PHYSOR 2004 -The Physics of Fuel Cycles and Advanced Nuclear Systems: Global Developments
Chicago, Illinois, April 25-29, 2004, on CD-ROM, American Nuclear Society, Lagrange Park, IL. (2004)

Adaptive solution of the multigroup diffusion equation on irregular structured grids using a conforming Finite Element Method formulation

Jean C. Ragusa *
CEA-Saclay, Service d'Etudes des Réacteurs et de Modélisations Avancées,
91 191 Gif sur Yvette, FRANCE

In this paper, a method for performing spatially adaptive computations in the framework of multigroup diffusion on 2-D and 3-D Cartesian grids is investigated.

The numerical error, intrinsic to any computer simulation of physical phenomena, is monitored through an *a posteriori* error estimator. In *a posteriori* analysis, the computed solution itself is used to assess the accuracy. By efficiently estimating the spatial error, the entire computational process is controlled through successively adapted grids.
Our analysis is based on a finite element solution of the diffusion equation. Bilinear test functions are used. The derived *a posteriori* error estimator is therefore based on the Hessian of the numerical solution.

KEYWORDS: mesh refinement, adaptive computation, a posteriori error estimator, numerical Hessian

1. Introduction

The trend in current nuclear engineering computations leads toward more unknowns to be solved for. This usually results from the following considerations:

- the often 3-D geometrical description can be made finer
- the number of energy groups can be increased
- the number of angles or moments in the S_N, SP_N, or P_N transport equation can also be increased
- transport methods can be applied to whole core calculations.

This will render the use of *a posteriori* error estimation and mesh refinement techniques obligatory for successfully converging solutions at a reasonable cost. Only spatial adaptivity is dealt with here. Recent research regarding angular adaptivity for instance can be found in the literature.

A posteriori error estimations and adaptive mesh refinement techniques allow for more pertinent solutions [1] because:

- the solution is reached with a smaller computational cost (fewer unknowns and shorter CPU time) than a series of computations performed with a sequence of successive meshes uniformly refined
- the solution is reached within the prescribed user-defined tolerance.

* Corresponding author, Tel. (+33)1 69 08 86 09, FAX: (+33) 1 69 08 94 90, E-mail: jean.ragusa@cea.fr

The paramount advantage of *a posteriori* error estimators lies in the fact that the current computed solution is used to assess the accuracy. By efficiently estimating the error, the entire computational process is controlled through successively adapted grids.

This paper is organized as follows: in Section 2, the Finite Element (FE) based solution process for the multigroup diffusion is briefly evoked; in Section 3, the error estimator is introduced; in Section 4, the geometric data structure is given; in Section 5, some results are presented.

2. Finite Element (FE) Solution

2.1 FE Primal Formulation

Let us consider the diffusion equation on a domain Ω with boundary $\partial\Omega=\partial\Omega_1\cup\partial\Omega_2$ ($\Omega \subset \mathbb{R}^d$, d = 2 or 3):

$$-\vec{\nabla}D\vec{\nabla}\Phi + \Sigma\Phi = S \text{ , on } \Omega \tag{1a}$$

$$\begin{cases} \Phi = 0, & \text{on } \partial\Omega_1 \\ D\dfrac{\partial\Phi}{\partial n} + \gamma\Phi = 0, & \text{on } \partial\Omega_2 \end{cases} \tag{1b}$$

The primal FE formulation is expressed as:

$$a(\Phi,\Psi) = f(\Psi), \qquad \forall\Psi \in H_0^1 \tag{2a}$$

$$\text{with } \begin{cases} a(\Phi,\Psi) = \displaystyle\int_\Omega \left(D\vec{\nabla}\Phi\cdot\vec{\nabla}\Psi + \Sigma\Phi\Psi\right) + \int_{\partial\Omega_2}\gamma\Phi\Psi \\ f(\Psi) = \displaystyle\int_\Omega S\Psi \end{cases} \tag{2b}$$

for any function Ψ belonging to the H_0^1 Sobolev space.

2.2 Polynomials Space and Triangulations Used

Bilinear test functions are used: they consist of a tensor product of the "classical" 1-D Lagrangian shape functions for each direction. Therefore the solution is sought after in the Q_1 polynomial space (the Q_1 space consists of polynomials of at most degree 1 in each variable). Gauss-Legendre quadrature is used to perform the numerical integration. The within group linear system is inverted either directly with a Cholesky factorization or iteratively with the conjugate gradient algorithm (depending on the user's choice). The outer iterations are accelerated by means of a standard Chebychev technique.

