Interdisziplinäres Zentrum
für Wissenschaftliches Rechnen (IWR)
Ruprecht–Karls–Universität Heidelberg

Prof. Dr. Vincent Heuveline
Dr. Philipp Gerstner

Programming Exam 1

11.10.2021 – 18.10.2021

Computational Fluid Dynamics (Summer Term 2021)

Exercise 1 (Programming). The Magnus Effect
In this exam, you will simulate the so-called Magnus effect. This phenomenon occurs if a rotating object is exposed to a fluid stream.

In practice, the Magnus effect can be observed in football free kicks (https://www.youtube.com/watch?v=crKwlbwvr88) and can be used to drive ships (https://www.youtube.com/watch?v=CODdHkbz6o0&t=182s).

In addition, we assume that the rotating foil is of higher temperature than the surrounding fluid. To this end, consider the instationary Boussinesq equations, which is obtained by augmenting the Navier-Stokes equation by the Buoyancy force $\gamma\theta\mathbf{g}$ and by the heat equation with temperature θ, on $[0,T]\times\Omega$:

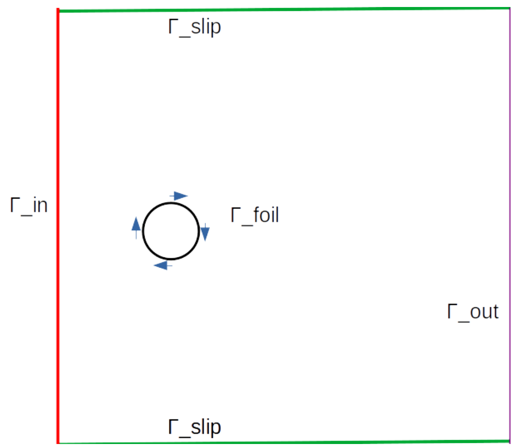
$$
\begin{aligned}
\partial_t \mathbf{v} - \nu\Delta\mathbf{v} + (\mathbf{v}\cdot\nabla)\mathbf{v} + \nabla p + \gamma\theta\mathbf{g} &= 0 \\
\nabla\cdot\mathbf{v} &= 0 \\
\partial_t\theta - \kappa\Delta\theta + (\mathbf{v}\cdot\nabla)\theta &= 0 \\
\mathbf{v} &= (v_x, 0) && \text{on } \Gamma_{in}\cup\Gamma_{slip} \\
\mathbf{v} &= \mathbf{v}_{rot} && \text{on } \Gamma_{foil} \\
(\nu\nabla\mathbf{v} - pI)\vec{n} &= 0 && \text{on } \Gamma_{out} \\
\theta &= 0 && \text{on } \Gamma_{in} \\
\theta &= \theta_f && \text{on } \Gamma_{foil} \\
\kappa\nabla\theta\cdot\vec{n} &= 0 && \text{on } \Gamma_{out}\cup\Gamma_{slip} \\
\mathbf{v}(0) &= 0 \\
\theta(0) &= 0
\end{aligned}
$$

with viscosity $\nu > 0$, thermal diffusion $\kappa > 0$, thermal expansion $\gamma > 0$, gravity $\mathbf{g} = (0,-1)^T$ and domain Ω and boundaries given by the figure below. The material numbers corresponding to the boundary sections are given in the .xml file. For a given time period Δt, the velocity inflow boundary condition is given by

$$
v_x(t) = \begin{cases} \frac{t}{\Delta t} v_{in} & t\in[0,\Delta t] \\ v_{in} & t > \Delta t \end{cases}.
$$

For the rotor (Γ_{foil}) we assume that the fluid sticks to the surface (i.e. no-slip boundary condition) and the foil is rotating clockwise with the following number of revolutions per second:

$$
f(t) = \begin{cases} \frac{t}{\Delta t} f_{max} & t\in[0,\Delta t] \\ f_{max} & t > \Delta t \end{cases}.
$$



In this setting, the foil is exposed to the force

$$\mathbf{f}_{foil} = \int_{\Gamma_{foil}} \sigma(\mathbf{v}, p)\vec{n}_{foil}\, \mathrm{dS}$$

with Cauchy stress tensor

$$\sigma(\mathbf{v}, p) = \mu(\nabla\mathbf{v} + \nabla\mathbf{v}^T) - pI, \text{ and } \mu = \nu \text{ (unit density) },$$

and $\vec{n}_{foil}$ denotes the unit normal on Γ_{foil} directing into the fluid domain. We additionally consider the following quantities of interest:

$$\begin{aligned}
&\text{kinetic energy:} && E_{kin} = \int_\Omega |\mathbf{v}|^2 dV \\
&\text{dissipation energy:} && E_{dis} = \nu \int_\Omega |\nabla\mathbf{v}|^2 dV \\
&\text{divergence:} && E_{div} = \left(\int_\Omega |\nabla \cdot \mathbf{v}|^2 dV \right)^{0.5}
\end{aligned}$$

