelization, what else? Intra-core optimization remains. For example, SIMD vectorization. Assuming that there is no major IF/THEN branch in forming an element-wise value, evaluation of this quantity in multiple elements or multiple integration points in an element can be performed not only in parallel, but also in exactly the same way. This fact naturally leads to SIMD. SIMD/short vectorization can be applied. SIMD implementation on GPU is also possible if a sufficient number of elements, for example, thousands or more, can be allocated in each GPU.

4.2 Linear Algebraic Solver

Unless the scheme of your FE code is purely explicit, it is necessary to prepare a linear algebraic solver to handle the matrix equation $[A]\ \{x\} = \{b\}$. The matrix $[A]$ may be represented as band, skyline or sparse. Here, sparse means storing only non-zero components in a certain way, such as compressed row storage (CRS) format or block CRS one.

If no iterative solver can be employed to solve the linear equation, because the matrix [A] is highly ill-conditioned or some other reasons, direct solver is the only choice. In this case, however, you'd better give up using a supercomputer. Even a small-scale PC cluster may have difficulty in scaling, unless the problem size is really huge. Currently, it can be said that it is very difficult or even impossible for direct solver to take advantage of distributed memory parallel architecture. As for a typical modern supercomputer with hundreds or thousands of computational nodes, either implicit code employing iterative solver or explicit code using EBE operations predominantly is the choice.

It can be said that iterative solver works fine on massively parallel environment, if domain decomposition is properly performed [2]. At least, the matrix–vector product, which is the main body of iterative solver, can be easily parallelized on any parallel environment, from SIMD, share memory to MPP. However, this is the case without pre-conditioner. In some problems, without efficient pre-conditioner, the convergence ratio is very bad. Then, the question is, are there any parallel pre-conditioners available? Currently, not so many pre-conditioners are ready. It is because, the stronger and the more effective the pre-conditioner is, the more it looks like a kind of direct solver. As we explained just before, it is very difficult to parallelize direct solver. Currently, finding a good parallel pre-conditioner is on-going challenge in this research field. Some parallel pre-conditioners based on hierarchical domain decomposition, in a similar way as that of sparse direct solver, are available. Also, multi-grid approaches are effective. Either geometrical multi-grid or algebraic multi-grid (AMG) solver may be used. The third way is, DDM, described in the next section.

Although direct solver alone is almost useless on massively parallel environment, it is still useful for two cases. One is, for a component of much more complicated numerical software stack. The other is, on a desktop workstation with accelerator like GPU or many-core. Basically, direct solver works very well with cache mechanism. It is quite different from iterative solver, which is usually memory bound. Especially, the factorization phase of the direct solver can easily achieve near the peak performance on a single computational node. It also works well on GPU and many-core accelerators. Therefore, it can be utilized as a sub-component inside more sophisticated and complicated solver. Direct solver can be used, for example, as the linear algebraic solver for the coarsest grid in multi-grid solver, as the solver of the local models in non-linear homogenization code, and as the local subdomain and the coarse grid solvers in DDM.

4.3 Domain Decomposition Method

The DDM is a way to solve partial differential equation, by solving each subdomain problem separately [3]. The solution of each subdomain problem requires assumed boundary condition. With the repetition of this process, this, initially assumed boundary condition along subdomain interface gradually converges to the true one. In this sense, it is a kind of iterative method. DDM is inherently parallel because each subdomain local problem can be solved independently [4, 5]. To solve the subdomain local problem, either iterative or direct solver may be used.

Anyway, DDM is different from just decomposing a mesh into multiple sub-domains. This type of domain decomposition reduces the communication cost in matrix–vector product operations, which is typically found in parallel iterative solver and explicit code. Instead, DDM is an independent numerical scheme originated from the mathematical field of partial differential equation. Sometimes, it is also called as Shur complement method or iterative sub-structuring method in the field of iterative solver.

DDM is a kind of hybrid method between iterative solver and direct one. The global view of DDM is iterative, and pre-conditioner may be required in the similar way as the case of iterative solver. Coarse grid solver is typically employed as pre-conditioner. In this case, the coarse grid is derived from the graph structure of domain decomposition itself, and it can be said to be a kind of two grid version in multi-grid solver. To solve the coarse problem and the subdomain local problems, direct solver may also be utilized.

4.4 Pre- and Post-Processing: Where and How?

When a supercomputer is employed to solve a huge-scale problem, its input and output data, namely, the mesh and the result data of such a huge-scale analysis can also be very huge. Then, the pre- and post-processing becomes a critical issue.

Somehow, you need to perform these tedious tasks, but where? Perhaps, on the supercomputer? No way! ... Sorry, it is not a joke. There are two, more specific questions about the issues to handle such a huge-scale analysis. The first one is, where and how is the mesh generated? And the other one is, can we get back such huge result data from the supercomputer onto our desktop workstation, through campus LAN and the Internet?

For the first question, the mesh generator has to be ported onto the supercomputer side. That means, it also has to be fully parallelized. Unfortunately, the development of highly parallel automatic mesh generator is still the on-going research topic. To overcome the issue, instead, mesh refiner may be utilized [6]. This tool simply refines the given initial mesh into half. Speaking more exactly, in case of 3-D solid/volume element, each element is divided into eight elements, by adding a mid-node on each edge. One step refinement increases the number of elements 8 times. Uniform mesh refinement itself can be easily parallelized. Because the refiner can be invoked on the supercomputer, it is sufficient to create an initial coarse mesh on your desktop PC and send it to the supercomputer through slow network.

For the second question about the visualization of the result data, however, things are more complicated. One obvious answer is, generate images and movie data on the supercomputer side [7]. Not only this post-processor simply runs on the supercomputer, it should also be parallelized. Assuming the analysis domain is already decomposed into subdomains, it is possible for each subdomain to generate its own result image independently, and to gather these images and compose a single image, by image convolution techniques. These processes can be parallelized. This software rendering process can be contained as a part of the main analysis process. After a job is completed, you can obtain not only the result data, but also their images and movies. While waiting for the download of the huge result data, you can quickly browse these images and movies.

Well, you may think that this solution is insufficient. We agree to you. Interactivity is missing. To do so, however, there are some challenges. First, how to bring back such a huge data set? Some kinds of data compression techniques are needed. In case of structural analysis, it might be sufficient to extract only the surface portion from the volume data. Part-by-part visualization may be used also. Then, the next issue is, how to handle such a huge data set on a single desktop workstation? Another PC cluster, dedicated for visualization purpose, may be needed. The third question is about the complexity itself of the huge result data. For example, the geometry data of the model also tends to be highly complicated. Just rotating and zooming does not work. Fly-through or walk-through visualization, typically found in the virtual reality field, may be helpful.

References

1. Dongarra, J.J., et al.: Numerical Linear Algebra for High-Performance Computers. SIAM, Philadelphia (1998)
2. Saad, Y.: Iterative Methods for Sparse Linear Systems. SIAM, Philadelphia (2003)
3. Smith, B., et al.: Domain Decomposition: Parallel Multilevel Methods for Elliptical Partial Differential Equations. Cambridge University Press, Cambridge (2004)
4. Bhardwaj, M., et al.: Salinas: A scalable software for high-performance structural and solid mechanics simulations. In: Proceedings of SC02 (2002)
5. Ogino, M., Shioya, R., Kawai, H., Yoshimura, S.: Seismic response analysis of full scale nuclear vessel model with ADVENTURE system on the earth simulator. J. Earth Simul. **2**, 41–54 (2005)
6. Murotani, K., Sugimoto, S., Kawai, H., Yoshimura, S.: Hierarchical domain decomposition with parallel mesh refinement for billions-of-DOF-scale finite element analyses. Int. J. Comput. Methods (2013). doi:10.1142/S0219876213500618
7. Kawai, H., Ogino, M., Shioya, R., Yoshimura, S.: Vectorization of polygon rendering for off-line visualization of a large scale structural analysis with ADVENTURE system on the earth simulator. J. Earth Simul. **9**, 51–63 (2008)

Simulation of Seismic Wave Propagation and Amplification

Muneo Hori, Tsuyoshi Ichimura, and Kohei Fujita

Abstract Large scale simulation of ground motion is a vital tool in engineering seismology and earthquake engineering. Need for high performance computing cannot be underestimated, because increasing the spatial resolution results in computing higher frequency components of ground motion which influence structure response. This chapter explains improvement of discretization for large scale simulation of ground motion simulation based on finite element method. Better mathematical treatment is needed for ground motion simulation since it solves a four-dimensional linear or non-linear wave equation so that mass matrix becomes diagonal. Diagonalization of the mass matrix is achieved by utilizing a set of orthogonal and discontinuous basis functions. While it sounds odd, the use of discontinuous basis functions provides us a larger capability of modeling. As examples of large scale simulation of ground motion, the use of K computer, the best supercomputer in Japan in the year of 2015, is explained. A non-linear finite element method is developed in K computer, so that a model of 10,000,000,000 degree-of-freedom is analyzed with 100,000 time steps in less than a half day. Numerical computation techniques and parallel computation enhancement are explained to realize this simulation, which leads to high scalability of the developed finite element method. As an illustrative example, Tokyo Metropolis is used as a target, and a K computer simulation of ground motion is presented.

M. Hori (✉) • T. Ichimura
Earthquake Research Institute, The University of Tokyo, Tokyo, Japan
e-mail: hori@eri.u-tokyo.ac.jp; ichimura@eri.u-tokyo.ac.jp

K. Fujita
Advanced Institute for Computational Science, RIKEN,
7-1-26 Minatoshima Minamimachi, Chuo-ku, Hyogo 650-0047, Japan
e-mail: kohei.fujita@riken.jp

© Springer International Publishing Switzerland 2016
S. Yoshimura et al. (eds.), *High-Performance Computing for Structural Mechanics and Earthquake/Tsunami Engineering*, Springer Tracts in Mechanical Engineering,
DOI 10.1007/978-3-319-21048-3_2

1 Improvement of Discretization

Earthquake ground motion is strongly influenced by the three-dimensional (3D) underground structure. Therefore, it is a standard practice to employ 3D wave propagation simulation that uses a model of the underground structure. Improving the accuracy of the simulation is a key challenge in seismology research, and various studies have attempted to verify 3D elasto-dynamic codes to improve and confirm the accuracies (e.g., [1]). Because of the simplicity and flexibility, the two most popular techniques for ground motion simulation are the finite difference method (FDM) (e.g., [2–5]) and the finite element method (FEM) (e.g., [6–11]); see also [12] for a brief history of these simulations. For a large and highly heterogeneous target structure, the structured-grid discretization of FDM is attractive. On the other hand, advancements in computing have led to greater acceptance of FEM, even though its computational cost is large; it is capable to analyze a complex underground model more readily than FDM. Moreover, the variable mesh of FEM can be tailored to local wavelengths and treat irregular free surface conditions accurately.

Dynamic explicit FEM is used in engineering fields, and it is often applied to earthquake ground motion simulation owing to its efficient numerical performance (e.g., [6–9]). Explicit FEM that is used in seismology research simply diagonalizes a consistent mass matrix and derives a diagonal lumped mass matrix. This treatment produces non-negligible errors; Belytschko et al. [13] noted “procedures for diagonalizing the mass matrix are quite ad hoc, and there is little theory for these procedures.” For the sake of clarity, we explain the approximations made in determining an element matrix. In a conventional displacement FEM, the element mass matrix is determined using $\rho\,(b^n b^m)_e$, where ρ is the density, b^n is a basis function for the displacement at node n, and $(.)_e$ implies the integration of $(.)$ over an element e, i.e.,

$$(.)_e = \int_e (.) dv.$$

Unless $(b^n b^m)_e$ satisfies $(b^n b^m)_e = 0$ for $n \neq m$, the global element mass matrix will not be diagonal. Some methods approximate the global element matrix by *lumping* it, or convert it into a diagonal matrix (e.g., [13–15]). This approximation is important in reducing the computational cost of applying an explicit time integration scheme. Several researchers have presented detailed discussions of the differences in numerical performance between lumped and consistent mass matrices (e.g., [16]). Because of its efficiency, the lumping approximation is taken for granted in earthquake motion simulation for actual 3D underground structure. However, the accuracy is lost, at least to a certain degree, if the lumping approximation is adapted.

Special finite elements which use orthogonal continuous polynomial basis functions have been proposed. For such elements, an explicit method can be naturally derived with the aid of orthogonality; for example, see [17–19] for recent works in seismology research and fluid dynamics. Komatitsch and Vilotte [17] and Dumbser and Kaser [18] demonstrated the usefulness of consistent lumped-mass matrices of higher-order elements, using a set of higher-degree polynomials. Although higher-order elements are used to compute these responses with high degree of accuracy, there is also a need for lower-order elements to compute the responses of highly heterogeneous bodies. Thus, Matsumoto and Takada [19] proposed a 4-node tetrahedron with 1 bubble node, using a set of complex lower-degree polynomials. The excellent numerical performance of this element demonstrated the usefulness of the lower-order elements with consistent lumped mass matrices. Given this usefulness, standard linear elements (4-node tetrahedral and 8-node hexahedral elements) should also be developed. We propose such standard linear elements for computing earthquake ground motion in a highly heterogeneous body.

In this chapter, we present standard linear elements using *orthogonal discontinuous* basis functions, from which the consistent and diagonalized mass matrices are naturally derived. The numerical characteristics of the mass matrix are categorized as extensions of simple standard finite elements. In Sect. 1.1, we present a formulation of a consistent and diagonalized mass matrix that does not require approximation. We present the detailed configurations for 4-node tetrahedral and 8-node hexahedral elements. The relationship between the proposed finite elements and conventional standard finite elements is investigated. In Sect. 1.2, we compare the solutions obtained from a conventional explicit FEM with analytical solutions in layered media to verify the numerical dispersion caused by the lumping approximation. Comparison of the solutions obtained by using the proposed finite elements with the analytical solutions demonstrates the usefulness of the technique. Examples are also presented to illustrate the effectiveness of the proposed method in earthquake ground motion modeling in the actual 3D crust structure.

1.1 Methodology

The major difference between the present new formulation and the conventional one is the use of orthogonal discontinuous basis functions in discretizing a displacement function. Because it is difficult to satisfy both orthogonality and smoothness, the basis functions we employ are orthogonal but discontinuous. FEM that is obtained by the present new formulation is considered an extension of a mixed FEM [14] with orthogonal basis functions and an FEM with a particle discretization scheme [20].

We start from Hellinger–Reissner's functional, i.e.,

$$J(\mathbf{u},\boldsymbol{\sigma}) = \int_V \int_T -\frac{1}{2}\rho\dot{\mathbf{u}}\cdot\dot{\mathbf{u}} - \frac{1}{2}\boldsymbol{\sigma} : \mathbf{d} : \boldsymbol{\sigma} + \boldsymbol{\sigma} : (\nabla \otimes \mathbf{u}) - \mathbf{u}\cdot\mathbf{f}\, dvdT. \qquad (1)$$

Here, $\mathbf{u}$ and $\boldsymbol{\sigma}$ are displacement and stress, ρ and $\mathbf{d}$ are the density and the compliance tensor (the inverse of the elasticity tensor $\mathbf{c}$), $\mathbf{f}$ is a body force, and

$\dot{(.)}$ and $\nabla(.)$ are temporal and spatial differential operators for (.). Various finite elements have been proposed using this functional (e.g., a *mixed* method in [14]).

Next, we discretize both the displacement and stress functions in J. To this end, we consider an element e, in which ϕ^α and ψ^β are employed as basis functions for $\mathbf{u}$ and $\boldsymbol{\sigma}$, i.e.,

$$\mathbf{u} = \sum_\alpha \mathbf{u}^\alpha \phi^\alpha \quad \text{and} \quad \boldsymbol{\sigma} = \sum_\beta \mathbf{s}^\beta \psi^\beta.$$

Here, $\mathbf{u}^\alpha$ and $\mathbf{s}^\beta$ are unknown coefficients of discretization. Substituting these equations into Eq. (1) and using stationary conditions in place of $\mathbf{u}^\alpha$ and $\mathbf{s}^\beta$, we derive the following matrix equation for $\mathbf{u}^\alpha$ and $\mathbf{s}^\beta$:

$$\mathbf{K}\mathbf{u} + \mathbf{M}\ddot{\mathbf{u}} = \mathbf{F}, \tag{2}$$

where $\mathbf{u}$ is a vector obtained by assembling $\mathbf{u}^\alpha$, and $\mathbf{K}$ and $\mathbf{M}$ are the assembly of the element stiffness and mass matrices, $\mathbf{K}_e$ and $\mathbf{M}_e$, respectively. These are computed as

$$\mathbf{K}_e = (\psi^\beta \nabla \phi^\alpha)_e^T \cdot (\psi^\beta \psi^{\beta'})_e^{-1} \mathbf{c} \cdot (\psi^{\beta'} \nabla \phi^{\alpha'})_e$$

and

$$\mathbf{M}_e = \rho(\phi^\beta \phi^{\beta'})_e.$$

The superscripts in $\mathbf{K}_e$ denote the transpose (T) and inverse (-1). Finally, $\mathbf{F}$ is a vector for the coefficients of $\mathbf{f}$ when $\{\phi^\alpha\}$ are used for the discretization.

While Eq. (2) holds for any choice of the basis functions, we restrict our attention to the special case that the set of $\{\phi^\alpha\}$ is orthogonal and the $\{\psi^\beta\}$ is not orthogonal but smooth. We use the Heaviside step function, H, to form the orthogonal basis functions. As will be shown later, even though the basis functions made from H are not continuous, we can form a suitable element stiffness matrix with an element mass matrix that is diagonal.

We next study the cases of a 4-node tetrahedron and an 8-node hexahedron element, denoted by TET4D and HEX8D, respectively. The configuration of TET4D and HEX8D is displayed in Fig. 1, where (x, y, z) is the global coordinate and (r_1, r_2, r_3) is a local coordinate. The numbers in italics indicate the node. For simplicity, we denote by TET4 and HEX8 an ordinary 4-node tetrahedron element and an 8-node hexahedron element, respectively. TET4 has a linear basis function of displacement, while HEX8 uses tri-linear functions.

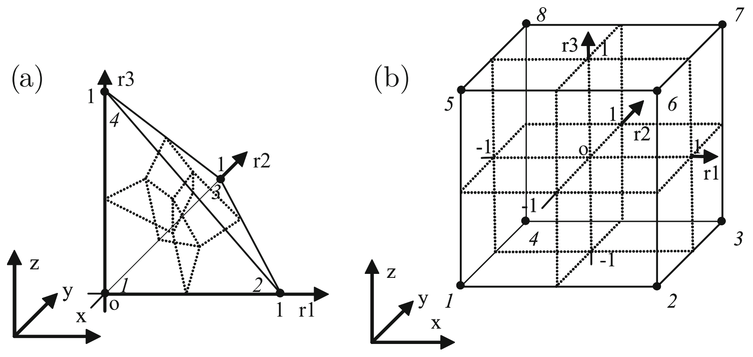


Fig. 1 Configuration of elements. *Dotted lines* indicate interfaces between Voronoi blocks. (**a**) 4-node tetrahedra element, (**b**) 8-node hexahedra element

Table 1 Table of $\bar{pl}_1, \bar{pl}_2$ and $\bar{pl}_3$ for 4-node tetrahedra element, where $pl_1 = r_1 + r_2 + 2r_3 - 1$, $pl_2 = 2r_1 + r_2 + r_3 - 1$, $pl_3 = r_1 + 2r_2 + r_3 - 1$, $pl_4 = r_1 - r_3$, $pl_5 = r_1 - r_2$, and $pl_6 = r_2 - r_3$

α	$\bar{pl}_1$	$\bar{pl}_2$	$\bar{pl}_3$
1	$-pl_1$	$-pl_2$	$-pl_3$
2	pl_2	pl_4	pl_5
3	pl_3	$-pl_5$	pl_6
4	pl_1	$-pl_4$	$-pl_6$

1.1.1 4-Node Tetrahedron Element

The basis functions of TET4D are as follows: four piece-wise constant functions in each Voronoi block are used for $\{\phi^\alpha\}$ and one function for $\{\psi^\beta\}$, i.e.,

$$\begin{cases} \phi^\alpha = H(\bar{pl}_1)H(\bar{pl}_2)H(\bar{pl}_3) \ \ (\alpha = 1, 2, 3, 4), \\ \psi^1 = 1, \end{cases}$$

where H is the Heaviside function and $\bar{pl}_i$ is defined in Table 1. A Voronoi block is defined as the region of space closer to its node than to any other node. Thus, $\{\phi^\alpha\}$ is orthogonal, but discontinuous.

For TET4D, $\mathbf{K}_e$ is identical to that of TET4. However, $\mathbf{M}_e$ becomes diagonal, and the diagonal terms are $\rho/4\,(1)_e$. Thus, $\mathbf{M}_e$ is identical to the lumped element mass matrix that is obtained by approximation for TET4.

1.1.2 8-Node Hexahedron Element

The basis functions of HEX8D are chosen as follows: eight piece-wise constant functions in each Voronoi block are used for $\{\phi^\alpha\}$, and eight tri-linear functions for $\{\psi^\beta\}$, i.e.,

$$\begin{cases} \phi^\alpha = H(\bar{r}_1 r_1)H(\bar{r}_2 r_2)H(\bar{r}_3 r_3) \ \ (\alpha = 1, 2, \ldots, 8), \\ \psi^1 = 1, \quad \psi^2 = r_1, \quad \psi^3 = r_2, \quad \psi^4 = r_3, \\ \psi^5 = r_1 r_2, \quad \psi^6 = r_1 r_3, \quad \psi^7 = r_2 r_3, \end{cases} \tag{3}$$

see Table 2 for $\bar{r}_i$. By definition, the $\mathbf{M}_e$ of HEX8D becomes diagonal, with terms $\rho/8\,(1)_e$. This $\mathbf{M}_e$ is identical to the lumped element mass matrix that is used for HEX8D. Unlike TET4D, however, the $\mathbf{K}_e$ of HEX8D is not identical to that of HEX8. The cost of computing $\mathbf{K}_e$ for HEX8D, which involves the inversion of $(\psi^\beta \psi^{\beta'})_e$, is also higher than that of computing $\mathbf{K}_e$ for HEX8. This disadvantage can be overcome through the use of structured elements, such as voxels or hybrids of voxels and unstructured tetrahedrons; see, e.g., [7, 8, 10, 11].

Let us now clarify the numerical performance of HEX8D, which is somewhat better than that of HEX8, as will be shown later. If we use $\psi = 1$ like TET4, the resulting stiffness matrix is identical to HEX8 with one point integration. Consequently, it is essential to use functions with order greater than 1 for ψ^β to avoid hourglass modes. In this context, we use Eq. (3) to avoid hourglass modes and derive appropriate hourglass control terms. $\mathbf{K}_e$ of HEX8D for a voxel the size of which is ds can be decomposed as

$$\mathbf{K}_e = \sum_{i=1}^{7} \gamma_i \, (\mathbf{h}_i)^T \, \mathbf{c} \, \mathbf{h}_i,$$

see Table 3 for γ_i and $\mathbf{h}_i$. (γ_1 $(\mathbf{h}_1)^T$ $\mathbf{c}$ $\mathbf{h}_1$) of $\mathbf{K}_e$ arises from the constant term in ψ^β and is identical to HEX8 with one point integration. The other terms, $\sum_{i=2}^{4} \gamma_i \, (\mathbf{h}_i)^T \, \mathbf{c} \, \mathbf{h}_i$ and $\sum_{i=5}^{7} \gamma_i \, (\mathbf{h}_i)^T \, \mathbf{c} \, \mathbf{h}_i$, are considered to be stabilization terms against hourglass mode I in Fig. 2a and mode II in Fig. 2b. On the basis of this

Table 2 Table of $\bar{r}_1$, $\bar{r}_2$, and $\bar{r}_3$ for 8-node hexahedra element

α or β	$\bar{r}_1$	$\bar{r}_2$	$\bar{r}_3$
1	−1	−1	−1
2	1	−1	−1
3	1	1	−1
4	−1	1	−1
5	−1	−1	1
6	1	−1	1
7	1	1	1
8	−1	1	1

Table 3 Coefficients and vectors for element stiffness matrix of voxel element whose size is *ds*

$\begin{cases} \gamma_1 = \frac{1}{ds}, \\ \gamma_2 = \gamma_3 = \gamma_4 = \frac{3}{64ds}, \\ \gamma_5 = \gamma_6 = \gamma_7 = \frac{9}{256ds}, \end{cases}$	$(\mathbf{h}_1)^T = \begin{pmatrix} -1 & 0 & 0 & -1 & 0 & -1 \\ 0 & -1 & 0 & -1 & -1 & 0 \\ 0 & 0 & -1 & 0 & -1 & -1 \\ 1 & 0 & 0 & -1 & 0 & -1 \\ 0 & -1 & 0 & 1 & -1 & 0 \\ 0 & 0 & -1 & 0 & -1 & 1 \\ 1 & 0 & 0 & 1 & 0 & -1 \\ 0 & 1 & 0 & 1 & -1 & 0 \\ 0 & 0 & -1 & 0 & 1 & 1 \\ -1 & 0 & 0 & 1 & 0 & -1 \\ 0 & 1 & 0 & -1 & -1 & 0 \\ 0 & 0 & -1 & 0 & 1 & -1 \\ -1 & 0 & 0 & -1 & 0 & 1 \\ 0 & -1 & 0 & -1 & 1 & 0 \\ 0 & 0 & 1 & 0 & -1 & -1 \\ 1 & 0 & 0 & -1 & 0 & 1 \\ 0 & -1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & -1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ -1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & -1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & -1 \end{pmatrix},$

(continued)

Table 3 (continued)

$$(\mathbf{h}_2)^T=\begin{pmatrix}0&0&0&1&0&1\\0&1&0&0&1&0\\0&0&1&0&1&0\\0&0&0&-1&0&-1\\0&-1&0&0&-1&0\\0&0&-1&0&-1&0\\0&0&0&1&0&-1\\0&1&0&0&-1&0\\0&0&-1&0&1&0\\0&0&0&-1&0&1\\0&-1&0&0&1&0\\0&0&1&0&-1&0\\0&0&0&1&0&-1\\0&1&0&0&-1&0\\0&0&-1&0&1&0\\0&0&0&-1&0&1\\0&-1&0&0&1&0\\0&0&1&0&-1&0\\0&0&0&1&0&1\\0&1&0&0&1&0\\0&0&1&0&1&0\\0&0&0&-1&0&-1\\0&-1&0&0&-1&0\\0&0&-1&0&-1&0\end{pmatrix},\ (\mathbf{h}_3)^T=\begin{pmatrix}1&0&0&0&0&1\\0&0&0&1&1&0\\0&0&1&0&0&1\\-1&0&0&0&0&1\\0&0&0&-1&1&0\\0&0&1&0&0&-1\\1&0&0&0&0&-1\\0&0&0&1&-1&0\\0&0&-1&0&0&1\\-1&0&0&0&0&-1\\0&0&0&-1&-1&0\\0&0&-1&0&0&-1\\1&0&0&0&0&-1\\0&0&0&1&-1&0\\0&0&-1&0&0&1\\-1&0&0&0&0&-1\\0&0&0&-1&-1&0\\0&0&-1&0&0&-1\\1&0&0&0&0&1\\0&0&0&1&1&0\\0&0&1&0&0&1\\-1&0&0&0&0&1\\0&0&0&-1&1&0\\0&0&1&0&0&-1\end{pmatrix},\ (\mathbf{h}_4)^T=\begin{pmatrix}1&0&0&1&0&0\\0&1&0&1&0&0\\0&0&0&0&1&1\\-1&0&0&1&0&0\\0&1&0&-1&0&0\\0&0&0&0&1&-1\\-1&0&0&-1&0&0\\0&-1&0&-1&0&0\\0&0&0&0&-1&-1\\1&0&0&-1&0&0\\0&-1&0&1&0&0\\0&0&0&0&-1&1\\-1&0&0&-1&0&0\\0&-1&0&-1&0&0\\0&0&0&0&-1&-1\\1&0&0&-1&0&0\\0&-1&0&1&0&0\\0&0&0&0&-1&1\\1&0&0&1&0&0\\0&1&0&1&0&0\\0&0&0&0&1&1\\-1&0&0&1&0&0\\0&1&0&-1&0&0\\0&0&0&0&1&-1\end{pmatrix},$$

(continued)

Table 3 (continued)

$$(\mathbf{h}_5)^T=\begin{pmatrix}
0&0&0&0&0&-1\\
0&0&0&0&-1&0\\
0&0&-1&0&0&0\\
0&0&0&0&0&1\\
0&0&0&0&1&0\\
0&0&1&0&0&0\\
0&0&0&0&0&-1\\
0&0&0&0&-1&0\\
0&0&-1&0&0&0\\
0&0&0&0&0&1\\
0&0&0&0&1&0\\
0&0&1&0&0&0\\
0&0&0&0&0&1\\
0&0&0&0&1&0\\
0&0&1&0&0&0\\
0&0&0&0&0&-1\\
0&0&0&0&-1&0\\
0&0&-1&0&0&0\\
0&0&0&0&0&1\\
0&0&0&0&1&0\\
0&0&1&0&0&0\\
0&0&0&0&0&-1\\
0&0&0&0&-1&0\\
0&0&-1&0&0&0
\end{pmatrix},\quad
(\mathbf{h}_6)^T=\begin{pmatrix}
0&0&0&-1&0&0\\
0&-1&0&0&0&0\\
0&0&0&0&-1&0\\
0&0&0&1&0&0\\
0&1&0&0&0&0\\
0&0&0&0&1&0\\
0&0&0&-1&0&0\\
0&-1&0&0&0&0\\
0&0&0&0&-1&0\\
0&0&0&1&0&0\\
0&1&0&0&0&0\\
0&0&0&0&1&0\\
0&0&0&1&0&0\\
0&1&0&0&0&0\\
0&0&0&0&1&0\\
0&0&0&-1&0&0\\
0&-1&0&0&0&0\\
0&0&0&0&-1&0\\
0&0&0&1&0&0\\
0&1&0&0&0&0\\
0&0&0&0&1&0\\
0&0&0&-1&0&0\\
0&-1&0&0&0&0\\
0&0&0&0&-1&0
\end{pmatrix},\quad
(\mathbf{h}_7)^T=\begin{pmatrix}
-1&0&0&0&0&0\\
0&0&0&-1&0&0\\
0&0&0&0&0&-1\\
1&0&0&0&0&0\\
0&0&0&1&0&0\\
0&0&0&0&0&1\\
-1&0&0&0&0&0\\
0&0&0&-1&0&0\\
0&0&0&0&0&-1\\
1&0&0&0&0&0\\
0&0&0&1&0&0\\
0&0&0&0&0&1\\
1&0&0&0&0&0\\
0&0&0&1&0&0\\
0&0&0&0&0&1\\
-1&0&0&0&0&0\\
0&0&0&-1&0&0\\
0&0&0&0&0&-1\\
1&0&0&0&0&0\\
0&0&0&1&0&0\\
0&0&0&0&0&1\\
-1&0&0&0&0&0\\
0&0&0&-1&0&0\\
0&0&0&0&0&-1
\end{pmatrix}.$$

clarification, HEX8D is categorized as a sort of HEX8 with one point integration and hourglass control along conventional lines (e.g., [21, 22]). Note that these hourglass control terms can be rigorously derived in our formulation.

1.1.3 Discontinuous Basis Function

The discretization scheme presented in the above may seem odd, because a discretized function is neither continuous nor differentiable and does not belong to the ordinary H1 Sobolev space for the problem. However, a solution obtained in terms of $\{\phi^\alpha\}$ is the *closest* to a solution in H1; the closeness is in the following sense: for a given function $f(\mathbf{x})$, the discretization scheme determines the coefficients by minimizing

$$E(f^1,f^2,\ldots)=\int\left(f-\sum_{\alpha}f^\alpha\phi^\alpha\right)^2\,dx.$$

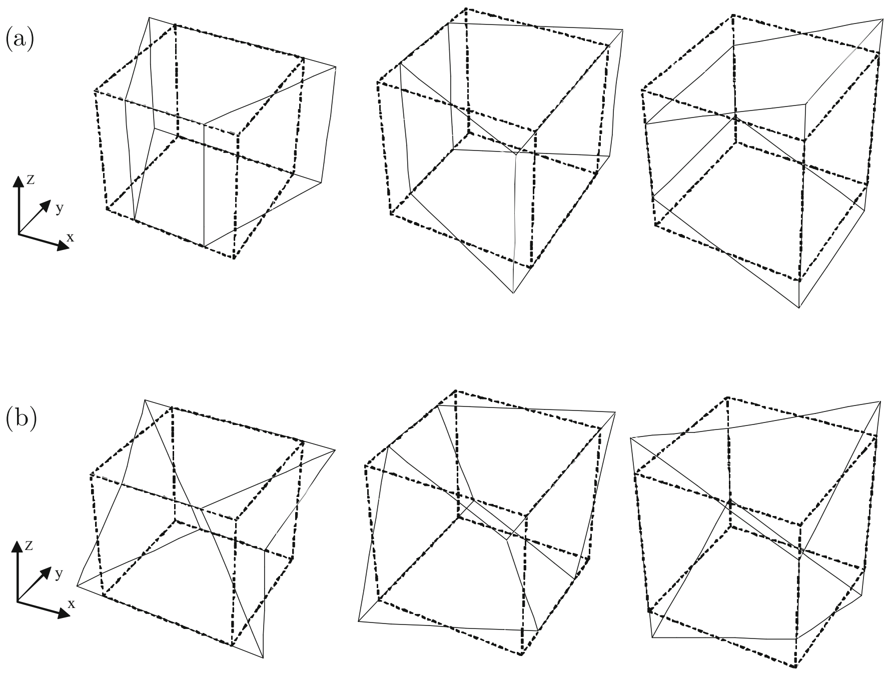


Fig. 2 Configuration of hourglass modes. *Solid* and *dotted line* indicates voxel and hourglass mode, respectively. (**a**) Hourglass modes I, (**b**) hourglass modes II

A key issue is that a function closest to the solution (which is continuous and differentiable) can be found in a space of continuous and differentiable functions.

Raised is a question regarding how close a function obtained by the present method (i.e., the numerical solution) is to the exact solution. It has not yet been proved in general that the above E vanishes at the limit as the number of ϕ^α increases infinitely. However, it has been rigorously demonstrated that the numerical solution coincides with a solution of an ordinary FEM with linear elements. In the numerical experiments presented here, it is shown that the numerical solution is quantitatively close to the exact solution and the discontinuity inherent to the numerical solution does not provide fatal error.

1.2 Numerical Experiment

We perform simulation of earthquake ground motion which is excited by a single point source in horizontally layered media and in an actual 3D underground structure, in order to test the numerical performance of the proposed elements. The conventional FEM and the proposed orthogonal FEM are designated by CFEM and OFEM, respectively, and the configurations of the two FEM's are summarized as follows:

- CFEM: unstructured TET4 and structured HEX8 (cubic). A simple lump approximation is applied to the element mass matrix.
- OFEM: unstructured TET4D and structured HEX8D (cubic).

Although the difference between CFEM and OFEM is the element stiffness matrix of the cubic element, we expect that the numerical dispersion of OFEM is less than that of CFEM because the stiffness matrix of HEX8D is consistent with a lumped mass matrix. Additionally, because the element stiffness matrices of TET4D are identical to those of TET4 and the stiffness matrices of structured HEX8D are stored in memory, the computational costs of OFEM are identical to those of the CFEM. We should point out that all numerical techniques applied to CFEM (e.g., parallel computation) are applicable to OFEM.

The target matrix equation is

$$\mathbf{Ku} + \mathbf{C}\dot{\mathbf{u}} + \mathbf{M}\ddot{\mathbf{u}} = \mathbf{F},$$

where $\mathbf{C}$ is Rayleigh's damping, the assembly of $\alpha\mathbf{M}_e + \beta\mathbf{K}_e$ with α and β being determined as [8]. Unstructured 4-node tetra and structured 8-node cubic elements are automatically generated for 3D models using [11], as long as more than 10 elements are used for one wavelength [6–8, 11]. We use the same mesh model in both CFEM and OFEM. The point source is implemented as an equivalent body force. To remove the outgoing waves, absorbing boundary conditions are applied to all surfaces except the top [23]. A two-step Adams-Bashforth method is applied for the time integration, and frequency components outside of the target frequency domain are filtered out from the solution.

1.2.1 Horizontal Layered Problem

Table 4 summarizes the problem setup, and Figs. 3 and 4 present a 3D model and a close-up view of the generated mesh. Observation points, denoted by obs$_i$ ($i = 1, 2, \sim 14$), are set on the surface at coordinates $(x, y) = (-37800 + 1800i, -37800 + 1800i)$.

Figure 5 compares a CFEM solution and an analytic solution [24] obtained by computing Green's function (GF). There is close agreement between the two solutions, although small differences are seen at observation points located far from the source. To estimate the total difference, we used the quantitative misfit criteria, em and pm, which indicate the misfit in terms of the envelope and phase, respectively [25]. Table 5 presents misfit criteria em and pm between the two solutions. As can be seen, they increased as the waves propagated.

Figure 6 shows wave profiles of an OFEM solution and the analytic solution. As can be seen, the agreement is also close in this case. Table 6 presents misfit criteria

Table 4 Horizontally layered half space problem setting

Time duration	40.96 s
Target frequency	≤ 1.0 Hz
Domain size	$-86.4\,\text{km} \leq x \leq 86.4\,\text{km}$
	$-86.4\,\text{km} \leq y \leq 86.4\,\text{km}$
	$-72\,\text{km} \leq z \leq 0\,\text{km}$
Depth of layer	3.6 km
Layer	$\rho = 2500\,\text{kg/m}^3$
	$V_p = 3900\,\text{m/s}$
	$V_s = 2250\,\text{m/s}$
	$Q = 500$
Half space	$\rho = 3000\,\text{kg/m}^3$
	$V_p = 7800\,\text{m/s}$
	$V_s = 4500\,\text{m/s}$
	$Q = 500$
Excitation	$M_0\ (2\,t^2/T_0^2) \quad 0 \leq t \leq T_0/2$
	$M_0\ (1\text{–}2\,(t - T_0)^2/T_0^2) \quad T_0/2 \leq t \leq T_0$
	$M_0 \qquad T_0 \leq t$
Strike, dip, rake	30°, 40°, 0°
Magnitude (M_0)	1.0×10^{18} Nm
Rise time (T_0)	2 s
Source location	(−35887.5 m, −35887.5 m, −1687.5 m)

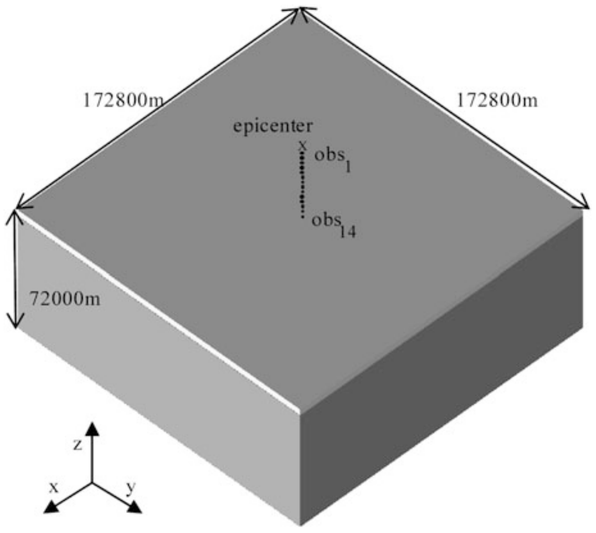


Fig. 3 3D model of horizontally layered half space problem

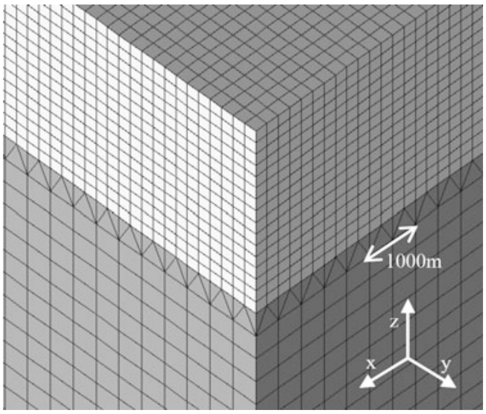


Fig. 4 Mesh configuration of horizontally layered half space problem. Number of nodes, tetrahedron elements, and hexahedron elements are 33,617,834, 12,681,216, and 30,965,760

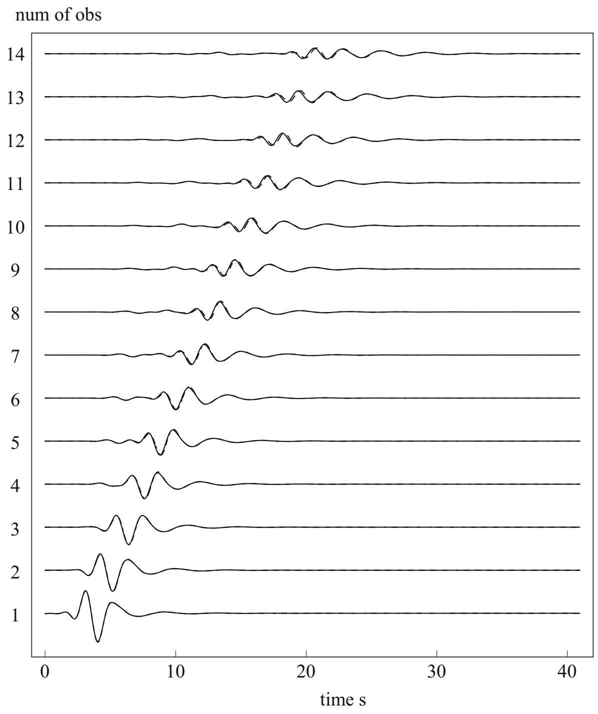


Fig. 5 Comparison of wave velocity profiles in z direction at each observation point, where *dotted* and *solid line* indicates CFEM and GF solutions

between the OFEM solution and the GF solution. It is seen that agreement of OFEM and GF is closer, compared with that of CFEM and GF. The misfits barely increased as the waves propagated.

Table 5 Misfits % of CFEM and GF solutions in each direction at obs$_i$ for horizontally layered half space problem

obs$_i$	em			pm		
i	x	y	z	x	y	z
1	1.34	1.45	1.71	0.71	0.56	0.95
2	2.62	1.07	2.37	1.51	0.45	1.71
3	2.75	4.21	3.05	2.11	1.64	2.46
4	2.79	5.45	4.29	2.05	3.84	3.62
5	3.31	5.58	5.12	2.26	5.21	4.74
6	4.69	6.66	5.66	3.32	4.07	5.16
7	4.68	7.03	6.17	3.30	5.87	5.82
8	3.82	8.12	7.56	2.23	7.65	7.36
9	6.58	8.57	8.21	4.16	8.36	8.32
10	8.12	7.51	8.96	7.65	7.24	8.66
11	7.41	7.72	10.20	8.18	7.36	10.61
12	10.93	8.98	13.62	8.93	8.48	13.65
13	10.20	9.27	17.77	11.54	7.64	15.92
14	10.60	7.68	37.47	12.30	4.15	16.69

Target frequency domain is [0.05 Hz, 1.0 Hz]

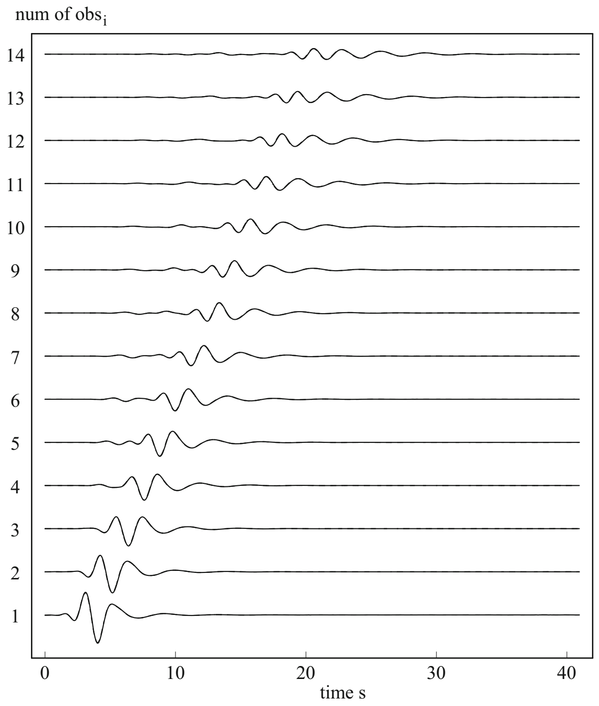


Fig. 6 Comparison of wave velocity profiles in z direction at each observation point, where *dotted* and *solid line* indicates OFEM and GF solutions

Table 6 Misfits % of OFEM and GF solutions in each direction at obs$_i$ for horizontally layered half space problem

obs$_i$	em			pm		
i	*x*	*y*	*z*	*x*	*y*	*z*
1	1.94	1.03	2.54	0.67	1.24	1.25
2	1.43	2.63	2.22	0.76	1.34	1.45
3	1.21	2.94	1.92	1.31	1.46	1.50
4	1.49	2.40	2.29	1.48	1.03	1.79
5	2.01	2.61	2.13	1.64	1.43	1.71
6	2.09	4.27	1.68	1.61	1.19	2.00
7	1.42	3.24	1.85	1.73	0.95	2.16
8	3.17	2.35	2.12	1.86	1.44	2.31
9	3.17	2.76	1.58	1.77	2.10	2.41
10	2.44	4.01	1.68	1.64	2.35	2.73
11	3.71	3.20	1.83	1.87	2.07	3.03
12	4.91	2.49	2.48	1.66	2.25	3.25
13	4.78	2.65	2.03	1.95	2.58	3.50
14	3.68	3.51	3.65	2.18	2.52	4.34

Target frequency domain is [0.05 Hz, 1.0 Hz]

Figure 7 shows wave profiles at obs$_1$ and obs$_{14}$. At obs$_1$, which is located near the source, the difference among CFEM, OFEM, and GF is small, although some errors are caused by the implementation of the point source as an equivalent body force. At obs$_{14}$, which is located far from the source, the difference between OFEM and GF is small. However, the difference between CFEM and GF is large. There is surely no significant difference in the first part of the waveform. However, the difference grows in the latter part and the waveforms break. This is because the numerical dispersion of CFEM increases as the waves propagate. These results demonstrate the usefulness of OFEM as well as the effect of approximation made for the diagonalization of a mass matrix in CFEM on the numerical accuracy.

1.2.2 Actual 3D Underground Structure

Next, we simulate earthquake ground motion from a single point source in an actual 3D underground structure. This analysis is conducted using both CFEM and OFEM to examine whether any differences in accuracy arise. Table 7 summarizes the problem setup, and Figs. 8 and 9 show a 3D model and a close-up view of the generated mesh. The 3D model is constructed based on [26]. Observation points obs$_i$ ($i = 1, 2, \sim 13$) are set on the top surface at coordinates $(x, y) = (2000i, 2000i)$.

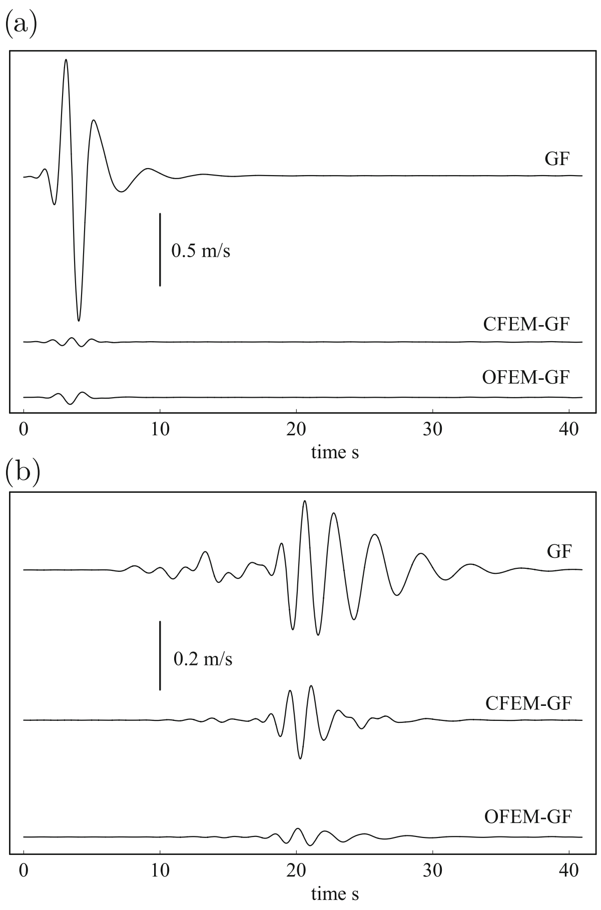


Fig. 7 Comparison of wave velocity profiles. (**a**) Wave profiles in z direction at obs_1, (**b**) wave profiles in z direction at obs_{14}

The misfit criteria (em and pm) between a CFEM solution and an OFEM solution at each observation point are summarized in Table 8. As the distance of the observation point from the source point increases, the difference tends to become larger. Because the wave path of the present underground structure is more complicated than in the horizontal layered media, the distribution of misfit criteria is not simple. However, it is seen that the difference increases as the waves propagated. Additionally, because the wave path is not simple, the difference increased locally at points close to the source. Moreover, the EM and PM maximum values are both about 10. Because this value is in the range of the numerical verification problem, significant differences in the accuracy occur in the earthquake ground motion simulation in an actual 3D crust. Wave profiles at obs_1 and obs_{13} are shown in Fig. 10. Although there is no large difference in

Table 7 Actual 3D crust structure problem setting

Time duration	40.96 s
Target frequency	$\leq$0.5 Hz
Domain size	$-28.8\,\text{km} \leq x \leq 28.8\,\text{km}$
	$-28.8\,\text{km} \leq y \leq 28.8\,\text{km}$
	$-30\,\text{km} \leq z \leq 0\,\text{km}$
First layer	$\rho = 2000\,\text{kg/m}^3$
	$V_p = 2000\,\text{m/s}$
	$V_s = 1000\,\text{m/s}$
	$Q = 500$
Second layer	$\rho = 2000\,\text{kg/m}^3$
	$V_p = 2500\,\text{m/s}$
	$V_s = 1200\,\text{m/s}$
	$Q = 600$
Third layer	$\rho = 2300\,\text{kg/m}^3$
	$V_p = 3900\,\text{m/s}$
	$V_s = 2000\,\text{m/s}$
	$Q = 1000$
Fourth layer	$\rho = 2500\,\text{kg/m}^3$
	$V_p = 4500\,\text{m/s}$
	$V_s = 2500\,\text{m/s}$
	$Q = 1250$
Fifth layer	$\rho = 2600\,\text{kg/m}^3$
	$V_p = 5000\,\text{m/s}$
	$V_s = 3000\,\text{m/s}$
	$Q = 1500$
Excitation	$\text{M}_0\,(2\,t^2/T_0^2) \quad 0 \leq t \leq T_0/2$
	$\text{M}_0\,(1\text{–}2\,(t - T_0)^2/T_0^2) \quad T_0/2 \leq t \leq T_0$
	$\text{M}_0 \qquad T_0 \leq t$
Strike, dip, rake	30°, 40°, 0°
Magnitude (M_0)	1.0×10^{18} Nm
Rise time (T_0)	2 s
Source location	(200 m, 200 m, −5000 m)

the first part of the wave profile and at the observation point located near the source, the difference increases due to numerical dispersion as the wave propagates. The accuracy of the analysis result differs by approximately 20 % of the velocity amplitude.

These results support the usefulness of OFEM in earthquake ground motion simulations. Recent research [27] has clarified the effects of surface geometry on seismic motion. Because FEM readily analyzes a body of complex structures, it is often used for numerical simulations of long-period earthquake motion. However, as demonstrated here, if the observation points are far from the source point in terms of

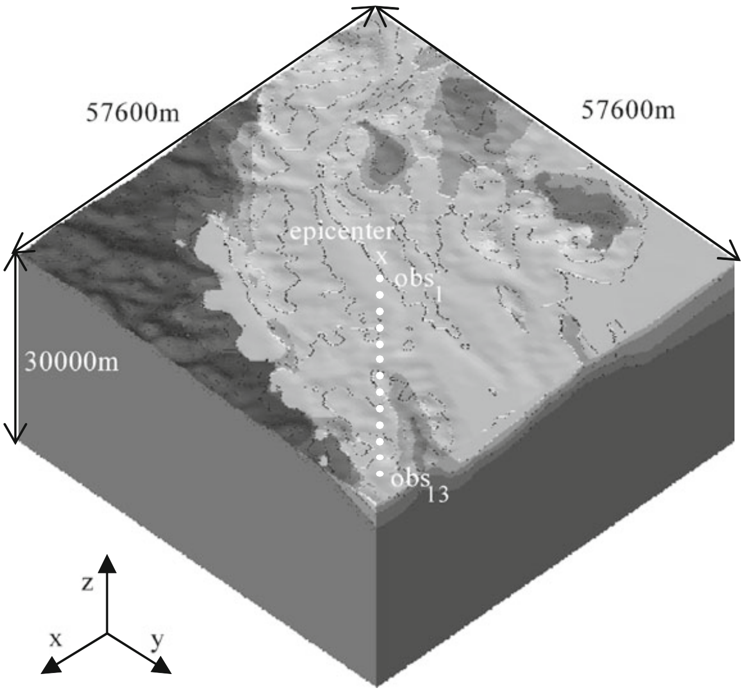


Fig. 8 3D model of actual crust structure problem

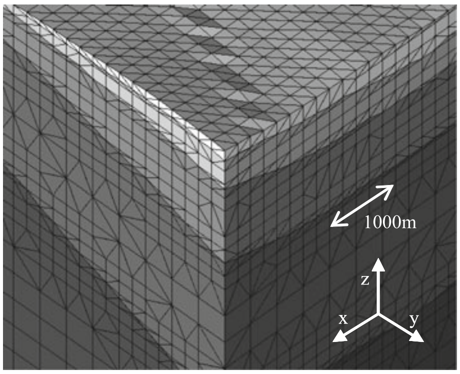


Fig. 9 Mesh configuration of actual crust structure problem. Number of nodes, tetrahedron elements, and hexahedron elements are 3,122,675, 4,643,138, and 2,257,905

wavelength, large errors in the envelope and phase may occur. Numerical simulation of OFEM is more accurate compared with SFEM, and it is expected that OFEM will contribute to estimate the long period earthquake ground motion.

Table 8 Misfits % of CFEM and OFEM solutions in each direction at obs$_i$ for actual 3D crust structure problem

obs$_i$	em			pm		
i	x	y	z	x	y	z
1	2.32	5.37	3.95	1.43	2.89	2.22
2	3.42	4.94	3.69	2.27	3.28	2.65
3	7.32	3.98	4.25	3.73	3.26	3.53
4	6.57	3.67	7.80	5.34	3.95	4.49
5	3.71	4.61	6.73	4.25	4.80	7.47
6	4.62	4.83	8.99	4.39	4.96	6.93
7	7.26	9.23	6.93	6.87	6.68	7.10
8	8.41	9.61	8.50	6.99	8.39	10.21
9	6.01	7.91	6.96	6.18	8.02	6.62
10	7.17	9.83	10.84	6.30	9.14	8.74
11	8.04	7.70	7.48	9.90	8.20	8.86
12	7.97	6.85	7.94	9.44	6.67	12.04
13	11.13	7.53	6.54	9.83	6.78	6.64

Target frequency domain is [0.05 Hz, 0.5 Hz]

1.3 Summary

This chapter develops an FEM with a diagonalized consistent mass matrix, using discontinuous orthogonal basis functions. The presented FEM formulation reduces numerical dispersion by generating a lumped element mass matrix without making any approximation. Furthermore, we present a detailed formulation of 4-node tetrahedra and 8-node hexahedra and verify their numerical performance for wave modeling through experiments on earthquake ground motion. The results of the numerical experiments reveal large differences between the conventional FEM solution and the analytical solution at points far from the point source because of the effect of approximation on diagonalizing the mass matrix. Moreover, the numerical experiments indicate that higher accuracy is obtained from the proposed FEM than from the conventional FEM. Earthquake ground motion simulations in an actual 3D underground structure are conducted as an example of the proposed method. The results show significant differences in accuracy between the conventional and proposed FEM's. Because recent research has shown that earthquake ground motion is significantly affected by surface geometry, explicit FEM that can handle complex surface geometry is of greater importance. Thus, the proposed FEM is expected to contribute to the field of earthquake ground motion simulation.

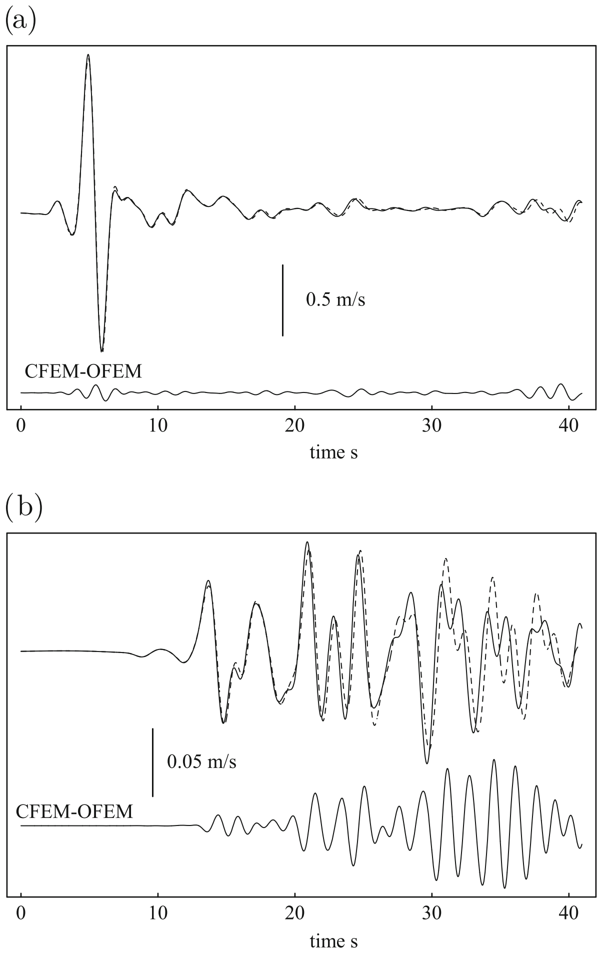


Fig. 10 Comparison of wave velocity profiles where *solid* and *dotted line* indicates OFEM and CFEM solutions. CFEM-OFEM is the difference between CFEM and OFEM solutions. (**a**) wave profiles in x direction at obs$_1$, (**b**) wave profiles in x direction at obs$_{13}$

2 Ground Motion Simulation

A large-scale earthquake is a menace to modern civilization, since it has a potential of literally destroying a densely populated urban area. A recent harrowing example is 2011 Great East Japan Earthquake, which has produced long-lasting damage to the eastern part of Japan. An earthquake of a similar scale is likely to hit another mega cities, such as San Francisco, Seattle, Istanbul, or Tokyo, in the near future. The first step in mitigating and reducing earthquake disasters is a reliable estimate of possible earthquake disaster. High spatial resolution is essential for such an estimate because the disaster is accumulation of damage to each building or structure in the area while the earthquake itself is a crustal-scale phenomenon. An urban area simulation of earthquake hazard and disaster is a unique solution to realizing such a reliable estimate.

An earthquake and the disaster it induces consist of the following three processes:

1. seismic waves generated by the fault propagate in the crust;
2. the seismic waves are amplified near soft ground-surface layers;
3. buildings and structures are shaken by the resulting ground motion;

see Fig. 11. The *physics* of these processes is quite simple. This is because the processes are described completely as solutions of linear or nonlinear solid wave equations. However, the physics has largely been ignored because solving the wave equation for all the elements of the processes requires a tremendous amount of computation. Empirical equations such as attenuation relations and vulnerability curves are usually used for earthquake hazard and disaster estimations [28–30].

Despite the simple physics of the three processes, constructing a model for analysis is not an easy task. Modern observation technologies in Earth Sciences are addressing this problem. However, seismic wave propagation simulation that solves the wave equation has become a hot topic since an accurate model is constructed for a fault and a shallow crust. This simulation requires a large amount of computation, and some achievements using high performance computing techniques have been presented even at the supercomputing conferences (SCs); for example, see [31–34].

The structural seismic response simulation of man-made structures, which is directly related to estimate of an earthquake disaster, is also being realized [35, 36]; an analysis model is automatically constructed for every building and structure using urban area digital information (which is stored in a form of Geographical Information System). Urban area seismic response is then computed as an assembly of structural seismic responses of all the buildings and structures. A good example is the Tokyo simulation [37], in which nonlinear responses of 253,405 buildings are

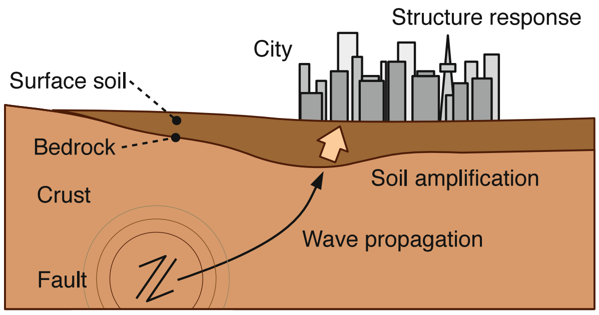


Fig. 11 Schematic view of earthquake disaster process

computed for 100 earthquake scenarios using 16,000 CPU cores of the K computer for 4088 s.

A missing item that links the seismic wave propagation simulation and the urban area seismic response simulation is the seismic wave amplification simulation; high impedance contrast between shallow soft soil and deep hard rock makes the degree of the amplification be larger than 10 times for ground motion components of 1–10 Hz. Seismic wave amplification processes are complicated due to the nonlinear material properties of the surface soil as well as the topographical effects of the surface and ground layer configuration. This results in a wide range of degree of the amplification which changes from place to place. The amplification process could lead to a condition that is likely fatal for structures [38]. The computation cost required for solving a nonlinear wave equation in a domain of irregular geometry is much larger than that of the seismic wave propagation simulation. This is because (1) nonlinearity of soil reduces stiffness, decreasing the wavelength (if stiffness becomes 25 %, the wavelength is reduced to 50 %) and (2) deformation is concentrated at irregular parts of the surface and layers (hills or valleys). These two issues contribute to the need to increase the spatial resolution.