For the time being, our implementation is limited to Cartesian grids. Thus the triangulations are made of either rectangles in 2-D or right prisms in 3-D.

3. Error Estimator

3.1. Problem Statement

The numerical solution of a partial differential equation (PDE) problem on domain Ω is customarily performed on a discretization on domain Ω; hereafter the following terms will be

used for that discretization: grid or mesh or triangulation T_h. Let u be the exact solution of a PDE problem on Ω, u_h be a numerical solution computed on the triangulation T_h of Ω. In the framework of mesh adaptation, it is desired to reduce the approximation error $e_h = u - u_h$. In *a posteriori* error analysis, we seek an estimation of that approximation error on the i^{th} triangulation T^i_h in order to generate the next mesh T^{i+1}_h. *A posteriori* error estimation in FE analysis has been investigated and developed by many researchers [1] but computing bounds of the error e_h is usually a complex task with the added difficulty that it varies from one PDE to another (i.e. from one physical problem to another).

In order to circumvent the physical operator in the PDE problem, an indirect approach is followed to bound the approximation error. Let $\Pi_h u$ be the linear interpolant of u on elements of T_h and $\varepsilon_h = u - \Pi_h u$ the interpolation error. As stated in Céa's lemma [2], the approximation error is bounded by the interpolation error such that: $\|u - u_h\| \le C\|u - \Pi_h u\|$.

Thus, monitoring the interpolation error indirectly allows us to control the approximation error and finally the practical aspects of the problematic can be stated as follows:

1. construct an interpolation error estimator
2. define a metric based on the estimator
3. adapt the triangulation to satisfy the metric

3.2. Upper Bound for the Interpolation Error

Following Alauzet's work for triangles [3], an upper bound for the interpolation error is derived for the Q_1 finite element constructed on rectangles and prisms. We introduce Taylor's expansion of the interpolation error $u - \Pi_h u$ for a element K of T_h. For brevity's sake, the demonstration is given for the bi-dimensional case here.

Without loss of generality, let us write Taylor's expansion with integral remainder for any point $\boldsymbol{x}$ within element K, starting the expansion from point $\boldsymbol{a}$, which is one of the corners of K :

$$(u - \Pi_h u)(\boldsymbol{a}) = (u - \Pi_h u)(\boldsymbol{x}) + \left\langle \boldsymbol{x} - \boldsymbol{a}, \vec{\nabla}(u - \Pi_h u)(\boldsymbol{x}) \right\rangle + \int_0^1 dt \left\langle \boldsymbol{x} - \boldsymbol{a}, H_u\left(\boldsymbol{x}(1-t) + t\boldsymbol{a}\right)(\boldsymbol{x} - \boldsymbol{a}) \right\rangle \quad (3)$$

where $\langle \cdot, \cdot \rangle$ is the usual Euclidean scalar product, $\vec{\nabla}$ the gradient and H_u the Hessian. Since $\Pi_h u$ is interpolating and since an extremum of the error is sought after, the interpolation error ε reduces to:

$$\varepsilon_h^K = \max_{\boldsymbol{x} \in K} \left| (u - \Pi_h u)(\boldsymbol{x}) \right| = \max_{\boldsymbol{x} \in K} \left| \int_0^1 dt \left\langle \boldsymbol{x} - \boldsymbol{a}, H_u\left(\boldsymbol{x}(1-t) + t\boldsymbol{a}\right)(\boldsymbol{x} - \boldsymbol{a}) \right\rangle \right| \quad (4)$$

After some simple Euclidean algebra, a purely geometric *a posteriori* error estimator is derived for rectangles and right prisms in the case of bilinear basis functions.

Let E_K be the edges of element K, then it can be easily shown that:

$$\varepsilon_h^K \le c \max_{\mathbf{v} \in K} \max_{\boldsymbol{x} \in K} \left\langle \mathbf{v}, \left| H_u(\boldsymbol{x}) \right| \mathbf{v} \right\rangle \le \tilde{c} \max_{\mathbf{e} \in E_K} \max_{\boldsymbol{x} \in K} \left\langle \mathbf{e}, \left| H_u(\boldsymbol{x}) \right| \mathbf{e} \right\rangle \quad (5)$$

where c is a constant depending on the geometry type (triangle, rectangle, right prism…). Vectors **v** are "chord" vectors, crossing across element K from point $\boldsymbol{a}$. The second inequality in Eq. 5 is straightforward when noting that any vector within element K can be expressed as a linear combination of the edge vectors and then applying Minkowsky's rule.

