Contemporary Mathematics
Volume **218**, 1998
B 0-8218-0988-1-03051-X

A Domain Decomposition Based Algorithm For Non-linear 2D Inverse Heat Conduction Problems

Charaka J. Palansuriya, Choi-Hong Lai, Constantinos S. Ierotheou, and Koulis A. Pericleous

1. Introduction

Inverse heat conduction problems (IHCPs) appear in many important scientific and technological fields. Hence analysis, design, implementation and testing of inverse algorithms are also of great scientific and technological interest. The numerical simulation of 2-D and 3-D inverse (or even direct) problems involves a considerable amount of computation. Therefore, the investigation and exploitation of parallel properties of such algorithms are equally becoming very important [**9, 2**]. Domain decomposition (DD) methods are widely used to solve large scale engineering problems and to exploit their inherent ability for the solution of such problems.

An area of particular interest in IHCPs is the cutting of sheet material such as metal. An accurate simulation of the temperature distribution of the metal, subject to cutting, is vital in order to lengthen the life time of the cutting tool and to guarantee the quality of the cutting. In addition, the real-time simulation of such temperature distributions is of industrial interest. For example, it is important to regulate the cutter speed and coolant application in order to keep the temperature (especially at the cutter points) below a threshold. When the temperature rises above the threshold this will cause deformation of the metal or it may become fatigued. In reality, the accurate measurement of temperature at the cutter points is not possible. Therefore, a direct problem cannot be formulated. Inverse methods can be used to retrieve the temperature at these points. It has been shown that accurate estimates can be obtained using such methods [**1**]. IHCPs, such as the metal cutting problem described above, are more difficult to solve analytically than direct problems [**1**]. Therefore, various approximation methods have been developed to solve such problems. These include graphical[**10**], polynomial [**5**], Laplace transform [**7**], dynamic programming [**11**], finite difference [**3**], finite elements [**6**]

1991 *Mathematics Subject Classification.* Primary 65M55; Secondary 35K55, 65M06, 65Y05.
Key words and phrases. Domain Decomposition, Problem Partitioning, Metal Cutting Problems, Domain Parallelism, Domain-Data parallelism.
The first author is partially funded by the University of Greenwich.
Computing time on Origin 2000 was sponsored by Parallab at the University of Bergen.

and finite volume. Here we will use a finite volume based method. The main objective of this work is to study DD methods to solve IHCPs and to explore algorithms which are suitable for distributed/parallel computing environments.

This paper is organised as follows. First, a description of the mathematical model for the dimensionless 2D non-linear metal cutting problem is given. Second, the description of the problem partitioning is given. Different numerical schemes are used in different sub-domains in order to solve different sub-problems. Numerical results are shown for a metal cutting application. Third, the exploitation of the parallel properties of the numerical schemes are explained. The resulting parallel implementation uses MPI (Message Passing Interface) directives [**4**] and is suitable for network-cluster (distributed) computing as well as for traditional tightly-coupled multi-processor systems. Finally, some conclusions are drawn.

2. Dimensionless 2D Non-linear Metal Cutting Problems

The metal cutting problem considered here is a 2D thin sheet of metal defined in the domain $D = \{(x, y) : 0 < x < 1 \; and \; 0 < y < 1\}$. The material property is assumed to be homogeneous across the domain of interest and the following assumptions are made for an idealised cutting :- (1) the application of a cutting tool at the cutter points is equivalent to the application of a source at these points, (2) no phase changes occur during cutting and (3) the thickness of the cutter is negligible. The cutting is considered to be applied along the y-axis at $x = x_c$. Assumption (1) introduces an unknown source of strength $Q_c(y, t)$ at x_c and together with assumption (2), the cutting problem can be described by the dimensionless 2D non-linear, unsteady, parabolic, heat conduction equation,

$$\frac{\partial u}{\partial t} \; = \; \frac{\partial}{\partial x}(k(u)\frac{\partial u}{\partial x}) \; + \; \frac{\partial}{\partial y}(k(u)\frac{\partial u}{\partial y}) \; + \; Q_c(y,t)\delta(x - x_c) \; \in D, \tag{1}$$

subject to initial condition $u(x, y, 0) = U_i(x, y)$, boundary conditions $u(0, y, t) = B_0(y, t)$, $u(1, y, t) = B_1(y, t)$, $u(x, 0, t) = C_0(x, t)$ and $u(x, 1, t) = C_1(x, t)$. Here $u(x, y, t)$ is the temperature distribution, $k(u)$ is the conductivity of the metal, $Q_c(y, t)$ is the unknown source being applied at $x = x_c$, $\delta(x - x_c)$ is the Dirac delta function and U_i, B_0, B_1, C_0 and C_1 are known functions.

