PHYSICAL REVIEW D **91,** 044033 (2015)

Binary black hole simulation with an adaptive finite element method: Solving the Einstein constraint equations

Zhoujian Cao[*]

Institute of Applied Mathematics and LSEC, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, China

(Received 15 December 2014; published 19 February 2015)

In this paper we start a systematic investigation of applying an adaptive finite element method to the Einstein equations, especially binary compact object simulations. To our knowledge, this is the first study on this topic. Puncture type initial data are solved with the adaptive finite element method. The numerical scheme proposed in the current work can be straightforwardly extended to the general type of initial data of the Einstein equations. The Parallel Hierarchical Grid library and the existing numerical relativity code AMSS-NCKU are used to develop the adaptive finite element Einstein solver. In the unsmooth toy model problem, the adaptive mesh refinement operation can catch the unsmooth region efficiently. The numerical solution deviates the exact solution by an error less than 10^{-5}. In the binary black hole problem, our solution is consistent with the one gotten by the TwoPuncture code which uses a pseudospectral method. As we expected, the solution gotten by the finite element method is less accurate than that gotten by the spectral method. But the relative error distributes almost uniformly. The adaptive mesh refinement method is quite efficient and it does not waste computational effort. Our finite element code is more flexible than the TwoPuncture code. It can be used to treat other general initial data problems such as the three black holes problem, besides the binary black hole problem. We test one typical three black holes problem also. In all of the test cases, our adaptive finite element code works quite well.

DOI: 10.1103/PhysRevD.91.044033 PACS numbers: 04.25.D-, 04.30.Db, 04.70.Bw, 95.30.Sf

I. INTRODUCTION

Numerical methods to solve partial differential equations include three categories. They are the finite difference method, spectral method, and finite element method. Among the existing numerical relativity codes relating to the Einstein equations, most of them use the finite difference method [1–3] and a few of them use the spectral method [4]. But unfortunately the finite element code for numerical relativity is still missing (but see [5–7]).

Based on the behavior of the existing numerical relativity codes, we can briefly summarize the advantages of the finite difference method and spectral method, respectively, as follows. The finite difference method is more robust and easier to code. Especially, it can be used to treat the coupling problem of dynamical spacetime and hydrodynamics directly. The spectral method is more accurate and more efficient than the finite difference method. But when the field in the question is less smooth, the accuracy advantage of the spectral method becomes weak. And more, the spectral method is hard to implement when a discontinuity appears. That is the reason why the spectral method is difficult for treating coupled Einstein equation and fluid equations.

The finite element method has many possible advantages. The finite element discretization admits the local property as a finite difference. In each element, meanwhile, the high order polynomial function basis and/or spectral function basis can be used, which is similar to the spectral method (spectral element method). So it is possible to use the finite element method to treat an unsmooth region with a small element (h refinement) and to treat a smooth region with a large element but high order basis or spectral basis (p refinement). This may combine the advantages of both the finite difference method and spectral method. Due to the global data property of the spectral method, the parallelization for the spectral method is quite limited. Within the finite difference method, the moving box style mesh refinement technique is commonly used in numerical relativity. And the data have to be transferred between different mesh levels. So the strong parallelization scalability for the finite difference method is limited by the size of the individual mesh level. In contrast, all elements in the finite element method are treated uniformly. This possibly makes the finite element method admit higher strong parallelization scalability than both the finite difference method and spectral method.

Although the finite element method has the above attractive properties, it does have a drawback. That is, the finite element method is hard to code, especially for large scale scientific computation such as simulations of binary compact objects. In [5,6] the authors have used the finite element method to treat scalar equations and Regge-Wheeler equations on the fixed Schwarzschild background. They took the advantage of the symmetry of the problems and reduced the equations to 1-D and 2-D in space,

[*]zjcao@amt.ac.cn

respectively. The results of [5,6] have already shown accuracy, efficiency, and flexibility advantages of the finite element method. In this project we exploit one recently developed adaptive finite element library—Parallel Hierarchical Grid (PHG) [8,9]—to develop a numerical relativity code and investigate the properties of the finite element method in solving the Einstein equations. We will treat the full Einstein equations in 3-D space. In the current paper we investigate the behavior of the finite element method in solving the constraint part equations. This will be the basis for us to develop a finite element code to evolve the Einstein equations. The puncture method will be adopted in this project. Since puncture is a singular point, we will pay special attention to the behavior of the finite element method around this object.

The rest of the paper is arranged as follows. In the next section we will introduce the numerical algorithm treating the Einstein constrain equations with the finite element method. Then in Sec. III we briefly describe the code we developed in this work. After that we present the numerical results in Sec. IV. Firstly we show the results for a toy model problem which admits an exact solution. Following that we present the results for the binary black hole problem and three black holes problem. At last we close the paper with a summary and a discussion in Sec. V.

The Einstein summation convention, i.e., the repeated super- and subindex mean summations, is adopted throughout the paper. And the geometrized units with $G = c = 1$ are used. We take the notation convention used in [10,11]. Latin indices are spatial indices and run from 1 to 3, whereas greek indices are space-time indices and run from 0 to 3.

II. EINSTEIN CONSTRAINT EQUATIONS AND NUMERICAL ALGORITHM

Through $3 + 1$ decomposition, the Einstein equations are split into the evolution part and the constraint part [2]. The constraint part reads as

$$\mathcal{H} \equiv R - K_{ij}K^{ij} + K^2 - 16\pi\rho = 0, \tag{1}$$

$$\mathcal{M}_i \equiv D_j K^j{}_i - D_i K - 8\pi s_i = 0, \tag{2}$$

which are named the Hamiltonian and momentum constraints, respectively. In the above equations, the R and K_{ij} are the scalar curvature and extrinsic curvature with respect to some given spatial slice. D is the covariant derivative operator consistent with the spatial metric. ρ and s_i are the mass density and momentum density of matter. We refer our reader to [2] for a detailed description of the quantities involved in the above constraint equations.

In order to get the initial data for the evolution part, one has to solve these constraint equations. Based on different physical scenario considerations, different solution schemes have been proposed [12] to solve the constraint equations. The existing initial data solvers for binary black holes include finite difference solvers and pseudospectral solvers. The bifunctional adaptive mesh code's elliptic solver [13], OLLIPTIC code [14], and AMRMG code [15] belong to the finite difference solver. All of these solvers use a full approximation storage multigrid method and a nonlinear Gauss-Seidel relaxation scheme. OLLIPTIC code uses fixed refinement levels, while AMRMG adaptively refines the mesh levels based on error estimate. OLLIPTIC code uses the refinement mesh levels together with some extra coarse levels to do multigrid operation. In contrast, AMRMG solves the equation on each mesh level through the multigrid method. The pseudospectral solvers include LORENE [16,17], the Spectral Einstein Code (SpEC) [18], and the TwoPuncture code [19]. The TwoPuncture code is specifically designed for puncture type initial data [13]. A series of meticulous coordinate transformations is involved in the numerical scheme of the TwoPuncture code. Regarding numerical implementation, only one spectral domain is used in the TwoPuncture code. SpEC was developed by Pfeiffer and his co-workers. It is mainly used for black hole simulations. Different from the TwoPuncture code, SpEC is a typical multidomain spectral code. Although most published works used SpEC for conformal thin-sandwich initial data with excision, SpEC can be used in principle to solve any type of initial data for the Einstein equations. At the end of the 1990s, LORENE was developed by Gourgoulhon, Grandclement, and other co-workers mainly for relativistic compact stars. But it can also be used for binary black hole initial data. Like SpEC, LORENE is also a multidomain spectral code. In this paper we will construct an adaptive finite element solver for the initial data of the Einstein equations.

We consider the puncture scheme proposed in [13] with conformally flat assumption. In this scheme, the momentum constraints are solved analytically. After that the 3-metric and the extrinsic curvature are written as

$$\gamma_{ij} = \psi^4 f_{ij}, \tag{3}$$

$$K_{ij} = \psi^{-2}\hat{K}_{ij}, \tag{4}$$

$$\hat{K}_{ij} = \frac{3}{2}\sum_I \frac{1}{r_I^2}[2P^I_{(i}n^I_{j)} - (f_{ij} - n^I_i n^I_j)P^k_I n^I_k + \frac{4}{r_I} n^I_{(i}\epsilon_{j)k\ell} S^k_I n^\ell_I], \tag{5}$$

where f_{ij} is the flat metric; P^i_I and S^i_I are constant vectors, standing for the linear momentum and the spin momentum of the Ith black hole, respectively; n^i_I is the radial normal vector with respect to f_{ij}, which points from the position of the Ith black hole to the space point. And the conformal factor ψ is determined by the Hamiltonian constraint equation

$$-(\partial_x^2+\partial_y^2+\partial_z^2)\psi=\frac{1}{8}\hat{K}^{ij}\hat{K}_{ij}\psi^{-7}+2\pi\rho\psi^{-3}. \tag{6}$$

The points $r_I=0$ are called puncture points and the above equation is singular at these points. After variable transformation

$$\psi\equiv 1+\sum_I\frac{m_I}{2r_I}+u, \tag{7}$$

the Hamiltonian constraint equation becomes

$$-(\partial_x^2+\partial_y^2+\partial_z^2)u=\frac{1}{8}\hat{K}^{ij}\hat{K}_{ij}\psi^{-7}+2\pi\rho\psi^{-3} \tag{8}$$

with unknown function u. Approaching puncture points, r_I goes to zero and $\hat{K}_{ij}$ behaves as r_I^{-1}. Noting that ψ behaves as r_I^{-1}, we can see that the right-hand side of (8) approaches zero, which means that the equation behaves well at puncture points. Actually, the authors in [13] have proved that the equation is regular in the whole domain $\mathbb{R}^3$. But unfortunately the solution u is only C^4 around the puncture points.

When r goes to infinity, the geometry approaches a flat space asymptotically. This results in the boundary condition for (8):

$$u\to 0 \quad \text{when } r\to\infty. \tag{9}$$

Numerically we will approximate the domain $\mathbb{R}^3$ with some finite domain Ω. At the boundary of Ω if we consider the zeroth order approximation of (9), we get the following approximate Dirichlet boundary condition,

$$u=d(x) \quad \text{at } \partial\Omega, \tag{10}$$

with $d=0$. If we consider the leading order approximation of (9), $u\approx\frac{A}{r}$ with an unknown constant A, we get the following approximate Robin boundary condition,

$$\vec{n}\cdot\nabla u+\alpha(x)u=b(x) \quad \text{at } \partial\Omega, \tag{11}$$

with $b=0$ and $\alpha=\frac{1}{r}\frac{\partial r}{\partial n}$. Here $r=\sqrt{x^2+y^2+z^2}$ and $\vec{n}$ is the out normal vector of $\partial\Omega$.