In practice, the Hessian of u is not known. It will be numerically approximated on the mesh vertices and the numerical Hessian $\mathcal{H}_u$ will subsequently be defined in §3.5. Consequently, the interpolation error ε made on mesh K is *heuristically* estimated by:

$$\varepsilon_h^K = c \max_{\mathbf{e} \in E_K} \langle \mathbf{e}, \mathcal{M}_K \mathbf{e} \rangle, \tag{6a}$$

$$\text{where} \quad \mathcal{M}_K = \max_{p \in \{\text{vertices of mesh K}\}} |\mathcal{H}_u(p)|. \tag{6b}$$

Relation (4a) stipulates that the interpolation error on element K is proportional to the longest edge of mesh K, provided that the edge is measured in the metric $\mathcal{M}$. Therefore, controlling the measures or lengths of edges will lead to improved solutions as grids are successively refined where it is needed.

The obtained error estimator is referred to as *geometric* error estimator since it does not depend on the nature of the PDE solved. It is an *a posteriori* estimator since it is expressed in terms of the computed numerical solution.

3.3. Comments of the Use of the Hessian for Polynomial Orders greater than 1

The numerical Hessian is sometimes used to monitor the error in computations using FE polynomials basis of degree strictly greater than 1. We believe that approach to be wrong for polynomials of degree d strictly greater than 1.

In the case of polynomials of degree d strictly greater than 1, the $(d+1)^{th}$ derivatives of the solution should be examined rather than the 2^{nd} derivatives. This argument can easily be proven in 1-D: for instance, Poison's equation with an homogeneous source term leads to a quadratic analytical solution in 1-D. Solving Poison's equation using only one mesh and 2^{nd} order polynomials will lead to the exact solution. Nevertheless, the Hessian can be made arbitrarily great, forcing unnecessary mesh refinement.

3.4. Metric definition

For mesh refinement purposes, a metric $\overline{\overline{\mathcal{M}}}$ is defined such that :

$$\overline{\overline{\mathcal{M}}}_K = \frac{c}{\epsilon_{\text{user}}} \mathcal{M}_K \tag{7}$$

where ϵ_{user} is the user specified tolerance. Therefore, if an edge has a length of 1 in the metric $\overline{\overline{\mathcal{M}}}$, its interpolation error is of order ϵ_{user}.

The metric tensor $\overline{\overline{\mathcal{M}}}$ is diagonalizable since it is symmetric positive definite by construction and we have:

$$\overline{\overline{\mathcal{M}}} = \mathcal{R} \Lambda \mathcal{R}^{-1}, \tag{8}$$

where $\mathcal{R}$ is the matrix composed of the Hessian eigenvectors, and Λ the diagonal matrix composed of the d eigenvalues λ_i of the Hessian $\mathcal{H}_u$ (i = 1..d , where d is the problem dimension, i.e. 2 or 3).

Actually, the eigenvalues are modified as follows to account for meshes that would be

- either too small (λ_i is big near singularities of the solution)
- or too big (λ_i goes to zero where the solution is linear)

$$\tilde{\lambda}_i = \min\left(\max\left(\frac{c}{\epsilon_{\text{user}}} |\lambda_i|, \frac{1}{h_{\max}^2} \right), \frac{1}{h_{\min}^2} \right) \tag{9}$$

where h_{max} (resp. h_{min}) is the maximal (resp. minimal) allowed edge's length.

Since d eigenvalues are obtained for each mesh K, mesh refinement could be performed according to Eqs. 6a and 9 in any direction: a mesh could be refined in certain directions only, leading to anisotropic mesh refinement. This method has not been investigated yet for the purpose of this paper. It should be noted that the directions to be used to perform anisotropic refinement are the directions given by the eigenvectors of the Hessian (columns of matrix $\mathcal{R}$) and not the Cartesian unit vectors. In order to keep a structured grid, a criterion should be introduced to define the anisotropic refinement; Figure 1 shows an example of anisotropic and isotropic refinement for a given element K in 3-D). In Cartesian geometry, isotropic refinement in 2-D will result in a rectangle being cut into four rectangles. In 3-D, an isotropically refined right prism will be cut into eight smaller right prisms.