Assumption (3) suggests the continuity of the function $\frac{\partial u}{\partial t}$ at $x = x_c$, which in turn suggests that $\int_{x_c^-}^{x_c^+} \frac{\partial u}{\partial t} dx = 0$. Here x_c^- denotes a spatial point just to the left of x_c and x_c^+ denotes a spatial point just to the right of x_c. Hence the equivalent source strength can be retrieved by integrating (1) from $x = x_c^-$ to $x = x_c^+$ to give

$$k(u)\frac{\partial u}{\partial x}|_{x_c^+} \; - \; k(u)\frac{\partial u}{\partial x}|_{x_c^-} \; + \; \frac{\partial}{\partial y}(k(u)\frac{\partial u}{\partial y})(x_c^+ - x_c^-) + Q_c(y,t) \; = \; 0 \tag{2}$$

Assumption (3) also suggests that equation (2) can be truncated to:

$$k(u)\frac{\partial u}{\partial x}|_{x_c^+} \; - \; k(u)\frac{\partial u}{\partial x}|_{x_c^-} \; + \; Q_c(y,t) \; = \; 0 \tag{3}$$

That is, heat fluxes just to the left and just to the right of x_c must be known. Temperature sensors are attached at $x = x_s$, such that $0 < x_s < x_c < 1$, and let the temperature measured by means of the temperature sensors be $u(x_s, y, t) = u^*(y, t)$. It is not necessary to have x_s being less than x_c. Similar problem partitioning can be generated for $0 < x_c < x_s < 1$. The measured temperatures are used to retrieve temperatures at the cutting points. Such inverse methods avoid the basic

difficulties of a direct method since remote temperatures can be measured more easily and accurately.

For computer simulation purposes, the sensor temperatures are modelled by the function $u^*(y,t) = \alpha y(y-1)^2 sin(\omega t)$. Its maximum value is generated by the amplitude, α. Its variation with respect to time is generated by the angular frequency ω.

3. Problem Partitioning

Problem partitioning is a DD method applied at the mathematical/physical problem level. In other words, decomposition is carried out by only considering the problem at this level [**8**]. In order to solve the inverse problem given in (1) with the additional condition available at $x = x_s$, problem partitioning is carried out to produce three sub-domains, such that each subproblem may define different numerical algorithms. The three sub-domains are, $D_1 = \{(x,y) : 0 < x < x_s \; and \; 0 < y < 1\}$, $D_2 = \{(x,y) : x_s < x < x_c \; and \; 0 < y < 1\}$, and $D_3 = \{(x,y) : x_c < x < 1 \; and \; 0 < y < 1\}$. This problem partitioning removes the unknown source term $Q_c(y,t)$ and the Dirac delta function associated with it from the differential equations. The three sub-problems can be written as follows:

SP_1: $\frac{\partial u_1}{\partial t} = \frac{\partial}{\partial x}(k(u_1)\frac{\partial u_1}{\partial x}) + \frac{\partial}{\partial y}(k(u_1)\frac{\partial u_1}{\partial y}) \in D_1$
subject to $u_1(x,y,0) = U_i(x,y)$, $u_1(0,y,t) = B_0(y,t)$,
$u_1(x_s,y,t) = u^*(y,t)$, $u_1(x,0,t) = C_0(x,t)$, $u_1(x,1,t) = C_1(x,t)$.

SP_2: $\frac{\partial u_2}{\partial t} = \frac{\partial}{\partial x}(k(u_2)\frac{\partial u_2}{\partial x}) + \frac{\partial}{\partial y}(k(u_2)\frac{\partial u_2}{\partial y}) \in D_2$
subject to $u_2(x,y,0) = U_i(x,y)$, $u_2(x_s,y,t) = u^*(y,t)$,
$\frac{\partial u_2(x_s,y,t)}{\partial x} = \frac{\partial u_1(x_s,y,t)}{\partial x}$,$u_2(x,0,t) = C_0(x,t)$, $u_2(x,1,t) = C_1(x,t)$.

