Benchmarking Solvers for The One Dimensional Cubic Nonlinear Klein Gordon Equation on a Single Core*

B.K. Muite[1,2] and S. Aseeri[3]

[1] Arvutiteaduse Instituut, Tartu Ülikool, Tartu 08544, Estonia
[2] Pole Pole Enterprises, Nairobi, Kenya benson.muite@ut.ee
[3] King Abdullah University of Science and Technology, Thuwal, Saudi Arabia
samar.aseeri@kaust.edu.sa

Abstract. Many parallel computing benchmarks specify a specific algorithm that should be used to solve a specific problem, with much effort focused on machine specific optimization of a reference implementation. Since the solution of linear systems of equations, is often the most time consuming part for many problems in high performance scientific computing, a number of recent benchmarks for high performance computers suggest the use of an iterative method for solving sparse linear systems of equations to rank computer performance. Particular methods used include multigrid and conjugate gradients, though other methods such as the fast Fourier transform are also applicable in some cases. The best choice of method may vary with the problem chosen and the hardware the implemented software solution is executed on. Furthermore in solving a scientific computing problem, the level of accuracy can also be important, with some numerical methods being efficient for low accuracy simulations, but others more efficient for high accuracy simulations. Some of these tradeoffs are examined in the numerical solution of the one dimensional Klein Gordon equation on a single core of an x86-64 CPU and a single core of a NEC SX-ACE vector processor.

Keywords: Benchmarks · Numerical Methods · Computer Architecture.

1 Introduction

One use case of high performance computing is for the rapid numerical simulation of partial differential equations. It can be a challenge to determine the best computer architecture to use for solving a particular type of partial differential equation. The choice of numerical method may also depend on the computer architecture being chosen. While there has been significant effort on numerical analysis of different computational methods, there is less work comparing the effectiveness of a particular parallel numerical methods. In this work, several numerical methods for solving the one dimensional Klein Gordon equation on a single core are reviewed and their effectiveness evaluated.

* BKM was partially supported by HPC Europa 3 (INFRAIA-2016-1-730897).

2 Motivation

The solution of linear systems of equations is a time consuming process in numerical simulation of partial differential equations, and has motivated a number of benchmarks[2, 4, 6, 11]. The solution of many partial differential equations requires a choice of discretization methods, in space and typically also in time, each of which presents numerous choices, each of which may have different relative performance on different computer architectures[1, 5, 9, 10, 15, 18–20, 24]. The Klein Gordon equation is chosen as a mini-application because it is relatively simple, can be used to evaluate different time stepping methods and spatial discretization methods, and is representative of seismic wave solvers, and weather codes, all of which use a large amount of high performance computing time[1, 10, 21, 28]. As a prelude to a three dimensional study of parallel solvers, a comparison of solvers for the one dimensional Klein Gordon equation on two architectures is presented showing the effects of discretization method on time to solution for a specified accuracy on a single core. Such a method can be informative in choosing where to run an application to get the most cost efficient results.

3 Time Stepping Algorithms

The equations will be discretized first in time, and then in space. The Klein Gordon equation has a conserved energy. Numerical schemes which either conserve energy may still have phase errors, but have typically been found to be useful for preserving qualitative properties of the phenomena being simulated over long time periods[22].

3.1 Semi-Implicit Second Order Leap Frog Method

The leap frog method is a common algorithm for wave equations, and can also be applied to the real cubic Klein Gordon equation

$$\frac{u_{n+1} - 2u_n + u_{n-1}}{\delta t^2} = (\Delta - 1)\,\frac{u_{n+1} + 2u_n + u_{n-1}}{4} + u_n^3 \tag{1}$$

The method is second order accurate and is semi implicit. It has the summation by parts formula:

$$\begin{aligned} &\left\| \frac{u_{n+1} - u_n}{\delta t} \right\|^2 + \left\| \nabla \frac{u_{n+1} + u_n}{2} \right\|^2 - \int u_n^3 u_{n+1} \\ &= \left\| \frac{u_n - u_{n-1}}{\delta t} \right\|^2 + \left\| \nabla \frac{u_n + u_{n-1}}{2} \right\|^2 - \int u_{n-1} u_n^3 \end{aligned} \tag{2}$$

which gives a discrete conserved energy. This scheme requires the solution of a constant coefficient linear system of elliptic equations at each timestep.

