A Gradient Hybrid Parallel Algorithm to One-Parameter Nonlinear Boundary Value Problems

Dániel Pasztuhov[1] and János Török

Budapest University of Technology and Economics, Budapest, Hungary
dani@iit.bme.hu

Abstract. We present a novel global algorithm for parallel computers, suitable to solve nonlinear boundary value problems depending on one parameter. The existing scanning and solution following algorithms are extended by a gradient method which is performed on an artificial potential created from the equation system. All three components of the algorithm can be parallelized and thus used in a GRID network. We validate our algorithms on a few small examples.

1 Introduction

Engineers need a lot of computing power to solve problems about the behaviour of the reinforced concrete beams on the influence of several forces. [1] These problems can often be described as boundary value problems (BVP). In this paper we present a novel method to handle these problems.

The boundary value problems can be traced back to finding the solutions of a non-linear equation system in a multidimensional space. These solutions are always a collection one dimensional objects (lines). Once a small part of one of them is found it can be followed in both directions which is the first component of our algorithm. [4] We use two methods to find pieces of the solution a stochastic and a gradient one. [5] The latter is a new method in this field and is performed on a non-negative potential obtained by the transformation of the equation system, where the solutions are the minima of the potential with zero values.

The gradient extension does not make the algorithm scan the entire space much faster instead it can deliver the solutions much faster than the scanning algorithm. However, in principle we have to scan the whole space to find *all* solutions.

The aim of this work is to present the gradient algorithm. We implement the simplest possible solution to demonstrate the power of the new algorithm. As the dimension of the GRS [1] gets higher and the scanning of the whole GRS would need exponentially large times. The gradient algorithm with some extra calculation need helps to deliver the solutions earlier.

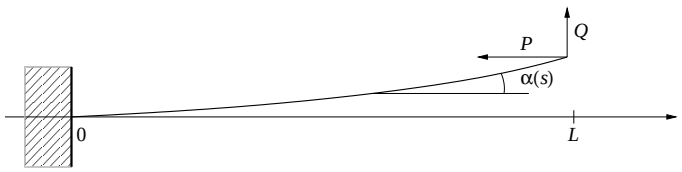

Fig. 1. Euler problem

2 Boundary value problem

If we get the deformations of the beam as an integration along the length, so position, forces and moments are known at one end of the beam, we are talking about an *initial value problem (IVP)*. Most of the cases, the position and/or forces and/or moments are given in both end of the beam, which transfers it to a *boundary value problem (BVP)*.

We will illustrate our method on the example of the axially compressed, uniform, elastic cantilever beam, illustrated in Fig. 1. The ordinary differential equation describing the shape of the beam in terms of the slope α as a function of the arclength s was first described by Euler:

$$EI\alpha'' + P\sin\alpha + Q\cos\alpha = 0. \tag{1}$$

The vertical force Q will be used as a small imperfection parameter which is *constant* during the loading process. The trajectories of this equation are uniquely determined by the three scalars $\alpha(0)$, $\alpha'(0)$, and P (the former ones being 'true' initial conditions, the latter one a parameter, Q is treated as a constant). However, we are not interested in all trajectories, only the ones which meet the boundary conditions $\alpha(0) = 0$ and $\alpha'(L) = 0$, which express zero slope at the left end and zero curvature at the right end of the beam respectively.

If we denote $x_1 \equiv \alpha'(0)$, $x_2 \equiv P$, the scalars x_i are called *global coordinates*, the space spanned by $[x_1, x_2]$ will be called *Global Representation Space* (GRS) of the BVP and we denote its dimension by D. In these problems there are $D-1$ equations to be solved which results in a line (one dimensional object) as the set of solutions. The function we have to solve is $f_1 : \alpha'(L) = 0$. This example is a simple one, in case of more complex problems GRS can have more dimensions, some of them exceed even the 20th dimension!

In order to solve globally a BVP in *moderate dimensions*, our algorithm discretizes the GRS into hypercubes and splits up the cubes into simplices (that are triangles in two dimensions and tetrahedron in three dimensions). [2] The original IVP (Eq. 1) is solved in each apex of the simplex, and the solution, if any is determined for each side by a linear interpolation algorithm. Solution points of all the simplex sides are taken as the global results of the BVP.