Typically, the metric shows a multipole expansion behavior asymptotically when r goes to infinity. The leading order corresponds to the monopole order and it shows a power-law decay in $\frac{1}{r}$ [20]. More elaborative analysis about the approximate boundary condition based on multipole expansion can be found in [21,22]. Although the above approximate boundary condition (11) only considers the leading order, previous works [20,22,23] have shown that they can approximate the ultimately desired boundary condition (9) well when the computational boundary is far away enough. The boundary condition (10) is more rough. It contains only zeroth order information of the asymptotical behavior of the metric. In the latter part of this paper, our numerical test results will show that this boundary condition can also work well. Of course, this result has taken advantage of the high efficiency of the finite element method which pushes the computational boundary quite far away.

In the following we will use the finite element method to solve Eq. (8). In this paper we only consider the vacuum case $\rho=s_i=0$.

Equation (8) takes the form

$$-\nabla^2u=f(u) \text{ in } \Omega, \tag{12}$$

which is a nonlinear Poisson equation. For the nonlinear Poisson equation (12) and the Robin boundary condition (11), we can use integration by part to get the following weak form,

$$\int_\Omega(\nabla u)\cdot(\nabla v)d^3x+\int_{\partial\Omega}\alpha uvds=\int_\Omega f(u)vd^3x+\int_{\partial\Omega}bvds. \tag{13}$$

Denoting the basis function of the finite element ϕ_i, we can expand the unknown function u as $u=u^i\phi_i$. The above weak form equation can be discretized as

$$u^i\left[\int_\Omega(\nabla\phi_i)\cdot(\nabla\phi_j)d^3x+\int_{\partial\Omega}\alpha\phi_i\phi_jds\right]=\int_\Omega f(u)\phi_jd^3x+\int_{\partial\Omega}b\phi_jds. \tag{14}$$

This is a set of nonlinear algebra equations for u^i. We use the Newton iteration method to solve this set of nonlinear equations. Given the numerical solution at the nth step $u_{(n)}$, we solve

$$[F'_{ij}-M_{ij}-\alpha_{ij}]\Delta u^j=\int_\Omega(\nabla u_{(n)})\cdot(\nabla\phi_i)d^3x+\int_{\partial\Omega}\alpha u_{(n)}\phi_ids-\int_\Omega f(u_{(n)})\phi_id^3x-\int_{\partial\Omega}b\phi_jds \tag{15}$$

for Δu^i, where we have defined the mass matrix M_{ij}, and the matrices F'_{ij} and α_{ij} as

$$M_{ij}=\int_\Omega(\nabla\phi_i)\cdot(\nabla\phi_j)d^3x, \tag{16}$$

$$F'_{ij}=\int_\Omega\frac{df}{du}(u_{(n)})\phi_i\phi_jd^3x, \tag{17}$$

$$\alpha_{ij}=\int_{\partial\Omega}\alpha\phi_i\phi_jds. \tag{18}$$

Then we iterate $u_{(n+1)} = u_{(n)} + \Delta u^i \phi_i$. Noting that all the involved matrices are symmetric, we use the preconditioned conjugate gradient to solve the linear equations. The diagonal elements of the matrix in question are used as the preconditioner.

For the Dirichlet boundary condition (10), we firstly use the function basis of the boundary elements to solve the boundary condition and then use a source term to treat the boundary condition. In this way we get the following weak form,

$$\int_\Omega (\nabla u)\cdot(\nabla v) d^3x = \int_\Omega f(u) v d^3x - \sum_j \int_{\partial\Omega} d\phi_j ds, \tag{19}$$

where the ϕ_j's are the function basis of boundary elements. Using the Newtonian iteration method similar to the Robin boundary case, we can solve the equation numerically.

III. CODE DESCRIPTION

The Parallel Hierarchical Grid library [8] is an adaptive finite element software. It is developed by the finite element research group in the State Key Laboratory of Scientific and Engineering Computing. This software is constructed through c language and the Message Passing Interface library. The kernel of this software provides an infrastructure for a distributed three-dimensional hierarchical finite element grid. The PHG refines the grid and redistributes the grid to keep the load balance automatically. Along with the grid hierarchy, the PHG also provides kinds of linear equation solvers and eigen equation solvers. In addition, the PHG provides the input and output interface to support Visualization Toolkit [24], OpenDX [25], and other formats.

For the Einstein equations part, most functions involved in this work are borrowed from the AMSS-NCKU numerical relativity code. The AMSS-NCKU code has been tested and applied to many astrophysical research problems in previous studies [26–29]. We refer our readers to [2,14,30] for the code details. In addition, the AMSS-NCKU code was also used to check the resulting initial data in this paper.

In this series of works, we plan to combine AMSS-NCKU and PHG to construct an Einstein Parallel Hierarchical Grid code (iPHG) for numerical relativity. This code will be used to study the finite element method specifically for the Einstein equations. And it will also be applied to the study of gravitational wave physics and astrophysics related to general relativity.

In order to check and compare the resulting initial data for the Einstein equations, we also reproduce the TwoPuncture code by following exactly the recipes described in [19]. We have reproduced the relevant results shown in [19] to make sure our TwoPuncture code works correctly. As an example, we show a result in Fig. 1. Regarding the numerical resolution, we use the grid numbers $n_A = 40$, $n_B = 40$, and $n_\phi = 20$. The numerical error tolerance is set as 5×10^{-12}. The meaning of these parameters can be found in [19]. This setting is a typical setting used in previous binary black hole studies (e.g., [14]).

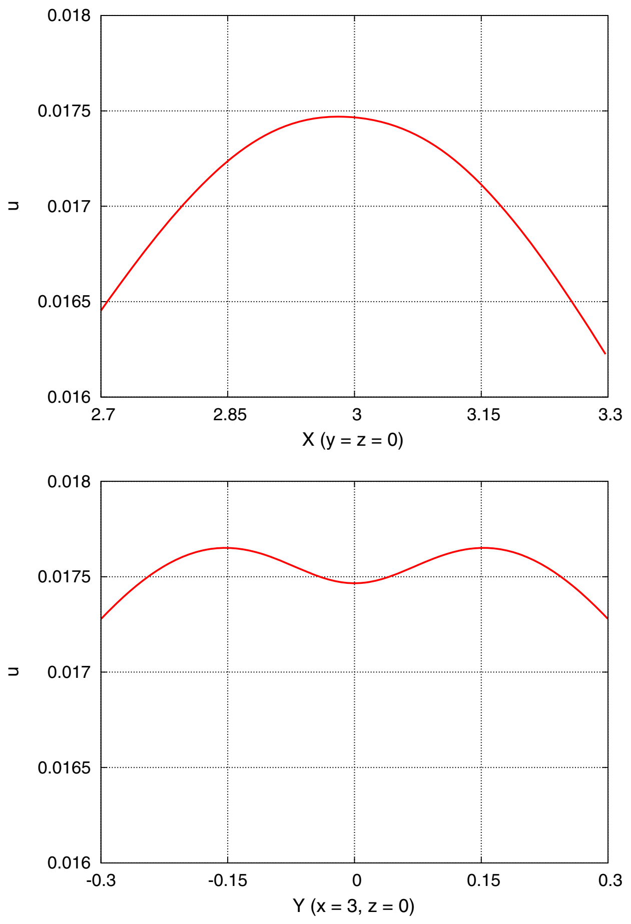

FIG. 1 (color online). Reproduction of Fig. 5 in [19] with our TwoPuncture code. Here we use the grid numbers $n_A = 40$, $n_B = 40$, and $n_\phi = 20$. The numerical error tolerance is set as 5×10^{-12}.

IV. NUMERICAL RESULTS

In this section we will present the behavior of the adaptive finite element method in solving the Einstein constraint equations. For code test and later comparison we firstly test a toy model of a linear Poisson equation.

A. Toy model

In our toy model we consider a Poisson equation with a discontinuous source:

$$-\nabla^2 u = f(x) \text{ in } \mathbb{R}^3, \tag{20}$$

$$f = \begin{cases} 1 & \text{if } r < 0.5 \\ 0 & \text{if } r > 0.5 \end{cases}. \tag{21}$$

The exact solution to this toy model is

$$u = \begin{cases} -\frac{r^2}{6} + \frac{1}{8} & \text{if } r < 0.5 \\ \frac{1}{24r} & \text{if } r > 0.5 \end{cases}. \tag{22}$$

In this paper, we always take the computational domain $r > 0.5$. Based on the above exact solution, we can deduce the Dirichlet boundary condition as

$$u = \frac{1}{24r}, \tag{23}$$

where $r = \sqrt{x^2 + y^2 + z^2}$, and the Robin boundary condition as

$$\vec{n} \cdot \nabla u + \frac{u}{r}\frac{\partial r}{\partial n} = 0. \tag{24}$$

This Robin boundary condition is nothing but Eq. (11) with $b = 0$.

Firstly we set the computational domain as a cube with $-1 \le x, y, z \le 1$. Regarding the grid, we start from the simplest tetrahedral element decomposition with 8 vertices and 5 elements. Then we uniformly refine the grid 6 times. After that we refine the boundary elements 3 times. Afterwards, we use the L_2 norm of the residual of the equation as the refinement indicator and let PHG do the adaptive refinement along the solving process. Specifically, the residual R and its L_2 norm E_i are defined as

$$R \equiv \nabla^2 u_h + f(x), \tag{25}$$

$$E_i \equiv \sqrt{\int_{\text{i-th element}} R^2 dx^3}, \tag{26}$$

where we have used u_h to denote the numerical solution. In the definition of E_i the integration is taken on the ith element. So our indicator E_i is a function of elements.

In Fig. 2 we plot the numerical solution against the exact solution. Both results with Dirichlet boundary condition (23) and Robin boundary condition (24) are shown. From this figure we can see our numerical solutions recover the exact solution very well.

