Paper I

Simulation of ground vibration on a massively data-parallel computer

M. Berglund[1] and S. Erlingsson[2]

[1]Center for Computational Mathematics and Mechanics (C^2M^2)
Royal Institute of Technology, S-100 44 Stockholm
[2]Department of Soil and Rock Mechanics
Royal Institute of Technology, S-100 44 Stockholm

ABSTRACT. A three dimensional code ELWA has been developed to solve the damped elastic wave equation in the time domain. Explicit finite difference method of order two was used for the discretization both in time and space. The code was implemented in CM Fortran on an 8K processor massively data-parallel computer CM-200 with 1 Gbyte RAM. Current performance is 334 Mflops in double precision. Two numerical examples are given for validation of the code.

1.1 Introduction

The rhythmic motion of a crowd of rock fans in a large sports arena in Gothenburg, Sweden, lead to violent structural vibrations resulting in structural damage. The incident has intensified research in elastic wave propagation in soil materials such as is found in the foundation.

The wave propagation in soil is approximated by a three dimensional linearly elastic model in which the modulus of elasticity, E, varies with depth, *i.e.* $E = E(x_3)$. The density ρ and Poisson ratio ν are constant. The damping is assumed to follow the common Rayleigh model with coefficients γ and κ. This gives opportunities to study both viscous and hysteretical types of damping by just changing the coefficients γ and κ. Thus, the governing equations are

$$\sigma_{ij,i} + K_j + \gamma \frac{\partial}{\partial t}\sigma_{ij,j} + \kappa \frac{\partial u_j}{\partial t} = \rho \frac{\partial^2 u_j}{\partial t^2}, \quad j = 1, 2, 3, \tag{1}$$

in which

$$\sigma_{ij} = \frac{E}{1+\nu}\left(\varepsilon_{ij} + \frac{\nu}{1-2\nu}\varepsilon_{kk}\delta_{ij}\right), \tag{2}$$

$$\varepsilon_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right), \tag{3}$$

where σ_{ij} is the stress tensor, u_j the j:th component in the displacement field and K_j are the external volume forces. Equation 1 is solved in the space domain

$$\Omega = \{(x_1, x_2, x_3) : 0 \le x_1 \le M_1, 0 \le x_2 \le M_2,$$

$$h(x_1, x_2) \leq x_3 \leq 0\},$$

and $t \geq 0$, see figure 1.

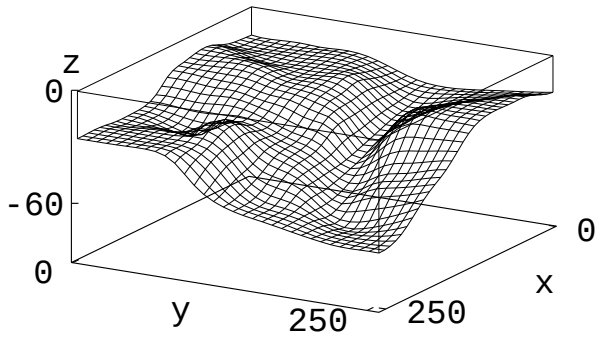

Figure 1: Geometry under sports arena "Nya Ullevi" where bedrock is approximated with splines.

The boundary condition on the upper surface, $x_3 = 0$, is

$$\sigma_{i,j} = f_j(x_1, x_2, t), \tag{4}$$

which expresses the prescribed surface tractions. If $f_j = 0$ for some j then we have absence of surface tractions in that direction. At the bedrock boundary $x_3 = h(x_1, x_2)$ displacements are zero, *i.e.*

$$u_i(x_1, x_2, h(x_1, x_2), t) = 0. \tag{5}$$

On the remaining faces of Ω absorbing boundary conditions are applied, which are paraxial approximations of order 1 of equation 1, see [CE] for details.

The initial conditions for this problem are

$$\begin{aligned} u_i(x_j, 0) &= f_0(x_j), \\ \frac{\partial}{\partial t} u_i(x_j, 0) &= g_0(x_j). \end{aligned} \tag{6}$$

This problem is of general interest in soil dynamics with all kinds of site response as application, such as earthquakes, machine vibrations, pile drivings and blastings. Also effects of moving loads like trains and trucks where the three dimensional effects are of great interest can be analyzed.