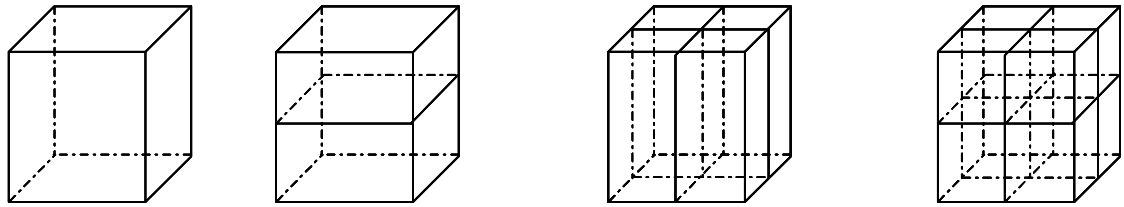

Fig.1 Original 3-D mesh and examples of anisotropic and isotropic refinements

3.5. Computation of the Numerical Hessian Matrix

One of the key points of the method is the evaluation of the numerical Hessian matrix $\mathcal{H}_u$. In order to compute the metric $\overline{\overline{\mathcal{M}}}$ at any given vertex P of the triangulation T_h, Taylor's expansion is written for the vertex of interest P. This yields for any points P_i in the neighborhood of P,

$$u(P_i) = u(P) + \overrightarrow{PP_i} \cdot \overrightarrow{\nabla u}(P) + \tfrac{1}{2} \left\langle \overrightarrow{PP_i}, \mathcal{H}_u(P) \overrightarrow{PP_i} \right\rangle + \dots \tag{10}$$

Points P_i are the 1st and 2nd nearest neighbors of point P; this usually results in an over-determined system since there are more neighbors than elements of matrix $\mathcal{H}_u$ (the number of unknowns per vertex in order to determine the Hessian matrix at a given vertex is: 3 unknowns in 2-D and 6 unknowns in 3-D). The resulting linear system is solved by a standard mean square root technique. The metric of element K is then computed according to Eq. 6b.

3.6. Isotropic Mesh Adaptation

In isotropic mesh adaptation, a mesh selected for refinement will be divided in 2d equal size meshes (i.e. a square in 2-D will become 4 smaller squares, and a cube in 3-D will become 8 smaller cubes, *c.f.* Figure 1).

In the case of isotropic refinement, no directions are privileged and the biggest eigenvalue is used $\tilde{\lambda} = \max_{i=1..d} \tilde{\lambda}_i$.

The criterion used for refining an element K is:

$$\frac{c}{\epsilon_{\text{user}}} \tilde{\lambda} \; e_K^2 > 1. \tag{11}$$

The above criterion could obviously allow for mesh coarsening but this feature has not been implemented.

4. Geometric Data Structure

Our work presented here focuses on 2-D and 3-D Cartesian geometries.

In order to assess the pertinence of mesh refinement techniques applied to the neutron diffusion equation, a specific geometric data structure needed to be created since the geometric data structure of our neutronics code CRONOS-2 [4] only allows for uniform regular meshes.

The following geometric data structure has been implemented:

- each volume has
  - a pointer to its faces (once a mesh has been selected for refinement, its faces need to be known for division)
- each face has
  - a pointer back to the two volumes it divides (this data is necessary to prepare the constrained conditions between two meshes)
  - as well as pointers to its edges (once a face has been selected for division, its edges need to be known for division)
- each edge has
  - a pointer back to the faces it separates (this data is necessary to prepare the constrained conditions between two meshes)
  - a pointer to its two vertices
- each vertex has a pointer to its coordinates

During the adaptation process, the former triangulation is used and specified meshes are divided. The previous data structure is therefore used and updated: new faces, edges, and vertices are added to the previous geometry to form the new one. Vertices are not re-numbered, therefore in order to reduce the linear system bandwidth, a classic Reverse Cuthill McKee algorithm is used (*c.f.* §5).

The geometry data structure utilized here is therefore a modified quadtree in 2-D (or octree in 3-D) [5]. Extension of the mesh refinement method to other, possibly more complex, geometries may require the integration of a generic mesh generator.