Realization of the seismic wave amplification simulation is a challenge even from the viewpoint of computer science. A target domain is $2000 \times 2000 \times 100$ m, and the spatial resolution needed is of the order of 0.1 m. An earthquake duration of 30 s and a temporal resolution of 0.001 s are required. The resulting degrees of freedom (DOF) are on the order of 10 BlnDOF (or 10×10^9 DOF), and the resulting time step is 30 K (or 30×10^3). Moreover, an unstructured element model must be constructed to account for the irregular geometry of the domain. A most efficient FEM must be developed for this simulation, and a sophisticated model construction method is needed.

The seismic wave amplification simulation is another hot topic for Earth Science, or more specifically, engineering seismology. One trial that used a voxel FEM [39, 40] employed the octree-base domain decomposition with a minimum element size of 5 m, but it was not able to accurately model the irregular geometry and implemented only relatively simple nonlinear soil constitutive models. Another trial [41] used an unstructured grid model, but the spatial resolution was limited; the minimum element size was 4 m, and the related temporal resolution was insufficient for the seismic response simulation of man-made structures.

In this chapter, we present the most updated trial aimed at realizing the seismic wave amplification simulation that numerically solves the nonlinear wave equation. In Sect. 2.1, we explain the development of an FEM code that is able to analyze a nonlinear model of 10 BlnDOF for a 30 K time step and the development of an unstructured element model for the irregularly configured domain. In Sect. 2.2, we show the computation performance of the developed code on the K computer [42]. We also examine the performance on an Intel CPU machine to assess the portability of the code developed. In Sect. 2.3, we show an example of the seismic wave amplification simulation. A 10.7 BlnDOF model of Tokyo is constructed and the seismic wave amplification process is computed for a 30 K time step. The results of the simulation are then inputted to the urban area seismic response simulation that is an assembly of the structural seismic response simulations of 13,275 buildings. Concluding remarks are provided in Sect. 2.4.

2.1 Key Ideas and Innovations

2.1.1 Target Equation

Underground structure is modeled as a layered medium, each layer of which often has complicated geometries because the thickness changes on the order of $10^{-1\sim1}$m in a $10^{0\sim1}$ m region. FEM with unstructured low-order solid elements is suitable for analyzing the response of such a medium together with satisfying traction-free boundary condition on the surface. Here, we use a nonlinear dynamic FEM with three-dimensional (3D) unstructured low-order solid elements for the seismic wave amplification simulation. The equations to be solved are as follows:

$$\left(\frac{4}{dt^2}\mathbf{M} + \frac{2}{dt}\mathbf{C}^n + \mathbf{K}^n\right)\delta\mathbf{u}^n = \mathbf{F}^n - \mathbf{Q}^{n-1} + \mathbf{C}^n\mathbf{v}^{n-1} + \mathbf{M}\left(\mathbf{a}^{n-1} + \frac{4}{dt}\mathbf{v}^{n-1}\right), \qquad (4)$$

with

$$\begin{aligned} \mathbf{Q}^n &= \mathbf{Q}^{n-1} + \mathbf{K}^n\delta\mathbf{u}^n, \\ \mathbf{u}^n &= \mathbf{u}^{n-1} + \delta\mathbf{u}^n, \\ \mathbf{v}^n &= -\mathbf{v}^{n-1} + \tfrac{2}{dt}\delta\mathbf{u}^n, \\ \mathbf{a}^n &= -\mathbf{a}^{n-1} - \tfrac{4}{dt}\mathbf{v}^{n-1} + \tfrac{4}{dt^2}\delta\mathbf{u}^n, \end{aligned} \qquad (5)$$

where $\delta\mathbf{u}$, $\mathbf{u}$, $\mathbf{v}$, $\mathbf{a}$, and $\mathbf{F}$ are vectors describing incremental displacement, displacement, velocity, acceleration, and external force vectors, respectively, $\mathbf{M}$, $\mathbf{C}$, and $\mathbf{K}$ are the consistent mass matrix, the damping matrix, and the stiffness matrix, respectively, dt is the time increment, and n is the time step. We use Rayleigh damping, where the element damping matrix $\mathbf{C}_e^n$ is calculated using the consistent element mass matrix $\mathbf{M}_e$ and element stiffness matrix $\mathbf{K}_e^n$ as follows:

$$\mathbf{C}_e^n = \alpha\mathbf{M}_e + \beta\mathbf{K}_e^n.$$

The coefficients α and β were determined by solving the least-squares equation, i.e.,

$$\text{minimize}\left[\int_{f_{\min}}^{f_{\max}}\left(h^n - \frac{1}{2}\left(\frac{\alpha}{2\pi f} + 2\pi f\beta\right)\right)^2 df\right],$$

where $f_{\max}$ and $f_{\min}$ are the maximum and minimum values of the target frequencies, and h^n is the damping ratio at time step n. The damping matrix is calculated at each time step because the stiffness and damping ratio change with time.

Small elements are generated locally when modeling complex geometry with solid elements. Satisfying the Courant condition in using an explicit time-integration method leads to small time increments, which results in considerable computational expense. To resolve this, we used the Newmark-β method for time integration (with $\beta = 1/4$, $\delta = 1/2$). Because complicated nonlinear constitutive relations are often used in analyzing nonlinear ground motion and there are cases that the implicit scheme does not reach an accurate solution, an explicit scheme is selected, in order to carry out the time integration robustly. Semi-infinite absorbing boundary conditions are used for the bottom and side boundaries of the simulation domain.

The modified Ramberg–Osgood model [43] and Masing rule [44] are used for the nonlinear constitutive relations of the surface soil; while more sophisticated constitutive relations are proposed, these models are standard for dynamic analysis of soil.

The unknowns in Eq. (4) are $\delta\mathbf{u}^n$. We solve for $\delta\mathbf{u}^n$ at each time step to obtain the history of $\mathbf{u}$. The calculation proceeds as follows:

- Read boundary conditions;
- Compute the stiffness and damping ratio for the n-th time step using the Ramberg–Osgood model and Masing rule with the strain obtained at the $n-1$-th time step;
- Compute $\mathbf{K}^n$ and $\mathbf{C}^n$ using the stiffness and damping ratio for the n-th time step;
- Compute $\delta\mathbf{u}^n$ by solving Eq. (4), and update the values in Eq. (5) using $\delta\mathbf{u}^n$;
- Output results;

Because most of the computation cost in this procedure is due to solving Eq. (4), the key point for achieving efficient analysis is the development of a suitable solver, considering the characteristics of the nonlinear dynamic FEM. A method for constructing such an FEM code with a large number of unstructured low-order elements was also developed to model the complex ground structure accurately.

2.1.2 Key Ideas and Innovations in the Solver

In the seismic wave amplification simulation, we must compute $\delta\mathbf{u}^n$ with a 30 K order time step using a high-resolution 10 Bln-order-DOF FEM model (FEMmodel) that can ensure the convergence of $\mathbf{u}^n$ in terms of acceleration response and the convergence of structural response in an urban area. In short, this problem results in solving a 10 Bln-order-DOF matrix equation with a 30 K-order time step, where the components are changed at every time step.

In view of limitations on computer resources, a fast solver is very desirable. For example, we must solve this problem with a 30 K time step in 12 h on the K computer. This requires that only 1.44 s be used to solve the response for one time step. Because a large-DOF problem must be solved with relatively small computer memory, we develop a fast iterative solver, considering the characteristics of the matrix equation. There has been much research on fast iterative solvers which are tuned for FEM with unstructured low-order elements; there are some researches which have been reported even at SCs (e.g., [45]). These researchers seek to reduce the computation time using a preconditioner with a high computation cost, in order to improve the poor convergency of the target problem. Others have reported similar research (e.g., [46, 47]). For example, Ogino et al. [46] presents a fast solver based on a balancing domain decomposition method [48]-type preconditioner, which could solve a 140 M-DOF problem in 577 s (3.88 TFLOPS, 24.26 % of peak) with 3.05 TB memory and 2048 CPU on the Earth Simulator [49].

Although a sophisticated preconditioner with smart program tuning can reduce the computation cost and resolve the poor convergency of the target problem, our problem needs a much faster solver. Although the nonlinear dynamic FEM has difficulties in solving a matrix equation repeatedly where the components are changing at every time step, there is a clue in that the characteristics of the matrix are not so bad because of the effect of inertia terms. However, the convergency of the current matrix equation is worse than the convergency of a matrix equation which comes from an "ordinary" problem with lower resolution, even though the same DOF is used for the problem. Thus, we need a smart preconditioner with high scalability under a many-compute-nodes environment and with lower computation cost for a fast iterative solver. Additionally, less use of computer memory is required for the preconditioner because each compute node manages a large number of DOF. As a result, it is concluded that a moderate preconditioner with less computation cost and high scalability in a many-compute-nodes environment is more suitable than a smart preconditioner with higher computation cost and only moderate scalability.

One candidate for such a fast iterative solver is the "preconditioned conjugate gradient" (PCG) method [50] with the block Jacobi method. However, the number of iterations tends to increase when solving matrices with poor characteristics. Indeed, we examined this in our numerical experiments, and the number of iterations often increased markedly. The slow convergency rate of PCG should not be overlooked in developing a faster iterative solver for our problem. In this viewpoint, it is a wise choice to employ a moderate preconditioner with lower computation cost, which is explained in the above. We must pay attention to the amount of computation in seeking to achieve fast computation, as well as to reducing the frequency of communication and the amount of synchronization necessary between compute nodes. Considering the characteristics of a nonlinear dynamic FEM, we develop an FEM code with the following four key ideas:

Predictor

The initial solution of the iterative solver is predicted as $\frac{dt}{24}(-9\mathbf{v}^{n-4} + 37\mathbf{v}^{n-3} - 59\mathbf{v}^{n-2} + 55\mathbf{v}^{n-1})$ by the Adams–Bashforth method [51]. Using this method with a small time increment leads to a good initial solution with a relative error of $10^{-4\sim-3}$, which reduces the number of CG iterations.

Adaptive Conjugate Gradient Method [52]

A preconditioner in a conventional conjugate-gradient method uses the inverse of a matrix, the characteristics of which are similar to the matrix of the target problem. In the adaptive conjugate-gradient method, the matrix equation is solved roughly using a conjugate-gradient iteration instead of the inverse matrix. The conventional gradient iteration and the gradient iteration for a preconditioner are called the inner and outer loops, respectively.

Multigrid (MG) Method [50]

The MG method constructs a coarse FEM model from the original model; for simplicity, the coarse model is denoted by M_c, and the original one by M. M_c is equivalent to M but the resolution of M_c is coarser. In the inner loop, a rough solution is first computed for M_c. Using this solution as an initial solution, a rough solution is computed for M with fewer iterations. Although additional computation of M_c is required, the computation cost and the communication cost become much smaller than those needed for M only, because the DOF and the nodes on the interfaces of M_c are much smaller than those of M. As a result, the computation cost of the inner loop is reduced by the MG method.

Mixed-Precision (MP) Arithmetic

Although double-precision arithmetic is needed in the outer loop, single-precision arithmetic is used for the inner loops since only estimation of low precision is needed for the preconditioner. With the DOF of M_c being much smaller than that of M and with halving the required memory size by using single precision, we can store the matrix of M_c in the CRS format. Also, a reduction in communication size is expected; the size of adjacent node communication for M with single precision is reduced to half that with double precision, and the size of adjacent node communication for M_c with single precision is reduced to 1/8 of that for M with double precision. Thus, faster computation and memory access are also expected.

Element-by-Element Method [53]

It is difficult to store **K** of M for solving Eq. (4) because the DOF computed by one compute node is large. Instead of storing **K**, computing **Kx** is made according to the element-by-element (EBE) method; EBE is applied to compute the second-order tetrahedral elements of M. With the advantage of displacement-stress mixed FEM [54] and the orthogonal-basis function, we successfully transformed **Kx** into

$$\sum_i \mathbf{N_b}_i^T(\mathbf{D}(\mathbf{N_b}_i\mathbf{x})),$$

where $\mathbf{D}$ and $\mathbf{N}_{\mathbf{b}_i}$ are the elastic tensor and equivalent mode, respectively. Sophisticated reformulation and careful tuning achieve a further reduction in the number of operations and lead to higher peak performance ratios (e.g., 21.68 % for EBE with single precision) on the K computer. Because the frequency of EBE with single precision is very high, this speed-up can reduce computation time significantly.

Hereafter, we call this code "GAMERA"[1] (the seismic wave amplification simulator, enhanced by a multiGrid method, Adaptive conjugate gradient method, Mixed precision arithmetic, EBE method, and pRedictor by Adams–Bashforth method). Although the computation cost of one outer loop is almost the same as that for one loop of PCG, the number of outer iterations is reduced markedly by the sophisticated preconditioner. Moreover, the computation cost of the preconditioner itself is also reduced. As a result, the total computation time of GAMERA can be reduced.

In actual computations, we decompose M into a set of M^i for the i-th MPI process. Then, we compute $\mathbf{u}^n$ with each pair of M^i and $\mathrm{M_c}^i$ using Algorithms 1 and 2; M^i and $\mathrm{M_c}^i$ are regarded as s submodel of M and $\mathrm{M_c}$, respectively, as will be shown in the next subsection. Adjacent node communication (MPI_Isend, MPI_Irecv, MPI_Waitall) after the matrix vector product and collective communication (MPI_Allreduce) after the inner product are conducted for synchronization. Because M^i and $\mathrm{M_c}^i$ are computed by OpenMP parallelization, the whole process of GAMERA is computed by MPI-OpenMP hybrid parallelization.

2.1.3 Key Ideas and Innovations in Model Construction

We present the construction method of a 10 Bln-order-DOF 3D FEM model for GAMERA using data of 3D layered ground structure. Because we seek to evaluate strain and stress in complex ground structures at high resolution, we use a large number of unstructured, second-order tetrahedral elements. The amount of work that is needed for constructing such a large 3D FEM model is considerably high. Thus, a fully automated meshing algorithm that is capable to generate elements of high quality is desirable. Although it is common to perform manual tuning to avoid elements with poor quality, manual tuning for the present model is unrealistic. For this reason, we combine various techniques to develop an automated model construction. We use the method of [55] as the base, which is developed for a 3D FEM model.

In this model generation, a structured grid is used to break up the simulation domain into a number of cells, and elements are constructed for each cell with small aspect ratios. This leads to robust mesh of high-quality elements in full automation. Because the generation of elements in each cell is performed individually, the problem is suitable for parallelization, leading to a reduction in the time for meshing. We generate linear-tetrahedral elements and cubic elements and then convert the

[1]An imaginary Japanese Earth guardian, supported by prayers for peace.

Algorithm 1 Details of GAMERA. Matrix vector product $(\frac{4}{dt^2}\mathbf{M} + \frac{2}{dt}\mathbf{C}^n + \mathbf{K}^n)(\ )$ is computed by EBE. diag[ ], $(\bar{\ })$, and ϵ indicate 3×3 block Jacobi of [ ], single-precision variable and tolerance for relative error. $(\ )_c$ indicates calculation related to FEMmodel$_c$, and the other is the related calculation of FEMmodel. $\bar{\mathbf{P}}$ is a mapping matrix, from FEMmodel$_c{}^i$ to FEMmodel^i, which is defined by interpolating the displacement in each element of FEMmodel$_c{}^i$

```
1:  read boundary condition of n-th time step
2:  $\mathbf{f} \Leftarrow \mathbf{F}^n - \mathbf{Q}^{n-1} + \mathbf{C}^n\mathbf{v}^{n-1} + \mathbf{M}(\mathbf{a}^{n-1} + \frac{4}{dt}\mathbf{v}^{n-1})$
3:  $\mathbf{x} \Leftarrow \frac{dt}{24}(-9\mathbf{v}^{n-4} + 37\mathbf{v}^{n-3} - 59\mathbf{v}^{n-2} + 55\mathbf{v}^{n-1})$
4:  $\bar{\mathbf{B}} \Leftarrow diag[\frac{4}{dt^2}\mathbf{M} + \frac{2}{dt}\mathbf{C}^n + \mathbf{K}^n]$
5:  $\bar{\mathbf{B}}_c \Leftarrow diag[\frac{4}{dt^2}\mathbf{M}_c + \frac{2}{dt}\mathbf{C}^n_c + \mathbf{K}^n_c]$
6:  $\bar{\mathbf{A}}_c \Leftarrow \frac{4}{dt^2}\mathbf{M}_c + \frac{2}{dt}\mathbf{C}^n_c + \mathbf{K}^n_c$ (stored on memory with CRS format)
7:  $\mathbf{r} \Leftarrow \mathbf{f} - (\frac{4}{dt^2}\mathbf{M} + \frac{2}{dt}\mathbf{C}^n + \mathbf{K}^n)\mathbf{x}$
8:  $\beta \Leftarrow 0$
9:  $i \Leftarrow 1$
10: (*outer loop start*)
11: while $\|\mathbf{r}\|_2/\|\mathbf{f}\|_2 \geq \epsilon$ do
12:     (*inner loop start*)
13:     $\bar{\mathbf{r}} \Leftarrow \mathbf{r}$
14:     $\bar{\mathbf{z}} \Leftarrow \mathbf{B}^{-1}\mathbf{r}$
15:     $\bar{\mathbf{r}}_c \Leftarrow \bar{\mathbf{P}}^T\bar{\mathbf{r}}$
16:     $\bar{\mathbf{z}}_c \Leftarrow \bar{\mathbf{P}}^T\bar{\mathbf{z}}$
17:     $\bar{\mathbf{z}}_c \Leftarrow \bar{\mathbf{A}}_c^{-1}\bar{\mathbf{r}}_c$ (*solved on FEMmodel_c by Algorithm 2 with $\epsilon_c^{in}$ and initial solution $\bar{\mathbf{z}}_c$*)
18:     $\bar{\mathbf{z}} \Leftarrow \bar{\mathbf{P}}\bar{\mathbf{z}}_c$
19:     $\bar{\mathbf{z}} \Leftarrow (\frac{4}{dt^2}\bar{\mathbf{M}} + \frac{2}{dt}\bar{\mathbf{C}}^n + \bar{\mathbf{K}}^n)^{-1}\bar{\mathbf{r}}$ (*solved on FEMmodel by Algorithm 2 with $\epsilon^{in}$ and initial
        solution $\bar{\mathbf{z}}$*)
20:     $\mathbf{z} \Leftarrow \bar{\mathbf{z}}$
21:     (*inner loop end*)
22:     if $i > 1$ then
23:         $\beta \Leftarrow (\mathbf{z}, \mathbf{q})/\rho$
24:     end if
25:     $\mathbf{p} \Leftarrow \mathbf{z} + \beta\mathbf{p}$
26:     $\mathbf{q} \Leftarrow (\frac{4}{dt^2}\mathbf{M} + \frac{2}{dt}\mathbf{C}^n + \mathbf{K}^n)\mathbf{p}$
27:     $\rho \Leftarrow (\mathbf{z}, \mathbf{r})$
28:     $\alpha \Leftarrow \rho/(\mathbf{p}, \mathbf{q})$
29:     $\mathbf{q} \Leftarrow -\alpha\mathbf{q}$
30:     $\mathbf{r} \Leftarrow \mathbf{r} + \mathbf{q}$
31:     $\mathbf{x} \Leftarrow \mathbf{x} + \alpha\mathbf{p}$
32:     $i \Leftarrow i + 1$
33: end while
34: (*outer loop end*)
35: $\delta\mathbf{u}^n \Leftarrow \mathbf{x}$
36: $\mathbf{Q}^n \Leftarrow \mathbf{Q}^{n-1} + \mathbf{K}^n\delta\mathbf{u}^n$
37: $\mathbf{u}^n \Leftarrow \mathbf{u}^{n-1} + \delta\mathbf{u}^n$
38: $\mathbf{v}^n \Leftarrow -\mathbf{v}^{n-1} + \frac{2}{dt}\delta\mathbf{u}^n$
39: $\mathbf{a}^n \Leftarrow -\mathbf{a}^{n-1} - \frac{4}{dt}\mathbf{v}^{n-1} + \frac{4}{dt^2}\delta\mathbf{u}^n$
40: output results of n-th time step
41: modify material properties considering nonlinearity
```

Algorithm 2 PCG with a preconditioner. This is used for roughly solving $\mathbf{z} = (\frac{4}{dt^2}\mathbf{M} + \frac{2}{dt}\mathbf{C}^n + \mathbf{K}^n)^{-1}\mathbf{r}$ and $\mathbf{z}_c = \mathbf{A}_c^{-1}\mathbf{r}_c$ in Algorithm 1. (or) is used for estimation of $\mathbf{z}_c = \mathbf{A}_c^{-1}\mathbf{r}_c$. $\mathbf{B}$ and ϵ^{in} are a block Jacobi matrix and the tolerance for the relative error. All the calculations are conducted using single-precision variables

```
$\mathbf{x} \Leftarrow \mathbf{z}$
$\mathbf{e} \Leftarrow \mathbf{r} - (\frac{4}{dt^2}\mathbf{M} + \frac{2}{dt}\mathbf{C} + \mathbf{K})\mathbf{x}$   $(or\ \ \mathbf{A}_c\mathbf{x})$
$\beta \Leftarrow 0$
$i \Leftarrow 1$
while $(\|\mathbf{e}\|_2/\|\mathbf{r}\|_2 \geq \epsilon^{in}$   $(or\ \ \epsilon_c^{in})$ do
    $\mathbf{z} \Leftarrow \mathbf{B}^{-1}\mathbf{e}$   $(or\ \ \mathbf{B}_c^{-1}\mathbf{e})$
    $\rho_a \Leftarrow (\mathbf{z}, \mathbf{e})$
    if $i > 1$ then
        $\beta \Leftarrow \rho_a/\rho_b$
    end if
    $\mathbf{p} \Leftarrow \mathbf{z} + \beta\mathbf{p}$
    $\mathbf{q} \Leftarrow (\frac{4}{dt^2}\mathbf{M} + \frac{2}{dt}\mathbf{C} + \mathbf{K})\mathbf{p}$   $(or\ \ \mathbf{A}_c\mathbf{p})$
    $\alpha \Leftarrow \rho_a/(\mathbf{p}, \mathbf{q})$
    $\rho_b \Leftarrow \rho_a$
    $\mathbf{e} \Leftarrow \mathbf{e} - \alpha\mathbf{q}$
    $\mathbf{x} \Leftarrow \mathbf{x} + \alpha\mathbf{p}$
    $i \Leftarrow i + 1$
end while
$\mathbf{z} \Leftarrow \mathbf{x}$
```

cubic elements to linear-tetrahedral elements. Next, the linear-tetrahedral elements are converted into second-order tetrahedral elements. We use the model with linear-tetrahedral elements as the M_c and the model with second-order tetrahedral elements as M. We divide M and M_c, so that the number of elements in each domain becomes uniform, using METIS 5.1.0 [56] with suitable options on a shared-memory computer; recall that the i-th submodels are denoted by M^i and ${M_c}^i$, respectively. Although there has been research related to the efficient generation of M^i from M on distributed memory computers, we use this generation method because load imbalance for M^i and ${M_c}^i$ can be reduced significantly, even for a complicated ground structure model. Such reduced load imbalance contributes to high scalability with Algorithms 1 and 2, and the advantage in reducing the analysis time with high scalability is much greater than the disadvantage of increased model generation time.

FEM models for GAMERA are generated on a virtual shared memory machine: 20 computers (Dell R720xd, each with dual Intel Xeon E5-2670 CPUs and DDR3 384 GB memory) are connected with InfiniBand FDR and shared virtually by vSMP_Foundation [57]. For the generation of the 27 BlnDOF, 6.7 Bln-element FEM model, most of the cost was spent on the generation of M_c, the conversion of M_c to M, and the division of M into $\{M^i\}$. The generation of M_c takes 3 h 43 min with 160 OpenMP threads and 698 GB computer memory, the conversion of M_c to M takes 4 h 53 min with 1455 GB computer memory, and the division of M into $\{M^i\}$ for $i = 1, 2, \ldots, 82{,}944$ tales 13 h 59 min with 1708 GB computer memory.

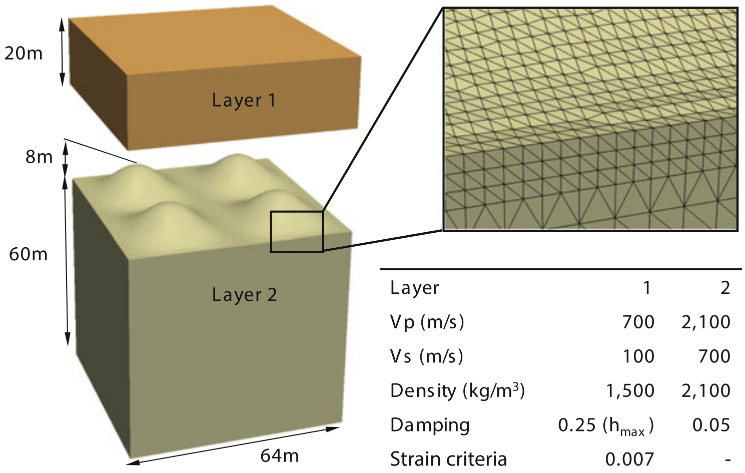
Because recent increases in computer memory have largely resolved the problem of the large computer memory required, only the construction time remains an issue. Almost half of the construction time is due to file I/O while the other half consists of meshing, calling the METIS library, and mapping between M and M_c. The throughput of the file system used (NL-SAS, 7200 rpm HDD) is 100–200 MB/s using the HDF5 library, while the total file size of the models is 1649 GB. It is expected that construction time could be reduced by using faster file systems with parallel I/O. However, because we conduct many nonlinear seismic wave amplification simulations using one FEM model to estimate the response against many earthquake scenarios, this model construction time is in fact a relatively minor problem. Indeed, construction of an FEM model with less load imbalance is much more desirable, even if the construction time is relatively long.

2.2 Performance Measurement

In this study, we measured the performance of GAMERA on the K computer, a massively parallel supercomputer at RIKEN, Advanced Institute for Computational Science [42]. The K computer consists of 82,944 compute nodes, each with single SPARC64$^{\mathrm{TM}}$ VIIIfx CPUs with 6 MB L2 shared cache. Each of the eight cores of the SPARC CPU has two sets of twin fused multiply-and-add (FMA) units, which leads to four multiplications and four additions in one clock cycle. The number of operations per clock cycle is the same for single and double precision floats. The operating clock speed of the CPU is 2.0 GHz, leading to a peak performance of 8 FLOP $\times$ 2 GHz $\times$ 8 CPU cores $=$ 128 GFLOPS per CPU. Each node has 16 GB of DDR3-SDRAM memory, with peak memory bandwidth of 64 GB/s. Tofu, a six-dimensional interconnection network, is used for communication between nodes [58]. Each node can communicate in four directions simultaneously, with 5 GB/s throughput in each direction. An OpenMPI 1.4.3 [59]-based MPI library optimized for the Tofu network was used, following the MPI 2.1 standard [60].

2.2.1 Problem Setting

We show the effectivity of GAMERA by comparing four codes: GAMERA, GAMERA (DP,MG), GAMERA (DP,SG), and PCGE. GAMERA (DP,MG) is GAMERA with the mixed-precision arithmetics disabled. GAMERA (DP,SG) is GAMERA with both the mixed-precision arithmetics and the multi-grid preconditioner disabled. PCGE is GAMERA whose solver is replaced with PCG using the 3×3 block Jacobi method for the preconditioner and the EBE method for matrix–vector products. The same tuning applied to GAMERA is applied to all codes. We used a relative error of $\epsilon = 10^{-8}$, $\epsilon^{\mathrm{in}} = 0.25$, $\epsilon_c^{\mathrm{in}} = 0.5$ for the tolerance of the three linear solvers, i.e., the outer loop solver, the inner loop



Layer	1	2
Vp (m/s)	700	2,100
Vs (m/s)	100	700
Density (kg/m^3)	1,500	2,100
Damping	0.25 (h_{max})	0.05
Strain criteria	0.007	-

Fig. 12 Ground model for performance measurement. The 94,407,951-DOF model shown is duplicated in the *x* and *y* directions for making large-scale models with periodical geometry

solver and the inner loop solver for the coarse model. We use one compute node (one MPI process) for each M^i and $M_c{}^i$ and parallelized each MPI domain using OpenMP. In the case of the K computer, 8*m* CPU cores are used for *m* compute nodes.

For performance measurement, we uniformly input a wave from the bottom of a two-layered ground that mimicked actual ground structures. The geometry of the ground was made periodical, so that it was suitable for measuring size-up efficiency. As shown in Fig. 12, the first layer is a soft layer that behaves nonlinearly under large earthquakes, and the second layer is a harder layer that mimics the bedrock layer. The interface between the two layers has bumps that mimic local changes in the ground structure. We duplicate this problem in the *x* and *y* directions when enlarging the problem size. We constructed an FEM model of this problem with an element size small enough (0.44 m) to obtain convergence in ground acceleration, even when the soils soften due to nonlinearity. In this case, we input a wave with 100 time steps of $dt = 0.002$ s, which was the main part of the Kobe earthquake wave [61]. Table 9 shows the model sizes and number of partitions used for measuring size-up efficiency (weak scaling) and speed-up efficiency (strong scaling).

2.2.2 Performance Results (1)

Although GAMERA is best suited for systems with a fast single-precision floating-point computing ability, we measure its performance using the K computer. The time-to-solution is important for solving large-scale problems, and thus the

Table 9 Configuration of models used for performance measurement

Model	# of MPI domains	# of CPU cores	Total DOF	Mean DOF per MPI domain	σ_n (%)	σ_e (%)	# of PCGE iterations
1	4608	36,864	1,505,755,135	326,767	4.13	0.86	355
2	9216	73,728	3,010,699,983	326,681	4.37	0.70	355
3	18,432	147,456	6,019,818,087	326,596	4.41	0.68	355
4-A	9216	73,728	12,038,067,615	1,306,213	2.67	0.46	355
4-B	18,432	147,456	12,038,067,615	653,107	3.40	0.55	355
4-C	36,864	294,912	12,038,067,615	326,553	4.55	0.68	355
5	82,944	663,552	27,082,129,647	326,511	4.51	0.75	355

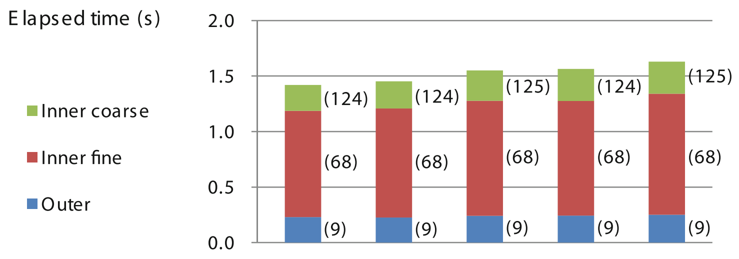
σ_n, σ_e indicates the maximum imbalance of the number of nodes and elements. $\sigma_n = (\max n - \text{mean } n)/\text{mean } n$, $\sigma_e = (\max e - \text{mean } e)/\text{mean } e$

scalability for solving the large-scale problems becomes essential. We measure performance using the ninth time step because this step is not influenced by the fluctuations in the initial part of the wave and a poor convergency problem arises at this step.

Although problem characteristics change according to the DOF of the problem, the models used here have almost the same convergence characteristics due to the periodical geometry. In fact, we confirm that the number of loops elapsed for convergence of the PCGE solver is the same for all of the models, as shown in Table 9, and thus we use this set of the FEM models to measure the size-up efficiency of the codes. We first fix the problem size per compute node to 326 K DOF and increase the problem size and the number of CPU cores to measure the size-up efficiency of the code.

Figure 13 shows the elapsed time and its breakdown for solving the ninth time step. Here, "inner coarse," "inner fine," and "outer" indicate line 17 of Algorithm 1, line 19 of Algorithm 1, and the outer loop, respectively. The numbers in brackets in the figure indicate the number of the loops elapsed for convergence of the linear solver. It is seen that the numbers of loops elapsed are almost the same for all the models. Also, it is seen that the increase in the elapsed time is small. Therefore, we conclude that the code scales well to a large number of CPU cores. The smallest model (model 1, 1.5 BlnDOF) is computed in 1.42 s using 36,864 CPU cores (4608 compute nodes $\times$ 8 CPU cores, 8.69 % of peak performance, 51.3 TFLOPS), and the largest model (model 5, 27 BlnDOF) is computed in 1.63 s using the whole K computer, with 663,552 CPU cores (82,944 compute nodes $\times$ 8 CPU cores, 7.57 % of peak performance, 0.804 PFLOPS), indicating a size-up efficiency of 87.1 %.

Compared with the outer and inner fine loops, the inner coarse loop has a larger increase in elapsed time. Next, we show the breakdown of the elapsed time in



Model	1	2	3	4-C	5
Elapsed time (s)	1.42	1.45	1.55	1.56	1.63
FLOPS/Peak (%)	8.69	8.50	7.95	7.87	7.57
Mem. throughput/Peak (%)	13.43	13.14	12.31	12.18	11.72

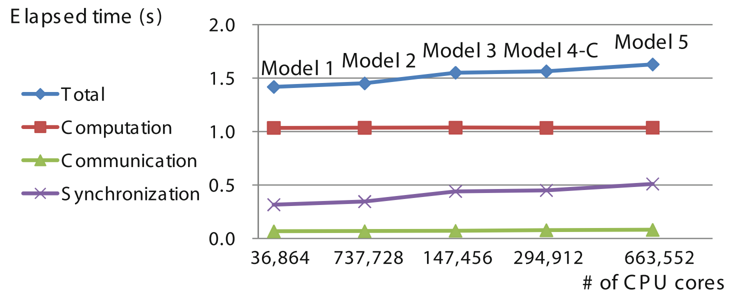


Fig. 13 Elapsed time and its breakdown (inner fine, inner coarse, outer loop) for the ninth time step of the weak-scaling problem set. *Numbers in brackets* indicate the number of elapsed loops required for convergence of the linear solver. The breakdown of the elapsed time for computation, communication, and synchronization is also shown

terms of computation, communication, and synchronization.[2] It may be expected that the communication time would increase with respect to model size due to the increase in the number of hops in peer-to-peer communications and the increase in the number of nodes involved in all-reduce operations. However, the increase in communication time is, in fact, only 0.014 s; the communication time of model 5 and

[2] Defined as:

$$\begin{aligned}
\text{communication} &= \text{AVG}\{\text{MPI\_Isend} + \text{MPI\_Irecv} \\
&\quad + \text{MPI\_Allreduce(total)} - \text{MPI\_Allreduce(wait)}\} + \text{MIN}\{\text{MPI\_Waitall}\}, \\
\text{synchronization} &= \text{AVG}\{\text{MPI\_Allreduce(wait)} + \text{MPI\_Waitall}\} - \text{MIN}\{\text{MPI\_Waitall}\}, \\
\text{computation} &= \text{AVG}\{\text{total}\} - \text{communication} - \text{synchronization}.
\end{aligned}$$

Here, AVG and MIN indicate the average and minimum values of all the compute nodes, respectively. MPI_Barrier is not used.

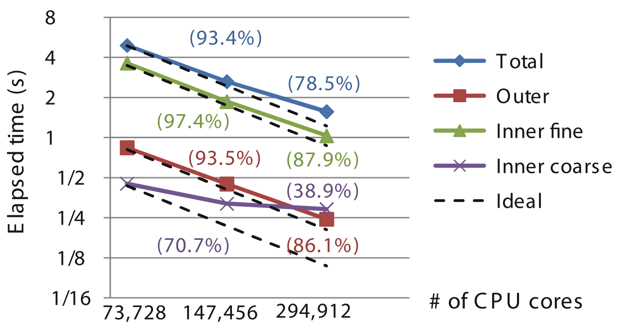
model 1 is 0.081 and 0.067 s. Such a small increase in the communication time is due to the decrease in the communication size for the peer-to-peer communication and the hardware enhancements in the K computer's interconnect for all-reduce operations [58]. However, we can see that synchronization time has increased from 0.316 (model 1) to 0.511 s (model 5).

The imbalance of the models are summarized in Table 9. It is seen that the imbalance in the number of elements and nodes increases with respect to the model size. The amount of EBE and CRS computations is correlated with the number of elements and nodes, respectively. We might guess that the larger imbalance in the number of nodes is the cause of worse deterioration in the performance of the inner coarse solver. Although load imbalance due to I/O or jitters of the OS is included in the synchronization time, we can expect to improve performance by decreasing the imbalance in the number of nodes and elements by tuning the parameters of METIS.

We measure the speed-up efficiency of the code. Figure 14 shows the change in elapsed time with respect to the change in the number of CPU cores. It is seen that the 12 BlnDOF is solved in 4.92 s using 73,728 CPU cores (9216 compute nodes $\times$ 8 CPU cores, 117.8 TFLOPS, 9.99 % of peak) and that the same problem is solved in 1.56 s using 294,912 CPU cores (36,864 compute nodes $\times$ 8 CPU cores, 371.4 TFLOPS, 7.87 % of peak), with a relatively good speed-up efficiency of 78.5 %. In view of the breakdown of the elapsed time, it is seen that the outer and inner fine loops have high scalability, above 85 %. However, the increase in the time used for solving the inner coarse loop is prominent (with 38.9 % speed-up efficiency); this is the primary reason for the deterioration in the scalability of the whole program. As is the case with the weak-scaling models, this may also be due to the large increase in the imbalance in the number of nodes with respect to the increase in the number of compute cores.

We show the breakdown of the elapsed time in terms of computation, communication, and synchronization. We can see that the time for communication and synchronization has increased, but the time for communication is still well suppressed. Figure 15 shows the effectivity of mixed-precision arithmetics and the multi-grid preconditioner for model 5 using the whole K computer with 663,552 CPU cores (82,944 compute nodes $\times$ 8 CPU cores). Because the number of floating-point operations per clock cycle does not change for single-precision or double-precision numbers, the peak FLOPS are the same for both. However, the bytes required are halved when using single precision, which would be expected to increase performance. First, we compare GAMERA and GAMERA (DP,MG). Because the B/F of the preconditioner doubles by changing to double precision, FLOPS decreases, and the elapsed time increases to 1.15 times that of GAMERA.

Next, we compare GAMERA and GAMERA (DP,SG). The computation of M_c^i (line 17 of Algorithm 1) uses linear elements with a CRS format, whereas the computation of M^i (line 19 of Algorithm 1) uses second-order elements with EBE. Thus, the B/F of the inner coarse loop becomes larger than that of the inner fine loop, and the floating-point arithmetic performance of the inner fine loop becomes better



Model	4-A	4-B	4-C
Elapsed time (s)	4.92	2.63	1.56
FLOPS/Peak (%)	9.99	9.36	7.87
Mem. Throughput/Peak (%)	19.14	16.05	12.18

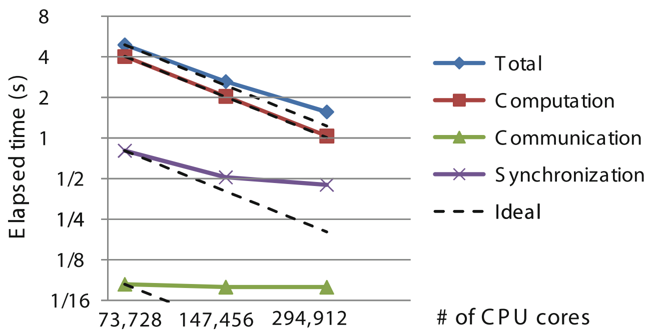
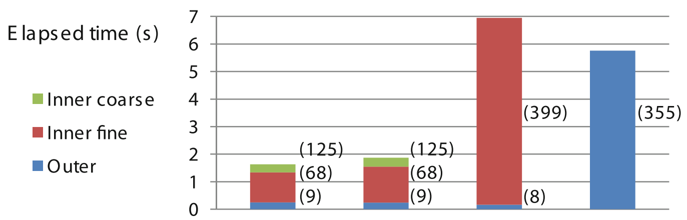


Fig. 14 Elapsed time and its breakdown (inner fine, inner coarse, outer loop) for the ninth time step of the strong-scaling problem set. *Numbers in brackets* show the speed-up efficiency with respect to Model 4-A (73,728 CPU cores). The breakdown of the elapsed time for computation, communication, and synchronization is also shown

than the inner coarse loop. Consequently, the FLOPS performance of GAMERA (DP,SG) increases by eliminating the inner coarse solver, but the number of loops needed for convergence increases, and thus the total time for solving the matrix equation increases, to 4.26 times that of GAMERA. Most of the cost for PCGE is involved in the EBE computation for second-order elements. Because we use the tuned EBE computation developed in this paper, the FLOPS and memory throughput are good, and thus it is possible to compute each loop in a short time. However, the number of loops needed for convergence becomes large and thus the time required to solve the matrix equation is 3.53 times that of GAMERA. Although the PCGE is slightly better in terms of peak performance when using the K computer, we can see that the multi-grid preconditioner is highly effective



	GAMERA (MP,MG)	GAMERA (DP,MG)	GAMERA (DP,SG)	PCGE
Elapsed time (s)	1.63	1.87	6.95	5.75
FLOPS/Peak (%)	7.57	6.60	7.88	8.11
Mem. Throughput/Peak (%)	11.72	19.40	20.40	21.66

Fig. 15 Comparison of performance of GAMERA, GAMERA (DP,MG), GAMERA (DP,SG), and PCGE for solving the ninth time step of Model 5 (27 BlnDOF, 663,552 CPU cores). *Numbers in brackets* indicate the number of elapsed loops required for convergence of the linear solver

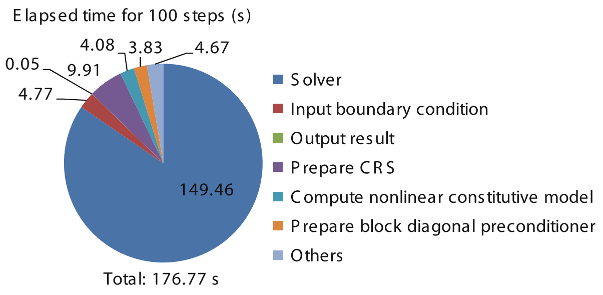


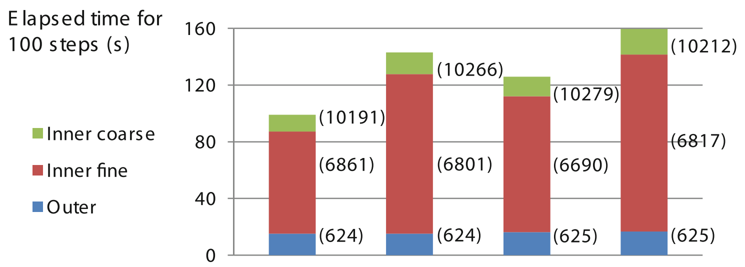
Fig. 16 Breakdown of the elapsed time used for analyzing 100 steps of Model 5 (27 BlnDOF, 663,552 CPU cores). 0.736 PFLOPS (6.93 % of peak) was attained for the whole program

and that the multi-grid method combined with mixed-precision arithmetics results in a time to solution more than 3 times faster compared with PCGE.

2.2.3 Performance Results (2)

All of the measurements above concerned the cost of the linear solver; we show the total cost of the 100 time steps in Fig. 16. In addition to the solver, the program involves preparing CRS of the coarse mesh for every time step, updating the material parameters using the strain of the last time step, and computing the block Jacobi matrix used for the preconditioner. Because the input is the same for all ranks, rank 0 reads the input wave from the file system and broadcasts it to other processors. The output of the 100-step simulation is gathered per rank and outputted in binary to reduce the number of files and increase throughput. By such measures, the time spent for I/O is reduced to 2.7 % of the whole program. Also, the other parts take 12.7 % of the whole elapsed time. Thus, we can see that most of the time is used for the solver (84.6 %) and that the performance of the solver dominates the performance of the whole analysis. Although the performance of GAMERA was high for the 100 steps solved here, we can expect increased effectivity of GAMERA when solving poorer convergency problems, considering more complex grounds or input waves with phase differences.

Finally, we checked the potential of GAMERA for use in an environment favorable for single-precision arithmetics. The floating-point performance of the Intel Xeon E5-2680 CPU for single-precision floats is twice that for double-precision floats, and this was expected to lead to a large change in time-to-solution. Figure 17 shows the elapsed time and performance of GAMERA and GAMERA (DP,MG). Here, we use 256 compute nodes of Stampede at Texas Advanced Computing Center, The University of Texas at Austin [62], and 256 compute nodes of the K computer. Although each compute node of Stampede has an Intel Xeon Phi Coprocessor, we only used the dual Xeon E5-2680 CPUs for measurement. We set the DOF per compute node to 369 K DOF, which is similar to that of the weak scaling models in Table 9. Then, 16 OpenMP threads were used per compute node of Stampede, and eight OpenMP threads were used per compute node of the K computer. From the figure, we can see that both the inner fine and inner coarse loops used in the preconditioner were accelerated when using mixed-precision arithmetics for both of the computer systems. The acceleration due to the use of mixed-precision arithmetics was about 1.27 times for the K computer with SPARC CPUs and 1.44 times for Stampede with Intel CPUs. Thus, we can expect further speed-up of GAMERA when using hardware with faster single-precision operation capabilities.



Compute environment	Stampede 256 nodes (each with dual Intel Xeon E5-2680 CPUs)		K computer 256 nodes (each with single SPARC64VIIIfx CPU)	
Solver	GAMERA (MP,MG)	GAMERA (DP,MG)	GAMERA (MP,MG)	GAMERA (DP,MG)
Elapsed time (s)	99.0	142.9	125.9	159.5
Ratio between (MP,MG) and (DP,MG)	1.44	-	1.27	-

Fig. 17 Comparison of the effectiveness of multi-precision arithmetics when using Intel CPU- and SPARC CPU-based systems. *Numbers in brackets* indicate the number of elapsed loops required for convergence

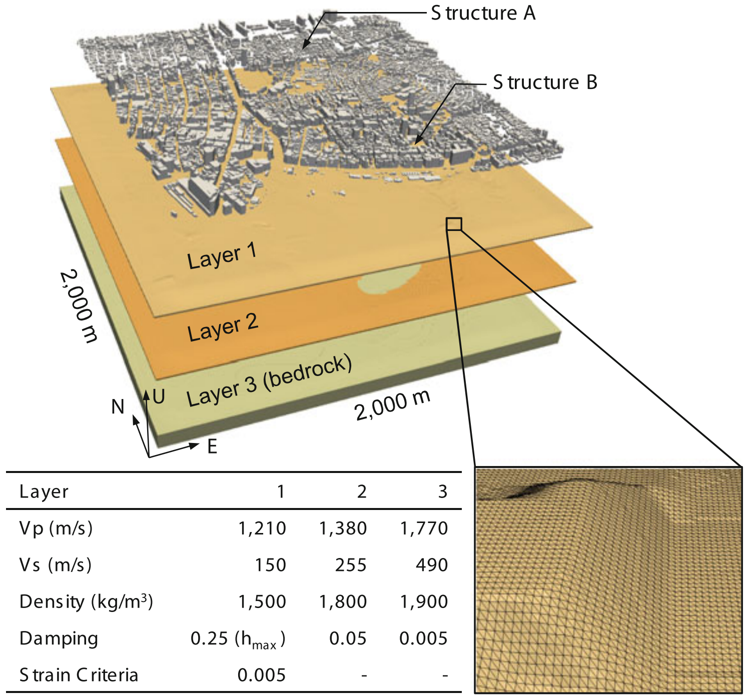
In the Appendix, we report performance of GAMERA$^{\mathrm{EBE4}}$, which is aimed to further reduce time-to-solution when solving larger DOF problems. Due to restrictions in computational resources, we have not been able to perform tests up to the full K computer, but we have attained favorable results compared with GAMERA, such as: elapsed time reduced by 49 % for model 4-A (73,728 CPU cores, 17.1 % of peak FLOPS) and 29 % for model 4-C (294,912 CPU cores, 11.8 % of peak FLOPS). We plan to develop both GAMERA and GAMERA$^{\mathrm{EBE4}}$ which are suitable for different types of hardware in the future.

2.3 Application Example

As an application example, GAMERA is used for the seismic wave amplification simulation that is input to the urban area seismic response simulation. The target area is a 2.0×2.0 km region of central Tokyo; this region consists of businesses, shops, and densely built residential districts, with a total of 13,275 buildings. The ground structure has a rectangular shape of 2.0×2.0 km and a depth of 100 m. The ground structure is modeled with three soil layers using the 5-m grid elevation map [63] and digital geotechnical database [64]. Figure 18 shows the geometry and the material properties of the ground. Material properties of each layer were set based on data at the nearest earthquake observation point [65]. The Kobe earthquake wave [61] with $dt = 0.001$ s and 30,000 time steps is inputted into the bottom of the model. This simulation is carried out on the K computer using 294,912 CPU cores (36,864 compute nodes $\times$ 8 CPU cores). Relative errors of $\epsilon = 10^{-8}$, $\epsilon^{\mathrm{in}} = 0.25$, $\epsilon_c^{\mathrm{in}} = 0.5$ are used for the tolerances of the linear solvers. The FEM model is constructed with an element size small enough to obtain convergence in ground acceleration, even when the soils softened due to nonlinearity. This resulted in a model with 10.7 BlnDOF and a minimum element size of 0.66 m.

Figure 19 shows the breakdown of the elapsed time of the simulation. It takes 41,521 s to compute 30,000 time steps, including I/O, meaning that only 1.38 s was used for solving each time step. Most of the computation time is spent solving the linear equation system (35,245 s). The performance of the whole program is high due to the reduction in I/O cost (1481.6 s for inputting data and 2.5 s for outputting the results). Displacements at the surface were outputted with 0.33-m spatial resolution every 10 time steps, leading to total output size of 1.32 TB. For the first 1000 time steps, the performance of the linear solver was 0.421 PFLOPS (8.93 % of peak), and the performance of the whole program was 0.358 PFLOPS (7.59 % of peak); we can see that high performance was realized on an actual application run.

Figure 20a shows the response of ground at the surface as well as the response of structures using the computed ground motion as an input. In Fig. 20b, it is shown that ground motion responses are far from being uniform and resulting structures shaking change. In an ordinary ground motion analyses, it is difficult to ensure



Layer	1	2	3
Vp (m/s)	1,210	1,380	1,770
Vs (m/s)	150	255	490
Density (kg/m^3)	1,500	1,800	1,900
Damping	0.25 (h_{max})	0.05	0.005
Strain Criteria	0.005	-	-

Fig. 18 Ground and structure model of the application example. A domain of 2.0 × 2.0 km with 13,275 structures was targeted. The three-layered ground was modeled with minimum element size of 0.66 m, leading to a 10,755,536,091-DOF model, and computed using 294,912 CPU cores (36,864 compute nodes) of the K computer

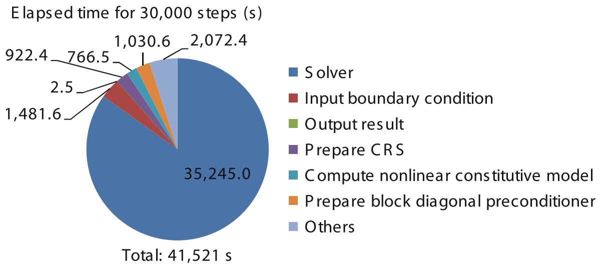


Fig. 19 Computation performance of a 10.7 BlnDOF ground model. For the first 1000 time steps, the performance of the linear solver was 0.421 PFLOPS (8.93 % of peak), and the performance of the whole program was 0.358 PFLOPS (7.59 % of peak)

the convergence of acceleration due to the fatal computational costs, and thus the reliability of results is not necessarily high. However, we are able to achieve highly reliable results that converge in terms of acceleration response for a large domain using GAMERA. Here, we use three models with different resolutions: model I with the highest resolution (10.7 BlnDOF model), model II with 1.5 times the element

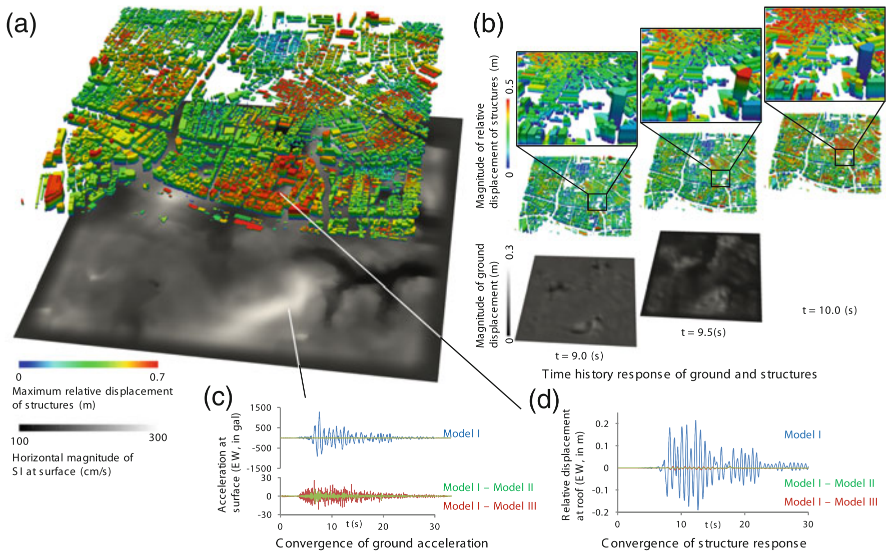

(a)
0
0.7
Maximum relative displacement of structures (m)
100
300
Horizontal magnitude of SI at surface (cm/s)
(b)
Magnitude of relative displacement of structures (m)
0
0.5
Magnitude of ground displacement (m)
0
0.3
t = 9.0 (s)
t = 9.5(s)
t = 10.0 (s)
Time history response of ground and structures
(c)
Acceleration at surface (EW, in gal)
Model I
Model I – Model II
Model I – Model III
t(s)
Convergence of ground acceleration
(d)
Relative displacement at roof (EW, in m)
Model I
Model I – Model II
Model I – Model III
t(s)
Convergence of structure response

Fig. 20 Response of ground and structures. (**a**) Horizontal magnitude of SI at surface and maximum relative displacement of structures. (**b**) Snapshots of time history response. (**c**) Acceleration waveforms at surface (Point A) obtained from Model I (10.7 BlnDOF model), Model II (with element sizes 1.5 times larger than Model I), and Model III (with element sizes 3 times larger than Model I). We can see that the ground acceleration converges with the increase in resolution of the models. (**d**) Relative displacement waveforms of structure A (a two-story RC building located above Point A) obtained using acceleration waveforms computed using Models I, II, and III. We can also see the structure responses converge with respect to the increase in ground model resolution

size of model I, and model III with 3.0 times the element size of model I. Figure 20c shows the comparison of acceleration time history at the surface (Point A). It is seen that the waveform converges with respect to the increase in the discretization resolution of the FEM model and that the waveform of the finest model converges within 1 % of the amplitude of the wave. As they are dependent on the input acceleration at the ground surface, the responses of structures also converge with the increase in resolution; see Fig. 20d.

From these results, we can conclude that high-resolution finite-element modeling of ground motion is essential for analyzing the response of structures. By comparing the ground and structure responses shown in Fig. 20a, it is seen that larger SI values (an index commonly used for evaluating the destructiveness of waves to buildings [66]) do not necessarily lead to large structural responses. Such a complexity in the structure seismic response distribution in an urban area can only be realized by combining high-resolution 3-D ground motion analysis and structure response analysis.

Considering the ambiguities included in the information used for making structure models may further improve the reliability of response analyses of structures in

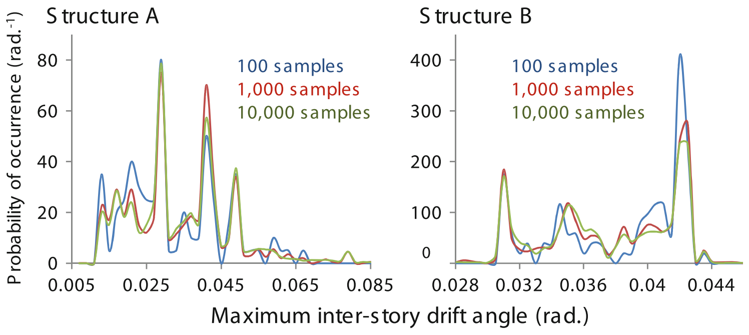


Fig. 21 Probability density of response of two two-story RC structures with stochastic structural parameters. The horizontal axis indicates the maximum inter-story drift angle (an index frequently used to evaluate damage of structures), and the vertical axis indicates the probability of occurrence. We can see that the distribution converges by increasing the number of samples from 100 to 1000 and then to 10,000 samples

a city [67]. Figure 21 shows the result of a stochastic analyses (Monte Carlo simulation) considering the ambiguity of concrete stiffness, following a normal distribution. Although we show only two structures in Fig. 21, we can simulate the stochastic response of all the buildings in the urban area. Such analyses are expected to contribute to improving the reliability of earthquake damage estimation in urban areas. On the other hand, large numbers of samples are needed for attaining convergence in the probability response distribution, and the computational costs for stochastic response analysis of structures become huge. For example, the seismic structural simulations for 13,275 buildings $\times$ 10,000 samples shown here took 3 h, 56 min using 80,000 CPU cores (10,000 compute node $\times$ 8 CPU cores) of the K computer. We can see that the supercomputers can also serve an important role for improving the reliability of structure response analyses as well as ground motion analyses.

2.4 Concluding Remarks

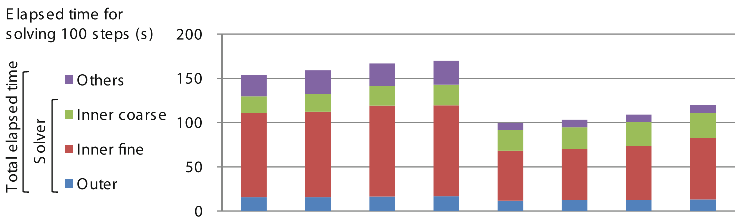
In this chapter, GAMERA is presented as an efficient code for a nonlinear dynamic FEM for the seismic wave amplification simulation that solves nonlinear wave equations. The high computation performance of GAMERA is attributable to the smart use of the computer architecture in solving a physics problem of a wave equation. In particular, the algorithm developed results in a very fast and portable iterative solver. We regard GAMERA as a next-generation dynamic FEM due to its excellent performance: 1.42-s run time for 1.5 BlnDOF with 36,864 CPU cores and 1.63-s run time for 27 BlnDOF with 663,552 CPU cores of the K computer. The scalability (size-up efficiency) was 87.1 %. This performance enabled us to perform a physics-based seismic wave amplification simulation for Tokyo. A problem of 10.7 BlnDOF and 30 K time steps was solved in 11 h, 32 min using 294,912 CPU cores of the K computer. The average runtime was 1.38 s, which is remarkably fast in terms of time-to-solution.

We should point out the high portability of GAMERA. The architecture of the K computer makes single-precision arithmetic not much faster than double-precision arithmetic. The algorithm using MP arithmetic is still efficient for the K computer, but it is expected that the performance will be improved with other computers, such as Intel CPU- or GPU-based machines, which allow single-precision arithmetic faster than double-precision arithmetic. Indeed, the speed-up due to this algorithm on the K computer was 1.27-fold, while that on an Intel CPU machine was 1.44-fold. A crude estimate shows that the speed-up of the GAMERA would be 4.0 ($= 3.53 \times 1.44/1.27$) with reference to the original PCGE. Currently, the fast solver of GAMERA is being implemented on Intel CPU and GPU machines; a speed-up of 6.99 on an Intel CPU was observed for a small-scale crust deformation problem [68] that used a model of 158 M DOF.

This study has several implications for future systems. GAMERA does not cause any problem with inter-node communication on the K computer, since the algorithm for computing vectors and matrices in the FEM was tuned to minimize data transfer time among the nodes in the most stable manner. Fast inter-node connections with Torus fusion (Tofu) on the K computer also contribute to this smooth inter-node communication. We expect further improvements in inter-node communication technology, which is available for a larger class of parallel computers. In particular, reducing the latency of MPI communication is very desirable in enhancing frequent communication of small amounts of data. Additionally, frequent I/O with large sizes and large numbers of files is involved in constructing the model and visualization of results; we expect further improvements in file I/O technology on a larger class of parallel computers in handling such files.

We explain GAMERA$^{\mathrm{EBE4}}$, which aims to further improve the time-to-solution for solving larger DOF problems. Since preparation of CRS matrices used for the inner coarse loop takes considerable time, we change the CRS used in the inner coarse loop to EBE in GAMERA$^{\mathrm{EBE4}}$. Computation of matrix–vector products by EBE involves more computation than matrix–vector products by CRS, but it is expected that the total time including preparation of CRS matrices can be reduced by using EBE. We also improve the summation of vectors computed by OpenMP in EBE computations in the inner and outer EBE loops, and the computation of block Jacobi matrices in line 4 of Algorithm 1.