The choice of algorithm is guided by the demand of high accuracy within a realistic computation time. This is achieved by a compromise in which the computing stencil and data structure are simple enough for good flop-rate on a parallel computer but general enough for numerical accuracy. The data structure for the unknowns u_i consists of a simple (regular) three dimensional grid. This implies that the original physical domain must first be transformed to a parallelepiped by a smooth mapping. The type of applications we have in mind have domains for which this is possible. The computational stencil is explicit, *i.e.* the new value of an unknown displacement in a time-step is given by a linear combination of known old-time values at neighbors in coordinate and diagonal directions. The same structure is used for the different boundary conditions. Second order finite differences are used on transformed equations in the interior of the domain. The absorbing boundary conditions are of first order numerical accuracy which is enough for overall second order accuracy [G].

The program ELWA has been implemented in an array extension of FORTRAN 90, CM Fortran, on an 8K (8192) processors CM-200 with 1 Gbyte RAM and the current performance is 334 Mflops in double precision. The program has been validated by comparing results with an analytic solution of a one dimensional problem and with the finite-element package ABAQUS [A] in a two dimensional problem.

1.2 Numerical algorithm

Due to the simple geometry of the physical domain we can transform, by a smooth mapping, the physical domain into a parallelepiped, which becomes the computational domain, see figure 2. If the physical domain is described in coordinates (x_1, x_2, x_3) and the computational domain in (x, y, z), the smooth mapping can be described as

$$x = x_1,\ y = x_2,\ z = x_3/h(x_1, x_2). \tag{7}$$

When rewriting (1) in the new coordinates the chain rule is used to express the first order partial derivatives as

$$\begin{aligned}
\frac{\partial}{\partial x_1} &= \frac{\partial}{\partial x} - \frac{z}{h(x,y)}\frac{\partial h(x,y)}{\partial x}\frac{\partial}{\partial z}, \\
\frac{\partial}{\partial x_2} &= \frac{\partial}{\partial y} - \frac{z}{h(x,y)}\frac{\partial h(x,y)}{\partial y}\frac{\partial}{\partial z}, \\
\frac{\partial}{\partial x_3} &= \frac{1}{h(x,y)}\frac{\partial}{\partial z}.
\end{aligned}$$

Higher order derivatives can be expressed in a similar way.

Equation (1) is now given as

$$\frac{\partial^2}{\partial t^2}u_j = P_1(x,y,z)\frac{\partial^2 u_1}{\partial x^2} + P_2(x,y,z)\frac{\partial^2 u_2}{\partial x^2} + \ldots, \tag{8}$$

where $P_r(x,y,z)$, $r = 1,2,3$, are functions of $h(x,y)$, $E(z)$ and their partial derivatives of first and second order.

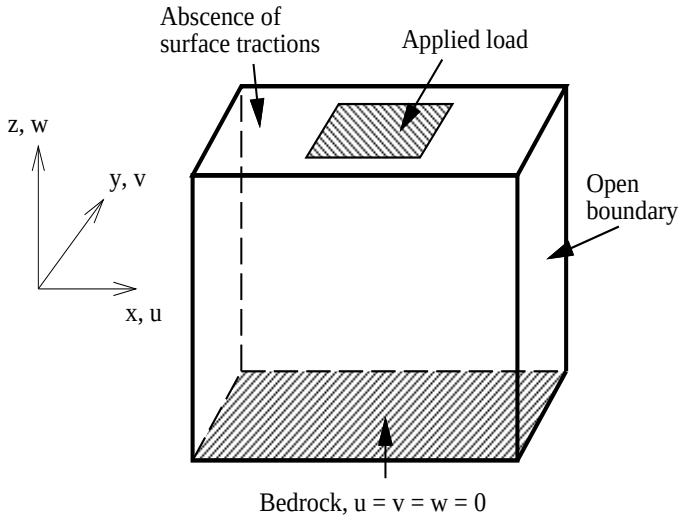

Figure 2: Computational domain with the different boundary conditions.

