The Journal of Open Source Software

SARAS: A general-purpose PDE solver for fluid dynamics

Roshan Samuel[1], Shashwat Bhattacharya[1], Ali Asad[2], Soumyadeep Chatterjee[2], Mahendra K. Verma[2], Ravi Samtaney[3], and Syed Fahad Anwer[4]

1 Department of Mechanical Engineering, Indian Institute of Technology - Kanpur, Uttar Pradesh - 208016, India **2** Department of Physics, Indian Institute of Technology - Kanpur, Uttar Pradesh - 208016, India **3** Mechanical Engineering, Physical Science and Engineering Division, King Abdullah University of Science and Technology, Thuwal - 23955, Saudi Arabia **4** Department of Mechanical Engineering, ZHCET, Aligarh Muslim University, Uttar Pradesh - 202002, India

DOI: 10.21105/joss.02095

Software

- Review
- Repository
- Archive

Editor: Kyle Niemeyer

Reviewers:

- @dlagrava
- @olgadoronina
- @nickwimer

Submitted: 06 February 2020
Published: 16 August 2021

Summary

The laws that govern natural systems can often be modelled mathematically using partial differential equations (PDEs). Usually the resultant PDEs are not solvable analytically, leaving numerical solutions as the only recourse to gain useful insights into such systems. As a result, it is important to efficiently calculate numerical solutions of PDEs to further our understanding of such systems. In this paper we briefly describe the design and validation of SARAS, a general-purpose PDE solver based on finite difference method (Anderson, 1995; Ferziger & Peric, 2001).

Statement of Need

There are a number of open-source solvers for Computational Fluid Dynamics. Some well known solvers include OpenFOAM (Weller et al., 1998), Pencil Code (Brandenburg & Dobler, 2002), Gerris (Popinet, 2003), SU2 (Palacios et al., 2013), etc. to name a few. However, many of them, for instance, SU2, OpenFOAM, etc. use finite-volume method with structured or unstructured meshes. On the other hand, the use of finite-difference method restricts the solver to simpler domains, but offers comparatively higher accuracy with lesser computational effort. In this regard, the Pencil Code also uses finite-difference method, but is used to solve compressible flows, unlike SARAS which is designed for incompressible flows. Moreover, SARAS is written in C++ with an object oriented structure like OpenFOAM.

The design of SARAS is inspired by TARANG (Verma et al., 2013), a pseudo-spectral solver developed in our lab. TARANG has been shown to scale up to 196608 cores (Chatterjee et al., 2018), and SARAS has been designed with the goal of achieving similar scaling performance. We conducted a preliminary scaling analysis of SARAS using up to 1024 cores for the lid-driven cavity (LDC) problem on a 512^3 grid, and observed strong scaling (Verma et al., 2020). However, we need to implement further optimizations to the code before scaling to larger grids and more cores. In solving the Rayleigh Benard Convection problem, we also observed that the computational efficiency of SARAS is at par with that of OpenFOAM. Recently, we also performed a comparative study between spectral and finite difference codes in the context of exascale computing (Verma et al., 2020), using SARAS.

In SARAS, the underlying mathematical constructs like vector and scalar fields are defined as classes. Moreover, vector calculus operations associated with such fields, like gradient and divergence, are defined using these classes. This design makes the code intuitive, allowing

users to quickly cast PDEs into readable codes. The initial conditions, boundary conditions, and source/forcing terms are implemented using `initial`, `boundary` and `force` classes. These classes are readily extensible so that users can add custom initial conditions, source terms, and so on.

SARAS includes solvers for hydrodynamic flows, namely the incompressible Navier-Stokes initial value problem (IVP), as well as for scalar convection, like Rayleigh Benard Convection. Presently, we use semi-implicit Crank-Nicholson (Crank & Nicolson, 1947) method for time-advancing the IVP. The solver uses Marker and Cell method (MAC) (Harlow & Welch, 1965) for discretizing the velocity and pressure fields.

Mathematics

The Navier-Stokes equations, which govern the dynamics of fluid flow, can be written as

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = -\nabla p + \mathbf{f} + \nu \nabla^2 \mathbf{u},$$

where $\mathbf{u}$ is the velocity field, p is the pressure field, $\mathbf{f}$ is the forcing term, and ν is the kinematic viscosity of the fluid. The fluid is assumed to be incompressible. Hence $\nabla \cdot \mathbf{u} = 0$, and density is constant (chosen to be unity).