Figure 22 shows the size-up efficiency of GAMERA and GAMERA$^{\mathrm{EBE4}}$. As expected, the inner coarse loop is slightly faster using CRS, but the inner coarse loop using EBE takes less time than the sum of preparation of CRS matrices (included in "Others") and the inner coarse loop. Compared with using CRS, using EBE for matrix–vector multiplication involves 5.6 times the floating point operations, but the data used for EBE computation fits inside the 6 MB L2 cache and improves the floating point performance (31.3 % of peak FLOPS is attained for serial performance of EBE in inner coarse loop). Thus, the time used for the inner coarse loop is only increased from 19.1 to 23.2 s for model 1. Combined with the improvement in OpenMP implementations of the inner fine loop, the

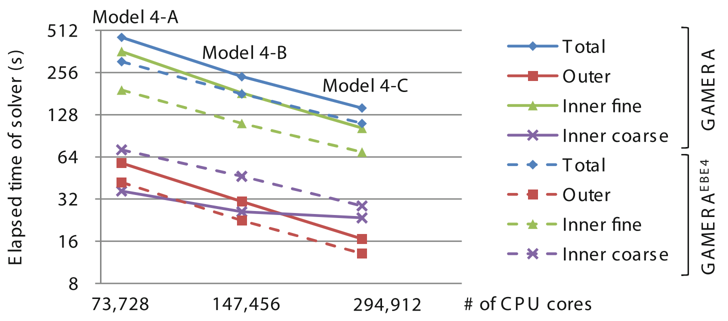


Method	GAMERA				GAMERA^{EBE4}			
Model	1	2	3	4-C	1	2	3	4-C
Total elapsed time (s)	158.9	159.0	166.8	169.8	99.6	103.2	109.0	119.0
Elapsed time for solver (s)	129.6	132.2	141.2	143.0	91.5	94.7	100.8	111.0
FLOPS/Peak of solver (%)	9.00	8.79	8.25	8.19	14.49	14.06	13.28	11.82

Fig. 22 Size-up efficiency of GAMERA and GAMERA^{EBE4}. GAMERA^{EBE4} uses EBE for matrix–vector multiplications in the inner coarse loop instead of CRS used in GAMERA. OpenMP implementations of EBE computations in inner/outer loops are also improved

total elapsed time for solving 100 time steps improved by 59.5 % for model 1 and 42.7 % for model 4-C. The floating point performance has also improved from 9.0 to 14.5 % for model 1, and from 8.2 to 11.8 % for model 4-C. Since the size and frequency of communication is the same for both methods, the ratio of time spent for communication and synchronization has increased in GAMERA^{EBE4} and thus the scalability has decreased from 93.5 to 83.7 % when comparing model 1 and 4-C. Since time-to-solution is greatly improved, it is possible to compute larger problems per CPU core, and thus it is expected that the drop in scalability will not be as bad when used in practice.

Figure 23 shows the speed-up efficiency of the solvers of GAMERA and GAMERA^{EBE4}. The scalability of the inner coarse loop using CRS deteriorates at a smaller number of CPU cores than the inner coarse loop using EBE. On the other hand, the scalability of the total solver drops from 79.8 % (GAMERA) to 68.9 % (GAMERA^{EBE4}), when comparing model 4-A and 4-C. The reason for the drop in scalability can also be due to the increase in ratio of communication and computation time as explained in the size-up efficiency case. From the lower figure, we can see that the time spent for synchronization using CRS does not decrease with the increase in number of CPU cores, while the time spent for synchronization using EBE is decreased by increasing the number of CPU cores. Such difference in synchronization time could be due to the variability in computation time among CPUs due to the difference in matrix structures (e.g., position of non-zero elements, node numbering, and element numbering), the effect of such matrix structures is larger for CRS than for EBE. In these measurements, inner coarse loops using CRS is faster regardless of the number of CPU cores. This can be due to the high B/F of the K computer's SPARC CPUs (64 GB/s/128 GFLOPS = 0.5 bytes/FLOP), which leads to fast access to memory. By changing from CRS to EBE, the total memory usage



Method	GAMERA			GAMERA^{EBE4}		
Model	4-A	4-B	4-C	4-A	4-B	4-C
Elapsed time of solver (s)	456.6	239.4	143.0	306.3	180.0	111.0
FLOPS/Peak (%)	10.3	9.65	8.19	17.1	14.7	11.8

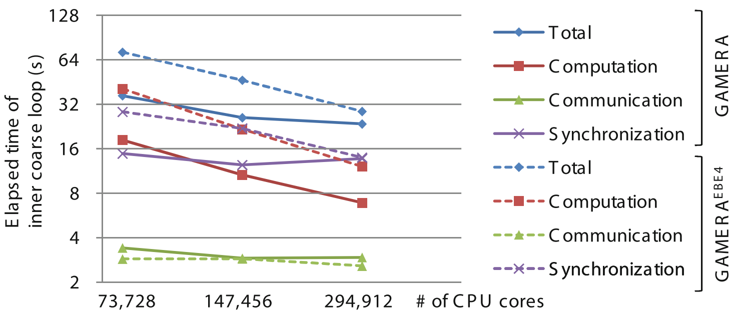


Fig. 23 Speed-up efficiency of GAMERA and GAMERA^{EBE4}

decreases from 1214.9 to 982.8 MB for model 4-A and 584.3 to 411.1 MB for model 4-C, and thus we can increase the problem size per compute node when using EBE.

In summary, we further improved total time-to-solution by GAMERA^{EBE4}, which uses EBE for inner coarse loop together with tuned OpenMP computation. By improving the OpenMP computation of GAMERA, we can expect similar improvements for the inner fine loop. In such case, the choice of CRS or EBE depends on the hardware characteristics; we expect that CRS will be suitable for hardware with large and fast memory, while EBE will be suitable for hardware with small memory but fast floating point computations.

References

1. Day, S.M., Bielak, J., Dreger, D., Graves, R.W., Larsen, S., Olsen, K.B., Pitarka, A.: Tests of 3D Elastodynamic Codes: Final Report for Lifelines Project 1A03, Pacific Earthquake Engineering Research Center (2005)

2. Frankel, A.: Three-dimensional simulations of ground motions in the San Bernardino Valley, California, for hypothetical earthquakes on the San Andreas fault. Bull. Seismol. Soc. Am. **83**, 1020–1041 (1993)
3. Graves, R.W.: Simulating seismic wave propagation in 3D elastic media using staggered-grid finite differences. Bull. Seismol. Soc. Am. **86**, 1091–1106 (1996)
4. Furumura, T., Koketsu, K.: Specific distribution of ground motion during the 1995 Kobe earthquake and its generation mechanism. Geophys. Res. Lett. **25**, 785–788 (1998)
5. Pitarka, A.: 3D elastic finite-difference modeling of seismic motion using staggered grids with non-uniform spacing. Bull. Seismol. Soc. Am. **89**, 54–68 (1999)
6. Bao, H., Bielak, J., Ghattas, O., Kallivokas, L.F., OH́allaron, D.R., Shewchuk, J., Xu, J.: Large-scale simulation of elastic wave propagation in heterogeneous media on parallel computers. Comput. Methods Appl. Mech. Eng. **152**, 85–102 (1998)
7. Koketsu, K., Fujiwara, H., Ikegami, Y.: Finite-element simulation of seismic ground motion with a voxel mesh. Pure Appl. Geophys. **161**, 2463–2478 (2004)
8. Bielak, J., Ghattas, O., Kim, E.J.: Parallel octree-based finite element method for large-scale earthquake ground motion simulation. Comput. Model. Eng. Sci. **10**, 99–112 (2005)
9. Ma, S., Liu, P.: Modeling of the perfectly matched layer absorbing boundaries and intrinsic attenuation in explicit finite-element methods. Bull. Seismol. Soc. Am. **96**, 1779–1794 (2006)
10. Ichimura, T., Hori, M., Kuwamoto, H.: Earthquake motion simulation with multi-scale finite element analysis on hybrid grid. Bull. Seismol. Soc. Am. **97**, 1133–1143 (2007)
11. Ichimura, T., Hori, M., Bielak, J.: A hybrid multiresolution meshing technique for finite element three-dimensional earthquake ground motion modeling in basins including topography. Geophys. J. Int. **177**, 1221–1232 (2009)
12. Moczo, P., Kristeka, J., Galisb, M., Pazaka, P., Balazovjecha, M.: The finite-difference and finite-element modeling of seismic wave propagation and earthquake motion. Acta Phys. Slovaca **57**, 177–406 (2007)
13. Belytschko, T., Liu, W.K., Moran, B.: Nonlinear Finite Elements for Continua and Structures. Wiley, New York (2000)
14. Zienkiewicz, O.C., Taylor, R.L.: The Finite Element Method, 4th edn. Basic Formulation and Linear Problems, vol. 1. McGraw-Hill, London (1989)
15. Wu, S.R.: Lumped mass matrix in explicit finite element method for transient dynamics of elasticity. Comput. Methods Appl. Mech. Eng. **195**, 5983–5994 (2006)
16. Strang, G., Fix, G.J.: An Analysis of the Finite Element Method. Prentice Hall, Englewood Cliffs (1973)
17. Komatitsch, D., Vilotte, J.: The spectral element method: an efficient tool to simulate the seismic response of 2D and 3D geological structures. Bull. Seismol. Soc. Am. **88**, 368–392 (1998)
18. Dumbser, M., Kaser, M.: An arbitrary high order discontinuous Galerkin method for elasticwaves on unstructured meshes ii: the three-dimensional isotropic case. Geophys. J. Int. **167**, 319–336 (2006)
19. Matsumoto, J., Takada, N.: Two-phase flow analysis based on a phase-field model using orthogonal basis bubble function finite element method. Int. J. Comput. Fluid Dyn. **22**, 555–568 (2008)
20. Wijerathne, M.L.L., Oguni, K., Hori, M: Numerical analysis of growing crack problems using particle discretization scheme. Int. J. Numer. Methods Eng. **80**, 46–73 (2009)
21. Flanagan, D.P., Belytschko, T.: A uniform strain hexahedron and quadrilateral with orthogonal hourglass control. Int. J. Numer. Methods Eng. **17**, 679–706 (1981)
22. Belytschko, T., Bindman, L.P.: Assumed strain stabilization of the eight node hexahedral element. Comput. Methods Appl. Mech. Eng. **105**, 225–260 (1993)
23. Lysmer, J., Kuhlemeyer, R.L.: Finite dynamic model for infinite media. J. Eng. Mech. ASCE **95**, 859–877 (1969)
24. Hisada, Y.: An efficient method for computing Green's functions for a layered half-space with sources and receivers at close depths. Bull. Seismol. Soc. Am. **84**, 1456–1472 (1994)

25. Kristeková, M., Kristek, J., Moczo, P., Day, S.M.: Misfit criteria for quantitative comparison of seismograms. Bull. Seismol. Soc. Am. **96**, 1836–1850 (2006)
26. Japan Nuclear Energy Safety Organization: Working report on stochastic estimation for earthquake ground motion. JNES/SAE05–048 (2005)
27. Ma, S., Archuleta, R.J., Morgan, T.: Effects of large-scale surface topography on ground motions, as demonstrated by a study of the San Gabriel Mountains, Los Angeles, California. Bull. Seismol. Soc. Am. **97**, 2066–2079 (2007)
28. Somerville, P., Collins, N., Abrahamson, N., Graves, R., Saikia, C.: Ground Motion Attenuation Relations for the Central and Eastern United States. Final Report, 30 June 2001 [Online]. http://www.earthquake.usgs.gov/hazards/products/conterminous/2008/\99HQGR0098.pdf
29. Second Report of the Nankai Trough Large Earthquake Model Committee, Cabinet Office, Government of Japan, 28 August 2012 [Online]. http://www.bousai.go.jp/jishin/nankai/model/index.html
30. Disaster Assessment of Tokyo Due to Large Earthquakes Such as the Nankai Trough Earthquake, Tokyo Metropolitan Government, 14 May 2013 [Online]. http://www.bousai.metro.tokyo.jp/taisaku/1000902/1000402.html
31. Tiankai, T., Hongfeng, Y., Ramirez-Guzman, L., Bielak, J., Ghattas, O., Kwan-Liu, M., O'Hallaron, D.R.: From mesh generation to scientific visualization: an end-to-end approach to parallel supercomputing. In: Proceedings of the 2006 ACM/IEEE Conference on Supercomputing (SC '06). ACM, New York, NY (2006), Article 91. doi:10.1145/1188455.1188551
32. Yifeng Cui, C., Olsen K.B., Jordan, T.H., Lee, K., Zhou, J., Small, P., Roten, D., Ely, G., Panda, D.K., Chourasia, A., Levesque, J., Day, S.M., Maechling, P.: Scalable earthquake simulation on petascale supercomputers. In: Proceedings of the 2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis (SC '10), pp. 1–20. IEEE Computer Society, Washington, DC (2010). doi:10.1109/SC.2010.45. http://www.dx.doi.org/10.1109/SC.2010.45
33. Rietmann, M., Messmer, P., Nissen-Meyer, T., Peter, D., Basini, P., Komatitsch, D., Schenk, O., Tromp, J., Boschi, L., Giardini, D.: Forward and adjoint simulations of seismic wave propagation on emerging large-scale GPU architectures. In: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis (SC '12), 11 pp. IEEE Computer Society Press, Los Alamitos, CA (2012), Article 38
34. Cui, Y., Poyraz, E., Olsen, K.B., Zhou, J., Withers, K., Callaghan, S., Larkin, J., Guest, C., Choi, D., Chourasia, A., Shi, Z., Day, S.M., Maechling, P.J., Jordan, T.H.: Hysics-based seismic hazard analysis on petascale heterogeneous supercomputers. In: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis, (SC'13), 12 pp. IEEE Computer Society Press, New York, NY (2013), Article 70
35. Hori, M., Ichimura, T.: Current state of integrated earthquake simulation for earthquake hazard and disaster. J. Seismol. **12**(2), 307–321 (2008). doi:10.1007/s10950-007-9083-x
36. Wijerathne, M.L.L., Hori, M., Kabeyazawa, T., Ichimura, T.: Strengthening of parallel computation performance of integrated earthquake simulation. J. Comput. Civil Eng. **27**, 570–573 (2013)
37. Fujita, K., Ichimura, T., Hori, M., Wijerathne, M.L.L., Tanaka, S.: Basic study on high resolution seismic disaster estimation of cities under multiple earthquake hazard scenarios with high performance computing. J. Jpn. Soc. Civil Eng. Ser. A2 (Appl. Mech.) **69**(2), I_415-I_424 (2013) (in Japanese with English abstract)
38. Liang, J., Sun, S.: Site effects on seismic behavior of pipelines: a review. J. Pressure Vessel Technol. (Am. Soc. Mech. Eng.) **122**, 469–475 (2000)
39. Taborda, R., Bielak, J.: Large-scale earthquake simulation: computational seismology and complex engineering systems. Comput. Sci. Eng. **13**, 14–27 (2011)
40. Taborda, R., Bielak, J., Restrepo, D.: Earthquake ground-motion simulation including nonlinear soil effects under idealized conditions with application to two case studies. Seismol. Res. Lett. **83**(6), 1047–1060 (2012)
41. Ichimura, T., Fujita, K., Hori, M., Sakanoue, T., Hamanaka, R.: Three-dimensional nonlinear seismic ground response analysis of local site effects for estimating seismic behavior of buried

pipelines. J. Pressure Vessel Technol. (Am. Soc. Mech. Eng.) **136**, 041702 (2014)
42. Miyazaki, H., Kusano, Y., Shinjou, N., Shoji, F., Yokokawa, M., Watanabe, T.: Overview of the K computer system. FUJITSU Sci. Tech. J. **48**(3), 302–309 (2012)
43. Idriss, I.M., Singh, R.D., Dobry, R.: Nonlinear behavior of soft clays during cyclic loading. J. Geotech. Eng. Div. **104**, 1427–1447 (1978)
44. Masing, G.: Eigenspannungen und verfestigung beim messing. In: Proceedings of the 2nd International Congress of Applied Mechanics, pp. 332–335 (1926) (in German)
45. Akiba, H., Ohyama, T., Shibata, Y., Yuyama, K., Katai, Y., Takeuchi, R., Hoshino, T., Yoshimura, S., Noguchi, H., Gupta, M., Gunnels, J., Austel, V., Sabharwal, Y., Garg, R., Kato, S., Kawakami, T., Todokoro, S., Ikeda, J.: Large scale drop impact analysis of mobile phone using ADVC on Blue Gene/L. In: Proceedings of the 2006 ACM/IEEE Conference on Supercomputing (SC '06), Article 46. ACM, New York, NY (2006). doi:10.1145/1188455.1188503
46. Ogino, M., Shioya, R., Kanayama, H.: An inexact balancing preconditioner for large-scale structural analysis. J. Comput. Sci. Technol. **2**(1), 150–161 (2008)
47. Kawai, H., Ogino, M., Shioya, R., Yoshimura, S.: Large-scale elast-plastic analysis using domain decomposition method optimized for multi-core CPU architecture. Key Eng. Mater. **462–463**, 605–610 (2011)
48. Mandel, J.: Balancing domain decomposition. Commun. Numer. Methods Eng. **9**(3), 233–241 (1993)
49. Earth Simulator (ES) (2014). http://www.jamstec.go.jp/es/en/es1/index.html
50. Saad, Y.: Iterative Methods for Sparse Linear Systems, 2nd edn. SIAM, Philadelphia (2003)
51. Hairer, E., Nørsett, S.P., Wanner, G.: Solving Ordinary Differential Equations I: Non stiff Problems, 2nd edn. Springer, Berlin (1993)
52. Golub, G.H., Ye, Q.: Inexact conjugate gradient method with inner-outer iteration. SIAM J. Sci. Comput. **21**(4), 1305–1320 (1997)
53. Winget, J.M., Hughes, T.J.R.: Solution algorithms for nonlinear transient heat conduction analysis employing element-by-element iterative strategies. Comput. Methods Appl. Mech. Eng. **52**, 711–815 (1985)
54. Zienkiewicz, O.C., Taylor, R.L.: The Finite Element Method for Solid and Structural Mechanics, 6th edn. Elsevier, Amsterdam (2005)
55. Ichimura, T., Hori, M., Bielak, J.: A Hybrid multiresolution meshing technique for finite element three-dimensional earthquake ground motion modeling in basins including topography. Geophys. J. Int. **177**, 1221–1232 (2009)
56. METIS 5.1.0 (2014). http://www.glaros.dtc.umn.edu/gkhome/metis/metis/overview
57. vSMP Foundation, ScaleMP Inc. (2014). http://www.scalemp.com/products/vsmp-foundation/
58. Ajima, Y., Inoue, T., Hiramoto, S., Shimizu, T.: Tofu: interconnect for the K computer. FUJITSU Sci. Tech. J. **48**(3), 280–285 (2012)
59. OpenMPI (2014). http://www.open-mpi.org/
60. MPI: A Message-Passing Interface Standard, Version 2.1 (2014). http://www.mpi-forum.org/docs/mpi21-report.pdf
61. Strong Ground Motion of the Southern Hyogo Prefecture Earthquake in 1995 Observed at Kobe JMA Observatory, Japan Meteorological Agency (2014). http://www.data.jma.go.jp/svd/eqev/data/kyoshin/jishin/hyogo_nanbu/dat/H1171931.csv
62. Stampede at Texas Advanced Computing Center, The University of Texas at Austin (2014). https://www.tacc.utexas.edu/resources/hpc/stampede-technical
63. 5m Mesh Digital Elevation Map, Tokyo Ward Area, Geospatial Information Authority of Japan (2014). http://www.gsi.go.jp/MAP/CD-ROM/dem5m/index.htm
64. National Digital Soil Map, The Japanese Geotechincal Society (2014). http://www.denshi-jiban.jp/
65. Strong-Motion Seismograph Networks (K-NET, KiK-net), National Research Institute for Earth Science and Disaster Prevention (2014). http://www.kyoshin.bosai.go.jp/
66. Housner, G.W.: Spectrum intensities of strong-motion earthquakes. In: Symposium on Earthquakes and Blast Effects on Structures, Los Angeles, CA (1952)

67. Homma, S., Fujita, K., Ichimura, T., Hori, M., Citak, S., Hori, T.: A physics-based Monte Carlo earthquake disaster simulation accounting for uncertainty in building structure parameters. In: The International Conference on Computational Science **29**, 855–865 (2014). doi:10.1016/j.procs.2014.05.077
68. Ichimura, T., Agata, R., Hori, T., Hirahara, K., Hori, M.: Fast numerical simulation of crustal deformation using a three-dimensional high-fidelity model. Geophys. J. Int. **195**, 1730–1744 (2013)

Seismic Response Analysis of Infrastructure

Muneo Hori, Tsuyoshi Ichimura, P. Errol Quninay, and M. Lalith L. Wijerathne

Abstract Large scale simulation of structural seismic response is a key technology for the evaluation of seismic safety of a large infrastructure facility with complicated configuration. This is because highest spatial resolution is needed for such a structure, in order to examine the safety in detail. This chapter explains high performance computing application to structural seismic response simulation based on finite element method. The targets are underground structures, since they are influenced by ground motion of velocity and acceleration as well as ground deformation or strain that provides forces to the structure. Temporal and spatial variation in strong ground motion can be evaluated through a fault-structure system, a model which includes both a fault emitting seismic wave and a target structure and is able to fully consider soil-structure interaction. To analyze this system at high resolution and accuracy, this chapter explains a multi-scale analysis method, paying an attention to the reduction of large computation cost that is needed for required numerical calculation, the construction of a three-dimensional numerical model, and the fast solver that is needed solving the discretized governing equations. Results of structural seismic response simulation of a tunnel and a pipeline network are discussed, to demonstrate the usefulness of high performance computing that enables high resolution analysis to estimate structure seismic safety.

1 Large-Scale Tunnel Structure

Tunnel is an underground structure and subjected to relatively small ground motion. Seismic loading is thus not a critical external force that determines the design of the structure; effects of long-term ground pressure are more concerned in practice. For

M. Hori (✉) • T. Ichimura • M.L.L. Wijerathne
Earthquake Research Institute, The University of Tokyo, Tokyo, Japan
e-mail: hori@eri.u-tokyo.ac.jp; ichimura@eri.u-tokyo.ac.jp

P.E. Quninay
Research Institute for Natural Hazard and Disaster Recovery, Niigata University, Niigata, Japan

S. Yoshimura et al. (eds.), *High-Performance Computing for Structural Mechanics and Earthquake/Tsunami Engineering*, Springer Tracts in Mechanical Engineering,
DOI 10.1007/978-3-319-21048-3_3

large-scale tunnel, however, more attentions are being paid on seismic loading, since it runs through a long distance of various ground structures. In particular, such a large-scale tunnel has rump tunnels that link the main tunnel to ground; the connecting part of the tunnel needs special consideration due to its complicated geometry as well as forces transmitted by the rump tunnel that run in shallower ground.

In order to make most rational design of such a large-scale tunnel structure, we need the following two items: (1) shear evaluation of ground motion that hits this long structure; and (2) accurate analysis of the connecting part of the ramped tunnel and the main tunnel. Application of HPC is a solution for these two items. More specifically, we consider a *fault-structure system*, a three-dimensional (3D) analysis model for a sufficiently large domain which includes a source fault and a target structure in it. For a given scenario of the rupture process, we are able to compute earthquake wave propagation and seismic structure responses. Soil–structure interaction is fully taken into consideration if a fault-structure system of high fidelity is analyzed.

We should mention that remarkable progress has been achieved in computing earthquake wave propagation [2–9]. Also, we should mention that in the field of structural dynamics, 3D numerical simulations with fine spatial discretization are used for the seismic structure analysis [10–14]. From a computational mechanics viewpoint, however, fault-structure system analysis, which combines both the earthquake wave propagation and the seismic response analysis, remains challenging, since this analysis is literally multi-scale; the target domain is in the order of 10^4 while the required resolution is less than 10^{-1} m.

We conduct numerical analysis of a fault-structure system, solving Navier's equation, which is discretized by the finite element method (FEM) in the spatial domain and by Newmark's β method ($\delta = 1/2, \beta = 1/4$) in the temporal domain. This is because the FEM is able to handle complicated geometry and Newmark's β method eliminates limitations due to Courant's condition that are caused by smaller elements of higher stiffness. The resulting equation is written as

$$\left(\mathbf{K} + \frac{2}{\delta t}\mathbf{C} + \frac{4}{\delta t^2}\mathbf{M}\right)\mathbf{u}^{n+1} = \left(\frac{2}{\delta t}\mathbf{C} + \frac{4}{\delta t^2}\mathbf{M}\right)\mathbf{u}^n + \left(\mathbf{C} + \frac{4}{\delta t}\mathbf{M}\right)\mathbf{v}^n + \mathbf{a}^n, \quad (1)$$

where $\mathbf{M}$, $\mathbf{C}$, and $\mathbf{K}$ are the mass, damping, and stiffness matrices, respectively; $\mathbf{u}$ is a displacement vector, with $\mathbf{v}$ and $\mathbf{a}$ being the velocity and acceleration vectors associated with $\mathbf{u}$, respectively; the superscript n for the vectors denotes the nth time step; and δt is the time increment.

We have to consider two computation costs, namely *modeling cost* for constructing a 3D FEM model and *simulation cost* for solving Eq. (1). For a fault-structure system, the both costs are large since the domain is large and the spatial resolution required is high. If the required spatial resolution is on the engineering scale of $10^{-2\sim0}$ m and the target domain size is geological, e.g., $10^{3\sim5}$ m, the degrees of freedom are of the order $10^{15\sim}$. This is extremely large compared to the capabilities of the best computers currently available.

Here, we briefly describe the characteristics of the method that is used for fault-structure system analysis. It is *comprehensive* in the sense that all earthquake processes are computed. This comprehensiveness enables us to estimate the seismic responses of structures in most objective manner, in the sense that they are the output of the fault-structure system as a consequence of a given earthquake scenario. The reliability of the fault-structure system analysis depends on the quality of the data from which the model of the system is constructed. Variability in the seismic response of a structure due to uncertainties in the model can be evaluated by comparing different models.

In this chapter, we first present a modeling method that has reduced modeling and simulation costs in Sect. 1.1. We then describe numerical verification of our tool for fault-structure system analysis, which is made by combining the proposed method and multi-scale analysis, in Sect. 1.2. An example, we demonstrate fault-structure system analysis for a long and complex highway tunnel to which a ramp tunnel is attached in Sect. 1.3. We discuss the seismic responses of the tunnels, paying attention to their dependence on a given earthquake scenario.

1.1 Methodology

We present a method for efficiently constructing a detailed 3D FEM model so that simulation cost is reduced. This method generates a hybrid mesh to handle complicated geometry, employing unstructured tetrahedral and cubic elements whose size is $2^{(\beta-1)}\mathrm{d}s$ for $\beta = 1, 2, \ldots$. Tetrahedral elements are used for thin parts, which include the surface, the layer boundaries, and structures. Cubic elements are used for other parts to avoid geometric approximations. The basis mesh size is determined as

$$\mathrm{d}s = \frac{c_2^{\min}}{n_w f_t},$$

where $c_2^{\min}$, n_w, and f_t are the minimum secondary wave velocity of the target model, the number of elements per target wavelength, and the maximum target frequency, respectively. Although the concept of a hybrid mesh is simple, it is difficult to assign tetrahedral and cubic elements to the suitable parts of a target model. In addition, because the size of the target domain is large and the required resolution is high, the computational cost of mesh generation is high.

We have therefore developed a method that forcibly assigns these elements for the decomposed domains one by one. Denoting V as the target domain, we decompose V as

$$V = V_L \oplus V_S,$$

where V_S is the domain near the structure and V_L is the remainder. We independently generate meshes for V_L and V_S. Here, we employ part of an algorithm proposed in [9] to generate the hybrid mesh for V_L; see Algorithm 1 and Fig. 1. This sequentially

Algorithm 1 For mesh generation for V_L, V_L is discretized by ds-sized unstructured tetrahedral and cubic elements based on the background mesh

```
set structured background mesh whose size is $ds$
for cell in $V_L$ do
    if the cell is intersected by the boundary of $V_L$ then
        the cell is discretized by Delaunay triangulation using the intersect points and vertex of
        the cell. The cell is filled by tetrahedral elements whose nodes are on the intersect points
        or vertex of the cell.
    else
        the cell is used as a cubic element.
    end if
end for
```

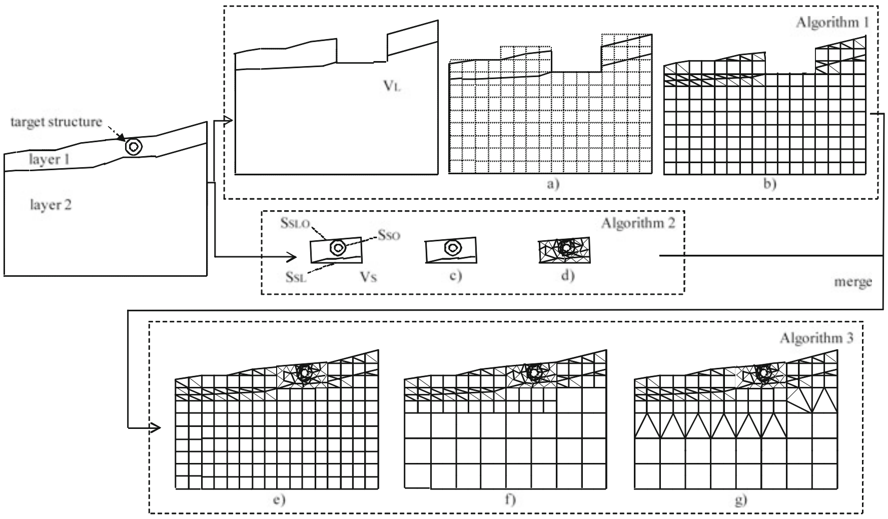


Fig. 1 Schematic of a method for constructing a 3D FEM model. Algorithm 1: the background mesh is covered to V_L in (**a**), and ds-size cubic and tetrahedral elements are generated in (**b**). Algorithm 2: in (**c**) a triangle surface mesh is generated considering compatibility with (**b**), and ds-size tetrahedral elements are generated in (**d**). Algorithm 3: the merged hybrid mesh in (**e**) is rearranged. Groups of eight adjacent inefficient cubic elements are recursively combined into octree arrangements in (**f**), and the incompatibility between different-sized cubic elements is resolved by generating transmit elements in (**g**)

generates cubic and unstructured tetrahedral elements for each cell of V_L using a background mesh and Delaunay triangulation. Because we independently generate mesh for V_L and V_S, mesh incompatibility occurs at the interface between V_L and V_S, denoted by S_S. To avoid this, one simple approach is mesh generation under the constrained conditions caused by the mesh of V_L. However, implementation of this approach is not straightforward, because V_S includes complex structures. To resolve this difficulty, we propose Algorithm 2, which automatically generates compatible meshes. The main feature of this algorithm is that after surface mesh generation, a

Algorithm 2 For V_S mesh generation that is compatible on the interface between V_L and V_S

```
< definition of surfaces set >
$S_S$ is a set of surfaces of $V_S$.
$S_{SL}$ is a subset of $S_S$ which touch $V_L$ on the face.
$S_{SLO}$ is a subset of $S_S$ whose edges touch $S_{SL}$ or $S_{SLO}$ and that are not included in $S_{SL}$.
$S_{SO}$ is a subset of $S_S$ that is not included in $S_{SL}$ and $S_{SO}$.

< definition of function >
$Mesh(A)$ := mesh of domain $A$ or surface $A$
$Node(A)$ := nodes of mesh $A$
$Delaunay2D(A,B)$ := using Delaunay triangulation, generate unstructured triangle elements for surface A using nodes B. When B is *, we use nodes on surface A, which are suitably generated under the discretization criteria.
$Advancing2D(A,B,C)$ := using the advancing front method, generate unstructured triangle elements on surface $A$ using line elements on line $C$ based on nodes $B$.
$Advancing3D(A,B)$ := using the advancing front method, generate unstructured tetrahedral elements in domain $A$ based on surface elements $B$

< surface mesh generation >
$Mesh(S_{SL}) \Leftarrow Delaunay2D(S_{SL}, Node(Mesh(V_L))\ on\ S_{SL})$
$Mesh(S_{SLO}) \Leftarrow Advancing2D(S_{SLO}, Node(Mesh(V_L))\ on\ S_{SLO}, S_{SL} \cap S_{SLO})$
$Mesh(S_{SO}) \Leftarrow Delaunay2D(S_{SO}, *)$

< merge surface meshes >
$Mesh(S_S) \Leftarrow Mesh(S_{SL}) \oplus Mesh(S_{SLO}) \oplus Mesh(S_{SO})$

< generate volume mesh using surface mesh >
$Mesh(V_S) \Leftarrow Advancing3D(V_S, Mesh(S_S))$
```

volume mesh is generated using the surface mesh, and by decomposition of S_S into S_{SL}, S_{SLO}, and S_{SO}, the places where incompatibility problems should be considered (S_{SL} and S_{SLO}) can be limited. We then independently generate the surface mesh for each surface without considering the other surfaces; see Algorithm 2 and Fig. 1 for the definition of S_S, S_{SL}, S_{SLO}, and S_{SO}. Because the meshes for V_L and V_S are compatible at the interface, we can generate the mesh for V simply by merging the meshes for V_L and V_S. These consist of cubic and unstructured tetrahedral elements of size ds.

Although the method mentioned above reduces modeling cost, simulation cost that is needed to compute **Ku** of Eq. (1) remains huge when the dimension of **u** is large and the storage of **K** requires large amounts of memory. To reduce these two, we combine the preconditioned conjugate gradient (PCG) method and the element-by-element (EBE) computation in solving Eq. (1). Instead of the explicit computation of **K** in the PCG method, **Ku** is computed using EBE without storing **K** in memory

$$\mathbf{Ku} = \sum_i \mathbf{Kt}_e^i \mathbf{u}_e^i + \sum_\alpha \sum_\beta \sum_i^{\mathrm{num}(\alpha,\beta)} \mathbf{Kv}_e^i(\alpha,\beta)\mathbf{u}_e^i. \tag{2}$$

Algorithm 3 To combine groups of inefficient cubic elements into an octree arrangement and generate transmit elements to resolve incompatibility

```
< combine inefficient elements >
for cubic elements is in $V$ do
    while (the eight adjoined elements have the same material properties and size) & (element size is smaller than $ds_{cr}/2$) do
        eight adjoined elements are combined as one cubic element, based on the octree arrangement.
    end while
end for

< generate transmit element >
for element whose size is changed do
    if the element is intersected by other elements then
        element is discretized by Delaunay triangulation using the intersect points and vertex of the cell. The cell is filled by tetrahedral elements whose nodes are on the intersect points or vertex of the cell.
    end if
end for
```

Here $\mathbf{Kt}_e^i$ and $\mathbf{Kv}_e^i(\alpha, \beta)$ are the element stiffness matrices for the ith unstructured tetrahedral and cubic elements, whose material properties and size are α and $2^{\beta-1}\mathrm{d}s$, respectively. num(α, β) is the number of cubic elements that have $\mathbf{Kv}_e^i(\alpha, \beta)$. $\mathbf{u}_e^i$ is the displacement associated with the ith element. Although the memory required is reduced if only ds-sized elements are used, we still cannot solve a large-scale problem because the number of elements is so huge. Thus, to reduce the number of elements, using an octree arrangement, we combine *inefficient* cubic elements that are smaller than the critical size $\mathrm{d}s_{\mathrm{cr}}$ (which is used to ensure the required accuracy)

$$\mathrm{d}s_{\mathrm{cr}} = \frac{c_2}{n_w f_t}.$$

Here c_2 is the secondary wave velocity of the element. After this combination (see Algorithm 3 and Fig. 1), the mesh of V becomes an efficient hybrid mesh with unstructured tetrahedral and cubic elements of size $2^{(\beta-1)}\mathrm{d}s$ ($\beta = 1, 2, \ldots$), and the total number of elements which must be stored is reduced. In addition, because there are not many types of $\mathbf{Kv}_e^i(\alpha, \beta)$, it is not effectively estimate

$$\sum_{\alpha} \sum_{\beta} \sum_{i}^{\mathrm{num}(\alpha,\beta)} \mathbf{Kv}_e^i(\alpha, \beta)\mathbf{u}_e^i,$$

by using $\mathbf{Kv}_e^i(\alpha, \beta)$ stored in memory. Implementation of Eq. (2) in the PCG method with the EBE computation gives Algorithm 4, which can efficiently solve Eq. (1).

To reduce the computation cost in strong ground motion simulation and fault-structure simulation, we have proposed the multi-scale analysis and demonstrated

Algorithm 4 Preconditioning conjugate gradient with the element-by-element method to compute $\mathbf{u}^{n+1}$ using $\mathbf{u}^n$, $\mathbf{v}^n$, and $\mathbf{a}^n$ based on Eq. (1). $\mathbf{D}$ (= diagonal parts of $(\mathbf{K}+\frac{2}{\delta t}\mathbf{C}+\frac{4}{\delta t^2}\mathbf{M})$) is the pre-conditioner matrix, which improves the convergence of the target matrix equation

```
$\mathbf{b} \Leftarrow (\frac{2}{\delta t}\mathbf{C} + \frac{4}{\delta t^2}\mathbf{M})\mathbf{u}^n + (\mathbf{C} + \frac{4}{\delta t}\mathbf{M})\mathbf{v}^n + \mathbf{a}^n$
$\mathbf{r} \Leftarrow \mathbf{b} - (\frac{2}{\delta t}\mathbf{C} + \frac{4}{\delta t^2}\mathbf{M})\mathbf{x} + \sum_i \mathbf{Kt}_e^i\mathbf{x}_e^i + \sum_\alpha \sum_\beta \sum_i^{num(\alpha,\beta)} \mathbf{Kv}_e^i(\alpha,\beta)\mathbf{x}_e^i$ for some initial guess solution $\mathbf{x}$ for $\mathbf{u}^{n+1}$
$i \Leftarrow 1$
$\rho_b \Leftarrow 1$
while $\|\mathbf{r}\|_2/\|\mathbf{b}\|_2 \geq$ tolerance do
    $\mathbf{z} \Leftarrow \mathbf{D}^{-1}\mathbf{r}$
    $\rho_a \Leftarrow (\mathbf{r},\mathbf{z})$
    if $i$ is 1 then
        $\mathbf{p} \Leftarrow \mathbf{z}$
    else
        $\beta \Leftarrow \rho_a/\rho_b$
        $\mathbf{p} \Leftarrow \mathbf{z} + \beta\mathbf{p}$
    end if
    $\mathbf{q} \Leftarrow (\frac{2}{\delta t}\mathbf{C} + \frac{4}{\delta t^2}\mathbf{M})\mathbf{p} + \sum_i \mathbf{Kt}_e^i\mathbf{p}_i^u + \sum_\alpha \sum_\beta \sum_i^{num(\alpha,\beta)} \mathbf{Kv}_e^i(\alpha,\beta)\mathbf{p}_e^i$
    $\alpha \Leftarrow \rho_a/(\mathbf{p},\mathbf{q})$
    $\mathbf{x} \Leftarrow \mathbf{x} + \alpha\mathbf{p}$
    $\mathbf{r} \Leftarrow \mathbf{r} - \alpha\mathbf{q}$
    $\rho_b \Leftarrow \rho_a$
    $i \Leftarrow i + 1$
end while
$\mathbf{u}^{n+1} \Leftarrow \mathbf{x}$
```

its usefulness (e.g., see [1] for a fault structure analysis and [8] for strong ground motion simulation). The multi-scale analysis procedure is summarized below (see [1] for details).

1. Construct a 3D numerical simulation model, including the fault-structure system (Fig. 2a).
2. Using the singular perturbation method, perform multi-scale analysis to decompose the original 3D model into two-step models consisting of macro- and micro-analysis (Fig. 2b), i.e.,

   macro-analysis: Compute the earthquake motion for the entire target model from fault to surface at low spatial resolution (on geological length scales).

   micro-analysis: Compute the structural seismic response within a small region near the target structure at high spatial resolution, using the results of macro-analysis (on engineering length scales).

Although multi-scale analysis can drastically reduce one-time computational costs, there still remains the difficulty of constructing a 3D FEM model and solving Eq. (1) in the large-scale problem.

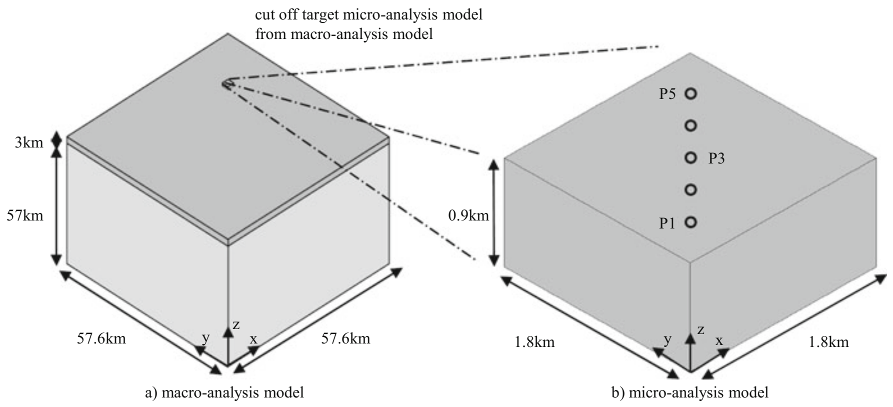


Fig. 2 Schematic view of multi-scale analysis. (**a**) 3D model including fault-structure system. (**b**) Decomposed systems by multi-scale analysis

1.2 Numerical Experiment

In this section, we present verification of our tool for fault-structure system analysis, which is developed by combining the multi-scale analysis and the methods presented in the previous section. An example of a fault-structure system, used is a long highway tunnel attached to a ramp tunnel as the target structure. We discuss the seismic responses of the tunnels in a given earthquake scenario. In the following numerical simulations, we use the following discretization settings:

- Four-node tetrahedral and eight-node cubic elements based on first-order interpolation of displacement.
- The target frequency is set at 1 Hz and the mesh size is defined such that one wavelength at 1 Hz is discretized by at least ten elements. Frequency components of the simulation results except [0, 1.0] Hz are removed by a band pass filter (0.0–0.1 Hz, 0.9–1.0Hz).
- The time increment is $\delta t = 0.01$ s, while the target time duration is 40.96 s.
- The Rayleigh damping matrix is used ($\mathbf{C}^e = a\mathbf{M}^e + b\mathbf{K}^e$, where $(\ )^e$ indicates the element matrix). The parameters a and b are set based on an anelastic quality factor (Q):

$$\min_{a,\,b}\left[\int_{f_{\min}}^{f_{\max}}\left(\frac{1}{2Q}-\left(\frac{a}{2\,2\pi f}+\frac{b\,2\pi f}{2}\right)\right)^2 df\right],$$

  where $f_{\max}$ and $f_{\min}$ are the maximum and minimum target frequencies. We set $f_{\max} = 1.0$ Hz and $f_{\min} = 0.1$ Hz.
- In Algorithm 4, we set 10^{-6} as the tolerance and the previous displacement as the initial guess solution $\mathbf{x}$.

- To eliminate outgoing waves from the target domain, absorbing boundary conditions (viscous damper [15] and an absorbing boundary region [16]) are applied to all surfaces of the model except the top.

1.2.1 Numerical Verification

The accuracy of the fault-structure system analysis is examined by using a problem which has a semi-analytic solution, i.e., a wave-field problem in horizontally layered media excited by a single point source; see Table 1 for details. We conduct a wave-propagation simulation using the proposed method, and compare wave profiles at observation points with Green's function solutions [17].

Figure 3 and Table 2 show numerical simulation models for macro- and micro-analysis. First, we conduct wave simulations in the macro-analysis model excited by a single point source. Next, we perform wave simulations in the micro-analysis model excited by the results of the macro-analysis. The construction of the FEM model and solution of the discretized matrix equation are conducted using the methods shown in the previous section.

Figure 4 shows the wave profiles of the micro-analysis and Green's function solution at several observation points. As can be seen, these wave profiles are in close agreement, so our tool can be used to evaluate a fault-structure system.

We also conduct wave simulations with high-impedance contrast. Material properties of only top layer are changed to (ρ, V$_p$, V$_s$, Q) = (2150 kg/m^3, 2600 m/s, 1500 m/s, 150.0). Although the resulting impedance contrast is high, our method can reproduce Green's function solution with high accuracy as shown in Fig. 5.

Table 1 Horizontal layered half space problem setting

Depth of top layer	3 km
Top layer	$\rho = 2300$ kg/m^3
	$V_p = 4700$ m/s
	$V_s = 3000$ m/s
	$Q = 250.0$
Half space	$\rho = 2800$ kg/m^3
	$V_p = 7400$ m/s
	$V_s = 6000$ m/s
	$Q = 800.0$
Excitation (point source)	$\begin{cases} M_0(2t^2/T_0^2) & 0 \le t \le T_0/2 \\ M_0(1-2(t-T_0)^2/T_0^2) & T_0/2 \le t \le T_0 \\ M_0 & T_0 \le t \end{cases}$
Strike, dip, rake:	30°, 40°, 50°
Magnitude (M_0):	1.0×10^{15} Nm
Rise time (T_0):	2 s
Source location:	(30.3 km, 30.3 km, 52.5 km)

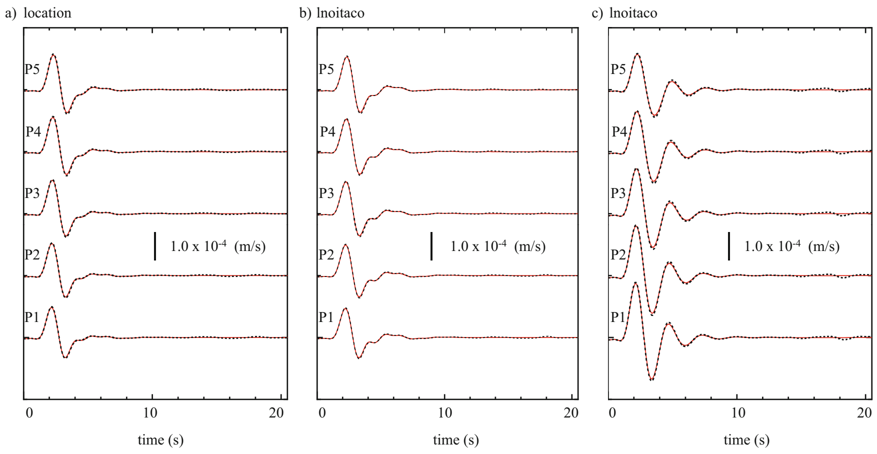


Fig. 3 Numerical simulation model for verification. (**a**) The macro-analysis model: a horizontal layered half space model. (**b**) The micro-analysis model: the target domain is separated from the macro-analysis model. Pi indicates the observation points, located at (32.4 + 0.3 i km, 32.4 + 0.3 i km, 60 km) on the top surface ($i = 1, 2, 3, 4, 5$)

1.2.2 Application Example

As an example, we estimate the seismic responses of a large-scale complex underground highway junction in a major earthquake in Tokyo; see Fig. 6 for the location of the target structure and fault plane. A model of an actual underground expressway junction is chosen as the target structure. The seismic response of this structure has been researched in 3D simulations based on soil-structure systems [13, 18]. Figure 7 shows the details of the target structure. This underground highway junction has an underground connection between the main tunnel and a ramp tunnel that passes through bedrock to a soft soil layer. The inclination of the ramp tunnel is about 4 degrees. The length, bore, and external diameter of the tunnel are 679, 11.77, and 12.83 m, respectively. The thickness and bore width of the ramp tunnel are 1 and 10 m, respectively. The target structure is made of reinforced concrete. The depth of the soft soil layer around the target structure is about 21 m.

Next, as near-field major earthquakes are expected to occur directly beneath urban areas of Tokyo, the effect of such an earthquake on this target structure is examined. The relevant properties are summarized in Table 3; see [19] for details. In the fault model, asperity is indicated by the gray rectangle and the background area is indicated by the white rectangle, as shown in Fig. 6; see Table 3 for details. To investigate the effects of the epicenter location, we conduct case studies 1 and 2, as shown in Fig. 6.

The target domain for macro-analysis is designated by a dotted line in Fig. 6. The size of the domain is 38.4 × 38.4 × 50 km. The 3D crust and soil structure are estimated based on [20, 21]; this crust and soil structure consist of six layers. The material properties of each layer are summarized in Table 4(a). Figure 8 shows

Table 2 Macro- and micro-analysis models for horizontal layered half space problem setting

(a) Macro-analysis model	
Domain size	$0\,\text{km} \le x \le 57.6\,\text{km}$
	$0\,\text{km} \le y \le 57.6\,\text{km}$
	$0\,\text{km} \le z \le 60\,\text{km}$
Depth of top layer	3 km
Top layer	$\rho = 2300\,\text{kg/m}^3$
	$V_p = 4700\,\text{m/s}$
	$V_s = 3000\,\text{m/s}$
	$Q = 250.0$
Half space	$\rho = 2800\,\text{kg/m}^3$
	$V_p = 7400\,\text{m/s}$
	$V_s = 6000\,\text{m/s}$
	$Q = 800.0$
Excitation (point source)	$\begin{cases} M_0(2t^2/T_0^2) & (0 \le t \le T_0/2) \\ M_0(1-2(t-T_0)^2/T_0^2) & (T_0/2 \le t \le T_0) \\ M_0 & (T_0 \le t) \end{cases}$
Strike, dip, rake	30°, 40°, 50°
Magnitude (M_0)	1.0×10^{15} Nm
Rise time (T_0)	2 s
Source location	(30.3 km, 30.3 km, 52.5 km)

(b) Micro-analysis model	
Domain size	$32.4\,\text{km} \le x \le 34.2\,\text{km}$
	$32.4\,\text{km} \le y \le 34.2\,\text{km}$
	$59.1\,\text{km} \le z \le 60\,\text{km}$
Top layer	$\rho = 2300\,\text{kg/m}^3$
	$V_p = 4700\,\text{m/s}$
	$V_s = 3000\,\text{m/s}$
	$Q = 250.0$
Excitation	Results of macro-analysis

the numerical model for macro-analysis that is automatically generated by the proposed algorithms. Here we can see that the complicated geometry of the ground surface and the interface of the crust layer are appropriately modeled. Structured and unstructured elements with multi-resolution are generated corresponding to the material properties of each layer. The numbers of nodes and elements of the macro-analysis model are summarized in Table 5. The fault rupture process is modeled as a combination of point sources, whose source time function is based on [22]. First, to check the validity of the 3D crust structure model, we reproduce the ground motion that results from an actual small earthquake in this domain by using the proposed method. The results are compared to the earthquake motion that is recorded at ten observation points [23]. The target earthquake and the source

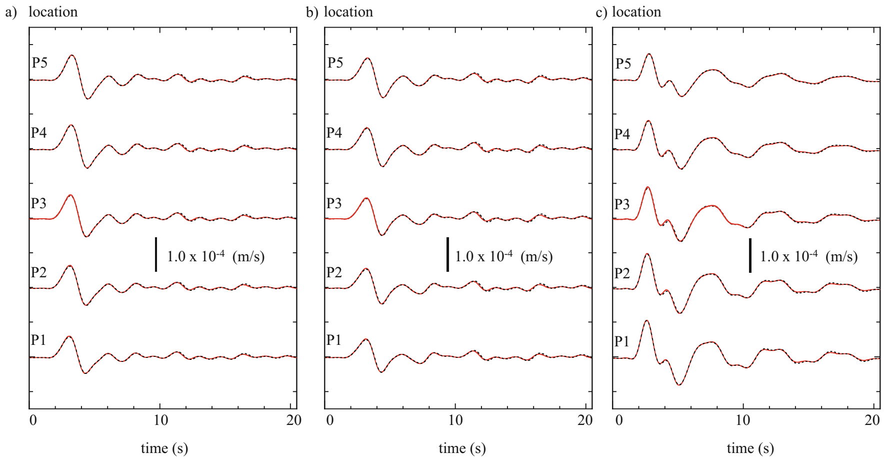


Fig. 4 Velocity waveforms (in m/s): comparison of micro-analysis (*dotted*) with Green's function (*solid*) solutions for layered half space problem. (**a**) *x* component. (**b**) *y* component. (**c**) *z* component

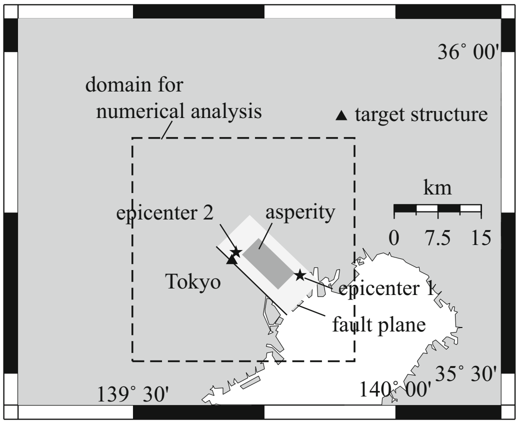


Fig. 5 Velocity waveforms (in m/s): comparison of micro-analysis (*dotted*) with Green's function (*solid*) solutions for layered half space problem with high-impedance contrast. (**a**) *x* component. (**b**) *y* component. (**c**) *z* component

parameters are summarized in Table 6 [24, 25]. Comparison of the computed motion and the observed motion is shown in Fig. 9. The amplitude and phase properties are reproduced with comparatively good accuracy. Using this 3D crust structure model, we conducted a macro-analysis for the earthquake scenario shown in Table 3.

Next, we perform a micro-analysis of the domain near the target structure using the results of the macro-analysis. The micro-analysis model is shown in Fig. 10. The material properties are presented in Table 4(b). To include nonlinear effects of the soft soil caused by the target earthquake, we estimate the equivalent linear

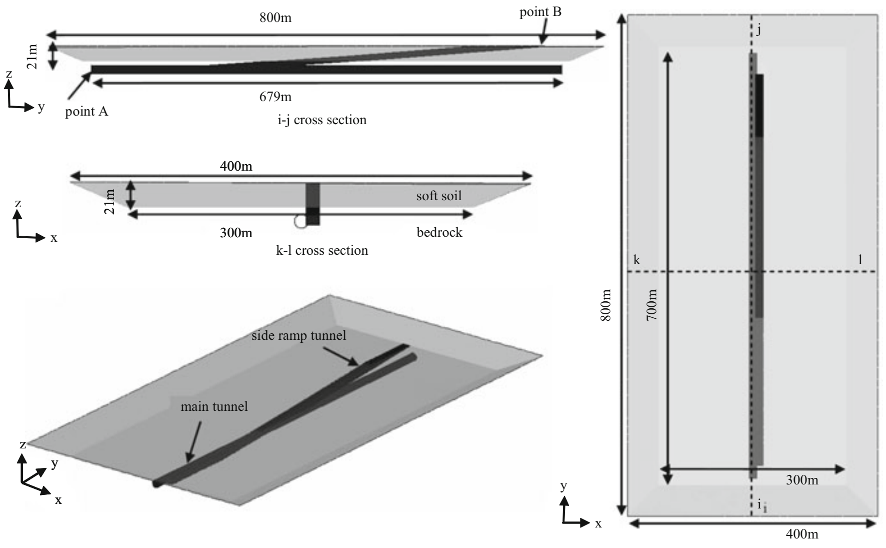


Fig. 6 Earthquake beneath urban areas of East Tokyo ($M = 6.9$). Location of fault plane and target structure

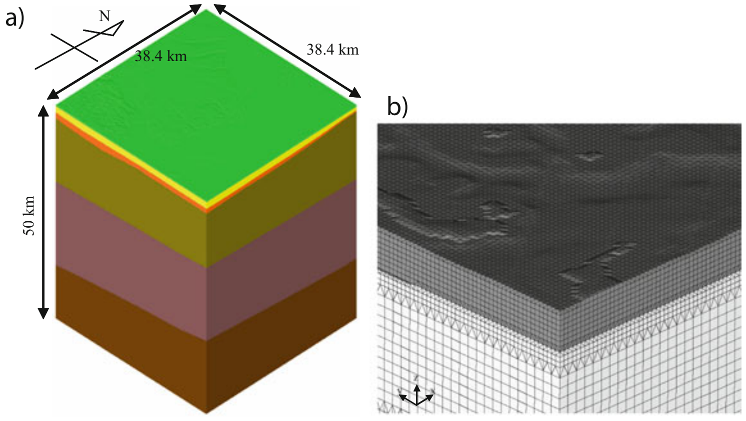


Fig. 7 Target underground expressway junction

elastic material properties using a conventional method [18]. The number of nodes and elements of the micro-analysis model are shown in Table 5. The size of the micro-analysis model is determined based on the discussion presented in [1]. The coordinate system is set with x-, y-, and z-axes corresponding to the east–west, north–south, and up–down directions, respectively.

Figure 11 shows snapshots of the complicated velocity distribution on the top surface in the macro-analysis for case 1. It is strongly affected by the fault processes

Table 3 Earthquake beneath urban areas of East Tokyo ($M = 6.9$)

Location of upper left corner	35.590N, 139.792E
Depth of upper edge	6 km
Strike, dip, rake	315°, 45°, 90°
Length, width	17.38 km, 11.22 km
Rigidity	3.4×10^{10} N/m^2
Rupture velocity	2.5 km/s
fmax	6 Hz
Asperity	Average slip = 2.99 m
	Area = 49 km^2
	Fault width = 5.6 km
	$\Delta\sigma$ = 12 MPa
Background	Average slip = 1.20 m
	Area = 146 km^2
	Fault width = 17.38 km
	$\Delta\sigma$ = 2.4 MPa

Properties of fault plane for scenario earthquake: all the parameters are set based upon [19]. Epicenter 1 and 2 located at (35.652N, 139.817E) and (35.688N, 139.695E)

Table 4 Material properties for scenario earthquake simulation based upon [13, 20, 21]

(a) For macro-analysis

	ρ (kg/m^3)	V$_p$ (m/s)	V$_s$ (m/s)	Q
1st layer	1950	1850	500	60
2nd layer	2150	2560	1000	150
3rd layer	2300	3200	1700	200
4th layer	2700	5800	3360	500
5th layer	2800	6600	3700	600
6th layer	4400	8040	4480	800

(b) For micro-analysis

	ρ (kg/m^3)	V$_p$ (m/s)	V$_s$ (m/s)	Q
Soft soil	1500	199	60	2.78
Bedrock	1950	1850	500	60
Tunnel	2500	3373	2127	100

and 3D crust structure. Concentration of the velocity distribution is observed locally near the target structure. This is caused by the directivity of the fault.

Using the micro-analysis model and the low resolution solution obtained by the macro-analysis, we conduct the micro-analysis which produces the high resolution solution; see [8] for a detailed explanation about the refinement of the low resolution solution made by the micro-analysis. Figure 12 shows the deformation in case 1, a total of 0, 7.75, 8.25, 8.75, 9.25, and 9.75 s after the fault rupture. In the figure, the

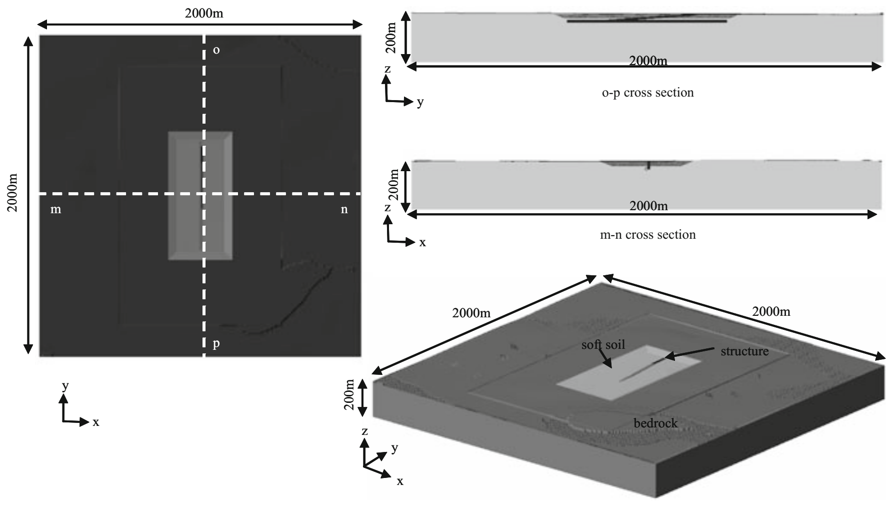


Fig. 8 3D model for macro-analysis. (**a**) Isometric view. (**b**) Close-up view

Table 5 Number of nodes and elements of macro- and micro-analysis model

	Macro-analysis	Micro-analysis
Number of degree of freedom	95,859,312	2,265,330
Number of tetrahedra elements	60,447,310	1,973,207
Number of cubic elements	21,179,520	348,575

Table 6 Properties of target observed earthquake [24, 25]

Date	2000.8.18
Epicenter location	35.7N, 139.7E
Depth	35 km
Strike, dip, rake	59°, 74°, 65°
Magnitude	Mw3.8
Rise time	0.18 s

magnification is set to 50×, and deformation is relative to point A in Fig. 7. As seen in Fig. 12, the structure is strongly deformed in part of the interface as a result of the difference in material properties.

Figure 13 shows the point-wise estimations, which indicate the displacement wave profiles at points A and B for cases 1 and 2. Figure 14 shows the maximum deformation of the structure. The maximum displacement of case 1 is 25.77 cm, which occurs at 8.70 s. For case 2, it is 5.92 cm at 8.38 s. These results show that

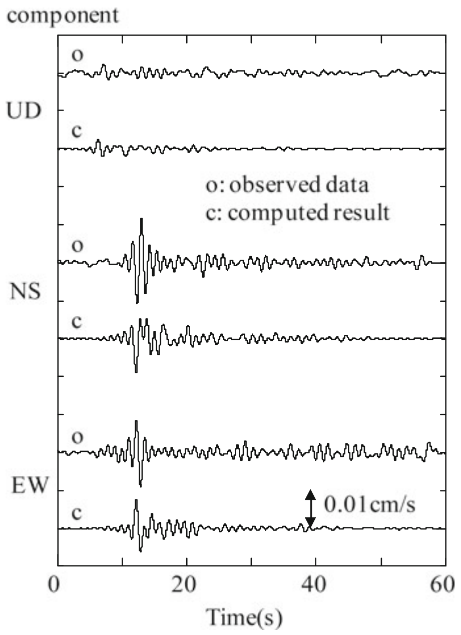


Fig. 9 An example of comparison between computed and observed earthquake ground motion (in cm/s) at an observation point [TFD-018 at (35.65934N, 139.80762E)]

the structural response in case 1 is considerably larger than that in case 2. One possible reason for this is that the rupture on the fault plane progresses toward the observation point in case 1, while it progresses away from the observation point in case 2. Thus, even if the energy emitted from the fault plane is the same, the directivity effect can vary greatly depending on the location of the rupture starting point on the fault plane. Hence, the structural response will also differ. This large-scale complex structure shows a complicated seismic response due to spatial and temporal variation in the earthquake ground motion. The difference between cases 1 and 2 is remarkable, even though the only difference is the location of the epicenter. These results highlight the usefulness of comprehensive numerical analysis of fault-structure systems.

While the directivity effect and phase difference can be directly attributed to the 3D soil-crust structure and the source setting, there are other features of the problem that affect the response of the structure. One is the surface topography, which has been reported in a number of studies [26, 27] to cause effects such as amplification, attenuation, scattering of surface waves, and increased duration of shaking. We emphasize that our method accounts for these topographic effects. There may be significant differences in the response of a structure between models that include topography and those that use a flat surface approximation.

