

nsCouette – A high-performance code for direct numerical simulations of turbulent Taylor–Couette flow

Jose M. López^{a,**}, Daniel Feldmann^{b,*}, Markus Rampp^c,
Alberto Vela-Martín^d, Liang Shi^e, Marc Avila^b

^a*Institute of Science and Technology Austria,
Am Campus 1, 3400 Klosterneuburg, Austria*

^b*University of Bremen, Center of Applied Space Technology and Microgravity (ZARM),
Am Fallturm 2, 28359 Bremen, Germany*

^c*Max Planck Computing and Data Facility (MPCDF),
Gießenbachstraße 2, 85748 Garching, Germany*

^d*School of Aeronautics, Universidad Politécnica de Madrid,
Plaza del Cardenal Cisneros 3, 28040 Madrid, Spain*

^e*Max Planck Institute for Dynamics and Self-Organization (MPIDS),
Bunsenstraße 10, 37073 Göttingen, Germany*

Abstract

We present **nsCouette**, a highly scalable software tool to solve the Navier–Stokes equations for incompressible fluid flow between differentially heated and independently rotating, concentric cylinders. It is based on a pseudospectral spatial discretization and dynamic time-stepping. It is implemented in modern **Fortran** with a hybrid **MPI-OpenMP** parallelization scheme and thus designed to compute turbulent flows at high Reynolds and Rayleigh numbers. An additional GPU implementation (**C-CUDA**) for intermediate problem sizes and a basic version for turbulent pipe flow (**nsPipe**) are also provided.

Keywords: Wall-bounded turbulence, Rotating shear-flow, Thermal convection, Direct numerical simulation (DNS), Hybrid parallelization, GPU

1. Motivation and significance

Flows in engineering and nature are often characterized by large Reynolds (Re) or Rayleigh (Ra) numbers. Examples are the flow of gas in astrophysical disks, atmospheric flows and the cooling of rotating machines. In most cases, it is impossible to resolve all scales of the turbulent flow in a direct numerical

*Corresponding author at ZARM: daniel.feldmann@zarm.uni-bremen.de

**Corresponding author at IST Austria: jlopez@ist.ac.at

simulation (DNS). However, DNS provide reliable data to allow extrapolation to the large Re limit and to enable the development of adequate subgrid-scale models. Taylor–Couette (TC) flow – the flow between two independently rotating concentric cylinders – stands out as a testbed for these purposes [1, 2]. It allows exploring a variety of physical mechanisms, including buoyancy, shear, rotation and boundary layers in the vicinity of curved walls. Our DNS code `nsCouette` integrates the incompressible Navier–Stokes equations for TC flow forward in time using cylindrical coordinates and primitive variables. Optionally, the cylinder walls can be differentially heated, in which case an additional equation for the temperature is solved. The goal of this paper is to make `nsCouette` publicly available and thus enable DNS of rotating turbulent shear flows to a wide range of users in the mathematics, physics and engineering communities.

2. Software description

2.1. *Functionality*

In `nsCouette`, the governing equations are discretized using a pseudospectral Fourier–Galerkin ansatz for the azimuthal (θ) and the axial (z) direction. High-order explicit finite differences are used in r ; the only inhomogeneous direction. Periodic boundary conditions are assumed in z , thereby avoiding the need for dense grids close to the vertical boundaries. Note, that the comparison between DNS with axially periodic boundary conditions and laboratory experiments with solid end-plates is extremely satisfactory for a wide range of Re ranging from laminar to highly turbulent flows in the ultimate regime [2]. In addition, z -periodicity often provides a more accurate model of astrophysical and geophysical flows and prevents misleading physical interpretations due to undesired end-wall-effects [3, 4]. Details of the method and implementation were published in [5].

The temporal integration scheme has been upgraded to a predictor-corrector method [6]. This enables a variable time-step size with dynamic control, which is of advantage if the flow state is either suddenly modified (applying disturbances, changing rotation rates etc.) or naturally undergoes strongly transient dynamics.

Another significant upgrade is the extension to heat transfer when a temperature difference between the inner and outer cylinder walls is imposed. For this purpose, the Boussinesq approximation for rapidly rotating flows [7] has been implemented to account for buoyancy effects. In the distributed version of `nsCouette`, a negative temperature gradient in r is considered, whereas the gravity vector is aligned in z . However, other scenarios can be easily investigated by changing only a few lines of source code.