An acceleration for simplex algorithm is the path-continuation extension: If we find a solution anywhere in the GRS, we suppose that this solution continues in two ways (as it is said, solutions are one-dimensional objects), so we

examine the neighbouring cubes of the solution provider cube to find the solutions sooner. Simplex algorithm extended by path-continuation method is called *hybrid* algorithm.

The parallel version of the hybrid algorithm [3] is implemented in the following way: The space is divided into large *primary hypercubes* (consisting of l^D hypercubes) which are handled by a slave processes in order to reduce the communication need among processes. The primary hypercubes are not distributed arbitrarily to the slaves but a weighting is applied. If a solution was found to leave a primary hypercube on a specific side the neighbouring primary hypercube is marked with the maximum weight. On the other hand primary hypercubes with the most unchecked neighbours have higher weights in order to scan more distant hypercubes earlier and thus find the solutions earlier.

As number of dimensions increase, size of GRS increases *exponentially*. Hybrid algorithm is good enough for problems with dimensions not higher than about 6. Problems with higher dimensions cannot be handled by this algorithm.

3 Gradient Method

There is more information in the functions than it is used by hybrid algorithm. The main new idea is that we construct a potential which has minima at the solutions:

$$U(p, x_1, x_2, \ldots, x_n) = \sum_{i=1}^{n} c_i f_i^2(p, x_1, x_2, \ldots, x_n), \tag{2}$$

where c_i denote positive constants. The above construction ensures that the value of U is always non-negative and zero values indicate the solution. Thus a gradient method may be used to find these points.

The gradient algorithm is implemented to step from every point to the neighbouring hypercube in the direction of the largest gradient. It stops if it would leave the examined parameter space or if it found a local minimum, where the U is smaller than in the neighbouring hypercubes. The primary hypercube with the found local minima is marked with high weight for the hybrid algorithm irrespect of the value of U.

4 Implementation and Problems

The aim of the present work is to justify the effectiveness of the gradient algorithm without any further optimizations. There are many aspects that may render this algorithm useless. Since we have no *a priori* knowledge about the above potential it may contain too many local minima making the gradient algorithm useless. The different components of f_i are in general of different unit and thus can be of different magnitude which might introduce anomalies. In the followings we note other optimization or implementation possibilities in parallel with the present simplest choice of realization.

We have a parallel algorithm where the balance between different slave types must be synchronized. This requires an elaborate weighting of the unchecked primary hypercubes as well as a fine tuning of the slave types. We show here that even the dumbest choice can deliver a considerable performance increase. At the beginning we let the gradient algorithm run a few times ($1 - 10$ in our simple examples) and then we mark with high weight the primary hypercubes found as local minima by the gradient algorithm to be the first candidates for the hybrid algorithm then we switch back to the old algorithm.

The same simplicity is followed in choosing the initial points for the gradient algorithms which is done randomly. Since many gradient runs may find the same solution in the future it would be important that the starting points are well distributed in the GRS. In higher dimensions methods of determining attraction zone by some negative gradient algorithm might be also helpful.

The role of the c_i coefficients is to bring the values of the f_i of different units in the same magnitude. This can be done e.g. by an initial scan of the space where we calculate f in m points and set

$$c_i = \left(\frac{1}{m} \sum_{j=1}^{m} \left(f_i^{(j)} \right)^2 \right)^{-1} \tag{3}$$

Since there might be huge differences among quarters of the GRS the coefficients should be calculated for the perimeter of each gradient run. An other possibility is to let c_i evolve in time as the gradient algorithm advances but in this case one has to care about the algorithm making cycles. In our case the best choice was to take the simplest possibility: $c_i \equiv 1$.

We chose that the gradient method is stepping into the neighbouring hypercube in the direction of the largest gradient. This might be inappropriate in more complex problems where a conventional gradeint method should be used.

In summary the only difference compared to the hybrid algorithm is to run a few gradient algorithm slaves at the beginning and then switch back to the old algorithm with the found local minima being the first to be scanned by the simplex method.