3.2 A Semi-Implicit Compact Fourth Order Leap Frog Method

To obtain a fourth order algorithm consider,

$$\frac{u_{n+1} - 2u_n + u_{n-1}}{\delta t^2} = (\Delta - 1)\, u_n + u_n^3 \tag{3}$$

for which time stepping error comes from approximating u_{tt}. As explained in among other places, [1], it is possible to approximate the leading order error term,

$$\begin{aligned}
&\frac{2(\delta t)^2}{4!} u_{n,tttt} \\
&\approx \frac{2(\delta t)^2}{4!} \left((\Delta - 1)\, u_n + u_n^3 \right)_{tt} \\
&\approx \frac{2(\delta t)^2}{4!} \left((\Delta - 1)\, u_{n,tt} + 3u_n^2 u_{n,tt} + 6u_n u_{n,t}^2 \right) \\
&= \frac{2}{4!} \left(\left(\Delta - 1 + 3u_n^2 \right) (u_{n+1} - 2u_n + u_{n-1}) + 6u_n (u_{n+1} - u_n)(u_n - u_{n-1}) \right)
\end{aligned} \tag{4}$$

One can subtract the leading error term to obtain a compact fourth order in time scheme

$$\begin{aligned}
&\frac{u_{n+1} - 2u_n + u_{n-1}}{\delta t^2} \\
&= (\Delta - 1)\, u_n + u_n^3 \\
&\quad + \frac{2}{4!} \left[\left(\Delta - 1 + 3u_n^2 \right) (u_{n+1} - 2u_n + u_{n-1}) \right. \\
&\quad \left. + 6u_n (u_{n+1} - u_n)(u_n - u_{n-1}) \right].
\end{aligned} \tag{5}$$

This scheme requires the solution of a non-constant coefficient linear elliptic system of equations at each timestep.

4 Spatial Discretizations

In all cases, uniform grids are used. In schemes that use the Fast Fourier transform, time stepping is done in Fourier space, and the nonlinear term is calculated in real space, no de-aliasing is done. Derivatives in spectral space are calculated by multiplying by the wave number. Descriptions of implementations of spectral methods can be found in [7, 8, 13, 26, 27]. For the compact time stepping scheme, fixed point iteration is used to calculate the nonlinear term.

High order finite difference discretizations for the one dimensional laplacian operator are described in [14] and given in table 1. A second iteration is not required to compute the nonlinear term, since the time discretization requires a variable coefficient elliptic equation to be solved at each timestep, for which the iterative conjugate gradient method is well suited, though multigrid methods can also be used.

Table 1. Stencils for high order finite difference schemes for the one dimensional Laplacian operator[14]

Order	Approximation for u_{xx}
2nd	$\frac{1}{(\delta x)^2}\left(u_{i-1}-2u_i+u_{i+1}\right)$
4th	$\frac{1}{(\delta x)^2}\left(-\frac{u_{i-2}}{12}+\frac{4u_{i-1}}{3}-\frac{5u_i}{2}+\frac{4u_{i+1}}{3}-\frac{u_{i+2}}{12}\right)$
6th	$\frac{1}{(\delta x)^2}\left(\frac{u_{i-3}}{90}-\frac{3u_{i-2}}{20}+\frac{3u_{i-1}}{2}-\frac{49u_i}{20}+\frac{3u_{i+1}}{2}-\frac{3u_{i+2}}{20}\frac{u_{i+3}}{90}\right)$
8th	$\frac{1}{(\delta x)^2}\left(-\frac{u_{i-4}}{560}+\frac{8u_{i-3}}{315}-\frac{u_{i-2}}{5}+\frac{8u_{i-1}}{5}-\frac{205u_i}{72}+\frac{8u_{i+1}}{5}-\frac{u_{i+2}}{5}+\frac{8u_{i+3}}{315}-\frac{u_{i+4}}{560}\right)$

5 Hardware Description

For the tests, two platforms have been used[4]. Hazelhen is a Cray XC 40 supercomputer with Intel Haswell E5-2680v3 chips with a nominal speed of 2.5 GHz and 30 Mb L3 Cache. Each node has 24 cores per node (2 chips with 12 cores each) with 136 Gb/s bandwidth and 960 Gflops per node peak performance. Kabuki is a NEC SX Ace supercomputer. Each node has 4 cores with 256 GB/s bandwidth and 256 Gflops peak performance. Each chip has a nominal speed of 1 GHz and each core has 1Mb vector Cache.

6 Numerical Experiments

In cases where iterations are required, both for the conjugate gradient algorithm and for fixed point iterations using the FFT, the solution at the previous time step is used as an initial starting guess. For these programs, memory bandwidth is a limiting factor and to minimize the number of memory accesses, the coefficients are programmed using a matrix free approach[24]. The example programs are written in Fortran and can be found at [23]. Accuracy is evaluated by comparing to the exact travelling wave solution

$$u=\sqrt{2}\text{sech}\left(\frac{x-ct}{\sqrt{1-c^2}}\right) \tag{6}$$

for $c=0.5$, $t\in[0,5]$ and $x\in[-9\pi,9\pi)$ with periodic boundary conditions.