Additionally, divergence-free initial conditions, which automatically satisfy all boundary conditions, can now be used to easily excite selected combinations of individual Fourier modes. This enables the user to systematically investigate different transition scenarios.

nsCouette uses a single input file to define all relevant parameters (number of points, modes and time steps, rotation rates etc.) at program startup. At every restart, the spatial resolution can be freely changed using an automated interpolation and mode padding functionality. This and many other convenient features together with a number of example **makefiles** for the most common high-performance computing (HPC) platforms provide newcomers and non-expert users an easy start into the world of highly-resolved and massively parallel DNS. A user manual with several step-by-step tutorials is also included.

2.2. Software architecture

nsCouette is written in **Fortran** and, over time, has been ported to all major CPU-based HPC platforms. Amongst IBM Power, BlueGene and **x86_64** architectures – including a few generations of the prevalent Intel Xeon multi-core processors – it has also been ported to Xeon Phi (KnightsLanding), AMD EPYC (Naples) and ARMv8.1 (Marvell ThunderX2) platforms. Optimization for the NEC SX-Aurora vector architecture is underway.

Building the executable requires a modern **Fortran** compiler, a standard C compiler and only very few additional libraries, namely **MPI**, **BLAS/LAPACK**, **FFTW**, and optionally **HDF5**. All of them are commonly available as high-quality, open source software (e.g. **GCC**, **OpenMPI** [8], **FFTW** [9], **OpenBLAS**) and as vendor-optimized tool chains (e.g. Intel Parallel Studio XE, PSXE).

nsCouette runs on laptops and – for large-enough problems – can efficiently scale up to the largest HPC systems with tens of thousands of processor cores [10]. The parallelization scheme relies on a standard, one-dimensional slab decomposition into Fourier modes (in θ and z), which can be treated independently of each other in the solution of the linear terms occurring in the governing equations. The hybrid **MPI-OpenMP** parallelization scheme allocates multiple cores per **MPI** task and uses **OpenMP** threads to parallelize the computations of the linear terms. For computing the non-linear terms, global data transpositions based on **MPI_Alltoall** and task-local transposes are employed for gathering all Fourier modes locally on each **MPI** task.

Additionally, we provide a basic version of **nsCouette** written in C-CUDA, which runs very efficiently on single **CUDA**-capable GPU devices. Compute capability 2.0 (or higher), support for double-precision arithmetic and **NVIDIA**’s **CUDA** toolkit are required. The GPU-accelerated version relies on the **cuFFT**

library to compute Fourier transforms and linear algebra is performed using custom CUDA kernels.

2.3. Computational performance

Between runs, the number of MPI tasks can be changed and freely selected at program start up with the only restriction that it must divide the number of radial grid points (N_r). The hybrid parallelization scheme of **nsCouette** relaxes the limit imposed by the slab decomposition on the maximum number of processor cores and thus enables highly resolved DNS with $N_r = \mathcal{O}(10^3)$ using $\mathcal{O}(10^4)$ cores on contemporary HPC platforms [5]. In absolute terms, for an intermediate problem size with $N_r = 512$ and 513×1025 Fourier modes (i.e. Re up to $\mathcal{O}(10^4)$), the computation of a single timestep takes less than a second on 64 nodes (2560 cores) of a contemporary HPC cluster (see Fig. 1) and requires roughly 450 GB of main memory (RAM). On the SKL platform, this run achieves a performance of 1.5 TFlop/s, which, due to a rather moderate arithmetic intensity of the algorithm (0.3), is bounded by the memory bandwidth. When increasing the number of cores for a fixed problem size, the computations of the FFT (blue) and linear terms (red) show very good strong scalability, whereas the global transposes (green) ultimately limit the total parallel efficiency at large core counts, see Fig. 1. A comprehensive study of the parallel scalability and efficiency of the original version of **nsCouette** has been presented in [5], and its potential to scale up to extremely high core counts was shown in [10]. The upgraded version presented here exhibits essentially the same performance characteristics and parallel scalability.

