Parallelisation of Perturbation Analysis: the method of Multiple Scales applied to vibration problems

Raya Khanin and Matthew Cartmell

Department of Mechanical Engineering, University of Glasgow, Glasgow G12 8QQ, UK
e-mails: *raya@mech.gla.ac.uk* and *matthewc@mech.gla.ac.uk*

Extended Abstract

In this paper we explore ways to parallelise the computer implementation of the methods of perturbation analysis. Perturbation analysis of the problems of even moderate number of degrees of freedom will greatly benefit from being parallelised as it will allow analytical study of a wider class of problems which so far are being solved numerically.

We concentrate on the Multiple Scales (MS) method applied to vibration problems in Engineering. These problems are given by

$$[\ddot{x}(t)] + [A][\dot{x}(t)] + [B][x(t)] + \varepsilon[N]([x(t)], [\dot{x}(t)], [\ddot{x}(t)]) + \varepsilon \sum_{j=1}^{J} [P_j] \cos(\Omega_j t + \phi_j)[x(t)] =$$

$$= [f_0] + \sum_{k=1}^{K} [f_k] \cos(W_k t + \xi_k), \quad \varepsilon \ll 1 \tag{1}$$

Most nonlinearities in $[N]$ are expressible as polynomials of $[x(t)]$ or its derivatives $[\dot{x}(t)]$, $[\ddot{x}(t)$ (usually up to the third power but sometimes higher), e.g.

$$[N] = \varepsilon([\alpha_1][x]^2 + [\alpha_2][x]^3 + [\alpha_3][x][\dot{x}] + [\alpha_4][x][\ddot{x}] + ... = \varepsilon \sum_k [\alpha_k][x]^{\beta_k^1}[\dot{x}]^{\beta_k^2}[\ddot{x}]^{\beta_k^3}. \tag{2}$$

The main idea of the MS method is that the single independent variable, t, is split up into several new independent variables, $T_0, T_1, ..., T_M$. Although such variables are not precisely independent (each being related to the others due to a simple scaling relationship via the perturbation parameter) they can be treated as such. The MS method allows the construction of a set of perturbation equations from which so-called *secular terms* are removed. The resulting *solvability equations* provide solutions for the coefficients while the removal of secular terms ensures a uniform expansion of the solution.

The MS solution procedure has an inherent parallelism: it contains many steps wherein the same series of operations are performed on different sets of data. These operations are (1) writing the perturbation equations, (2) substitutions of the k-th order ε solutions into the $k+1$-th order ε equations, (3) simplification, (4) identification of secular terms, (5) finding resonance conditions, (6) transforming the set of solvability conditions into the autonomous modulation equations. For these operations little or no communication between processors is needed. All the separate runs are independent of each other, and thus can be executed in parallel. Each processor is assigned one or more of the runs and executes them in parallel with other processors (*processor farming*). The expression $map[f, x_1, x_2, ...x_n]$ is an example of processor farming parallelism, as $f[x_1], f[x_2], ...f[x_n]$ can be executed independently of each other.

A *Mathematica* package *MultipleScale.m* to implement this method on serial and parallel computer architecture has been developed. The flow of control in the MS solution procedure is sequential, except when a parallel procedure is called. The controller dispatches appropriate instructions to available processors using a Single Instruction Multiple Data model (SIMD), wherein all processors involved perform the same operations, but on a different set of data. The parallel version based on *Mathematica* Parallel Computing Toolkit has been tested on an SGI Origin 2000 multi-processor computer.

On-going research has shown that it is possible to effectively parallelise the application of the perturbation method. The algorithms developed here could be used for the parallelisation of other perturbation techniques.

This work has been supported by the Engineering and Physical Sciences Research Council of the UK under grant GR/L30749 at the University of Glasgow.