If the velocity and pressure field at time $t = t_n$ are denoted as $\mathbf{u}_n$ and p_n respectively, then the corresponding fields at the next time-step, $t = t_{n+1}$, namely $\mathbf{u}_{n+1}$ and p_{n+1}, can be calculated as described below (Anderson, 1995; Ferziger & Peric, 2001; Patankar & Spalding, 1972). We compute an intermediate velocity field using the known values, $\mathbf{u}_n$ and p_n, as

$$\mathbf{u}^* = \mathbf{u}_n + \Delta t \left[\nu \nabla^2 \left(\frac{\mathbf{u}_n + \mathbf{u}^*}{2} \right) - \mathbf{u}_n . \nabla \mathbf{u}_n - \nabla p_n \right].$$

The forcing term has been neglected here for simplicity. Note that the diffusion term (also called the viscous term) is handled semi-implicitly, with equal contribution from $\mathbf{u}_n$ and $\mathbf{u}^*$, as given below:

$$\mathbf{u}^* - \Delta t \left[\frac{\nu \nabla^2 \mathbf{u}^*}{2} \right] = \mathbf{u}_n + \Delta t \left[\frac{\nu \nabla^2 \mathbf{u}_n}{2} - \mathbf{u}_n . \nabla \mathbf{u}_n - \nabla p_n \right].$$

The above equation has to be solved iteratively, and this is achieved through Jacobi iterations. The intermediate velocity field, $\mathbf{u}^*$, does not satisfy the continuity equation, and requires appropriate correction. This correction is obtained from the pressure correction term, which is calculated using the pressure Poisson equation,

$$\nabla^2 p^* = \frac{\nabla . \mathbf{u}^*}{\Delta t}.$$

SARAS uses a Geometric Multigrid library to solve the above equation (Briggs et al., 2000; Wesseling, 2004). Presently the library employs the Full Multigrid (FMG) V-Cycle to solve the Poisson equation. Other methods like F-Cycle and W-Cycle are planned updates to the library in future.

Finally, using the above pressure correction, the velocity and pressure fields corresponding to the next time-step are obtained as

$$p_{n+1} = p_n + p^*,$$

$$\mathbf{u}_{n+1} = \mathbf{u}^* - \Delta t (\nabla p^*).$$

The numerical implementation of the above procedure will be discussed in the next section.

Numerical Method and Implementation

SARAS uses finite-difference method (Ferziger & Peric, 2001) to calculate derivatives of the field variables. Presently the solver uses second-order central difference stencils to compute first and second derivatives. We use a highly optimized and fast array manipulation library, Blitz++ (Veldhuizen, 1998), for all array operations. Blitz++ also offers finite-difference stencils on uniformly spaced grids. SARAS augments this feature with grid-transformation terms (Anderson, 1995) to perform simulations on grids with non-uniform spacing.

SARAS offers a set of extensible boundary and initial conditions, as well as source terms. Presently the solver can switch between periodic, Neumann, and Dirichlet boundary conditions. There also exists a mixed boundary condition class that can simulate many practical applications, such as a conducting heating plate on an adiabatic wall (Teimurazov et al., 2017). Currently the solver is restricted to Cartesian grids, but it will be extended to cylindrical, toroidal, and spherical grids in the future. The solver also supports adaptive time-stepping, where the Courant-Friedrichs-Lewy (CFL) condition (Courant et al., 1928) is used to dynamically compute the appropriate time-step.

Results

We validate our code using two very well-known test problems: lid-driven cavity and decaying turbulence. We simulate these problems using SARAS and compare the results with standard and validated solutions.

Problem 1

We solve the two-dimensional lid-driven cavity (LDC) problem using SARAS, and compare the results with those of Ghia et al. (1982). LDC is an important fluid system and it serves as a benchmark for testing numerical methods. This system consists of a square cavity of dimension 1×1 with no-slip boundary conditions on all the four walls. However, the top wall moves laterally to the right with a constant velocity of $U = 1.0$ that serves as the reference velocity for non-dimensionalization of the problem. The length of the side of the cavity, $L = 1.0$, is the reference length. The Reynolds number of the flow is approximately $\mathrm{Re} \approx 1000$.