Another important feature is the soil–structure interaction of underground structures. A number of analytical [28, 29] and numerical studies [30–32] have focused

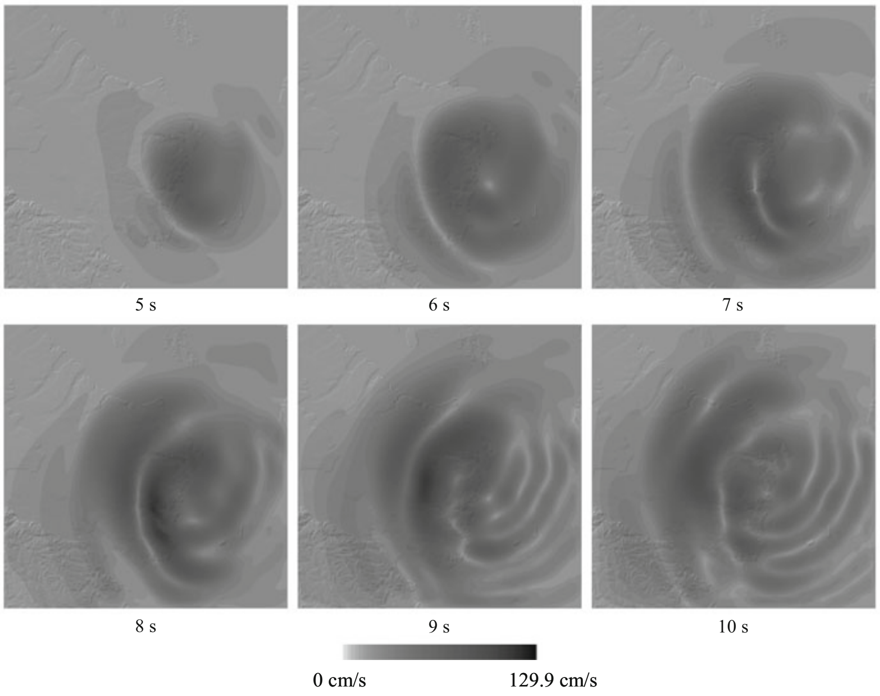


Fig. 10 3D model for micro-analysis

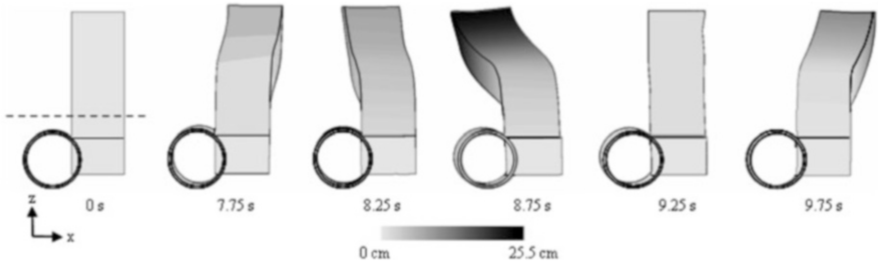


Fig. 11 Snapshots of velocity norm distribution on top surface of macro-analysis model in case 1

on topics related to this. Examples include the structural response to spatially varying ground motion or transversely varying soil types, the influence of the mode of vibration of the soil, comparisons between two-dimensional and 3D analysis, and analysis of the effect of the spacing of two tunnel structures. While these studies have identified the importance of analyzing different aspects of soil–structure interaction, they have limitations, such as the configuration of the structures or the boundary conditions. Extension to large-scale realistic models is not straightforward. In our fault-structure analysis system, all of the necessary components of such large-scale problems are considered and important details are

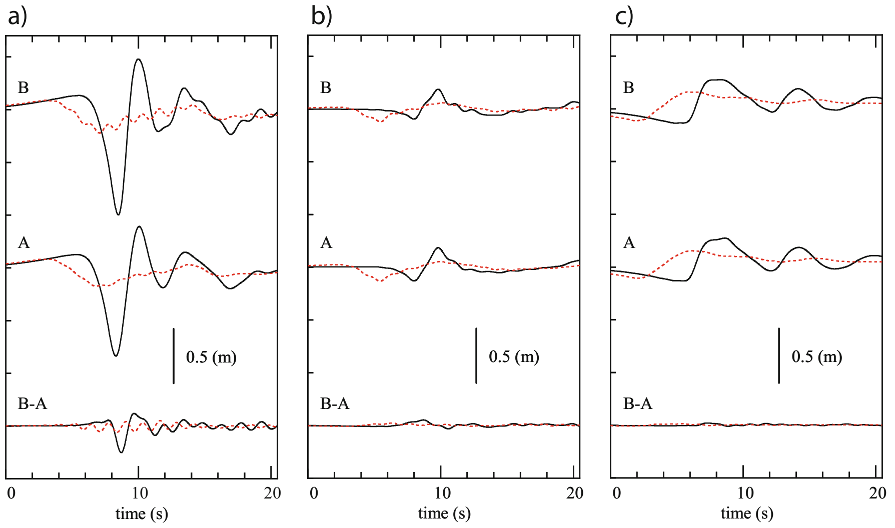


Fig. 12 Snapshots of structural behavior in case 1 (deformation × 50): the relative deformation with respect to point A shown in Fig. 7 is presented. *Dotted lines* indicate the interface between soft soil and bedrock

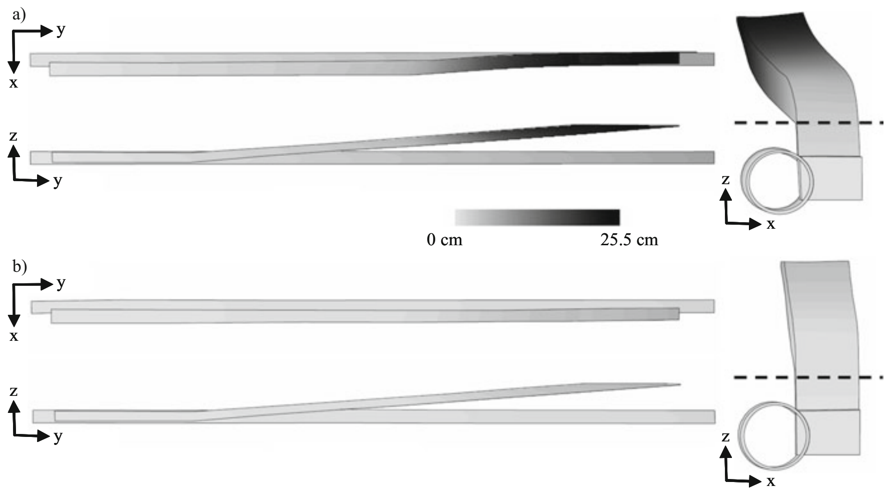


Fig. 13 Displacement wave profiles: *A* and *B* indicate the displacement wave profiles at points *A* and *B*, respectively, in Fig. 7. *B-A* indicates the difference. The *solid* and *dotted lines* indicate the results of cases 1 and 2. (**a**) *x* component. (**b**) *y* component. (**c**) *z* component

given sufficient attention (for example, detailed fault plane rupture and subsurface layers, and high spatial resolution in soft soil-structure models) without imposing constraints. Hence, comprehensive analysis of any large-scale structure can be performed.

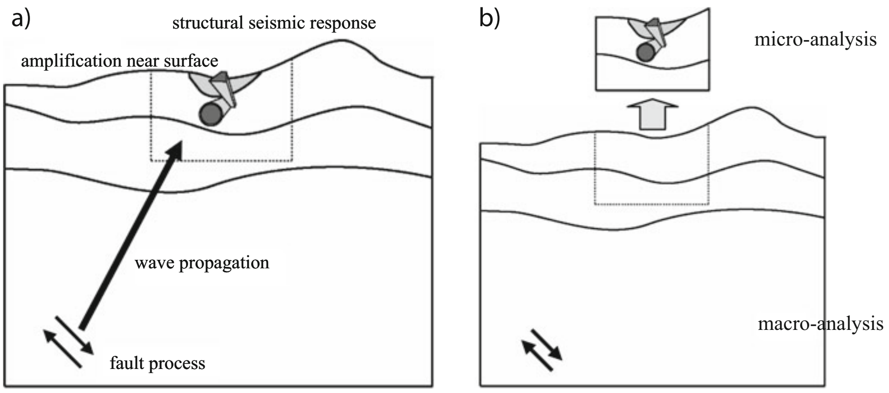


Fig. 14 Maximum deformation of structure (deformation × 50): the relative deformation with respect to point A shown in Fig. 7 is presented. The *dotted line* indicates the interface between soft soil and bedrock. (**a**) Case 1. (**b**) Case 2

1.3 Summary

We present efficient methods for constructing a detailed FEM model and for solving large-scale problems. By combining these methods with multi-scale analysis, we develop a comprehensive numerical analysis tool for a fault-structure system in order to compute a large-scale seismic responses in a given earthquake scenario. The accuracy of the developed tool is examined by comparing it to Green's function. We estimate the seismic response of a complex large-scale underground highway junction in the event of a major earthquake in Tokyo. The numerical results indicate the importance of considering the fault-structure system, along with the effects of various factors on the structural seismic response. To reproduce the actual behavior of structures with higher accuracy, we are planning to verify the validity of the present fault-structure system analysis by comparing the simulation results with the observed data; nonlinear constitutive laws for soils and structures are being implemented in the FEM code.

2 Buried Pipeline Network

Non-uniformity of ground is a major factor in pipeline damage induced by seismic ground motion; see [33] for a review of seismic behavior of pipelines. Non-uniform ground exists, for example, where there is a cut-and-fill area composed of two distinct soil layers. Indeed, a disproportionately large incidence of pipeline damage has been reported in cut-and-fill areas (e.g., [33, 34]). Seismic pipeline damage can occur due to strain concentration, where the impedance contrast between two layers induces an amplification of velocity or acceleration, as well as strain.

To engineer seismic resistance in pipelines, ground strain, rather than ground velocity or acceleration, should be taken into consideration, in order to determine the external force that acts on the pipelines (e.g., [35, 36]). Estimation of the distribution of on-site seismic ground strain should be the first phase in engineering seismic resistance in pipelines in areas of non-uniform ground. However, it is difficult to observe the ground strain distribution since a time-synchronized network with a resolution of $\sim 10^{-1}$ m is required for such observation in non-uniform ground. For this reason, we aimed to develop a method to simulate seismic behavior in non-uniform ground. The simulation ought to be able to analyze the strain concentration near the interface between different layers. Therefore, a three-dimensional (3D) analysis with high spatial resolution is a unique solution. It should also account for the nonlinearity of soil. The finite element method (FEM) is suitable for solving problems with complex geometry, and nonlinear constitutive relations can be implemented.

In this chapter, we describe the development of a 3D nonlinear FEM simulation that can evaluate seismic ground strain. We then apply the model to a large-scale 3D seismic response analysis of a site with a complex underground soil structure. The numerical results are compared with observed seismic ground motions, and local amplification processes are discussed focusing on the local strain. The remainder of the chapter is organized as follows. First, we describe the method developed for the pipeline analysis in Sect. 1. Next, we analyze a target site, compare the simulated data with observed seismic ground motion, and discuss wave amplification processes based on the numerical analysis in Sect. 2. We make a few concluding remarks in Sect. 1.3.

2.1 Nonlinear Seismic Ground Response Analysis

FEM is suitable for analyzing the response of structures with a complex geometry and for satisfying traction-free boundary conditions. Here, we use a nonlinear, dynamic FEM model with 3D solid elements. The equations to be solved are written as follows:

$$\left(\frac{4}{\mathrm{d}t^2}\mathbf{M} + \frac{2}{\mathrm{d}t}\mathbf{C}^n + \mathbf{K}^n\right)\delta\mathbf{u}^n = \\ \mathbf{F}^n - \mathbf{Q}^{n-1} + \mathbf{C}^n\mathbf{v}^{n-1} + \mathbf{M}\left(\mathbf{a}^{n-1} + \frac{4}{\mathrm{d}t}\mathbf{v}^{n-1}\right), \tag{3}$$

with

$$\begin{cases} \mathbf{Q}^n = \mathbf{Q}^{n-1} + \mathbf{K}^n\delta\mathbf{u}^n, \\ \mathbf{u}^n = \mathbf{u}^{n-1} + \delta\mathbf{u}^n, \\ \mathbf{v}^n = -\mathbf{v}^{n-1} + \frac{2}{\mathrm{d}t}\delta\mathbf{u}^n, \\ \mathbf{a}^n = -\mathbf{a}^{n-1} - \frac{4}{\mathrm{d}t}\mathbf{v}^{n-1} + \frac{4}{\mathrm{d}t^2}\delta\mathbf{u}^n. \end{cases} \tag{4}$$

Here, $\delta\mathbf{u}$, $\mathbf{u}$, $\mathbf{v}$, $\mathbf{a}$, and $\mathbf{F}$ are vectors describing incremental displacement, displacement, velocity, acceleration, and external force, respectively; $\mathbf{M}$, $\mathbf{C}$, and $\mathbf{K}$ are matrices for mass, damping, and stiffness; dt is time increment; and n is time step. We used Rayleigh damping, where the element damping matrix $\mathbf{C}_e^n$ is calculated using the element mass matrix and the element stiffness matrix, $\mathbf{K}_e^n$ and $\mathbf{M}_e$, as follows:

$$\mathbf{C}_e^n = \alpha \mathbf{M}_e + \beta \mathbf{K}_e^n.$$

The coefficients α and β are determined by solving the least-squares equation, i.e.,

$$\text{minimize}\left[\int_{f_{\min}}^{f_{\max}} \left(h^n - \frac{1}{2}\left(\frac{\alpha}{2\pi f} + 2\pi f \beta\right)\right)^2 df\right],$$

where $f_{\max}$ and $f_{\min}$ are the maximum and minimum target frequencies, and h^n is the damping ratio at time step n.

Note that the damping matrix is calculated at every time step, since the stiffness and damping ratio changes with time. Small elements are generated locally when modeling complex geometry with solid elements. Satisfying the Courant condition when using explicit time integration methods (such as the central difference method) leads to small time increments and considerable computational expense. In order to resolve this problem, we use the Newmark-β method for time integration (with $\beta = 1/4, \delta = 1/2$) and semi-infinite absorbing boundary conditions for the bottom and side boundaries of the simulation domain, and the modified Ramberg–Osgood model [37] and Masing rule [38] for the nonlinear constitutive relations describing the soil. The calculation proceeds as follows:

1. Compute the stiffness and damping ratio for nth time step using the Ramberg–Osgood model and Masing rule with the strain obtained at the $n - 1$th time step.
2. Compute $\mathbf{K}^n$ and $\mathbf{C}^n$ using the stiffness and damping ratio for the nth time step.
3. Compute $\delta\mathbf{u}^n$ by solving Eq. (3) and update the values in Eq. (4) using $\delta\mathbf{u}^n$.

Since we aim to evaluate strain in complex soil structures, we use non-structured, second-order tetrahedral elements to construct the 3D FEM model. The size of the simulation domain will be of the order of $10^3 \times 10^3 \times 10^2$ m, and the required spatial resolution should be the order of $\sim 10^{-1}$ m to satisfy a temporal resolution of a few hertz; therefore, the number of degrees of freedom (DOFs) in the model becomes large, of the order of 10^7 to 10^9. Furthermore, the number of time steps required for the analysis should be of the order of 10^3 to 10^4. The computational cost of constructing such a large 3D FEM model is expected to be considerable, especially when the problem is nonlinear.

A fully automated meshing algorithm, which can generate elements of a high quality, is desirable for constructing the 3D FEM model. High quality element does not have a large aspect ratio, as this leads to stiffness and a loss of accuracy. Although it is common to perform manual tuning in order to avoid elements with poor quality, doing so with large models can become time consuming. For this reason, a number of methods have been developed for automatically generating FEM meshes. We use a modified version of the method of [39] for constructing the 3D mesh, with the soil structure defined as a layered medium. In the method described by Ichimura et al. [39], a structured grid is used to break up the simulation domain into a number of cells, and elements are constructed for each cell. The elements are generated in each cell with small aspect ratios, and this leads to a fully automated, robust mesh of high-quality elements. Since the generation of elements in each cell is performed individually, the problem is suitable for parallelization, leading to a reduction in the time for meshing. We generate linear-tetrahedral elements and cubic elements using the method described by Ichimura et al. [39] and then convert both types of element to second-order tetrahedral elements.

The unknowns in Eq. (3) are $\delta\mathbf{u}^n$. We solve this $\delta\mathbf{u}^n$ at each time step, in order to obtain the history of $\mathbf{u}$. Most of the computational cost is used in solving the matrix equation of Eq. (3). We use the Block-Jacobi method, with mixed precision arithmetics, the geometric multigrid method, and the predictor-corrector method in solving the matrix equation. The problem is parallelized using OpenMP together with MPI.

2.2 Analysis of Local Site Effects

The aim of the present study is to reproduce the seismic ground motion during the 2011 Tohoku Earthquake, which occurred on March 11, 2011, and to compare the simulated results with the measured seismic ground motion. Using the simulated results, we discuss the distribution of strain in association with the damage to buried pipelines. We consider a region in Yokohama City, Kanagawa Prefecture, Japan, where soft sedimentary layers form a complex soil structure. The site is known to have significantly larger observed seismic ground motion compared with that of nearby observation sites during earthquakes.

We model a region with dimensions of 1696 m in the east–west direction and 1920 m in the north–south direction, with a 50 m grid. The surface topography is extracted from a 50-m grid digital elevation map produced by the Geospatial Information Authority of Japan, and the soil layer properties are extracted from borehole data from the Super-dense Real-time Monitoring of Earthquakes (SUPREME) system [40] and the Administration-Geographic Information System Database of Yokohama City [41]. The site has a three-layer structure. The first layer consists of clay, silt, and humus soil with N-values in the range 0–2; the second layer consists of sand, gravel, and consolidated silt, with an N-value of approximately 50; and the bedrock layer consists of hardpan and mudstone. We employ the soil parameters of the modified Ramberg–Osgood model, which are determined by the Central Disaster Management Council in Japan [42] (clay for layer 1 and gravel for layer 2); see Table 7. The Metropolitan Seismic Observation Network (MeSO-net), a high-density and high-accuracy ground motion observation network, operates in

Table 7 Material properties of the soil structure

	V_p (m/s)	V_s (m/s)	ρ (kg/m^3)	$h_{\max}$	γ_r
1st layer	700	100	1500	0.23	0.007
2nd layer	1400	300	1800	0.23	0.001
Bedrock	2100	700	2100	0.01	∞

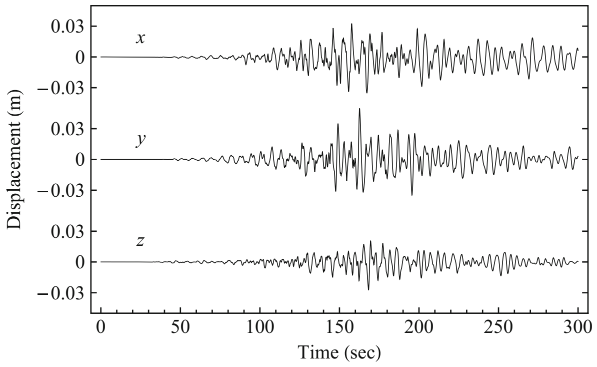


Fig. 15 Input waveform used to excite the model at the bottom of bedrock

the metropolitan areas of Japan. We use the seismic ground motion data from the nearby MeSO-net observation point for the 2011 Tohoku Earthquake to excite the simulation at the base of the bedrock layer.

In order to assure the convergence of strain response, we carry out a preliminary analysis for a part of the target region, using the same material properties and the input wave. From this analysis, we confirm that strain response converges when assuring accuracy between 0.1 and 2.5 Hz. This is attained by generating elements of the nonlinear layers (layers 1 and 2) so that $\chi > 10$, and $\chi > 5$ for the linear layer (bedrock) with time increments of 0.005 s. Here, χ is given by

$$\chi = \frac{V_s}{f_{\max}\, \mathrm{d}s},$$

where V_s is the shear velocity, $f_{\max}$ is the maximum target frequency, and ds is the element size. A band-pass filter is applied to the input waveform to remove components out of the target frequency range (below 0.1 Hz and above 2.5 Hz); the input waveform is shown in Fig. 15.

Figures 16a, b show the 3D structure of the model and illustrate the complicated geometry with the three soil layers. The minimum size of the elements is approximately 4 m. The number of DOFs is 32,509,107, and the model contains 7,779,048 tetrahedral elements and 10,836,369 nodes. We use Cartesian coordinates, with the x, y, and z axes corresponding to east–west, north–south, and up–down directions, respectively, with the southwest lower corner as the origin.

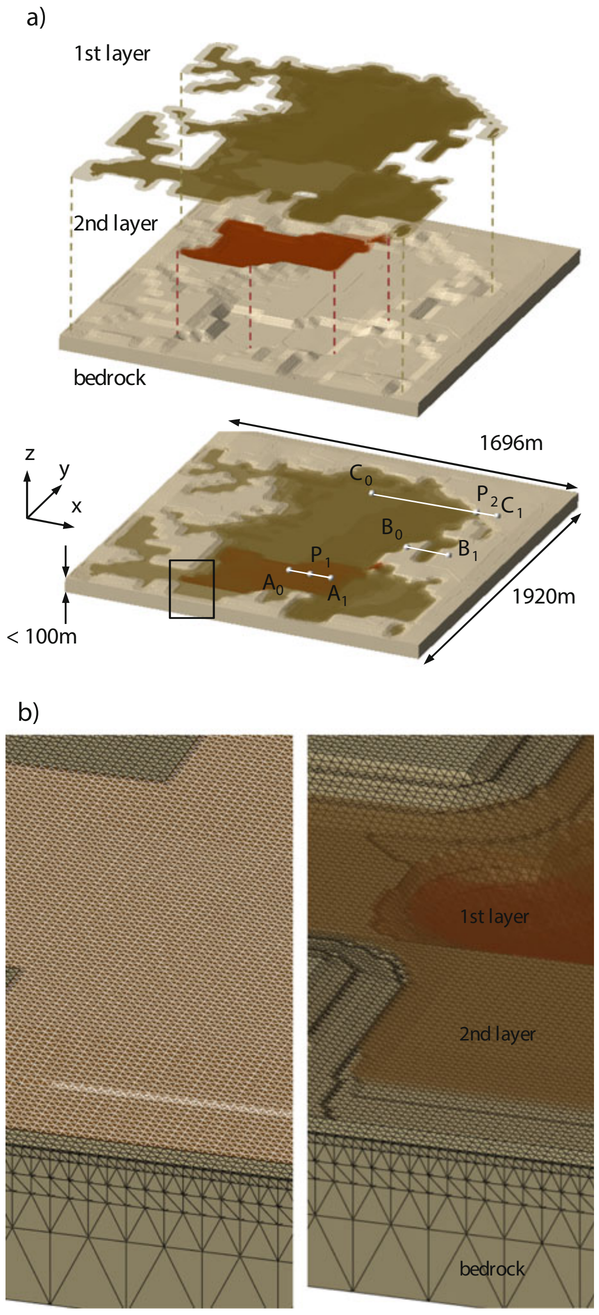


Fig. 16 3D ground structure model, and the position of observation points P_1 and P_2, and lines A, B, and C. (**b**) Close-up view of the region shown by the rectangle in (**a**)

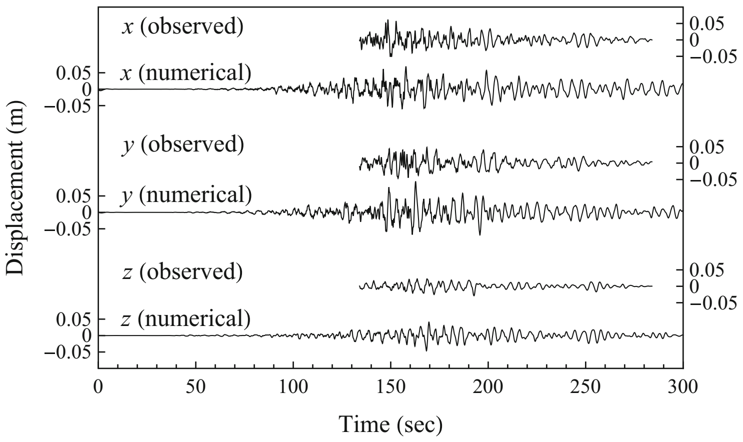


Fig. 17 Comparison of the measured and simulated waveforms at P_1

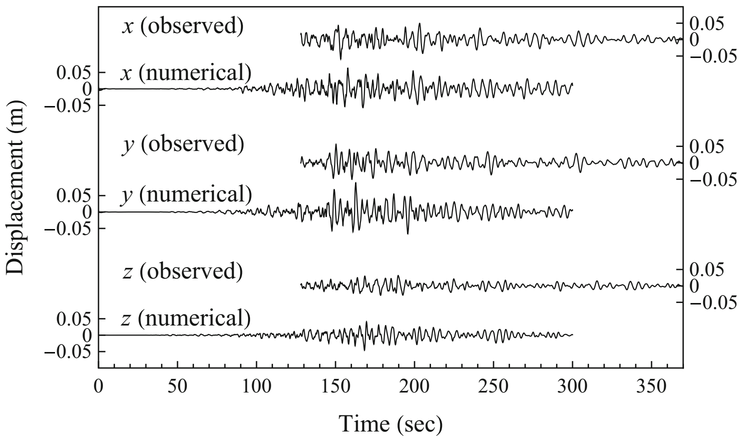


Fig. 18 Comparison of the measured and simulated waveforms at P_2

We simulate the seismic response using 60,000 time steps of 0.005 s. Equation (3) is solved for each time step with a relative error of less than 1.0×10^{-6}. We use a cluster with eight nodes, each with dual hexa-core Intel(R)Xeon X5680 CPUs, connected using InfiniBand quad data rate communication links. Although the cluster is relatively small, the analysis takes 1,107,495 s. On average, each time step is solved in 18.46 s. Thus, large dynamic, 3D, nonlinear FEM is applicable to solve nonlinear soil analysis.

We compare the simulated results at the observation points P_1 and P_2 of SUPREME, as shown in Figs. 17 and 18. The locations of points P_1 and P_2 are shown in Fig. 16a. Since SUPREME is an observation network designed for measuring the maximum acceleration for disaster estimation purposes, the two data sets are not synchronized, and the duration of the recorded data is limited. Because the duration of 2011 Tohoku Earthquake is longer than the time window of the data acquisition system, the first half of the wave is missing. Nevertheless, we find that the latter half of the displacement records is reproduced fairly well when the frequency components below 0.1 Hz are removed. Furthermore, we compare the simulated results with the complete records from a nearby observation point and find good agreement.

Although detailed comparison is difficult because of the limitation of the observed data, we can see that the latter half of the simulated results (140–240 s) matches well with that of the observed seismic ground motion. Here, we describe a quantitative comparison of this part of the data. We first compare the spectrum intensity, SI, which is defined as

$$SI = \frac{1}{2.4}\int_{0.1}^{2.5} S_v(T)\,dT,$$

where S_v is the velocity response spectrum using a damping factor of 20 %, and T is the natural period in seconds. A comparison of the SI is given in Table 8. It is seen that the horizontal components at P_1 match well. However, the numerical and measured SI values differ by 50 % for some of the other components. Because the SI is based on the maximum response of a one-DOF mass-spring model, it may be difficult to quantify the similarities in the waveforms using this single value. In addition, nonlinear dynamic analysis is essential for the improvement of seismic resistance in structural engineering, and thus a quantitative comparison of the history waveform becomes important.

Thus, in addition to an index of SI, we use an index with which we can compare the phase and envelope of the waveform. Here, we use the time-frequency misfit (described in [44]) and an example of Goodness-Of-Fit (GOF) criteria (described in [43, 45]). The goodness of fit is categorized as "poor" for GOF index values under 4, "fair" for values 4–6, "good" for values 6–8, and "excellent" for values more than 8 [43, 45]. It is reported that one horizontal direction of an actual seismogram typically fits the other horizontal component in the "good" range [43]; from Table 9, the waves in the z-axis have a fit comparable to this, while the horizontal components have a slightly weaker fit ("fair" to "good" fit). We conclude that the simulated results and the measured data match well, considering the accuracy of the measured waveform

Table 8 Spectrum intensity of the measured and simulated waveforms

Point/component	Observed (cm/s)	Numerical (cm/s)
P_1, x	25.6	28.2
P_1, y	27.1	25.5
P_1, z	6.25	9.35
P_2, x	9.92	15.9
P_2, y	10.7	13.2
P_2, z	6.53	9.28

Table 9 Agreement between the measured and simulated waveforms using time-frequency misfit and goodness-of-fit criteria [44, 45]

	Misfit		Goodness-of-fit (values)		Goodness-of-fit (category)	
Point/Component	Envelope	Phase	Envelope	Phase	Envelope	Phase
P_1, x	0.652	0.523	5.21	4.77	Fair	Fair
P_1, y	0.776	0.470	4.60	5.30	Fair	Fair
P_1, z	0.401	0.286	6.70	7.14	Good	Good
P_2, x	0.494	0.527	6.10	4.73	Good	Fair
P_2, y	0.576	0.530	5.62	4.70	Fair	Fair
P_2, z	0.370	0.346	6.91	6.54	Good	Good

and that no tuning was carried out on the input waveform, soil parameters, or soil model. It would be possible to improve the accuracy of the model by using data from dense observation networks and tuning the parameters to achieve better agreement with the available data.

We evaluate the distribution of seismic ground motion on the surface, in order to determine the forces that act on the pipelines; pipelines are buried near the surface and move with the surrounding soil. Figure 19 shows snapshots of the magnitude of the displacement at a number of time steps, and Fig. 21a the maxima. It is seen that the maximum displacement is highly correlated with the depth of layer 1, as shown in Fig. 22a. According to one-dimensional multiple reflection theory, the natural frequency of soil, estimated by $V_s/4D$ with V_s and D being the shear wave velocity and the layer depth, is 1.2 Hz for a layer depth of 21 m (i.e., where the displacement is amplified) and 2.5 Hz for a layer depth of 10 m (i.e., where the displacement is not amplified). The major frequency components of the input wave are around 1 Hz, which is consistent with the above considerations.

Snapshots of the maximum principal strain are shown in Fig. 20, together with the maxima during the whole time-history in Fig. 21b. The snapshots indicate that large strain occurs between the soft and hard layers at 151.3 s, which then propagate towards the inner side of the soft layer during 151.3–151.7 s, leading to a complex distribution of the maximum strain, as shown in Fig. 21b. Contrary to the expectation that a maximum strain distribution will correlate with the rate of change in the depth of layer 1 (shown in Fig. 22b), which is based on the assumption that the ground motion varies where the ground structure changes rapidly, these two distributions are not strongly correlated. To account for the amplification of the wave by layer 1, we compare the distribution of strain with a more complex index that considers the rate of change in layer 1 and the depth of the layer, which is shown in Fig. 22c. This index is the product of the depth of layer 1 and the rate of spatial change of layer 1. The maximum principal strain distribution is more strongly correlated with this index than with the rate of change in layer-1 depth shown in Fig. 22b. From these comparisons, we can see that it is not straightforward to relate the distribution of strain to the ground structure, and thus 3D numerical simulations of ground motion are effective for determining the distribution of strain during earthquakes.

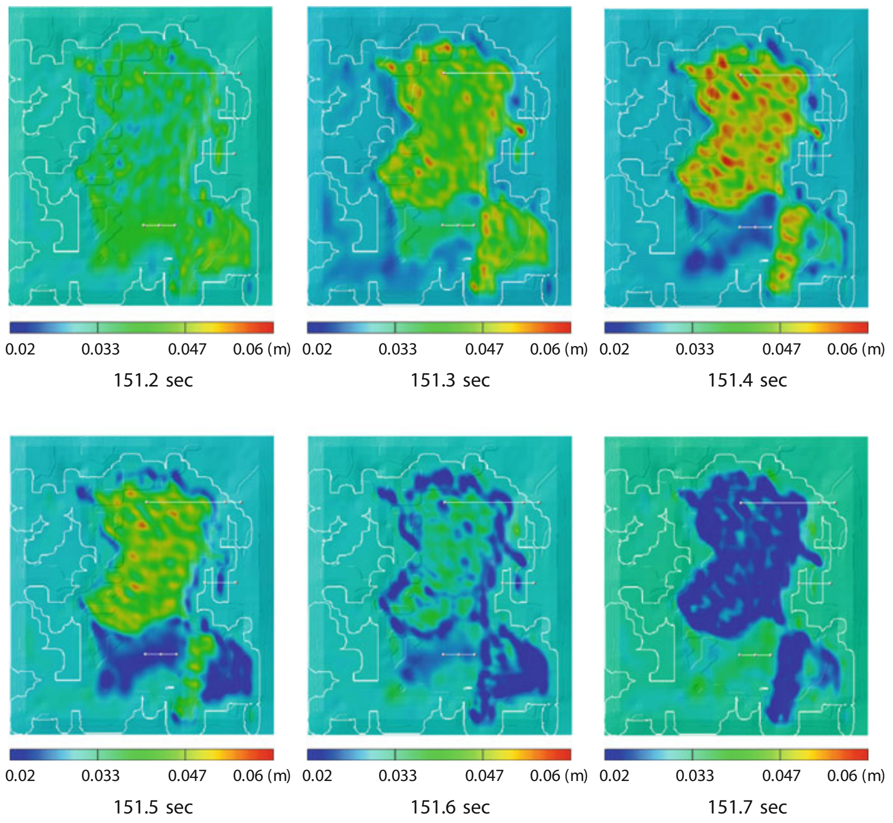


Fig. 19 Distribution history of the norm of displacement. The *white lines* indicate lines A, B, C, and the boundary between the bedrock and layer 1

To evaluate the effects of ground strain on buried pipelines, we set observation lines in the model, and compute the axial strain along the lines. We use three lines: Line A is from (871.1 m, 515.6 m) to (1071.1 m, 515.6 m), Line B is from (1250 m, 975 m) to (1450 m, 975 m), and Line C is from (880.2 m, 1487.9 m) to (1480.2 m, 1487.9 m); see Fig. 16. Line A goes through observation point P_1 where the depth of layer 1 is constant; Line B corresponds to a V-shaped depth of layer 1, which is similar to that reported in [46]; and Line C goes through observation point P_2. Layer 1 has a monotonically increasing depth, followed by a region with constant depth. Figure 23 shows the maximum axial strain and the soil structure under each line. Line A does not have a rapidly changing soil structure, and the displacement is amplified but the maximum principal strain is small and relatively uniform. Although the displacement of the pipe is large, the axial strain is small. The axial strain along line B is similar to the observed strain of buried pipelines at V-shaped surface soil layers reported by Tsukamoto et al. [46]. The displacement in the bedrock is small but becomes large for the soft layer and is nearly constant in

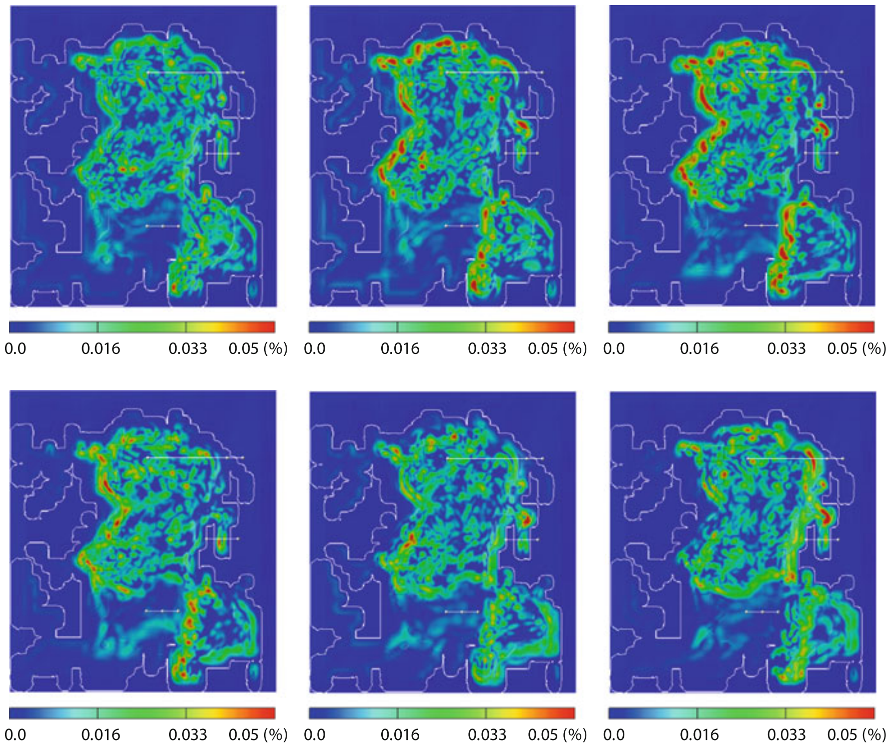


Fig. 20 Distribution history of the maximum principal strain. The *white lines* indicate lines A, B, C, and the boundary between the bedrock and layer 1

the center of the V-shaped valley. This leads to small axial strain in the center of the V-shaped valley and large strain near the edges of the soft soil layer. This phenomenon is reported by Liang and Sun [33] and Tsukamoto et al. [46] to be the cause of pipeline damage. The response of line C is more complex than those of lines A and B. Similar to line B, large axial strain occurs near the boundaries of layer 1; however, we observe a complex distribution of axial strain from $x = 900$ m to $x = 1200$ m, where the ground structure is similar to that of line A, having a horizontally layered structure. This is because the soil structure around line C is complex and so the two-dimensional structure shown in Fig. 23c does not accurately describe the topography of the underlying bedrock, which is presented in Fig. 22a. From these observations, we can conclude that non-uniform soil structure may give rise to complex ground motion, suggesting the advantages of three-dimensional analysis.

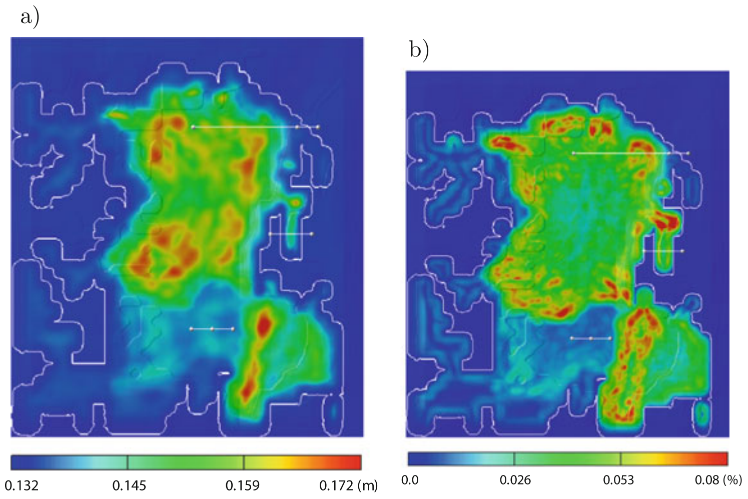


Fig. 21 Distribution history of the maximum norm of displacement and the maximum principal strain. The *white lines* indicate lines A, B, C, and the boundary between the bedrock and layer 1. (**a**) Norm of displacement. (**b**) Maximum principle strain

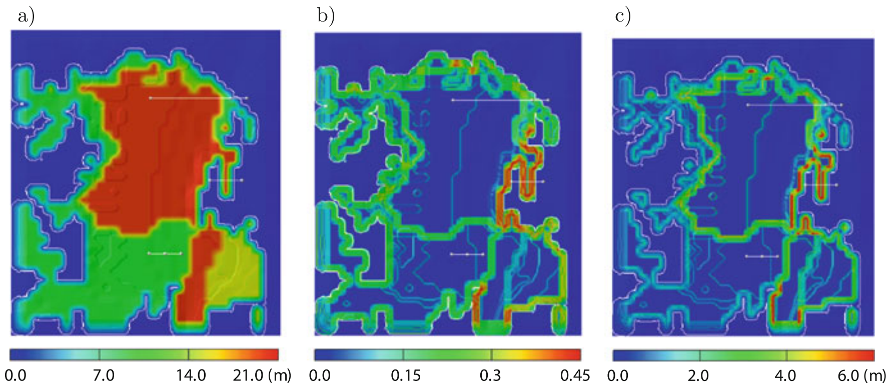


Fig. 22 Geometric properties of layer 1. The *white lines* indicate lines A, B, C, and the boundary between the bedrock and layer 1. (**a**) h: depth of layer 1. (**b**) $|\nabla h|$: rate of spatial change in depth of layer 1. (**c**) $h|\nabla h|$: product of depth of layer 1 (h) and rate of spatial change in depth of layer 1 ($|\nabla h|$)

2.3 Concluding Remarks

We have developed a large nonlinear seismic ground response simulation and analyzed a site with non-uniform soil structure using 3-D nonlinear dynamic FEM. In regions with non-uniform soil structure, body waves are locally amplified and strain is amplified accordingly. Amplification of the velocity or acceleration has been extensively investigated because it can be measured on site. However, an

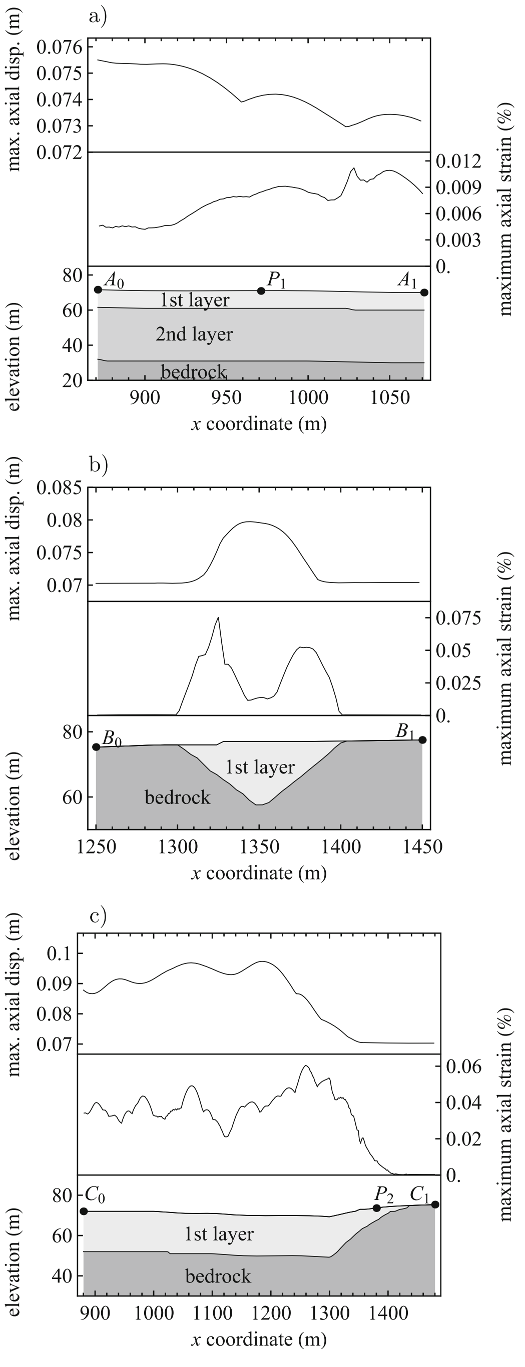


Fig. 23 Maximum axial displacement, maximum axial strain, and the underground structure of lines A, B, and C. (**a**) Line A. (**b**) Line B. (**c**) Line C

analysis of strain requires a numerical analysis, which is difficult using conventional techniques. The strain concentration, including the amplitude, and temporal and spatial variations are obtained using 3D nonlinear FEM analysis, and we are able to reproduce the observed ground motion. It is difficult to describe such complex strain behavior using simplified methods, such as seismic design based on surface waves. Numerical methods such as presented here are thus expected to aid in improving the seismic resistance of structures. The method might be further improved by additional comparisons with available measured data to optimize the parameters used in the calculation.

References

1. Ichimura, T., Hori, M.: Structural seismic response analysis based on multiscale approach of computing fault-structure system. Earthq. Eng. Struct. Dyn. **38**, 439–455 (2009)
2. Koketsu, K., Fujiwara, H., Ikegami, Y.: Finite-element simulation of seismic ground motion with a voxel mesh. Pure Appl. Geophys. **161**, 2463–2478 (2004)
3. Komatitsch, D., Liu, Q., Tromp, J., Suss, P., Stidham, C., Shaw, J.H.: Simulations of ground motion in the Los Angeles Basin based upon the spectral element method. Bull. Seismol. Soc. Am. **94**, 187–206 (2004)
4. Bielak, J., Ghattas, O., Kim, E.J.: Parallel octree-based finite element method for large-scale earthquake ground motion simulation. Comput. Model. Eng. Sci. **10**, 99–112 (2005)
5. Furumura, T.: Large-scale parallel simulation of seismic wave propagation and strong ground motions for the past and future earthquakes in Japan. J. Earth Simul. **3**, 29–38 (2005)
6. Kaeser, M., Dumbser, M.: An arbitrary high-order discontinuous Galerkin method for elastic waves on unstructured meshes – I. The two-dimensional isotropic case with external source terms. Geophys. J. Int. **166**, 855–877 (2006)
7. Ma, S., Liu, P.: Modeling of the perfectly matched layer absorbing boundaries and intrinsic attenuation in explicit finite-element methods. Bull. Seismol. Soc. Am. **96**, 1779–1794 (2006)
8. Ichimura, T., Hori, M., Kuwamoto, H.: Earthquake motion simulation with multi-scale finite element analysis on hybrid grid. Bull. Seismol. Soc. Am. **97**, 1133–1143 (2007)
9. Ichimura, T., Hori, M., Bielak, J.: A hybrid multiresolution meshing technique for finite element three-dimensional earthquake ground motion modeling in basins including topography. Geophys. J. Int. **177**, 1221–1232 (2009)
10. Ogino, M., Shioya, R., Kawai, H., Yoshimura, S.: Seismic response analysis of nuclear pressure vessel model with ADVENTURE system on the earth simulator. J. Earth Simul. **2**, 41–54 (2005)
11. Elgamal, A., Lu, J., He, L., Law, K., Yang, Z.: Techniques for simulation of large-scale nonlinear soil-structure systems. In: International Workshop on Constitutive Modelling, Development, Implementation, Evaluation, and Application (2007)
12. Dobashi, H., Ochiai, E., Ichimura, T., Yamaki, Y., Hori, M., Yamada, T., Ohbo, N., Moriguchi, M.: 3D FE analysis for seismic response of complicated large-scale ramp tunnel structure. In: ECCOMAS Thematic Conference on Computational Methods in Structural Dynamics and Earthquake Engineering (2007)
13. Dobashi, H., Ichimura, T., Ohbo, N., Hori, M., Yamada, T.: Study on necessity of large-scale three-dimensional analysis for evaluating seismic response of large tunnel having complex structure. J. Struct. Mech. Earthq. Eng.; Jpn. Soc. Civ. Eng. **64**, 639–652 (2008) (in Japanese with English abstract)
14. Ohsaki, M., Miyamura, T., Kohiyama, M., Hori, M., Noguchi, H., Akiba, H., Kajiwara, K., Ine, T.: High-precision finite element analysis of elastoplastic dynamic responses of super-high-rise steel frames. Earthq. Eng. Struct. Dyn. **38**, 635–654 (2009)

15. Lysmer, J., Kuhlemeyer, R.L.: Finite dynamic model for infinite media. Proc. Am. Soc. Civ. Eng.; J. Eng. Mech. Div. **95**, 859–877 (1969)
16. Cerjan, C., Kosloff, D., Kosloff, R., Reshef, M.: A nonreflecting boundary condition for discrete acoustic and elastic wave equations. Geophysics **50**, 705–708 (1985)
17. Hisada, Y.: An efficient method for computing Green's functions for a layered half-space with sources and receivers at close depths. Bull. Seismol. Soc. Am. **84**, 1456–1472 (1994)
18. Hatsuku, T., Dobashi, H., Ohbo, N., Moriguchi, M., Yamada, T., Itami, H., Ichimura, T., Hori, M.: Three-dimensional seismic response of large-scale center type ramp tunnel structures: evaluation of seismic response for various earthquakes. In: Proceedings of the 43rd Annual Japanese Society of Geotechnical Engineering conference (2008) (in Japanese)
19. Central Disaster Prevention Council: Working Report on Near-field Earthquake Beneath Central Tokyo. http://www.bousai.go.jp/jishin/chubou/shutochokka/12/shiryo2-1.pdf; http://www.bousai.go.jp/jishin/chubou/shutochokka/12/shiryo2-2.pdf (2004) (in Japanese)
20. Tanaka, Y., Miyake, H., Koketsu, K., Furumura, T., Hayakawa, T., Baba, T., Suzuki, H., Masuda, T.: The DaiDaiToku Integrated Model of the Velocity Structure Beneath the Tokyo Metropolitan Area, S116–P014 (2006)
21. Yamanaka, H., Furumura, T., Sato, H., Higashi, S., Shiba, Y., Sato, T., Hayakawa, T.: Regional characterization of the crust in metropolitan areas for prediction of strong ground motion/3D velocity and Q model. Working Report "Special Project for Earthquake Disaster Mitigation in Urban Areas", pp. 486–514 (2006) (in Japanese)
22. Nakamura, H., Miyatake, T.: An approximate expression of slip velocity time functions for simulation of near-field strong ground motion. Zisin **53**, 1–9 (2000) (in Japanese)
23. K-net and SK-net Broadband Seismograph Network: Japan National Research Institute for Earth Science and Disaster Prevention and Earthquake Research Institute, The University of Tokyo. http://www.k-net.bosai.go.jp; http://www.sknet.eri.u-tokyo.ac.jp/ (2014)
24. F-net Broadband Seismograph Network: Japan National Research Institute for Earth Science and Disaster Prevention. http://www.fnet.bosai.go.jp/freesia/ (2014)
25. Kikuchi, M., Ishida, M.: Source retrieval for deep local earthquake with broadband records. Bull. Seismol. Soc. Am. **83**, 1855–1870 (1993)
26. Ma, S., Archuleta, R.J., Page, M.T.: Effects of large-scale surface topography on ground motions, as demonstrated by a study of the San Gabriel Mountains, Los Angeles, California. Bull. Seismol. Soc. Am. **97**, 2066–2079 (2007)
27. Lee, S.-J., Komatitsch, D., Huang, B.-S., Tromp, J.: Effects of topography on seismic wave propagation: an example of Northern Taiwan. Bull. Seismol. Soc. Am. **99**, 314–325 (2009)
28. Romanel, C., Kundu, T.: A hybrid modelling of soil-structure interaction problems for deeply embedded structures in a multilayered medium. Earthq. Eng. Struct. Dyn. **22**, 557–571 (1993)
29. Moore, I.D., Guan, F.: Three-dimensional dynamic response of lined tunnels due to incident seismic waves. Earthq. Eng. Struct. Dyn. **25**, 357–369 (1996)
30. Stamos, A.A., Beskos, D.E.: Dynamic analysis of large 3-D underground structures by the BEM. Earthq. Eng. Struct. Dyn. **24**, 917–934 (1995)
31. Gil, L.M., Hernández, E., De La Fuente, P.: Simplified transverse seismic analysis of buried structures. Soil Dyn. Earthq. Eng. **21**, 735–740 (2001)
32. Park, D., Sagong, M., Kwak, D.-Y., Jeong, C.-G.: Simulation of tunnel response under spatially varying ground motion. Soil Dyn. Earthq. Eng. **29**, 1417–1424 (2009)
33. Liang, J., Sun, S.: Site effects on seismic behavior of pipelines: a review. J. Press. Vessel Technol. **122**, 469–475 (2000)
34. Advisory Committee for Natural Resources and Energy, Urban area thermal energy committee, Gas safety subcommittee, Working group for earthquake disaster prevention, 2012. Report on Disaster Mitigation for Gas Supply in View of Great East Japan Earthquake (2013) (in Japanese)
35. Nishio, N., Hamura, A., Sase, T.: Observation of pipeline behavior at geographically complex site during earthquake. In: 9th World Conference on Earthquake Engineering, Tokyo-Kyoto, vol. 7, pp. 24–28 (1988)

36. Trifunac, M.D., Todorovska, M.I.: Northridge, California, Earthquake of 1994: density of pipe breaks and surface strains. Soil Dyn. Earthq. Eng. **16**, 193–207 (1997)
37. Idriss, I.M., Singh, R.D., Dobry, R.: Nonlinear behavior of soft clays during cyclic loading. J. Geotech. Eng. Div. **104**, 1427–1447 (1978)
38. Masing, G.: Eigenspannungen und Verfestigung beim Messing. In: Proceedings of the 2nd International Congress of Applied Mechanics, pp. 332–335 (1926) [in German]
39. Ichimura, T., Hori, M., Bielak, J.: A hybrid multiresolution meshing technique for finite element three-dimensional earthquake ground motion modeling in basins including topography. Geophys. J. Int. **177**, 1221–1232 (2009)
40. Shimizu, Y., Watanabe, A., Koganemaru, K., Nakayama, W., Yamazaki, F.: Super high-density realtime disaster mitigation system. In: 12th World Conference on Earthquake Engineering, New Zealand (2000)
41. The Administration-Geographic Information System Database of Yokohama City (in Japanese). http://wwwm.city.yokohama.lg.jp/ (2014)
42. Working group of central disaster management council in Japan for Tokai earthquake (in Japanese). http://www.bousai.go.jp/jishin/tokai/index.html (2014)
43. Anderson, J.G.: Quantitative measure of the goodness-of-fit of synthetic seismograms. In: 13th World Conference on Earthquake Engineering Conference Proceedings, Vancouver, Paper 243 (2004)
44. Kristekova, M., Kristek, J., Moczo, P., Day, S.: Misfit criteria for quantitative comparison of seismograms. Bull. Seismol. Soc. Am. **96**(5), 1836–1850 (2006)
45. Kristekova, M., Kristek, J., Moczo, P.: Time-frequency misfit and goodness-of-fit criteria. Geophys. J. Int. **178**, 813–825 (2009)
46. Tsukamoto, K., Nishio, N., Satake, M., Asano, T.: Observation of pipeline behavior at geographically complex site during earthquake. In: 8th World Conference on Earthquake Engineering, San Francisco, vol. 7, pp. 247–254 (1984)

Seismic Response Simulation of Building Structures

Makoto Ohsaki, Tomoshi Miyamura, Masayuki Kohiyama, Takuzo Yamashita, and Hiroshi Akiba

Abstract In this chapter, we present an overview of the E-Simulator for application to seismic response analysis of building structures. Accuracy and computational performance of simulation using the E-Simulator are discussed through examples of seismic response analysis of a four-story steel frame, buckling analysis of a column, and simulations of static cyclic responses of a composite beam and an exterior wall. Results of seismic response analysis with fixed base as well as soil-structure interaction analysis are presented for a high-rise building frame. Computational performance using the K computer is also discussed.

M. Ohsaki (✉)
Department of Architecture and Architectural Engineering, Kyoto University, Kyoto-Daigaku Katsura, Nishikyo, Kyoto 615-8540, Japan
e-mail: ohsaki@archi.kyoto-u.ac.jp

T. Miyamura
Department of Computer Science, College of Engineering, Nihon University, 1 Nakagawara, Tokusada, Tamuramachi, Koriyama 963-8642, Japan
e-mail: miyamura@cs.ce.nihon-u.ac.jp

M. Kohiyama
Department of System Design Engineering, Keio University, 3-14-1 Hiyoshi, Kohoku, Yokohama 223-8522, Japan
e-mail: kohiyama@sd.keio.ac.jp

T. Yamashita
Hyogo Earthquake Engineering Research Center, National Research Institute for Earth Science and Disaster Prevention, 1501-21 Nishikameya, Mitsuda, Shijimi, Miki 673-0515, Japan
e-mail: tyamashi@bosai.go.jp

H. Akiba
Earthquake Research Institute, The University of Tokyo, 1-1-1 Yayoi, Bunkyo-ku, Tokyo 113-0032, Japan
e-mail: akiba@eri.u-tokyo.ac.jp

S. Yoshimura et al. (eds.), *High-Performance Computing for Structural Mechanics and Earthquake/Tsunami Engineering*, Springer Tracts in Mechanical Engineering,
DOI 10.1007/978-3-319-21048-3_4

1 Overview of E-Simulator for Building Structures

A software package called E-Simulator is under development at the Hyogo Earthquake Engineering Research Center (E-Defense) of the National Research Institute for Earth Science and Disaster Prevention (NIED), Japan [1, 2]. E-Defense has the world's largest shaking table to carry out real-scale shake-table test of building and civil structures under seismic motions. One of the purposes of shake-table tests using the E-Defense is to validate simulation codes including the E-Simulator developed in NIED.

The E-Simulator is a parallel finite element (FE) analysis software package for precise simulation of collapse behaviors of building and civil structures under earthquake motions [3]. In conventional analysis methods for building frames, empirically defined macro models such as plastic hinges and hysteresis models are often used [4, 5]. The simulation results using such macro models strongly depend on the assumptions incorporated in the models, and model parameters should be identified appropriately from experimental results of the structural components.

On the other hand, using the E-Simulator, we can carry out large-scale analysis with a very fine mesh of solid elements after identification of the material properties only from a simple material test. Although experiments of structural components are needed for validation of the analysis code and structural model, global and local responses of building frames under seismic excitation can be simulated without resort to any macro model, and the cost for experiments for design of complex structures and development of new structural components can be drastically reduced by utilizing detailed FE-analysis.

The main framework of the E-Simulator is the commercial software package called ADVENTURECluster [6, 7], which utilizes the Coarse Grid Conjugate Gradient (CGCG) method developed by Akiba et al. [8, 9]. The ADVENTURECluster/E-Simulator has basic functions that are necessary for a general-purpose FE-analysis code. For implicit dynamic analysis, the Hilber–Hughes–Taylor time integration method (α-method) [10] is implemented. Various finite elements including those with incompatible modes and selective/reduced integration are prepared together with the standard constitutive models. In the E-Simulator, elastoplastic constitutive equations and ductile fracture models that are particular to building structures are implemented as extended functions to the ADVENTURECluster.

In this chapter, we present overview and computational results of the E-Simulator for seismic response simulation of building structures.

2 Four-Story Steel Frame

2.1 FE-Model of Four-Story Steel Frame

Elastoplastic dynamic analysis is carried out using the E-Simulator to validate the analysis code by comparing its results [11–13] with full-scale shake-table test results of a four-story steel frame [14, 15], as shown in Fig. 1, under the JR Takatori record of the 1995 Hyogoken–Nanbu earthquake with various scales.

The FE-mesh generated in the Noguchi Laboratory at Keio University, Japan, is used as a prototype of the mesh, which is modified, as shown in Fig. 2, using the 3D-CAD software called I-deas [16]. The details of member sections and mass distribution are described in [14, 15]. The FE-mesh has 4,532,742 hexahedral elements, 6,330,752 nodes, and 18,992,256 degrees of freedom (DOFs). The plates used for the square tube columns as well as the flanges and webs of beams are divided into at least two layers of solid elements as shown in Fig. 2b. The size of each element in the longitudinal direction of a beam/column member is approximately 13 mm near the connections, where severe plastic deformation is expected, while a coarser mesh is used in the region far from the connections. See [11–13] for details.

Mesh division in the lateral (width) directions of a column is not uniform, because the column is connected through a diaphragm, which is divided into at least two layers, to a beam flange by sharing the nodes without using multi-point constraints (MPCs). The numbers of meshes in the longitudinal, lateral, and width directions are determined after verification through buckling analysis of the C2 column in Fig. 1. Although mesh division in the lateral direction is coarser than that of the reference mesh (Model *t*8*b*48*L*320) defined in Sect. 3, division in the longitudinal direction is fine enough to reproduce the local buckling at column ends. See Sect. 3 for details of the verification of column buckling analysis. The minimum

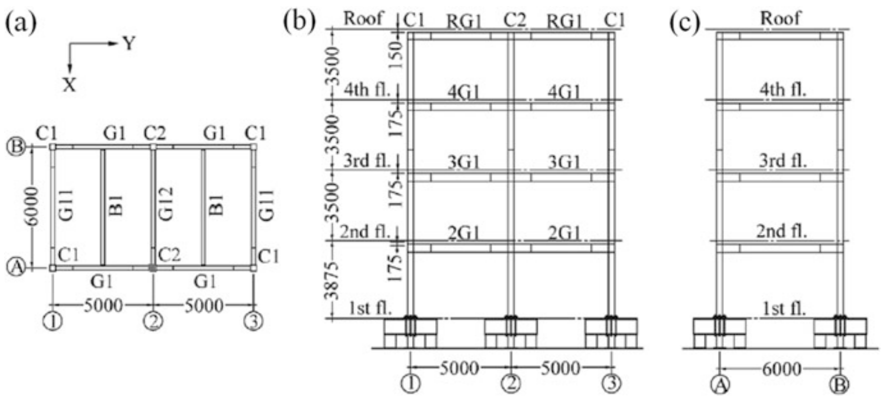


Fig. 1 Four-story steel frame model; (**a**) plan, (**b**) Y-elevation, (**c**) X-elevation [14, 15]

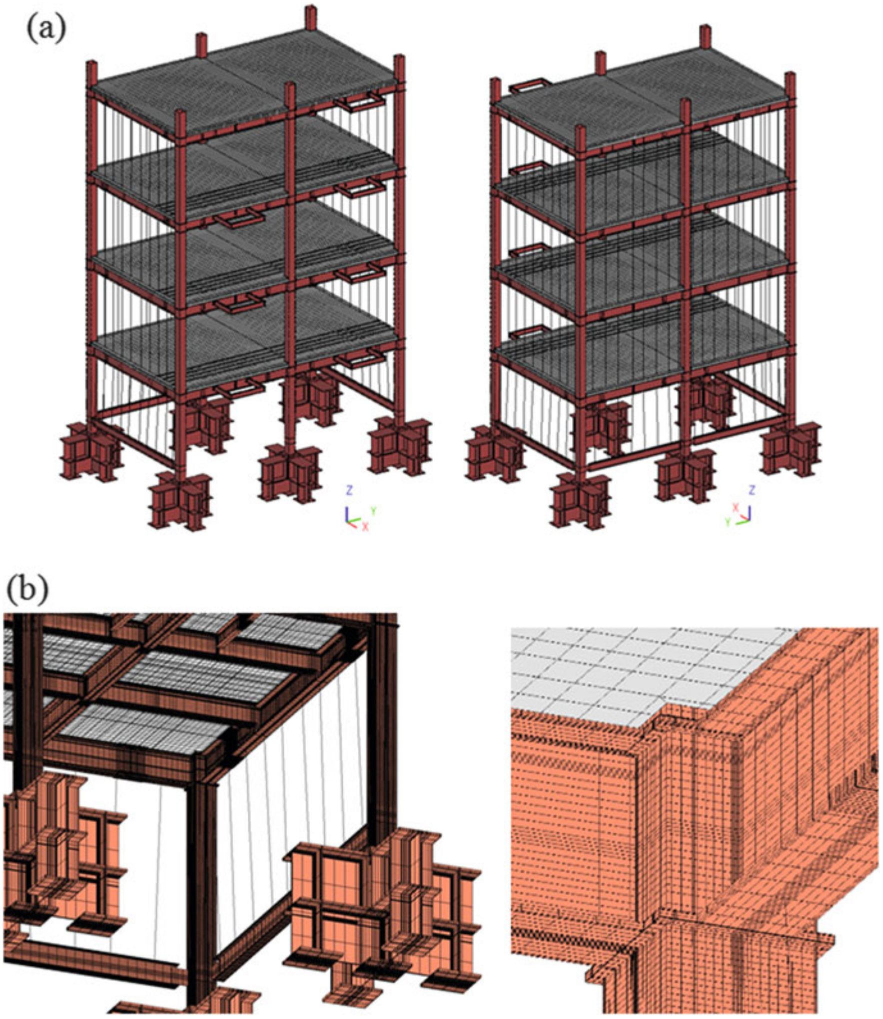


Fig. 2 FE-mesh of four-story steel frame; (**a**) entire structure, (**b**) close-up view [12]

value of aspect ratio and the maximum value of edge length among all elements of the model are 0.0204 and 147.0 mm, respectively.