For time discretization, we use the 1-step BDF method: for $n = 1, \dots, N$:

$$\begin{aligned}
\frac{1}{k}(\mathbf{v}_n - \mathbf{v}_{n-1}) - \nu\Delta\mathbf{v}_n + \delta_{conv}(\mathbf{v}_{n-1} \cdot \nabla)\mathbf{v}_n + \nabla p_n + \gamma\theta_n\mathbf{g} &= 0 \\
\nabla \cdot \mathbf{v}_n &= 0 \\
\frac{1}{k}(\theta_n - \theta_{n-1}) - \kappa\Delta\theta_n + \delta_{conv}(\mathbf{v}_{n-1} \cdot \nabla)\theta_n &= 0 \\
\mathbf{v}_0 &= 0 \\
\theta_0 &= 0
\end{aligned} \tag{1}$$

and the 2-step BDF method: for $n = 2, \ldots, N$:

$$\begin{aligned}
\frac{1}{k}(\frac{3}{2}\mathbf{v}_n - 2\mathbf{v}_{n-1} + \frac{1}{2}\mathbf{v}_{n-2}) - \nu\Delta\mathbf{v}_n + \delta_{conv}((2\mathbf{v}_{n-1} - \mathbf{v}_{n-2}) \cdot \nabla)\mathbf{v}_n + \nabla p_n + \gamma\theta_n\mathbf{g} &= 0 \\
\nabla \cdot \mathbf{v}_n &= 0 \\
\frac{1}{k}(\frac{3}{2}\theta_n - 2\theta_{n-1} + \frac{1}{2}\theta_{n-2}) - \kappa\Delta\theta_n + \delta_{conv}((2\mathbf{v}_{n-1} - \mathbf{v}_{n-2}) \cdot \nabla)\theta_n &= 0 \\
\mathbf{v}_0 &= 0
\end{aligned} \tag{2}$$

with time step size $k = \frac{T}{N}$. When using BDF2, the first step $\mathbf{v}_1$ can be computed by using BDF1 with time step size $\frac{1}{2}k$. The parameter $\delta_{conv} \in \{0, 1\}$ indicates whether to solve the Boussinesq equations with or without convection.

In order to cope with convection-dominated situations, we use the following SUPG stabilization term, that should be added to the variational formulation:

$$\begin{aligned}
a_{supg,\mathbf{v}}(\mathbf{v}_h, \mathbf{w}_h) &:= \sum_{K \in \mathcal{T}_h} \delta_K^{(v)}((\overline{\mathbf{v}} \cdot \nabla)\mathbf{v}_h, (\overline{\mathbf{v}} \cdot \nabla)\mathbf{w}_h)_K \\
a_{supg,\theta}(\theta_h, \beta_h) &:= \sum_{K \in \mathcal{T}_h} \delta_K^{(v)}((\overline{\mathbf{v}} \cdot \nabla)\theta_h, (\overline{\mathbf{v}} \cdot \nabla)\beta_h)_K
\end{aligned} \tag{3}$$

with convection velocity $\overline{\mathbf{v}} = \mathbf{v}_{n-1}$ in case of BDF1 and $\overline{\mathbf{v}} = 2\mathbf{v}_{n-1} - \mathbf{v}_{n-2}$ for BDF2. The cell-wise coefficients are given by

$$\delta_K^{(v)} = \frac{1}{2}C_v \begin{cases} h_K & \text{if } \nu < h_k \\ h_K^2 & \text{else} \end{cases}$$

with non-negative constant C_v.

You have to implement the following steps:

a. Implement the computation of the quantities of interest $\mathbf{f}_{foil}$, E_{kin}, E_{dis} and E_{div}. After being computed, these values are written into a csv file.

b. Implement a time loop for performing the described time stepping schemes. The choice of the method (BDF1 or BDF2) is given by the parameter BDFType in exam_2.xml.

c. Implement the Dirichlet boundary conditions. The necessary physical parameters and material numbers are specified in exam_2.xml and are passed to the Dirichlet evaluation class.

d. Implement the local assembler for the left-hand side system matrix and right-hand side system vector that correspond to the Finite Element discretization of the time-discrete systems (1) and (2). Include the stabilization terms (3).

The choice of the underlying method is determined by the control parameters bdf_type, with_convection ($=\delta_{conv}$) and use_stab (if true, stabilization terms are active). All occurring constants and parameters are passed to the assembler class LocalFlowAssembler by the function set_parameters.

Your implementation should be done in such a way, that these parameters can be set at run-time (they are all specified in exam_2.xml), i.e. no recompilation should be necessary.

e. Implement a pressure stabilization technique of your choice to cope with velocity-pressure pairs, that do not fulfill the inf-sup condition. Pressure stabilization should be active, if use_stab=true and Cp > 0.

After you have implemented the previously described steps, you can perform the following simulations. For each of the upcoming sub-tasks, use the csv data to generate a single time-plot to visualize the quantities of interest (e.g. with gnuplot, python matplotlib, excel or libre office calc). Further, use the provided ParaView state to visualize the solution and submit a screenshot of the solution at the final time.

If not specified otherwise, let $\nu = 10^{-3}$, $\kappa = 10^{-3}$, $\theta_f = 2$, $\gamma = 1$, $T = 1$, $k = 0.01$, $f_{max} = -10$, $v_{in} = 1$, $\Delta t = 0.1$, $C_v = 0$ velocity of polynomial degree 2, pressure and temperature of polynomial degree 1.

1.) Solve the Boussinesq equations without convection equations with BDF1.

2.) Solve the Boussinesq equations with BDF1.

3.) Solve the Boussinesq equations with BDF2.

4.) Solve the Boussinesq equations with BDF2, $C_v = 1$, $\nu = 10^{-4}$ on the longer time interval $T = 5$.

5.) Solve the Boussinesq equations with BDF2, $C_v = 1$, $\nu = 10^{-4}$, pressure stabilization and velocity degree 1 on the longer time interval $T = 5$.

Note: Even though you can't manage to do all of the previously described implementation steps, it is still possible to perform some of the simulations. If your BDF2 implementation does not work, you may also use BDF1 for task 4.) and 5.) (though getting less points for these sub-tasks). Analogously, you may dispense SUPG in task 5.) and use $\nu = 10^{-3}$ instead.

Note: You are free in choosing the number of cores to be used for the calculations. However, we recommend to use at least 2.