The extra calculation of the gradient algorithm increases the overall computation need of the algorithm but the solutions may be found earlier. We also note that the gradient algorithm has much less computation need than the scanning one. The gradient algorithm generates the function values in the neighbouring hypercubes of the actual point. There are $2D$ such points. On the other hand to test whether a hypercube contains a solution the simplex algorithm has to calculate the functions at the corners which means 2^D function calls. It is also important to note that in the case of the gradient algorithm we do not check all the hypercubes of the primary hypercube but we follow the gradient which means again in average l^{D-1} factor advantage for the gradient method, where l is the linear size of the primary hypercube. This means that the gradient method can be considered instantaneous compared to the scanning algorithm in high dimensions.

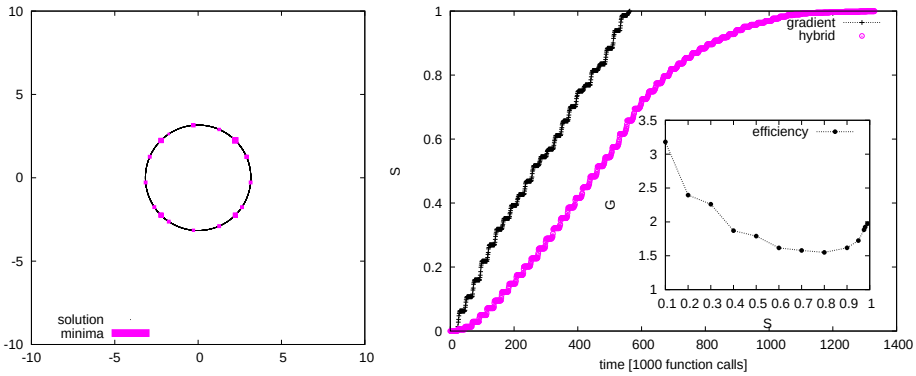

Fig. 2. Circle problem (a) The continuous line shows the solution curve the squares indicate the local minima found by the gradient algorithm, the area of the squares is proportional of its occurrence in the 100 runs. (b) The variation of S in time for both the gradient and hybrid algorithm. The inset shows the variation of the efficiency with the percentage of the found solutions.

5 Results

As we already mentioned the aim of the gradient algorithm is not to scan the GRS in less time but to find the solutions earlier than the stochastic algorithm. We chose to measure the time by calculating the number of function calls needed to find a primary hypercube with a solution in it. Measuring the time would be misleading as more complicated problems have functions with heavy computation need which requires most of the evaluation time while the simple examples presented here spend relatively more time for communication etc. Therefore we measure the time in 1000 function calls and denote it by τ. The time is measured independently for each slave.

In all test cases we did the following procedure: Two series of runs were done, one with the gradient algorithm and the other one with the hybrid algorithm. Each series consisted of 100 runs with different random seeds. We present the averaged results.

We note by $S(\tau)$ the ratio of the found primary hypercubes with solutions compared to the total one. The efficiency of the gradient algorithm is defined by the ratio of the time needed to find S part of the solutions with the hybrid algorithm and with the gradient: $G \equiv \tau_{\mathrm{hybr}}(S)/\tau_{\mathrm{grad}}(S)$

The first test we performed is not a real problem. The function was chosen to be $x^2 + y^2 = 10$ which corresponds to no real BVP. The GRS space was set to the $[-10 : 10] \times [-10 : 10]$ space the size of the primary hypercube is 1.5. The solution is a circle with $\sqrt{10}$ radius (see Fig. 2. (a)). A single gradient algorithm was run at the beginning and only one slave was working. The efficiency is $1.5 - 2$. The time evolution of S for the gradient algorithm is quite linear indicating a scenario where the gradient algorithm found a solution and then

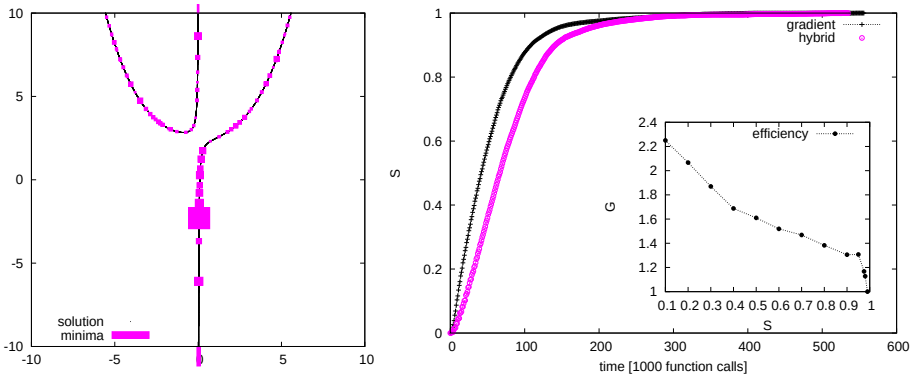

Fig. 3. Cantilever beam with $Q = 0.1$. Graphs are same as in Fig. 2

the path following algorithm completed it. The hybrid algorithm has a curved shape indicating an exponential distribution of the time when the first point of the solution was found. The time gain very well corresponds to the average time needed for the hybrid algorithm to find the first solution.