SP_3: $\frac{\partial u_3}{\partial t} = \frac{\partial}{\partial x}(k(u_3)\frac{\partial u_3}{\partial x}) + \frac{\partial}{\partial y}(k(u_3)\frac{\partial u_3}{\partial y}) \in D_3$
subject to $u_3(x,y,0) = U_i(x,y)$, $u_3(x_c,y,t) = u_2(x_c,y,t)$,
$u_3(1,y,t) = B_1(y,t)$, $u_3(x,0,t) = C_0(x,t)$, $u_3(x,1,y) = C_1(x,t)$.

Since the temperature values are given at $y = 0$, $y = 1$, $x = 0$ and there are temperature sensors located at $x = x_s$, Dirichlet boundary conditions are defined at the boundary of D_1. Solutions of the differential equation provide the required data to calculate the heat flux $\frac{\partial u}{\partial x}(x_s,y,t)$. Therefore, with the knowledge of the temperatures $u(x_s,y,t)$ acquired by the temperature sensors at $x = x_s$, an initial value problem can be formulated in D_2. $u(x_c,y,t)$ values are obtained by solving this initial value problem. Finally, with the calculated temperatures $u(x_c,y,t)$, another Dirichlet problem can be formulated in D_3. The above three subproblems are well-defined [**1**] [**12**], and a unique solution exists for each of them. The direct sum of these subproblem solutions gives the temperature distribution of the original problem, i.e.

$$u(x,y,t) = \begin{cases} u_1(x,y,t), & x \in D_1 \\ u_2(x,y,t), & x \in D_2 \\ u_3(x,y,t), & x \in D_3 \end{cases} . \tag{4}$$

3.1. Numerical schemes. To solve the problems in SP_1 and SP_3 a first order forward difference approximation of the temporal derivative and a second order FV approximation of the spatial derivatives are used. A five-point explicit scheme is produced from the approximation. Dropping the subscript used in denoting the sub-domains, the explicit scheme for the sub-domains D_1 and D_3 can be written as,

$$u_{i,j}^{(n+1)} = r_x b_i^{(n)} u_{i-1,j}^{(n)} + r_x a_i^{(n)} u_{i+1,j}^{(n)} + (1 - r_x a_i^{(n)} - r_x b_i^{(n)} - r_y c_j^{(n)} - r_y d_j^{(n)}) u_{i,j}^{(n)} + r_y d_j^{(n)} u_{i,j-1}^{(n)} + r_y c_j^{(n)} u_{i,j+1}^{(n)} \tag{5}$$

where (i,j) denotes the (i,j)-th grid point, $r_x = \frac{\triangle t}{(\triangle x)^2}$, $r_y = \frac{\triangle t}{(\triangle y)^2}$, $a_i^{(n)} = \frac{k_{i+1,j}^{(n)} + k_{i,j}^{(n)}}{2}$, $b_i^{(n)} = \frac{k_{i-1,j}^{(n)} + k_{i,j}^{(n)}}{2}$, $c_j^{(n)} = \frac{k_{i,j+1}^{(n)} + k_{i,j}^{(n)}}{2}$, $d_j^{(n)} = \frac{k_{i,j-1}^{(n)} + K_{i,j}^{(n)}}{2}$, (n) denotes the time-step, $\triangle t$ is the step size along the temporal axis and $\triangle x$, $\triangle y$ are the grid spacing along the spatial axis x, y, respectively. The initial value problem in SP_2 is solved by employing a second order Euler Predictor-Corrector (P-C) method along the x-axis for each time-step. Again, the spatial derivatives are discretised using second order FV approximations and the time derivative with a first order finite difference approximation. The two step P-C method can be written as:

$$\begin{pmatrix} u \\ v \end{pmatrix}^* = \begin{pmatrix} u \\ v \end{pmatrix} + \triangle x \underline{f}\ , \quad \begin{pmatrix} u \\ v \end{pmatrix}^{\text{new}} = \begin{pmatrix} u \\ v \end{pmatrix} + \frac{\triangle x}{2} \left\{ \underline{f} + \underline{f}^* \right\}\ , \tag{6}$$

where $v = \frac{\partial u}{\partial x}$, $\underline{f} = \underline{f} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} v \\ \frac{1}{k(u)} (\frac{\partial u}{\partial t} - \frac{\partial}{\partial y}(k(u)\frac{\partial u}{\partial y}) - k'(u)v^2) \end{pmatrix}$ and $\underline{f}^* = \underline{f} \begin{pmatrix} u \\ v \end{pmatrix}^*$. A second order spatially accurate solution may be obtained for each of the three subproblems. Therefore, it is expected to have a second order spatially accurate global solution for the inverse problem (1). The effect of the local truncation error for SP_2 is minimised because of the small size of the sub-domain which usually consists of only a few Euler P-C steps. All experiments carried out gave stable results as long as the CFL condition r_x, $r_y \leq 0.25$ was satisfied.

