Generating Equidistributed Meshes in 2D via Domain Decomposition

Ronald D. Haynes[1] and Alexander J. M. Howse[1]

1 Introduction

There are many occasions when the use of a uniform spatial grid would be prohibitively expensive for the numerical solution of partial differential equations (PDEs). In such situations, a popular strategy is to generate an adaptive mesh by either varying the number of mesh points, the order of the numerical method, or the location of mesh points throughout the domain, in order to best resolve the solution. It is the latter of these options, known as *moving mesh methods*, which is our focus. In this case the physical PDE of interest is coupled with equations which adjust the position of mesh points to best "equidistribute" a particular measure of numerical error. This coupled system of equations is solved to generate the solution and the corresponding mesh simultaneously, see [7] for a recent overview.

A simple method for adaptive grid generation in two spatial dimensions is outlined in [8] by Huang and Sloan, in which a finite difference two dimensional adaptive mesh method is developed by applying a variation of de Boor's equidistribution principle (EP) [1, 2]. The equidistribution principle states that an appropriately chosen mesh should equally distribute some measure of the solution variation or computational error over the entire domain. Mackenzie [9] extends upon the work of [8] by presenting a finite volume discretization of the mesh equations, as well as an efficient iterative approach for solving these equations, referred to as "an alternating line Gauss-Seidel relaxation approach".

In this paper, we propose a parallel domain decomposition (DD) solution of the 2D adaptive method of [8]. In Section 2 we review the derivation of the mesh PDEs of [8] and discuss possible boundary conditions. In Sections 3 and 4 we present classical and optimized Schwarz methods for the generation of 2D equidistributed meshes, and in Section 5 we describe the numerical implementation of this approach and provide numerical results.

2 2D Mesh Generation

To begin, we review the derivation of the equations which govern mesh equidistribution in two spatial dimensions from [8], defining a mesh in the physical variables (x,y) which *best* resolves a given function $u(x,y)$. Let $\mathbf{x} = [x,y]^T$ be the spatial coor-

[1]Memorial University of Newfoundland, Department of Mathematics & Statistics, St. John's, NL, Canada A1C 5S7 e-mail: {rhaynes}{z37ajmh}@mun.ca

dinates of a mesh in a 2D physical domain, Ω_p. We introduce the coordinate transformation $\mathbf{x} = \mathbf{x}(\boldsymbol{\xi})$, where $\boldsymbol{\xi} = [\xi, \eta]^T$ denotes the spatial coordinates on the computational domain, $\Omega_c = [0,1] \times [0,1]$. Here we determine a mesh which equidistributes the arc-length of $u(x,y)$ over Ω_p. The scaled arc-length variation of u along the arc element from $\mathbf{x}$ to $\mathbf{x} + d\mathbf{x}$ can be expressed as

$$ds = [a^2(du)^2 + d\mathbf{x}^T d\mathbf{x}]^{1/2} = [d\mathbf{x}^T \boldsymbol{M} d\mathbf{x}]^{1/2}, \tag{1}$$

where $\boldsymbol{M} = a^2 \nabla u \cdot \nabla u^T + \boldsymbol{I}$ and $a \in [0,1]$ is a user specified relaxation parameter. The extreme cases are $a = 0$, which produces a uniform mesh, and $a = 1$, which produces a mesh equidistributing the arc-length monitor function. Making use of the mesh transformation $\mathbf{x} = \mathbf{x}(\boldsymbol{\xi})$, (1) can be expressed as

$$ds = [d\boldsymbol{\xi}^T \boldsymbol{J}^T \boldsymbol{M} \boldsymbol{J}\, d\boldsymbol{\xi}]^{1/2}, \tag{2}$$

where $\boldsymbol{J}$ is the Jacobian of the transformation.