The computational domain is $\Omega = \{(x,y,z) : 0 \leq x \leq L_x, 0 \leq y \leq L_y, -L_z \leq z \leq 0\}$, and N_x, N_y and N_z are the number of grid-points in each direction in the three dimensional (regular) grid. Let $u_r(i,j,k)$ be the numerical approximation of $u_r(x,y,z,t)$, where $r = 1,2,3$, $x = i\Delta x$, $y = j\Delta y$, $z = k\Delta z - L_z$, $t = n\Delta t$, $\Delta x = L_x/N_x$, $\Delta y = L_y/N_y$, $\Delta z = L_z/N_z$ and Δt is the time-step. We can now describe the explicit time stepping scheme as follows

1. for each time-step
    2. for each grid-point collect displacements from all neighbor grid-points
        3. evaluate the stencil
4. if one more time-step, go to 1, else stop.

Step 2-3 consists of a single matrix multiplication with a large sparse matrix,

$$u_r^{n+1}(i,j,k) = \sum_{q=1}^{3}\sum_{s1=-1}^{1}\sum_{s2=-1}^{1}\sum_{s3=-1}^{1}\sum_{s4=-1}^{0}[m^r_{s1,s2,s3,s4,q} u_q^{n+s4}(i+s1, j+s2, k+s3)], \tag{9}$$

for $i = 0, 1, \ldots N_x$, $j = 0, 1, \ldots N_y$, $k = 0, 1, \ldots N_z$ and $r = 1, 2, 3$ and $m^r_{s1,s2,s3,s4,q}$ are weights from the discretization. The ghost points (which have at least one index $i = -1, N_x + 1$, $j = -1, N_y + 1$ or $k = -1, N_z + 1$) are updated by using the boundary conditions. On the upper part of the domain, $z = 0$, the loads and/or displacements are specified. If the load is prescribed the boundary conditions (4) are given as

$$\begin{aligned}
\frac{1}{h}\frac{\partial u_1}{\partial z} &= -\frac{\partial u_3}{\partial x} + \frac{f_1(t)}{M}, \\
\frac{1}{h}\frac{\partial u_2}{\partial z} &= -\frac{\partial u_3}{\partial y} + \frac{f_2(t)}{M}, \\
\frac{1}{h}\frac{\partial u_3}{\partial z} &= \Lambda\left(\frac{\partial u_1}{\partial x} + \frac{\partial u_2}{\partial y}\right) + \frac{f_3(t)}{M},
\end{aligned} \tag{10}$$

where $\Lambda = \nu/(\nu - 1)$ and M is the constrained modulus defined as

$$M = \frac{E(1-\nu)}{(1-2\nu)(1+\nu)}.$$

Discretizing (10) gives

$$\begin{aligned}
u_1^n(i,j,N_z+1) = u_1^n(i,j,N_z-1) - h(i,j,N_z)\Delta z/\Delta x[\\
u_3^n(i+1,j,N_z) - u_3^n(\\
i-1,j,N_z)] + \Delta z f_1(n\Delta t)/M
\end{aligned}$$

$$\begin{aligned}
u_2^n(i,j,N_z+1) = u_2^n(i,j,N_z-1) - h(i,j,N_z)\Delta z/\Delta y[\\
u_3^n(i,j+1,N_z) - u_3^n(i,j-1, \\
N_z)] + \Delta z f_2(n\Delta t)/M
\end{aligned}$$

$$\begin{aligned}
u_3^n(i,j,N_z+1) = u_3^n(i,j,N_z-1) + h(i,j,N_z)\Lambda\Delta z/ \\
\Delta x[u_1^n(i+1,j,N_z) - u_1^n(i-1, \\
j,N_z) + u_2^n(i,j+1,N_z) - \\
u_2^n(i,j-1,N_z)] + \\
\Delta z f_3(n\Delta t)/M,
\end{aligned} \tag{11}$$

for $i = 1, 2, \ldots, N_x - 1$ and $j = 1, 2, \ldots, N_y$. The displacements at the bottom of the parallelepiped are zero, *i.e.* $u_r = 0$, which gives