The floor slab is also divided into solid elements with two layers. We consider two cases for the contact condition between slab and column; (a) stick condition, where the surfaces of slab and column are always in contact without slip, and (b) no-contact condition, where the surfaces separate and penetrate with each other. The slab is connected to beam flanges with the gap separated using studs, which are modeled as rigid beams as will be described in Sect. 5. The number of rigid beam elements for modeling the studs is 2196. Steel reinforcement bars (rebars) of

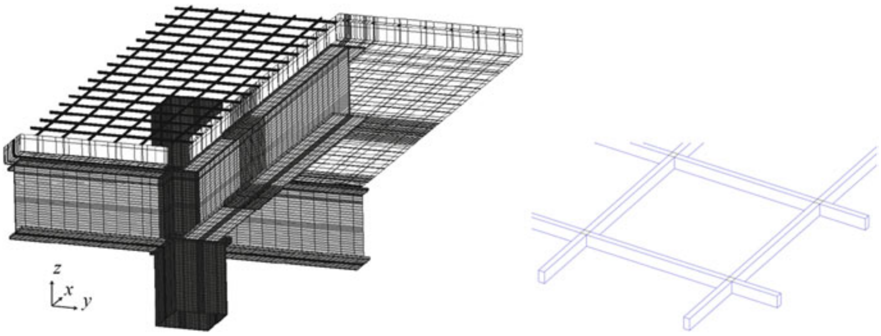


Fig. 3 Model of beam, column, slab, and steel bars in the slab [12]

diameter 6 mm in the slab are also modeled by solid elements as shown in Fig. 3. The circular cross section of a rebar is approximated to a square section with different area, and the axial stiffness is corrected by adjusting Young's modulus.

H-shaped steel members of the column bases are modeled using solid elements. The anchor bolts to connect the column to the base with the initial tension 100 kN and the seamless pipes around the bolts are modeled by truss elements. Stress concentration at the end of bolt is avoided by connecting each end of the truss element to the surface of base using many rigid beams. The lower end of column is stiffened by a base plate, and a layer of non-shrink mortar is inserted between the base plate and the upper face of base member. Stick conditions are assumed among the base plate, the non-shrink mortar, and the base member; i.e., the nodes on the surfaces are shared by the elements with different material properties.

The exterior walls made of autoclaved lightweight aerated concrete (ALC) panels are modeled by tri-linear elastoplastic shear springs, which are introduced to incorporate the effect of hysteretic damping due to the plastic and frictional energy dissipation in the exterior wall. The material parameters are determined from the experimental results in [17]. Properties of ALC panels under cyclic deformation are investigated in detail in Sect. 6.

2.2 Constitutive Models

A continuum constitutive model of steel material for large-scale FE-analysis has been implemented in the E-Simulator. In the field of mechanical engineering and material science, various constitutive models, including Armstrong–Frederic model, multi-layer model, and bounding surface model, have been developed for simulating cyclic elastoplastic behavior of various types of steel materials [18]. By contrast, in the field of civil engineering, rolled mild steel materials are widely used. One of the distinct properties of these materials is the existence of a sharp yield plateau and Bauschinger effect, which lead to different characteristics between the first and subsequent loading cycles. Several models based on explicit differential evolution equations have been developed for simulation of such behaviors [19]. However, it is very difficult to derive a constitutive model that can be stably used for large-scale analysis.

In contrast to complex evolution rules, linear isotropic–kinematic hardening model has a very stable algorithm using consistent tangent [20]. Therefore, we add implicit rules to the piecewise linear isotropic–kinematic hardening model for steel material to simulate the yield plateau and Bauschinger effect [21, 22]. Different rules in algorithmic and implicit manner are implemented in the first and subsequent loading states. The material parameters, including hardening coefficients and ratios between isotropic and kinematic parts of hardening, are determined using an optimization algorithm [21] to fit the stress–strain curve of the cyclic uniaxial coupon test [23, 24].

Let $\bar{e}^{\mathrm{p}}$ denote the equivalent plastic strain. The hardening coefficient $H'(\bar{e}^{\mathrm{p}})$, which is a piecewise constant function of $\bar{e}^{\mathrm{p}}$, is divided into the components of kinematic hardening $H'_{\mathrm{K}}(\bar{e}^{\mathrm{p}})$ and isotropic hardening $H'_{\mathrm{I}}(\bar{e}^{\mathrm{p}})$ as

$$H'(\bar{e}^{\mathrm{p}}) = c_{\mathrm{K}} H'(\bar{e}^{\mathrm{p}}) + c_{\mathrm{I}} H'(\bar{e}^{\mathrm{p}}) = H'_{\mathrm{K}}(\bar{e}^{\mathrm{p}}) + H'_{\mathrm{I}}(\bar{e}^{\mathrm{p}})\, Z \tag{1}$$

where c_{K} and c_{I} are the kinematic and isotropic hardening ratios satisfying $c_{\mathrm{K}} + c_{\mathrm{I}} = 1$.

To simulate a wide yield plateau in the first loading (initial yielding) as shown in Fig. 4, we use several piecewise linear hardening curves. Furthermore, we add a small region with an artificial negative value of c_{I} with positive $H'(\bar{e}^{\mathrm{p}})$ just below the initial yielding; thus, the yield surface shrinks to simulate the Bauschinger effect in a similar manner as the bounding surface method.

The stress–strain curve of the first loading is called Curve 1. Since hardening property under cyclic deformation depends on the maximum experienced value $\widehat{\sigma}^{\max}$

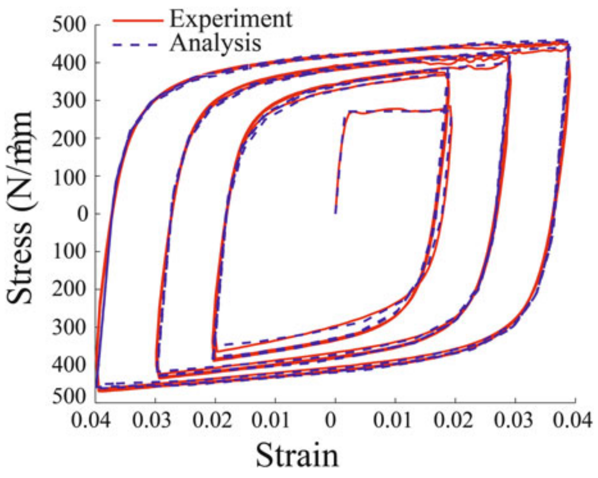


Fig. 4 Stress–strain relation of SS400 material (Japanese specification) under cyclic loading [21–23]

of the equivalent stress $\widehat{\sigma}$, the stress–strain relation after second loading is classified into two curves; i.e., Curve 2 for $\widehat{\sigma} < \widehat{\sigma}^{\max}$ and Curve 3 for $\widehat{\sigma} \geq \widehat{\sigma}^{\max}$.

For analysis of the four-story frame, the parameters of constitutive model are determined from the results of the uniaxial tests under monotonic loading distributed for the blind analysis contest [24], and from those of the uniaxial test under cyclic loading conducted by Yamada et al. [23]. We can use some appropriate optimization methods for parameter identification; e.g., random selection with data mining techniques as presented by Ohsaki et al. [21]. Note that different stress–strain curves are used for members in different groups, and the common parameters are Young's modulus $E = 205.0$ kN/mm^2, Poisson's ratio $\nu = 0.3$, and mass density $\rho = 7.86 \times 10^{-6}$ kg/mm^3.

The extended Drucker–Prager yield criterion with associated flow rule is used for concrete material of slab. The yield condition is given using the yield function F as

$$F = \sqrt{l_0{}^2 + \widehat{\sigma}^2} - p \tan\beta - d' = 0 \tag{2}$$

where p is the hydrostatic stress, β is the internal friction angle, and l_0 and d' are the parameters to determine the shape of yield surface. The parameter values are$\beta = 67.14°$, $l_0 = 3.0$ N/mm^2, $d' = 5.433$ N/mm^2, the Young's modulus E is 25.61 kN/mm^2, the Poisson's ratio is 0.2, the compressive strength is 25.1 N/mm^2, the tensile strength is 2.18 N/mm^2, and the hardening coefficient is $E/1000$.

The self-weight of steel is computed from its mass density. By contrast, the mass density 2.3×10^{-6} kg/mm^3 of the slab is increased to 6.26×10^{-6} kg/mm^3 to appropriately include the mass of all nonstructural components installed in the experimental model.

2.3 Eigenvalue Analysis and Seismic Response Analysis Under 60 % Takatori Record

Eigenvalue analysis is carried out to check the quality of the FE-mesh and to determine the coefficients for Rayleigh damping. The five lowest natural periods obtained by eigenvalue analysis are 0.8262, 0.8081, 0.5418, 0.2660, and 0.2611 s. The 1st natural period identified from experimental results is 0.82 s [25], which agrees with the numerical result with good accuracy. However, the 2nd natural period obtained from experiment is between 0.74 and 0.78 s, which is smaller than the numerical result.

The coefficients for Rayleigh damping are calculated with damping factor 0.021 and 0.019 for the 1st and 4th modes, respectively, which are the two lowest modes in the x-direction defined in Fig. 1. The coefficients for mass matrix and the tangent stiffness matrix under assumption that all materials are in elastic state are 0.251 and 0.00118, respectively.

A 20-s segment of the 60 % Takatori record is used as an input ground motion for the shake-table test using the E-Defense, where the EW, NS, and UD components are applied in the x-, y-, and z-directions, respectively. The acceleration measured

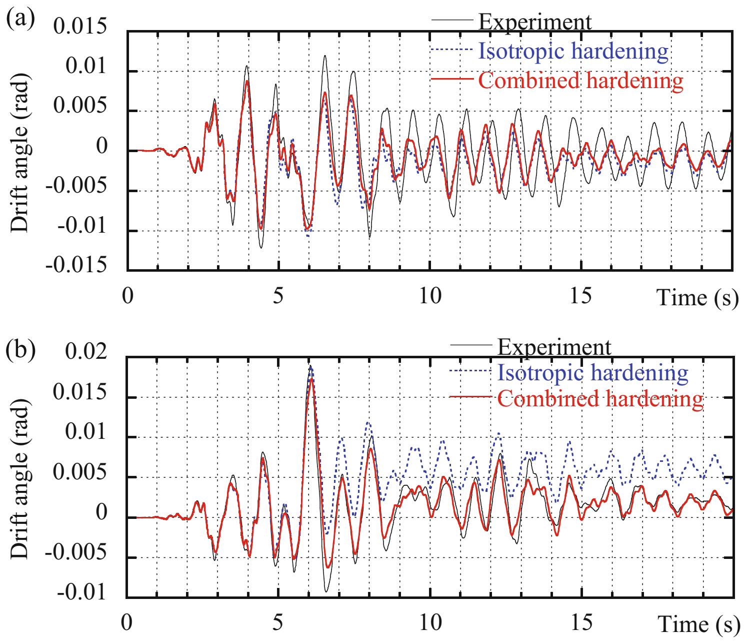


Fig. 5 Time histories of interstory drift angle of 1st story; (**a**) *x*-direction, (**b**) *y*-direction [13]

on the shaking table during the test [26] is applied at the bottom of the base of the analysis model.

The Hilber–Hughes–Taylor method (α-method) is used for time integration, where the parameter α for numerical damping is −0.05. Computation takes approximately one month using 256 cores (1 node) of SGI Altix 4700 (CPU: 1.66 GHz dual core Intel Itanium processor, 256 cores/node, 8 nodes). The total number of time steps is 5020 including static analysis for application of the self-weight. Therefore, the average computation time for one time step is about 1592 s (=26.53 min). It is also noted that the average computation time for one time step is about 556 s when 360 nodes of the K computer [27] are used.

Figures 5a, b and 6a, b show the time histories of interstory drift angle and story shear force of the 1st story, respectively. The results obtained using the piecewise linear isotropic–kinematic hardening model described in Sect. 2.2, which is called combined hardening model for brevity, and the isotropic hardening model for steel material are shown in the figures. The interstory drift angle is calculated from the lateral displacement at the center of the 2nd floor. The shear force is calculated as the sum of the concentrated mass multiplied by the acceleration at the center of each floor for both the analysis and E-Defense test.

The *x*-directional interstory drift angle obtained by the E-Simulator agrees well with the experimental result, as shown in Fig. 5a, until approximately 7 s. However, the differences in the phase and in the amplitude become larger after 7 s, when

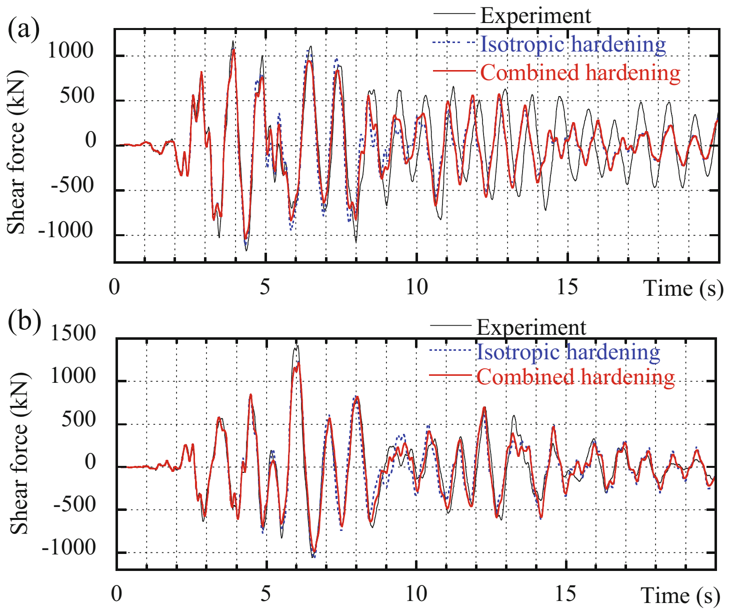


Fig. 6 Time histories of story shear force of 1st story; (**a**) *x*-direction, (**b**) *y*-direction [13]

the amplitude of the input acceleration decreases. Furthermore, small difference is observed between the results obtained using combined hardening model and isotropic hardening model. The *y*-directional interstory drift angle in Fig. 5b is close to the experimental result when the combined hardening model is used. On the other hand, the interstory drift angle remains in the positive region in the later part of the time history when the isotropic hardening model is used.

The time histories of the shear force in Fig. 6a, b exhibit similar characteristics as those of the interstory drift angle; however, the difference between the results obtained by combined hardening model and isotropic hardening model is small in both *x*- and *y*-directions.

2.4 *Seismic Response Analyses Under 100 % and 115 % Takatori Records*

The acceleration measured on the shaking table during the test under the excitation of the 100 % Takatori record [26] is applied at the bottom of the base of analysis model. An important phenomenon against such a severe input motion is the contact between column and slab. Figure 7a, b show the deformed shapes around the connections between beams and C2 column of the 2nd floor obtained by the models with the separate/penetrate condition and stick condition, respectively, at

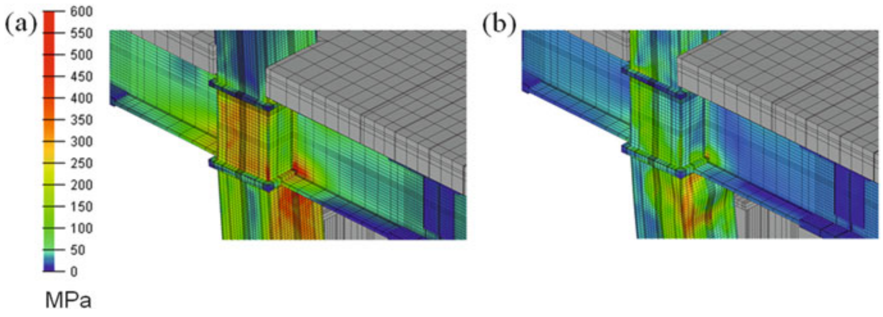


Fig. 7 Deformations around C2 column of 2nd floor at maximum displacement under 100 % Takatori record; deformation is not magnified, color contours indicate equivalent stress; (**a**) separate/penetrate condition at 6.24 s, (**b**) stick condition at 6.29 s [13]

the maximum deformation. The deformation due to buckling at the upper end of the column of the 1st story is observed for the model with the stick condition; however, no buckling is observed if the separate/penetrate condition is used.

Figure 8a, b show the time histories of interstory drift angle in the *x*- and *y*-directions, respectively, of the 1st story. In the E-Defense test, the frame collapsed completely in the positive *x*- and *y*-directions after 6 s. However, the frame did not collapse completely in the simulation using the E-Simulator. Therefore, the input acceleration obtained at the E-Defense test is multiplied by 1.15 to generate the 115 % Takatori record, which is applied at the base of the E-Simulator model to obtain the time histories of the interstory drift angle as shown in Fig. 8a, b. The magnitude of the drift angle is smaller than the experimental result, even for the case under the 115 % record. In the *x*-direction, the difference between the responses of the models with the separate/penetrate condition and the stick condition is small. Figure 9 shows the deformation and distribution of the equivalent stress at 6.46 s using the model with the stick condition under the 115 % Takatori record. The collapse behavior due to local buckling at the top and bottom of the columns in the 1st story has been simulated successfully.

3 Buckling and Dynamic Analyses of Column for Verification of FE-Model

3.1 Buckling Analysis

As mentioned above, the E-Simulator can simulate both global and local behaviors, such as local buckling, simultaneously using a fine mesh of solid elements. However, the FE-mesh should be fine enough to reproduce deformation due to buckling. To investigate appropriate sizes of elements, we first carry out static

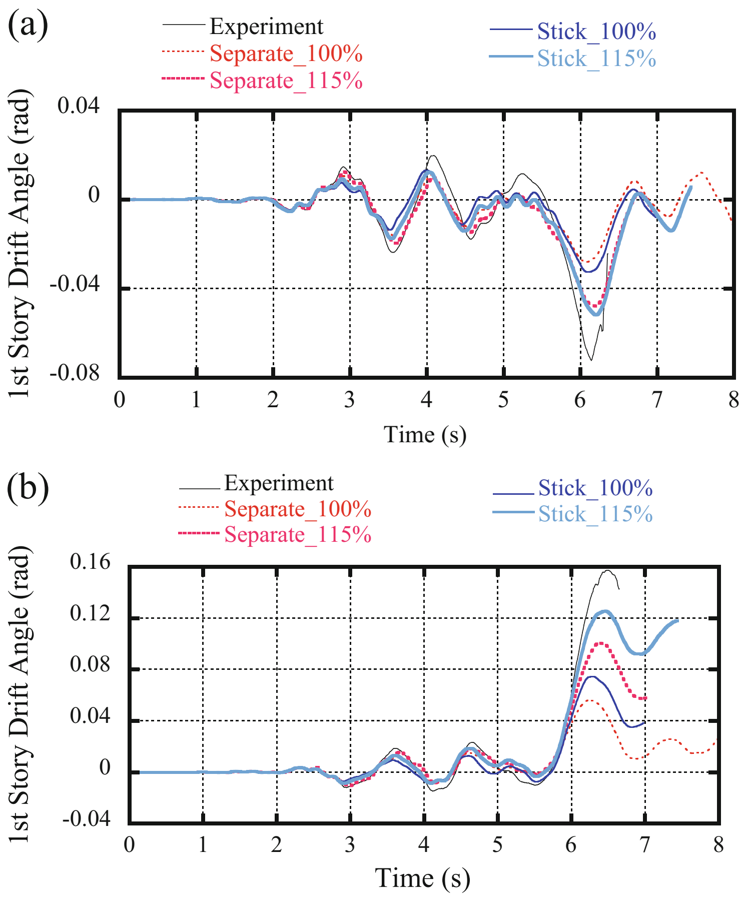


Fig. 8 Time-histories of interstory drift angle of 1st story obtained by E-Simulator and E-Defense full-scale test; (**a**) *x*-direction, (**b**) *y*-direction [13]

elastoplastic buckling analysis of a square steel tube column subjected to lateral displacement using meshes of different densities in order to verify the analysis model [13, 28].

The column corresponds to the C2 column in the 1st story of the four-story frame in Fig. 1 in Sect. 2. Figure 10 shows a typical FE-mesh of the column. The numbers of division in the lateral direction along each edge of the cross section, in the thickness direction, and in the longitudinal direction are denoted by *n*, *m*, and *k*, respectively. Then, the mesh of the column is represented by *tmbnLk*.

The lower end of the column is fixed. The nodes in the upper end are connected by rigid beams to an additional node called top node, which is located at the center of section and its translation in the *x*-direction and rotation around the *x*-axis are fixed. A concentrated mass of 5.169×10^4 kg representing mass of the upper stories

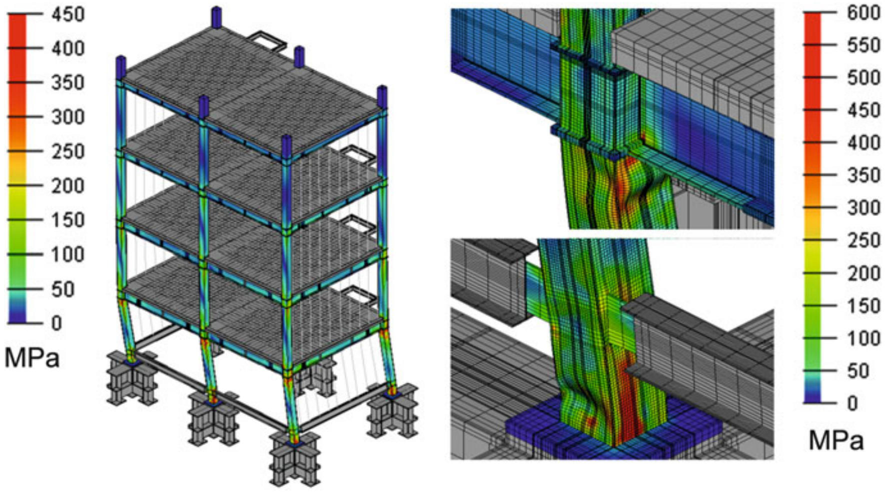


Fig. 9 Deformation (not magnified) and distribution of the equivalent stress at 6.46 s obtained by E-Simulator under the 115 % Takatori record (stick model) [13]

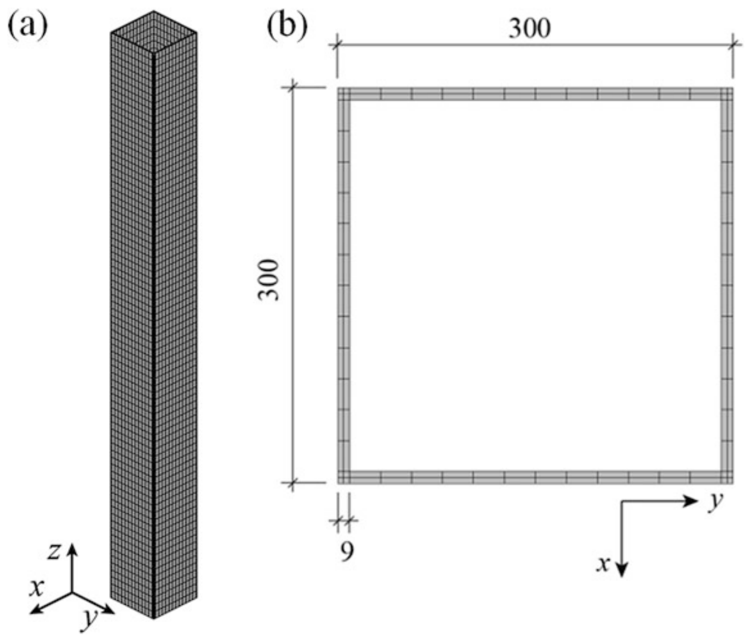


Fig. 10 Analysis model of a steel column (*t*2*b*12*L*80); (**a**) hexahedral mesh, (**b**) cross section of square tube column (unit: mm) [13, 28]

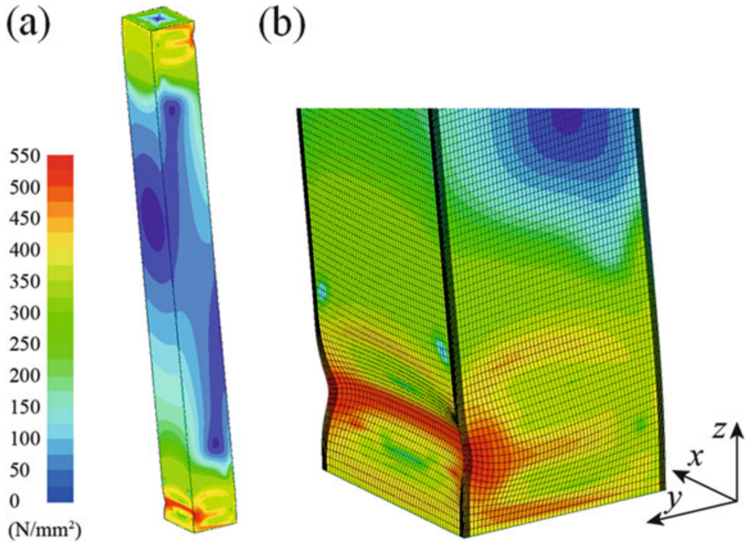


Fig. 11 Deformation (without magnification) and contour plot of equivalent stress at displacement 300 mm of Model *t*8*b*48*L*320; (**a**) entire column, (**b**) close-up view of column base [13, 28]

of the frame is placed at the top node, and the corresponding downward force of 507.1 kN is applied at the top node before application of monotonically increasing forced *y*-directional displacement.

The piecewise linear isotropic hardening model is used for steel, and its parameters are determined from the uniaxial test results distributed for the blind analysis contest of the four-story frame [24]. The hexahedral element with linear interpolation functions enhanced by incompatible modes [29, 30] is used in the analysis. The element is fully integrated using eight integration points.

Figure 11 shows a deformation due to elastoplastic local buckling with inextensible mode. Figure 12 shows relation between the forced *y*-directional displacement and the reaction force in the *y*-direction at the top node. The solution obtained using Model *t*8*b*48*L*320 with a very fine mesh is plotted in solid line, and this solution is designated as the *reference solution* or *Ref.* in the following figures. The solution using the fully integrated linear hexahedral elements without incompatible modes is plotted in dotted line, which shows that incompatible modes are needed for accurate simulation of local buckling.

The aspect ratios r_b and r_t are defined as

$$r_b = \min\{b_e/L_e, L_e/b_e\} \tag{3}$$

$$r_t = \min\{t_e/L_e, L_e/t_e\} \tag{4}$$

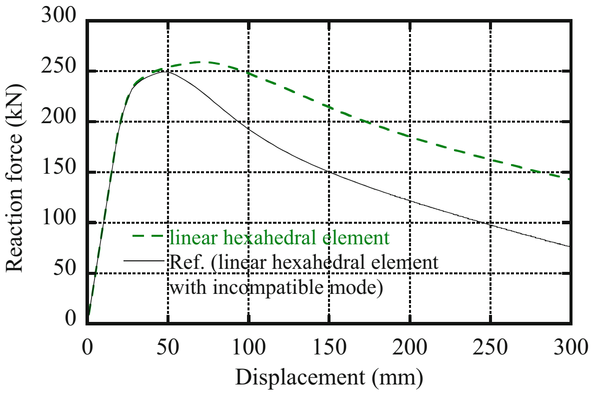


Fig. 12 Relation between forced y-directional displacement and reaction force of Model $t8b48L320$ with linear hexahedral elements and linear hexahedral elements with incompatible modes [13, 28]

where b_e, t_e, and L_e are the element sizes in b (lateral), t (thickness), and L (longitudinal) directions, respectively; i.e., the element has an unfavorable shape if r_b and/or r_t are far smaller than 1.

Figure 13a, b show the relation between the forced y-directional displacement and the reaction force for different mesh divisions in longitudinal and lateral directions, respectively. It is seen from these figures that fine mesh in the longitudinal direction contributes to the improvement of the solution; whereas fine mesh in the lateral direction does not improve the accuracy. Three different numbers of division in the thickness direction, that is, 2, 4, and 8, have also been compared for Models $b12L80$, $b24L160$, and $b48L320$, which have the same aspect ratio r_b. Although the details are omitted, Model $b12L80$ cannot be improved by increasing the division in the thickness direction, while improvement is achieved for Model $b24L160$. It has also been confirmed that two elements in the thickness direction are enough for Model $b48L320$ that has sufficiently fine mesh in the lateral and longitudinal directions.

Note that the number of CG iterative steps of the CGCG method increases when the aspect ratios r_b and/or r_t are very small because the current version of the CGCG method does not utilize any preconditioning method even some elements have unfavorable aspect ratios. By contrast, a coarser mesh in the intermediate region between the top and bottom reduces the computational cost without sacrificing the accuracy, simply because it leads to smaller number of DOFs.

3.2 *Dynamic Analysis*

Next, dynamic elastoplastic buckling analysis is carried out using different mesh densities and time increments. Sinusoidal input acceleration of frequency 0.8 Hz and maximum amplitude 4 m/s^2 is applied in the y-direction at the lower end of the

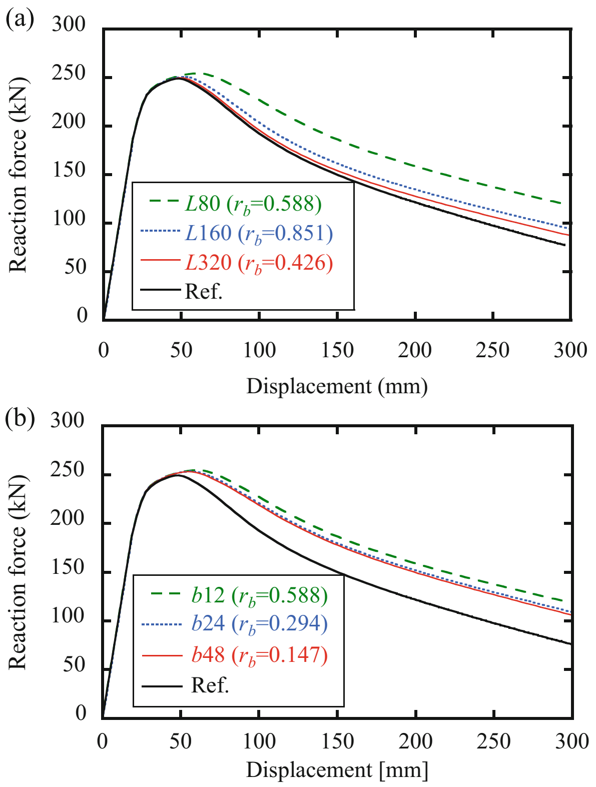


Fig. 13 Relation between forced *y*-directional displacement and reaction force for different numbers of mesh divisions in (**a**) longitudinal direction, Models *t*4*b*12*L*{80, 160, 320}, and (**b**) lateral direction, Models *t*4*b*{12, 24, 48}*L*80 [13, 28]

column. The amplitude increases gradually from time 0 to 5 s. The displacements in the *x*- and *z*-directions are constrained at the lower end. A mass of 51.69×10^3 kg is placed at the top node, and its weight is applied before dynamic analysis. The ADVENTURECluster/E-Simulator has function of automatic adaptation of the time increment Δt. The lower bound for Δt is 2.0×10^{-4} s, and the initial time increment that is also used as the upper bound is denoted by Δt_0.

Figure 14 shows the time histories of the displacement in the *y*-direction at the top node for different values of Δt_0 and mesh division. When buckling occurs, Δt is reduced automatically to improve the convergence of the Newton–Raphson iteration. However, the cases with $\Delta t_0 = 0.04$ s have smaller number of total time steps than $\Delta t_0 = 0.01$ s. As shown in Fig. 14, Model *t*2*b*48*L*320 with fine mesh has almost the same results for $\Delta t_0 = 0.01$ and 0.04 s. On the other hand, the results are different for $\Delta t_0 = 0.01$ and 0.04 s of Model *t*2*b*12*L*80 with a rather coarse mesh. Although the history of displacement of Model *t*2*b*12*L*80 with $\Delta t_0 = 0.04$ s is similar to that of Model *t*2*b*48*L*320, deformation and stress distribution of Model *t*2*b*12*L*80 with $\Delta t_0 = 0.04$ s are very different from those of Model *t*2*b*12*L*320. Hence, it is concluded that accurate results can be obtained using a fine mesh with a rather large time increment for the dynamic buckling problem, which is categorized as a slow dynamics problem.

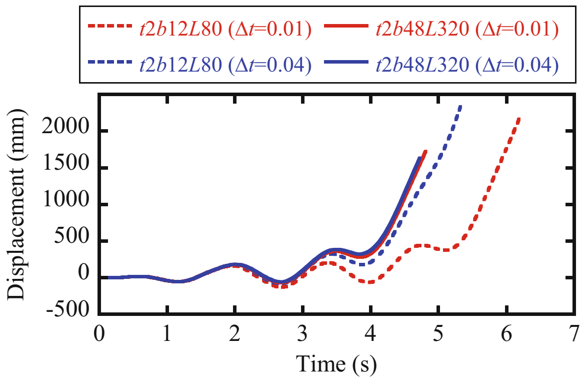


Fig. 14 Time histories of horizontal displacement at top node [13, 28]

4 31-Story Steel Frame

4.1 FE-Analysis Model

We carry out eigenvalue analysis and time-history seismic response analysis of a 31-story super-high-rise steel building frame as shown in Fig. 15 [31]. The story height is 5.4 m for the 1st and 2nd stories, and 4.1 m for the upper stories, and the total height is 129.7 m. Buckling-restrained braces are located in the core (center region of the plan). The beams, columns, panels at connections, gusset plates connecting braces to beams and columns, and diaphragms connecting beam flanges to columns are divided into hexahedral FE-mesh as shown in Fig. 16 with 15,592,786 elements, 24,765,275 nodes, and 74,295,825 DOFs. The plates such as flanges and webs of beams are divided into at least two layers of solid elements. The rebars in the slab and the studs connecting the flange and slab are omitted; accordingly, the lower surface along the boundary of the slab is directly connected to the upper surface of the flange. The size of each element in the longitudinal direction of a beam or a column is approximately 70 mm near the connections, while a coarser mesh is used in the region far from the connections.

The materials of the frame are steel for beams, columns, and plates, and reinforced concrete for slabs. Young's modulus E and Poisson's ratio ν of the steel material are 205 kN/mm^2 and 0.3, respectively, and kinematic hardening with yield stress $\sigma_Y = 330$ N/mm^2 or 445 N/mm^2 and hardening coefficient $H' = 205$ N/mm^2 is used. The concrete material of slab is also assumed to be elastoplastic with kinematic hardening, where $E = 22.7$ kN/mm^2, $\sigma_Y = 20$ N/mm^2, $\nu = 0.2$, and $H' = 22.7$ N/mm^2. The mass density of steel is 7.86×10^3 kg/m^3, whereas that of slab with thickness 0.1275 m is increased to incorporate the floor mass.

The base beams are assumed to remain in elastic range and have the same sections as those in the 2nd floor; however, Young's modulus is multiplied by 5.5 to represent the stiffness of the underground structure. The nodes in each column base are connected by rigid beams to a node at the center of the column base, which is pin-supported.

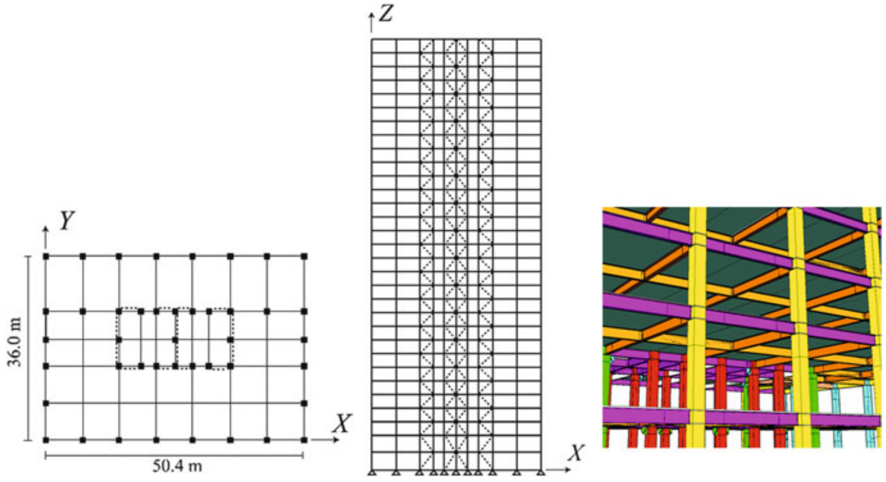


Fig. 15 31-story super-high-rise building frame [31]

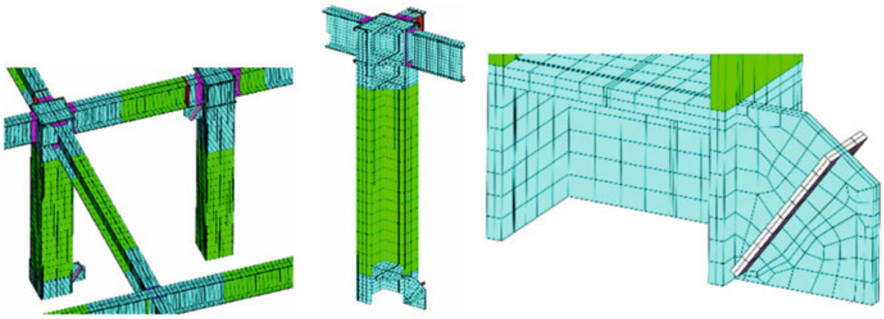


Fig. 16 Hexahedral FE-mesh of 31-story frame [31]

4.2 Analysis Results

The six lowest natural periods obtained by eigenvalue analysis are 3.253, 2.870, 2.616, 1.032, 0.951, and 0.850 s. The three lowest eigenmodes are shown in Fig. 17. The 1st and 3rd modes correspond to the two lowest modes in the y-direction, whereas the 2nd mode exhibits torsional vibration.

Time-history analysis is carried out for the three dimensional input motions of the Takatori record. Note that the EW, NS, and UD components correspond to x-, y-, and z-directions, respectively. The duration of the motion is 10 s (from 1.7 to 11.7 s of the original motion). The stiffness-proportional damping is used, where the damping factor is 0.02071 for the 1st mode. The Hilber–Hughes–Taylor method is used for time integration with parameters $\alpha = -0.05$, $\beta = (1-\alpha)^2/4 = 0.275625$, and the initial time increment 0.01 s, which is adapted automatically. In this analysis,

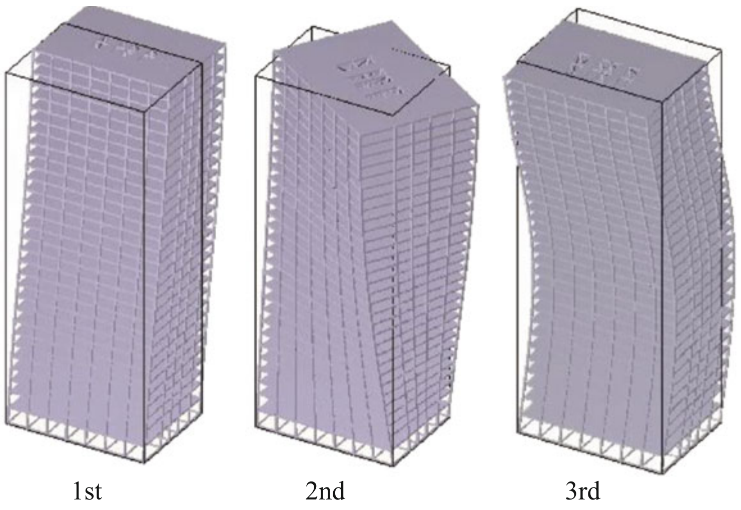


Fig. 17 Three lowest eigenmodes of 31-story frame [31]

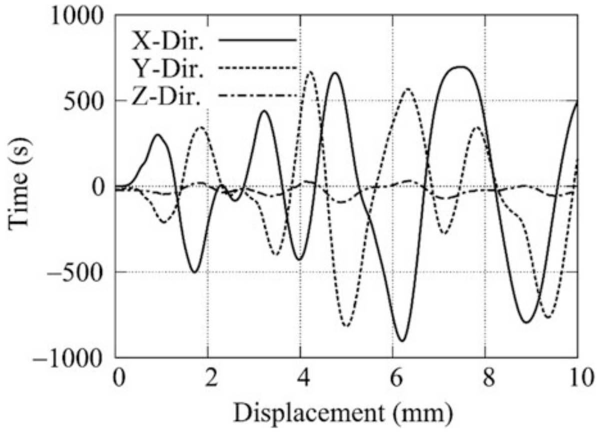


Fig. 18 Time histories of displacement at a node in the corner column in the roof; *solid line*: *x*-direction, *dashed line*: *y*-direction, *chained line*: *z*-direction [11]

192 cores of the super-computer T2K with AMD Quad Core Opteron 2.3 GHz (24 nodes $\times$ 8 cores/node) at the University of Tokyo are used. Average computation time for single time step is 12312 s (=3.42 h).

Figure 18 shows time history of the displacement at a node in the corner column in the roof (32nd floor). Figure 19 shows distribution of the equivalent plastic strain at time 6.21 s, when nearly maximum displacement is observed at the corner column of the roof. Note that red and blue colors correspond to large and small equivalent plastic strains, respectively. This way, the global and local responses of a super-high-rise building can be simulated using a sophisticated FE-model without resort to a macro model such as plastic hinge.

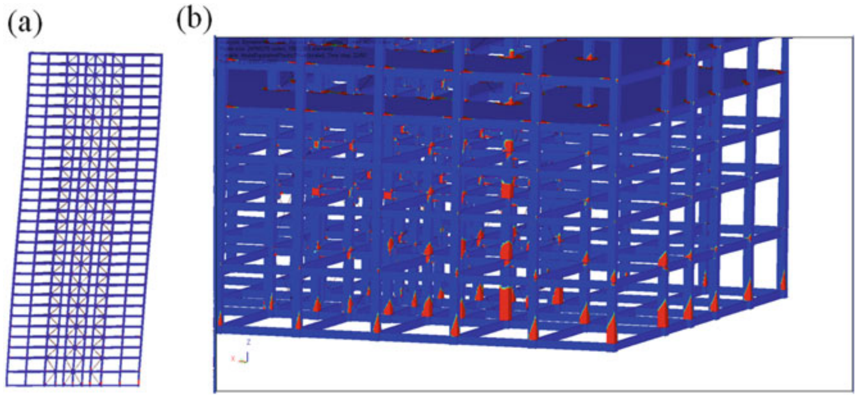


Fig. 19 Deformation (magnified 10 times) at 6.21 s; (**a**) entire frame, (**b**) close-up view with distribution of equivalent plastic strain [11]

4.3 Soil-Structure Interaction Analysis Using the K Computer

Soil-structure interaction analysis is carried out for the 31-story frame [32]. We use the FE-model described in Sect. 4.1; however, we omit the truss elements representing braces. Figure 20a shows the analysis model of the super-high-rise steel frame combined with the soil region. A mat slab of the size 51.0, 36.9, and 3.9 m in the *x*-, *y*-, and *z*-directions, respectively, is placed on the soil to support the frame.

The size of the soil region is 1000, 1000, and 100 m in the *x*-, *y*-, and *z*-directions, respectively. The hexahedral solid elements of the mat slab and the soil region are combined with those of the super-high-rise steel frame. As seen in Fig. 20b, the element sizes for the frame, mat slab, and soil region are very different. The mat slab and soil are divided into cubes with edge lengths 0.3 and 2.0 m, respectively. These meshes are assembled using 2,962,659 independent MPCs. The total numbers of elements, nodes, and DOFs are 28,363,862, 37,311,413, and 111,934,239, respectively. The materials of frame members and mat slab are steel and reinforced concrete, respectively, both of which are represented by the von Mises yield criterion with kinematic hardening. Soil is assumed to be elastic.

A preliminary analysis is conducted using the K computer to demonstrate feasibility of the ultra-large-scale parallel computation for soil-structure interaction analysis of building structures [32]. The Takatori record is applied at the bottom of soil region. Free boundary condition is assigned on the lateral faces and top face of the soil region. The dead load due to the gravity is applied statically before the dynamic analysis for simulating the seismic response. The computation time for the duration 17 s of the seismic response analysis is approximately 18 days using 256 nodes (2048 cores) of the K computer. The initial time increment is 0.1 s, which is reduced automatically when the Newton–Raphson iteration does not converge.

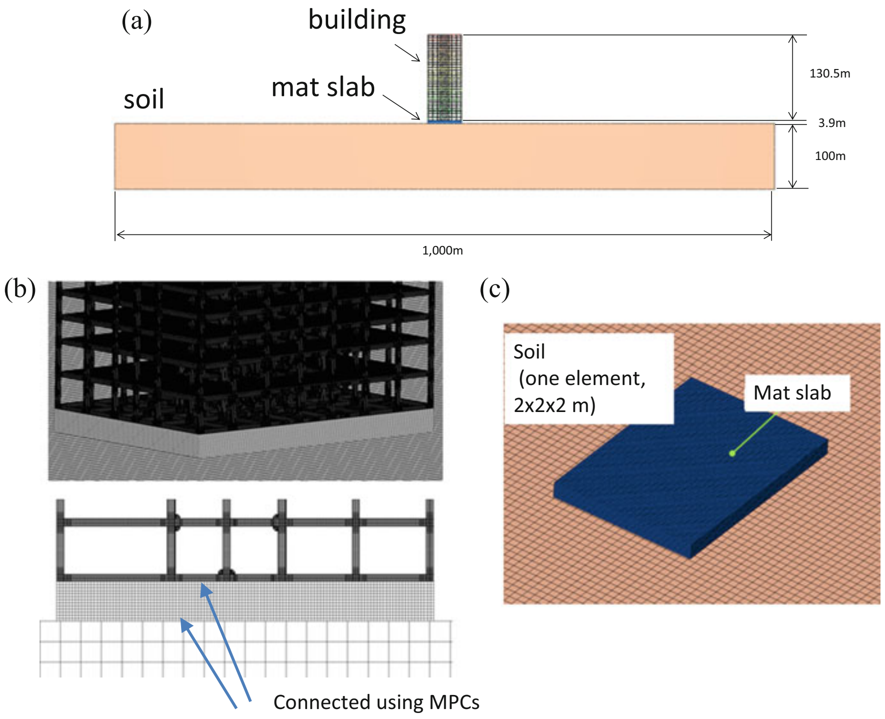


Fig. 20 Mesh of super-high-rise steel frame, mat slab, and soil region; (**a**) elevation, (**b**) details, (**c**) mat slab [32]

One of the key technologies for ultra-large-scale parallel FE-analysis is visualization, which is almost impossible using a graphic workstation because the amount of output data is too large to transfer from a super-computer to a graphic workstation. A parallel offline (server side; software) rendering code (e.g., [33]) is developed by Ogino [34] using a library called VSCG developed by Wada et al. [35]. The VSCG library can generate ultra-precise images, e.g., an image with $10\text{K} \times 10\text{K}$ pixels, without using a general graphics application programming interface (API) such as OpenGL. Since this code is also implemented on the K computer, the images for all time steps can be generated by a batch job on the K computer without transfer of the analysis results. Only the ultra-precise images are to be transferred from the K computer to a personal computer. An example of output visualization is shown in Fig. 21.

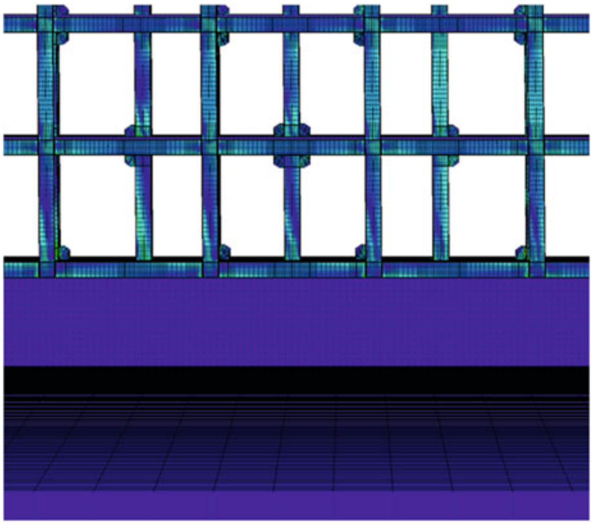


Fig. 21 Deformed configuration with the contour of equivalent stress (*blue*: 0.0, *red*: 300 MPa) [32]

5 Static Analysis of Composite Beam Model

5.1 FE-Model

We next carry out cyclic static analysis of a composite cantilever, as shown in Fig. 22, supported by a column [22, 36]. "Composite" means that interaction between the steel beam and concrete slab is taken into account. The sections of beam and column are RH-$400 \times 200 \times 8 \times 13$ (height $=400$, width $=300$, web thickness $=8$, flange thickness $=13$ mm) and RHS-300×9 (section size $=300$, thickness $=9$ mm), respectively, in Japanese specification. The beam, column, connection, slab, and rebar (wire mesh) are discretized into hexahedral solid elements in the same manner as Fig. 3 in Sect. 2 for the four-story frame.

The piecewise linear isotropic–kinematic hardening model described in Sect. 2.2 is used for the steel material. The hardening coefficients in initial loading are identified from the result of tensile uniaxial coupon test as shown in Fig. 23 for the column, web, flange, and diaphragm [36]. Although material parameters for flange and web of the beam are different, those for flange are used also for web. Note that only the hardening coefficients for the first loading can be obtained from the tensile uniaxial test; therefore, the parameters after reloading are estimated based on the properties observed in the cyclic test [23] as described in Sect. 2.2. The extended hyperbolic Drucker–Prager model is used for the concrete material to simulate the asymmetric behavior in tension and compression. The parameters are the same as those for the four-story frame in Sect. 2.

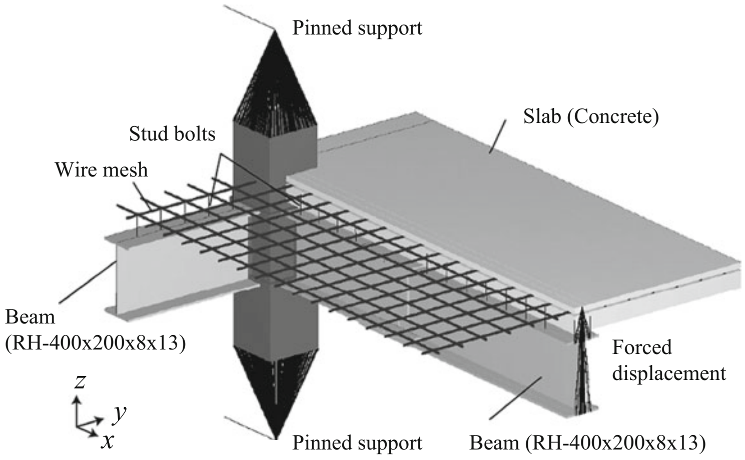


Fig. 22 Composite beam model with details of slab, rebars, and boundary conditions [22]

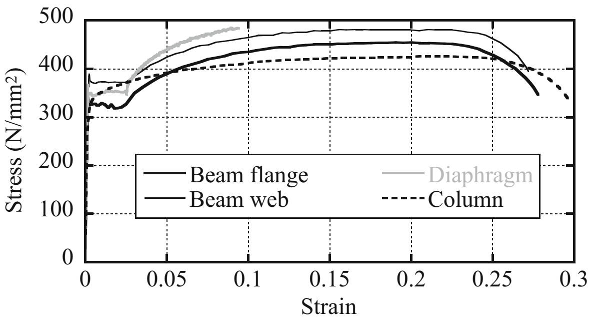


Fig. 23 Relations between true stress and logarithmic strain obtained from tensile uniaxial test [36]

The slab has rebars as shown in Fig. 22, and is supported by stud bolts as shown in Fig. 24. The column is pin-supported at its top and bottom using MPCs. Forced cyclic vertical displacement is given at the free-end of the cantilever. Non-frictional contact condition is assigned between the slab and column, and re-contact is appropriately incorporated. The column face is considered to be master, and the slab nodes are slave.

5.2 Analysis Results

Responses of pure steel beam (without slab) and composite beam (with slab) are compared under forced cyclic displacements. Let M_{b} denote the bending moment at the positive (right) side of the beam-column connection as defined in Fig. 25, which

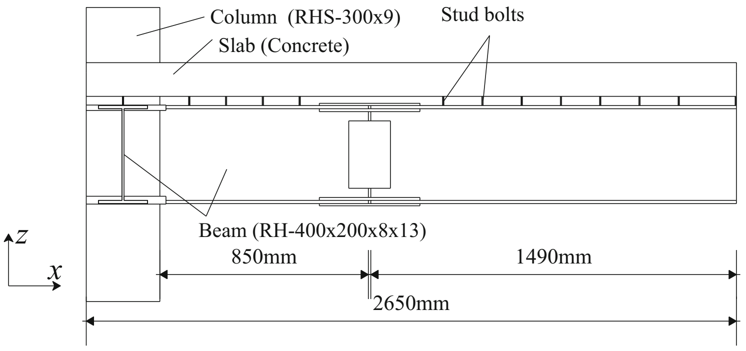


Fig. 24 Details of sections of beam, column, and slab supported by stud bolts [22]

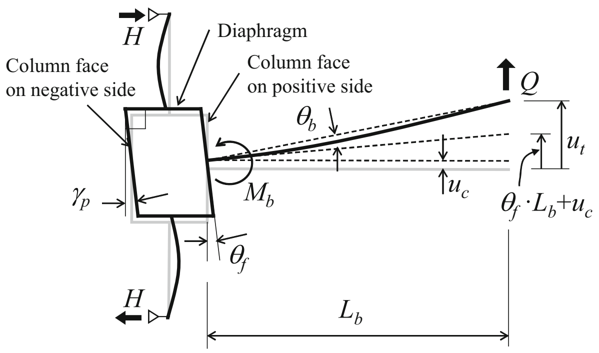


Fig. 25 Deformation of beam, column, and connection panel [36]

is computed from the reaction force Q and the length of beam L_b as [22, 36]

$$M_\mathrm{b} = QL_\mathrm{b} \tag{5}$$

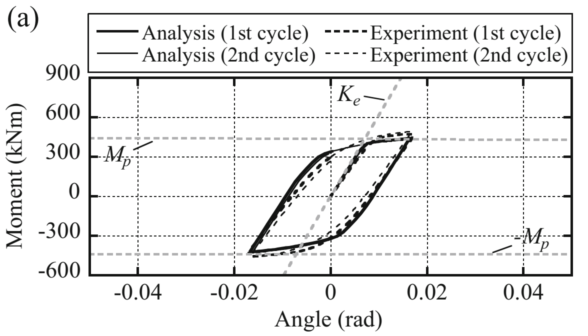
The net deflection angle θ_b is defined as

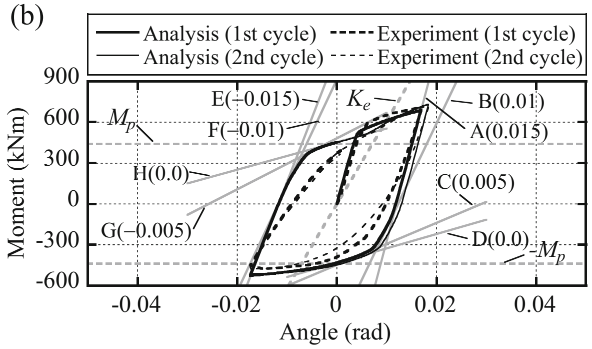
$$\theta_\mathrm{b} = [u_\mathrm{t} - (\theta_\mathrm{f}L_\mathrm{b} + u_\mathrm{c})]\,/L_\mathrm{b} \tag{6}$$

where u_t and u_c are the vertical displacements at the free-end and column face, respectively, and θ_f is the rotation at the column face.

Figure 26a, b show the $M_\mathrm{b} - \theta_\mathrm{b}$ relations of pure steel beam and composite beam, respectively, in the 1st and 2nd cycles, where K_e is the theoretical elastic

Fig. 26 Relation between deflection angle and bending moment in 1st and 2nd cycles; (**a**) pure steel beam, (**b**) composite beam





stiffness of the pure steel beam, and M_p is the fully plastic moment of the steel beam. The $M_b - \theta_b$ relations in the 3rd and 4th cycles are shown in Fig. 27a, b, respectively.

For the pure steel beam, the initial stiffness of the FE-model is 93 % of the theoretical value, which is slightly larger than the experimental result. The unloading stiffness of FE-model is slightly smaller than experimental result. The strength by analysis also agrees well with the experimental result. By contrast, for the composite beam, the initial stiffness is slightly larger than the experimental result, and the unloading stiffness is larger than the experimental result. Since we did not incorporate fracture property in the Drucker–Prager model, the neutral line of the beam remains near the upper flange in the transition process of the slab from compression to tension; accordingly, the bending stiffness is overestimated.

The deformation of the pure steel frame in the 4th cycle is almost symmetric with respect to the middle plane of the beam as seen in Fig. 28a. On the other hand, the lower flange of the composite beam has larger deformation than the upper flange due to local buckling as seen in Fig. 28b.

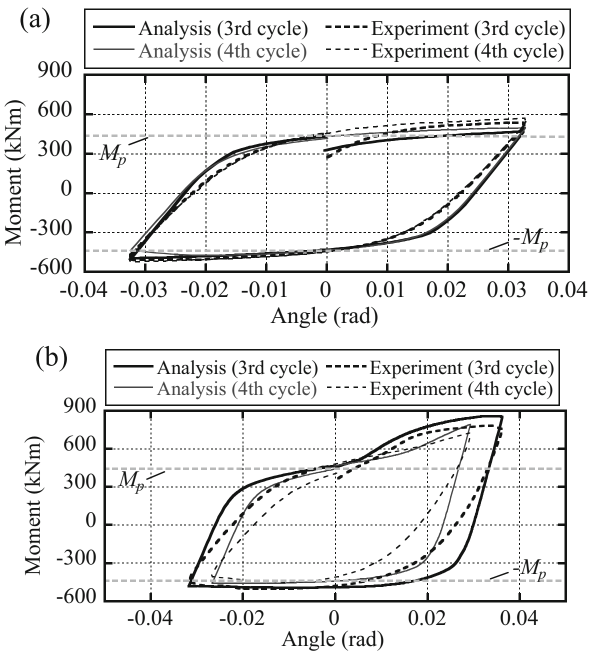


Fig. 27 Relation between deflection angle and bending moment in 3rd and 4th cycles; (**a**) pure steel beam, (**b**) composite beam [22, 36]

The stresses of concrete and rebars at maximum upward deformation in the 3rd cycle are plotted in Fig. 29, where the slab is divided into two parts showing the concrete and rebars, respectively. As seen from the figure, the slab has large stress due to contact to column face.

6 ALC Panel

6.1 FE-Model

In the design process of building structures, empirical damping models such as stiffness-proportional damping and the Rayleigh damping are usually used with a conventional damping factor 0.02 or 0.03. However, a detailed analysis is expected to simulate the responses of nonstructural components such as ALC panels, which are supposed to contribute to the damping properties of the frame under seismic motions. In this section, some analysis results are presented for static cyclic response of ALC panels [37, 38].