The second example consist of the already illustrated cantilever problem with a very small imperfection parameter $Q = 0.1$. The resulting GRS with the solution is presented on Fig. 3. Five gradient algorithms were run prior to the hybrid algorithm and both ran on 5 slaves simultaneously. Due to the small imperfection the solution line is cut into two distinct parts. The ratio of the primary hypercubes with solution is lower than before (5.7% instead of 8%). In spite of this the efficiency [Fig. 3 (b)] is less than in the previous example. This is due to the fact that the solutions are relatively long lines which takes a long time to follow. This process can only be done by one or two slaves and the others are free to look for new solution in a stochastic way. This example emphasizes the importance of the well planned parallelization of the algorithm.

We present the third example on Fig. 4 which is different from the previous one only in the imperfection parameter $Q = 2$ and the number of slaves which was set to 2. The efficiency changes only little: It gets worse for small S but gets better for large S. Where the low number of slaves does not let for a free scanning of GRS while the others are following a solution. On Fig. 4 (a) we can see that a local minimum with no solution at the point $(4.7, -8.4)$. It is also visible on the surface map of the potential on Fig. 6 (a). It has a considerable attraction range but with sufficient gradient runs the solutions are found with very high probability.

The last example we analyze here is a three dimensional one. The imperfection parameter is no longer a constant but may change on condition that its absolute value equals to $|Q|{=}P$. The GRS is 3 dimensional in this example with Q being the third dimension. The problem was run on two slaves. The solution probability is 0.6%. The position of the solution is shown on Fig. 5 and 6 (b).

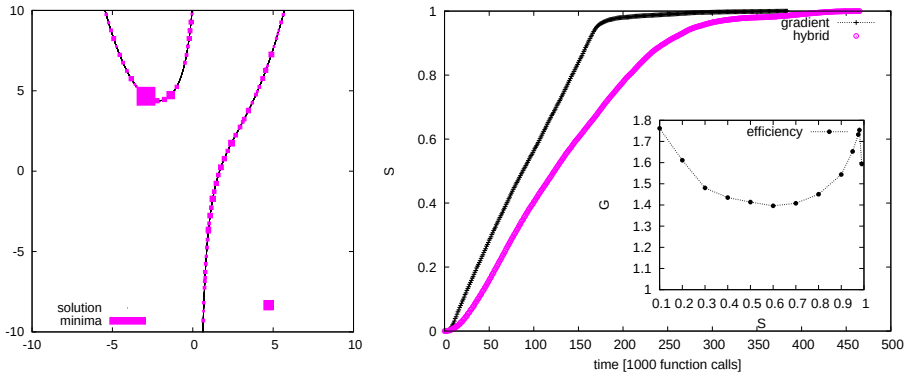

Fig. 4. Cantilever beam with $Q = 2.0$. Graphs are same as in Fig. 2

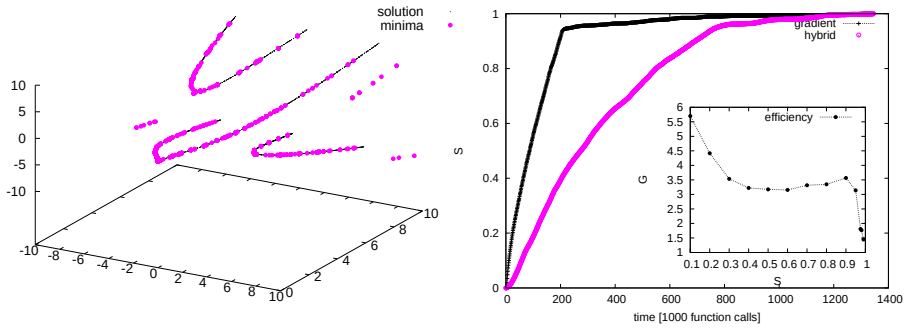

Fig. 5. Cantilever beam with $|Q|=P$ (three-dimensional GRS). (a) The continuous line shows the solution curve the points indicate the local minima found by the gradient algorithm (occurrence is not shown!). (b) Graph is same as in Fig. 2 (b).