The equidistribution principle follows from (2): if $u(\mathbf{x}(\boldsymbol{\xi}))$ is to have the same value ds along any arc element in the computational domain with fixed length $[d\boldsymbol{\xi}^T d\boldsymbol{\xi}]^{1/2}$, then (2) must be independent of the coordinate $\boldsymbol{\xi}$. This implies that $\boldsymbol{J}^T \boldsymbol{M} \boldsymbol{J}$ should be independent of $\boldsymbol{\xi}$, or

$$[d\boldsymbol{\xi}^T \boldsymbol{J}^T \boldsymbol{M} \boldsymbol{J}\, d\boldsymbol{\xi}]^{1/2} = [d\boldsymbol{\xi}^T \widetilde{\boldsymbol{M}}\, d\boldsymbol{\xi}]^{1/2}, \tag{3}$$

where $\widetilde{\boldsymbol{M}}$ is a constant and hence $\boldsymbol{\xi}$-independent matrix. If a coordinate transformation can be found which satisfies (3), u will have the same variation at any point in Ω_p along any arc of length

$$\left[\left(\tfrac{\partial \mathbf{x}}{\partial \xi} d\xi + \tfrac{\partial \mathbf{x}}{\partial \eta} d\eta \right)^T \left(\tfrac{\partial \mathbf{x}}{\partial \xi} d\xi + \tfrac{\partial \mathbf{x}}{\partial \eta} d\eta \right) \right]^{1/2}.$$

A transformation satisfying (3) for some matrix $\widetilde{\boldsymbol{M}}$ will be called an *equidistribution*, and (3) an *equidistribution principle*.

Usually (3) cannot be satisfied by the coordinate transformation on the whole computational domain. However, if (3) is weakened so that the transformation is only required to satisfy (3) locally; that is, we only require $\widetilde{\boldsymbol{M}}$ to be constant along a given coordinate line, it is possible to find a local equidistribution on Ω_p. In 2D this leads to the system:

$$\left[\begin{pmatrix} \frac{\partial x}{\partial \xi} \\ \frac{\partial y}{\partial \xi} \end{pmatrix}^T \boldsymbol{M} \begin{pmatrix} \frac{\partial x}{\partial \xi} \\ \frac{\partial y}{\partial \xi} \end{pmatrix} \right]^{1/2} = c_1(\eta), \qquad \left[\begin{pmatrix} \frac{\partial x}{\partial \eta} \\ \frac{\partial y}{\partial \eta} \end{pmatrix}^T \boldsymbol{M} \begin{pmatrix} \frac{\partial x}{\partial \eta} \\ \frac{\partial y}{\partial \eta} \end{pmatrix} \right]^{1/2} = c_2(\xi), \tag{4}$$

where $c_1(\eta)$ and $c_2(\xi)$ are constant in the ξ and η directions respectively. These constants are eliminated by numerical differencing.

Instead of using the scaled arc-length matrix $\boldsymbol{M}$, in practice we modify $\boldsymbol{M}$ as $\boldsymbol{M} = k \nabla u \cdot \nabla u^T + \boldsymbol{I}$, where $k = a^2/(1 + b \nabla u^T \nabla u)$. The parameter $b \geq 0$ is used

to prevent problems where extremely small mesh spacing or mesh tangling could occur, that is when $|\nabla u|$ is very large.

System (4) will determine the internal mesh points. In [8] a combination of Dirichlet and Neumann conditions are used along $\partial\Omega_c$:

$$x(0,\eta) = y(\xi,0) = 0, \quad x(1,\eta) = y(\xi,1) = 1, \tag{5}$$

$$\frac{\partial x}{\partial \eta}(\xi,0) = \frac{\partial x}{\partial \eta}(\xi,1) = \frac{\partial y}{\partial \xi}(0,\eta) = \frac{\partial y}{\partial \xi}(1,\eta) = 0, \tag{6}$$

where $\xi,\eta \in [0,1]$. The Dirichlet conditions are consistent with the requirement that there are mesh points on the boundary of the domain. The Neumann orthogonality conditions are arbitrary, and in fact can cause smoothness issues near the domain boundaries. As an alternative, we follow [9] and apply the 1D EP,

$$(M(x)x_\xi)_\xi = 0, \quad x(0) = 0, \quad x(1) = 1, \tag{7}$$

to determine $x(\xi,0)$, $x(\xi,1)$, $y(0,\eta)$ and $y(1,\eta)$. The 1D analog of the system (4), given in (7), has previously been solved by DD methods in [3, 5, 6].