The performance of the provided GPU-accelerated version of **nsCouette** was tested on two different NVIDIA graphics cards based on the Volta architecture: a Titan V and a Tesla V100. The runtime per timestep has been found to be similar on both cards over a wide range of problem sizes, see Fig. 2. The speed-up of the GPU version compared to the **MPI-OpenMP** version running on one node was shown to vary between a factor of three and 17, depending on the problem size and the particular choice of platform used as reference, see Fig. 2. Comparing a single GPU run against an **MPI-OpenMP** run on a single CPU node is a reasonable choice, since for server-class hardware both set-ups are roughly comparable in terms of price and electrical power consumption. However, 16 nodes (256 cores) were necessary to outperform the GPU version for small problem sizes, see Fig. 2. Currently, the maximum problem size applicable to the GPU version is limited by the amount of RAM available on the graphics card.

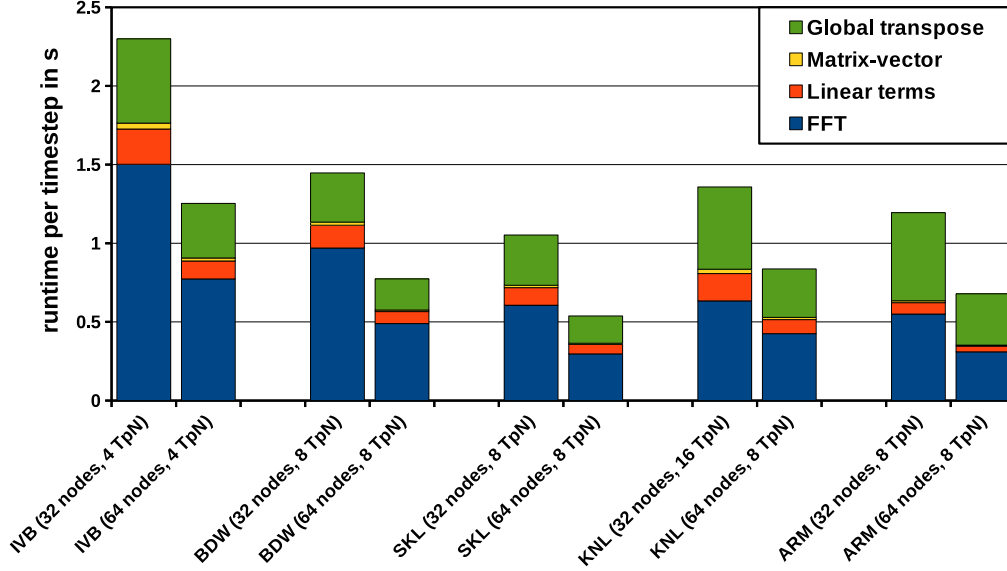


Figure 1: Runtime per timestep and breakdown into the main algorithmic components (different colours) of a typical **nsCouette** run ($N_r = 512$ and 513×1025 Fourier modes) computed on 32 and 64 dual-socket nodes of various HPC clusters, using a platform-specific number of MPI tasks/node (TpN). IVB: Intel Xeon E5-2680v2 (IvyBridge), 20 cores/node. BDW: Intel Xeon E5-2698v4 (Broadwell), 40 cores/node. SKL: Intel Xeon 6148 (Skylake), 40 cores/node. KNL: Xeon Phi 7230 (KnightsLanding), 64 cores/node. ARM: Marvell ThunderX2 ARM v8.1, 64 cores/node. The IVB and BDW clusters employ a Mellanox InfiniBand FDR network (56 Gbit/s), whereas SKL and KNL use Intel OmniPath (100 Gbit/s). The ARM cluster is interconnected with Cray Aries (80 Gbit/s). **nsCouette** was built using platform-optimized software tool chains (i.e. compilers and libraries) but no platform-specific optimization of the source code was performed. Corresponding **makefiles** are shipped with the code.

2.4. Data analysis and visualization

In **nsCouette** the Fourier coefficients and optionally also the primitive variables are dumped to individual files for each time step at user-specified output intervals. It also implements an easy-to-use checkpoint-restart mechanism based on the Fourier-coefficients for handling long-running DNS. The primitive variables – velocity (u_r , u_θ , u_z), pressure (p) and optionally temperature (T) – are written in **HDF5** format, along with metadata in a small **xdmf** file in order to facilitate analysis with common visualization tools like **ParaView** and **VisIt**. Both tools allow loading sequences of **xdmf** files produced by **nsCouette** and enable the user to interactively perform comprehensive visual and quantitative analysis of the flow field. Sample scripts based on the **Python** interface of **VisIt**, as well as a custom-made **ParaView** filter for handling the cylindrical coordinate system are distributed with **nsCouette**.