At the start of the simulation, the fluid, which is at rest, is driven impulsively by the top lid. This results in the formation of a vortex at the upper-right corner of the cavity, and this vortex rapidly grows in size and occupies the entire region of the cavity. For the simulation we employ a 129×129 grid, and carry it out till $t = 30$. The solver computes this solution using 4 MPI processes in approximately 12 minutes on an Intel workstation. The output by SARAS at the final time is used for comparison with the results of Ghia et al. (1982)

In the website we supply the Bash and Python scripts to compile and run this test case. After automatic execution of SARAS, a Python script reads the output from SARAS and compares the horizontal and vertical velocity profiles across the geometric center of the square cavity. The corresponding results from Ghia et al. (1982) are also available with the installation. We compare the two results and exhibit them in Figure 1. We observe that the profiles computed by SARAS match very well with those by Ghia et al. (1982), thus providing a strong validation for our code. This quick validation of the solver, which can be done as a part of its installation, is one of the strengths of this package.

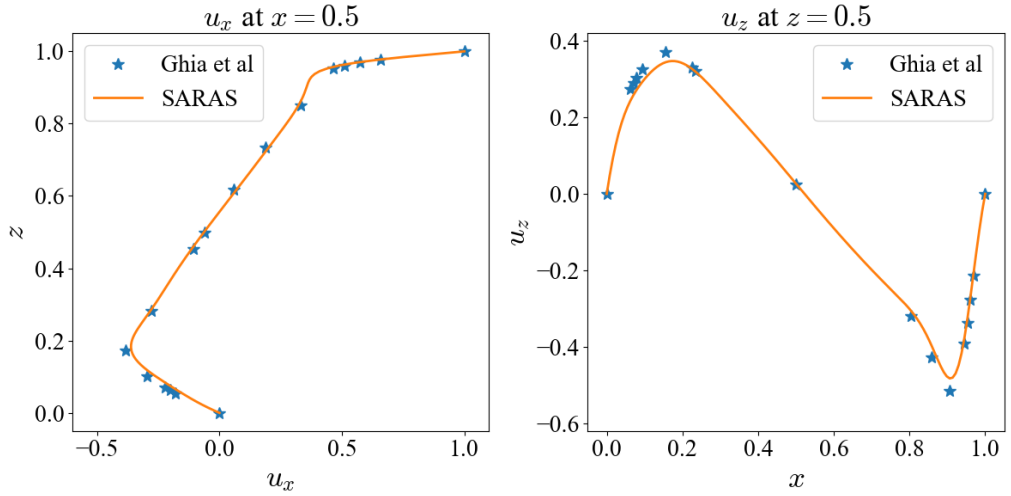

Figure 1: Velocity profiles from the simulation of lid-driven cavity on a 129^2 grid with SARAS (orange lines), plotted along with the data from Ghia et al. (1982) (blue stars): (a) The vertical profile of the x-component of velocity, u_x, along the line across the geometric center of the cavity (b) The horizontal profile of the z-component of velocity, u_z, along the line across the geometric center of the cavity.

Problem 2

We simulate decaying turbulence using SARAS with Taylor-Green vortex (Taylor & Green, 1937) as the initial condition. That is,

$$\mathbf{u}(x, y, z, t = 0) = u_0 \begin{bmatrix} \sin(2\pi k_0 x)\cos(2\pi k_0 y)\cos(2\pi k_0 z) \\ -\cos(2\pi k_0 x)\sin(2\pi k_0 y)\cos(2\pi k_0 z) \\ 0 \end{bmatrix},$$

where $u_0 = 1$ and $k_0 = 1$. We perform our simulation in a periodic box of size $1 \times 1 \times 1$ ($L = 1$) with a grid resolution of 257^3 up to $t = 3.0$. Here, we nondimensionalize time using L/u_0. The initial Reynolds number of the flow is $\mathrm{Re} = 1000$. We choose a constant $dt = 0.001$ for time-integration. Besides, we use a uniform mesh along all the three directions.