3 Classical Schwarz Domain Decomposition

For the two dimensional mesh adaptation problem, the computational domain $\Omega_c = [0,1]\times[0,1]$, can either be decomposed in just the ξ or just the η directions, or in both directions. This results in "strip" or "block" configurations of subdomains respectively. Here we discuss DD applied in the ξ direction only. That is, we decompose the ξ interval $[0,1]$ into subintervals $[\alpha_\xi^i, \beta_\xi^i]$, $i = 1,\ldots,S$, where $\alpha_\xi^1 = 0$, $\beta_\xi^S = 1$, and we assume the subintervals satisfy the overlap conditions:

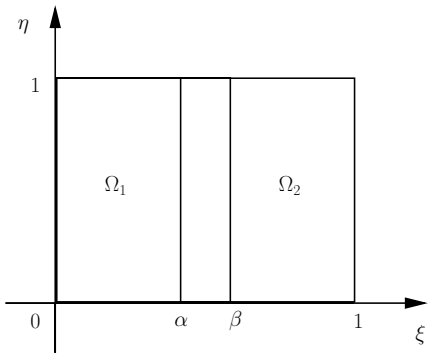

Fig. 1 DD in ξ using in 2 subdomains.

$$\alpha_\xi^i < \alpha_\xi^{i+1} < \beta_\xi^i < \beta_\xi^{i+1}.$$

The resulting decomposition has S subdomains, denoted by $\Omega_i = [\alpha_\xi^i, \beta_\xi^i]\times[0,1]$ for $i = 1,\ldots,S$. The boundary conditions (5–6) or (7) are used along the ends of each strip and transmission conditions are specified along the newly created interfaces.

Consider the 2D adaptive mesh system, (4), for the $S = 2$ case. We split Ω_c into subdomains Ω_1 and Ω_2 as in Figure 1. Let x_i^n denote the subdomain solution on Ω_i, for $i = 1,2$. We consider the following DD iteration: for $n = 1,2,\ldots$, solve

$$\left[\begin{pmatrix} \frac{\partial x_i^n}{\partial \xi} \\ \frac{\partial y_i^n}{\partial \xi} \end{pmatrix}^T \boldsymbol{M}(x_i^n, y_i^n) \begin{pmatrix} \frac{\partial x_i^n}{\partial \xi} \\ \frac{\partial y_i^n}{\partial \xi} \end{pmatrix}\right]^{1/2} = c_1(\eta), \tag{8}$$

$$\left[\begin{pmatrix} \frac{\partial x_i^n}{\partial \eta} \\ \frac{\partial y_i^n}{\partial \eta} \end{pmatrix}^T \boldsymbol{M}(x_i^n, y_i^n) \begin{pmatrix} \frac{\partial x_i^n}{\partial \eta} \\ \frac{\partial y_i^n}{\partial \eta} \end{pmatrix}\right]^{1/2} = c_2(\xi), \tag{9}$$

for $i = 1,2$ and $\boldsymbol{\xi} \in \Omega_i$. The classical Schwarz method uses the transmission conditions

$$x_1^n(\beta,\eta) = x_2^{n-1}(\beta,\eta), \quad y_1^n(\beta,\eta) = y_2^{n-1}(\beta,\eta), \tag{10}$$

$$x_2^n(\alpha,\eta) = x_1^{n-1}(\alpha,\eta), \quad y_2^n(\alpha,\eta) = y_1^{n-1}(\alpha,\eta). \tag{11}$$

On $\partial(\Omega_c \cap \Omega_i)$ the boundary conditions (5) are used, along with the 1D EP to determine $x(\xi,0)$, $x(\xi,1)$, $y(0,\eta)$ and $y(1,\eta)$.

Each DD iteration requires a pair of PDEs to be solved, each a "smaller" version of the local EP (4). These problems are solved in an iterative manner: given initial approximations to be used along interfaces, the PDEs (8–9) are solved, and then solution information along the interfaces is exchanged between subdomains. The PDEs are then solved again, now with updated boundary data, and the process repeats. By iterating, the subdomain solutions converge to the desired solution $\mathbf{x}$ on their respective subdomains. As is well known, classical Schwarz requires the subdomains to overlap [4].

4 Optimized Boundary Conditions

Classical Schwarz is known to converge slowly. As a way to remedy this, we propose the use of higher order, Robin type, transmission conditions along the artificial interfaces. As before, we decompose $\Omega_c = [0,1] \times [0,1]$ into subdomains $\Omega_1 = [0,\beta] \times [0,1]$ and $\Omega_2 = [\alpha,1] \times [0,1]$, where $\alpha \leq \beta$.