In Fig. 3 we investigate the numerical error and the convergence behavior of the numerical solutions. In panel (a) of this figure, we show the difference between the numerical solutions and the exact solution (22). We calculate the difference on the mesh grid first. Then we interpolate the result to the x-axis as shown in the figure. In panel (b), we show the infinity norm of E_i with respect to the adaptive mesh refinement steps n. As we explained

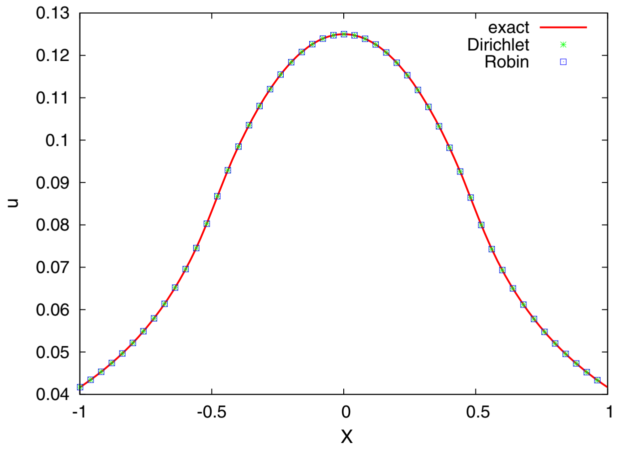

FIG. 2 (color online). Comparison of numerical solution with the Dirichlet boundary condition (23) and Robin boundary condition (24) to the exact solution (22).

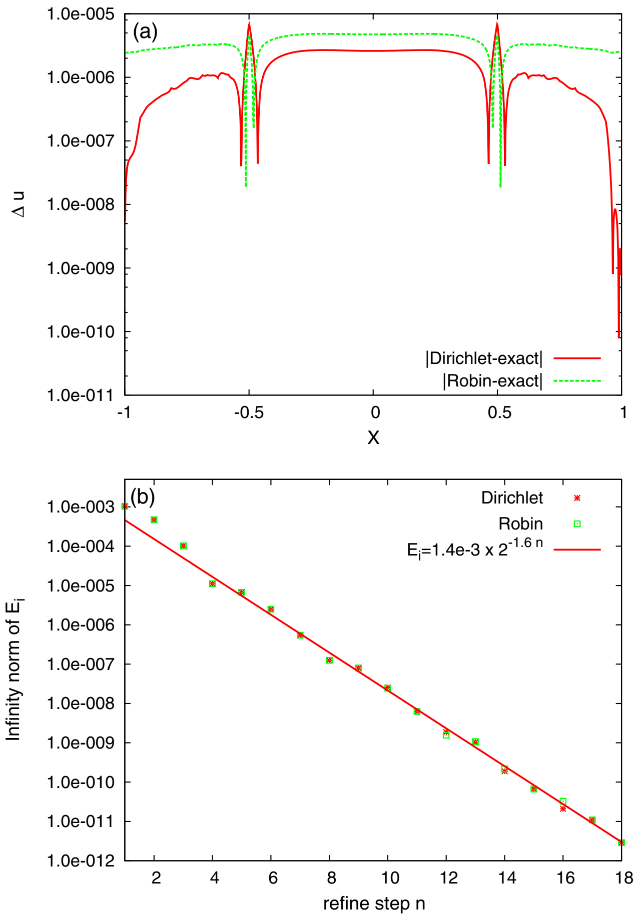

FIG. 3 (color online). (a) Difference between the numerical solution and the exact solution (22). The Dirichlet boundary condition (23) and the Robin boundary condition (24) are used, respectively. (b) Convergence behavior during the solving process. The x-axis is the mesh refinement steps n. The effective resolution is proportional to $\frac{1}{2^n}$.

above, E_i is a function of elements. We search through all the elements to find the maximal $|E_i|$ which corresponds to the infinity norm. Since we have set the numerical error tolerance 10^{-11}, the refinement process stops at the 18th step when the numerical error indicator, the infinity norm of E_i, becomes smaller than the given tolerance. If we denote the initial resolution as Δ, the effective resolution at the nth refinement step is $\Delta/2^n$. We can estimate the convergence order according to the relationship $E_i \propto (\frac{\Delta}{2^n})^p$, where we have denoted the convergence order with p. Together with the numerical results in Fig. 3(b), we also show the fitted line which implies that the convergence order is about 1.6.

In the practical initial data problem for the Einstein equations, we can neither calculate the whole $\mathbb{R}^3$ domain with boundary condition (9), nor have enough information to get a Dirichlet boundary condition like (23). Instead, the Dirichlet boundary (10) with $d = 0$ and the Robin boundary condition (11) with $b = 0$ are popularly used. The Robin boundary condition has already been tested in (24).

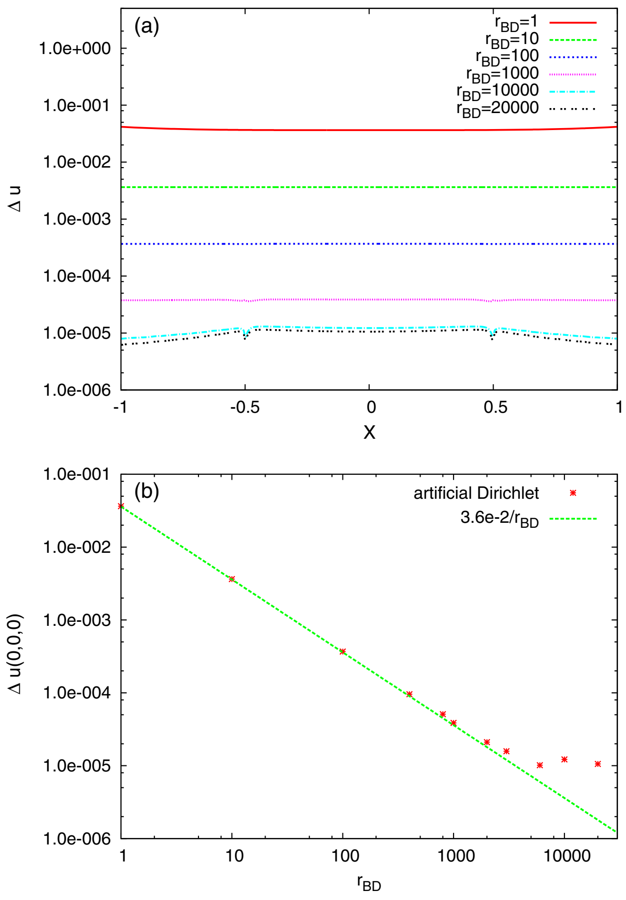

FIG. 4 (color online). Investigation of the artificial Dirichlet boundary condition (10). (a) Difference between the numerical solution and the exact solution (22). Different computational domains $-r_{BD} < x, y, z < r_{BD}$ are tested. (b) Convergence behavior with respect to r_{BD}. Here we have used the difference between the numerical solution and the exact solution (22) at point (0,0,0) to measure the numerical error.

Here we would like to test the Dirichlet boundary (10) with $d = 0$. For clarity, we call this boundary condition the artificial Dirichlet boundary condition. In the following we will solve Eq. (20) together with the artificial Dirichlet boundary condition and check how well it approximates the exact boundary condition.

The cube domain $-r_{BD} \le x, y, z \le r_{BD}$ with different r_{BD}'s is used. The difference between the numerical solutions and the exact solution (22) is shown in Fig. 4(a). In panel (b) of this figure we present the convergence behavior with respect to r_{BD}. We find that the numerical error decays as a power law in $\frac{1}{r_{BD}}$. When r_{BD} is larger than about 6×10^3, the power law stops. This is because the numerical truncation error dominates afterwards. Note that the number 10^{-5} is consistent with the numerical truncation error shown in Fig. 3(a).

In Fig. 5 we check the grid refinement configuration. In the top subplot of Fig. 5 we show the grid structure before the adaptive mesh refinement takes place. In the bottom subplot we show the grid structure after the adaptive mesh refinement is done and the numerical solution which satisfies the given error tolerance has been gotten. Note that we use different plotting scales for these two subplots. In the top subplot the size of the elements is much larger than the whole plotting scale shown in the bottom subplot. In the bottom subplot, besides the fine grids which are added by the refinement operation automatically, we can see a clear circle structure of the grids around $r = 0.5$. Recalling that the position $r = 0.5$ corresponds to the location of the discontinuity of the source function in the toy model (20), this figure indicates that our adaptive mesh refinement can effectively catch the character of the problem being solved.

B. Initial data for two punctured black holes

Now we turn to the Einstein constraint equations. We use H to denote the residual of Eq. (8),

$$H \equiv \nabla^2 u_h + \frac{1}{8}\hat{K}^{ij}\hat{K}_{ij}\psi^{-7} + 2\pi\rho\psi^{-3}, \qquad (27)$$

where we have again used u_h to denote the numerical solution. For vacuum spacetime, we have $\rho = 0$. Compared to the toy model problem in the above subsection, the solution function for Eq. (8) is smoother. So we can use the H_1 norm, instead of L_2 norm, of H in each element to indicate the numerical error of the numerical solution. Explicitly, the error indicator reads as

$$E_i \equiv \sqrt{\int_{\text{i-th element}} (H^2 + \nabla H \cdot \nabla H) dx^3}. \qquad (28)$$

The integration is taken on the ith element, which is similar to Eq. (26). For the Einstein constraint equations, we use E_i

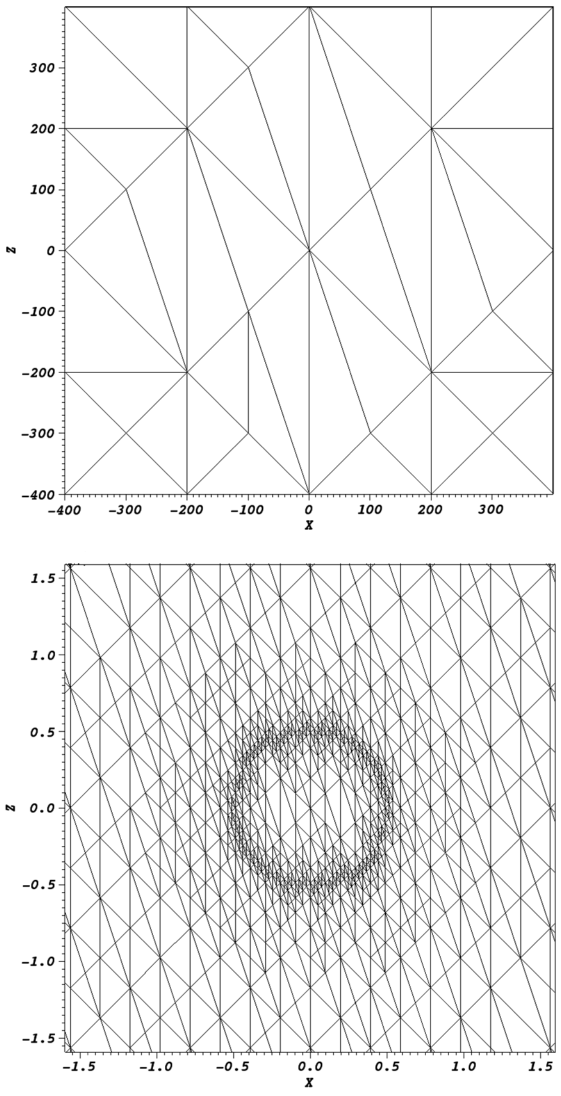

FIG. 5. Grid structure in the plane $y = 0$ for solving the toy model (20). Top: Before the adaptive mesh refinement. Bottom: After the adaptive mesh refinement.

in (28) as an indicator for adaptive mesh refinement. We have also tested to use the L_2 norm of H as the indicator and gotten roughly the same numerical result. In the following we only report the result with indicator (28).