$$u_r^n(i,j,0) = 0, \quad r = 1, 2, 3. \tag{12}$$

On the remaining faces absorbing boundary conditions are applied. For simplicity of presentation we consider the case h and E constant. The paraxial approximation of order 1 of equation (1), see [CE], at $x = N_x\Delta x$ is,

$$\frac{\partial u_r}{\partial t} = -C_r\frac{\partial u_r}{\partial x}, \quad r = 1, 2, 3, \tag{13}$$

in which

$$C_1 = [E(\alpha + 2\beta)]^{1/2},$$
$$C_2 = C_3 = [E\beta]^{1/2},$$

where $\alpha = \nu/[\rho(1+\nu)(1-2\nu)]$ and $\beta = 1/[2\rho(1+\nu)]$. C_1 and C_2 are the P- and S-wave velocities respectively.

Discretize (13) with forward differences in time and central differences in space,

$$\begin{aligned} u_r^n(N_x+1,j,k) = u_r^{n-1}(N_x+1,j,k) - C_r\Delta t/\Delta x[\\ u_r^{n-1}(N_x+1,j,k) - \\ u_r^{n-1}(N_x,j,k)], \end{aligned} \tag{14}$$

which is used to update the ghost points (N_x+1,j,k), where $j = 1,2,\ldots,N_y-1$ and $k = 1,2,\ldots,N_z+1$. Good absorbing boundary conditions on the edges between absorbing boundary faces is constructed by rotating the coordinate system and the equations 45° and applying the paraxial approximation in the bisector direction to the adjoining faces. At the edge between faces $x = L_x$ and $y = L_y$ we get

$$\frac{\partial u_r}{\partial t} = D_r\left(\frac{\partial u_r}{\partial x} + \frac{\partial u_r}{\partial y}\right), \quad r = 1,2,3, \tag{15}$$

where $D_1 = -D_2 = [(c_1+c_2-c_3)^{1/2} - (c_1+c_2+c_3)^{1/2}]/2$, $D_3 = (c_2)^{1/2}/2$, $c_1 = E(\alpha+2\beta)$, $c_2 = E\beta$ and $c_3 = E(\alpha+\beta)$. Equation (15) is discretized as

$$\begin{aligned} u_r^{n+1}(N_x+1,N_y+1,k) = u_r^n(N_x+1,N_y+1,k)+ \\ D_r\Delta t[2u_r^n(N_x+1,N_y+1,k) - u_r^n(N_x,N_y+1,k) - \\ u_r^n(N_x+1,N_y,k)], \end{aligned}$$

for $r = 1,2,3$ and $k = 1,2,\ldots N_z+1$. The absorbing conditions allow free passage of waves incident at 90°.

1.3 Computer implementation

In the data-parallel programming model (DPPM), the operations are done on all the data with the same instruction at once [T4]. For example, matrix addition is well suited for DPPM where the matrix elements are the data and the instruction is addition. Also, explicit time stepping methods, such as the one employed here, is well suited for DPPM. The program is implemented in CM Fortran on an 8K processors CM-200 massively data-parallel computer with 1 Gbyte RAM. Timings will be given for runs in double precision on this configuration.

The most important thing when programming a massively data-parallel computer (as CM-200) is to distribute the data in a clever way onto the processors to minimize communications and do computations locally in every processor (elemental computation).

Since nearest neighbor processor or news communication is fast on a CM-200, the natural distribution with all data related to each grid-point on separate processors is well suited for our case. The main problem in our algorithm is the updating of the ghost points. This requires a lot of communication and few floating point operations per second (flops). Fortunately we can include the absorbing boundary ghost points in the computational stencil for equation (1). The zero displacement boundary condition at $z = -L_z$ can also be included in the computational stencil by setting all stencil coefficients to zero. The only ghost points left are those which belong to the upper surface $k = N_z$. The performance on this updating is 3 Mflops to be compared with 334 Mflops for the computational stencil operation when one uses a CMSSL "multi-news" communication subroutine, `pshift`. The CMSSL subroutine `pshift` takes advantage of the hypercube net and its bandwidth [T1].