We define, for any differentiable functions $x(\xi,\eta)$ and $y(\xi,\eta)$, the operators

$$B_1(x) = x_\xi + px, \qquad B_2(x) = x_\xi - px,$$
$$B_3(x,y) = S_1(x,y) + px, \quad B_4(x,y) = S_1(x,y) - px,$$

where

$$S_1(x,y) = \sqrt{\begin{pmatrix} x_\xi \\ y_\xi \end{pmatrix}^T M \begin{pmatrix} x_\xi \\ y_\xi \end{pmatrix}}, \quad M = \frac{a^2 w \cdot w^T}{1 + b w^T \cdot w} + I$$

and

$$w = \frac{1}{x_\xi y_\eta - x_\eta y_\xi} \left[u_\xi y_\eta - u_\eta y_\xi, -u_\xi x_\eta + u_\eta x_\xi\right]^T.$$

We propose two possible sets of transmission conditions. The first are simple linear Robin conditions using the derivative normal to the artificial boundaries:

$$\begin{aligned} B_1(x_1^n(\beta,\eta)) &= B_1(x_2^{n-1}(\beta,\eta)), \quad B_1(y_1^n(\beta,\eta)) = B_1(y_2^{n-1}(\beta,\eta)) \\ B_2(x_2^n(\alpha,\eta)) &= B_2(x_1^{n-1}(\alpha,\eta)), \quad B_2(y_2^n(\alpha,\eta)) = B_2(y_1^{n-1}(\alpha,\eta)). \end{aligned} \tag{12}$$

The second set are of nonlinear Robin type, similar to those used in an optimized Schwarz algorithm for 1D mesh generation in [3]. We replace the x equations of (12) by

$$\begin{aligned} B_3(x_1^n(\beta,\eta),y_1^n(\beta,\eta)) &= B_3(x_2^{n-1}(\beta,\eta)y_2^{n-1}(\beta,\eta)) \\ B_4(x_2^n(\alpha,\eta),y_2^n(\alpha,\eta)) &= B_4(x_1^{n-1}(\alpha,\eta),y_1^{n-1}(\alpha,\eta)). \end{aligned} \tag{13}$$

Note, the mesh PDE (8) indicates that the nonlinear term S_1 in the operator B_3 is constant across the $\xi = \alpha$ and $\xi = \beta$ interfaces. Furthermore, as the system of equations resulting from (8-9) are already nonlinear, the nonlinear transmission conditions will not have a large impact on the cost of solving the system.

5 Numerical Implementation and Results

The local EP (4), the physical boundary conditions on Ω_c, and the transmission conditions (10, 11), (12) or (13), are discretized using standard finite differences on a uniform grid in the computational (ξ,η) variables. Second order centered differences are used, using the ghost value technique as needed at the boundaries to ensure the scheme is second order. The nonlinear transmission conditions require nonlinear, rather than linear, equations to be solved at the interface. This is not onerous as the whole system is solved with a Newton iteration.

In the examples we use the test function $u(x,y) = \left[1 - e^{15(x-1)}\right]\sin(\pi y)$. The function is shown, along with its locally equidistributed mesh, in Figures 2 and 3. The physical mesh (x,y) is generated by solving (4) using a grid of 18×18 uniformly spaced mesh points in Ω_c. For this example, we use an optimized Schwarz iteration, with transmission conditions (12), on 2 subdomains with 4 points of overlap in the ξ direction. Here the number of points of overlap refers to the number of shared grid points, the overlap width is approximately half of this number times $\Delta\xi$. We choose the parameters $a = 0.7$, $b = 0.05$ and $p = 2.3$. The mesh on subdomain 1 is shown in red, on subdomain 2 in blue, and the overlap region in purple. In general, the meshes obtained by the different methods will be visually indistinguishable from one another at convergence. To compare the DD methods we will plot their convergence histories.

In Figure 4 we plot the maximum error between the subdomain and single domain solutions for each of x_1^n, x_2^n, y_1^n, and y_2^n obtained using classical Schwarz. These are obtained over a 12 by 12 grid with 4 points of overlap in ξ and parameters

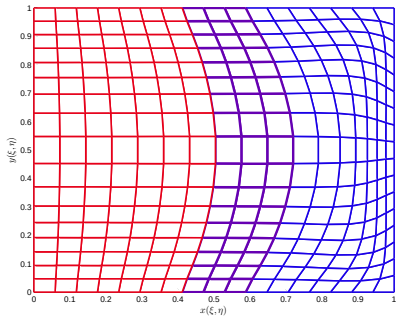

Fig. 2 Adaptive mesh generated for the test function.