3.2. Numerical results. Numerical results are obtained for equation (1) with $x_s = 0.5$, $x_c = 0.6$, $U_i(x,y) = 0$, $B_0(y,t) = 0$, $B_1(y,t) = 0$, $C_0(x,t) = 0$ and $C_1(x,t) = 0$. Sensor points are modelled as $u^*(y,t) = \alpha y(y-1)^2 \sin(\omega t)$, with $\alpha = 0.1$ and $\omega = 2\pi$. Non-linear heat conductivity is given by $k(u) = \frac{1}{1+u^2}$. Temperature distributions are shown for time $t = 0$ to $t = 0.5$ in Figure 1. The retrieved source/sink strength is also shown, Figure 2, it reflects the shape of the function used in the modelling of sensor temperatures, i.e. a sine function in time.

4. Exploiting Parallelism

Exploiting the parallel properties of an algorithm provides several key advantages. One of them is the expectation of a very fast execution of the algorithm. This in turn facilitates the real-time simulation (e.g. one-minute of temperature evolution is calculated using no more than one minute of computation time.). Another added advantage is that very large problem sizes (i.e. problem sizes that may not fit into the memory of a single-processing element) can be solved by using the total memory available from all the processors.

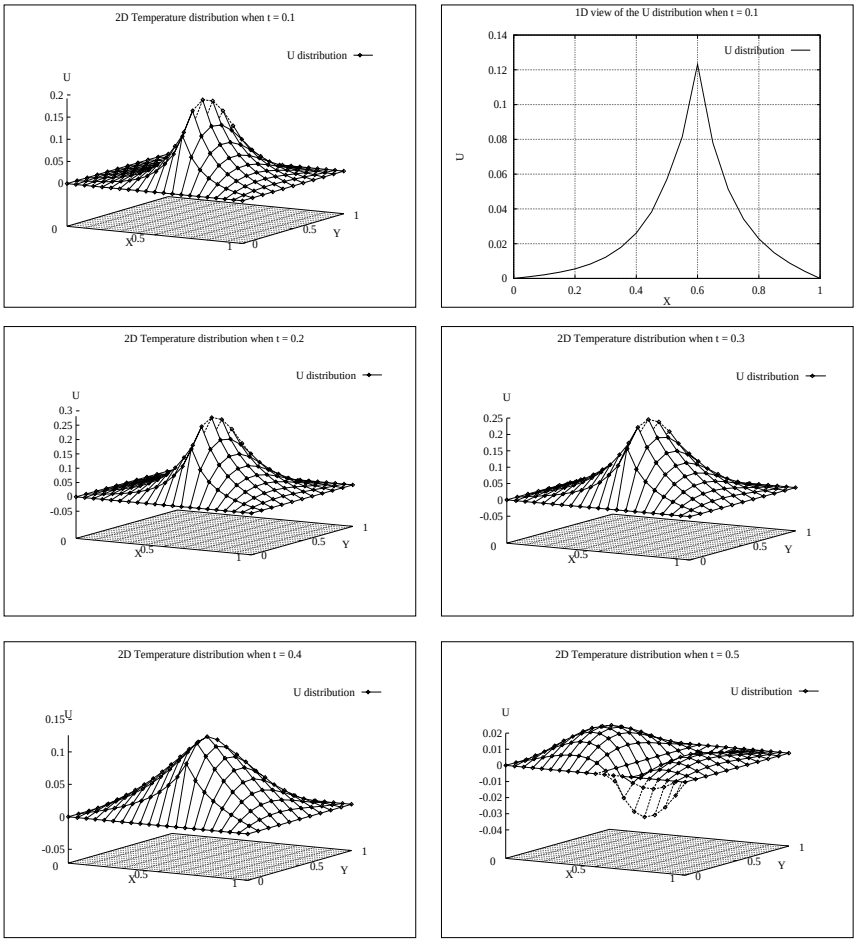

FIGURE 1. Temperature distributions from $t = 0.1$ to $t = 0.5$ and a 1D view at $t = 0.1$.