For easy comparison we consider the configuration of two punctured black holes, which has been investigated before in [14,19]. In this configuration the two black holes are located on the x-axis at positions $(\pm 3, 0, 0)$. The linear momenta for these two spinless black holes are $(0, \pm 0.2, 0)$. With these parameters, the two black holes inspiral around each other along a quasicircular orbit.

Firstly we check the effects of the boundary condition and computational size on the numerical solution. We use the cubic computational domain $-r_{BD} < x, y, z < r_{BD}$ and the Dirichlet boundary condition (10). Like in the toy model problem, we set up the initial mesh grid with the

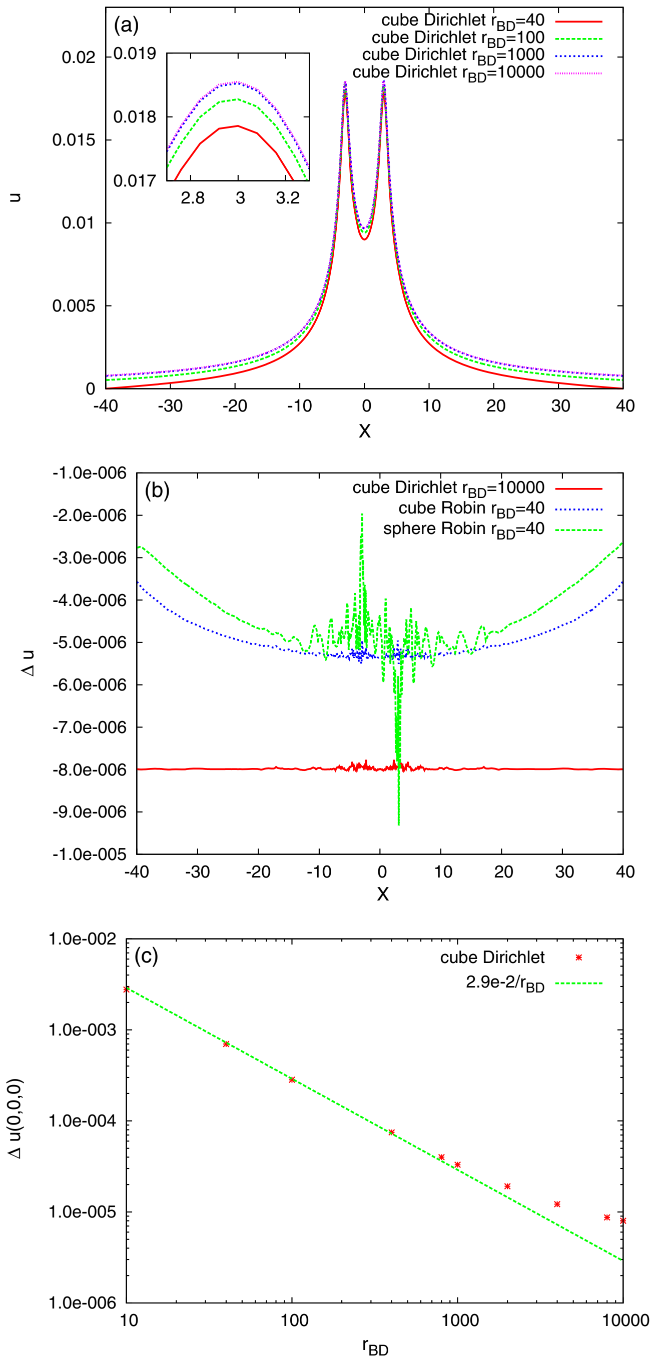

FIG. 6 (color online). (a) Numerical solution comparison among different computational sizes r_{BD} for a binary punctured black hole. The cubic domain and the Dirichlet boundary condition (10) are used in this subplot. (b) Difference Δu between the numerical solution and the reference solution. (c) The relationship between $\Delta u(0, 0, 0)$ and r_{BD}. The cubic domain is used for this subplot. Here we use the numerical solution of "cube Robin $r_{BD} = 10^4$" as the reference solution. Variant computational domains and variant boundary conditions are tested. "Cube" and "sphere" mean the cubic domain and spherical domain, respectively. "Dirichlet" and "Robin" represent the Dirichlet boundary condition (10) and the Robin boundary condition (11), respectively.

simplest tetrahedral element decomposition with 8 vertices and 5 elements. Then we uniformly refine the grid 6 times. After that we refine the boundary elements 3 times. Afterwards, we use the E_i in (28) as the refinement indicator and let PHG do the adaptive refinement along the solving process. We compare the solution with different r_{BD}'s in Fig. 6(a). In this plot, we have interpolated the numerical solutions onto the x-axis like in Fig. 2.

In the framework of the finite element method, it is easy to treat any shape of boundary. Here we have compared the numerical result with the spherical computational domain to the one with the cubic computational domain. To implement the spherical domain $\sqrt{x^2+y^2+z^2} < r_{BD}$, we use NetGen [31] to generate an initial grid for a given sphere first. Then we transform the output grid file to Albert format [32] and add a "curved boundaries" description to the Albert file. The curved boundaries are an extension of Albert format to support grid refinement near the boundary, which lets the refined grids approach the wanted boundary better. Starting from this mesh grid, we use the E_i in (28) as the refinement indicator and let PHG do the adaptive refinement along the solving process.

The previous work [14] has shown that the Robin boundary condition (11) works very well in the current problem. And since the exact solution is not available, we use the numerical solution with the boundary condition (11) and with a large computational domain $-10^4 < x, y, z < 10^4$ as our reference solution. In Fig. 6(b) we investigate the difference Δu between the numerical solutions and the reference solution. The difference is calculated in the following way. We firstly interpolate both the reference solution and the numerical solution to some given points on the x-axis. Then we calculate the difference between them. The numerical solution of "sphere Robin $r_{BD} = 40$" behaves differently around the two punctures. This is due to the mesh grid structure. We will comment on this point more in the following. In Fig. 6(c) we investigate the relationship between Δu and r_{BD} for the Dirichlet boundary condition (10). Here the cubic computational domain is used. Roughly when $r_{BD} < 2000$, a clear power law in $\frac{1}{r_{BD}}$ can be seen. The power-law exponent 1 is consistent with the expectation. Since the Dirichlet boundary condition (10) only considers the zeroth order asymptotic behavior of the metric, the next order which corresponds to the power-law exponent 1 contributes to the numerical error. So the power-law exponent 1 is expected. This means the boundary effects of the monopole dominate in this range. When $r_{BD} > 2000$, the higher multipole effects and the numerical truncation error come in, so the power law stops. These higher multipole effects and the numerical truncation error together contribute about 1×10^{-5}.

We compare the grid structure for cube Robin $r_{BD} = 40$ and sphere Robin $r_{BD} = 40$ in Fig. 7. Firstly we find that the adaptive mesh refinement can find out the puncture positions automatically. In addition, from this figure we can see the grid structure for the spherical computational domain is less efficient. Some fine grids appear at the wrong places (lower part region). This phenomenon occurs for two reasons. One is because the initial grid is not efficient enough. This is due to less tuning usage of NetGen. The other reason is that the curved-boundaries object has not been adjusted to work well. PHG developers are still working on this point. These two points are out of the scope of the current work. In the following tests, we will only be concerned with the cubic computational domain.

We check the convergence behavior of the numerical solution in Fig. 8. We have tested three different order polynomial basis functions. "P1" ("P1h"), "P3," and "P6" correspond to the first order, third order, and sixth order, respectively. Except for the tests done here, we use the third order polynomial by default. In panel (a), we plot the infinity norm of E_i against the adaptive mesh refinement step n. From the definition (28), we can see E_i is nothing but a weak form of the Hamiltonian constraint violation. So this plot shows the convergence of the Hamiltonian constraint with respect to the adaptive mesh refinement process. As we explained after Eq. (28), E_i is a function of elements. We search through all the elements to find the maximal $|E_i|$ which corresponds to the infinity norm. Given some numerical error tolerance ϵ, when the infinity norm of E_i becomes less than ϵ, we stop the adaptive mesh refinement process and take the solution as the desired solution. We have set $\epsilon = 1 \times 10^{-10}$ for P1, P3, and P6 runs and set $\epsilon = 4 \times 10^{-13}$ for the P1h run. In panel (b), we plot the L_2 norm of Δu with respect to the adaptive mesh refinement steps. Δu is again the difference between the reference solution and the numerical solution. The reference solution corresponds to cube Robin $r_{BD} = 10^4$ as before. We interpolate both the reference solution and the numerical solution to the x-axis. Then we calculate the difference. Here we only consider the $-40 < x < 40$ part and the L_2 norm in this part is calculated. These two plots are similar to Fig. 3(b). Both plots show a qualitatively similar behavior of convergence. The lines in both plots are clearly divided into two segments. Through checking the numerical solutions and the grid structure, we can find that the grids have not resolved the puncture regions during the first segment. During the second segment, the refinement operations mainly treat the region around the puncture points. In this sense, the first segment does not correspond to the real convergence. We estimate the convergence order based on the second segment and get the convergence order $p \approx 2$ for all of the four runs. In Fig. 8(c) we check the convergence behavior of E_i. The plot shows the interpolation result of E_i to the x-axis. We can see clear steps in this plot. This is because E_i is a function of elements. So the points locating in the same element admit the same value. Here we only show the result for P3. The results for other runs are similar.