Obviously, such mesh refinements result in the creation of irregular vertices a.k.a. “hanging” nodes, which would require the use of non-conforming FEM techniques. The other strategy chosen here for handling continuity at those irregular nodes is to constrain the basis. For Q_1 elements, the constraint condition along an edge expresses that the irregular node value is a linear combination of regular nodes on the considered edge.

In order to circumvent difficulties when too many irregular nodes are present on an edge, Bank’s “one-irregular” and “three-neighbor” rules are implemented [6]. The “one-irregular” node rule limits the number of irregular nodes on an edge to one, and the “three-neighbor” rules states that any element having irregular nodes on three of its four edges must be refined. Figure 2 illustrates Bank’s rules.

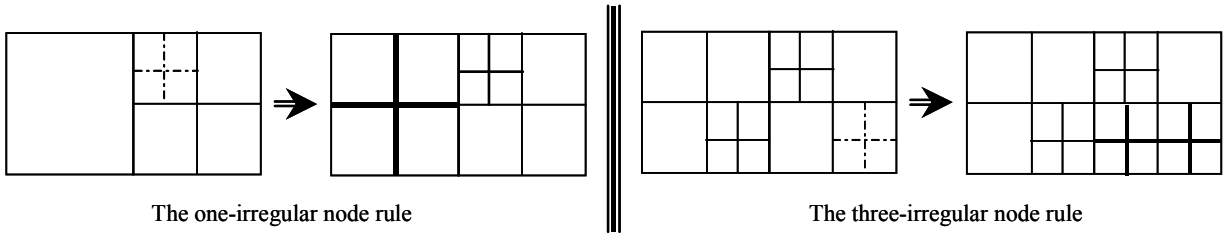

Fig.2 Illustrations of the “one-irregular” and the “three-irregular” nodes rules.

The use of lagrangian multipliers could be another way of handling continuity between elements but in our case, test functions are linear on the element’s edges and constraining the midpoint value is straightforward. Nevertheless, lagrangian multipliers could be preferable for higher order polynomials or if a mixed FEM were used.

5. Results

5.1. First Application

For testing purposes, several 1-group diffusion problems with a known fixed source were run. Figure 3 gives the initial mesh and the converged mesh after 2 grid adaptations (user’s tolerance of 5%) of a diffusion problem whose exact solution is:

$\sin(2\pi x)\sin(2\pi y)$ on $[0,1]\times[0,1]$ with Dirichlet boundary conditions.

The adapted mesh clearly reveals the solution sinusoidal variations.

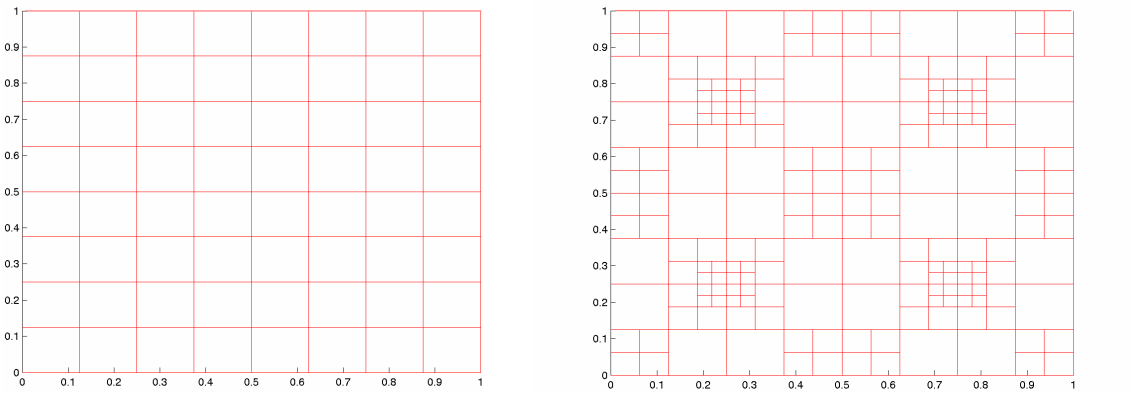

Fig.3 Initial mesh and final mesh converged at 5% for test problem (*c.f.* §5.1.)