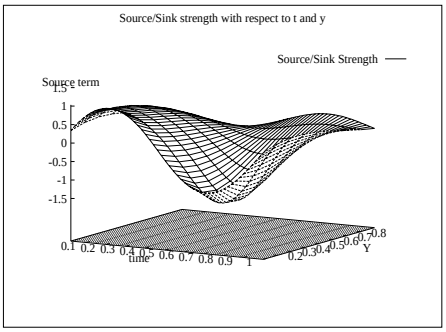

FIGURE 2. Source/Sink strength.

4.1. Domain-data parallelism. There are different ways of partitioning existing serial algorithms in order to utilise parallel and in particular distributed

environments. An advantage of using the DD approach to solve inverse problems is that the technique naturally provides a coarse-grained parallel algorithm. In other-words, each sub-domain generated due to the application of DD can be mapped directly to a processor and these sub-problems may be solved concurrently. In this paper this concept is referred to as "domain parallelism". The calculation performed in sub-domain D_2 is much smaller (due to sensor and cutter points being very close to each other) than D_1 and D_3. Also, calculations in D_1 can be carried out independently and gradients from D_2 and D_3 at $x = x_c$ are used to retrieve the source term. Considering these details, the calculations in sub-domains D_2 and D_3 are computed in one processor and the calculations in D_1 are computed in another. This minimises the communication and gives a better load balance among processors. That is, the domain parallelism requires two processors to solve the inverse problem defined by (1).

Domain parallelism has an obvious limitation in that it does not scale with an increasing number of processors. For the above cutting problem, only two processors are required. Data partitioning may be carried out within each sub-domain. In this paper we refer to this as "domain-data parallelism". In partitioning the data the number of grid-points is divided amongst the processors as evenly as possible. If N denotes the total number of grid points and P denotes the number of processors and if $\frac{N}{P}$ is not an integer, then some processors will have more grid points than others. As a result, the remaining data is distributed as evenly as possible so as to reduce the load imbalance amongst the processors.

4.2. Parallel results. The domain-data parallel version of the numerical algorithm is implemented using FORTRAN 77 with MPI directives. The parallel implementation is tested using a loosely coupled and tightly coupled multi-processor environments. The loosely coupled environment used is, a set of Sun Sparc 5 workstations connected together by an Ethernet network. The SGI Origin 2000 machine describe the tightly coupled multi-processor environment. Performance of the parallel implementation on the two platforms is shown in Figure 3 and shows that, the trend is similar for both platforms. Differences in speedups between the platforms, appear significant for smaller problem sizes (e.g., for 10000 and 20000 mesh points). This is due to the differences in communication startup latencies and message transfer times between the platforms. The Origin 2000 has a very low startup latency and message transfer times relative to a network of Sun Sparc 5 stations. This difference becomes less significant with the larger problem sizes (e.g., for 80000 mesh points).

5. Conclusions

The use of a numerical algorithm developed by applying DD to the problem domain, in order to retrieve heat source/sink at the cutter and the calculation of the temperature distribution is presented. It is shown that good parallelism can be exploited from the DD based algorithm by using domain-data partitioning as the parallelisation strategy. MPI is used to investigate the parallel performance of the domain-data parallel version of the algorithm in a loosely coupled and tightly coupled multi-processor environments. The parallel performance results show that domain-data parallelism can be utilised effectively in both network clusters and tightly coupled multiprocessor machines.