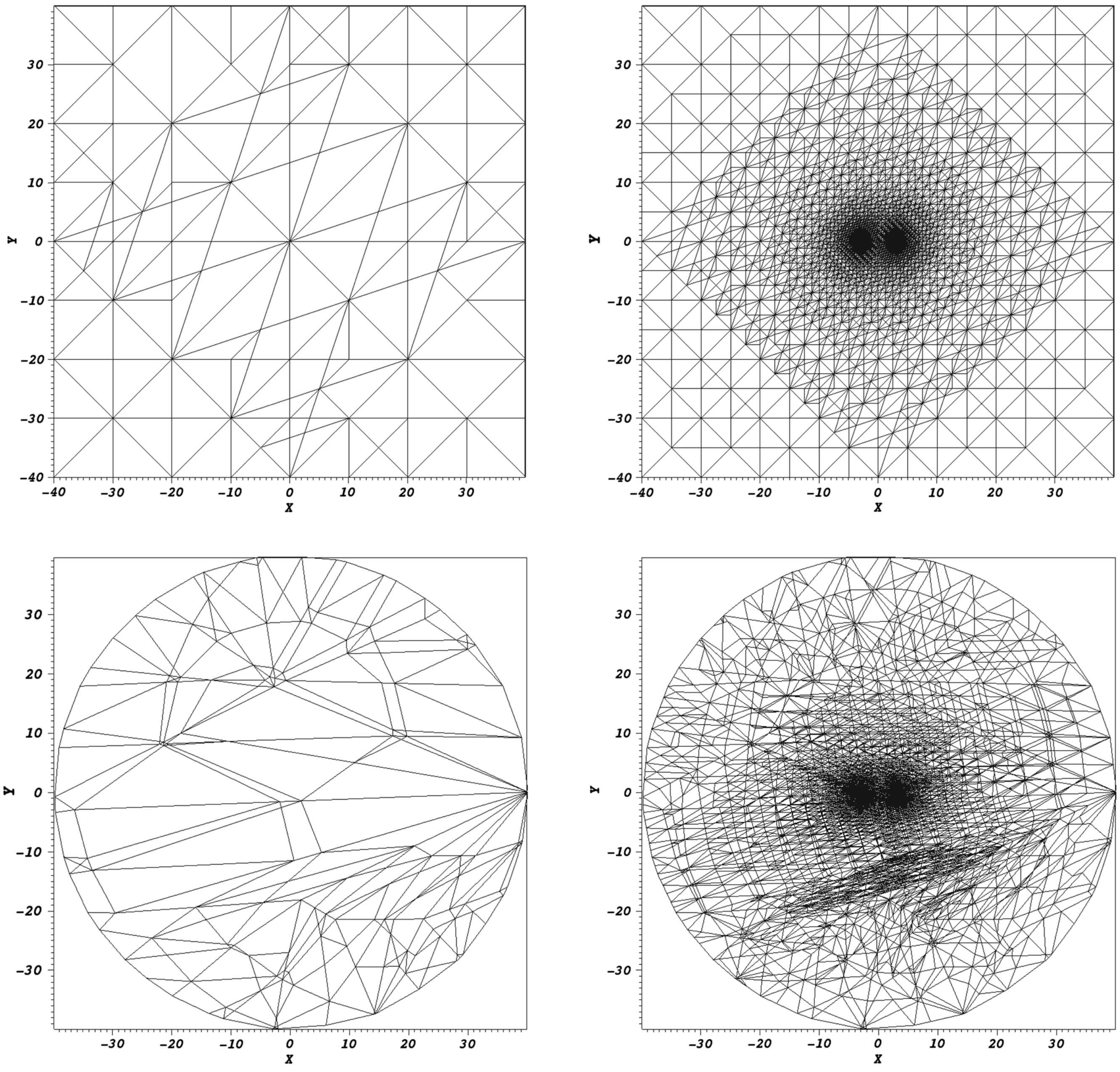

FIG. 7. Grid structure comparison for the cubic computational domain (the top row) and spherical one (the bottom row). Here we show the grid structure in the $z = 0$ plane. The left column corresponds to the grid before adaptive mesh refinement takes place. The right column corresponds to the grid when the desired solution is gotten.

In Fig. 9 we investigate more about the behavior for different order polynomial basis functions. In panel (a), we plot the function E_i along the x-axis. From this plot we can see that the steps are wider when the polynomial order is higher. This is because higher order polynomial basis functions make the solving process reach the given error tolerance with coarser grids. This result implies the efficiency of introducing a higher polynomial basis function in the smooth region, which is consistent with the expectation we gave in the Introduction. In panel (b), we present the difference between the reference solution and the numerical solutions. We again use the numerical solution of cube Robin $r_{BD} = 10^4$ as the reference solution. We can see that the difference of P1 is larger than that of P1h. This implies that the numerical solution along the solving process does converge to the reference solution. We also note that the difference of P1h is larger than that of P3 and P6, although P1h uses 2 orders smaller tolerance. This cautions us that smaller E_i alone does not mean a more accurate numerical solution.

Since our goal is to get the initial data which satisfy the Hamiltonian constraint equation, we would like to check

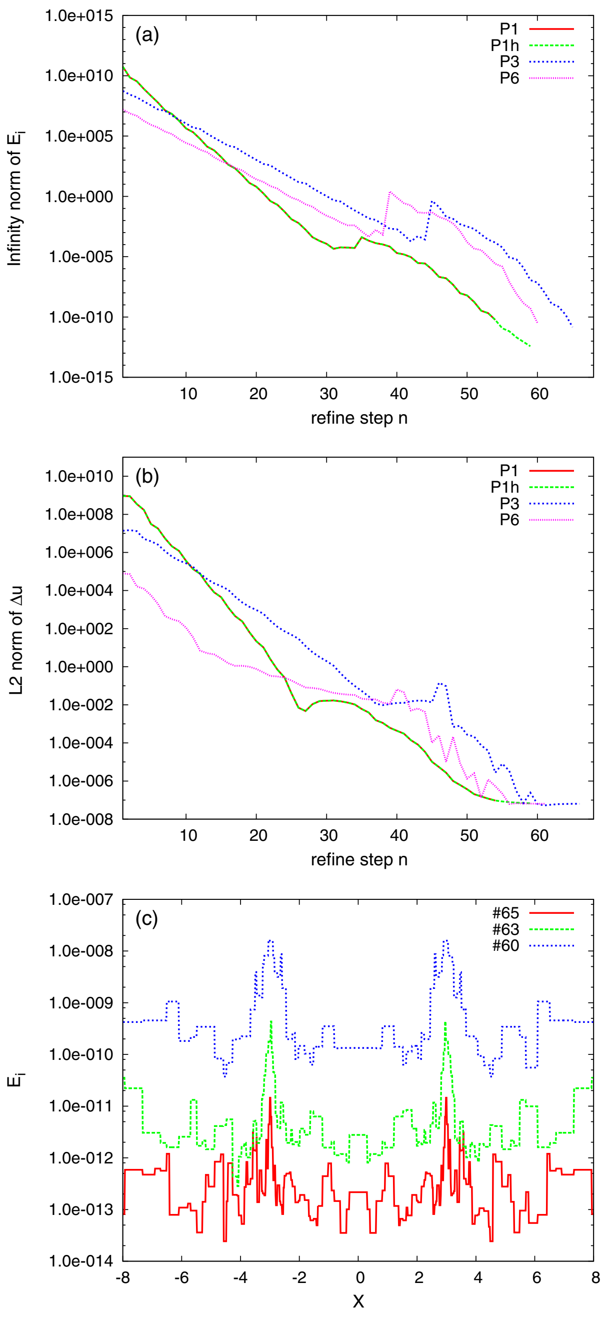

FIG. 8 (color online). Convergence behavior for the punctured binary black hole along the solving process. This figure corresponds to the "cube Dirichlet $r_{BD} = 10000$." For other cases the convergence behavior is similar. Here the lines marked with P1, P3, and P6 correspond to the first order, third order, and sixth order polynomial basis function runs, respectively. We use numerical error tolerance 1×10^{-10} for P1, P3, and P6. For P1h, we use the first order polynomial as P1 but set much more stringent tolerance 4×10^{-13}. (a) The infinity norm of E_i with respect to the mesh refinement step n. (b) The L_2 norm of Δu on the x-axis with respect to n. Since each refinement makes the effective grid double the resolution, the x-axis can also be looked at as the logarithm of the resolution of the mesh grid. (c) The profile of E_i at the last mesh refinement steps for P3. "#n" means the nth step of mesh refinement.

the effect of different polynomial orders on the Hamiltonian constraint violation. In order to make the comparison fair, we import the resulting different initial data into AMSS-NCKU code. On the side of AMSS-NCKU code, the same grid setting and the same calculation procedure are taken. We set up a uniform mesh grid based on Cartesian coordinates within AMSS-NCKU code. Then we copy the numerical solution u into this mesh grid and construct the Baumgarte-Shapiro-Shibata-Nakamura variables based on the puncture initial data instruction. After that we use a fourth order finite difference to calculate the partial derivatives and follow Eqs. (12) of [2] to compute the Hamiltonian constraint violation. This procedure is the same as the one we used to check constraint violation during evolution within AMSS-NCKU code [2]. Finally we interpolate the resulting constraint violation onto the x-axis and plot in Figs. 9(c)–(f). Subplots (c) to (f) correspond to different uniform grids with different resolutions within AMSS-NCKU code. Where (c) uses resolution $\frac{1}{8}$, (d) uses $\frac{1}{16}$, (e) uses $\frac{1}{32}$, and (f) uses $\frac{1}{64}$. The amplitude of the shown Hamiltonian constraint violation depends on two factors. One is the resolution used in AMSS-NCKU code. So we can see when the resolution becomes higher, the Hamiltonian constraint around the puncture points becomes smaller. [Apparently the amplitude in (d) is higher than that in (c). This is because the sampling point in (d) is nearer to the puncture point.] The second factor comes from the numerical error of the solution u. The effect comes from this factor that distinguishes among different numerical solutions. We can see that the higher order polynomial gives smoother Hamiltonian constraint violation, and moreover it results in a smaller constraint amplitude even though the higher order uses a coarser mesh grid. At the same time we can note that P1h results in a slightly smaller constraint violation than P1. The higher order polynomial corresponds to the p-refinement. The finer mesh grid corresponds to the h-refinement. The results in Figs. 9(c)–(f) imply that p-refinement is more efficient than h-refinement in the current context.

The solution function u is expected to admit only a smooth profile except at the puncture points, where it is C^4. Recalling this particular singular property of puncture points, we borrow the idea from the TwoPuncture method to put these puncture points at some element vertices. Then in all elements the unknown function is smooth. We test this idea with the cubic computational domain $r_{BD} = 40$ and Robin boundary condition (11). In order to put the two punctures at some element vertex, we use 12 boxes, as shown in Fig. 10, to decompose the computational domain. Then we let NetGen generate the initial grid. After that we use an adaptive mesh refinement procedure to solve the constraint equation. We need many fewer elements to get the same numerical error tolerance in this way than that gotten without paying special attention to the puncture point (for clarity we call this the "usual" method).