5.2. Multigroup Case

5.2.1. Problem statement

In the case of a multigroup problem, a metric—and therefore an adapted mesh— could be computed for each energy group. Having a different mesh for each energy group would involve some non negligible bookkeeping for all the meshes and would require spatial projections and interpolations between source terms (fission source and scattering term) of different energy groups. Such a bookkeeping already exists in the framework of the multigroup transport code APOLLO-2 [7] where, for instance, the lower energy groups' meshes are constructed by refinement of the higher energy groups' meshes and the mesh sizes determined according to the mean free path of each group.

One way around having different meshes for each energy group could be to solve the multigroup equations on a unique mesh. This unique mesh could consist of elements whose Hessian evaluation , for any group, would verify Eq. 11. As a consequence, it is conceivable that, for some groups, the solution would be sought on an unnecessarily over-refined mesh. This would seriously diminish the gain expected from adaptivity.

5.2.2. A Goal Oriented Approach

Another approach for spatial adaptivity in the framework of multigroup equations is to focus, not necessarily on every multigroup flux, but on certain physical or observable quantities of interest, *e.g.*, reaction rates integrated over all energy groups or some groups only. For instance, the Hessian of the power distribution or of the capture reaction rate could be used to generate the new triangulation.

This goal-oriented approach is tested on a 64-assembly cluster, solved with 2 energy groups. The chosen cross-sections are those of a snapshot of a Main Steam Line Break accident [8], the power is strongly peaked near the North-East corner of the domain. The Hessian based on the power distribution is used for the mesh refinement.

Figure 4 provides the initial mesh and the converged mesh at the 4^{th} adaptation (tolerance of 1%). As can be seen, the mesh refinement takes place in the vicinity of the local power peak as well as near the boundaries where the fluxes vary rapidly from their maximum to zero. Figure 5 gives the sparsity of the linear system at the 4^{th} adaptation, before and after renumbering with the Cuthill-McKee algorithm. Table 1 summarizes the number of faces, edges, and vertices for each adaptation and compares these figures with their equivalent in the case of uniform refinement. After 4 adaptations, the saving in the number of vertices is of ~80%.

Table 1 Comparison of geometric data between adapted grids and uniformly refined grids

Adaptation number	# of faces	# of edges	# of vertices	# of vertices for a uniform mesh composed of the smallest size elements of the adapted mesh
Initial mesh	64	144	81	81
1	199	430	232	256
2	568	120	636	1024
3	164	3431	1778	4096
4	3340	7305	3751	16384