To validate the accuracy of the finite difference scheme of SARAS, we compare the results of SARAS with those of a pseudo-spectral code TARANG, which has been benchmarked and scaled up to 196608 cores of Cray XC40, Shaheen II of KAUST (Chatterjee et al., 2018; Verma et al., 2013). Note that pseudo-spectral method yields very accurate derivatives (Canuto et al., 1988). We perform our spectral simulation using the same initial condition and grid resolution as above, up to $t = 3.0$. The comparison of the results from the two codes is discussed below. Comparison of SARAS results with those of TARANG provides another validation of SARAS.

The results of TARANG and SARAS are quite similar, as exhibited in Figures 2 and 3. In Figure 2(a) we exhibit the plots of the total energy ($\int d\mathbf{r} u^2/2$) vs. time for both the runs. As is evident from the plots, both the codes yield very similar evolution profiles for the total energy. In addition, we also compute the energy spectra for the results at $t = 1$. As shown in Figure 2(b), both the energy spectra are very similar. Interestingly, in both the plots, the energy spectra in the inertial range are close to the $k^{-5/3}$ power law (Kolmogorov, 1941; Verma, 2019).

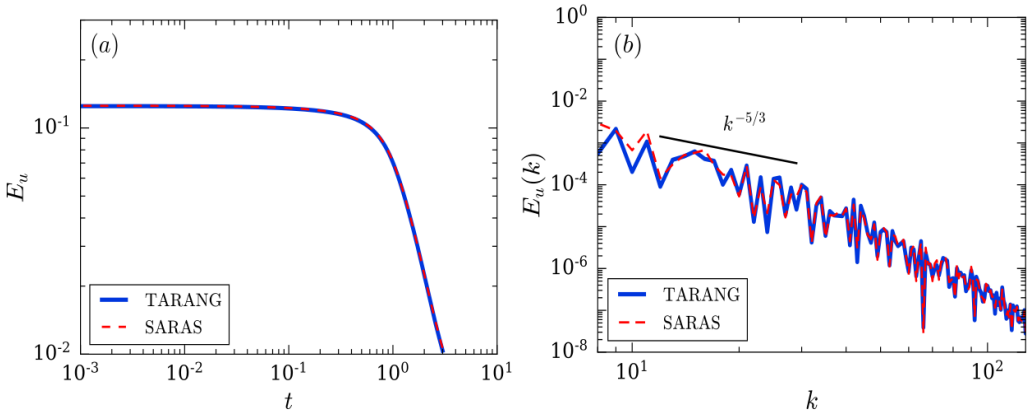

Figure 2: For the simulation of decaying turbulence on a 257^3 grid with `TARANG` (thick blue lines) and `SARAS` (red-dashed lines): (a) plot of the total energy $E_u = \int d\mathbf{r} u^2/2$ vs t, (b) plot of $E_u(k)$ vs k at $t = 1$.

Figure 3 exhibits the magnitude of velocity, $|\mathbf{u}|$, on the horizontal mid-plane ($z = 1/2$) at $t = 1$ and $t = 3$. Clearly, the results of `TARANG` and `SARAS` are very similar.