The finite difference programs were compiled and run, without much further tuning other than the choice of compilation flags. The programs which use the Fourier transform differ in the choice of Fast Fourier transform library, on Hazelhen, FFTW 3[12] was used, and on Kabuki, MathKeisan's FFT[25] was used. Two sets of results are shown, accuracy against computation time for both Hazelhen (Fig. 1) and Kabuki (Fig. 2). For each of the implementations that was tested a final plot comparing the efficiency frontiers for Kabuki and Hazelhen is shown in Fig. 3.

[4] A third platform is currently being tested and results will be reported if accepted for presentation.

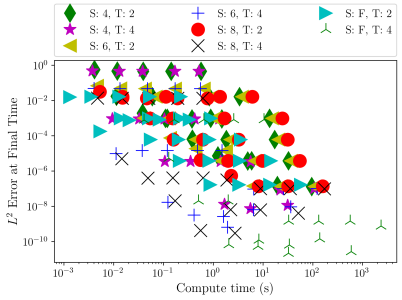

Fig. 1. Time to solution against accuracy for solution of the one dimensional Klein Gordon equation for a single Intel Haswell E5-2680v3 core. S gives the order of the spatial discretization for finite differences, while S: F indicates a Fourier spatial discretization. T gives the order of the temporal discretization.

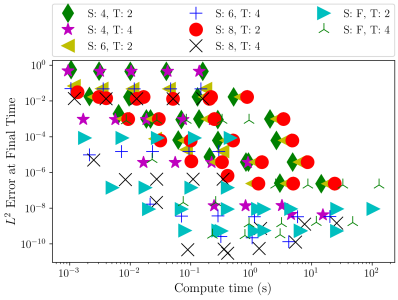

Fig. 2. Time to solution against accuracy for solution of the one dimensional Klein Gordon equation for a single NEC SX-ACE core. S gives the order of the spatial discretization for finite differences, while S: F indicates a Fourier spatial discretization. T gives the order of the temporal discretization.

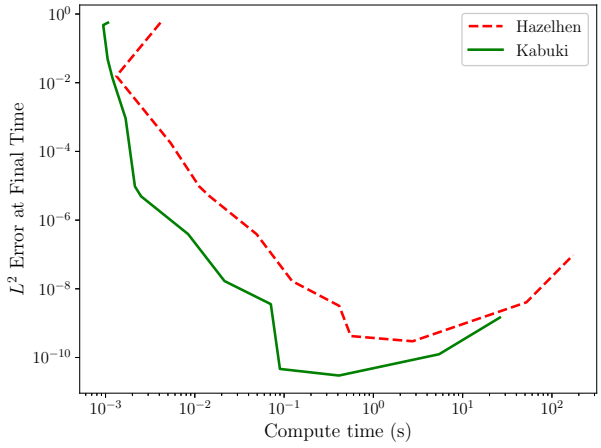

Fig. 3. A figure comparing efficiency frontiers for single Intel Haswell E5-2680v3 on Hazelhen and a NEC SX-ACE core on Kabuki for time to solution against accuracy for approximation of the one dimensional Klein Gordon equation.

Finite difference methods have a low computational intensity, thus high order methods which reuse data are able to produce much more accurate results without significant increases in computational time. The Fast Fourier transform requires significant data movement, thus the eighth order finite difference method can sometimes be more efficient in getting high accuracy results within a specified time – though only fourth order time stepping schemes are tested here, results may differ for higher order time stepping schemes.

The NEC SX-ACE allows for all the bandwidth to be utilized by a single core, which is not possible on the Intel Haswell E5-2680v3. This largely explains the greater single core performance. Other studies have found that on a single node, the performance advantage of the NEC SX-ACE is much reduced[3]. Performance evaluation for single nodes, and multiple nodes, in particular for three dimensional implementations would be good extensions of this work. In addition, comparisons of energy consumption to solution would also be helpful as this becomes an important consideration for large scale simulation.

7 Conclusions

High order methods can take advantage of multiple floating point units and not require much more computation time and give smaller error than low order methods, and so should be encouraged in the numerical approximation of partial differential equations, this hold also for spectral element methods[18]. For benchmarks based on mini-applications, compute resources to solution at specified accuracy may be a good metric to use in evaluating performance rather than speed of performing a fixed set of operations. This would allow for architecture specific flexibility and can minimize cost to solution, though may require some programming effort.