On CMs with 64-bit floating point units (FPU) and version 1.0 of the CM Fortran compiler a "slice-wise" code organization should be used, which can use the FPU registers efficiently [T2], [T1]. To optimize a slicewise code one shall try to break up the code in communication and elemental computation blocks. This can be achieved by saving the shifted fields in temporaries and doing the computations after the communication. For example, the five-point stencil operator

$$u^{n+1}(i,j) = c_0 u^n(i,j) + c_1 u^n(i+1,j) + c_2 u(i-1,j) + c_3 u(i,j+1) + c_4 u(i,j-1),$$

is implemented in FORTRAN 90 as

```
a1 = CSHIFT(u, DIM=1, SHIFT= 1)
a2 = CSHIFT(u, DIM=1, SHIFT=-1)
a3 = CSHIFT(u, DIM=2, SHIFT= 1)
a4 = CSHIFT(u, DIM=2, SHIFT=-1)
u = c0*u + c1*a1 + c2*a2 + c3*a3 + c4*a4
```

This strategy did not give satisfactory results and timings on communication and elemental computation showed that communication took 3.3 times longer time than the elemental computations. Improvements of communication has been done by using multi-news subroutine, `pshift` [T1]. If the communication pattern which be used several times one may compile this pattern during run-time and then use this setup. For example the five-point stencil with `pshift`,

```
  pattern_nr = pshift_setup(4,u,ier,
& CMSSL_CSHIFT,1, 1,
```

```
& CMSSL_CSHIFT,1,-1,
& CMSSL_CSHIFT,2, 1,
& CMSSL_CSHIFT,2,-1)
   .
   .
  DO 100 i=1,no_iter
  call pshift(6,pattern_nr,ier,
& a1,u,1, 1,
& a2,u,1,-1,
& a3,u,2, 1,
& a4,u,2,-1)
  u = c0*u + c1*a1 + c2*a2 + c3*a3 + c4*a4
   .
   .
```

performs the communication faster. Using `pshift` decreases the communication time by 50 % compared with ordinary communication routines (like `cshift`). It is however still the communication which takes the main part of the time in the algorithm. In fact it is the (general) communication which is the main problem to do in an optimal and efficient way on massively data-parallel computers.

1.4 Soil dynamical application

In June 1985 two rock concerts were held at the soccer stadium "Nya Ullevi" in Gothenburg, Sweden. The artist Bruce Springsteen and his group succeeded in exciting the sixty thousand present so they started to jump periodically to the rhythm of the music. One half of the audiences were standing on the stands and the other half on the field in front of the stage.

Both the ground in and around the stadium as well as the stadium itself started to vibrate. People living several hundred meters from the stadium felt strong shaking motions due to the propagated waves through the soil. The roof of the stadium which forms a thin oval structure around the field was vibrating heavily. After the concerts damages were observed in the roof [S].

Gothenburg city is built on a very thick river stratum. The deposit material is uniform soft high-plastic clay. The deposit thickness varies under the stadium from less than 10 m to more than 60 m with many irregularities, see figure 1. This results in a highly three dimensional geometry which gives rise to effects where the propagated waves can locally give high displacement amplitudes. This can only be analyzed in a three dimensional code.

This kind of analysis gives valuable information about the vibration levels in the area, how the waves propagate and focus in the soil due to the complex geometry. Further this kind of analysis can estimate applied forces and movements which affected the structure through its foundation and pile system. Energy was

presumable infiltred through the structural basement and transmitted through the overall structure where local resonance phenomena occurred.

The dynamic characteristics of Gothenburg clay was studied by Andreasson [An]. Based on his results the unit weight ρ of the clay material can be assumed to be 1550 kg/m^3, nearly constant with depth. The Young's modulus E on the other hand increases with depth and can be estimated from $E = -1.39z + 16$, giving E in MPa when z is in meters. Poisson's ratio was estimated to $\nu = 0.495$. The damping characteristics were studied in a resonant column apparatus and in the low strain amplitude range $\gamma < 10^{-5}$ the material damping constant D was estimated to be 0.02.