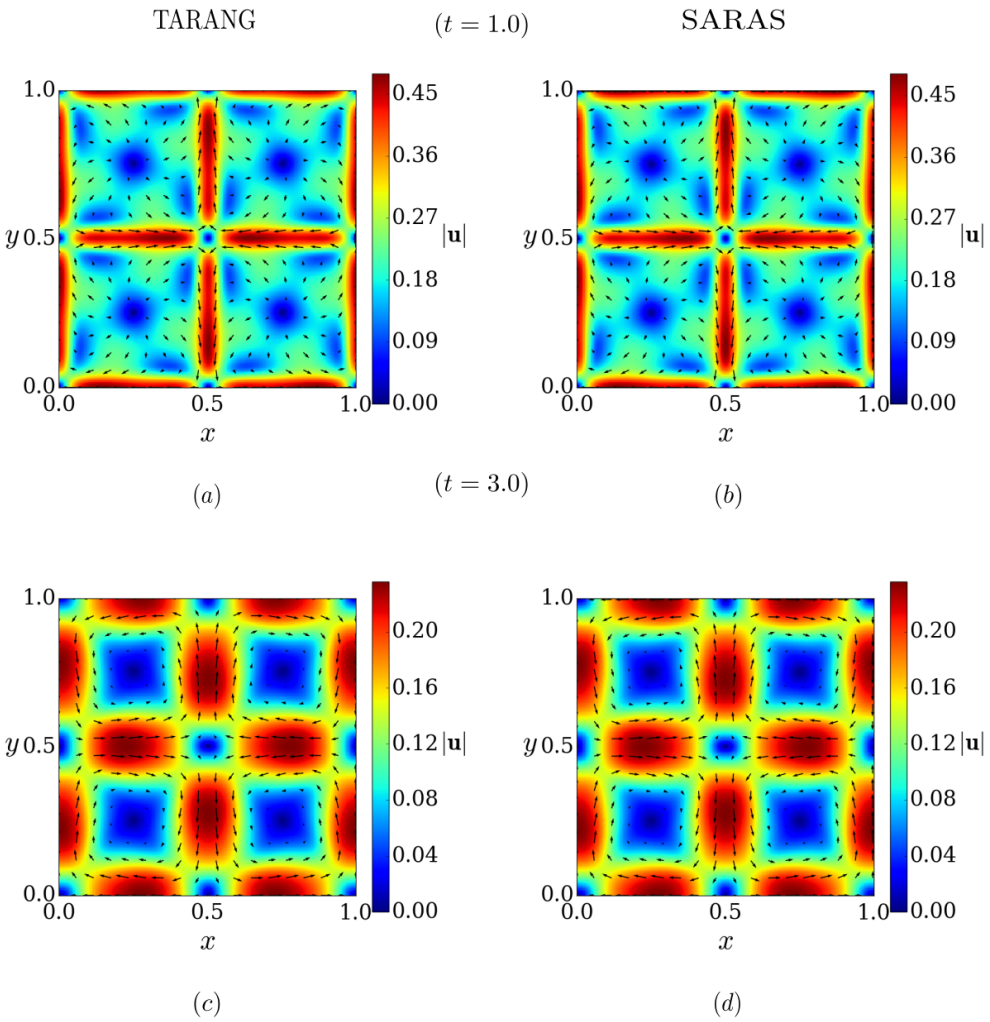

Figure 3: For the simulation of decaying turbulence on a 257^3 grid, vector plots of the velocity field, and the density plots of the magnitude of velocity ($|\mathbf{u}|$) computed at the horizontal mid-plane: for the data from `TARANG`(a, c), and `SARAS`(b, d) at $t = 1$ (top row) and $t = 3$ (bottom row).

Conclusions

This paper provides a brief description and validations tests of a new parallel finite-difference code `SARAS`. `SARAS` has been designed as a general PDE solver using the object-oriented features of C++. We validate `SARAS` using two test cases: lid-driven cavity and decaying turbulence.

Acknowledgements

We gratefully acknowledge the contributions from Gaurav Gautham, Saurav Bhattacharjee, Rishabh Sahu and Mohammad Anas during the development of SARAS. We also thank Kyle Niemeyer and the reviewers at JOSS, whose useful suggestions greatly improved the solver, with true open-source ethos. Part of our computations were performed on the Cray XC40 (Shaheen II) of KAUST supercomputing laboratory, Saudi Arabia, through Projects k1052 and k1416.

References

Anderson, J. D. (1995). *Computational Fluid Dynamics: The Basics With Applications*. McGraw-Hill. ISBN: 9781259025969

Brandenburg, A., & Dobler, W. (2002). Hydromagnetic turbulence in computer simulations. *Computer Physics Communications*, *147*(1), 471–475. https://doi.org/10.1016/S0010-4655(02)00334-X

Briggs, W. L., Henson, V. E., & McCormick, S. F. (2000). *A multigrid tutorial: Second edition*. Society for Industrial; Applied Mathematics. https://doi.org/10.1137/1.9780898719505

Canuto, C., Hussaini, M. Y., Quarteroni, A., & Zang, T. A. (1988). *Spectral Methods in Fluid Dynamics*. Springer-Verlag. https://doi.org/10.1007/978-3-642-84108-8

Chatterjee, A. G., Verma, M. K., Kumar, A., Samtaney, R., Hadri, B., & Khurram, R. (2018). Scaling of a Fast Fourier Transform and a pseudo-spectral fluid solver up to 196608 cores. *J. Parallel Distrib. Comput.*, *113*, 77–91. https://doi.org/10.1016/j.jpdc.2017.10.014