8 Acknowledgements

We thank Holger Berger and José Gracia for helpful conversations. We also thank Höchstleistungsrechenzentrum Stuttgart (HLRS), the KAUST Supercomputing Laboratory and the University of Tartu High Performance Computing Center for access to supercomputing resources used in development and testing.

References

1. Abdulkadir, Y.A.: Comparison of Finite Difference Schemes for the Wave Equation Based on Dispersion, Journal of Applied Mathematics and Physics, **3**, 1544–1562 (2015) https://doi.org/10.4236/jamp.2015.311179
2. Adams, M.F., Brown, J., Shalf, J., Van Straalen, B., Strohmaier, E., Williams, S.: HPGMG 1.0: A Benchmark for Ranking High Performance Computing Systems, Lawrence Berkely National Laboratory Preprint, (2014) https://escholarship.org/uc/item/00r9w79m Last Accessed 16 July 2019

3. Afanasyev, I.V., Antonov, A.S., Nikitenko, D.A., Voevodin, V.V., Voevodin, V.V, Komatsu, K., Watanabe, O., Musa, A., Kobayashi, H.: Developing efficient implementations of Bellman-Ford andForward-Backward Graph Algorithms for NEC SX-ACE, Supercomputing Frontiers and Innovations, **5**(3), 65–69 (2018) https://doi.org/10.14529/jsfi180311
4. Aseeri, S., Batrašev, O., Icardi, M., Leu, B., Liu, A., Muite, B.K, Müller, E., Palen, B., Quell, M., Servat, H., Sheth, P., Speck, R., Van Moer, M., Vienne, J.: Solving the Klein-Gordon Equation using Fourier Spectral Methods: A Benchmark Test for Computer Performance, In: HPC' 15 Proceedings of the Symposium on High Performance Computing, 182-191, Society for Computer Simulation International (2015)
5. Auzinger, W., Březinová, I. Hofstätter, H., Koch, O., Quell, M.: Practical Splitting Methods for the Adaptive Integration of Nonlinear Evolution Equations. Part II: Comparisons of Local Error Estimation and Step-Selection Strategies for Nonlinear Schrödinger and Wave Equations, Computer Physics Communications **234**, 55–71 (2018) https://doi.org/10.1016/j.cpc.2018.08.003
6. Bailey, D.H., Barszcz, E., Barton, J.T., Browning, D.S., Carter, R.L., Dagum, L., Fatoohi, R.A., Frederickson, P.O., Lasinski, T.A., Schreiber, R.S., Simon, H.D., Venkatakrishnan, V., Weeratunga, S.K.: The NAS parallel benchmarks, The International Journal of High Performance Computing Applications **5**(3), 63–73, (1991) https://doi.org/10.1177/109434209100500306
7. Balakrishnan, S., Bargash, A.H., Chen, G., Cloutier, B., Li, N., Malicke, D., Muite, B.K., Quell, M., Rigge, P., San Roman Alerigi, D., Solimani, M., Souza, A., Thiban, A.S., West, J., van Moer, M.: Parallel Spectral Numerical Methods. http://en.wikibooks.org/wiki/Parallel_Spectral_Numerical_Methods, Last accessed 24 June 2019
8. Canuto, C., Hussaini, M.Y., Quarteroni, A., Zang, T.A.: Spectral methods fundamentals in single domains, Springer (2010) https://doi.org/10.1007/978-3-540-30726-6
9. Cloutier, B., Muite, B.K., Rigge, P.: A Comparison of CPU and GPU performance for Fourier Pseudospectral Simulations of the Navier-Stokes, Cubic Nonlinear Schrödinger and Sine Gordon Equations (2012)
10. Deconinck, W., Hamrud, M., Kühnlein, C., Mozdzynski, G., Smolarkiewicz, P.K., Szmelter, J., Wedi, N.P.: Accelerating Extreme-Scale Numerical Weather Prediction, In: Wyrzykowski R., Deelman E., Dongarra J., Karczewski K., Kitowski J., Wiatr K. (eds) Parallel Processing and Applied Mathematics. Lecture Notes in Computer Science, vol 9574. Springer, (2016) https://doi.org/10.1007/978-3-319-32152-3_54
11. Dongarra ,J., Heroux, M.A., Luszcek, P.: A New Metric for Ranking High-Performance Computing Systems , The International Journal of High Performance Computing Applications **30**(1), 3-10 (2016) https://doi.org/10.1177/1094342015593158
12. Frigo, M., Johnson, S.G.: The design and implementation of FFTW, Proceedings of the IEEE **93**(2), 216-231, (2005) https://doi.org/10.1109/JPROC.2004.840301
13. Fornberg, B.: A Practical Guide to Pseudospectral Methods, Cambridge University Press (1996) https://doi.org/10.1017/CBO9780511626357
14. Fornberg, B.: Generation of Finite Difference Formulas on Arbitrarily Spaced Grids, Mathematics of Computation **51**, 699–706, (1988) https://doi.org/10.1090/S0025-5718-1988-0935077-0
15. Gholami, A., Malhotra, D., Sundar, H., Biros, G.: FFT, FMM, or Multigrid? A Comparative Study of State-Of-the-Art Poisson Solvers for Uniform and Nonuni-