It consists of three disjunct branches. The figure 5 also shows the local minima found by the gradient algorithm where it is obvious that it nicely finds the solution as well as other local minima lines. In spite of these false local minima the algorithm is very efficient and needs about 3 times less function calls than the hybrid algorithm to find the majority of the solution, which shows that the efficiency of the gradient algorithm compared to the hybrid one increases rapidly with the dimension of the GRS.

6 Conclusion

In this paper we introduced a gradient algorithm to find the solution of boundary value problems faster than by the existing scanning and solution following hybrid algorithm. We showed that even the easiest implementation of this algorithm brings a considerable time gain. This is achieved despite the fact that the

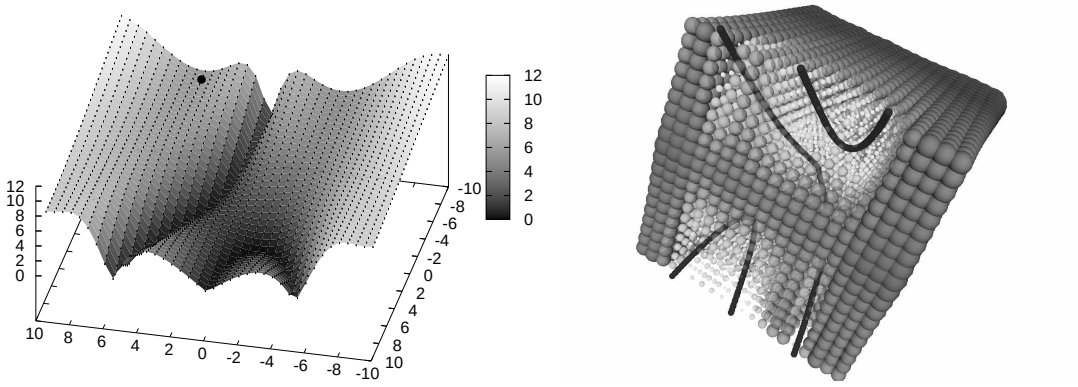

Fig. 6. (a) The surface plot of the square root of the potential of the cantilever problem with Q=2. The monotonic square root function was taken to visually enhance the view of the potential. The local minimum is shown with a black point. (b) Visualization of the potential in the 3 dimensional problem. The size of the balls is proportional to the square root of the potential. The solution is shown with a thick black line.

gradient runs do not deliver solutions but just alter the weighting of the primary hypercubes of the discretized GRS. We also showed that in high dimensions the computation need of the gradient algorithm is negligible.

On the other hand we showed that this dump implementation lack many feature that could make the algorithm run faster. We expect the most efficiency gain by developing a starting point choosing mechanism and a much better weighting of the primary hypercubes in parallel with an elaborate selection of slave types.

Acknowledgements

Part of this work was funded by the Péter Pázmány program (RET-06/2005) of the National Office for Research and Technology, and the authors would like to thank EU INFSO-RI-031688 program.

References

1. Domokos, G.: Global Description of Elastic Bars ZAMM **74** (1994) T289–T291
2. Domokos, G., Gáspár, Zs.: A global, direct algorithm for path-following and active static control of elastic bar structures, Mech. of Struct. and Mach. **23** (1995) 549–571
3. Gáspár, Zs., Domokos, G., Szeberényi, I.: A parallel algorithm for the global computation of elastic bar structures, Comp. Ass. Mech. Eng. Sci. **4** (1997) 55–68
4. Domokos, G., Szeberényi, I.: A hybrid parallel approach to nonlinear boundary value problems, Comp. Ass. Mech. Eng. Sci. **11**, (2004) 15–34
5. Várkonyi, P., Sipos, A. Á., Domokos, G.: Structural design on the internet: options to accelerate a computing algorithm (in Hungarian), Építés és Építészettudomány **34** (2006) 271–291