Courant, R., Friedrichs, K., & Lewy, H. (1928). Über die partiellen differenzengleichungen der mathematischen physik. *Mathematische Annalen*, *100*(1), 32–74. https://doi.org/10.1007/BF01448839

Crank, J., & Nicolson, P. (1947). A practical method for numerical evaluation of solutions of partial differential equations of the heat-conduction type. *Proc. Camb. Phil. Soc.*, *43*(1), 50–67. https://doi.org/10.1017/S0305004100023197

Ferziger, J. H., & Peric, M. (2001). *Computational Methods for Fluid Dynamics* (3rd ed.). Springer-Verlag. https://doi.org/10.1007/978-3-642-56026-2

Ghia, U., Ghia, K. N., & Shin, C. T. (1982). High-Re solutions for incompressible flow using the navier-stokes equations and a multigrid method. *J. Comput. Phys.*, *48*(3), 387–411. https://doi.org/10.1016/0021-9991(82)90058-4

Harlow, F. H., & Welch, J. E. (1965). Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface. *Phys. Fluids*, *8*(12), 2182–2189. https://doi.org/10.1063/1.1761178

Kolmogorov, A. N. (1941). Dissipation of Energy in Locally Isotropic Turbulence. *Dokl Acad Nauk SSSR*, *32*, 16–18. https://doi.org/10.1098/rspa.1991.0076

Palacios, F., Alonso, J. J., Colonno, M. R., Aranake, A. C., Campos, A., Copeland, S. R., Economon, T. D., Lonkar, A. K., Lukaczyk, T. W., & Taylor, T. W. R. (2013). Stanford University Unstructured (SU2): An open-source integrated computational environment for multi-physics simulation and design. In *51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*. https://doi.org/10.2514/6.2013-287

Patankar, S. V., & Spalding, D. B. (1972). A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows. *International Journal of Heat and Mass Transfer*, *15*(10), 1787–1806. https://doi.org/10.1016/0017-9310(72)90054-3

Popinet, S. (2003). Gerris: A tree-based adaptive solver for the incompressible Euler equations in complex geometries. *Journal of Computational Physics*, *190*(2), 572–600. https://doi.org/10.1016/S0021-9991(03)00298-5

Taylor, G. I., & Green, A. E. (1937). Mechanism of the production of small eddies from large ones. *Proceedings of the Royal Society of London. Series A - Mathematical and Physical Sciences*, *158*(895), 499–521. https://doi.org/10.1098/rspa.1937.0036

Teimurazov, A., Sukhanovskii, A., Evgrafova, A., & Stepanov, R. (2017). Helicity sources in a rotating convection. *Journal of Physics: Conference Series*, *899*, 022017. https://doi.org/10.1088/1742-6596/899/2/022017

Veldhuizen, T. L. (1998). Arrays in blitz++. In D. Caromel, R. R. Oldehoeft, & M. Tholburn (Eds.), *Computing in object-oriented parallel environments* (pp. 223–230). Springer Berlin Heidelberg. https://doi.org/10.1007/3-540-49372-7_24

Verma, M. K. (2019). *Energy transfers in fluid flows: Multiscale and spectral perspectives*. Cambridge University Press. https://doi.org/10.1017/9781316810019

Verma, M. K., Chatterjee, A. G., Yadav, R. K., Paul, S., Chandra, M., & Samtaney, R. (2013). Benchmarking and scaling studies of pseudospectral code Tarang for turbulence simulations. *Pramana-J. Phys.*, *81*(4), 617–629. https://doi.org/10.1007/s12043-013-0594-4

Verma, M. K., Samuel, R., Chatterjee, S., Bhattacharya, S., & Asad, A. (2020). Challenges in fluid flow simulations using exascale computing. *SN Computer Science*, *1*(3), 178. https://doi.org/10.1007/s42979-020-00184-1

Weller, H. G., Tabor, G., Jasak, H., & Fureby, C. (1998). A tensorial approach to computational continuum mechanics using object-oriented techniques. *Computers in Physics*, *12*(6), 620–631. https://doi.org/10.1063/1.168744

Wesseling, P. (2004). *An introduction to multigrid methods*. R.T. Edwards. https://doi.org/10.1063/1.1761178