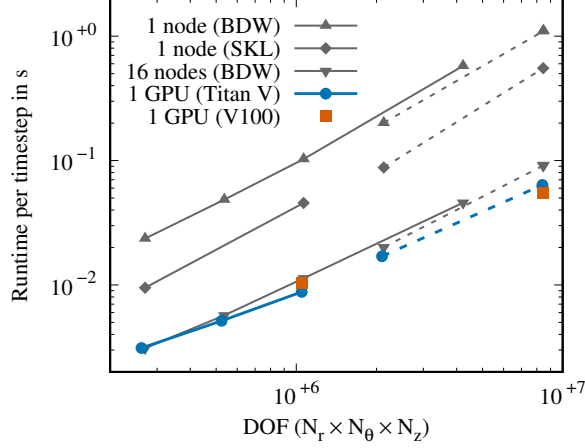


Figure 2: Performance of the GPU-accelerated version of **nsCouette** compared to the **MPI-OpenMP** version. Runtime per timestep for different numbers of degrees of freedom (DOF). The GPU code ran on a single NVIDIA Titan V and a single Tesla V100 graphics card. It was built using NVIDIA’s **CUDA** toolkit version 10.1. The hybrid code was built using Intel’s **PSXE2018** and ran on one and 16 nodes of different platforms. BDW: Intel Xeon E5-2620v4 (Broadwell), 16 cores/node, Mellanox InfiniBand FDR network (56 Gbit/s). SKL: Intel Xeon 6148 (Skylake), 40 cores/node. Solid (dashed) lines represent runs with $N_r = 64$ ($N_r = 128$) radial points and different numbers of Fourier modes.

A detailed visualization tutorial is included in the manual.

2.5. Quality assurance

The verification and validation of **nsCouette** is documented in [5]. For maintaining the correctness of the source code we make extensive use of the continuous integration (CI) functionality of **gitlab**. Upon every push to the repository, a number of regression tests are automatically triggered, including builds of the code in various configurations, and a static code analysis using the **Forcheck** tool. In addition, a number of short test runs are automatically launched using runtime-checks and the tightest debug settings of the compiler to identify undefined variables, out-of-bounds errors and alike. The numerical results are then rigorously verified against previously recorded reference runs. A final validation run compares the wave speed of a simulated wavy vortex flow with an experimentally determined value [11], which is considered successful if the wave speeds match up to 10^{-4} . The entire CI configuration and results for every push are publicly accessible through the web interface of our **nsCouette** development site.

3. Illustrative Example

The flow of a fluid between a hot rotating cylinder and a cooled stationary cylindrical enclosure is a simple model to investigate heat transfer in many engineering applications [12], including the cooling of rotating machinery [13]. At low angular speeds (Re_i) and small temperature differences (Ra), the heat transfer is purely conductive and the only non-zero flow component is the azimuthal one. In this simple case, termed basic state, the governing equations admit simple analytic solutions for $u_\theta(r)$ and $T(r)$, which only depend on r . The heat transfer can be enhanced by either increasing Re_i (forced convection) or by increasing Ra (natural convection). In both cases, the basic state exhibits a sequence of distinct instabilities ultimately leading to turbulent heat transfer [14]. A measure of the efficiency is given by the Nusselt number (Nu_i), which is the ratio of total heat transfer at the inner cylinder wall, normalized by that of the basic state at the same temperature difference.