ALC panels are often used for exterior wall of building frames in Japan. We simulate the experimental results of elastoplastic cyclic responses of an ALC panel by Matsuoka et al. [17, 39]. Figure 30 shows the front view of the model. The width, height, and thickness of a panel are 600, 2560, and 100 mm, respectively. Six panels

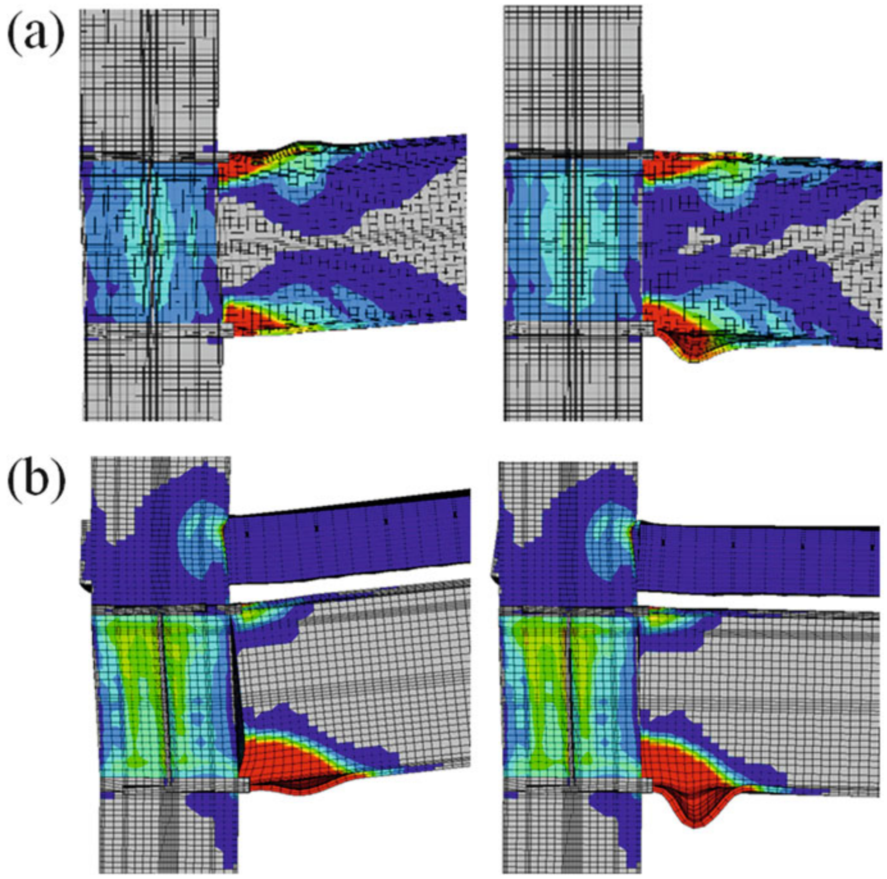


Fig. 28 Deformation in 4th cycle; (**a**) pure steel beam (*left*; $\theta_b = 0.0329$ rad, *right*: $\theta_b = -0.0325$ rad), (**b**) composite beam (*left*: $\theta_b = 0.0293$ rad, *right*: $\theta_b = -0.0266$ rad) [22]

are connected in the experiment; however, we use only two panels to investigate fracture properties due to friction and contact.

The ALC panels, attachment plates, the supporting members of steel angles and steel channels, and the H-section steel beams of loading frame are modeled using hexahedral solid elements with linear displacement interpolation and incompatible mode. The section details of the FE-model are shown in Fig. 31. The ALC panels and all steel plates are discretized into two layers of elements in the thickness direction. The columns are modeled using rigid beam elements, which are pin-jointed and linked with pin joints to the end surfaces of the upper beam, which are constrained also by rigid beam elements to prevent stress concentration under forced horizontal displacement.

An O-bolt connects an anchor steel bar inside an ALC panel and an attachment plate as shown in Fig. 32. It allows rotation of an ALC panel around two horizontal

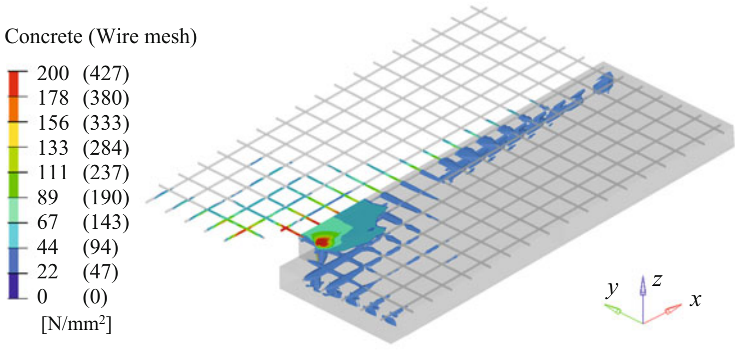


Fig. 29 Stresses of concrete and rebars (wire mesh) at maximum upward deformation in 3rd cycle [22]

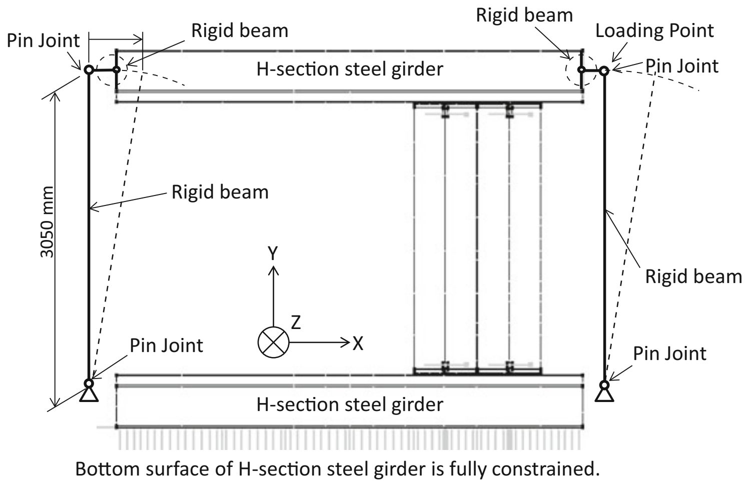


Fig. 30 ALC panel wall model and loading frame [17, 39]

axes and translational displacement in the direction of the member axis of the anchor steel bar. The model has 35,396 nodes, 22,972 hexahedral solid elements, and 66 rigid beam elements. The total number of DOFs is 106,188.

6.2 Material Properties and Loading Conditions

The mass density, Young's modulus, and Poison's ratio of steel members are 7.86×10^3 kg/m^3, 205.0 kN/mm^2, and 0.3, respectively. A bilinear model with isotropic hardening is used with the yield stress 335 N/mm^2 and the hardening coefficient 205.0 N/mm^2. The mass density, Young's modulus, Poison's ratio, and

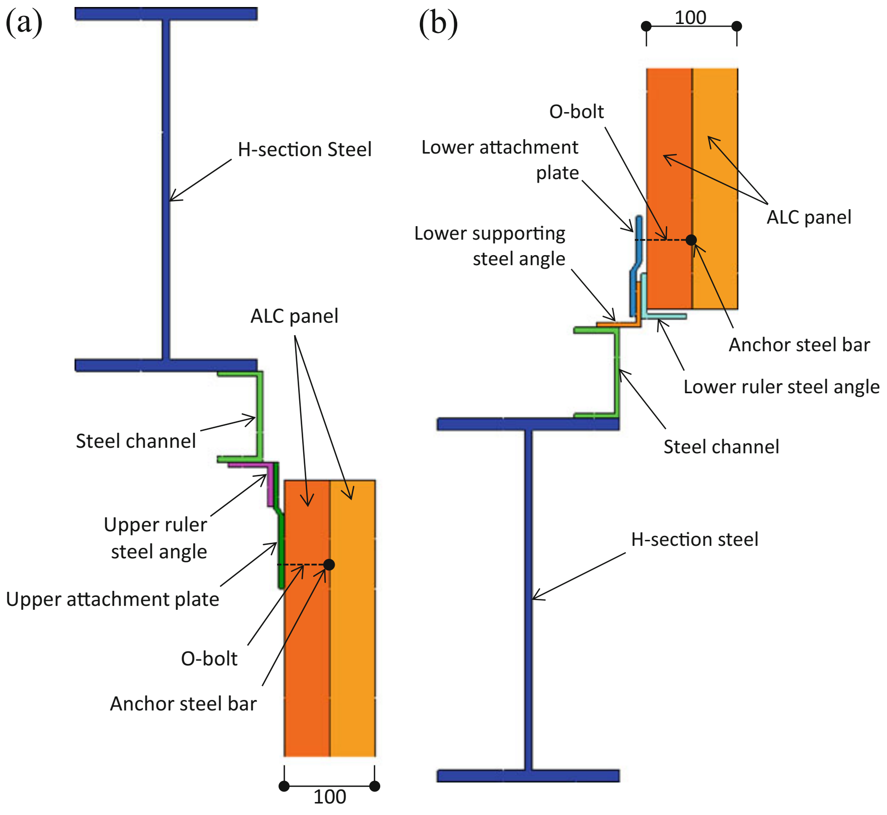


Fig. 31 Section details of ALC panel wall model; (**a**) detail of upper attachment plate, (**b**) detail of lower attachment plate [37, 38]

compressive strength of the ALC panels are 6.50×10^2 kg/m^3, 2.47 kN/mm^2, 0.2, and 5.35 N/mm^2, respectively. The extended hyperbolic Drucker–Prager model is used, where the tensile and shear yield stresses are 0.669 N/mm^2 and 0.717 N/mm^2, respectively.

Contact conditions are considered between three pairs of faces; (1) adjoining lateral faces of two ALC panels, (2) attachment plate and side surface of ALC panel, and (3) bottom surface of ALC panel and lower ruler steel angle. Separation and re-contact processes are considered although the change of normal vectors on the faces is neglected. Since the friction coefficient between steel and concrete ranges from 0.2 to 0.8 [40, 41], the friction coefficient between a steel member and an ALC panel is assumed to have a smaller value 0.1.

Static cyclic loading with incremental deformation amplitude up to the drift angle $\theta = 0.02$ rad is conducted. Forced displacement is applied at the upper-right pin joint shown in Fig. 30, and the rotation angle of column is calculated by dividing

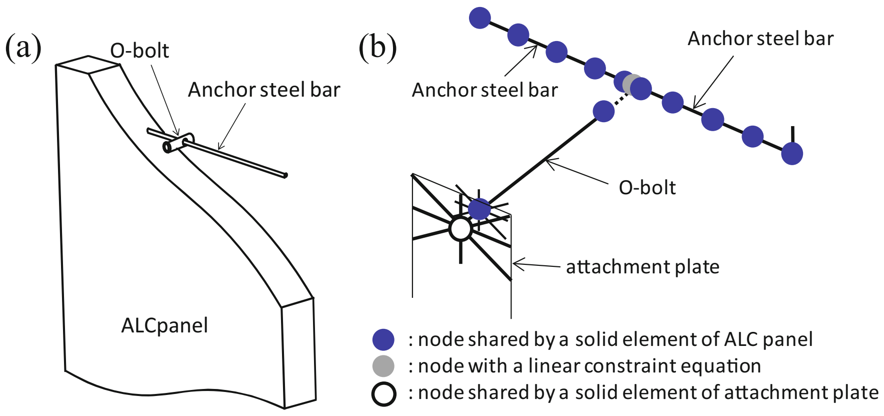

(a)
O-bolt
Anchor steel bar
ALCpanel
(b)
Anchor steel bar
Anchor steel bar
O-bolt
attachment plate
● : node shared by a solid element of ALC panel
● : node with a linear constraint equation
○ : node shared by a solid element of attachment plate

Fig. 32 O-bolt and anchor steel bar inside ALC panel; (**a**) schematic view, (**b**) FE-model [37, 38]

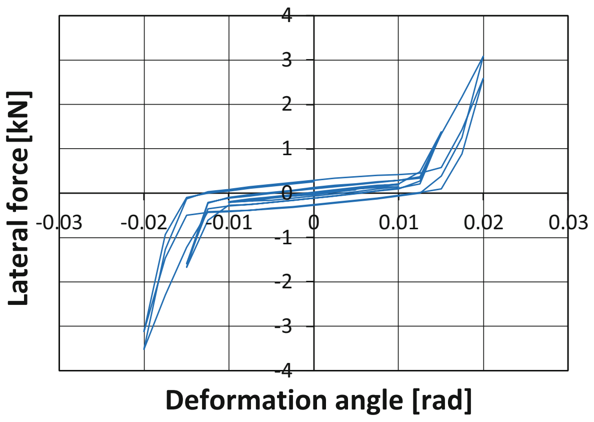


Fig. 33 Relation between lateral force and deformation angle [37, 38]

the forced displacement by 3050 mm, which is the height of the column of the loading frame.

6.3 *Analysis Result and Discussions*

Figure 33 shows the lateral force–deformation angle relation. The lateral resisting force drastically increases at $\theta = 0.015$ rad, when the corners of ALC panels come into contact with the lower ruler steel angle. After this contact, the slope of curve in the unloading process becomes larger than that in the loading process. This way, the ALC panels dissipate plastic energy that leads to hysteretic damping during seismic excitation.

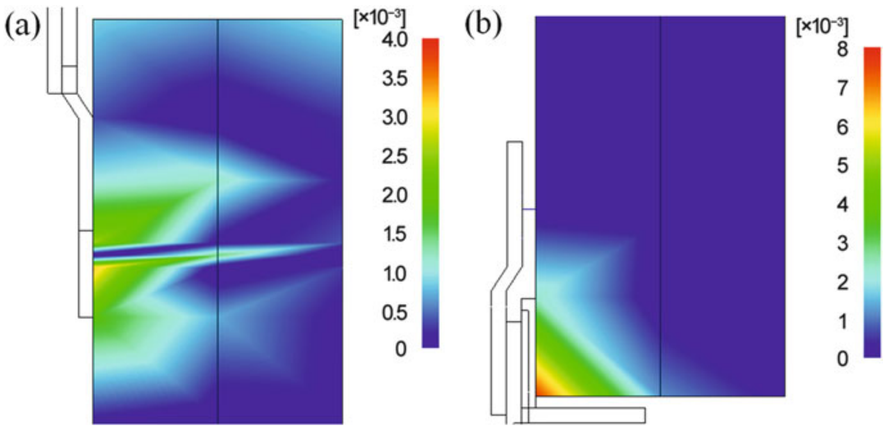


Fig. 34 Distribution of equivalent plastic strain at $\theta = 0.02$ rad; (**a**) around upper O-bolt, (**b**) at lower-left corner contacting lower ruler steel angle [37, 38]

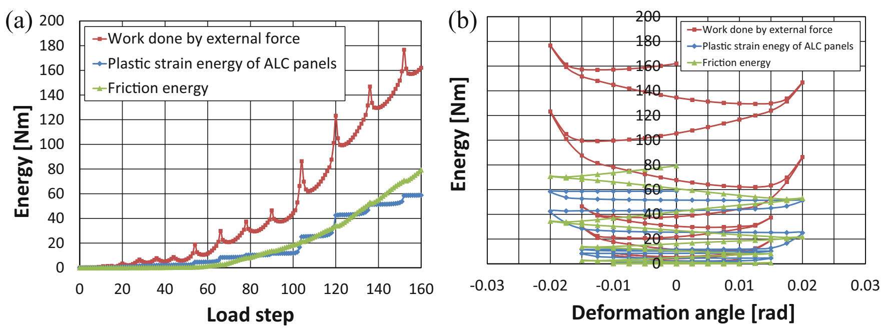


Fig. 35 Transitions of energy; (**a**) work done by external force and dissipated energy, (**b**) energy–deformation angle relation [37, 38]

Figure 34 shows the distribution of equivalent plastic strain at $\theta = 0.02$ rad; large plastic strain is observed near the corner contacting the lower ruler steel angle and the elements around the upper O-bolt. After this plastic deformation, the lateral force at a small deformation angle has nonzero positive value. This is because a slide with friction between the two ALC panels occurred due to the residual deformation, which was caused by the contact between the ALC panels and the lower ruler steel angle.

Figure 35a, b show transitions of external work, plastic strain energy, and friction energy with respect to load step and deformation angle, respectively. It is confirmed that friction becomes significant when $\theta \geq 0.015$ rad due to contact at the bottom of the panel. Note that the steep increase of the work done by the external force corresponds to accumulation of elastic potential energy after contact, which does not contribute to damping. It is seen from Fig. 35 that the energy dissipated by friction is comparable to the plastic strain energy of the ALC panels, and these energies are almost the same at the end of the loading steps. These results confirm that the friction and plastic deformation of ALC panels have significant effects on damping of the frame when story drift is moderately large.

7 Computational Performance

The main framework of the E-Simulator is the commercial software package ADVENTURECluster [6, 7], which has been extended from the open source version ADVENTURE system [42, 43]. These packages are based on the domain decomposition method for the linear algebraic solver and parallel implementation of finite element procedures such as stress integration. Implementation of the ADVENTURECluster uses the framework of the hierarchical domain decomposition method (HDDM) proposed by Yagawa and Shioya [44] in the similar manner as the ADVENTURE system. Akiba et al. [8, 9] developed the CGCG method for the ADVENTURECluster. In the CGCG method, a preconditioner is made by motions of the decomposed subdomains, which is similar to the coarse grid correction used in the balancing domain decomposition (BDD) method. In the CGCG method, however, static condensation using a direct solver in each subdomain is not conducted.

For complex structures such as building frames, incorporation of MPCs into domain decomposition methods is a challenging task especially when the number of imposed MPCs is very large. In the ADVENTURECluster, the CGCG method is applied to a projective space (see, for example, Ref. [45]) represented by a constraint matrix of MPCs, and several millions of MPCs can be taken into account. A rigid body element is also implemented using a set of MPCs to connect six DOFs including three translational DOFs and three rotational DOFs. An arbitrary shaped rigid body is represented by using a number of rigid beam elements. A point load or a forced displacement can be applied at a node connected by rigid beam elements.

We evaluate the performance of the ADVENTURECluster/E-Simulator using the 31-story high-rise building frame model presented in Sect. 4 [46]. The FE-model has 15,592,786 linear hexahedral elements, 24,765,275 nodes, and 74,295,825 DOFs.

The analysis of the 31-story frame model has been performed on various computer systems, such as TSUBAME 1.0; Hitachi HA8000, up to 64 nodes, 1024 cores (2008$\sim$2012) [47]; HP Sandy-Bridge Cluster system SL230s (2012); Fujitsu FX10, up to 128 nodes, 2048 cores (2011$\sim$2012); K computer, up to 512 nodes, 16,384 cores (2012$\sim$2014).

Table 1 shows the strong scaling performance with FX10. The vertical axis corresponds to the computational time for one step (0.01 s). “Flat” indicates that the parallel processing uses the flat MPI architecture, and “Hybrid” indicates the OpenMP-MPI hybrid architecture. Although each node of FX10 has 16 cores, the best performance is obtained when each of four OpenMP threads are assigned to 4 cores. The number of cores is between 256 and 2048. In “Flat,” the computation ended

Table 1 Strong scaling performance on FX10 [46]

Flat	No. of cores	256	384	512	1024	2048
	Calculation time (s)	2324.7	2163.0	2395.5	N/A	N/A
Hybrid	No. of cores	256	384	512	1024	2048
	Calculation time (s)	4590.1	3640.0	2978.4	1645.0	1730.0

Table 2 Strong scaling performance on K computer [46]

No. of cores	128	256	512	1024	1536	2048	4096	16,384
Calculation time (s)	9752.9	6828.8	3624.1	2199.5	1940.4	1516.3	1420.1	2460.9

abnormally when 1024 or 2048 cores are used, whereas in "Hybrid," the run ended normally. The computational times in "Flat" with 256, 384, and 512 cores are all smaller than "Hybrid" using the same number of cores; however, in "Flat," 1024 and 2048 cores failed. In this sense, overall good performance is obtained by the hybrid architecture. The best performance is 1645 s with the hybrid 1024 core architecture. However, 2048 cores could not achieve a better performance than 1024 cores.

Table 2 shows the strong scaling performance with the K computer. The number of cores is between 128 and 16,384, in which all of them assumed the hybrid architecture assigning one OpenMP thread to each of 8 cores per node. Each node of the K computer has 8 cores, and this type of hybrid architecture best performed. The scaling performance may not be good enough, and 4096 cores gives best performance, 1420 s; however, it is remarkable that computation can be successfully carried out with 16,384 cores.

8 Concluding Remarks

The results of detailed FE-analysis using the E-Simulator, which is a parallel FE-analysis software package for virtual shake-table tests of civil and building structures, are presented for the seismic response analyses of a four-story steel building frame, a 31-story super-high-rise steel frame, and structural components such as composite beam and external wall. It is shown through numerical examples that elastoplastic dynamic responses can be estimated with good accuracy using an FE-analysis using solid elements without resort to macro models such as plastic hinge and hysteresis model. With development of computational model and algorithms, the FE-analysis using solid elements is expected to be an indispensable tool for investigation of seismic response and design of complex building structures, as well as development of new components and devices of seismic control and design.

Acknowledgments This study is a part of Building Structures Working Group (Leader: Prof. Makoto Ohsaki, Kyoto University) of E-Simulator Development Committee (Leader: Prof. Muneo

Hori, The University of Tokyo) of NIED. The authors acknowledge the valuable contribution from the committee members, and the financial support by NIED. The present study was also supported by MEXT of Japan through a grant for research on HPCI Strategic Program Field No. 3. The contribution by Dr. Tomonobu Ohyama, Mr. Kiyoshi Yuyama, Mr. Shuhei Takaya, and Mr. Koji Nishimoto at Allied Engineering Corporation for model building and numerical analysis is also acknowledged.

References

1. Ohtani, K., Ogawa, N., Katayama, T., Shibata, H.: Construction of E-defense (3-D full-scale earthquake testing facility). In: Proceedings 13th World Conference Earthquake Engineering (13WCEE) Vancouver, Canada, Paper No. 189 (2004)
2. Website of Hyogo Earthquake Engineering Research Center (E-Defense) of the National Research Institute of Earth Science and Disaster Prevention (NIED). http://www.bosai.go.jp/hyogo/ehyogo/index.html
3. Hori, M., Noguchi, H., Ine, T.: Project report of development of numerical shaking table coping with E-defense. J. Earthq. Eng. Jpn. Soc. Civil Eng. **29**, 1420–1425 (2007) (in Japanese)
4. Nakashima, M., Ogawa, K., Inoue, K.: Generic frame model for simulation of earthquake responses of steel moment frames. Earthq. Eng. Struct. Dyn. **31**(3), 671–692 (2002)
5. Ibarra, L.F., Medina, R.A., Krawinkler, H.: Hysteretic models that incorporate strength and stiffness deterioration. Earthq. Eng. Struct. Dyn. **34**(12), 1489–1511 (2005)
6. Website of Allied Engineering Corporation. http://www.alde.co.jp/english/index.html
7. Akiba, H., et al.: Large scale drop impact analysis of mobile phone using ADVC on Blue Gene/L. In: Proceedings of the International Conference High Performance Computing Networking and Storage (SC06), Tampa, FL, USA (2006)
8. Suzuki, M., Ohyama, T., Akiba, H., Yoshimura, S., Noguchi, H.: Development of fast and robust parallel CGCG solver for large scale finite element analyses. Trans. Jpn. Soc. Mech. Eng. **A68**, 1010–1017 (2002) (in Japanese)
9. Akiba, H., Ohyama, T., Shibata, Y.: CGCG method for structural analysis and its enhancement. In: Proceedings of the 7th International Conference on Engineering Computational Technology (ECT2010), Valencia, Spain (2010)
10. Hughes, T.J.R.: The finite element method: linear static and dynamic finite element analysis. Dover publications, Mineola, NY (2000)
11. Miyamura, T., Ohsaki, M., Kohiyama, M., Isobe, D., Onda, K., Akiba, H., Hori, M., Kajiwara, K., Ine, T.: Large-scale FE analysis of steel building frames using E-Simulator. Prog. Nucl. Sci. Technol. **2**, 651–656 (2011)
12. Miyamura, T., Ohsaki, M., Yamashita, T., Isobe, D., Kohiyama, M.: Dynamic collapse analysis of four-story steel frame using E-Simulator. In: Proceedings of the 4th ECCOMAS Thematic Conference on Computational Methods in Structural Dynamics and Earthquake Engineering (CompDyn2013), vol. 49, Kos, pp. 1449–1469 (2015)
13. Miyamura, T., Yamashita, T., Akiba, H., Ohsaki, M.: Dynamic FE simulation of four-story steel frame modeled by solid elements and its validation using results of full-scale shake-table test. Earthq. Eng. Struct. Dyn. **44**(9), 1449–1469 (2015)
14. Yamada, S., Suita, K., Tada, M., Kasai, K., Matsuoka, Y., Shimada, Y.: Collapse experiment of 4-story steel moment frame: Part 1, Outline of test results. In: Proceeding of the 14th World Conference Earthquake Engineering (14WCEE), Beijing, China, Paper ID. S17-01-004 (2008)
15. Suita, K., Yamada, S., Tada, M., Kasai, K., Matsuoka, Y., Shimada, Y.: Collapse experiment of 4-story steel moment frame: Part 2 detail of collapse behavior. In: Proceedings of the 14th World Conference Earthquake Engineering, Beijing, China, Paper ID. S17-01-011 (2008)
16. IDEAS Ver. 2, User’s Manual, Amnis Corp., Seattle, WA (2006)

17. Matsuoka, Y., Matsumiya, T., Suita, K., Nakashima, M.: Test on seismic performance evaluation of exterior ALC walls with opening: E-defense experimental projects for steel buildings: Part 14. Summaries of technical papers of annual meeting, Architectural Institute of Japan, **C-1**, pp. 1081–1082 (2007) (in Japanese)
18. Chaboche, J.L.: A review of some plasticity and viscoplasticity constitutive theories. Int. J. Plast. **24**, 1642–1693 (2008)
19. Ucak, A., Tsopelas, P.: Constitutive model for cyclic response of structural steels with yield plateau. J. Struct. Eng. ASCE **137**(2), 195–206 (2011)
20. Simo, J.C., Taylor, R.L.: Consistent tangent operator for rate-independent elastoplasticity. Comp. Meth. Appl. Mech. Eng. **48**, 101–118 (1985)
21. Ohsaki, M., Zhang, J.Y., Miyamuram T.: A Heuristic algorithm for parameter identification of steel materials under asymmetric cyclic elastoplastic deformation. In: Proceedings of the 7th China-Japan-Korea Joint Symposium on Optimization of Structural and Mechanical Systems (CJK-OSM7), Huangshan, China, Paper No. J045 (2012)
22. Yamashita, T., Ohsaki, M., Kohiyama, M., Miyamura, T., Zhang, J.Y., Tagawa, H.: Detailed finite element analysis of composite beam under cyclic loads. J. Struct. Constr. Eng., Arch. Inst. Jpn. **74**, 1481–1490 (2014) (in Japanese)
23. Yamada, S., Imaeda, T., Okada, K.: Simple hysteresis model of structural steel considering the Bauschinger effect. J. Struct. Constr. Eng. Arch. Inst. Jpn. **67**(559), 225–232 (2002) (in Japanese)
24. Ohsaki, M., Kasai, K., Matsuoka, Y., Zhang, J.Y.: Results of recent E-Defense tests on full-scale steel building: Part 2, Collapse simulation and blind analysis contest. In: Proceedings of the Structures Congress. ASCE, Vancouver, Canada (2008)
25. Matsuoka, Y., Suita, K., Yamada, S., Shimada, Y.: Non-structural component performance in 4-story frame tested to collapse. In: Proceedings of the 14th World Conference on Earthquake Engineering, Beijing, China, Paper ID. S17-01-014 (2008)
26. Website of ASEBI at E-Defense, NIED, Project: E-Defense tests on full-scale four-story steel building. https://www.edgrid.jp/data/
27. Website of K Computer (RIKEN Advanced Institute for Computational Science). http://www.aics.riken.jp/en/.
28. Yamashita, T., Miyamura, T., Akiba, H., Kajiwara, K.: Verification of finite element elastic–plastic buckling analysis of square steel tube column using solid element. Transactions of Japan Society for Computational Engineering and Science, Paper No. 20130001 (2013) (in Japanese)
29. Simo, J.C., Rifai, M.S.: A class of assumed strain methods and the method of incompatible modes. Int. J. Numer. Meth. Eng. **29**, 1595–1638 (1990)
30. Simo, J.C., Armero, F.: Geometrically nonlinear enhanced strain mixed methods and the method of incompatible modes. Int. J. Numer. Meth. Eng. **33**, 1413–49 (1992)
31. Ohsaki, M., Miyamura, T., Kohiyama, M., Hori, M., Noguchi, H., Akiba, H., Kajiwara, K., Ine, T.: High-precision finite element analysis of elastoplastic dynamic responses of super-high-rise steel frames. Earthq. Eng. Struct. Dyn. **38**, 635–654 (2009)
32. Miyamura, T., Akiba, H., Hori, M.: Large-scale seismic response analysis of super-high-rise steel building considering soil-structure interaction using K computer. Int. J. High-Rise Building. **4**(1), 75–83 (2015)
33. Kawai, H., Ogino, M., Shioya, R., Yoshimura, S.: Vectorization of polygon rendering for off-line visualization of a large scale structural analysis with ADVENTURE system on the Earth Simulator. J. Earth Simul. **9**, 51–63 (2008)
34. HDDMPPS Project, "LexADV". http://adventure.sys.t.u-tokyo.ac.jp/lexadv/
35. Wada, Y., Kawai, H., Shioya, R.: Development of high resolution visualization library for very large scale analysis. In: Proceedings of the Conference on Computational Engineering and Science 18, Tokyo, Japan (2013) (in Japanese)
36. Yamada, S., Satsukawa, K., Kishiki, S., Shimada, Y., Matsuoka, Y., Suita, K.: Elasto-plastic behavior of panel zone in beam to external column connection with concrete slab. J. Struct. Constr. Eng., Arch. Inst. Jpn. **74**(644), 1841–1849 (2009) (in Japanese)

37. Kohiyama, M., Ohsaki, M., Miyamura, T., Yamashita, T.: Finite element analysis of contact corner of ALC panel using E-Simulator. J. Struct. Eng., Arch. Inst. Jpn. **60B**, 463–470 (2014) (in Japanese)
38. Kohiyama, M., Ohsaki, M., Miyamura, T., Yamashita, T.: Finite element analysis of damping mechanism of autoclaved lightweight aerated concrete panels for exterior walls of steel structures. In: Proceedings of the 11th World Congress of Computational Mechanics (WCCM11), Barcelona, Spain, Paper No. 1137 (2014)
39. Matsuoka, Y., Suita, K., Yamada, S., Shimada, Y., Akazawa, M., Matsumiya, T.: Evaluation of seismic performance of exterior cladding in full-scale 4-story building shaking table test. J. Struct. Constr. Eng., Arch. Inst. Jpn. **74**, 1353–1361 (2009) (in Japanese)
40. Matsuzawa, K., Ozawa, J., Watanabe, T. Experimental research on friction characteristics of steel and cementitious material. In: Proceedings of the Japan Concrete Institute, **30**, pp. 1141–1146 (2008) (in Japanese)
41. Katsuo, M., Ikenaga, M., Nagae, T., Nakashima, M.: Effect of velocity on dynamic coefficient of friction between steel and mortar. Summaries of technical papers of annual meeting, Architectural Institute of Japan, **C-1**, pp. 639–640 (2008) (in Japanese)
42. Website of ADVENTURE Project. http://adventure.sys.t.u-tokyo.ac.jp/
43. Yoshimura, S., Shioya, R., Noguchi, H., Miyamura, T.: Advanced general-purpose computational mechanics system for large scale analysis and design. J Comput. Appl. Math. **149**, 279–96 (2002)
44. Yagawa, G., Shioya, R.: Parallel finite elements on a massively parallel computer with domain decomposition. Comput. Syst. Eng. **4**, 495–503 (1993)
45. Miyamura, T.: Incorporation of multipoint constraints into the balancing domain decomposition method and its parallel implementation. Int. J. Numer. Meth. Eng. **69**, 326–46 (2007)
46. Akiba, H., Miyamura, T., Ohsaki, M., Kohiyama, M., Yamashita, T., Hori, M. Kajiwara, K.: Performance of seismic analysis using E-Simulator on K computer. In: Proceedings of the International Conference on Simulation Technology (JSST2013), Tokyo, Japan (2013), Paper No. 53
47. Akiba, H., Hashizume, K., Miyamura, T.: Large scale nonlinear seismic analysis of high-rise office building with T2K and E-Simulator. Supercomputing News **13**(3), 66–78 (2011) (in Japanese)

Seismic Response Simulation of Nuclear Power Plant

Tomonori Yamada and Shinobu Yoshimura

Abstract Recent strong earthquakes attacking some Japanese nuclear power plants such as Niigataken-Chuetsu-Oki earthquake of 6.8 Mw on July 16, 2007 as well as the Great East Japan Earthquake of 9.0 Mw on March 11, 2011 recalled the importance of seismic design of nuclear power plants seriously. Here, the computational mechanics and simulation are only possible approach that gives us the ability to predict complicated future scenarios and to take measures accordingly. In this chapter, we first describe the current simulation methodologies of nuclear power plants under seismic events and their problems. Then we explain the recent advances in seismic simulation of nuclear power plants and discuss the challenges to realize disaster prevention by using cutting-edge simulation technologies.

1 Introduction

The Great East Japan Earthquake (GEJE) centered off the Pacific coast of Tohoku occurred on March 11, 2011. The magnitude was 9.0 Mw and it was the greatest earthquake recorded in the history of Japan. The consequent Tsunami inflicted immense damage on the Pacific coast area near the focal region in the east of Japan. Most of the large industrial plants including the nuclear power plants located along the focal Pacific coast were damaged by the GEJE and the subsequent Tsunami. In case of nuclear power plants, 11 units among 14 units of 4 NPP

T. Yamada (✉)
Research into Artifacts, Center for Engineering (RACE), The University of Tokyo, Chiba, Japan
e-mail: yamada@race.u-tokyo.ac.jp

S. Yoshimura
Department of Systems Innovation, School of Engineering, The University of Tokyo, Tokyo, Japan
e-mail: yoshi@sys.t.u-tokyo.ac.jp; http://save.sys.t.u-tokyo.ac.jp/; http://adventure.sys.t.u-tokyo.ac.jp/

S. Yoshimura et al. (eds.), *High-Performance Computing for Structural Mechanics and Earthquake/Tsunami Engineering*, Springer Tracts in Mechanical Engineering,
DOI 10.1007/978-3-319-21048-3_5

facilities (Onagawa, Fukushima Dai-ichi, Fukushima Dai-ni, and Tokai Dai-ni) located along the Pacific coast were under operation. All these 11 units were safely shutdown with automatic scram systems. At Tokyo Electric Power Company's (TEPCO) Fukushima Dai-ichi nuclear power plant facilities, reactor 1, 2, and 3 under operation were shutdown with automatic scram systems just after the earthquake occurred and the emergency core cooling system started up. However the subsequent Tsunami of huge height far exceeding a design limit caused long-term station blackout (SBO). As a result, the nuclear fuel in the reactor core and the used fuel storage pool were not cooled down sufficiently which led to extremely serious problems, meltdown, failure in containing radioactive materials inside, and emission of radioactive materials outside of the facilities [1]. Even today, almost four years after the accident, the evacuation of the local residents still continues and the shortage of electricity supply due to the halt of all nuclear power plants across the country is a common problem. At the same time, there are nationwide discussions about energy security including continuation or elimination of nuclear power itself. This large-scale nuclear disaster of unprecedented severity and the fact that it happened in Japan, an advanced industrial country well-known for earthquakes, drew worldwide attention.

Before the nuclear disaster happened in Fukushima, nuclear power plants were regarded as large complex industrial structures that were achieved by comprehensive engineering knowledge. It was commonly known that they required high reliability in the manufacturing because of the radioactive materials involved. The severe impact of the GEJE was explained by the subsequent Tsunami of huge height that exceeded TEPCO's assumptions. There was criticism, however, that these assumptions had not been adapted by considering the Jyogan earthquake and associated Tsunami that struck in 869, more than 1000 years ago, in the same area.

Today, over 430 NPPs are under operation to provide electricity in worldwide. At the same time, a huge number of earthquakes occur everywhere in the world. Establishing methodologies of reliable and accurate seismic proof design plays a key role in maintaining safety and stability of the NPPs regardless of operation or not. Recent strong earthquakes attacking some Japanese NPPs such as Niigataken-Chuetsu-Oki (NCO) earthquake of 6.8 Mw on July 16, 2007 as well as GEJE recalled their practical importance seriously. Computational mechanics and simulation are only possible approach that gives us the ability to predict complicated future scenarios and to take measures accordingly. In this chapter, we first describe the current simulation methodologies of nuclear power plant and its problems. Then we explain the recent advances in seismic simulation of nuclear power plants as an example, and finally we discuss the challenges to realize disaster prevention by using simulation technology and give our concluding remarks.

2 Current Simulation Methodology of Nuclear Power Plant Under Seismic Event

2.1 Current Seismic Situation

In the nuclear engineering, validation and verification tests using shaking tables, for example, those conducted at the Tadotsu engineering test facilities in Japan, had been carried out eagerly under the support of governmental funds following high safety requirement [2]. However, those validation and verification tests were only conducted for each component/device of nuclear facilities and for a scaled component because the physical performance of the shaking table has limitations of available weight on the shaking table and the magnitude of excited acceleration. Hence it was impossible to evaluate the earthquake-proof safety of the entire system of the nuclear power plant by the validation and verification tests. Therefore, evaluations of the earthquake-proof safety using simulation to complement validation and verification tests were carried out ever since the first nuclear power plant was built in Japan. The verification of these evaluations based on simulation was confirmed by a cross-checking approach of industrial operators and the government agency.

In recent seismic simulations, the observed largest earthquake on the site and design standard one, which assume possibly largest earthquake are given as input seismic waves. Large important components, such as pressure reactor vessel, regarded to have a coupling effect with the reactor building are embedded into the mass-spring model of the reactor building, as shown in Fig. 1. Then, the simulation that includes a relatively conservative response spectrum analysis method (a superposition of each natural frequency mode and maximum response value) is executed.

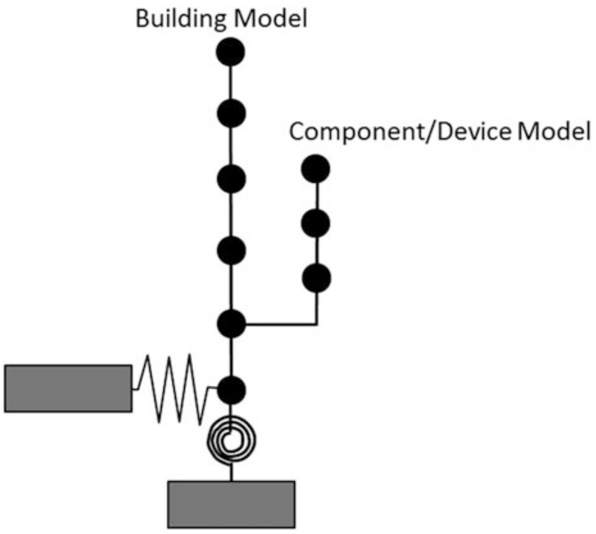


Fig. 1 Mass-spring model

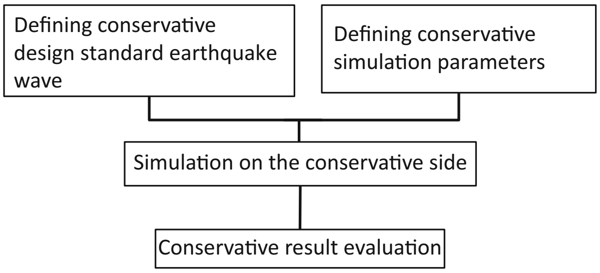

Defining conservative design standard earthquake wave
Defining conservative simulation parameters
Simulation on the conservative side
Conservative result evaluation

Fig. 2 Flow of conservative safety assessment

Furthermore, as part of the evaluation of the simulation results, an assessment is made on the allowable stress value that is based on a larger safety factor. For components whose coupling effect with the reactor building is considered to be low, a vibration simulation using the floor response obtained from the above-mentioned building simulation as input wave is performed. Various parameters used for such simulation and safety assessment, for example, damping coefficients, are defined using more conservative values. In this way, many measures to obtain reliable results on the conservative side have been designed to secure the seismic safety in a reliable way. Figure 2 shows the flow of this conservative safety assessment. The word, "conservative," mentioned here should mean that extremely severe values and criteria are applied for variations and uncertainties of experimental data for the preparations of all kinds of contingency. In the nuclear engineering, the margin of safety factor is much larger than in other industrial sectors. For instance, in the NCO earthquake in 2007, the Kashiwazaki Kariwa nuclear plants were hit by unexpected earthquake motions exceeding a design limit, yet all the functions of "Stopping," "Cooling," and "Confining" [3] could be completed. This fact was regarded as an achievement from the evaluations that relied on conservative safety assessment.

2.2 *Problems in the Current Nuclear Power Seismic Simulations*

In the past, seismic simulations on the conservative side have been emphasized to guarantee the safety even in the supposed worst situations. So, the main purpose of these simulations is that the nuclear power plant is qualitatively safe even in the supposed worst situations. However, the situation worse than the supposed worst situation, for instance, earthquake motion exceeding the design limit (supposed worst situation) at the Kashiwazaki Kariwa nuclear power plants in NCO, requires quantitative safety evaluation. Moreover, conducting simulation to obtain the conservative side evaluations repeatedly does not provide the concrete margin of safety factor and may lead to a belief on safety without confidence. As a result, it can deprive our passion and activities for other disaster prevention. Despite the guidelines that were revised in 2006 which include recommendations on how to reduce the residual risks from earthquakes exceeding design limit, one of the implementation approaches, the so-called "Seismic Probabilistic Safety Assessment (PSA)" was not applied before the GEJE and the accident management for the external events such as earthquakes was not sufficient. After the GEJE, Seismic PSA was renamed into Seismic Probabilistic Risk Assessment (PRA) in Japan for focusing on "risk" more than "safety." Furthermore, the Tsunami, which was considered as one of the reasons for the nuclear disasters, was defined only as one of earthquake associated events before the GEJE. There were no preparations to establish realistic accident scenarios in spite of the fact that a Tsunami's impact is completely different to the accident process of earthquakes.

It is possible to secure the safety for the expected events by current seismic simulations which do not show the concrete margin of safety factor. However, it is difficult to provide quantitative insights for unexpected events and to develop accident scenarios and measures in order to reduce the risks. Since the accident in the nuclear power plants caused by the GEJE, the people in Japan have a sense of danger about unexpected events. As mentioned in the first section, computer simulation is the only methodology to predict complicated future scenarios in order to plan measures for them today. Nevertheless, current simulation technologies can only provide deterministic assumptions on what happened or will happen. Discussions are then necessary to implement appropriate probabilistic measures. Many people in Japan think that this is what is needed for the future.

3 Recent Advances in Seismic Analysis of Nuclear Power Plants

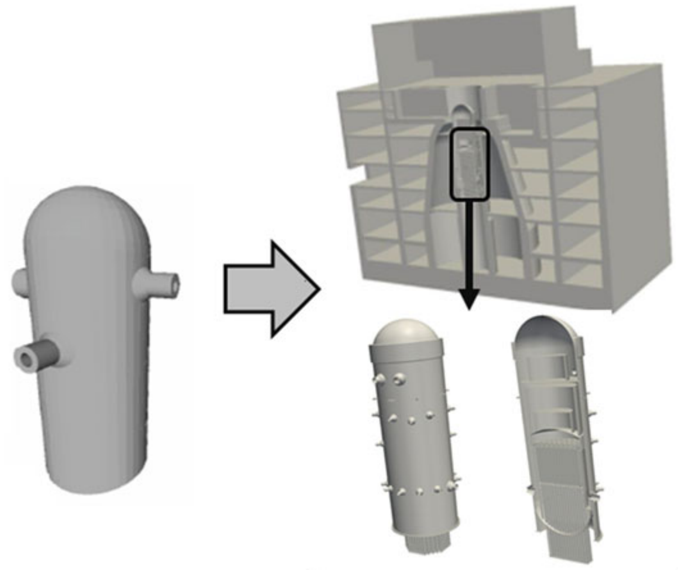
3.1 Current Situation of the Simulation System for Large-Scale Analysis and Design

"Guidelines in seismic design assessment for power generating nuclear reactor facilities" related to the seismic design of nuclear power plant mentioned in the second section, was enacted for the first time in 1978 after the high economic growth period and first installation of nuclear power plant in Japan. It was subsequently revised by adding latest knowledge in various periods. After the GEJE, the regulatory body for nuclear power has been improved and became a body supervised by the Nuclear Regulation Authority in Japan. Then "Rules for standards about components, structure, and equipment of actual power generating nuclear reactors and associated facilities" were enacted in June 2013. In these rules, statements including those for three-dimensional ground structures are incorporated in the "Screening guide for standard earthquake motion and seismic design policy" and in the "Official screening guide for seismic design." Still, more precise definitions and procedures of such seismic events and their simulation are required because the screening guides determine the performance target. With regard to the evaluation procedure, we need to wait for the rules that are defined by the private sector, such as the "Seismic design technical rule for nuclear power plant" by The Japan Electric Association. However from the current problems mentioned in the previous section, we can say that a simulation platform may become necessary, to ensure the quantitative seismic safety assessment as much as possible, despite the existence of uncertainty.

We started the "Development project of a computational mechanics system for large-scale analysis and design" as one of the computational science research activities for the future program of the Japan Society for the Promotion of Science in 1997. (The leader is Prof. Shinobu Yoshimura, one of the authors in this chapter.) Ever since then, we have continued our research and development of a general-purpose parallel computational mechanics system called ADVENTURE [4–8] in order to achieve the most quantitative seismic safety assessment possible. The ADVENTURE system is very unique open source finite element software that enables very precise analyses of practical structures and machines using over 100 million–10 billion DOFs mesh. Those analyses can be very efficiently and easily performed on not only ordinary PC clusters, but also latest supercomputers such as the Earth simulator, Blue Gene/L, and the K-computer (10 peta-flops). The basic parallel solution algorithms employed are the hierarchical domain decomposition method (HDDM) [9] with balancing domain decomposition (BDD) [10] as a preconditioner. In recent years, we have aimed to perform seismic simulations on a peta-scale computing environment by participating in the "Research and development for innovative seismic simulation of large plants including nuclear facilities" of the Strategic Programs for Innovative Research (SPIRE) Industrial Innovation of MEXT (Ministry of Education, Culture, Sports, Science and Technology Japan) [11]. Furthermore, exa-scale systems are expected to be available by the end of this decade [12]. Hence, the computing power is increasing rapidly to make the impossible things possible.

The world's fastest supercomputers in 1978, when the first "Guidelines in seismic design assessment for power generating nuclear reactor facilities" were enacted, were CRAY-1 and X-MP, and their theoretical performance was a few hundred mega-flops. Then in 1997 when we started the "Development project of computational mechanics system for large-scale analysis and design," CP-PACS offered 600 giga-flops (its commercial version Hitachi SR2202 offered 300 giga-flops), and it was the world's best at the time. Along with the growth of computing capability, the simulation models for entire nuclear power plants have evolved from mass-spring models in meters to solid models discretized in centimeters or millimeters.

Figure 3 shows the transition of the simulation model for nuclear power plants and how it improved due to the growing computing power over the past 15 years. The boiling-water reactor (BWR) connected to the building in Fig. 3 is modeled in as much detail as possible including its pressure vessel and inner reactor structures, such as fuel assembly and control rod guide thimble. This is a FEM model holding 240 million DOF totally. Although there are still some simplifications for



1997	**2012**
SR2201 (CP-PACS)	**K Computer**
300GFLOPS	**10PFLOPS**

Fig. 3 The transition of simulation models over the past 15 years

components, the recent model is much better for advanced simulations compared to the traditional mass-spring model. On the other hand, in simulations using detailed models, model simplifications sometimes cause considerable influences on the local response obtained as a result of seismic simulation. So, now we are in a phase where we perform preliminary simulations over and over again, and where we check the responses and modeling methods with the aim to improve our simulation models. Finally, detailed view of our three-dimensional finite element analysis model of BWR [13] is illustrated in Figs. 4 and 5. A snapshot of the Mises stress distributions around the stabilizer of reactor pressure vessel with NCO 2007 wave is shown in Fig. 6.

3.2 Numerical Analysis Technology for Entire Large-Scale Models

The research, development, and maintenance for the general-purpose parallel computational mechanics system ADVENTURE has been continued energetically by industrial-academic project members as an open source project. It was adopted

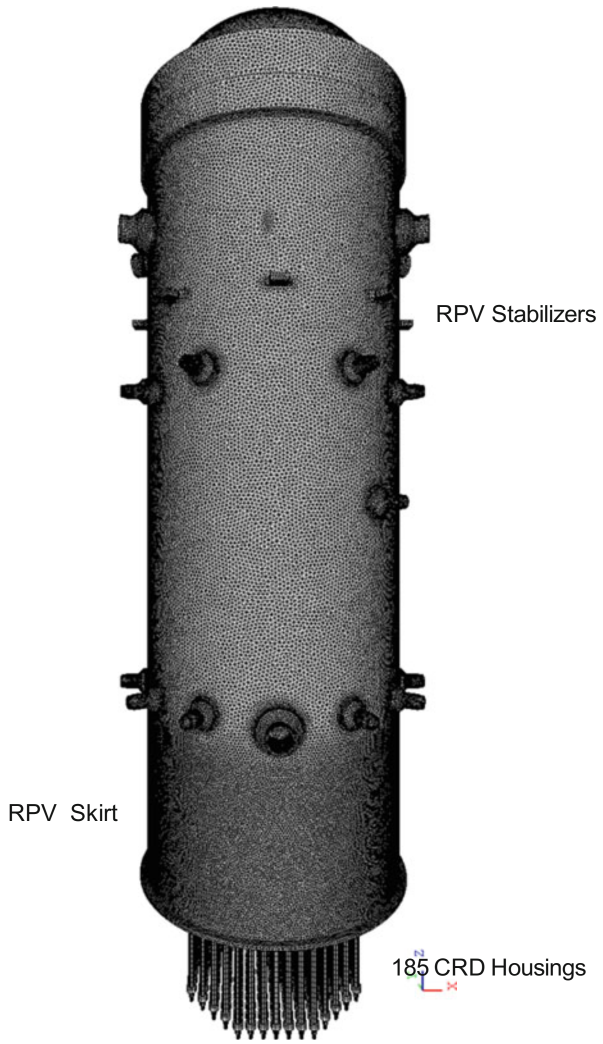


Fig. 4 Finite element discretization of BWR model

as “seismic resistance prediction simulation for nuclear power plants” in the multiphysics simulation field of the Japan Science and Technology Agency’s Strategy Basic Research Programs (CREST type) in March 2013. The main R&D results of this project were released as ADVENTURE_Solid ver.2.0 released in August 2012. The ver.2.0 has a much wider field of application as it implements not only linear elastic analysis based on the traditional HDDM with BDD preconditioner, but also vibration analysis using the same method thus enhancing the pre-post environment. Compared to when we worked on the research for the future program by the Japan Society for the Promotion of Science that was started in 1997, the numerical analysis algorithm has been improved dramatically. For example, the

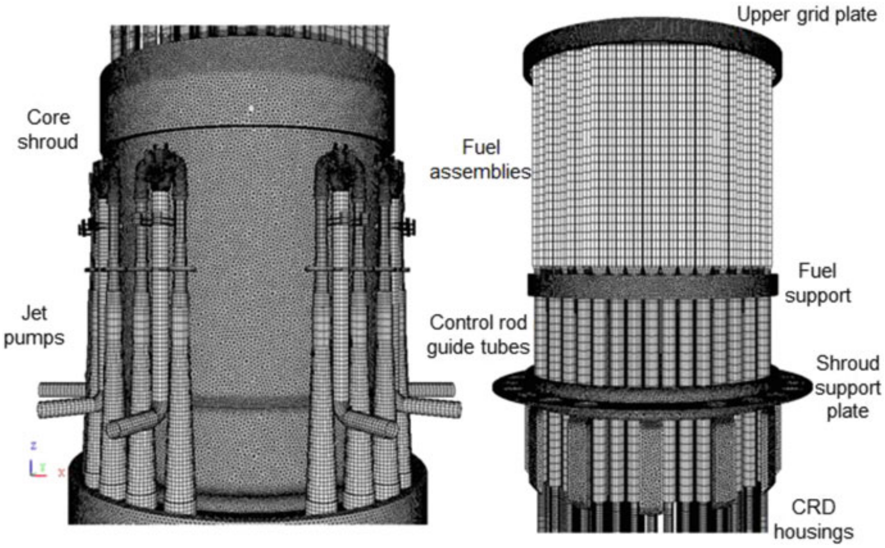


Fig. 5 Core shroud and shroud inside

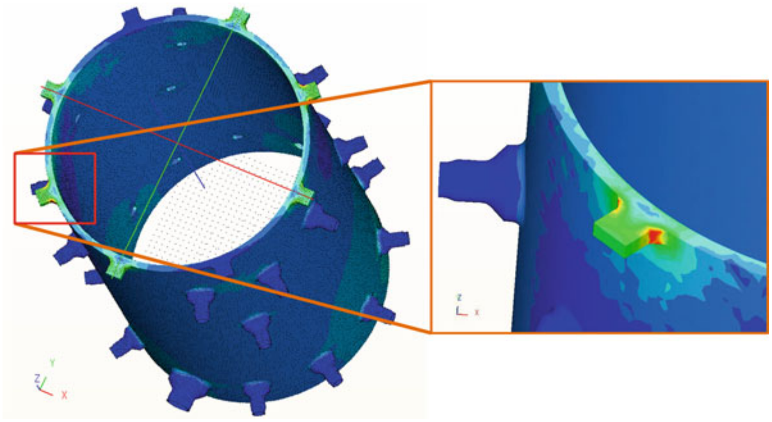


Fig. 6 The Mises stress distributions around the stabilizer of reactor pressure vessel

BDD has been implemented, and the algorithm could be speeded up considerably. Nowadays, we tune it on the K-computer [14, 15] as the SPIRE of MEXT, as mentioned before. As far as the advanced mounting technology for simultaneous linear equations is concerned, we have achieved 40 % better execution efficiency compared to theoretical peak performance when using 4096 nodes (32,768 cores), and 10 % above peak performance when using 32,768 nodes (262,144 cores).

In addition, we have proceeded with R&D for fluid-structure coupled field analysis to take into account the influence of cooling water in reactors. BWR core consists of several hundreds of fuel assemblies. The fuel assemblies are supported with both top guide and fuel support, and are surrounded with coolant water. Since the safety requirements of BWR core are extremely high, various types of experiments and numerical simulations are performed. The quantitative estimation of deflection of a fuel assembly during an earthquake is one of the most important design issues for ensuring the seismic safety of a nuclear reactor because the deflection is the main parameter for assessing control rod scrammability, maintaining core coolable geometry, and also measuring the structural integrity of the fuel assembly itself. For this purpose, we have also continued the development of ADVENTURE_Coupler [16] which ensures the coupled analyses by collaborating with a computational fluid dynamics (CFD) code and a computational structural dynamics (CSD) code with slight additional changes.

Here, fuel assemblies are surrounded by coolant water. Cruciform gaps in fuel assemblies, annular gap between core shroud and the fuel assemblies are filled with water. The coolant water among the fuel assemblies is assumed to be stationary fluid since the flow speed is sufficiently slow compared to the vibration speed of each fuel assembly. Hence, the flow in our problem is regarded as acoustic fluid characterized by having low velocity and no viscosity. The governing equation of the acoustic fluid domain, Ω_F, is Laplace's one and described as follows:

$$\nabla^2 \mathbf{p} = 0 \text{ in } \Omega_F, \tag{1}$$

where $\mathbf{p}$ is the pressure vector in the fluid domain. Assuming the gravity is ignored, the boundary condition on the free surface of the fluid domain is given as:

$$\mathbf{p} = \mathbf{0}. \tag{2}$$

For structure domain, Ω_S, the equilibrium equation of structural motion and deformation is described as follows:

$$\rho_S \frac{\partial^2 \mathbf{u}}{\partial t^2} - \frac{\partial \boldsymbol{\sigma}_S}{\partial \mathbf{x}} = \mathbf{b} \text{ in } \Omega_S, \tag{3}$$

where ρ_S, $\mathbf{u}$, $\boldsymbol{\sigma}_S$ $\mathbf{b}$ are the mass density, the structural displacement vector, the stress tensor, and the body force vector, respectively.

The governing equations of Eqs. (1)–(3) are discretized separately in fluid and structure domains using the 3D finite element method, respectively. A CSD code computes the structural deformation with a given fluid pressure vector, while a computational acoustic fluid dynamics (CAFD) code does a fluid pressure vector with the given structural deformation.

On the interface, Γ_{FSI}, between the fluid and structure domains, the following interaction condition must be satisfied:

$$\frac{\partial \mathbf{p}}{\partial \mathbf{n}} = -\rho_F \mathbf{n}^T \ddot{\mathbf{u}} \text{ on } \Gamma_{\text{FSI}}, \tag{4}$$

where ρ_F and $\mathbf{n}$ are the mass density of fluid and the normal outward vector to the fluid domain. To satisfy the above interaction condition in each time step, fixed-point iteration is introduced.

Let us formulate the nonlinear coupling scheme with fixed-point iteration in each time step, assuming CSD and CAFD codes are separately running. The simulation processes of the AFSI at *i*-th time step are written with acceleration and pressure on the interface in the following:

$$\ddot{\mathbf{u}}^i_{\Gamma_{\mathrm{FSI}}} = \mathrm{CSD}\left(\mathbf{p}^i{}_{\Gamma_{\mathrm{FSI}}}\right), \tag{5}$$

$$\mathbf{p}^i{}_{\Gamma_{\mathrm{FSI}}} = \mathrm{CAFD}\left(\ddot{\mathbf{u}}^i_{\Gamma_{\mathrm{FSI}}}\right), \tag{6}$$

where the superscript i stands for values at i-th time step, while the subscript Γ_{FSI} does for those on the interaction interface. By substituting Eq. (6) into Eq. (5), the nonlinear equation with the interaction condition can be obtained as follows:

$$\ddot{\mathbf{u}}^i_{\Gamma_{\mathrm{FSI}}} = \mathrm{CSD}\left(\mathrm{CAFD}\left(\ddot{\mathbf{u}}^i_{\Gamma_{\mathrm{FSI}}}\right)\right). \tag{7}$$

One of the most popular and robust numerical methods to find the solution of a nonlinear problem is Newton method. However, the Newton method requires the direct evaluation of Jacobian of function, which requires significant modification of existing codes. To treat existing CSD/CAFD codes almost as black boxes, some of the authors have conducted numerical benchmarks with several nonlinear algorithms. Finally, we employ Broyden method [17], which is a family of quasi-Newton methods in this study. In the quasi-Newton methods, the Jacobian of a function does not need to be computed directly, but an approximation of the Jacobian is calculated. To reduce memory requirement, we adopt its restart version [18] instead of the original one. The convergence of the fixed-point iteration is judged with sufficiently small value of tolerance, in terms of Euclid norm of the relative residual, $\mathbf{r}$, of the structural acceleration vector given as follows:

$$\mathbf{r} = \ddot{\mathbf{u}}^i_{\Gamma_{\mathrm{FSI}}} - \mathrm{CSD}\left(\mathrm{CAFD}\left(\ddot{\mathbf{u}}^i_{\Gamma_{\mathrm{FSI}}}\right)\right). \tag{8}$$

In this simulation, we employ our developing parallel finite element codes, i.e., ADVENTURE_Solid for CSD, ADVENTURE_Thermal for CAFD, and ADVENTURE_ Coupler for coupling tool. The detail of the method can be found in [16]

When we combined our CSD and CAFD codes with an iterative partitioned coupling algorithm between the acoustics fluid and structure, a large-scale simulation, such as 8.6 million DOF (5.8 million DOF for the structural fields and 2.8 million DOF for the fluid field, respectively) model including the shroud and the fuel assembly with 368 units as shown in Fig. 7, has become a feasible and efficient solution.

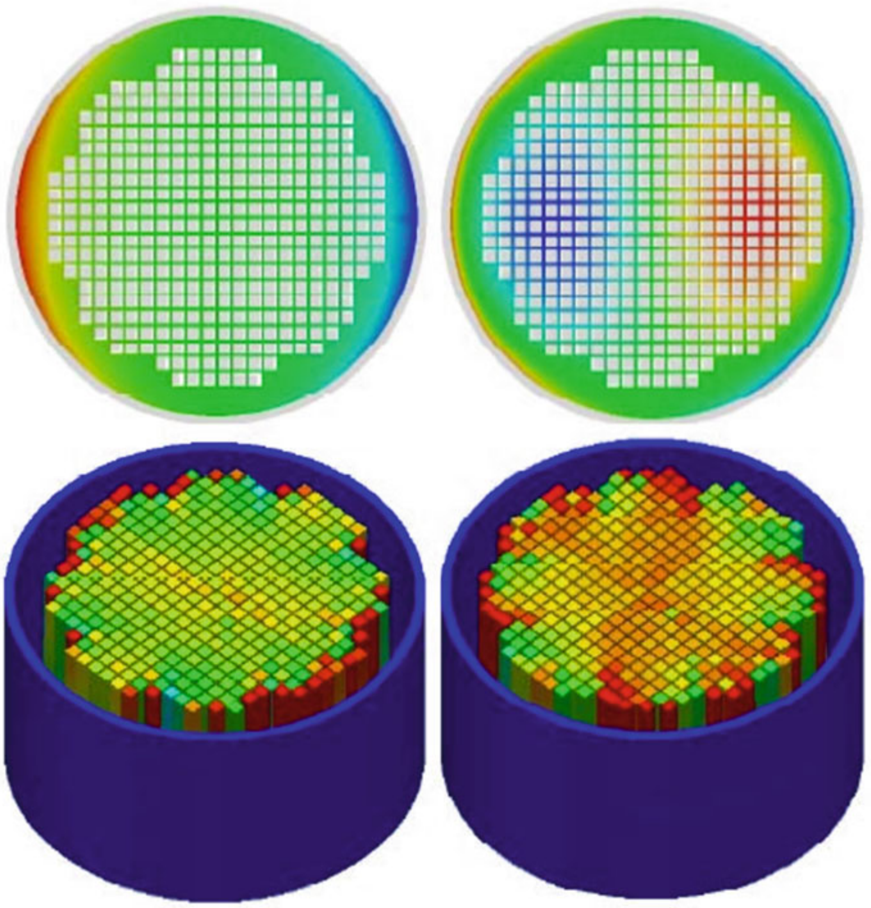


Fig. 7 Multiphysics analysis of fuel assembly (*upper*) pressure distribution of the fluid domain at the location 2.1 m from the bottom of the beam and (*lower*) displacement modes at the times $t = 2.7$ s and $t = 2.77$ s, respectively. Displacements are magnified by factor of 25

4 Future Problems for Seismic Simulations of Nuclear Power Plants

In the past, conservative side evaluations had been assessed in the seismic simulation fields using mass-spring models. However, as we know today, the simulation management to obtain layers of conservative side evaluations is not able to indicate the concrete margin of the safety factor for nuclear power plants. It also cannot take any measures for the quantitative risks of a possible nuclear disaster. The simulation of entire large-scale models has just started for exemplifications. It will take a lot of time until we complete the new technology approach just like with the mass-spring

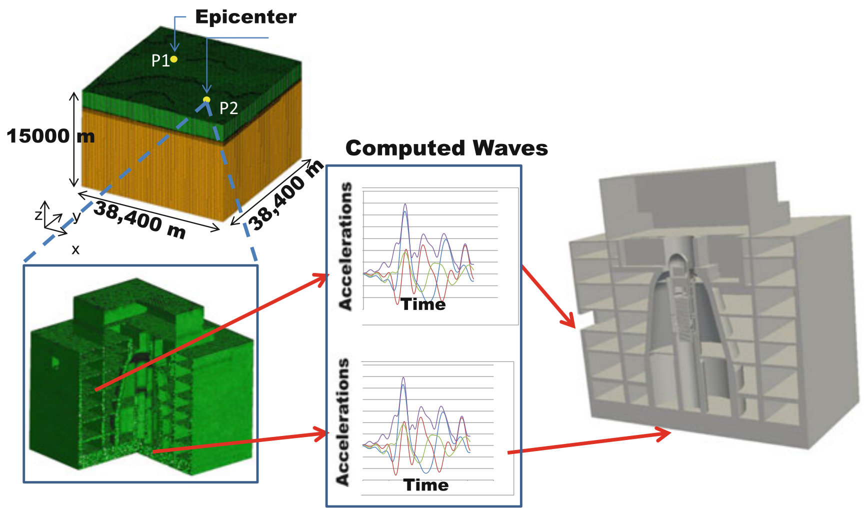


Fig. 8 Collaboration of soil-building simulation and detailed nuclear power plant one

model which has a long history. For example, there are only a few nuclear power plants that possess digital geometrical data by computer aided design (CAD) software which can be used and managed in the manufacturing industry. So the first big hurdle is the improvement of the surrounding environment including creating CAD data from drawings for the entire large-scale model simulation. However, if we can overcome the first hurdle and create detailed models and enhance the simulation model using preliminary simulation and indicate the exemplification studies, we can expect great progress for the entire large-scale model simulation technology. Another problem is the collaboration of seismic simulation of nuclear power plant with other seismic simulations, wider-area simulations. To capture the effect of the earthquake well, wave propagation from epicenter to nuclear power plant must be well evaluated. An illustrative example is given in Fig. 8. Here, the result of the wide range simulation, which includes crust, soil, and the reactor building, is conducted by the techniques written in chapter "Simulation of Seismic Wave Propagation and Amplification" and this result is converted as an input wave in the subsequent seismic simulation of nuclear power plant. Our preliminary study conducted with NCO 2007 configurations. A snapshot of the distribution of Mises stress of the reactor building and reactor pressure vessel and its internals are illustrated in Fig. 9. This approach will contribute the realistic input wave design instead of current conservative one and reveals the real situation inside the nuclear power plant under the seismic events.