The load consists of thirty thousand people jumping on the ground in front of the stage. There are approximately 4–5 persons/m^2 leading to a load amplitude of 3.0 kPa. Pernica has studied dynamic live loads at a rock concerts [P]. He found out that audiences can easily produce rhythmic forces in the frequency range 1–3 Hz. Sahlin estimated the frequency to be in the 2.2–2.5 Hz range for the "Nya Ullevi" concerts [S].

1.5 Computational results

Two simple examples are presented here which were used to check the code ELWA. Assuming harmonically varying displacements with constant amplitude on the whole surface of the cube the problem reduces to an one dimensional problem. The damping was assumed to be 2 %. Periodical boundaries were applied in the ELWA code to simulate the infinite boundaries. In the other example the displacement on the surface is assumed to vary harmonically with time and as $\cos(2\pi x/L_x + 1)$ in space. The damping was still assumed to be 2 %.

The simplicity of the first model has the advantages that an analytical solution is available to check the numerical solution. In the second example the ELWA code solution is compared with a solution based on the finite element program ABAQUS where absorbing boundaries according to Lysmer and Kuhlemeyer have been used [LK].

Depth in m	3	6	9
Analytical solution	8.1955	6.2621	4.2302
ELWA 32 grid-points	8.1916	6.2558	4.2240

Table 1: Comparison of maximum amplitude between ELWA and an 1-D analytical solution.

The results of the one dimensional problem is given in table 1. Maximum amplitude of the solution in the middle of the cube at 3, 6 and 9 m depth in a 15 m thick soil layer are presented. All values are in mm. As can be seen, the results are quite satisfactory.

The results of the second example is given in figure 3 to 6. Result from the ELWA code is given in figure 3 and 4. In figure 3 both the horizontal (solid) and vertical (dashed) displacements are shown as a function of time. In figure 4 the horizontal displacement component is plotted as a function of the vertical one. The solution oscillate into a periodic solution as it shall do. For comparison both the vertical and the horizontal displacement amplitude are shown as a function of time in figure 5 and 6 for the ELWA code as well as for the ABAQUS runnings for two points at 2 respectively 3 m depth from the surface where the total depth was 15 m. In the ELWA code 32^3 grid-points where used. In the ABAQUS runnings 32×30 plain strain 4 node bilinear elements were used. The vertical displacements are almost identical. The horizontal displacement component are basically identical the first 0.2 seconds. However after that a slight difference can bee seen between the two solutions probably due to the different absorbing boundaries (paraxial approximation in the ELWA code and Lysmer and Kuhlemeyer type of boundaries in the ABAQUS code).

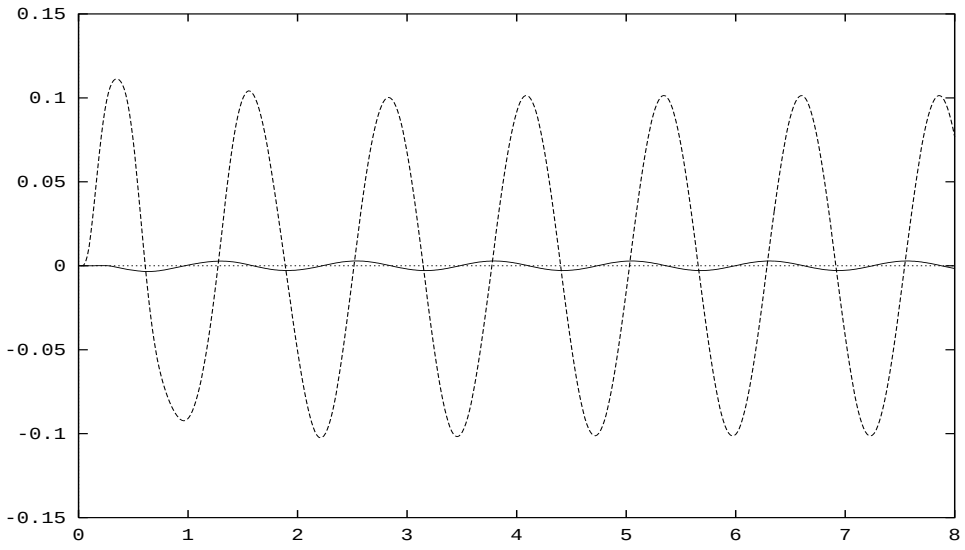

Figure 3: Horizontal (solid) and vertical (dashed) displacements as a function of time with 2% damping.