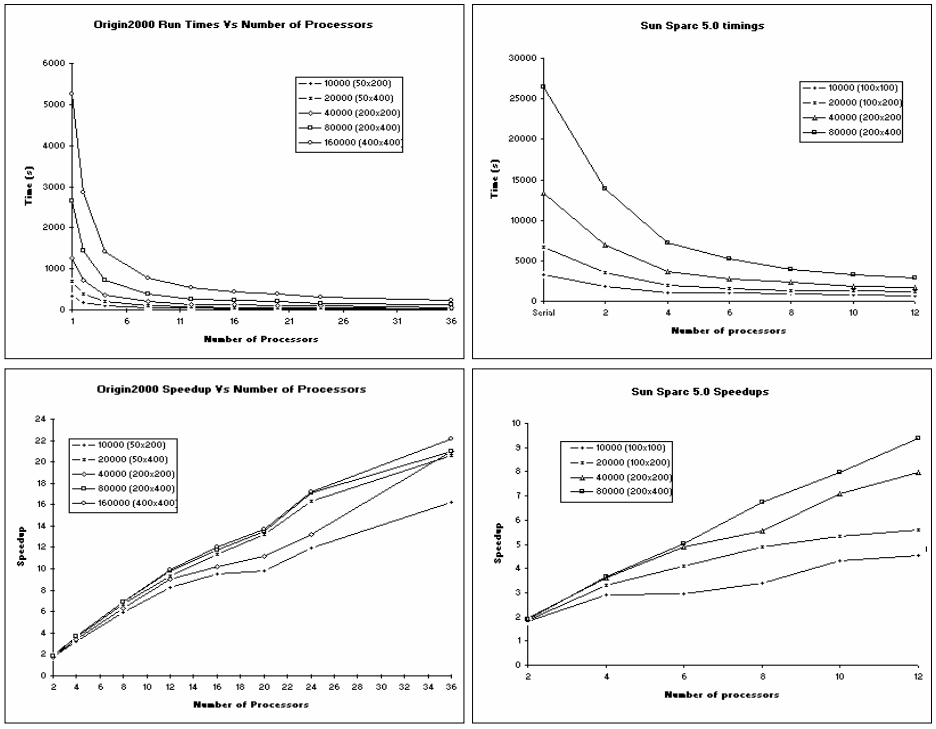

FIGURE 3. Parallel results for Origin 2000 and Sun Sparc 5s.

References

1. J. V. Beck, B. Blackwell, and C. R. St.Clair Jr, *Inverse heat conduction: Ill-posed problems*, Wiley-Interscience, 1985.
2. C-H.Lai, *A distributed algorithm for 1-d non-linear heat conduction with an unknown point source*, in proceedings of 9^{th} International Conference on Domain Decomposition (To Appear).
3. N. D'Souza, *Numerical solution of one-dimensional inverse transient heat conduction by finite difference method*, A.S.M.E. (1975), Paper No. 68-WA/HT-81.
4. Message Passing Interface Forum, *Mpi: A message-passing interface standard*, June 1995, available from http://www.mcs.anl.gov/mpi/.
5. I. Frank, *An application of least squares method to the solution of the inverse problem of heat conduction*, Heat Transfer **85C** (1963), 378–379.
6. G. W. Krutz, R. J. Schoenhals, and P. S. Hore, *Application of finite element method to the inverse heat conduction problem*, Num. Heat Transfer **1** (1978), 489–498.
7. G. Krzysztof, M. C. Cialkowski, and H. Kaminski, *An inverse temperature field problem of the theory of thermal stresses*, Nucl. Eng. Des. **64** (1981), 169–184.
8. C-H. Lai, *Diakoptics, domain decomposition and parallel computing*, The Computer Journal **37** (1994), 840–846.
9. C-H. Lai and C. J. Palansuriya, *A distributed algorithm for the simulation of temperatures in metal cutting*, High Performance Computing and Networking, Lecture Notes in Computer Science **1067** (1996), 968–969.
10. G Stolz, Jr., *Numerical solutions to an inverse problem of heat conduction for simple shapes*, Heat Transfer **82** (1960), 20–26.
11. D. M. Trujillo, *Application of dynamic programming to the general inverse problem*, Int. j. numer. methods eng. **12** (1978), 613–624.
12. D. Zwillinger, *Handbook of differential equations*, Academic Press Inc.,San Diego, 1989.

Centre for Numerical Modelling and Process Analysis, University of Greenwich, London SE18 6PF, UK
E-mail address: c.j.palansuriya@gre.ac.uk

Centre for Numerical Modelling and Process Analysis, University of Greenwich, London SE18 6PF, UK
E-mail address: c.h.lai@gre.ac.uk

Centre for Numerical Modelling and Process Analysis, University of Greenwich, London SE18 6PF, UK
E-mail address: c.s.ierotheou@gre.ac.uk

Centre for Numerical Modelling and Process Analysis, University of Greenwich, London SE18 6PF, UK
E-mail address: k.pericleous@gre.ac.uk