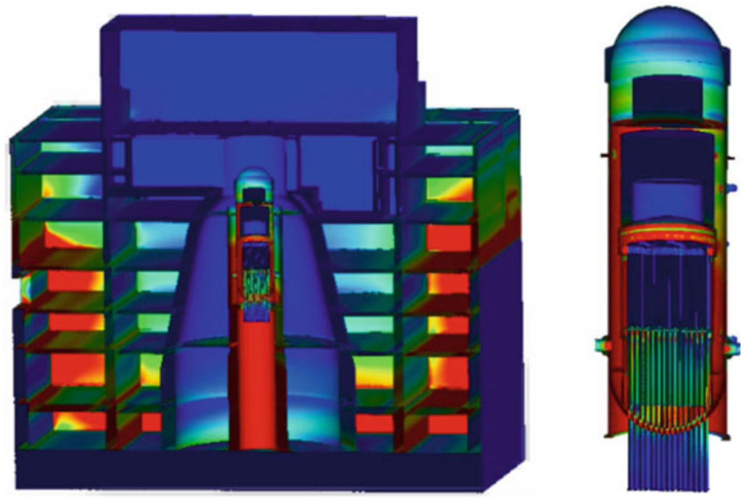


Fig. 9 The Mises stress distributions of the reactor building and the reactor pressure vessel

5 Conclusions

There is a traditional belief in the power of words in Japan, and a culture not to talk about something ominous. This may be one of the reasons why we have avoided to face the risks intentionally. However, the first step of the challenge for disaster prevention and reduction is to face the risks eye-to-eye and not keep them obscure. Although the seismic assessment by the simulation system for entire large-scale models, which we have worked on, is still on the way to its realization, we will continue our R&D to reduce uncertainties and to face disaster risks quantitatively.

Acknowledgments We have received support for our research for the future program, from the computational science field from the Japan Society for the Promotion of Science. From the multiphysics simulation field, the Japan Science and Technology Agency's Strategy Basic Research Programs (CREST type), and the Strategic Programs for Innovative Research (SPIRE) Industrial Innovation of MEXT have supported our work. In addition, we have established a close cooperation and received valuable guidance from the ADVENTURE project members. We wish to express our great appreciation to all those who supported us.

References

1. Narabayashi, T., Sugiyama, K.: Fukushima 1st NPPs accidents and disaster caused by the Pacific coast Tsunami of Tohoku earthquake: lessons from evaluation of the Fukushima 1st NPPs accidents. J. At. Energy Soc. Jpn. **53**(6), 387–400 (2011) (in Japanese)
2. Mouri, Y.: Tadotsu shaking table established the nuclear history. J. At. Energy Soc. Jpn. **52**(11), 715–718 (2010) (in Japanese)

3. Miyano, H., Takagi, T., Sakai, S.: The research committee of Chuetsu-Oki earthquake influences to Kashiwazaki-Kariwa nuclear power station: seismic margin on nuclear power station design. Trans. JSME, B **75**(751), 450–452 (2009) (in Japanese)
4. http://adventure.sys.t.u-tokyo.ac.jp/
5. Yoshimura, S., Shioya, R., Noguchi, H., Miyamura, T.: Advanced general-purpose computational mechanics system for large-scale analysis and design. J. Comput. Appl. Math. **49**, 279–296 (2000)
6. Ogino, M., Shioya, R., Kawai, H., Yoshimura, S.: Seismic response analysis of full scale nuclear vessel model with ADVENTURE system on the Earth simulator. J. Earth Simul. **2**, 41–54 (2005)
7. Yoshimura, S.: Virtual demonstration tests of large-scale and complex artifacts using an open source parallel CAE system ADVENTURE. J. Solid State Phenom. **110**, 133–142 (2006)
8. Murotani, K., Sugimoto, S., Kawai, H., Yoshimura, S.: Hierarchical domain decomposition with parallel mesh refinement for billions-of-DOF scale finite element analyses. Int. J. Comput. Methods **11**, 1350061[30 pages] (2014)
9. Yagawa, G., Shioya, R.: Parallel finite elements on a massively parallel computer with domain decomposition. Comput. Syst. Eng. **4**, 495–503 (1993)
10. Mandel, J.: Balancing domain decomposition. Commun. Numer. Meth. Eng. **9**, 233–241 (1993)
11. http://www.ciss.iis.u-tokyo.ac.jp/supercomputer/
12. Dongarra, J., et al.: The international exascale software roadmap. Int. J. High Perform. Comput. Appl. **25**, 3–60 (2011)
13. Yoshimura, S., Kobayashi, K., Akiba, H., Suzuki, S., Ogino, M.: Seismic response analysis of full-scale boiling water reactor using three-dimensional finite element method. J. Nucl. Sci. Technol. **52**(4), 546–567 (2015). doi: 10.1080/00223131.2014.963000
14. Kawai, H., Ogino, M., Shioya, R., Yoshimura, S.: Large-scale linear dynamic analysis based on domain decomposition method using local Schur complement and inverse of coarse matrix. In: Proceedings of the 5th APCOM and 4th ISCM, USB-Memory (2013)
15. Ogino, M., Shioya, R.: Scalable non-overlapping domain decomposition method for finite element simulations with 100 billion degrees of freedom model. In: Proceedings of the 1st International Conference on Computational Engineering and Science for Safety and Environmental Problems, pp. 96–99 (2014)
16. Kataoka, S., Minami, S., Kawai, H., Yamada, T., Yoshimura, S.: A parallel iterative partitioned coupling analysis system for large-scale acoustic fluid-structure interactions. Comput. Mech. **53**(6), 1299–1310 (2014)
17. Broyden, C.G.: Quasi-Newton methods and their application to function minimization. Math. Comput. **21**(99), 368–381 (1967)
18. Kelley, C.T.: Solving Nonlinear Equations with Newton’s Method. Philadelphia, SIAM (2003)

Tsunami Run-Up Simulation

Kohei Murotani, Seiichi Koshizuka, Eiichi Nagai, Toshimitsu Fujisawa, Akira Anju, Hiroshi Kanayama, Satoshi Tanaka, and Kyoko Hasegawa

Abstract Ishinomaki city was severely damaged by the tsunami of the Great East Japan Earthquake on 2011. Our target is to simulate impact by the tsunami run-up with many floating objects on coastal areas and an urban area of Ishinomaki. Zoom-up analysis by three analyses stages is adopted to solve a large area from an epicenter to an urban area. In the first stage, the two-dimensional shallow-water analysis is solved from the epicenter to the coastal areas. In the second and third stages, the three-dimensional tsunami run-up analyses are solved for the coastal areas using the hierarchical domain decomposition explicit moving particle simulation (MPS) method. Since our target is the tsunami run-up analysis in the urban area in the third stage, we performed three kinds of tsunami analyses. The first analysis was the two tanks floating between buildings. The second analysis was the analysis of 431 floating objects. The third analysis was the elastic analysis for buildings by fluid pressure.

1 Introduction

The coastal area of the Tohoku region suffered serious damage from the tsunami caused by the Great East Japan Earthquake on March 11, 2011 [1]. Today, numerical simulations are being developed to predict potential damage and to perform disaster

K. Murotani (✉) • S. Koshizuka
Department of Systems Innovation, School of Engineering, The University of Tokyo, 7-3-1 Hongo, Bunkyo, Tokyo 113-8656, Japan
e-mail: muro@sys.t.u-tokyo.ac.jp; koshizuka@sys.t.u-tokyo.ac.jp

E. Nagai • T. Fujisawa
Prometech Software, Inc., Tokyo, Japan

A. Anju
Kozo Keikaku Engineering Inc., Tokyo, Japan

H. Kanayama
Department of Mathematical and Physical Sciences, Japan Women's University, Tokyo, Japan

S. Tanaka • K. Hasegawa
Department of Media Technology, Ritsumeikan University, Shiga, Japan

S. Yoshimura et al. (eds.), *High-Performance Computing for Structural Mechanics and Earthquake/Tsunami Engineering*, Springer Tracts in Mechanical Engineering, DOI 10.1007/978-3-319-21048-3_6

prevention measures for such tsunami disasters. From the results of numerical simulations, we can not only design disaster prevention facilities such as coastal breakwaters and levees, but also predict tsunami attacks immediately after an earthquake occurs [2, 3]. The Central Disaster Prevention Council [4] published a basic disaster prevention plan and participates in the determination of important disaster prevention matters. If Tonankai-Nankai earthquake occurs, as predicted by earthquake cycles, it could cause considerable damage. Consequently, the council has performed numerical simulations to predict the wave height and the arrival time of tsunami on the coastal areas.

Today, in most cases, a shallow-water analysis is used in the numerical simulation of tsunami propagation from open sea to coastal areas [5–13]. The shallow-water equations are derived from Navier–Stokes equations and are based on the assumption of hydrostatic pressure in the direction of gravity. Therefore, the shallow-water analysis becomes a two-dimensional analysis that can solve the problem of a very large area of more than 1000×1000 km area. However, the waveform of a tsunami changes significantly in the coastal area compared with the open sea, because water depth becomes shallow and wave height becomes large when the tsunami reaches coastal areas. Additionally, the wave height is amplified due to the complicated shapes of bays and the topography of the ocean floor. For these reasons, it is difficult to solve a shallow-water analysis for the area from sea to dry land. Therefore, solutions of the shallow-water analysis at approximately 1.5 km off a bay shore are used in this research.

We adopted the moving particle simulation (MPS) method [14, 15] to analyze the area from sea to dry land. The MPS method is a popular particle method. Particle methods regard a set of particles as a continuum, discretize physical laws governed by differential equations for interactions between particles, and calculate the states and motions of particles. Since the particles, which are the calculation points, move by time-marching processes, the particle method is superior to grid methods in terms of solving dynamic physical phenomena such as free surfaces and large deformations. However, the motion of particles makes it difficult to parallelize the particle method in distributed memory parallel computers.

Iribe et al. [16] developed the distributed memory parallelization of the semi-implicit MPS method by using a PC cluster. They adopted a sliced grid method for domain decomposition and concentrated their efforts on improving the renumbering of particles to reduce communication volumes for solving the Poisson equation. Though Iribe et al. could increase the communication speed to a certain extent, their results were not adequate for an analysis with more than 16 processing elements. Iribe et al. were able to perform a tsunami simulation of 60 s with 6.3 million particles and 4 processing elements in 237 h. Marrone et al. [17] and Leffe et al. [18] developed distributed memory parallelization of the smoothed particle hydrodynamics (SPH). Marrone et al. also simulated the wave pattern generated by a ship by using hundreds of millions of particles. Leffe et al. [18] performed a simulation of the lifeboat impact in still water by using 100 millions of particles. In

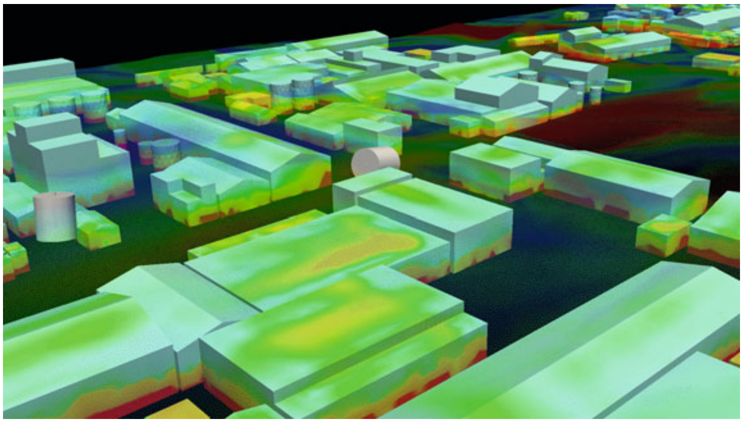


Fig. 1 Fluid velocity and equivalent stress of buildings due to tsunami on Ishinomaki urban area

addition, Marrone et al. and Leffe et al. achieved high parallel efficiency by using the SPH of an explicit algorithm.

In this research, the hierarchical domain decomposition explicit MPS method [19, 20] is used to solve the area from sea to dry land. If equally spaced particles are used from a bay area to an urban area, a large number of particles would be required. In this research, we target analyses of tens of millions to hundreds of millions of particles at maximum due to computational resource issues. Therefore, we performed a zoom-up analysis consisting of the following three analysis stages. In the first stage, the shallow-water analysis is solved for approximately a 1000×1000 km area from epicenter to the coastal areas. The second and third stages of the tsunami run-up analyses are solved for coastal areas totaling less than 10×10 km area and an urban area of approximately 500×500 m area, respectively, by using the hierarchical domain decomposition explicit MPS method. Since our target is tsunami run-up analyses in the urban area of the third stage, we performed a floating analysis of many floating objects and an elastic analysis of buildings by employing the fluid pressure in the urban area, as shown in Fig. 1.

2 Explicit MPS Method

Shakibaenia et al. [21] proposed the explicit MPS method to analyze a weakly compressible flow with free surface. The equation of state for the pressure calculation is solved explicitly. The introduction of artificially weak compressibility under a low Mach number (0.1–0.2) enables high-speed explicit calculation of pressure and good approximation of incompressibility. The governing equations are the equation of state and the Navier–Stokes equations for incompressible flow. These are expressed as

$$\frac{\partial P}{\partial \rho} = c^2, \tag{1}$$

$$\frac{D\vec{v}}{Dt} = -\frac{1}{\rho}\nabla P + \nu \Delta \vec{v} + \vec{g}, \tag{2}$$

where $P, \rho, c, \vec{v}, \nu$, and $\vec{g}$ are the pressure, density, sound speed, velocity, kinematic viscosity coefficient of fluid, and gravity, respectively. The explicit MPS algorithm discretized for governing Eqs. (1) and (2) is summarized below:

$$\vec{v}^{\,*} = \vec{v}^{\,k} + \Delta t \left(\nu \left[\Delta \vec{v} \right]^k + \vec{g} \right), \tag{3}$$

$$\vec{r}^{\,*} = \vec{r}^{\,k} + \Delta t \, \vec{v}^{\,*}, \tag{4}$$

$$P_i^{k+1} = c^2 \rho \frac{n_i^* - n^0}{n^0}, \tag{5}$$

$$\vec{v}^{\,\prime} = -\frac{\Delta t}{\rho} [\nabla P]^{k+1}, \tag{6}$$

$$\vec{v}^{\,k+1} = \vec{v}^{\,*} + \vec{v}^{\,\prime}, \tag{7}$$

$$\vec{r}^{\,k+1} = \vec{r}^{\,*} + \Delta t \, \vec{v}^{\,\prime}, \tag{8}$$

where $\vec{r}$ is the position of a particle, Δt is the time increment, superscript k is the time step number, superscript $*$ is the temporal value at time t, subscript i is the particle number, n is the number of particles in a unit volume, and n^0 is the number of particles in a unit volume at an initial state. In the MPS method, the governing equations are discretized by replacing the differential operators with approximating discrete operators.

The particle interaction models of the MPS assume that particles within a radius of an interaction domain are uniformly located in a grid. However, since this assumption is not applicable for the random distribution of particles including boundaries, computational accuracy is low, as pointed out by Tamai et al. [22]. Therefore, higher-order spatial derivative schemes such as the moving least-squares particle [23, 24] and the reproducing kernel particle methods [25] have been proposed. Here, we adopt spatial discretization schemes with arbitrary high-order consistency based on a Taylor expansion proposed by Tamai et al. [22].

3 Parallel Algorithm

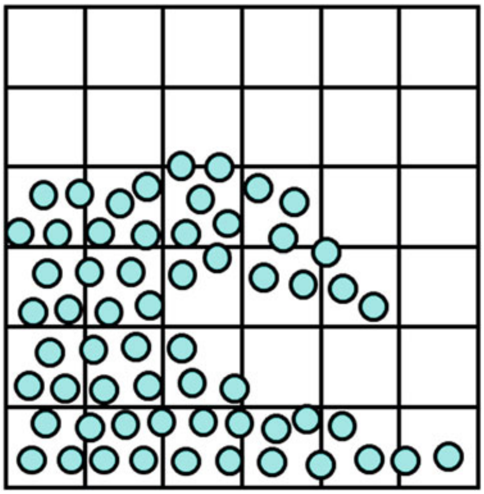
The hierarchical domain decomposition explicit MPS method [19] is described in this section. The hierarchical domain decomposition of two levels and the data management by buckets were introduced in [19].

First, after a bounding box of a whole analysis domain is defined, the bounding box is filled with three-dimensional (or two-dimensional) buckets, as shown in Fig. 2. Since the influence radius of a particle is defined in the MPS method, the width of the bucket must be larger than the influence radius. All particles are assigned to buckets. In Figs. 2, 3, 4, and 5, particles and buckets are expressed as circles and squares, respectively. Next, the bucket-based domain decomposition is performed with an equal number of particles in each processing element (PE), as shown in Fig. 3, where each bucket color represents a subdomain assigned to each processing element.

In this research, domain decomposition is performed by METIS and ParMETIS [26–28]. METIS is a library for partitioning graphs developed by George Karypis at the University of Minnesota. Since METIS is fast and robust, it has been used in a wide variety of applications. ParMETIS is the version for parallel computers. Since a graph consisting of vertices and edges provides the input data for METIS, the subdomain buckets are converted into a graph whose vertices correspond to the elements of the buckets and whose edges correspond to the faces of the buckets. The number of particles in a bucket is assigned to the weight of the vertex of the graph because we want to equalize the number of particles in the subdomains.

Each decomposed subdomain is expanded from the boundaries by one bucket width. The particles in the expanded domains are assigned to one processing element, as shown in Fig. 4. The regions expanded by one bucket width (areas of faint color in Fig. 4) are called “halos.” The consistency of the particle data in a halo is maintained by a halo exchange of communication between neighboring

Fig. 2 Assigning particles to buckets



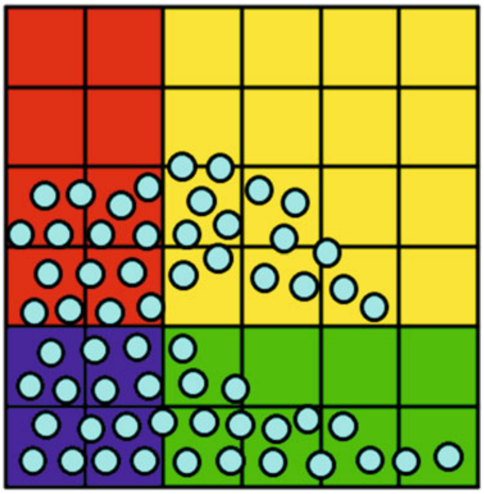


Fig. 3 Domain decomposition for buckets

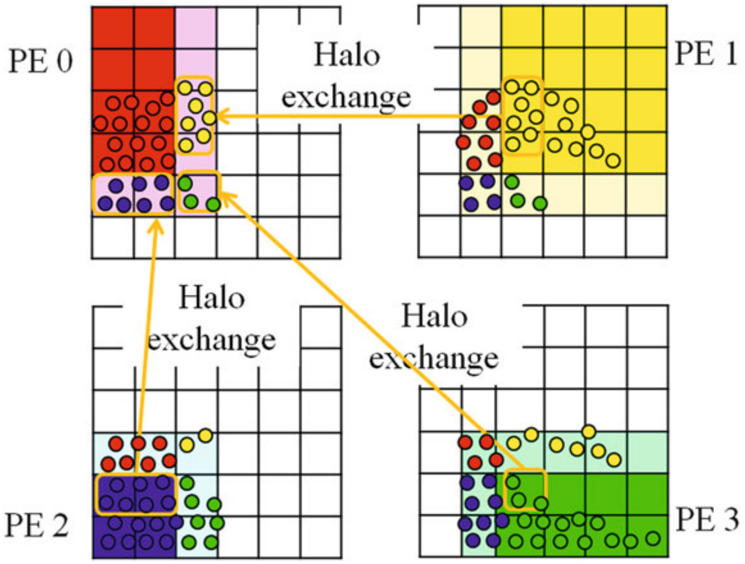


Fig. 4 Halo exchange pattern

processing elements. The arrows in Fig. 4 indicate that the particle data of the other processing elements are sent to the halo of processing element 0. In the distributed parallel explicit MPS algorithm, the halo exchange is performed three times in each time step after Eqs. (4), (5), and (8). If an imbalance in the number of particles between subdomains appears as the analysis progresses, domain decomposition by METIS is performed again to recover the balance of the number of particles.

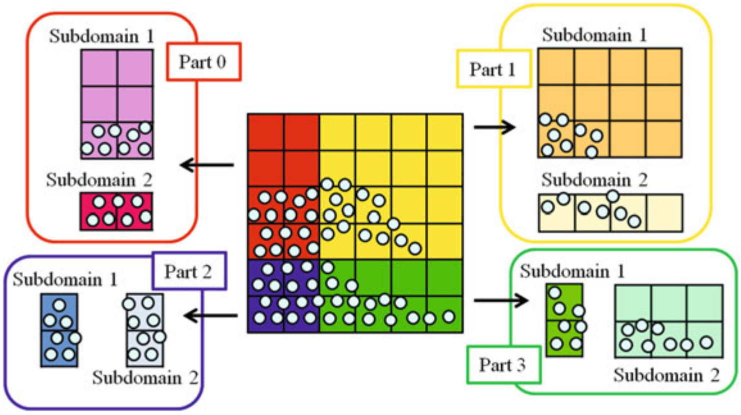


Fig. 5 Hierarchical domain decomposition of two levels

4 Coupled Fluid-Rigid Body Interaction Analysis

In this research, some floating objects flowing by tsunami are analyzed. This analysis is a problem of weak coupling due to the interplay between rigid bodies and fluid. We adopt Koshizuka's method in [15] as the coupled fluid-rigid body interaction algorithm. In the method, rigid bodies are represented by particles with fixed relative configurations, and the rigid body particles and the fluid ones are calculated by the same procedure. Translations and rotations of rigid bodies are calculated from the rigid body particles. Finally, the positions of the rigid body particles are calculated and replaced by the motions of the rigid bodies. This algorithm corresponds to the calculations of the volume integral of forces for rigid bodies (Fig. 6).

If the coupled fluid-rigid body interaction algorithm of [15] is used for distributed memory parallel computing, a method of assigning the rigid body data to processing elements and of communication between processing elements becomes a problem. In this research, we adopted the simple method in which the values of translations, rotations (angular momentum), and updated centers of gravity of all rigid bodies are stored to all processing elements by MPI_Allreduce. The values of translations, rotations, and updated centers of gravity are the variables required in the coupled fluid-rigid body interaction algorithm [15].

In this way of data storage, communication occurs only in summation operations. The summations are done three times to obtain the updated centers of gravity, translation, and rotations, which have three components of the double-precision floating point number, respectively. If communication is done after summation in each node, the communication data size is nine components times the number of rigid bodies at maximum (not the number of rigid body particles). Therefore,

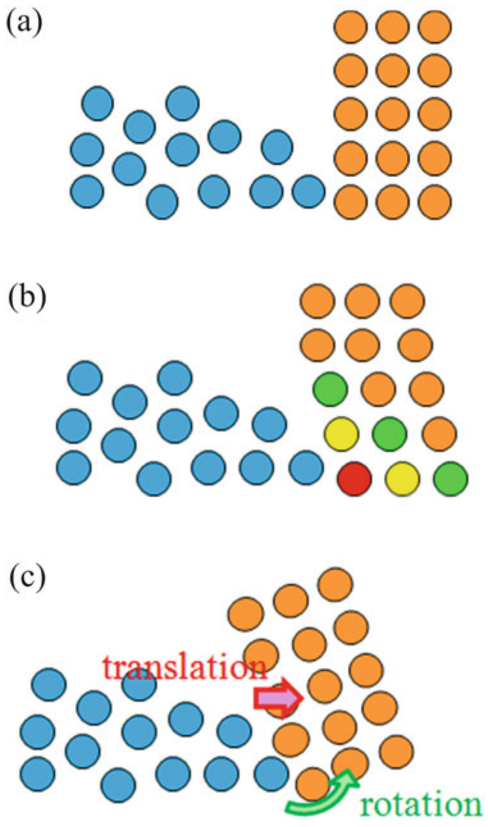


Fig. 6 Coupled fluid-rigid body interaction algorithm in [15]. (**a**) *Blue* particles are fluid and *orange* particles rigid body. (**b**) Rigid body particles and the fluid particles are calculated by the same procedure. (**c**) Replacing the positions of the rigid bodies by the motions of the rigid bodies

the communication data size is very small. For example, for 100 rigid bodies, the communication size is only 8 (size of double) $\times$ 9 $\times$ 100 bytes.

5 HPC Systems Used in This Research

The parallel computers used in this research are the FX10 at the Information Technology Center of the University of Tokyo and the CX400 at the Research Institute for Information Technology of Kyushu University. The FX10 and the CX400 are both made by Fujitsu. The FX10 has 4800 processing elements with one SPARC64 IXfx 1.848-GHz CPU of 16 cores. The CX400 has 1476 processing elements with two Intel Xeon E5-2680 2.7-GHz CPUs of 8 cores.

6 Tsunami Run-Up Analysis for Ishinomaki City

6.1 *Zoom-up Tsunami Analysis in Three Analysis Stages*

In this research, we ran tsunami analyses, such as one on the inundation of the Ishinomaki urban area, through which two 10-m-diameter tanks were carried along by the tsunami resulting from the Great East Japan Earthquake on March 11, 2011. Figure 7 shows the location of the analysis area, Ishinomaki City. Figure 7d shows the installation location and the drift location of one of the tanks. The land surface is generated from 5-m-resolution digital elevation data from the Geospatial Information Authority of Japan. The sea depth is obtained from JTOPO30 (1-km-resolution) and the river depth is assumed as 10 m. The buildings are generated from data prior to the Great East Japan Earthquake.

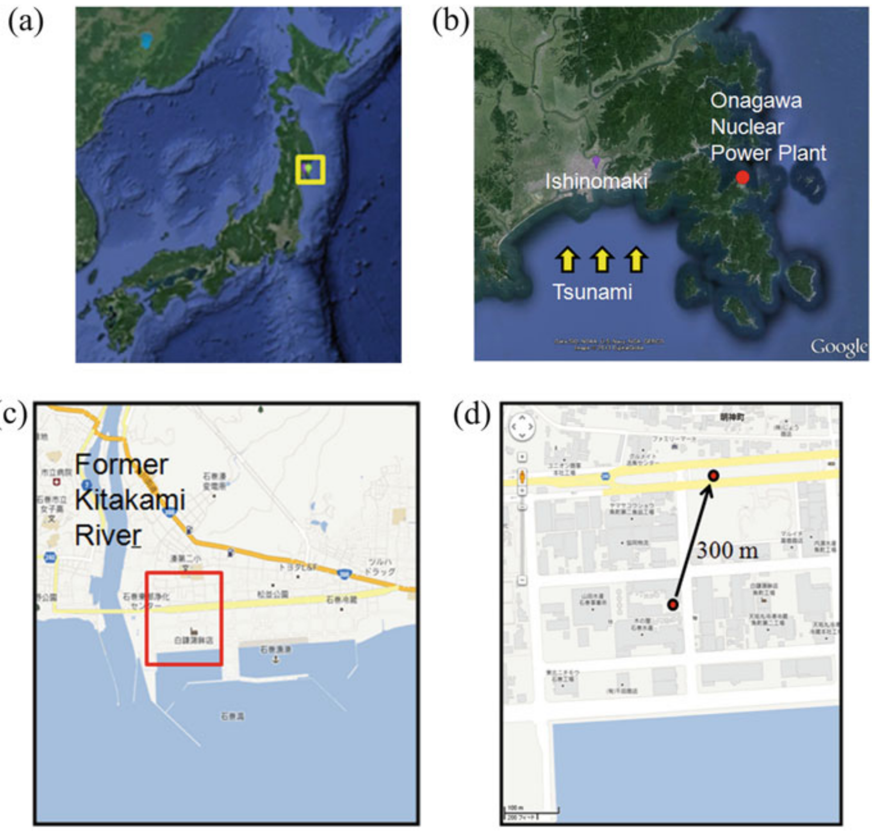


Fig. 7 Location of analysis area (from Google Maps). (**a**) Location of Miyagi prefecture. (**b**) Ojika peninsula. (**c**) Ishinomaki bay. (**d**) Trajectory of floating tanks in the Ishinomaki urban area

6.2 1st-Stage Analysis from Earthquake Center to Coastal Area by Shallow-Water Analysis

This section describes the shallow-water analysis from the earthquake center to the Ishinomaki coastal area. The analysis was performed by using TSUNAMI-K, which is a commercial software made by Kozo Keikaku Engineering Inc. TSUNAMI-K has a database of 50-m-resolution elevation and sea depth data and fault parameters of many past earthquakes. An analysis mesh of TSUNAMI-K can be converted to a hierarchical adaptive mesh with multi-resolutions from 5 to 1350 m. A time sequence is shown in Fig. 8. The water height and velocity data at desired points can be outputted as the analysis result. Figure 8 shows the results of TSUNAMI-K, which used the analysis result of Fujii et al. [29] as an initial water height. In this section, we discuss two 1st-stage results of TSUNMAI-K, solved by Kokusai Kogyo Co., Ltd.

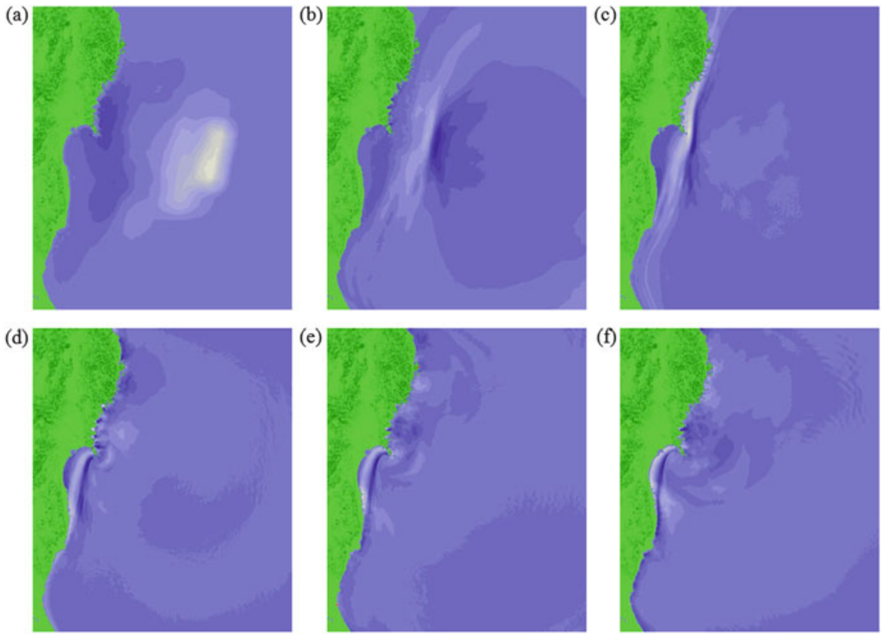


Fig. 8 1st-stage results of shallow-water analysis from earthquake center to coastal area by using TSUNAMI-K. (**a**) 0 s, (**b**) 1000 s, (**c**) 2000 s, (**d**) 3000 s, (**e**) 3500 s, (**f**) 4000 s

6.3 Inflow and Outflow Boundary Conditions in 2nd- and 3rd-Stage Analyses

Inflow and outflow boundary conditions are set on the boundaries of the analysis area shown in Fig. 9. Under the inflow boundary condition, wall particles outside the analysis area are moved toward the inside. If the wall particles come within the analysis area, the wall particles are changed to fluid particles and new wall particles are generated behind the outermost wall particles. Additionally, all particles above the water height on the boundary and all particles on the inside area between the boundary and l_0 are deleted. This procedure produces a constant water height around the boundary.

In the outflow boundary condition, wall particles arranged outside the analysis area keep still (not move), and all the particles above the water height on the boundaries and the area inside of the boundaries by l_0 are deleted. In our analyses, since both anaseism and backwash of the tsunami occur several times, the inflow and outflow are often switched. Since specified boundaries do not always remain in an inflow or outflow state, a simple outflow boundary condition such as this was adopted in order to keep the wall particles neatly arranged.

6.4 2nd-Stage Analysis in Ishinomaki Coastal Area by Particle Method

This section describes the tsunami run-up (2nd-stage) analysis for Ishinomaki Bay and the coastal area. The analysis area was 4.0 × 3.5 km. The initial spacing between particles was 1.0 m. In this analysis, the sound speed 200 m/s was adopted so that

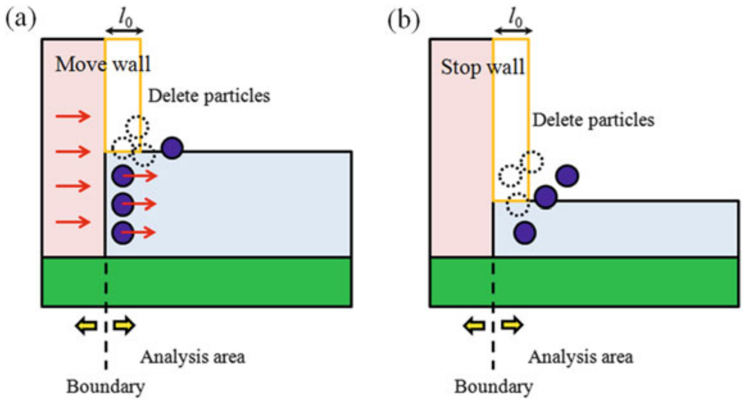


Fig. 9 Inflow and outflow boundary conditions. (**a**) Inflow boundary condition. (**b**) Outflow boundary condition

the Mach number became less than 0.1. The time increment in each time step was determined so that the Courant number became 0.1.

The inflow and outflow data provided by the Tsunami Engineering Laboratory of Tohoku University and Kokusai Kogyo Co., Ltd. were set on the bottom (south) side of Ishinomaki Bay and the coastal area in Fig. 10. The inflow and outflow data are the result of a simulation of the Great East Japan Earthquake obtained by solving a shallow-water equation. The particle wall was put on the other side of Ishinomaki Bay and the coastal area in Fig. 10. In the analysis, the standard Dirichlet and Neumann boundary conditions of pressure were explicitly set for the free surface and the others, respectively.

About 7 days for 144 processing elements of the FX10 at the University of Tokyo were required for this analysis of 800 s, which was set 3200 s after the earthquake occurrence. The maximum number of particles was 260 million. The wall-clock time of one time step was an average of 5.3 s. An average time increment was 0.0072 s. The number of time steps was about 110,000. The wall-clock time of domain decomposition was on average 60.0 s. Figure 10 is the result that transparency over the sea and river is increased and transparency over dry land is decreased.

Since the tsunami caused by the Great East Japan Earthquake came from the south of the Ojika Peninsula, the tsunami struck Ishinomaki Bay southeast to northwest, as shown in Fig. 7b. On the other hand, we know that the tanks were carried from the southwest to the northeast in the Ishinomaki urban area shown in Fig. 7d. The first reason for this is that the tsunami was turned east by the

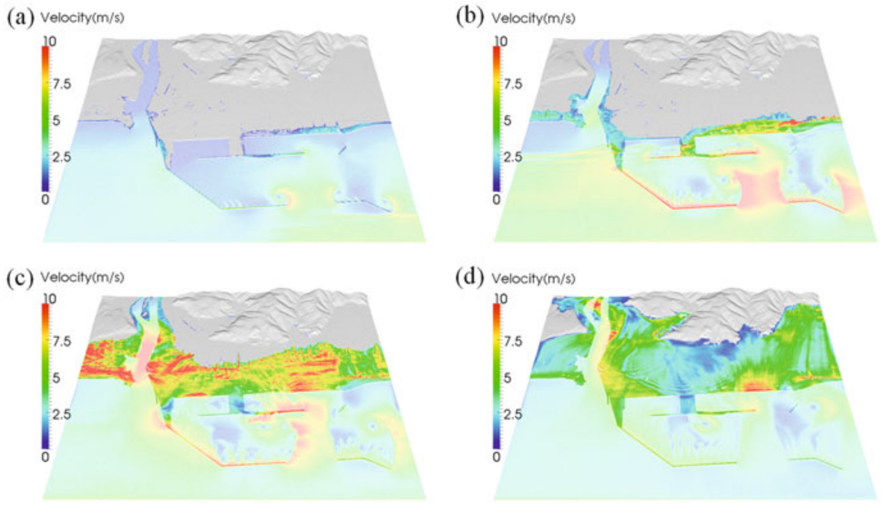


Fig. 10 Tsunami run-up analyses for Ishinomaki Bay and the coastal area 3200 s after the earthquake occurrence. (**a**) 200 s (+3200 s), (**b**) 300 s (+3200 s), (**c**) 400 s (+3200 s), (**d**) 600 s (+3200 s)

breakwaters in Ishinomaki Bay. The second reason is that the tsunami was going upstream along the former Kitakami River, and consequently the river overflowed. These phenomena were simulated and the results are shown in Fig. 10c, d. In this analysis, we succeeded in demonstrating the flow direction of the tsunami in the Ishinomaki urban area by the detailed three-dimensional land surface data.

Next, we performed a tsunami run-up analysis of another situation for the 2nd-stage analysis. The analysis area of 10.0×10.5 km was set to be a larger area than that in the analysis shown in Fig. 10. The initial spacing between particles was 2.0 m. The inflow and outflow data generated by TSUNAMI-K shown in Fig. 8 are set on the bottom (south) side of Ishinomaki Bay and the coastal area in Fig. 11.

About 3 days for 120 processing elements of the FX10 at the University of Tokyo were required for this analysis of 2000 s at 3200 s after the earthquake occurrence. The maximum number of particles was 130 million. The wall-clock time of one time step was an average of 2.4 s. An average time increment was 0.018 s. The number of time steps was about 108,000. The wall-clock time of domain decomposition was on average 24.2 s. Figure 11 is a visualization made with transparency points for fluid particles and surfaces for structural objects such as tanks and buildings by using ParaView.

The appearance of run-up and overflow of the tsunami along the former Kitakami River in Fig. 11 is shown more clearly than in the analysis shown in Fig. 10. The yellow and red lines in Fig. 11f indicate 0 and 2 m, respectively, in the real inundated

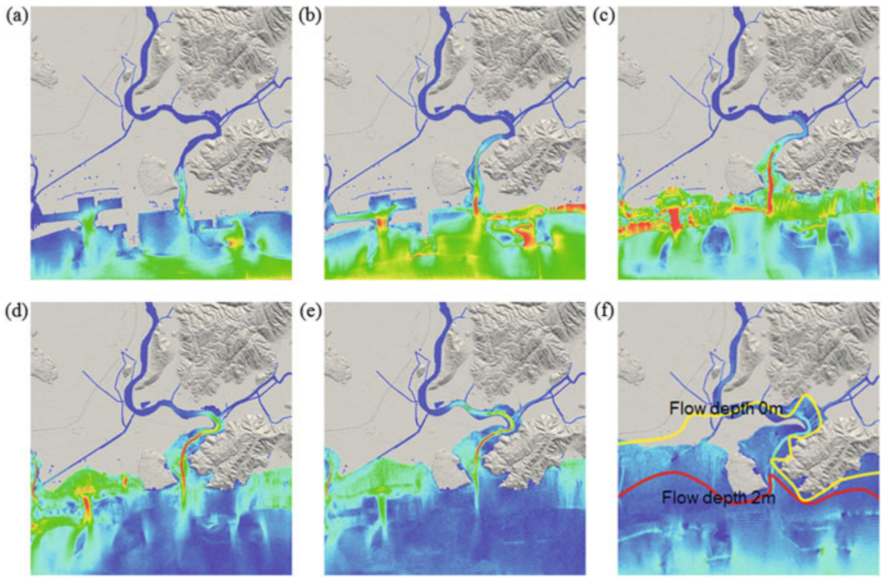


Fig. 11 Tsunami run-up analysis for Ishinomaki Bay and the coastal area 3200 s after the earthquake occurrence. (**a**) 200 s (+3200 s), (**b**) 350 s (+3200 s), (**c**) 500 (+3200 s), (**d**) 650 s (+3200 s), (**e**) 800 s (+3200 s) (**f**) 2000 (+3200 s)

areas on March 11, 2011 measured by the government. The inundation area shown by the numerical simulation is smaller than the real inundation area in Fig. 11f, because the particle spacing 2 m is too large for the resolution required in this analysis.

6.5 3rd-Stage Analysis in Ishinomaki Urban Area by Particle Method

6.5.1 Tsunami Run-Up Analysis with Two Floating Objects

We performed a zoom-up analysis between Ishinomaki Bay and the coastal area (4.0 × 3.5 km) and the Ishinomaki urban area (400 × 550 m). In this section, the event in which two 10-m-diameter tanks were carried along by the tsunami is simulated. The two tanks were regarded as rigid bodies. Since a fragmentation phenomenon is not simulated in our research, only the restraints of the two tanks were released at 60 s. The boundary conditions for the Ishinomaki urban area in Fig. 12b are generated along the red lines in Fig. 12a for the zoom-up analysis. This analysis used values interpolated from the particle data for grid positions arranged at equal distances on the boundary lines of the Ishinomaki urban area in Fig. 12b as the inflow and outflow conditions.

The analysis area was 550 × 400 m, and the initial spacing between particles was 0.2 m. In this analysis, the sound speed 200 m/s was adopted so that the Mach number became less than 0.1. The time increment in each time step was determined so that the Courant number became 0.1.

About 30 days and 32 processing elements of the CX400 at Kyushu University were required for this analysis of 200 s at 3540 s after the earthquake occurrence. The maximum number of particles reached 390 million particles. The wall-clock time of one time step was an average of 11.7 s. An average time increment was 0.0009 s. The number of time steps was about 221,000. The wall-clock time of

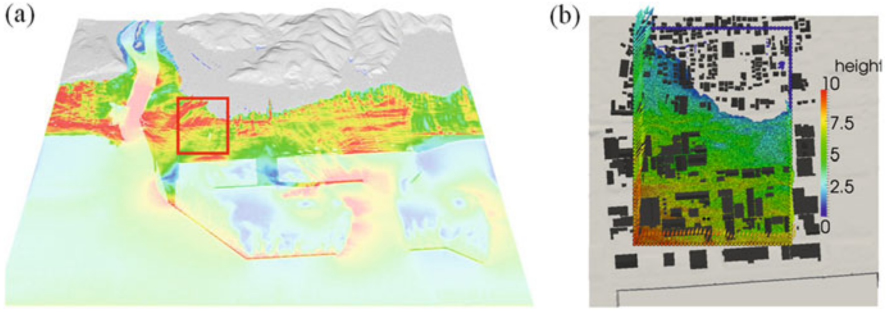


Fig. 12 Location of Ishinomaki urban area. (**a**) Location of Ishinomaki urban area. (**b**) Extraction of boundaries

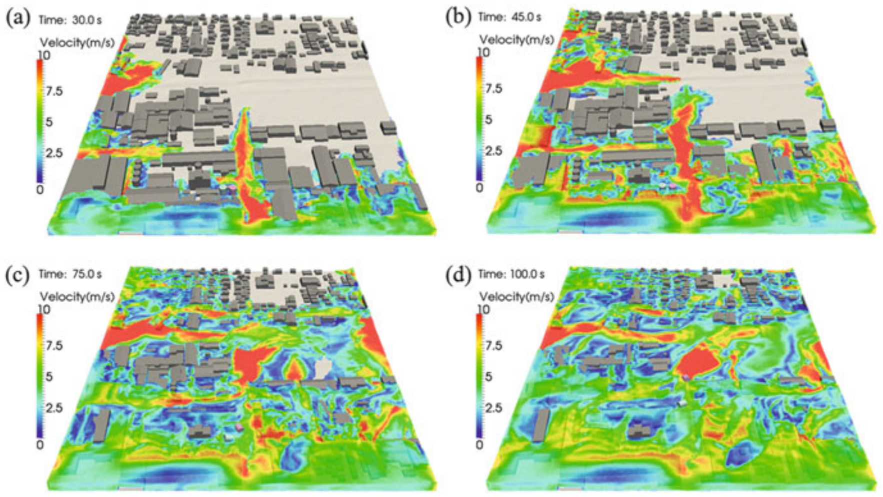


Fig. 13 Tsunami run-up analyses for Ishinomaki urban area 3540 s after the earthquake occurrence. (**a**) 30 s (+3540 s), (**b**) 45 s (+3540 s), (**c**) 75 s (+3540 s), (**d**) 100 s (+3540 s)

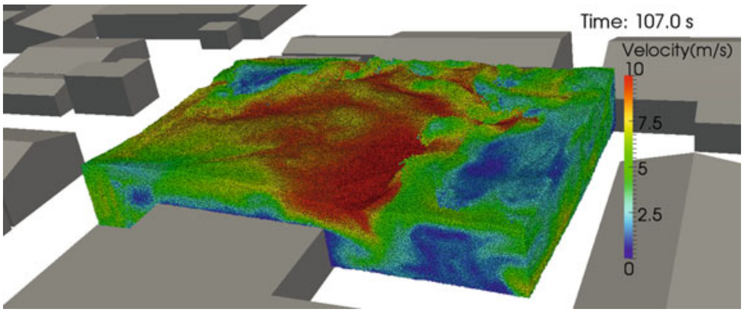


Fig. 14 Sectional side view by opacity point sprite

the domain decomposition was on average 62 s. Figure 13 is a visualization using ParaView. In this visualization, transparency points represent fluid particles and opacity surfaces represent structural objects such as tanks and buildings. Figure 14 is the sectional side view obtained by the opacity point sprite.

From Fig. 13b, we can see that the tsunami was running up along an avenue from south to north and a prefectural road from west to east. Figure 13a–d shows the contracted flow phenomena generated by the rapid flooding by the tsunami around buildings. Figure 13c, d shows that the two tanks are carried along at high speed. The flow direction of the two tanks in Fig. 13c, d is the same as the direction of one tank carried from southwest to northeast on the day of the Great East Japan Earthquake in Fig. 7d. In this analysis, we succeeded in demonstrating the flow direction of the tanks in the Ishinomaki urban area by the use of detailed three-dimensional building data, although the tanks were moving out of the analysis area.

6.5.2 Tsunami Run-Up Analysis with 431 Floating Objects

Using the fluid-rigid body interaction coupling method described in Sect. 4, we performed an imaginary analysis in which 431 floating objects flow and collide with each other in the Ishinomaki urban area. Although it is difficult for grid methods such as FEM and FDM to carry out this analysis, particle methods can do so easily. Additionally, this analysis was successfully executed on a distributed memory parallel supercomputer.

The 431 buildings objects in the Ishinomaki urban model were regarded as rigid bodies. Since a fragmentation phenomenon was not simulated in our research, only the restraints of the 431 buildings were released at 200 s. The analysis area was 660×810 m. The initial spacing between particles was 0.5 m. In this analysis, the sound speed 200 m/s was adopted so that the Mach number became less than 0.1. The time increment in each time step was determined so that the Courant number became 0.1.

About 40 h for 72 processing elements of the CX400 at Kyushu University were required for this analysis of 400 s at 3540 s after the earthquake occurrence. The maximum number of particles was 80 million. The wall-clock time of one time step was on average 1.38 s. An average time increment was 0.0021 s. The number of time steps was about 187,000. The wall-clock time of domain decomposition was on average 4.0 s (Fig. 15).

6.5.3 Elastic Analysis for Buildings by Fluid Pressure

Using a fluid analysis of the same setting as the tsunami run-up analysis with two floating objects in Sect. 6.5.1, a structural analysis for buildings by the fluid pressure of the tsunami was performed. Figure 16 is a mesh of the Ishinomaki urban area with 10 million elements of 2-m mesh size generated for a model connecting the terrain and buildings. Figure 17 is the result of a domain decomposition of 600 domains using ADVENTURE_Metis for the terrain and building mesh to solve the structural analysis by FEM using ADVENTURE_Solid. Figure 18 is the result of the domain decomposition of 600 domains using ParMETIS for buckets to solve the fluid analysis using the particle method. The colors in Figs. 17 and 18 correspond to the rank number assigned to each domain.

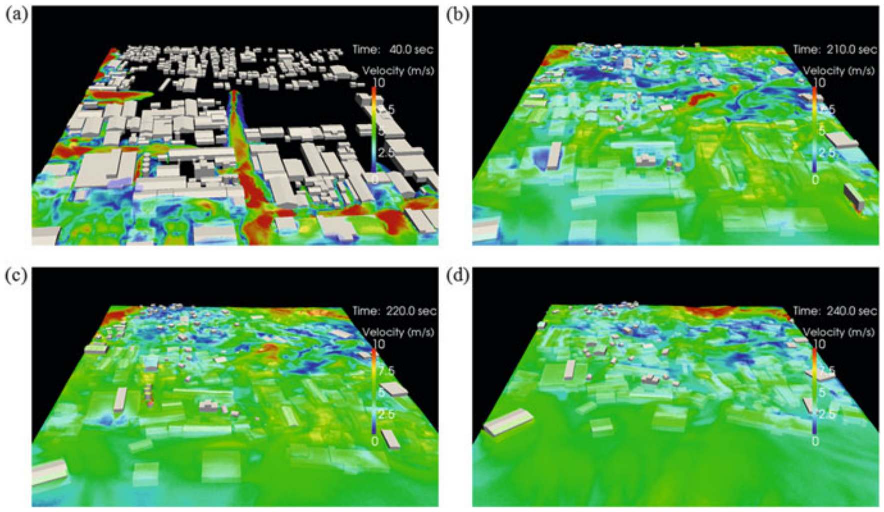


Fig. 15 Tsunami run-up analysis with 431 floating objects for Ishinomaki urban area 3540 s after the earthquake occurrence. (**a**) 40 s (+3540 s), (**b**) 210 s (+3540 s), (**c**) 220 s (+3540 s), (**d**) 240 s (+3540 s)

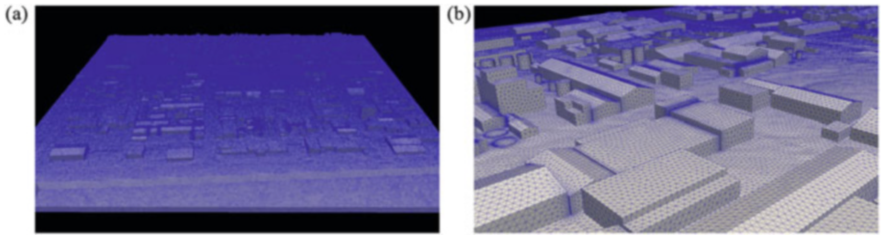


Fig. 16 Mesh of Ishinomaki urban area with 10 million elements of 2-m mesh size. (**a**) Entire Ishinomaki urban area. (**b**) Center of Ishinomaki urban area

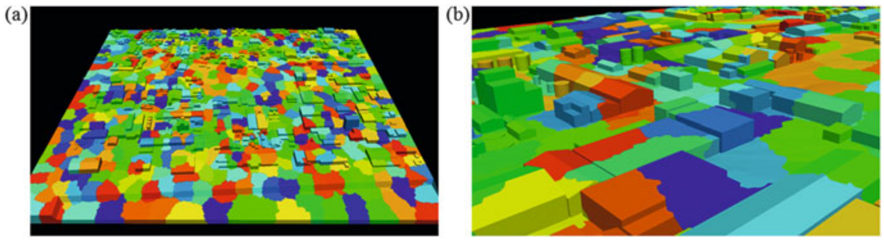


Fig. 17 Domain decomposition of 600 domains for the terrain and building mesh. (**a**) Entire Ishinomaki urban area. (**b**) Center of Ishinomaki urban area

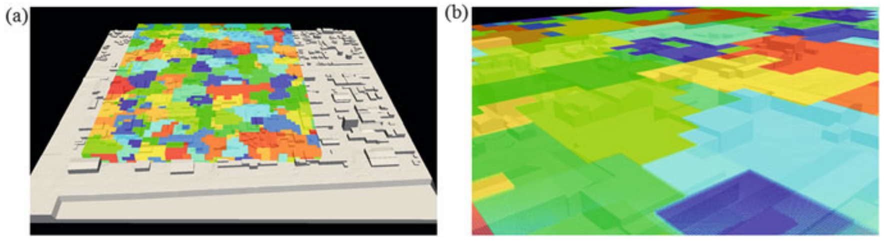


Fig. 18 Domain decomposition of 600 domains for buckets. (**a**) Entire Ishinomaki urban area. (**b**) Center of Ishinomaki urban area

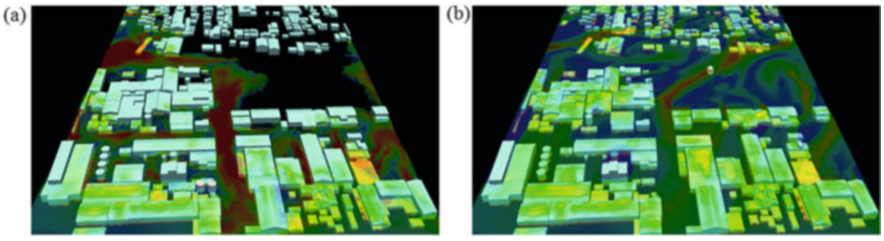


Fig. 19 Fluid velocity and equivalent stress of buildings by tsunami on Ishinomaki urban area 3540 s after the earthquake occurrence. (**a**) 50 s (+3540 s), (**b**) 150 s (+3540 s)

The fluid–structure interaction (FSI) analysis of this section is a one-way coupling analysis from fluid to structure. First, the fluid analysis is completed. The input files of the structural analysis are generated from the output files of the fluid analysis. Then, the structural analysis is started by using the output files of the fluid analysis. The structural analysis is the static elastic analysis. The outside of the mesh receives the fluid pressure of particles and the bottom of the mesh is fixed.

The fluid analysis area was 550×400 m. The initial spacing between particles was 0.5 m. In this analysis, the sound speed 200 m/s was adopted so that the Mach number became less than 0.1. The time increment in each time step was determined so that the Courant number became 0.1. About 6 h for 600 processing elements of the FX10 at the University of Tokyo were required for this analysis at 200 s after 3540 s from the earthquake occurrence. The maximum number of particles was 40 million. The wall-clock time of one time step was on average 0.25 s. An average time increment was 0.0023 s. The number of time steps was about 86,000. The wall-clock time of domain decomposition was on average 2.1 s.

Figures 19 and 20 show the fluid velocity and the equivalent stress of buildings due to the tsunami on the Ishinomaki urban area. We can see that the equivalent stress on buildings occurs by the wave force as the tsunami progresses.

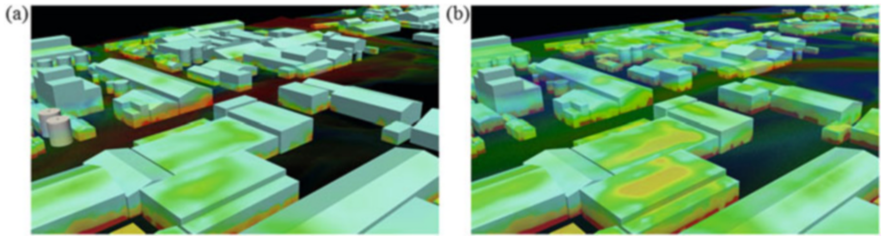


Fig. 20 Fluid velocity and equivalent stress of buildings by tsunami on the center of Ishinomaki urban area 3540 s after the earthquake occurrence. (**a**) 50 s (+3540 s), (**b**) 150 s (+3540 s)

7 Conclusions

In this research, we successfully conducted zoom-up tsunami simulations by using three analysis stages of the Great East Japan Earthquake in Ishinomaki City. Since our target is the tsunami run-up analysis in the urban area in the third stage, we performed three kinds of tsunami analyses. The first analysis was the two tanks floating between buildings. The second analysis was the analysis of 431 floating objects. The third analysis was the elastic analysis for buildings by fluid pressure. It is difficult for grid methods such as FEM and FDM to carry out these analyses. On the other hand, particle methods can easily perform these analyses. Finally, we successfully developed the hierarchical domain decomposition explicit MPS method [19] to solve very large and complicated models such as the Ishinomaki urban model.

Acknowledgments This research was financially supported by JSPS KAKENHI Grant Number 26390127 and JST CREST project "Development of a Numerical Library based on Hierarchical Domain Decomposition for Post Petascale Simulation." The computational resource of the Fujitsu FX10 was awarded in the "Large-scale HPC Challenge" Project, Information Technology Center, the University of Tokyo. This research was supported in part by the results of the HPCI and JHPCN Systems Research Projects (Project IDs hp120232/hp140199/jh130031/14-NA07). The authors also wish to thank Professor Kentaro Imai and Professor Shunichi Koshimura of Tohoku University and Kokusai Kogyo Co., Ltd. for the tsunami inlet boundary conditions, Ishinomaki City for the map data, Prometech Software, Inc. for generation of the Ishinomaki City model, Kozo Keikaku Engineering Inc. for the use of TSUNAMI-K and all the members of the ADVENTURE project for their cooperation.