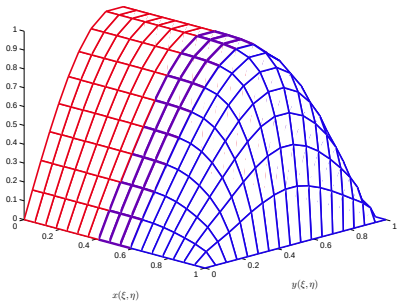

Fig. 3 The test function plotted using an adaptive mesh.

$a = 0.7$ and $b = 0.05$. As can be seen, each component of the solution converges at approximately the same rate, so we simplify our discussion by comparing the convergence of only x_1^n in the remaining figures.

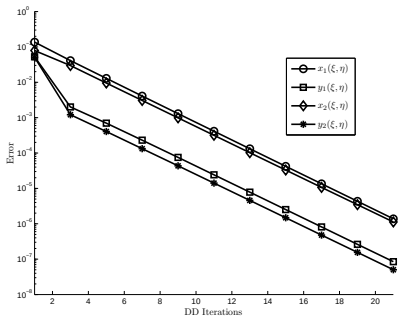

Fig. 4 Classical Schwarz convergence histories for each part of the solution, $x_{1,2}^n$ and $y_{1,2}^n$.

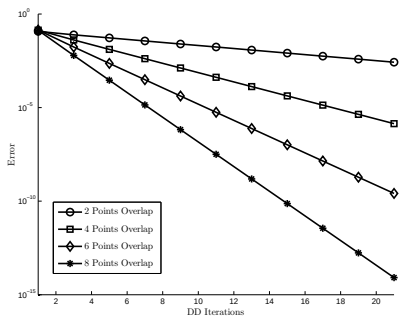

Fig. 5 Classical Schwarz convergence histories for varying amounts of overlap.

In Figure 5 we compare the classical Schwarz algorithm for varying amounts of overlap, using 2, 4, 6 and 8 points of overlap in the ξ direction. As expected, the rate of convergence improves as the overlap increases.

For the two possible optimized Schwarz iterations, we examine the effect of varying the parameter p in Figures 6 and 7. To generate these results we use a 12 by 12 mesh with two points of overlap in the ξ direction and parameters $a = 0.7$ and $b = 0.05$. For both types of transmission conditions, the best performance observed occurs for $p = 2$. Comparing the linear Robin condition (Figure 6) and nonlinear Robin condition (Figure 7), we see that the convergence histories for a general p are very similar. To examine these similarities, we plot the convergence histories for both optimized iterations for $p = 1, 2, 3$ on the same set of axes in Figure 8. We see that while the variations in this particular case are small, the nonlinear trans-

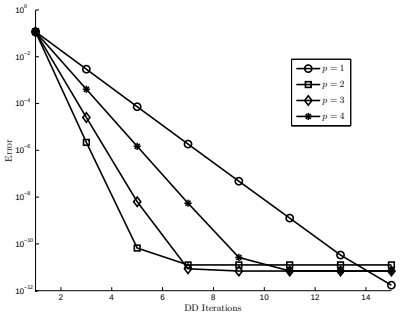

Fig. 6 Convergence histories for the Schwarz iteration using linear Robin conditions for varying p.

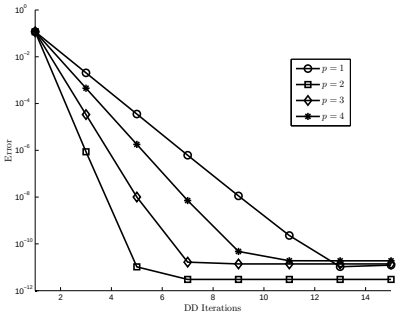

Fig. 7 Convergence histories for the Schwarz iteration using nonlinear Robin conditions for varying p.

mission conditions consistently outperform the linear Robin conditions. This is also observed in the results of Figure 9, in which we plot convergence histories for all three proposed DD algorithms. For this example we use a 12 by 12 mesh decomposed into two subdomains, with two points of overlap in ξ and parameters $a = 0.7$ and $b = 0.05$. In this example we see that both optimized Schwarz methods vastly outperform classical Schwarz, with the nonlinear transmission conditions slightly outperforming the linear Robin conditions.