Fig. 3 summarizes the results of three DNS with increasing temperature difference at a fixed rotation rate, using `nsCouette`. The first DNS was ini-

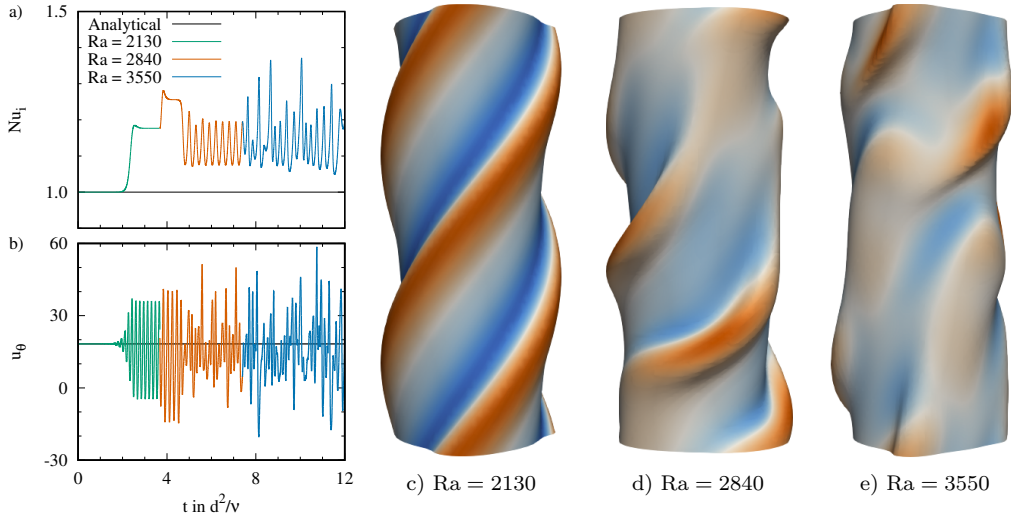


Figure 3: Temporal evolution of the Nusselt number (Nu_i) at the inner cylinder wall and the streamwise velocity component (u_θ) at a mid-gap position as Ra increases for a fixed inner cylinder rotation ($Re_i = 50$). The final flow state for each Ra is visualized with instantaneous temperature iso-surfaces ($T = 0$), which are color-coded by inwards/outwards (blue/red) facing values of the wall-normal velocity component u_r .

tialized by applying small single harmonic disturbances to the basic state at $Re_i = 50$ and $Ra = 2130$. Fig. 3a shows that initially $Nu_i \approx 1$, corresponding

to purely conductive heat transfer. However, after roughly two viscous time units, a sharp increase of Nu_i is observed, indicating that the basic state has become unstable. This is confirmed by the time-series of u_θ at a fixed mid-gap probe location in the computational domain, see Fig. 3b. The final state of this run ($t \approx 3.5d^2/\nu$) is visualized in Fig. 3c, which shows a three-dimensional rendering of a $T = 0$ iso-surface generated with **ParaView**. The temperature surface is color-coded by inwards/outwards (blue/red) facing values of the wall-normal velocity component u_r . By saving several snapshots, a movie can be easily produced, which reveals a spiral flow pattern rotating at a constant speed like a barber pole. This explains why the u_θ signal reaches a state where it is periodic time: the spiral pattern passes repeatedly through the probe location at which the velocity is recorded without changing its shape. This also explains why the integral heat flux (Nu_i) remains constant. The second and third DNS were initialized with the final state of the former runs and by increasing the Rayleigh number to $Ra = 2840$ and 3550 , respectively. The time series and the final states of these runs are also shown in Fig. 3. They reveal that the flow state undergoes a sequence of transitions to different flow states with increasing spatio-temporal complexity as Ra increases. This, and other illustrative examples, are documented in the tutorial section of the provided user guide.

4. Impact

Our software completes the list of publicly available Navier–Stokes solvers for the three most common prototypes of wall-bounded shear flows: plane Couette flow (channelflow.org [15]), pipe flow (openpipeflow.org [16]) and Taylor–Couette flow (this paper). **nsCouette** can be quickly installed and productively used by researchers interested in pattern formation and chaos, for which TC flow has long been a paradigm [17]. The example of section 3 can be run in a laptop and is meant to illustrate how easy results can be obtained, analyzed and interpreted. We however remark that **nsCouette** has been designed to enable users with little experience in HPC and DNS to easily perform highly-resolved simulations of turbulence. It is a powerful tool to study angular momentum transport, mixing and heat transfer and has already contributed to a better understanding of astrophysical [18] and geophysical [19, 20] flows. Because of its modular structure and moderate code complexity, **nsCouette** is easy to read and new functionality can be added with little effort. As an example for this, we also provide **nsPipe**; a modified version to simulate turbulent flows in a straight pipe geometry. It follows the numerical formulation of openpipeflow.org [16], and uses the same code structure and parallelization as **nsCouette**. Extensions including

the modelling of polymer additives [21] and two-phase flows [22] have already been developed and tested and will be released in the future.