References

1. National Institute for Land and Infrastructure Management, Ministry of Land, Infrastructure, Transport and Tourism, Japan and Building Research Institute, Incorporated Administrative Agency, Japan: Quick report of the field survey and research on the 2011 off the Pacific Coast of Tohoku Earthquake. Technical Note, No. 636, Building Research Data. No. 132 (2011) (in Japanese)

2. Goto, C., Sato, K.: Development of tsunami numerical simulation system for Sanriku coast in Japan. Rep. Port Harbour Res. Inst. **32**(2), 3–44 (1993) (in Japanese)
3. Takahashi, T.: Application of numerical simulation to tsunami disaster prevention. J. Jpn. Soc. Comput. Fluid Dyn. **12**(2), 23–32 (2004) (in Japanese)
4. The Central Disaster Prevention Council: Special Investigation Council Concerning Tounankai and Nankai Earthquakes (2003) (in Japanese)
5. Ulutas, E.: The 2011 off the Pacific Coast of Tohoku-Oki earthquake and tsunami: influence of the source characteristics on the maximum tsunami heights. In: Proceedings of the International Symposium on Engineering Lessons Learned from the 2011 Great East Japan Earthquake, pp. 602–611 (2012)
6. Imamura, F., Yalciner, A.C., Ozyurt, G.: Tsunami modelling manual (2006), http://www.tsunami.civil.tohoku.ac.jp/hokusai3/J/projects/manual-ver-3.1.pdf. Accessed 27 Mar 2012
7. Kanayama, H., Ushijima, T.: On the viscous shallow-water equations I: derivation and conservation laws. Mem. Numer. Math. **8**(9), 39–64 (1981/1982)
8. Kanayama, H., Ohtsuka, K.: Finite element analysis on the tidal current and COD distribution in Mikawa Bay. Coast. Eng. Jpn **21**, 157–171 (1978)
9. Kanayama, H., Dan, H.: A finite element scheme for two-layer viscous shallow-water equations. Jpn. J. Ind. Appl. Math. **23**(2), 163–191 (2006)
10. Kanayama, H., Dan, H.: Two-layer viscous shallow-water equations and conservation laws. J. Comput. Sci. Technol. **3**(1), 373–384 (2009)
11. Murakami, K.: Calculation of vertical circulation in stratified waters by 2-level and 2-layer models. In: Proceedings of Coastal Engineering, JSCE, vol. 36, pp. 204–208 (1989) (in Japanese)
12. Kanayama, H., Dan, H.: A tsunami simulation of Hakata Bay using the viscous shallow-water equations. Jpn. J. Ind. Appl. Math. **30**(3), 605–624 (2013)
13. Bresch, D.: Shallow-water equations and related topics. In: Dafermos, C., Pokorny, M. (Eds.) Handbook of Differential Equations: Evolutionary Equations, Elsevier, North Holland. vol. 5, pp. 1–104 (2009)
14. Koshizuka, S., Oka, Y.: Moving-particle semi-implicit method for fragmentation of incompressible fluid. Nucl. Sci. Eng. **123**, 421–434 (1996)
15. Koshizuka, S., Nobe, A., Oka, Y.: Numerical analysis of breaking waves using the moving particle semi-implicit method. Int. J. Numer. Meth. Fluids **26**, 751–769 (1998)
16. Iribe, T., Fujisawa, T., Koshizuka, S.: Reduction of communication in parallel computing of particle method for flow simulation of seaside areas. Coast. Eng. J. **52**(4), 287–304 (2010)
17. Marrone, S., Bouscasse, B., Colagrossi, A.: Numerical modeling of ship wave patterns through a hybrid OpenMP/MPI SPH solver. In: 2nd International Conference on Violent Flows, Nantes, France, pp. 221–228 (2012)
18. Leffe, M.D., Guilcher, P.M., Candelier, J., LeTouzé, D., Oger, G., Grenier, N.: SPH for naval applications. In: 2nd International Conference on Violent Flows, Nantes, pp. 229–237, September 2012
19. Murotani, K., Koshizuka, S., Tamai, T., Shibata, K., Mitsume, N., Yoshimura, S., Tanaka, S., Hasegawa, K., Nagai, E., Fujisawa, T.: Development of hierarchical domain decomposition explicit MPS method and application to large-scale tsunami analysis with floating objects. J. Adv. Simul. Sci. Eng. **1**(1), 16–35 (2014)
20. Murotani, K., Oochi, M., Fujisawa, T., Koshizuka, S., Yoshimura, S.: Distributed memory parallel algorithm for explicit MPS using ParMETIS. In: Transaction of JSCES, No. 20120012 (2012) (in Japanese).
21. Shakibaeinia, A., Jin, Y.C.: A weakly compressible MPS method for modeling of open-boundary free-surface flow. Int. J. Numer. Meth. Fluids **63**, 1208–1232 (2010)
22. Tamai, T., Koshizuka, S.: Least squares moving particle semi-implicit method. Comput. Part. Mech. **1**(3), 277–305 (2014)
23. Lancaster, P., Salkauskas, K.: Surfaces generated by moving least squares methods. Math. Comput. **37**(155), 141–158 (1981)

24. Dilts, G.A.: Moving-least-squares-particle hydrodynamics: I. Consistency and stability. Int. J. Numer. Meth. Eng. **44**, 1115–1155 (1999)
25. Liu, W.K., Jun, S., Zhang, Y.F.: Reproducing kernel particle methods. Int. J. Numer. Meth. Fluids **20**, 1081–1106 (2005)
26. Karypis, G., Kumar, V.: A fast and highly quality multilevel scheme for partitioning irregular graphs. SIAM J. Sci. Comput. **20–1**, 359–392 (1999)
27. Karypis, G., Kumar, V.: Multilevel k-way partitioning scheme for irregular graphs. Technical report, Department of Computer Science, University of Minnesota, TR 95-064 (1995)
28. Karypis, G., Kumar, V.: Parallel multilevel k-way partitioning scheme for irregular graphs. Technical Report, Department of Computer Science, University of Minnesota, TR 96-036 (1996)
29. Fujii, Y., Satake, K., Sakai, S., Shinohara, M., Kanazawa, T.: Tsunami source of the 2011 off the Pacific coast of Tohoku earthquake. Earth Planets Space **63**(7), 815–820 (2011)

Inundation Simulation Coupling Free Surface Flow and Structures

Naoto Mitsume, Shinobu Yoshimura, Kohei Murotani, and Tomonori Yamada

Abstract Water-related disasters such as tsunamis, storm surges, and floods involve fluid–structure interaction (FSI) problems with free surface flow. Since failures of artifacts are caused due to inundation, water forces and impact forces by floating objects, simulation of such problems has great importance to design for safety and robustness.

In this chapter, we present a robust and efficient coupled method for fluid–structure interaction with violent free surface flow, named the MPS-FE method and its improved method. The MPS-FE method adopts the finite element (FE) method for structure computation and the moving particle semi-implicit/simulation (MPS) method for fluid computation involving free surface flow. The conventional MPS-FE method, in which MPS wall boundary particles and finite elements are overlapped in order to exchange information at fluid–structure interface, is not versatile and reduces the advantages of software modularity. We developed a non-overlapping approach in which the interface in the fluid computation corresponds to that in the structure computation through an MPS polygon wall model. The accuracy of the improved MPS-FE method was verified by solving a dam break problem with an elastic obstacle and by comparing the result obtained with that of the conventional MPS-FE method and other methods.

1 Introduction

Large facilities such as electric power, energy, and chemical plants built along coastal regions are vulnerable to various water-related disasters such as tsunamis [53], storm surges and floods. The resulting damage to equipments and instruments

N. Mitsume (✉) • S. Yoshimura • K. Murotani
Department of Systems Innovation, The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan
e-mail: mitsume@save.sys.t.u-tokyo.ac.jp; yoshi@sys.t.u-tokyo.ac.jp; muro@sys.t.u-tokyo.ac.jp

T. Yamada
RACE, The University of Tokyo, 5-1-5 Kashiwanoha, Kashiwa-shi, Chiba 277-8568, Japan
e-mail: yamada@race.u-tokyo.ac.jp

S. Yoshimura et al. (eds.), *High-Performance Computing for Structural Mechanics and Earthquake/Tsunami Engineering*, Springer Tracts in Mechanical Engineering,
DOI 10.1007/978-3-319-21048-3_7

has the potential to cause catastrophic harm to people and society. The Fukushima Daiichi nuclear disaster occurred on March 11, 2011, in Japan is a prominent example of such a phenomenon. It is economically impossible to completely mitigate the effects of disasters of extreme severity; however, damages and loss could be minimized through quantitative measures. From the perspective of numerical simulation, tsunami disasters are fluid–structure interaction (FSI) phenomena involving free surface flows. In some situations, not only elastic behavior but also plastic and fracture behavior should be considered in designing for safety. Therefore, a two-way coupling analysis for FSI problems involving free surfaces and moving boundaries is strongly required.

Solution strategies for multi-physics analyses including FSI are primarily classified into monolithic methods [7, 27] and partitioned methods [14, 16]. The monolithic methods lead and solve a single set of algebraic equations in order to treat all the domains simultaneously. Although such approaches provide high accuracy, a code developed for particular combination of physical problems is required. On the other hand, the partitioned methods solve each physical field separately while exchanging information. Although smaller time step sizes are required for sufficient coupling to be achieved in the case of strong interaction problems, the partitioned approaches have the advantage of software modularity, which allows the use of previously developed codes. With an iteration loop within a time step, tightly coupled partitioned iterative methods [41, 42] have drawn a great deal of attention.

A great deal of research has been conducted in order to solve FSI problems using various numerical methods. As a mesh-based method, finite element method (FEM) has been used in numerous studies. Because of its high accuracy and widespread application, the FEM is the de facto standard method for structural analysis. However, in flow analyses by FEM, we need to take care of the nonlinearity of an advective term and free surfaces. For the numerical stability, the stabilized finite element formulation for incompressible flow computations [8, 19, 20] has been proposed. Most numerical techniques to handle free surfaces and moving boundaries can be classified into two typical categories: interface tracking methods and interface capturing methods. In the interface tracking methods, including the arbitrary Lagrangian–Eulerian (ALE) method [48] and the space-time FEM [3], moving interfaces are treated as boundaries of finite elements that move to track the interfaces. The interface tracking methods are known to provide high accuracy and are applied to FSI analyses [57, 59]. However, since remeshing, which is not easy to parallelize efficiently, should be applied to maintain good mesh quality, their applicability is limited in the case of violent interface motions. On the other hand, the interface capturing methods, such as the volume of fluid (VOF) method [24] and the level set method [15, 51], use an artificial scalar field to describe the interfaces implicitly. These approaches are able to perform robust calculations even if the interfaces are subjected to severe motion involving topological changes. Nevertheless, the interface capturing methods suffer from mass conservation and interface smearing problems to some extent. In addition to the above approaches, which are Eulerian or ALE formulations in fluids, the particle FEM (PFEM)

[29, 30, 49, 54] uses Lagrangian formulations in fluid and eliminates the problems of free surfaces and moving boundaries. However, it should be noted that PFEM requires frequent remeshing.

In contrast to mesh-based methods, mesh-free particle methods, such as the smoothed particle hydrodynamics (SPH) [21, 40] method and the moving particle simulation (MPS) [36] method, are inherently Lagrangian methods, in which a continuum is discretized as a group of moving particles. One of the characteristics of these methods is that they discretize strong form of partial differential equations in the Lagrangian description without node connectivity information. This characteristic allows them to deal with free surfaces and moving boundaries easily. Another important characteristic is that mass conservation is automatically satisfied, assuming each particle has its own mass. In addition to this assumption, Koshizuka and Oka [36] applied a density validation term to the right-hand side of the pressure Poisson equation in the derivation of the MPS algorithm instead of the velocity divergence term generally used in the finite difference method (FDM) with the projection method [10]. Since this term has the effect of recovering the fluid volume and contributes robustness to the computations, it has been widely employed in both MPS and SPH computations [25, 33, 35] as a stabilization term. Thus, such mesh-free particle methods have the significant advantages of convenience and robustness in long-term analyses of free-surface flows with moving boundaries.

Examples of simulations that can take advantage of the mesh-free particle methods include disasters involving water, such as tsunamis [11, 47] and sloshing problems [13]. To solve problems which have large analytical areas, fully explicit algorithms, such as the weakly compressible SPH (WCSPH) method [46, 55] and the explicit MPS (E-MPS) method [50, 56, 61], have often been adopted. Since the explicit methods have high scalability for parallel computing, distributed-memory parallel algorithms have been investigated and developed [17, 34]. In recent years, coupled analyses including fluid–rigid body and fluid–structure interactions have been conducted using the mesh-free particle methods [4, 9, 28, 52]. These methods, however, are less accurate on solid surfaces and have limited applicability to structural analysis, such as with the FDM, because, unlike the FEM, which solves weak form equations, the Neumann boundary conditions on solid surfaces must be explicitly imposed and the derivatives are approximated by particles within the circular support domain.

In order to accurately compute structures, hybrid coupling approaches have been developed. These approaches have been used in structure–structure models to investigate fracture and penetration of solids subjected to impact or blast loads [2, 5, 12, 32, 39] and was later applied to FSI problems [18, 22, 26, 37, 44, 45, 58, 62], in which mesh-free particle methods are used for free-surface flows and the FEM for structures.

We developed the MPS-FE method [45] which adopts the E-MPS method for fluid computation involving free surfaces and the FEM for structure computation. These two methods are coupled with a partitioned approach, i.e., the conventional serial staggered (CSS) scheme [14], which can set different time step sizes for the fluid and structure computations. The method combines the advantages of both methods to achieve efficiency as well as robustness. However, this conventional MPS-FE method, in which MPS wall boundary particles and finite elements are overlapped in order to exchange information on fluid–structure interfaces, has difficulty in dealing with complex shaped fluid–structure boundaries, because the wall particles have to be set in an orthogonal and uniform grid manner for accurate execution of the MPS computation. This requires cumbersome interpolation for the exchange of physical values based on node-particle correspondence relation. Thus, the MPS-FE method lacks versatility and reduces software modularity. In addition, forces on fluid–structure interfaces are not balanced when the pressure on the walls is calculated in the above conventional way.

Therefore, we improved the conventional MPS-FE method and proposed the improved MPS-FE method [44] by using the MPS polygon wall boundary model [23, 61] instead of the MPS wall particle model. Since the MPS polygon wall boundary model can express wall boundaries as plane polygons, the interface in the fluid computation now corresponds to that in the structure computation. This improves versatility of the MPS-FE method.

The outline of this chapter is as follows. The discretizations of the governing equations for the MPS method and the FEM are presented in Sects. 2 and 3, respectively. The basis of the fluid–structure interaction model and the weak coupling scheme are introduced in Sect. 4. The MPS polygon wall boundary model is presented in Sect. 5. Here, we describe the computation of the pressure on the polygons that is necessary to apply the polygon model to the improved MPS-FE method. Considering the pressure on the polygons, the formulation and algorithm of the improved MPS-FE method including the MPS polygon wall boundary model is presented in Sect. 6. To verify the proposed method, we solve a dam break problem with an elastic obstacle by the improved MPS-FE method, and compare the results obtained by the conventional MPS-FE method and other methods in Sect. 7. Finally, conclusions are given in Sect. 8.

2 MPS Formulation for Fluid Dynamics

2.1 Governing Equations

The Navier–Stokes equations and the continuity equation for a quasi-incompressible Newtonian fluid in a Lagrangian reference frame are given as follows:

$$\frac{D\boldsymbol{v}}{Dt} = -\frac{1}{\rho}\nabla p + \nu_f \nabla^2 \boldsymbol{v} + \boldsymbol{g} \tag{1}$$

$$\frac{1}{\rho}\frac{\partial \rho}{\partial t} + \nabla \cdot \boldsymbol{v} = 0 \tag{2}$$

where ρ is the density of the fluid, $\boldsymbol{v}$ is the velocity vector, p is the pressure, ν_f is the kinetic viscosity, and $\boldsymbol{g}$ is the gravitational acceleration vector.

2.2 MPS Discretization

In the MPS discretization, the differential operators acting on a particle i are evaluated using the neighboring particles j that are located within an effective radius r_e. $\mathbb{P}_i$ represents the set of the neighboring particles of the particle i as follows:

$$\mathbb{P}_i = \{j \mid r_e > |\boldsymbol{x}_{ij}| \wedge j \neq i\}. \tag{3}$$

In this section, ϕ_{ij} denotes $\phi_j - \phi_i$, where ϕ represents a property of the particle. The neighboring particles are weighted using the weight function of their separation from the particle i, $r = |\boldsymbol{x}_{ij}|$. In the original MPS [36], the weight function is given as

$$w(r) = \begin{cases} \dfrac{r_e}{r} - 1 & (0 \leq r < r_e) \\ 0 & (r_e \leq r). \end{cases} \tag{4}$$

To calculate the weighted average, a normalization factor termed the particle number density is defined as

$$n_i = \sum_{j \in \mathbb{P}_i} w(|\boldsymbol{x}_{ij}|). \tag{5}$$

In the MPS method the fractional step algorithm is applied for time discretization, so each time step is divided into prediction and correction steps, as follows:

① **Prediction step**

$$\frac{\boldsymbol{v}^* - \boldsymbol{v}^n}{\Delta t} = \nu_f \left\langle \nabla^2 \boldsymbol{v} \right\rangle^n + \boldsymbol{g}. \tag{6}$$

② **Correction step**

$$\frac{\boldsymbol{v}^{n+1} - \boldsymbol{v}^*}{\Delta t} = -\frac{1}{\rho} \langle \nabla p \rangle^{n+1}. \tag{7}$$

Here $\boldsymbol{v}^*$ is the intermediate velocity vector. The angle brackets $\langle\rangle$ indicate discretization by the MPS differential operators which are given as follows:

$$\langle \nabla p \rangle_i = \frac{d}{n^0} \sum_{j \in \mathbb{P}_i} \left[\frac{\boldsymbol{x}_{ij}}{|\boldsymbol{x}_{ij}|} \frac{p_{ij}}{|\boldsymbol{x}_{ij}|} w(|\boldsymbol{x}_{ij}|) \right] \tag{8}$$

$$\left\langle \nabla^2 \boldsymbol{v} \right\rangle_i = \frac{2d}{\lambda^0 n^0} \sum_{j \in \mathbb{P}_i} \left[\boldsymbol{v}_{ij} w(|\boldsymbol{x}_{ij}|) \right]. \tag{9}$$

Here d is the number of dimensions and n^0 is the initial value of the particle number density given by Eq. (5). Similarly to n^0, λ^0 is a quantity calculated for the initial geometry:

$$\lambda_i = \frac{\sum_{j \in \mathbb{P}_i} |\boldsymbol{x}_{ij}|^2 w(|\boldsymbol{x}_{ij}|)}{\sum_{j \in \mathbb{P}_i} w(|\boldsymbol{x}_{ij}|)}. \tag{10}$$

2.3 Pressure Calculation

To calculate the correction step, Eq. (7), the pressure values in the next time step p^{n+1} are required. For the pressure calculation, there are two methods, i.e. semi-implicit MPS (SI-MPS) methods [36] in which the pressure Poisson equation is solved implicitly to determine the pressure values and an explicit MPS (E-MPS) [50, 56] method that assumes weak compressibility.

We employ the E-MPS method to compute FSI problems with free surface flow. Assuming the weak compressibility of fluids, the explicit calculation of the pressure can be written as

$$p_i^{n+1} = c^2 \rho \left(\frac{n_i^*}{n^0} - 1 \right). \tag{11}$$

where c is a parameter set arbitrarily to satisfy the conditions of stability and incompressibility. In addition, the E-MPS uses the following pressure gradient model instead of Eq. (8):

$$\langle \nabla p \rangle_i = \frac{d}{n^0} \sum_{j \in \mathbb{P}} \left[\frac{\boldsymbol{x}_{ij}}{|\boldsymbol{x}_{ij}|} \frac{p_j + p_i}{|\boldsymbol{x}_{ij}|} w(|\boldsymbol{x}_{ij}|) \right]. \tag{12}$$

3 FE Formulation for Structure Dynamics

3.1 Governing Equations for Finite Deformation

The virtual work equation of the total Lagrangian formulation with geometric non-linearity is given as follows:

$$\int_{\Omega_s} \rho \frac{\partial^2 \boldsymbol{u}}{\partial t^2} \cdot \delta \boldsymbol{u} d\Omega + \int_{\Omega_s} \boldsymbol{S} : \delta \boldsymbol{E} d\Omega - \int_{\Omega_s} \rho \boldsymbol{g} \cdot \delta \boldsymbol{u} d\Omega - \int_{\Gamma_s} \boldsymbol{t} \cdot \delta \boldsymbol{u} d\Gamma = 0. \qquad (13)$$

where Ω_s is the structure domain with boundary Γ_s, $\boldsymbol{S}$ is the second Piola–Kirchhoff stress tensor, $\boldsymbol{E}$ is the Green-Lagrange strain tensor, ρ is the density of the structure, and $\boldsymbol{t}$ is the boundary traction in a reference configuration. $\boldsymbol{E}$ and $\boldsymbol{S}$ in Eq. (13) are expressed using the deformation gradient tensor $\boldsymbol{F} = \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{X}} + \boldsymbol{I}$ as follows:

$$\boldsymbol{E} = \frac{1}{2}(\boldsymbol{F}^T \boldsymbol{F} - \boldsymbol{I}) \qquad (14)$$

$$\boldsymbol{S} = \mathsf{C} : \boldsymbol{E}. \qquad (15)$$

where $\boldsymbol{I}$ is the second-order identity tensor and C is the fourth-order elastic modulus tensor. In the verification computation in Sect. 7, we use the Saint Venant–Kirchhoff model, which represents the linear relationship between the second Piola–Kirchhoff stress tensor and the Green-Lagrange strain tensor. In this case, C is expressed using the fourth-order identity tensor I and Lamé constants λ and μ as follows:

$$\mathsf{C} = \lambda \boldsymbol{I} \otimes \boldsymbol{I} + 2\mu \mathsf{I}. \qquad (16)$$

3.2 Solution of Non-Linear Problems

Because of the nonlinear term in Eq. (13), nonlinear solvers such as the Newton–Raphson method are required. Using the Newton–Raphson method, the discretized form of Eq. (13),

$$\boldsymbol{\Psi} \equiv \boldsymbol{f} - \boldsymbol{R}(\boldsymbol{u}) - \boldsymbol{M}\ddot{\boldsymbol{u}} = 0, \qquad (17)$$

is solved iteratively as follows:

$$\boldsymbol{K}_T^i d\boldsymbol{u}_{n+1}^i = \boldsymbol{\Psi}_{n+1}^i \qquad (18)$$

$$\boldsymbol{u}_{n+1}^{i+1} = \boldsymbol{u}_{n+1}^i + d\boldsymbol{u}_{n+1}^i. \qquad (19)$$

Here n is the time step, i is the iteration counter, $\boldsymbol{f}$ is the external force vector, $\boldsymbol{R}(\boldsymbol{u})$ is the internal work vector, and $\boldsymbol{M}$ and $\boldsymbol{K}_T$ are the global mass matrix and the global tangent matrix, respectively, which are assembled from each element matrix.

3.3 Time Integration Scheme

The time integration scheme used in the present study is the following Newmark's β method :

$$\boldsymbol{u}_{n+1} = \boldsymbol{u}_n + \Delta t\dot{\boldsymbol{u}}_n + \left(\frac{1}{2} - \beta\right)\Delta t^2\ddot{\boldsymbol{u}}_n + \beta\Delta t^2\ddot{\boldsymbol{u}}_{n+1} \tag{20}$$

$$\dot{\boldsymbol{u}}_{n+1} = \dot{\boldsymbol{u}}_n + (1-\gamma)\Delta t\ddot{\boldsymbol{u}}_n + \gamma\Delta t\ddot{\boldsymbol{u}}_{n+1}. \tag{21}$$

In this study, we use the coefficient values $\beta = 0.3025$ and $\gamma = 0.6$, which add numerical damping. This kind of damping has been frequently used as a stabilization technique for FSI computation [6, 43, 60].

4 MPS-FE Method

In the MPS-FE method [45], the MPS wall boundary particles and the finite elements are overlapped in order to exchange information on the fluid and structure interface, as shown in Fig. 1. Some layers of the wall boundary particles are also set inside the structure because the calculation of the particle number density given by Eq. (5) must fill the area within an effective radius r_e with particles. The interaction between the fluid and the structure is expressed by converting the

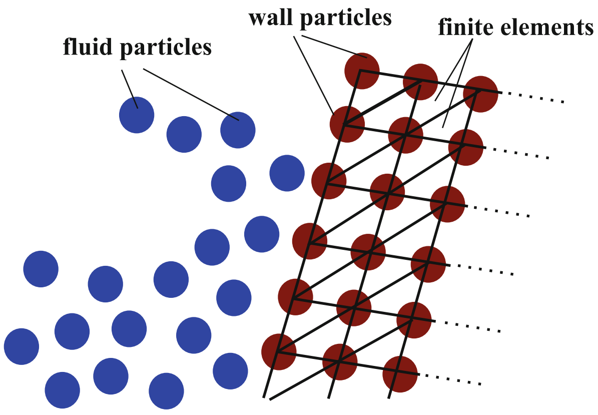


Fig. 1 Overlapping finite elements and wall particles in the conventional MPS-FE method

obtained pressures of the MPS wall boundary particles into equivalent nodal forces, and the displacements and velocities of the finite elements are transmitted to the wall particles.

The MPS-FE method assumes a linear pressure distribution on the boundary side of an element when the pressures are converted into nodal forces. Assuming such a pressure distribution, $p(\boldsymbol{x})$, if the wall particles and the finite elements are set as shown in Fig. 1, the equivalent forces of the node i are expressed as follows:

$$\boldsymbol{f}_i^{\text{node}} = \int_\Gamma N_i^\Gamma(\boldsymbol{x}) p(\boldsymbol{x}) \boldsymbol{n} d\Gamma. \tag{22}$$

where N_i^Γ is the shape function on the fluid–structure interfaces of node i, and $\boldsymbol{n}$ is the inward unit normal vector.

The particle position $\boldsymbol{x}^{\text{fluid}}$ of an element is updated by interpolation using the current position of the structure $\boldsymbol{x}^{\text{structure}} = (\boldsymbol{X} + \boldsymbol{u})^{\text{structure}}$ as follows:

$$\boldsymbol{x}^{\text{fluid}} = \sum_i N_i^\Gamma \boldsymbol{x}_i^{\text{structure}}. \tag{23}$$

The velocity can also be determined in a similar manner.

5 Polygon Wall Boundary Model

In general, MPS computations use wall particles to model wall boundaries as illustrated in Fig. 2. However, since the particles have to be set in an orthogonal and uniform grid manner to correctly calculate the particle number density defined by Eq. (5), the wall particle model ineffectively simulates complex shaped boundaries. Harada et al. proposed the polygon wall boundary model [23] which can express wall boundaries as plane polygons. By applying the polygon wall boundary model to the SI-MPS computation, they showed that the number of matrix elements in the pressure Poisson calculation and the computation time are reduced. Yamada et al. also proposed a different polygon wall boundary model applied to the E-MPS [61]. The differences between these two methods stem from the character of the pressure gradient term. In the improved MPS-FE method, we use Yamada's polygon wall boundary model because it has higher stability than Harada's.

5.1 Wall Weight Function

To implement the polygon wall with no wall particles, the interpolation of the contributing part (green area at the right of Fig. 2) within the particle number density computation is required. The wall weight function is defined by the sum of the

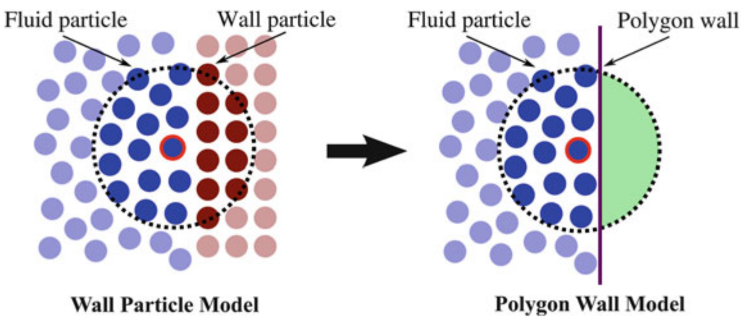


Fig. 2 Wall particle model and polygon wall boundary model

virtual wall particles' weights created inside the wall. Using the distance between the particle and the wall $|\boldsymbol{x}_{ib}|$, the wall weight function $z(|\boldsymbol{x}_{ib}|)$ is defined as

$$z(|\boldsymbol{x}_{ib}|) = \sum_{j \in \text{wall}} w(|\boldsymbol{x}_{ij}|). \tag{24}$$

Within the neighboring particles $j \in \mathbb{P}_i$ of the particle i, we define the fluid particles as $j \in$ particle and the virtual wall particles as $j \in$ wall. The particle number density Eq. (5) can be divided into the contributions of the fluid particles and the wall weight function as follows:

$$\begin{aligned} n_i &= \sum_{j \in \text{particle}} w(|\boldsymbol{x}_{ij}|) + \sum_{j \in \text{wall}} w(|\boldsymbol{x}_{ij}|) \\ &= \sum_{j \in \text{particle}} w(|\boldsymbol{x}_{ij}|) + z(|\boldsymbol{x}_{ib}|). \end{aligned} \tag{25}$$

In actual computation, the wall weight function is determined by the linear interpolation of values at the discrete distances $d_0, d_1, \dots, d_n (= r_e)$, which are calculated in advance of fluid computation (Fig. 3).

5.2 Viscosity Term

The viscosity term discretized by the Laplacian model (9) is divided into fluid particle and polygon wall contributions:

$$\left\langle \nabla^2 \boldsymbol{v} \right\rangle_i = \left\langle \nabla^2 \boldsymbol{v} \right\rangle_i^{\text{particle}} + \left\langle \nabla^2 \boldsymbol{v} \right\rangle_i^{\text{wall}}. \tag{26}$$

Assuming that the velocity of the wall b is a constant value $\boldsymbol{v}_b$ within the effective radius of the particle i, the contributing region of the polygon wall can be deformed

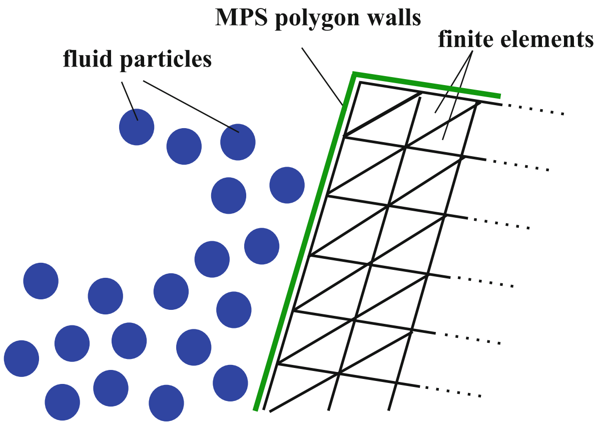


Fig. 3 Improved MPS-FE method with MPS polygon wall boundary model

as follows:

$$\left\langle \nabla^2 \boldsymbol{v} \right\rangle_i^{\text{wall}} = \frac{2d}{\lambda^0 n^0} \boldsymbol{v}_{ib} \sum_{j \in \text{wall}} w(|\boldsymbol{x}_{ij}|) \tag{27}$$

$$= \frac{2d}{\lambda^0 n^0} \boldsymbol{v}_{ib}\, z(|\boldsymbol{x}_{ib}|). \tag{28}$$

Substituting the term into Eq. (26), the equation for the viscosity in the polygon wall model is derived as follows:

$$\left\langle \nabla^2 \boldsymbol{v} \right\rangle_i = \frac{2d}{\lambda^0 n^0} \sum_{j \in \text{particle}} \left[\boldsymbol{v}_{ij} w(|\boldsymbol{x}_{ij}|) \right] + \frac{2d}{\lambda^0 n^0} \boldsymbol{v}_{ib}\, z(|\boldsymbol{x}_{ib}|). \tag{29}$$

5.3 Pressure Gradient Term

As is the case with the viscosity term, the pressure gradient term is divided into the fluid particle and polygon wall contributions as follows:

$$\langle \nabla p \rangle_i = \langle \nabla p \rangle_i^{\text{particle}} + \langle \nabla p \rangle_i^{\text{wall}}. \tag{30}$$

As mentioned previously, the improved MPS-FE method adopts the following Yamada's pressure gradient model [61]:

$$\langle \nabla p \rangle_i^{\text{wall}} = \frac{d}{n^0} \frac{\boldsymbol{x}_{ib}}{|\boldsymbol{x}_{ib}|} \frac{p_i}{|\boldsymbol{x}_{ib}|} z(|\boldsymbol{x}_{ib}|) \tag{31}$$

$$\langle \nabla p \rangle_i = \frac{d}{n^0} \sum_{j \in \text{particle}} \left[\frac{\boldsymbol{x}_{ij}}{|\boldsymbol{x}_{ij}|} \frac{p_j + p_i}{|\boldsymbol{x}_{ij}|} w(|\boldsymbol{x}_{ij}|) \right] + \frac{d}{n^0} \frac{\boldsymbol{x}_{ib}}{|\boldsymbol{x}_{ib}|} \frac{p_i}{|\boldsymbol{x}_{ib}|} z(|\boldsymbol{x}_{ib}|). \tag{32}$$

5.4 Pressure on Polygon Wall

Let $\mathbb{W}_b$ be the set of fluid particles that are influenced by the presence of the polygon wall b. The pressure on the polygon wall surface is the force exerted by the fluid particles per unit area. The force on the polygon wall b exerted by fluid particles $\boldsymbol{f}_b$ is the reaction to the sum of the pressure gradient forces, $\boldsymbol{f}_{bj} = -\boldsymbol{f}_{jb}$, exerted by every particle j belonging to the set $\mathbb{W}_b$ as follows:

$$\boldsymbol{f}_b = -\sum_{j \in \mathbb{W}_b} \boldsymbol{f}_{jb} \tag{33}$$

$$\boldsymbol{f}_{jb} = -\frac{m_i}{\rho} \langle \nabla p \rangle_j^{\text{wall}}, \tag{34}$$

where m_i is the mass of the fluid particle. Assuming that all fluid particle radii are identical and the initial particle spacing is l^0, the mass of each fluid particle can be calculated as $m_i = \rho (l^0)^d$. Thus, the distributed load $\boldsymbol{q}_b$ on the polygon wall can be computed as follows:

$$\boldsymbol{q}_b = \frac{\sum_{j \in \mathbb{W}_b} (l^0)^d \langle \nabla p \rangle_j^{\text{wall}}}{s_b}, \tag{35}$$

where s_b is the area of the polygon b, and pressure p can be computed as the magnitude of $\boldsymbol{q}$ in the following:

$$p = |\boldsymbol{q}_b|. \tag{36}$$

6 Improved MPS-FE Method

6.1 Fluid–Structure Interaction Model

In the improved MPS-FE method [44], the force on the polygon wall b exerted by the particles j is described by $\boldsymbol{f}_{jb}$. In addition, the point on the polygon surface from which a line perpendicular to the surface can extend to the particle is termed the effective position of the particle. In d-dimensional computations, an effective position on a polygon wall can be expressed as a $(d-1)$-dimensional vector $\boldsymbol{x}^\Gamma$

using an arbitrary $(d-1)$-dimensional basis. Furthermore, there are shape functions in the normalized natural coordinate $\boldsymbol{\xi}$ expressed as $N_i^\Gamma(\boldsymbol{\xi})(i = 1, 2, \dots n_{\text{node}}^\Gamma)$ on the surfaces of a d-dimensional solid element, where n_{node}^Γ is the number of nodes on the element surface. The effective position $\boldsymbol{x}_j^\Gamma$ of particle j expressed in the normalized natural coordinate is $\boldsymbol{\xi}_j$. The point loads $\boldsymbol{f}_{jb}^{\text{particle}}$ are distributed as equivalent nodal forces $\boldsymbol{f}_i^{\text{node}}$ on the node i using the shape function $N_i^\Gamma(\boldsymbol{\xi})$ over the element surface as follows:

$$\boldsymbol{f}_i^{\text{node}} = \sum_{j \in \mathbb{W}_b} N_i^\Gamma(\boldsymbol{\xi}_j)\boldsymbol{f}_{jb}^{\text{particle}}. \tag{37}$$

The pressure on the polygon described by Eq. (36) means the pressure values are constant. To replace Eq. (37) which expresses the fluid force as point loads, the equivalent nodal forces representing the load distribution $\boldsymbol{q}(\boldsymbol{x})$ are expressed as follows:

$$\boldsymbol{f}_i^{\text{node}} = \int_{\Gamma_{\boldsymbol{x}}} N_i^\Gamma(\boldsymbol{x})\boldsymbol{q}(\boldsymbol{x})d\Gamma_{\boldsymbol{x}} \tag{38}$$

$$= \int_{\Gamma_{\boldsymbol{\xi}}} N_i^\Gamma(\boldsymbol{\xi})\boldsymbol{q}(\boldsymbol{\xi})\det \boldsymbol{J} d\Gamma_{\boldsymbol{\xi}}. \tag{39}$$

where $\boldsymbol{J}$ is the Jacobian matrix. If the load distribution on the polygon b is computed using Eq. (36), the values on the polygon are constant, $\boldsymbol{q}_b$, and $\det \boldsymbol{J}$ is approximated as

$$\det \boldsymbol{J} \simeq \frac{\int_{\Gamma_{\boldsymbol{x}}} d\Gamma_{\boldsymbol{x}}}{\int_{\Gamma_{\boldsymbol{\xi}}} d\Gamma_{\boldsymbol{\xi}}} \tag{40}$$

$$\simeq \frac{s_b}{\int_{\Gamma_{\boldsymbol{\xi}}} d\Gamma_{\boldsymbol{\xi}}}. \tag{41}$$

Hence, Eq. (39) can be rewritten as follows:

$$\boldsymbol{f}_i^{\text{node}} = c_1 c_2 \boldsymbol{f}_b \tag{42}$$

$$c_1 \equiv \frac{1}{\int_{\Gamma_{\boldsymbol{\xi}}} d\Gamma_{\boldsymbol{\xi}}} \tag{43}$$

$$c_2 \equiv \int_{\Gamma_{\boldsymbol{\xi}}} N_i^\Gamma(\boldsymbol{\xi}) d\Gamma_{\boldsymbol{\xi}}. \tag{44}$$

In this study, we used Eq. (42) for the computation of the problem in Sect. 7.

6.2 Weak Coupling Scheme

Numerical simulations of the FSI problem in the present method use the finite deformation FEM and the E-MPS method. In such a case, the fluid and structure computation times in a time step are quite different because the structure problem is solved implicitly and involves nonlinear iteration, whereas the fluid problem is solved explicitly. Explicit fluid computation requires a finer temporal resolution. If the structure time step Δt_s set to be the same as the fluid time step Δt_f, computation requires an enormous amount of time. Thus, we adopt a CSS scheme [14] as an FSI coupling strategy, which applies subcycling to fluid computation in order to set different values of Δt_s and Δt_f. In addition, the fluid loads that are transferred to structure are averaged over $k = \Delta t_s / \Delta t_f$ fluid subcycles in order to suppress the instability caused by the nonphysical pressure oscillation in MPS.

The procedure at the n-th time step is illustrated in Fig. 4 and summarized below.

① Determine the predicted values of displacement $\hat{\boldsymbol{u}}_{n+1}$ and velocity $\hat{\dot{\boldsymbol{u}}}_{n+1}$ from previous values.

② Perform k times MPS calculation and determine the averaged loads on the polygon walls $\bar{\boldsymbol{f}}$ as follows:

$$\bar{\boldsymbol{f}} = \frac{1}{k} \sum_{i=1}^{k} \boldsymbol{f}_{n+\frac{i}{k}}. \tag{45}$$

by applying the wall boundary condition that was defined according to the behavior of the structure. If the predictor is obtained by linear extrapolation as:

$$\hat{\boldsymbol{u}}_{n+1} = \boldsymbol{u}_n + \Delta t_s \dot{\boldsymbol{u}}_n, \tag{46}$$

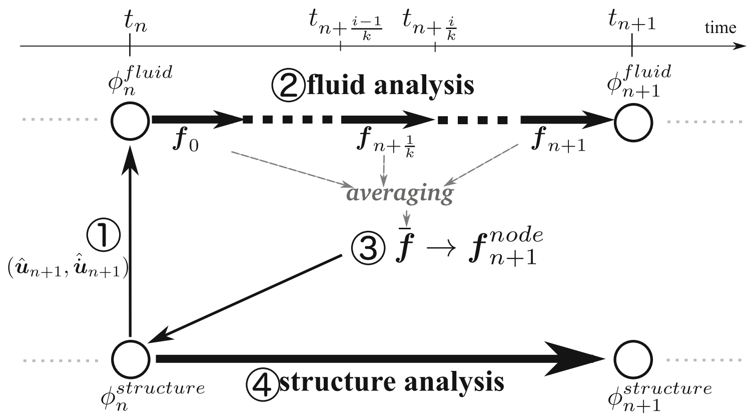


Fig. 4 Weak coupling scheme based on CSS

the boundary values $\hat{\boldsymbol{u}}_{n+\frac{i-1}{k}}$ and $\hat{\dot{\boldsymbol{u}}}_{n+\frac{i-1}{k}}$ are expressed as follows:

$$\hat{\boldsymbol{u}}_{n+\frac{i-1}{k}} = \boldsymbol{u}_n + \frac{i-1}{k}(\hat{\boldsymbol{u}}_{n+1} - \boldsymbol{u}_n). \tag{47}$$

③ Convert $\bar{\boldsymbol{f}}$ into the nodal equivalent forces $\boldsymbol{f}^{\mathrm{node}}_{n+1}$.
④ Perform FEM calculation to obtain updated displacements $\boldsymbol{u}_{n+1}$ and velocities $\dot{\boldsymbol{u}}_{n+1}$.

7 Verification of the Improved MPS-FE Method

To verify the improved MPS-FE method, we solved a dam break problem with an elastic obstacle and compared the results with those obtained by other methods.

7.1 Simulation Setup

The initial configuration of the dam break problem with an elastic obstacle is illustrated in Fig. 5. This problem was initially solved in [59] using a space-time FEM. Since then other researchers have investigated [31, 52, 54]. Calculation parameters are given in Tables 1 and 2.

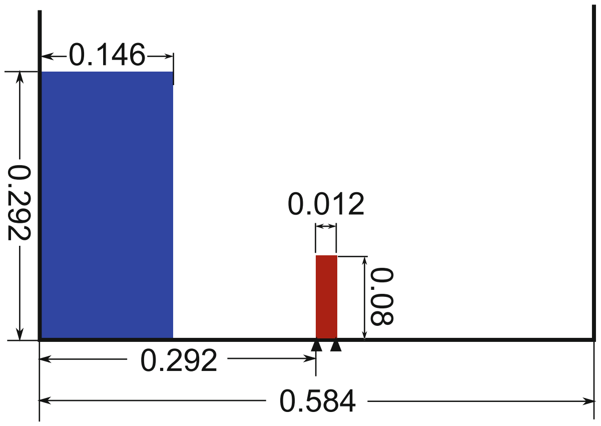


Fig. 5 Verification problem: initial configuration (units: m)

Table 1 Verification problem: fluid parameters

Time step width	1.0×10^{-6} (s)
Number of particles	10,658
Particle spacing	2.0×10^{-3} (m)
Effective radius	5.8×10^{-3} (m)
Density	1.0×10^{3} (kg/m^3)
Kinetic viscosity	1.0×10^{-6} (m^2/s)
Gravitational acceleration	10.0 (m/s^2)

Table 2 Verification problem: structure parameters

Time step width	1.0×10^{-4} (s)
Number of elements	120
Young's modulus	1.0×10^{6} (kg/s^{2}m)
Poisson's ratio	0.0
Density	2.5×10^{-3} (kg/m^3)
Gravitational acceleration	10.0 (m/s^2)

7.2 *Simulation Result and Comparison with Other Methods*

Figure 7 shows the displacement of the upper-left corner of the elastic obstacle at each time step. In the figure, the result of the improved MPS-FE method is compared with that of the conventional MPS-FE method [45], the PFEM [54], and the SPH method [52]. Whereas the computation of the conventional MPS-FE method in the previous study did not use any numerical damping for stabilization, the conventional MPS-FE computation shown here uses the same damping conditions as the improved MPS-FE method as described in Sect. 3.3.

Snapshots of the simulation using the improved MPS-FE method are shown in Fig. 6. The elastic structure is first deformed to the left when the fluid hits the lower part of the structure and immediately deflects to the right as the fluid rises and climbs over the structure. The maximum deflection occurs at approximately $t = 0.25$ (s), after which the structure pushes the fluid back. The structure deflects to the left as a result of the impact of the fluid that strikes the left wall and recovers at $t = 0.6$ (s).

As shown in Fig. 7, the improved MPS-FE result is in good agreement with the PFEM result. Although the conventional MPS-FE result is also in agreement with that of the PFEM to some extent, unnatural vibration after the initial deflection and a phase shift after 0.8 (s) are observed. As mentioned previously, in the conventional MPS-FE approach, the forces on the fluid particles exerted by the structure are not consistent with the force on the structure computed by interpolation using the pressure on the wall particles. Since this inconsistency causes the fluid particles to exhibit unstable behaviors near the fluid–structure interface, some fluid particles become too close or too far from the wall particles. This results in incorrect free surface determination and unreasonably high pressure values near the interface, leading to unnatural vibrations. The phase shift after 0.8 (s) in the conventional MPS-FE method also comes from inconsistency in exchanging physical values, which causes excessive decays in energy. In converting the pressure on the wall

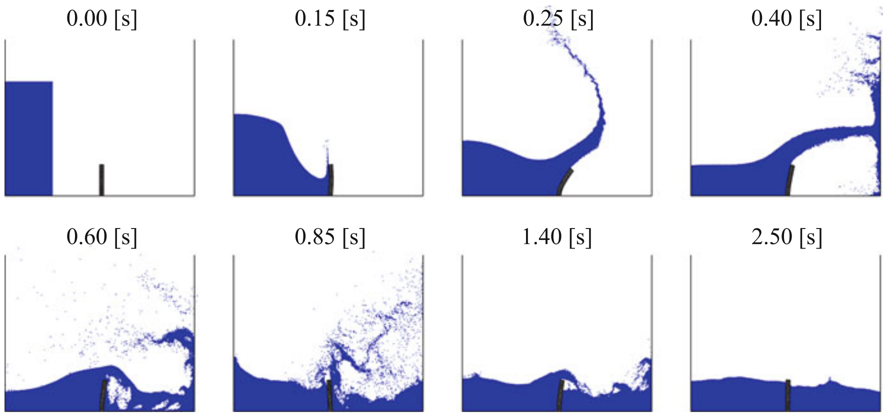


Fig. 6 Verification problem: visualization of results

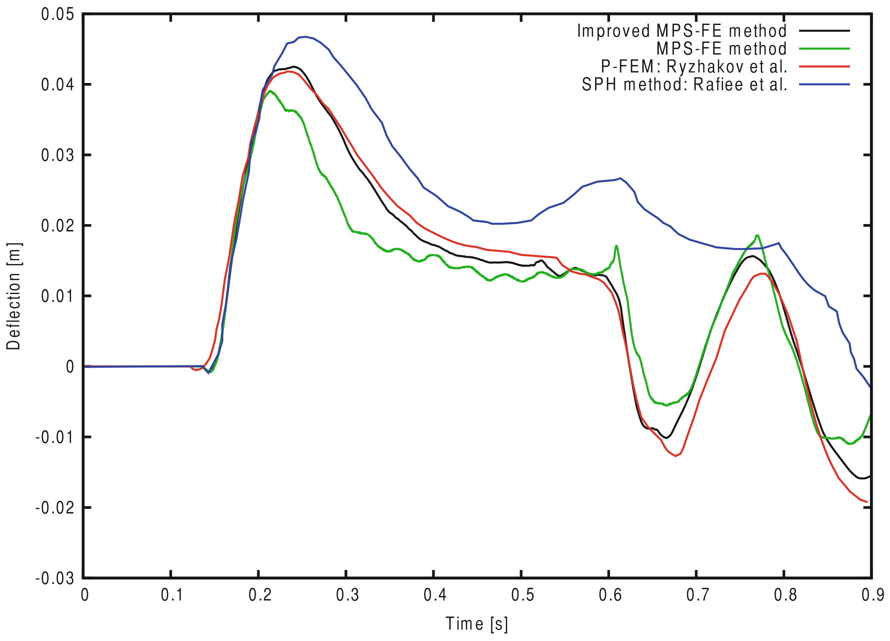


Fig. 7 Verification problem: comparison with other methods

particles to the equivalent nodal forces, energy is not conserved. In contrast, the improved MPS-FE method overcomes such problems in the transmission of force from the fluid to the structure by computing the force on the polygon wall directly using Eq. (36). Moreover, whereas the results obtained using the SPH method exhibit a different trend from the results obtained using the other methods, the improved MPS-FE method provided appropriate results. It can be achieved by applying FEM to the structure computation in the present MPS-FE method.

8 Conclusions

In this chapter, we introduced the MPS-FE method and its improved version, in which the MPS polygon wall boundary model is applied instead of using MPS wall particles. The computation method used to evaluate the pressure on the polygon wall was presented and the FSI model for the improved MPS-FE method was formulated. To verify the proposed method, the dam break problem with an elastic obstacle was solved. Comparing the result of the proposed method with those of the conventional method and PFEM clearly indicated that the improved MPS-FE method has adequate accuracy, higher stability, and versatility.

Now, we have been developing a large-scale parallel MPS-FE analysis system using the large-scale parallel code for the E-MPS method (LexADV_EMPS [38, 47]) and FEM (ADVENTURE_Solid [1, 63]). This system makes it possible to conduct robust simulations of three-dimensional fluid–structure interaction. An example is shown in Fig. 8, which is a three-dimensional simulation of dam break and an elastic column with constraints on the bottom surface.

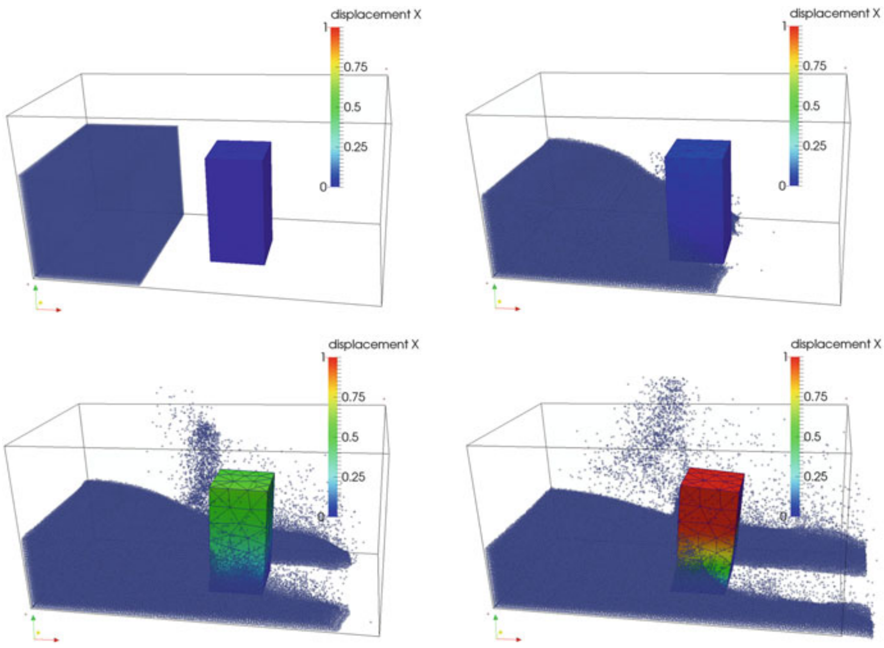


Fig. 8 Example of three-dimensional MPS-FE computation

We believe that hybrid approaches using mesh-free particle methods and FEM will be powerful tools to predict the damage to the equipments and instruments of facilities important to safety, and will help in design for disaster mitigation.

Acknowledgements This study was supported by the JSTCREST project "Development of a Numerical Library Based on Hierarchical Domain Decomposition for Post Petascale Simulation."

References

1. Adventure project: Development of computational mechanics system for large scale analysis and design. http://adventure.sys.t.u-tokyo.ac.jp/
2. Aktay, L., Johnson, A.F.: Fem/sph coupling technique for high velocity impact simulations. In: Advances in Meshfree Techniques, pp. 147–167. Springer, New York (2007)
3. Aliabadi, S.K., Tezduyar, T.E.: Space-time finite element computation of compressible flows involving moving boundaries and interfaces. Comput. Methods Appl. Mech. Eng. **107**(1), 209–223 (1993)
4. Antoci, C., Gallati, M., Sibilla, S.: Numerical simulation of fluid–structure interaction by SPH. Comput. Struct. **85**(11), 879–890 (2007)
5. Attaway, S., Heinstein, M., Swegle, J.: Coupling of smooth particle hydrodynamics with the finite element method. Nucl. Eng. Des. **150**(2), 199–205 (1994)
6. Bazilevs, Y., Calo, V., Hughes, T., Zhang, Y.: Isogeometric fluid-structure interaction: theory, algorithms, and computations. Comput. Mech. **43**(1), 3–37 (2008)
7. Blom, F.J.: A monolithical fluid-structure interaction algorithm applied to the piston problem. Comput. Methods Appl. Mech. Eng. **167**(3), 369–391 (1998)
8. Brooks, A.N., Hughes, T.J.: Streamline upwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier-stokes equations. Comput. Methods Appl. Mech. Eng. **32**(1), 199–259 (1982)
9. Chikazawa, Y., Koshizuka, S., Oka, Y.: A particle method for elastic and visco-plastic structures and fluid-structure interactions. Comput. Mech. **27**(2), 97–106 (2001)
10. Chorin, A.J.: Numerical solution of the Navier-stokes equations. Math. Comput. **22**(104), 745–762 (1968)
11. Crespo, A., Gómez-Gesteira, M., Dalrymple, R.A.: 3d sph simulation of large waves mitigation with a dike. J. Hydraul. Res. **45**(5), 631–642 (2007)
12. De Vuyst, T., Vignjevic, R., Campbell, J.: Coupling between meshless and finite element methods. Int. J. Impact Eng. **31**(8), 1054–1064 (2005)
13. Delorme, L., Colagrossi, A., Souto-Iglesias, A., Zamora-Rodriguez, R., Botia-Vera, E.: A set of canonical problems in sloshing, part i: Pressure field in forced roll–comparison between experimental results and sph. Ocean Eng. **36**(2), 168–178 (2009)
14. Farhat, C., Lesoinne, M.: Two efficient staggered algorithms for the serial and parallel solution of three-dimensional nonlinear transient aeroelastic problems. Computer Methods Appl. Mech. Eng. **182**(3), 499–515 (2000)
15. Fedkiw, S.O.R.: Level Set Methods and Dynamic Implicit Surfaces (2003)
16. Felippa, C.A., Park, K., Farhat, C.: Partitioned analysis of coupled mechanical systems. Comput. Methods Appl. Mech. Eng. **190**(24), 3247–3270 (2001)
17. Ferrari, A., Dumbser, M., Toro, E.F., Armanini, A.: A new 3d parallel sph scheme for free surface flows. Comput. Fluids **38**(6), 1203–1217 (2009)
18. Fourey, G., Oger, G., Le Touzé, D., Alessandrini, B.: Violent fluid-structure interaction simulations using a coupled sph/fem method. In: IOP Conference Series: Materials Science and Engineering, vol. 10, p. 012041. IOP Publishing, Bristol (2010)

19. Franca, L.P., Frey, S.L.: Stabilized finite element methods: II. The incompressible Navier-stokes equations. Comput. Methods Appl. Mech. Eng. **99**(2), 209–233 (1992)
20. Franca, L.P., Frey, S.L., Hughes, T.J.: Stabilized finite element methods: I. Application to the advective-diffusive model. Comput. Methods Appl. Mech. Eng. **95**(2), 253–276 (1992)
21. Gingold, R.A., Monaghan, J.J.: Smoothed particle hydrodynamics: theory and application to non-spherical stars. Mon. Not. R. Astron. Soc. **181**(3), 375–389 (1977)
22. Groenenboom, P.H., Cartwright, B.K.: Hydrodynamics and fluid-structure interaction by coupled sph-fe method. J. Hydraul. Res. **48**(S1), 61–73 (2010)
23. Harada, T., Koshizuka, S., Shimazaki, K.: Improvement of wall boundary calculation model for MPS method. Trans. Jpn. Soc. Comput. Eng. Sci. (20080006) (2008) (in Japanese)
24. Hirt, C.W., Nichols, B.D.: Volume of fluid (vof) method for the dynamics of free boundaries. J. Comput. Phys. **39**(1), 201–225 (1981)
25. Hu, X., Adams, N.A.: An incompressible multi-phase sph method. J. Comput. Phys. **227**(1), 264–278 (2007)
26. Hu, D., Long, T., Xiao, Y., Han, X., Gu, Y.: Fluid–structure interaction analysis by coupled fe–sph model based on a novel searching algorithm. Comput. Methods Appl. Mech. Eng. **276**, 266–286 (2014)
27. Hübner, B., Walhorn, E., Dinkler, D.: A monolithic approach to fluid–structure interaction using space–time finite elements. Comput. Methods Appl. Mech. Eng. **193**(23), 2087–2104 (2004)
28. Hwang, S.C., Khayyer, A., Gotoh, H., Park, J.C.: Development of a fully lagrangian MPS-based coupled method for simulation of fluid–structure interaction problems. J. Fluids Struct. **50**, 497–511 (2014)
29. Idelsohn, S.R., Oñate, E., Pin, F.D.: The particle finite element method: a powerful tool to solve incompressible flows with free-surfaces and breaking waves. Int. J. Numer. Methods Eng. **61**(7), 964–989 (2004)
30. Idelsohn, S., Oñate, E., Pin, F.D., Calvo, N.: Fluid–structure interaction using the particle finite element method. Comput. Methods Appl. Mech. Eng. **195**(17), 2100–2123 (2006)
31. Idelsohn, S.R., Marti, J., Limache, A., Oñate, E.: Unified lagrangian formulation for elastic solids and incompressible fluids: application to fluid–structure interaction problems via the pfem. Comput. Methods Appl. Mech. Eng. **197**(19), 1762–1776 (2008)
32. Johnson, G.R.: Linking of lagrangian particle methods to standard finite element methods for high velocity impact computations. Nucl. Eng. Des. **150**(2), 265–274 (1994)
33. Khayyer, A., Gotoh, H.: Enhancement of stability and accuracy of the moving particle semi-implicit method. J. Comput. Phys. **230**(8), 3093–3118 (2011)
34. Kohei, M., Oochi, M., Fujisawa, T., Koshizuka, S., Yoshimura, S.: Distributed memory parallel algorithm for explicit MPS using ParMETIS. Trans. Jpn. Soc. Comput. Eng. Sci. (20120012) (2012) (in Japanese)
35. Kondo, M., Koshizuka, S.: Improvement of stability in moving particle semi-implicit method. Int. J. Numer. Methods Fluids **65**(6), 638–654 (2011)
36. Koshizuka, S., Oka, Y.: Moving-particle semi-implicit method for fragmentation of incompressible fluid. Nucl. Sci. Eng. **123**(3), 421–434 (1996)
37. Lee, C.J.K., Noguchi, H., Koshizuka, S.: Fluid–shell structure interaction analysis by coupled particle and finite element method. Comput. Struct. **85**(11), 688–697 (2007)
38. LexADV. http://adventure.sys.t.u-tokyo.ac.jp/lexadv/
39. Lu, Y., Wang, Z., Chong, K.: A comparative study of buried structure in soil subjected to blast load using 2d and 3d numerical simulations. Soil Dyn. Earthq. Eng. **25**(4), 275–288 (2005)
40. Lucy, L.B.: A numerical approach to the testing of the fission hypothesis. Astron. J. **82**, 1013–1024 (1977)
41. Matthies, H.G., Steindorf, J.: Partitioned strong coupling algorithms for fluid–structure interaction. Comput. Struct. **81**(8), 805–812 (2003)
42. Matthies, H.G., Niekamp, R., Steindorf, J.: Algorithms for strong coupling procedures. Comput. Methods Appl. Mech. Eng. **195**(17), 2028–2049 (2006)

43. Minami, S., Yoshimura, S.: Performance evaluation of nonlinear algorithms with line-search for partitioned coupling techniques for fluid–structure interactions. Int. J. Numer. Methods Fluids **64**(10–12), 1129–1147 (2010)
44. Mitsume, N., Yoshimura, S., Murotani, K., Yamada, T.: Improved MPS-FE fluid-structure interaction coupled method with MPS polygon wall boundary model. Comput. Model. Eng. Sci. **101**(4), 229–247 (2014)
45. Mitsume, N., Yoshimura, S., Murotani, K., Yamada, T.: MPS-FEM partitioned coupling approach for fluid-structure interaction with free surface flow. Int. J. Comput. Methods **11**(4), 1350101, 16 (2014)
46. Monaghan, J.J.: Simulating free surface flows with sph. J. Comput. Phys. **110**(2), 399–406 (1994)
47. Murotani, K., Koshizuka, S., Tamai, T., Shibata, K., Mitsume, N., Shinobu, Y., Tanaka, S., Hasegawa, K., Nagai, E., Fujisawa, T.: Development of hierarchical domain decomposition explicit MPS method and application to large-scale tsunami analysis with floating objects. J. Adv. Simul. Sci. Eng. **1**(1), 16–35 (2014)
48. Nomura, T.: Ale finite element computations of fluid-structure interaction problems. Comput. Methods Appl. Mech. Eng. **112**(1), 291–308 (1994)
49. Oñate, E., Idelsohn, S.R., Del Pin, F., Aubry, R.: The particle finite element method - an overview. Int. J. Comput. Methods **1**(02), 267–307 (2004)
50. Oochi, M., Koshizuka, S., Sakai, M.: Explicit MPS algorithm for free surface flow analysis. Proc. Conf. Comput. Eng. Sci. **15**(2), 589–590 (2010)
51. Osher, S., Sethian, J.A.: Fronts propagating with curvature-dependent speed: algorithms based on Hamilton-Jacobi formulations. J. Comput. Phys. **79**(1), 12–49 (1988)
52. Rafiee, A., Thiagarajan, K.P.: An SPH projection method for simulating fluid-hypoelastic structure interaction. Comput. Methods Appl. Mech. Eng. **198**(33), 2785–2795 (2009)
53. Report of the JSME Research Committee on the Great East Japan Earthquake Disaster. http://www.jsme.or.jp/English/report/geje/Full%20Text.pdf
54. Ryzhakov, P., Rossi, R., Idelsohn, S., Oñate, E.: A monolithic lagrangian approach for fluid–structure interaction problems. Comput. Mech. **46**(6), 883–899 (2010)
55. Shadloo, M.S., Zainali, A., Yildiz, M., Suleman, A.: A robust weakly compressible sph method and its comparison with an incompressible sph. Int. J. Numer. Methods Eng. **89**(8), 939–956 (2012)
56. Shakibaeinia, A., Jin, Y.C.: A weakly compressible MPS method for modeling of open-boundary free-surface flow. Int. J. Numer. Methods Fluids **63**(10), 1208–1232 (2010)
57. Takizawa, K., Tezduyar, T.E.: Computational methods for parachute fluid–structure interactions. Arch. Comput. Methods Eng. **19**(1), 125–169 (2012)
58. Thiyahuddin, M., Gu, Y., Gover, R., Thambiratnam, D.: Fluid–structure interaction analysis of full scale vehicle-barrier impact using coupled sph–fea. Eng. Anal. Boundary Elem. **42**, 26–36 (2014)
59. Walhorn, E., Kölke, A., Hübner, B., Dinkler, D.: Fluid–structure coupling within a monolithic model involving free surface flows. Comput. Struct. **83**(25), 2100–2111 (2005)
60. Yamada, T., Yoshimura, S.: Line search partitioned approach for fluid-structure interaction analysis of flapping wing. Comput. Model. Eng. Sci. **24**(1), 51 (2008)
61. Yamada, Y., Sakai, M., Mizutani, S., Koshizuka, S., Oochi, M., Muruzono, K.: Numerical simulation of three-dimensional free-surface flows with explicit moving particle simulation method . Trans. Atomic Energy Soc. Jpn. **10**(3), 185–193 (2011) (in Japanese)
62. Yang, Q., Jones, V., McCue, L.: Free-surface flow interactions with deformable structures using an SPH–FEM model. Ocean Eng. **55**, 136–147 (2012)
63. Yoshimura, S., Shioya, R., Noguchi, H., Miyamura, T.: Advanced general-purpose computational mechanics system for large-scale analysis and design. J. Comput. Appl. Math. **149**(1), 279–296 (2002)