form Grids in the Unit Cube, SIAM J. Sci. Comput., **38**(3), C280–C306 (2016) https://doi.org/10.1137/15M1010798
16. Höchstleistungsrechenzentrum Stuttgart (HLRS): Hazelhen. https://www.hlrs.de/systems/cray-xc40-hazel-hen/, Last accessed 15 July 2019
17. Höchstleistungsrechenzentrum Stuttgart (HLRS): Kabuki. https://kb.hlrs.de/platforms/index.php/NEC_SX-ACE, Last accessed 15 July 2019
18. Hutchinson, M., Heinecke, A., Pabst, H., Henry, G., Parsani, M., Keyes, D.: Efficiency of High Order Spectral Element Methods on Petascale Architectures, In: Kunkel J., Balaji P., Dongarra J. (eds) High Performance Computing. ISC High Performance 2016, LNCS, vol. 9697, Springer (2016) https://doi.org/10.1007/978-3-319-41321-1_23
19. Ibeid, H., Olson, L., Gropp, W.: FFT, FMM, and Multigrid on the Road to Exascale: Performance Challenges and Opportunities, arXiv:1810.11883v1 (2018)
20. Ketcheson, D.I., Mortensen, M., Parsani, M., Schilling N.: More efficient time integration for Fourier pseudo–spectral DNS of incompressible turbulence, arXiv:1810.10197v1
21. Komatitsch, D., Vilotte, J.-P., Tromp, J., Ampuero, J.-P.; Bai, K.; Basini, P.; Blitz, C.; Bozdag, E.; Casarotti, E.; Charles, J.; Chen, M.; Galvez, P.; Goddeke, D.; Hjorleifsdottir, V.; Labarta, J.; Le Goff, N.; Le Loher, P.; Lefebvre, M.; Liu, Q.; Luo, Y.; Maggi, A.; Magnoni, F.; Martin, R.; Matzen, R.; McRitchie, D.; Meschede, M.; Messmer, P.; Michea, D.; Nadh Somala, S.; Nissen-Meyer, T.; Peter, D.; Rietmann, M.; de Andrade, E.S. ; Savage, B.; Schuberth, B.; Sieminski, A.; Strand, L.; Tape, C.; Xie, Z.; Zhu, H. (9999), SPECFEM3D Cartesian [software], https://doi.org/GITHASH8, https://geodynamics.org/cig/software/specfem3d/ Last Accessed 16 July 2019
22. Leimkuhler, B., Reich, S.: Simulating Hamiltonian Dynamics, Cambridge University Press (2009) https://doi.org/10.1017/CBO9780511614118
23. Muite, B.K.: https://github.com/bkmgit/KleinGordon1D [software] Last Accessed 16 July 2019
24. Müller, E.H., Scheichl, R., Vainikko, E.: Petascale solvers for anisotropic PDEs in atmospheric modelling on GPU clusters. Parallel Computing 50, 53-69, (2015) https://doi.org/10.1016/j.parco.2015.10.007
25. NEC: http://mathkeisan.com/ [software] Last Accessed 16 July 2019
26. Shen, J., Tang, T., Wang, L.-L.: Spectral Methods: Algorithms, Analysis and Applications, Springer (2011) https://doi.org/10.1007/978-3-540-71041-7
27. Trefethen, L.: Spectral Methods in MATLAB, SIAM (2000) https://doi.org/10.1137/1.9780898719598
28. Yang, C., Xue, W., Fu, H., You, H., Wang, X., Ao, Y., Liu, F., Gan, L., Xu, P., Wang, L., Yang, G., Zheng, W.: 10M–core scalable fully–implicit solver for nonhydrostatic atmospheric dynamics, SC'16: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis, 57-68 (2016), https://doi.org/10.1109/SC.2016.5