Acknowledgements

This work was supported by the Max Planck Society and partially funded by the German Research Foundation (DFG) through the priority programme *Turbulent Superstructures* (SPP1881). AVM was supported by the European Research Council (ERC) through the *COTURB* project (ERC-2014.AdG-669505). Computational resources provided by the following institutions are gratefully acknowledged: The Argonne Leadership Computing Facility, which is a DOE Office of Science User Facility (DE-AC02-06CH11357). The Isambard UK National Tier-2 HPC Service operated by GW4 and the UK Met Office, and funded by EPSRC (EP/P020224/1). Further computations were performed on the HPC systems Hydra, Draco and Cobra at the MPCDF in Garching.

References

References

- [1] Y. Bazilevs, I. Akkerman, Large eddy simulation of turbulent Taylor-Couette flow using isogeometric analysis and the residual-based variational multiscale method, *Journal of Computational Physics* 229 (9) (2010) 3402–3414. doi:10.1016/j.jcp.2010.01.008.
- [2] S. Grossmann, D. Lohse, C. Sun, High Reynolds Number Taylor-Couette Turbulence, *Annual Review of Fluid Mechanics* 48 (1) (2016) 53–80. doi:10.1146/annurev-fluid-122414-034353.
- [3] E. M. Edlund, H. Ji, Nonlinear stability of laboratory quasi-Keplerian flows, *Physical Review E* 89 (2) (2014) 021004. doi:10.1103/PhysRevE.89.021004.
- [4] J. M. Lopez, M. Avila, Boundary-layer turbulence in experiments on quasi-Keplerian flows, *Journal of Fluid Mechanics* 817 (2017) 21–34. doi:10.1017/jfm.2017.109.
- [5] L. Shi, M. Rampp, B. Hof, M. Avila, A hybrid MPI-OpenMP parallel implementation for pseudospectral simulations with application to Taylor–Couette flow, *Computers & Fluids* 106 (2015) 1–11. doi:10.1016/j.compfluid.2014.09.021.

- [6] A. Guseva, A. P. Willis, R. Hollerbach, M. Avila, Transition to magnetorotational turbulence in Taylor–Couette flow with imposed azimuthal magnetic field, *New Journal of Physics* 17 (9) (2015) 093018. doi:10.1088/1367-2630/17/9/093018.
- [7] J. M. Lopez, F. Marques, M. Avila, The Boussinesq approximation in rapidly rotating flows, *Journal of Fluid Mechanics* 737 (2013) 56–77. doi:10.1017/jfm.2013.558.
- [8] E. Gabriel, G. E. Fagg, G. Bosilca, T. Angskun, J. J. Dongarra, J. M. Squyres, V. Sahay, P. Kambadur, B. Barrett, A. Lumsdaine, R. H. Castain, D. J. Daniel, R. L. Graham, T. S. Woodall, Open MPI: Goals, Concept, and Design of a Next Generation MPI Implementation, in: D. Kranzlmüller, P. Kacsuk, J. Dongarra (Eds.), *Recent Advances in Parallel Virtual Machine and Message Passing Interface. EuroPVM/MPI 2004. Lecture Notes in Computer Science*, Vol. 3241, Springer, Berlin, Heidelberg, 2004, pp. 97–104. doi:10.1007/978-3-540-30218-6_19.
- [9] M. Frigo, S. Johnson, The Design and Implementation of FFTW3, *Proceedings of the IEEE* 93 (2) (2005) 216–231. doi:10.1109/JPROC.2004.840301.
- [10] M. Rampp, J. M. Lopez, L. Shi, B. Hof, M. Avila, Extreme scaling of NScouette, a pseudospectral DNS code, in: *INSIDE: Innovative Supercomputing in Germany*, Vol. 12, 2014, pp. 48–50.
- [11] G. P. King, W. Lee, Y. Li, H. L. Swinney, P. S. Marcus, Wave speeds in wavy Taylor-vortex flow, *Journal of Fluid Mechanics* 141 (1984) 365–390. doi:10.1017/S0022112084000896.
- [12] M. Ali, P. D. Weidman, On the stability of circular Couette flow with radial heating, *Journal of Fluid Mechanics* 220 (1990) 53–84. doi:10.1017/S0022112090003184.
- [13] D. A. Howey, P. R. N. Childs, A. S. Holmes, Air-Gap Convection in Rotating Electrical Machines, *IEEE Transactions on Industrial Electronics* 59 (3) (2012) 1367–1375. doi:10.1109/TIE.2010.2100337.
- [14] J. M. Lopez, F. Marques, M. Avila, Conductive and convective heat transfer in fluid flows between differentially heated and rotating cylinders, *International Journal of Heat and Mass Transfer* 90 (2015) 959–967. doi:10.1016/j.ijheatmasstransfer.2015.07.026.