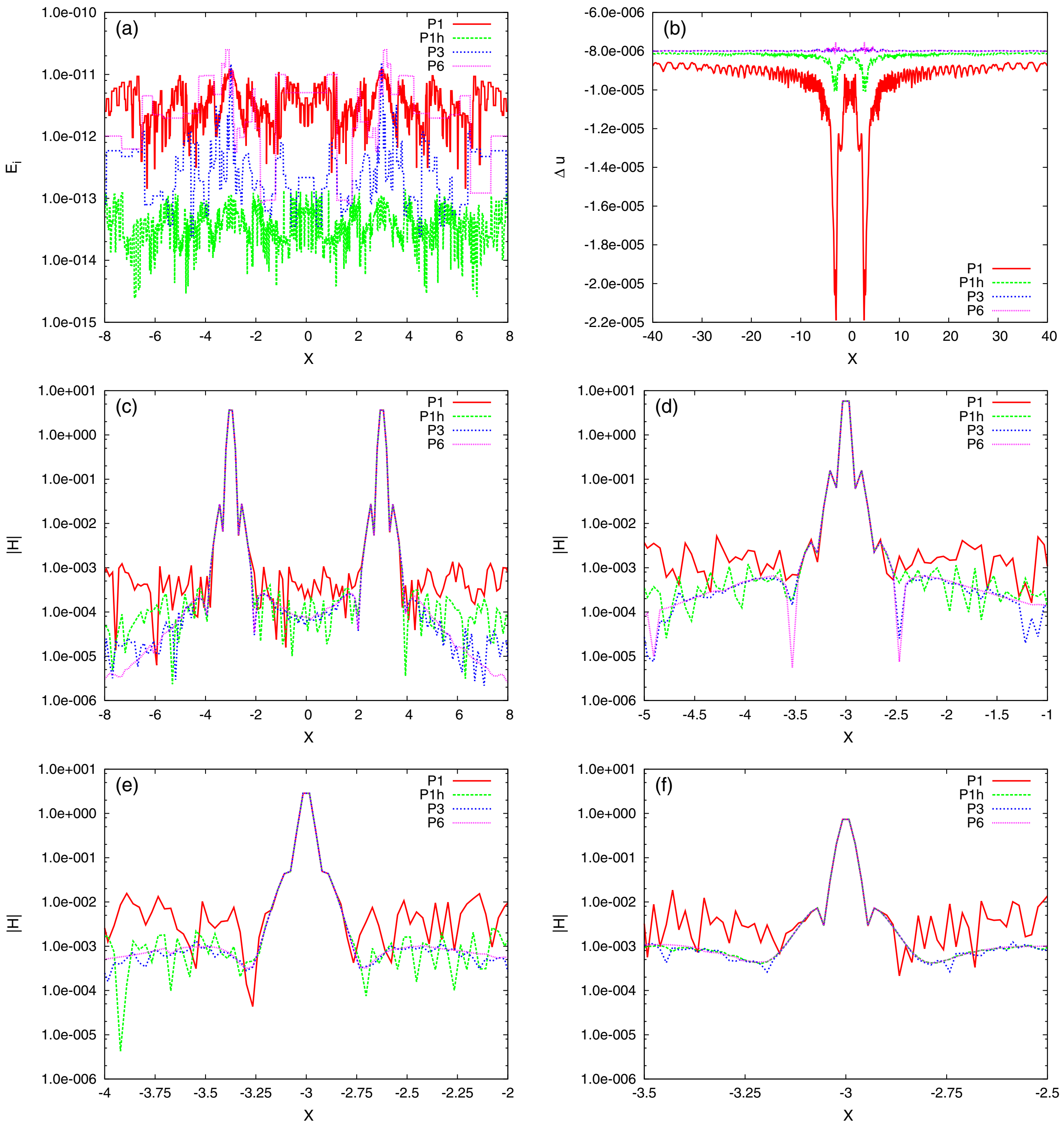

FIG. 9 (color online). Comparison of the numerical solutions with different order polynomial basis functions. The meaning of the legend is the same as in Fig. 8. (a) E_i along the x-axis. (b) Difference between the reference solution and the numerical solutions. Again the numerical solution of cube Robin $r_{BD} = 10^4$ is used as the reference solution. (c)–(f): Comparison of Hamiltonian constraint violation. The constraint violation is calculated through AMSS-NCKU code based on a uniform mesh grid. (c) uses resolution $\frac{1}{8}$, (d) uses $\frac{1}{16}$, (e) uses $\frac{1}{32}$, and (f) uses $\frac{1}{64}$.

We compare the results gotten by "puncture at vertex" to the one gotten by the usual method in Fig. 11. In panel a we show the difference of the numerical solution between these two methods. We can see the difference is 1 order smaller than the numerical error with respect to the reference solution cube Robin $r_{BD} = 10^4$ (compare Fig. 6). In this sense the puncture at vertex method does not improve the solution's accuracy. In panel (b), we compare the E_i function along the x-axis. For the usual method, two peaks at the puncture points are clear, while for the puncture at

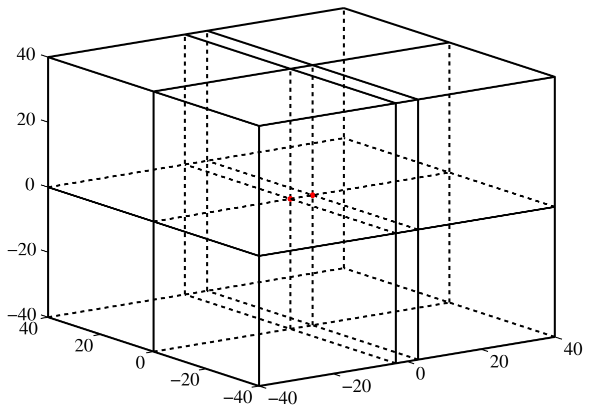

FIG. 10 (color online). Subdomain settings for putting the punctures at vertex. The subdomains include 12 boxes. The two punctures are marked with two red dots in the sketch.

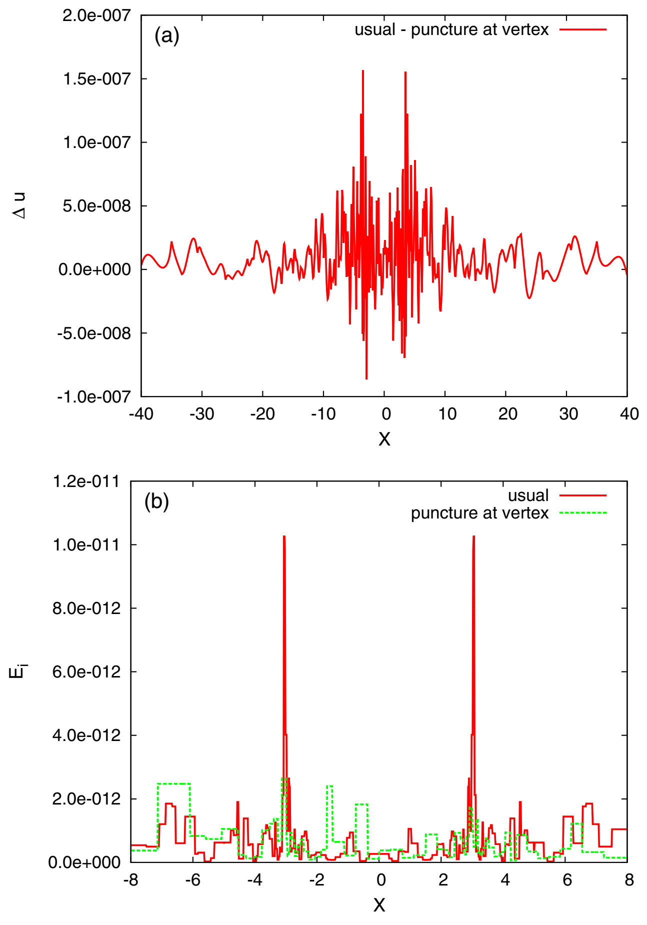

FIG. 11 (color online). Test for the method putting the puncture points at some element vertex (marked with "puncture at vertex"). The line marked with "usual" corresponds to the usual method which does not consider the puncture points especially. In this figure, the cubic domain with $r_{BD} = 40$ and the Robin boundary condition are used. (a) Numerical solution difference between these two methods. (b) Comparison of the E_i function along the x-axis.

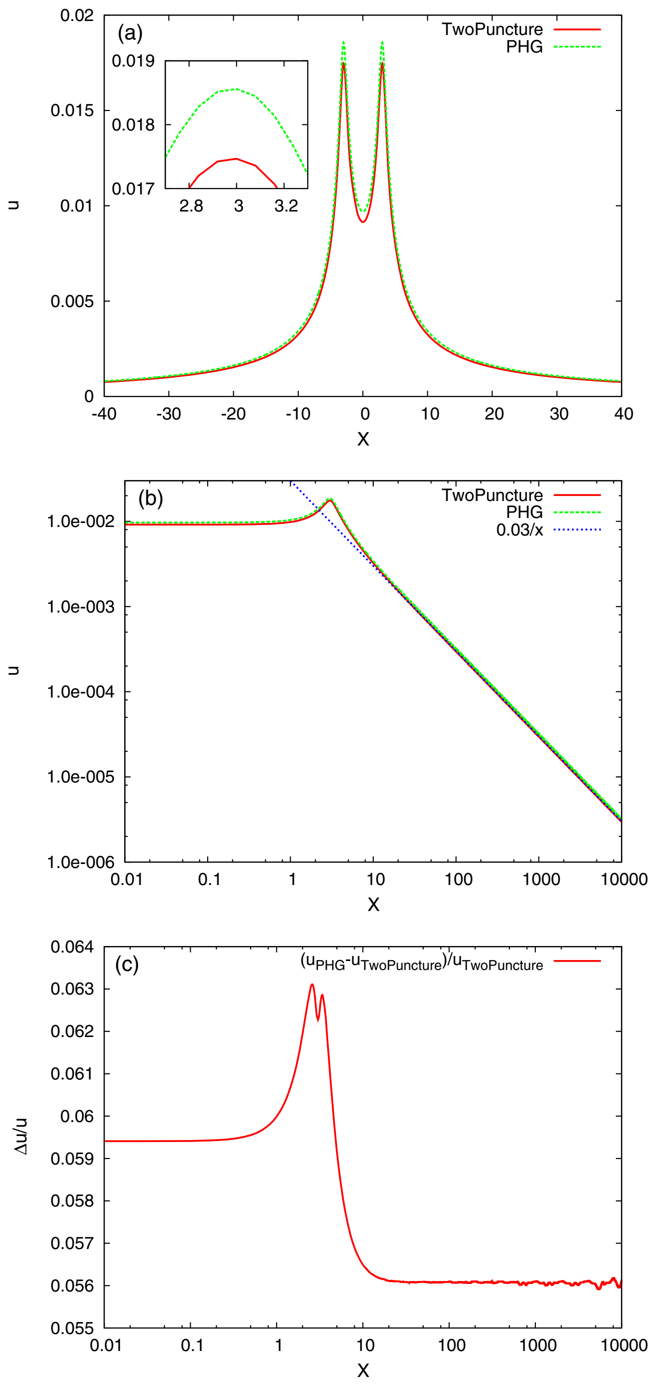

FIG. 12 (color online). Numerical solution comparison between the spectral method and finite element method for binary punctured black holes. The line marked with "PHG" is the solution gotten through the finite element method, which corresponds to the one with the cube computational domain $r_{BD} = 10^4$ and the Robin boundary condition (11). The line marked with "TwoPuncture" is the solution gotten by the spectral method, which corresponds to the one shown in Fig. 1. (a) Comparison of the numerical solution near the puncture points. (b) Comparison of the numerical solution in a large scale. (c) Relative difference between the two numerical solutions.

vertex method, the puncture points do not show any special performance. The much larger step profile for the puncture at vertex method indicates that the desired numerical error tolerance is gotten with much larger grid elements than the one with the usual method. In this sense the puncture at vertex method is more efficient.