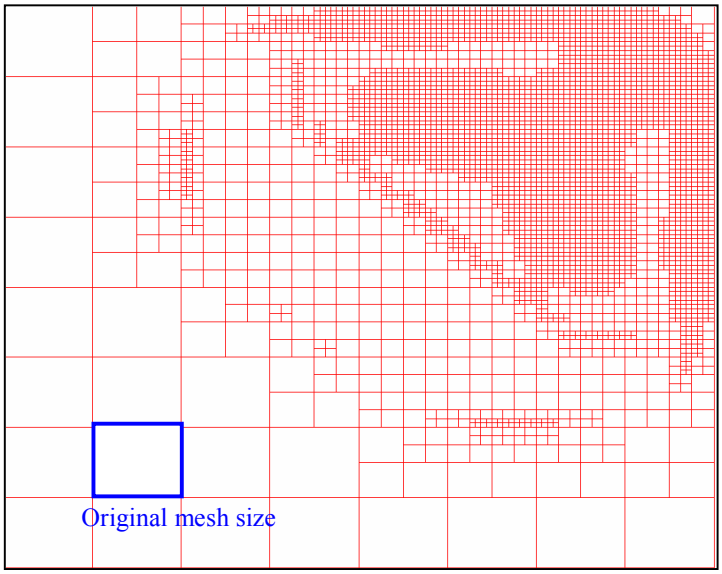

Fig.4 Initial mesh and final mesh converged at 1% for MSLB test problem

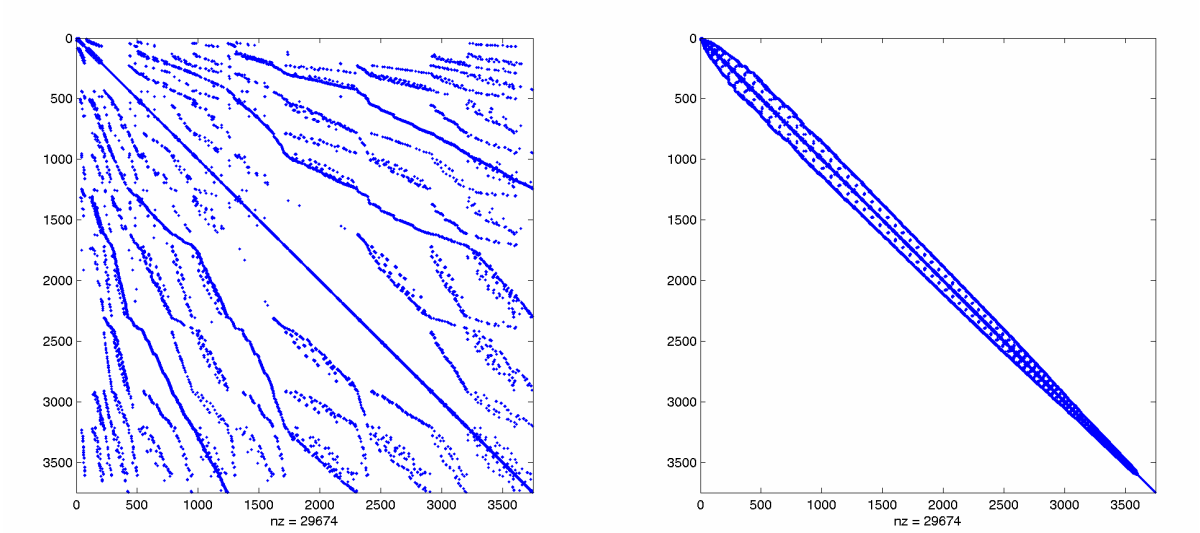

Fig.5 Linear system matrix at the 4th mesh adaptation; left: pattern resulting from the geometric data structure; right: pattern after Cuthill-McKee reordering

Conclusions

In this paper, we presented a method for performing spatially adaptive computations in the framework of multigroup diffusion on Cartesian grids.

The approximation error was controlled through a geometric *a posteriori* error estimator where the numerical solution itself was used to assess the accuracy. By efficiently estimating the spatial error, mesh refinement was utilized in regions where the error was the largest, leading, through successively adapted grids, to a numerical solution computed within a user specified tolerance.

Our analysis, based on a bilinear finite element solution, used the numerical Hessian of the solution as an *a posteriori* error estimator. Encouraging results were presented.

Future work may involve the use of a mesh generator to handle more complex geometries, the derivation of transition elements between meshes of different sizes to handle continuity, and the investigation of this method for mixed finite element methods.

References

1) Verfurth R. " A Review of A Posteriori Error Estimation and Adaptative Mesh refinement Techniques", Wiley & Teubner, 1996.
2) S. Brenner, L. Scott, *The Mathematical Theory of Finite Element Methods*, Springer, 2002.
3) Alauzet F., Frey P., "Estimateur d'erreur géométrique et métriques anisotropes pour l'adaptation de maillage: partie I: aspects théoriques" , Rapport de Recherche INRIA 4759. (in *French*)
4) J.-J. Lautard, S. Loubière, C. Fedon-Magnaud, "CRONOS: a Modular Computational System for Neutronic Core Calculations", IAEA Specialists Meeting on Advanced Calculational Methods for Power Reactors, Sept. 1990, Cadarache, France.
5) Remacle J.F., Shephard M.S., "An algorithm Oriented Mesh Database" International Journal for Numerical Methods in Engineering, 2002.
6) R.E. Bank, "The Efficient Implementation of Local Mesh Refinement Algorithms", in I Babuska, J Chandra and J Flaherty, editors, *Adaptive Computational Methods for Partial Differential Equations*, p.74-81, Philadelphia, 1983, SIAM
7) S. Loubière, R. Sanchez, M. Coste, A. Hébert, Z. Stankovski, C. Van der Gucht, I. Zmijarevic, "APOLLO2 Twelve Years Later", Proceeding of the ANS Mathematics and Computation International Conference, Madrid, Spain, Sept. 1999.
8) J. Ragusa, "Steady State Analysis of Multigroup Diffusion and SPN methods in an MSLB-like situation using the Cronos Code System", Proceeding of the ANS Mathematics and Computation International Conference, Gatlinburg, TN, USA, April 2003.