Practical Session 3/3
Multiple Shooting and Parareal

All of these problems are based on the slides provided during the morning sessions. Some additional source code (PYTHON or MATLAB) may be provided to help you with the implementation tasks. For each problem, you don't have to implement a parallel version of the PinT algorithm : you may use only one process that does the work of all parallel processes.

Problem 1 *Multiple Shooting in Time. We first want to analyze and implement the method of Chartier and Philippe.*

1. *First, we consider the linear system*
$$f(\boldsymbol{x}) = A\boldsymbol{x} + \boldsymbol{b} = 0, \tag{1}$$
with $\boldsymbol{b} \in \mathbb{R}^p$ a given vector, A an invertible square matrix of size $(p \times p)$, and x the vector of unknowns. Show that the Newton method applied to this problem converges in one iteration.
2. *We now consider the linear system of ODEs*
$$\frac{d\boldsymbol{u}}{dt} = A\boldsymbol{u}, \quad \boldsymbol{u}(0) = \boldsymbol{u}_0 \in \mathbb{R}^p, \tag{2}$$
with A an invertible square matrix of size $(p \times p)$. Show that the Multiple Shooting method applied to this problem, using either an exact solver or the Forward Euler method, converges in only one iteration.
3. *Implement the Multiple Shooting method in a generic solver. Consider the two following approaches to compute the Jacobian matrix*
$$V_n(t) = \frac{\partial \boldsymbol{u}_n}{\partial \boldsymbol{U}_n}(t, \boldsymbol{U}_n) : \tag{3}$$
   - (a) *Solve the ODE for the matrix $V_n(t)$ using the Forward Euler method and a given time step.*
   - (b) *Define the propagator function on one time sub-interval*
$$\psi_n(\boldsymbol{U}) := \boldsymbol{u}_n(T_{n+1}, \boldsymbol{U}), \tag{4}$$
and then approximate the product of the Jacobian matrix with a vector $\boldsymbol{V}$ using finite differences,
$$J_{\psi_n}(\boldsymbol{U}) \cdot \boldsymbol{V} = \frac{\psi_n\left(\boldsymbol{U} + \epsilon\frac{\boldsymbol{V}}{||\boldsymbol{V}||}\right) - \psi_n(\boldsymbol{U})}{\epsilon}, \tag{5}$$
with ϵ a small number (usually, close to the square root of the machine precision).
4. *Solve the Lorenz system for $\sigma = 10, r = 28, b = 8/3, u_0 = (20, -5, 5)$ and $T = 5$. You may use $N = 180$ time sub-intervals, and the appropriate amount of time steps such that the numerical error using the Forward Euler scheme is less than $1e^{-3}$ for the whole trajectory. Could you compare the computational cost of this method with the one using a sequential approach ? What is its potential in terms of parallelism ?*

Problem 2 *Parareal. First, we want to obtain an analytical model to determine the maximum theoretical speed-up that can be obtained with the parallel algorithm. We define it as :*
$$S = \frac{\textit{computation time of sequential solve}}{\textit{computation time of parallel solve}}. \tag{6}$$
We denote T_F and T_G the computation time for the fine and coarse solver on one time sub-interval. N is the number of parallel processes (and also the number of time sub-intervals), and we define the parallel efficiency as
$$E = S/N. \tag{7}$$

1. *Find a distribution of tasks between processors that maximizes the speed-up of Parareal for any T_F and T_G value.*
2. *Show that the speed-up is bounded by N/K.*

We now want to implement Parareal as a generic solver. It should be generic enough so it could be applied to any kind of vector (or scalar) ODE. We denote by n_F and n_G the number of time steps per time sub-interval for the fine and the coarse solver.

3. *Solve the Lorenz system using the same settings as in the previous problem (initial value, problem parameters, use of Forward Euler, fine solver accuracy). What is the value of n_G (using Forward Euler), that maximizes the speedup of Parareal ? In the same way, could you find the optimal value for (N, n_G) that maximizes the parallel efficiency ?*
4. *Solve the heat equation with Parareal, using the same settings as in the first problem of the second series. For a given fine solver accuracy, can you find the optimal value for (N, n_G) that maximizes the parallel efficiency ? How does this evolve when you increase the fine solver accuracy ?*
5. *Solve the transport equation with Parareal, using the same settings as in the frist problem of the second series. How does Parareal convergence evolve when you increase the fine solver accuracy ? What does this tell about the applicability of the method for the transport equation ?*