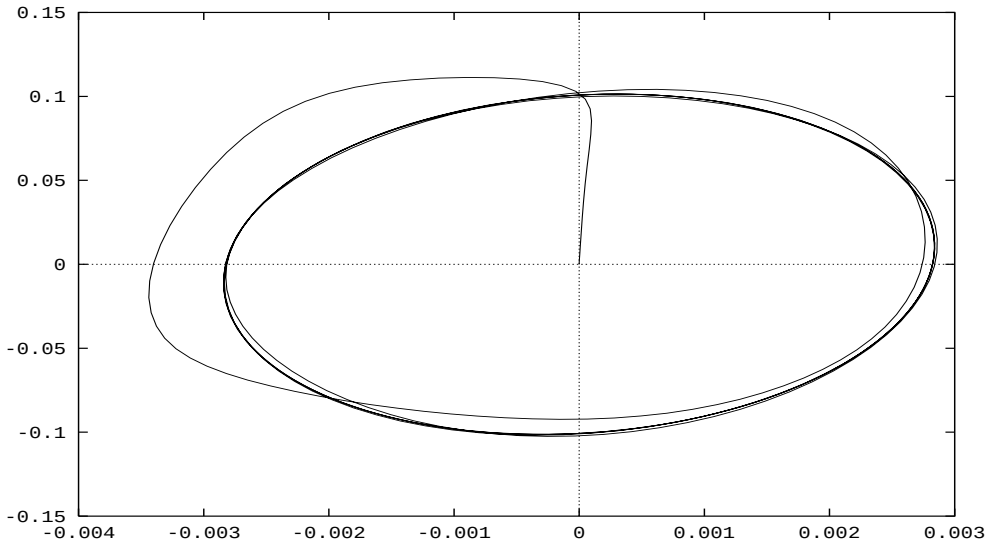

Figure 4: Horizontal versus vertical displacement with 2% damping.

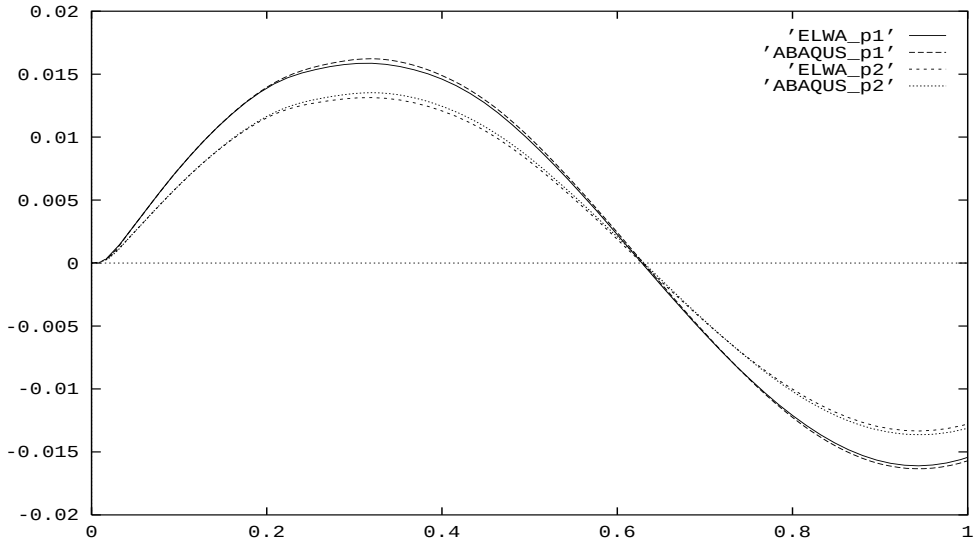

Figure 5: Comparison of vertical displacements between ELWA and ABAQUS.