In the following we compare the numerical solution gotten by the finite element method to the one gotten by the spectral method. The spectral method code is an implementation of the TwoPuncture scheme proposed in [19]. The result is shown in Fig. 12. The solution gotten by the finite element method (marked with "PHG") corresponds to cube Robin $r_{BD} = 10^4$, which has been used as the reference solution above. The solution gotten by the spectral method (marked with "TwoPuncture") corresponds to the one shown in Fig. 1. In panel (a), we show the two solutions near the puncture points. The value for TwoPuncture at the puncture point is about 0.0175. For the finite element method, the value is roughly 0.0185. This difference is much larger than that introduced by a different grid setting, different boundary condition, different numerical error tolerance setting, and many other factors of the finite element method, as analyzed before. In panel (b), we show the two solutions TwoPuncture and PHG in a much larger spatial scale. Together with the plot we also show the power-law relation $0.03/x$ for comparison. When x goes to infinity, both solutions behave as power-law $\frac{1}{x}$ decay. This supports our approximated Robin boundary condition (11) at finite distance. Based on this analysis, we suspect that the difference between TwoPuncture and PHG comes from the numerical truncation error of finite element methods. In panel (c), we show the relative difference between these two solutions. Interestingly, the relative difference almost uniformly distributes in a quite large scale. This result implies that the adaptive finite element method is quite effective. The fine grids are only added in needed regions and roughly no computational effort is wasted.

In Fig. 13 we compare the Hamiltonian constraint violation resulting from the solutions TwoPuncture and PHG. We again use the AMSS-NCKU code to calculate the constraint. The detailed procedure is the same as described for Fig. 9. Here we have used 12 unigrids to cover different regions and calculate the constraints individually. After that we combine these results together and show them in Fig. 13. The vertical lines shown in the plot indicate the interface of these regions. Higher resolution is used for the regin nearer to the puncture point. For TwoPuncture, the line segments within $x - 3 < 0.1$ and $x - 3 > 10$ show clear level-off behavior. For PHG the $x - 3 < 0.1$ part shows level-off behavior. This is because the constraint violation is dominated by the numerical error of AMSS-NCKU code in these regions. And this level-off behavior corresponds to the different unigrids with different resolutions used in AMSS-NCKU implementation. In general, PHG results in larger constraint violation than

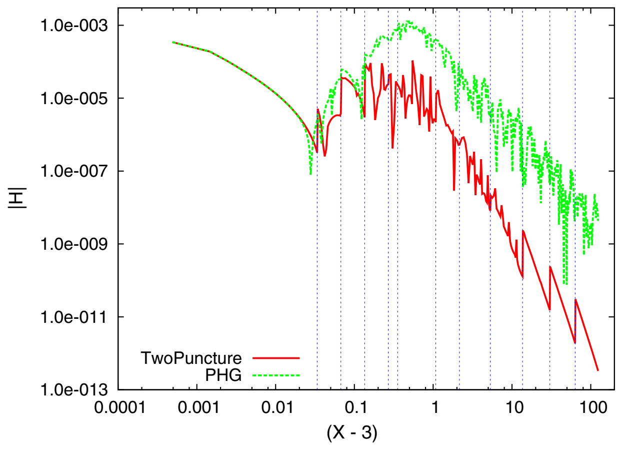

FIG. 13 (color online). Comparison of Hamiltonian constraint violation between the spectral method and finite element method for binary punctured black holes. The numerical solutions correspond to the ones shown in Fig. 12. The Hamiltonian constraint violation is calculated through AMSS-NCKU code. Within AMSS-NCKU code we here have combined 12 unigrid calculation results. Different unigrids use different resolutions. In the plot the vertical lines mark out the interface of these unigrids. From left to right the resolutions for the unigrids are $\frac{1}{1024}$, $\frac{1}{512}$, $\frac{1}{256}$, $\frac{1}{128}$, $\frac{1}{64}$, $\frac{1}{32}$, $\frac{1}{16}$, $\frac{1}{8}$, $\frac{1}{4}$, $\frac{1}{2}$, 1, and 2, respectively.

TwoPuncture. This is consistent with our expectation that the spectral method admits higher accuracy. In conclusion, the adaptive finite element method results in less accurate solutions than the spectral method. But the adaptive finite element method is quite effective in the sense that the resulting numerical error distributes almost uniformly. No redundant computational cost is involved.

C. Initial data for three punctured black holes

The three-body problem is quite interesting even in Newton gravity. For planar motion within Newton gravity, it is difficult to investigate the dynamics [33,34]. Three well-known configurations, including Lagrange's triangle, Henon's crisscross, and Moore's figure eight [35,36], have been extensively studied. When the motion of three bodies touches a general relativity region, one should consider post-Newtonian (PN) correction to the dynamics. Currently, the 2.5PN correction for three-body interaction is available [37–40]. When the interaction among the three bodies becomes stronger, the full Einstein equations must be used. Now only numerical relativity is possible to treat this kind of problem. The authors in [41,42] pioneered the three-black-hole simulations. Due to the lack of a proper initial data solver for three black hole problems, the authors used approximate analytic initial data [43]. They investigated several configurations of three black hole problems. All configurations were motivated by gravitational wave detection, and special attention was paid to inspiral and merger. The most interesting one (denoted as 3BH102) results in two successive inspiral-merger events. The tracks

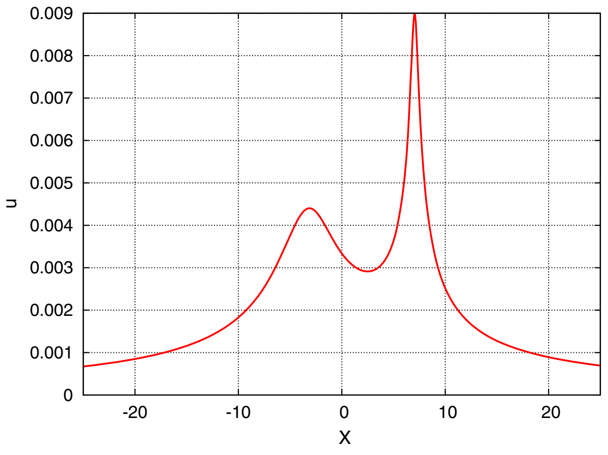

FIG. 14 (color online). The numerical solution for the 3BH102 configuration. Here we use the cubic computational domain with boundary located at $r_{BD} = 40$, and the Robin boundary condition (11) is used.

of the three black holes look like three greek letters γ, σ, and τ. Later the authors in [14] developed the finite difference solver OLLIPTIC for three black hole problems. They found that the simulation result is quite sensitive to the initial data. The final fate of the three black holes that started from numerical initial data is different from the one started from approximate analytic initial data. Regarding the tracks of the three black holes, the γ is not clear any more. σ and τ are deformed. Regarding the two inspiral-merger events, only one is clear. The other one almost changes to head-on collision. So only one burst is presented in the gravitational wave form.

Unlike the TwoPuncture code that is limited to two black hole problems, the finite element method is much more flexible. In this subsection, we apply it to the three black hole problem. We reinvestigate the configuration 3BH102 discussed in [14,41,42]. The three spinless black holes with puncture mass parameters 0.317578, 0.317578, and 0.318585 sit at $(-3.52238, 2.58509, 0)$, $(-3.52462, -2.58509, 0)$, and (7.04476,0,0), respectively. Their linear momenta are $(0.0782693, -0.0433529, 0)$, $(-0.0782693, -0.0433529, 0)$, and (0,0.0867057,0).

We have tested different computational domains and different boundary conditions. All of them result in roughly the same solution. In the following we only present the result with cube Robin $r_{BD} = 40$. We show the numerical solution in Fig. 14. The numerical solution shown in Fig. 14 is consistent with the one shown in Fig. 8 of [14]. In Fig. 15 we plot the resulting numerical error indicator function E_i and the Hamiltonian constraint violation. Again the Hamiltonian constraint violation is calculated through porting the initial data to AMSS-NCKU code.

In Fig. 16 we present the convergence behavior. After roughly 20 mesh refinement steps, the solving process goes into a convergent region. The estimated convergence order is about 1.9.