- [15] J. F. Gibson, Channelflow: A spectral Navier-Stokes simulator in C++, Tech. rep., University of New Hampshire (2014).
URL www.channelflow.org
- [16] A. P. Willis, The Openpipeflow Navier–Stokes solver, *SoftwareX* 6 (2017) 124–127. doi:[10.1016/j.softx.2017.05.003](https://doi.org/10.1016/j.softx.2017.05.003).
- [17] M. A. Fardin, C. Perge, N. Taberlet, "The hydrogen atom of fluid dynamics" – introduction to the Taylor–Couette flow for soft matter scientists, *Soft Matter* 10 (20) (2014) 3523. doi:[10.1039/c3sm52828f](https://doi.org/10.1039/c3sm52828f).
- [18] L. Shi, B. Hof, M. Rampp, M. Avila, Hydrodynamic turbulence in quasi-Keplerian rotating flows, *Physics of Fluids* 29 (4) (2017) 044107. doi:[10.1063/1.4981525](https://doi.org/10.1063/1.4981525).
- [19] C. Leclercq, J. L. Partridge, P. Augier, C.-C. P. Caulfield, S. B. Dalziel, P. F. Linden, Nonlinear waves in stratified Taylor-Couette flow. Part 1. Layer formation. [arXiv:1609.02885](https://arxiv.org/abs/1609.02885).
- [20] C. Leclercq, J. L. Partridge, C.-C. P. Caulfield, S. B. Dalziel, P. F. Linden, Nonlinear waves in stratified Taylor-Couette flow. Part 2. Buoyancy flux. [arXiv:1609.02886](https://arxiv.org/abs/1609.02886).
- [21] J. M. Lopez, G. H. Choueiri, B. Hof, Dynamics of viscoelastic pipe flow at low Reynolds numbers in the maximum drag reduction limit, *Journal of Fluid Mechanics* 874 (2019) 699–719. doi:[10.1017/jfm.2019.486](https://doi.org/10.1017/jfm.2019.486).
- [22] B. Song, C. Plana, J. M. Lopez, M. Avila, Phase-field simulation of core-annular pipe flow, *International Journal of Multiphase Flow* 117 (2019) 14–24. doi:[10.1016/j.ijmultiphaseflow.2019.04.027](https://doi.org/10.1016/j.ijmultiphaseflow.2019.04.027).

Required Metadata

Current code version

Nr.	Code metadata description	Please fill in this column
C1	Current code version	1.0
C2	Permanent link to code/repository used for this code version	https://github.com/dfeldmann/nsCouette
C3	Legal Code License	GPLv3
C4	Code versioning system used	git
C5	Software code languages, tools, and services used	Fortran, C (for some housekeeping tasks), MPI, OpenMP
C6	Compilation requirements, operating environments & dependencies	Developed and tested under Linux and IBM AIX. Compiler: A Fortran 2003 compiler which is OpenMP-3 compliant, a basic C compiler, an MPI library with support for MPI_THREAD_SERIALIZED, a serial BLAS/LAPACK library, a serial but fully thread-safe FFTW3 installation or equivalent, for output and visualization (optional): an MPI-parallel HDF5 installation.
C7	Link to developer documentation/manual	https://gitlab.mpcdf.mpg.de/mjr/nsCouette
C8	Support email for questions	nsCouette@zarm.uni-bremen.de

Table 1: Code metadata (mandatory)