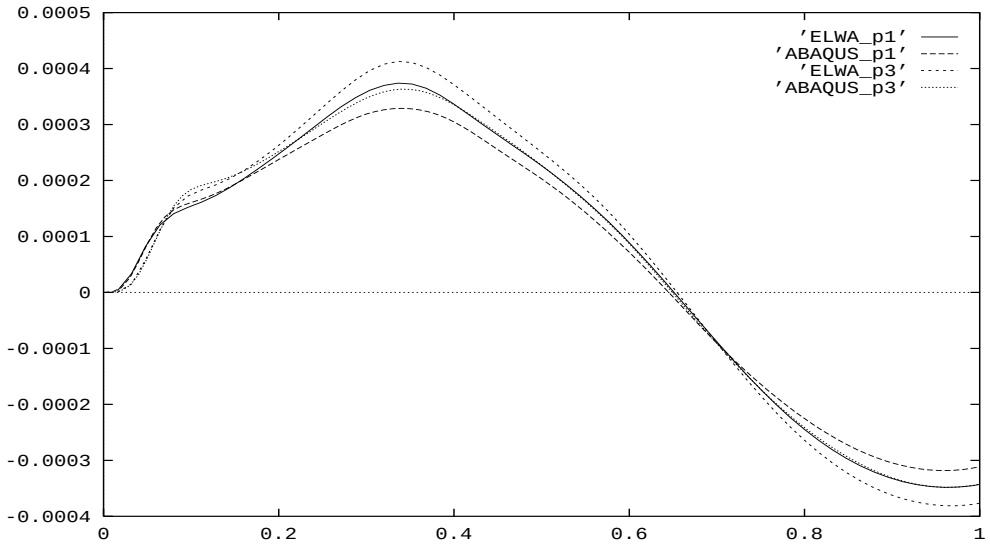

Figure 6: Comparison of horizontal displacements between ELWA and ABAQUS.

1.6 Conclusion

A computer code ELWA has been developed for the simulation of elastic wave propagation in soil material. The numerical algorithm was designed for efficient implementation on parallel computers. A performance of 334 Mflops in double precision was achieved on an 8K CM-200. With this efficiency realistic three dimensional cases with variable geometry can be simulated. The accuracy of the code with different boundary conditions has been evaluated in comparison with other codes and with known analytical solutions.

1.7 Acknowledgment

The Swedish Institute for Applied Mathematics (ITM) and the Swedish Council for Building Research (BFR) for their financial supports. B. Engquist, J. Oppelstrup, F. Hedman and A. Bodare for their contributions.

References

[A] Hibbitt, Karlson & Sorensen, Inc. *ABAQUS User Manual. Version 4.8.*

[An] B. Andreasson. *Deformation Characteristics of Soft High-Plastic Clays under Dynamic Loading Conditions.* PhD Thesis, Chalmers University of Technology, Göteborg, Sweden, 1979.

[CE] R. Clayton and B. Engquist. *Absorbing boundary conditions for acoustic and elastic wave equations.* Bulletin of the Seismological Society of America, 67(6):1519–1540, December 1977.

[G] B. Gustafsson. *The convergence rate for difference approximations to general mixed initial boundary value problems.* SIAM J. Num. Analysis, 18:179–190, 1981.

[LK] J. Lysmer and R. L. Kuhlemeyer. *Finite Dynamic Model for Infinite Medium.* ASCE, Journal of the Engineering Mechanics Division, 95(4):859–877, 1969.

[P] G. Pernica. *Dynamic live loads at a rock concerts.* Canadian Journal of Civil Engineering, 10(2):185–191, 1988.

[S] S. Sahlin. *On Site Measurements of Soil and Structure Response of the Soccer Stadium "Nya Ullevi" i n Göteborg.* In *A course in Fundamentals of Earthquake Engineering 1989.* Statens Provningsanstalt, 1989.

[T1] Thinking Machines Corporation. *CMSSL for CM Fortran. Version 2.2.*

[T2] Thinking Machines Corporation. *CM Fortran Reference Manual. Version 1.0 and 1.1.*

[T3] Thinking Machines Corporation. *CM User's Guide. Version 6.0.*

[T4] Thinking Machines Corporation. *Getting started in CM Fortran.*