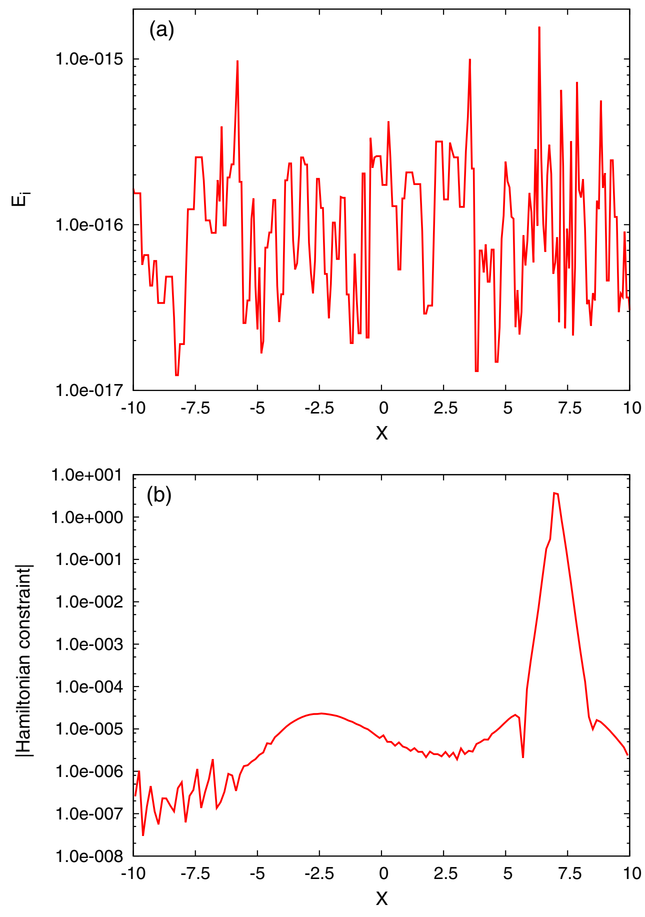

FIG. 15 (color online). (a) E_i function along the x-axis. (b) The Hamiltonian constraint violation. Here the resolution used in AMSS-NCKU code is 0.15625. This figure corresponds to the numerical solution u in Fig. 14.

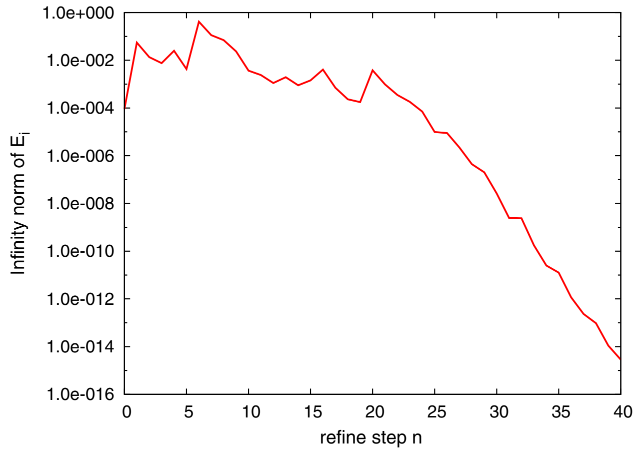

FIG. 16 (color online). Convergence behavior for the 3BH102 configuration along the solving process. The plot convention is the same as Fig. 8. The x-axis is the adaptive refinement step n. This plot corresponds to the numerical solution u in Fig. 14.

V. SUMMARY AND DISCUSSION

As one of the three categories' methods to solve partial differential equations, the finite element method is still missing in numerical relativity. In an ongoing series of works, we investigate the possible advantage of the adaptive finite element method in the application to numerical relativity. We are interested especially in the binary compact object simulation. For this goal, we combine the existing numerical relativity code AMSS-NCKU and the library PHG to develop an adaptive finite element code, Einstein PHG (iPHG), for the Einstein equations. In this paper we have studied the constraint part of the Einstein equations. Based on the character of PHG software, we have designed a numerical scheme for the Einstein constraint equations. Specifically we apply our scheme and the developed code to the puncture type initial data problem for multiple black holes. Starting from a simple grid setting, our code can catch the punctured regions and refine them automatically. Along with the adaptive refinement process, the converged numerical solution is efficiently achieved.

Before we treat the initial data problem for the Einstein equations, we have also tested our code with a toy model whose exact solution is known. Our code works very well in the toy model problem to catch the discontinuous region and get an accurate numerical solution. The numerical error is less than 10^{-5}. For the multiple black hole problems we have compared the results to the ones gotten in previous works. We find our numerical results are consistent with previous results in general. For the two punctured black hole problems, we have compared our solution to the one gotten through the spectral method. As expected, the solution gotten by the finite element method is less accurate than that gotten by the spectral method. But the numerical error resulting from the finite element method distributes almost uniformly. This implies that the adaptive mesh refinement operation in our implementation is quite efficient. Only the region needs refinement is refined. Roughly no computational cost is wasted.

It is possible to use a high order polynomial function basis to mimic the spectral method within each element. We have tested high order polynomials till the eighth. The advantages of the application of these high order polynomial functions are the efficiency and the accuracy, as expected. But for current punctured black hole problems, a high convergence order cannot be gotten by using a high order polynomial. This is due to the nonsmoothness of the source term of the constraint equations near puncture points. In principle, some ingenious error estimator [44] can be constructed and used as the refinement indicator; then high order convergence along the refinement process may be achieved. This is out of the scope of the current work, so we are not concerned about this point here.

The results in this work have implied some possible advantages of applying the finite element method to numerical relativity. Regarding the difficulty of coding the finite element method for large scale scientific computation, the PHG library may be a good tool. Very recently T. Cui and J. Tao have ported the PHG library into Cactus [45], which is the most popular tool in the numerical relativity community. So the numerical schemes proposed in the current work can be easily implemented through Cactus in principle.

ACKNOWLEDGMENTS

We wish to thank the anonymous referee for numerous helpful suggestions. We are very grateful to Tao Cui, Lin-Bo Zhang, and other friends in the PHG developing group for helping with PHG usage. We are also thankful to Kai-Bo Hu, Chunxiong Zheng, and Pei-Nan Zhong for discussion about the finite element method. This work was supported by the NSFC (Grant No. 11375260).

[1] B. Aylott, J. G. Baker, W. D. Boggs, M. Boyle, P. R. Brady, D. A. Brown, B. Brügmann, L. T. Buchman, A. Buonanno, L. Cadonati *et al.*, Classical Quantum Gravity **26**, 165008 (2009).
[2] Z. Cao, H.-J. Yo, and J.-P. Yu, Phys. Rev. D **78**, 124011 (2008).
[3] T. Yamamoto, M. Shibata, and K. Taniguchi, Phys. Rev. D **78**, 064054 (2008).
[4] M. A. Scheel, M. Boyle, T. Chu, L. E. Kidder, K. D. Matthews, and H. P. Pfeiffer, Phys. Rev. D **79**, 024003 (2009).
[5] C. F. Sopuerta and P. Laguna, Phys. Rev. D **73**, 044028 (2006).
[6] C. F. Sopuerta, P. Sun, P. Laguna, and J. Xu, Classical Quantum Gravity **23**, 251 (2006).
[7] G. Zumbusch, Classical Quantum Gravity **26**, 175011 (2009).
[8] Parallel Hierarchical Grid library, 2014, http://lsec.cc.ac.cn/phg/index_en.htm.
[9] L.-B. Zhang, Numer. Math. **2**, 65 (2009).
[10] R. M. Wald, *General Relativity* (University of Chicago Press, Chicago, 1984).
[11] C. Liang, *Introductory Differential Geometry and General Relativity I, II* (Beijing Normal University Press, Beijing, 2000).
[12] H. P. Pfeiffer, arXiv:gr-qc/0510016.
[13] S. Brandt and B. Brügmann, Phys. Rev. Lett. **78**, 3606 (1997).

[14] P. Galaviz, B. Brügmann, and Z. Cao, Phys. Rev. D **82**, 024005 (2010).
[15] J. D. Brown and L. L. Lowe, J. Comput. Phys. **209**, 582 (2005).
[16] E. Gourgoulhon, P. Grandclement, K. Taniguchi, J.-A. Marck, and S. Bonazzola, Phys. Rev. D **63**, 064029 (2001).
[17] http://www.lorene.obspm.fr/, 2008.
[18] H. P. Pfeiffer, L. E. Kidder, M. A. Scheel, and S. A. Teukolsky, Comput. Phys. Commun. **152**, 253 (2003).
[19] M. Ansorg, B. Brügmann, and W. Tichy, Phys. Rev. D **70**, 064011 (2004).
[20] G. B. Cook, M. W. Choptuik, M. R. Dubal, S. Klasky, R. A. Matzner, and S. R. Oliveira, Phys. Rev. D **47**, 1471 (1993).
[21] G. B. Cook, Ph.D. thesis, The University of North Carolina at Chapel Hill, 1990.
[22] G. B. Cook, Phys. Rev. D **44**, 2983 (1991).
[23] J. W. York, Jr. and T. Piran, in *Spacetime and Geometry* (University of Texas, Austin, 1982), Vol. 1, p. 147.
[24] Visualization ToolKit, http://www.kitware.com/opensource/vtk.html, 2014.
[25] OpenDX, http://www.opendx.org/index2.php, 2014.
[26] Z. Cao and C. Liu, Int. J. Mod. Phys. D **20**, 43 (2011).
[27] Z. Cao, Int. J. Mod. Phys.D **21**, 1250061 (2012).
[28] H.-J. Yo, C.-Y. Lin, and Z. Cao, Phys. Rev. D **86**, 064027 (2012).
[29] Z. Cao, P. Galaviz, and L.-F. Li, Phys. Rev. D **87**, 104029 (2013).
[30] Z. Cao and D. Hilditch, Phys. Rev. D **85**, 124032 (2012).
[31] NetGen, 2014, http://www.hpfem.jku.at/netgen/.
[32] A. Schmidt, K. Siebert, D. Köster, O. Kriessl, and C. Heine, http://www.alberta-fem.de, 2014.
[33] M. Šuvakov and V. Dmitrašinović, Phys. Rev. Lett. **110**, 114301 (2013).
[34] V. Dmitrašinović, M. Šuvakov, and A. Hudomal, Phys. Rev. Lett. **113**, 101102 (2014).
[35] T. Imai, T. Chiba, and H. Asada, Phys. Rev. Lett. **98**, 201102 (2007).
[36] Y. Torigoe, K. Hattori, and H. Asada, Phys. Rev. Lett. **102**, 251101 (2009).
[37] G. Schäfer, Phys. Lett. A **123**, 336 (1987).
[38] C. O. Lousto and H. Nakano, Classical Quantum Gravity **25**, 195019 (2008).
[39] P. Galaviz and B. Brügmann, Phys. Rev. D **83**, 084013 (2011).
[40] P. Galaviz, Phys. Rev. D **84**, 104038 (2011).
[41] C. O. Lousto and Y. Zlochower, Phys. Rev. D **77**, 024034 (2008).
[42] M. Campanelli, C. O. Lousto, and Y. Zlochower, Phys. Rev. D **77**, 101501 (2008).
[43] P. Laguna, Phys. Rev. D **69**, 104020 (2004).
[44] J. Hu, L. Jiang, and Z. Shi, J. Comput. Appl. Math. **265**, 173 (2014).
[45] Cactus code, http://cactuscode.org/, 2014.