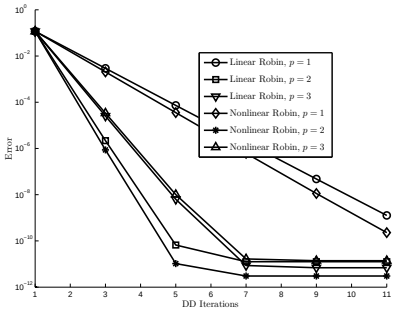

Fig. 8 Convergence histories for linear and nonlinear Robin transmission conditions with varying p.

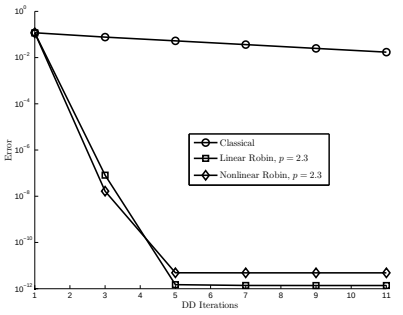

Fig. 9 Convergence histories for all three iterations considered.

Another way to assess the meshes obtained from a DD iteration is to compute a mesh quality measure. An equidistribution quality measure for each element K of the grid, $Q_{eq}(K)$, is presented in [7]. The maximum of Q_{eq} over all elements is 1 if and only if the equidistribution condition is satisfied exactly. The larger the value of $max_K Q_{eq}(K)$ the farther the mesh is from equidistributing $\boldsymbol{M}$. In Table 1 we compute the $max_K Q_{eq}(K)$ for the first five iterations of each proposed Schwarz algorithm. The zero column gives the mesh quality measure for the initial uniform 12×12

mesh and the ∞ column gives the mesh quality measure for the mesh obtained by solving system (4) over a single domain. Note, local equidistribution will not give a value of 1 for the mesh quality measure. We see that the meshes obtained by the optimized Schwarz algorithms rapidly give good meshes.

Table 1 Mesh quality measures for the grids obtained by the proposed Schwarz iterations.

Iterations	0	1	2	3	4	5	∞
Classical	1.6375	1.3630	1.3629	1.3178	1.3136	1.2795	1.1979
Linear Robin	1.6375	2.0076	1.1979	1.1979	1.1979	1.1979	1.1979
Nonlinear Robin	1.6375	2.0114	1.1979	1.1979	1.1979	1.1979	1.1979

6 Conclusion

In summary, we have proposed three different Schwarz DD iterations for obtaining 2D adaptive meshes defined by a local equidistribution principle. The numerical results show that the optimized methods provide a significant improvement over the slow convergence of classical Schwarz, with the nonlinear transmission conditions inspired by the work of [3] exhibiting the best results. Ongoing work includes the theoretical analysis of these DD approaches for 2D mesh generation and coupling the DD mesh generation with a DD solver for the physical PDE of interest.

References

1. de Boor, C.: Good approximation by splines with variable knots. In: Spline functions and approximation theory (Proc. Sympos., Univ. Alberta, Edmonton, Alta., 1972), pp. 57–72. Internat. Ser. Numer. Math., Vol. 21. Birkhäuser, Basel (1973)
2. de Boor, C.: Good approximation by splines with variable knots. II. In: Conference on the Numerical Solution of Differential Equations (Univ. Dundee, Dundee, 1973), pp. 12–20. Lecture Notes in Math., Vol. 363. Springer, Berlin (1974)
3. Gander, M., Haynes, R.: Domain decomposition approaches for mesh generation via the equidistribution principle. SIAM Journal on Numerical Analysis **50**(4), 2111–2135 (2012)
4. Gander, M.J.: Schwarz methods over the course of time. Electron. Trans. Numer. Anal. **31**, 228–255 (2008)
5. Gander, M.J., Haynes, R.D., Howse, A.J.: Alternating and linearized alternating schwarz methods for equidistributing grids (2012). In Press, 8 pages
6. Haynes, R., Howse, A.: Alternating schwarz methods for mesh equidistribution (Submitted Oct 5, 2012)
7. Huang, W., Russell, R.D.: Adaptive moving mesh methods, *Applied Mathematical Sciences*, vol. 174. Springer, New York (2011)
8. Huang, W.Z., Sloan, D.M.: A simple adaptive grid method in two dimensions. SIAM J. Sci. Comput. **15**(4), 776–797 (1994)
9. Mackenzie, J.A.: The efficient generation of simple two-dimensional adaptive grids. SIAM J. Sci. Comput. **19**(4), 1340–1365 (electronic) (1998)