COUPLING METHODS FOR HEAT-TRANSFER AND HEAT-FLOW: OPERATOR SPLITTING AND PARAREAL.

JÜRGEN GEISER * AND STEFAN GÜTTEL †

Abstract.
In this paper, we propose an operator splitting method for coupling heat-transfer and heat-flow equations. The motivation arose of industrial software packages, which have developed independently heat-transfer solvers (e.g. Aura-Fluid software package) and heat-flow solvers (e.g. Openfoam). Such packages are often simple coupled by A-B splitting and they simulate the influence of solar heat in car bodies. One of the main problems is to accelerate the code by iterative operator splitting methods and with a parallel interfaces.

We propose Parareal as well applicable for parallel method for parabolic problems and also a speedup to mixed parabolic and hyperbolic problems is possible and delicate. Our idea is to combine the splitting schemes as a time-splitting method to adapt the time steps with Parareal as a time-parallelization method. Our analysis of these new splitting techniques provides error estimates which are validated at certain benchmark problems.

Keywords: Heat transfer, Heat flux, operator splitting, iterative operator splitting, higher order time-discretization schemes, Parareal.
AMS subject classifications. 65M15, 65L05, 65M71.

1. Mathematical Model. In the following, we have a delicate model of coupled heat transfer and radiation. While the heat transfer and radiation is solved with the Aura software package, the flow field of the temperature is computed with a flow-field solver, e.g. Openfoam or Vectis, see [14]. The idea is to obtain a speedup with fast coupling schemes and parallel time schemes.

We deal with the following equations.

1. Heat-transfer equation:

$$\begin{aligned} \partial_t\, T &= \nabla\, \cdot\, (K\nabla T) - \nabla \cdot \mathbf{v}T, \\ T(\mathbf{x}, t_0) &= T_0(\mathbf{x}), \end{aligned} \tag{1.1}$$

where the unknown temperature is T, A is the diffusion operator and B the convection operator.

2. Heat-Flow equation:

$$\begin{aligned} \partial_t \mathbf{v} &= -(\mathbf{v} \cdot \nabla)\mathbf{v} - \nabla p(T), \\ \mathbf{v}(\mathbf{x}, t_0) &= \mathbf{v}_0(\mathbf{x}), \end{aligned} \tag{1.2}$$

where the unknown flux is $\mathbf{v}$, A is nonlinear flow operator and B is the pressure operator.

Both equations are solved simultaneously together with an iterative scheme.

REMARK 1.1. *For simplifications we assume p is not dependent of T, and we solve equation 1.2 first and use the result for equation 1.1. Further, we decouple equation 1.1 with operator splitting schemes.*

REMARK 1.2.

*Humboldt University of Berlin, Unter den Linden 6, D-10099 Berlin, Germany, E-mail: geiser@mathematik.hu-berlin.de
†University of Geneva, Switzerland, E-mail: guettel@gmail.com

The more delicate case is to couple the two equations, if they are strong coupled and nonlinear, means $\nabla p(T)$. Here we apply an iterative splitting scheme, that couples the two equations via relaxation.

2. Splitting Methods. In the following we briefly discuss the coupling methods that are used for the heat transfer equation.

2.1. Lie-Trotter or A-B Splitting method. The standard implemented scheme is the well-known $A - B$ splitting method.

A-B splitting:

$$\begin{aligned}
&\partial_t \mathbf{v} = -(\mathbf{v}\cdot\nabla)\mathbf{v}, && (2.1)\\
&\text{with } t^n \le t \le t^{n+1},\ \mathbf{v}(\mathbf{x},t^n) = \mathbf{v}^n(\mathbf{x}),\\
&\frac{\partial T}{\partial t} = \nabla\ \cdot\ (K\nabla T) - \nabla\cdot\tilde{\mathbf{v}}T, && (2.2)\\
&\text{with } t^n \le t \le t^{n+1},\ \tilde{\mathbf{v}}(\mathbf{x},t^n) = \mathbf{v}^{n+1}(\mathbf{x})\ ,\ T(\mathbf{x},t^n) = T(\mathbf{x},t^n)
\end{aligned}$$

where T^n is the known initial value of the previous solution and $T(t^{n+1}) = T_2(\mathbf{x},t^{n+1})$ is the approximated solution of the full equation.

We have a global splitting error of $O(\Delta t)$, where Δt is the time step.

2.2. Strang-Splitting Method. Strang Splitting:

$$\begin{aligned}
&\frac{\partial T_1(x,t)}{\partial t} = \nabla\ \cdot\ (K\nabla T_1) - \nabla\cdot\mathbf{v}_1 T_1, && (2.3)\\
&\text{with } t^n \le t \le t^{n+1/2},\ \mathbf{v}_1(\mathbf{x},t^n) = \mathbf{v}^n(\mathbf{x})\ ,\ T_1(\mathbf{x},t^n) = T(\mathbf{x},t^n)\\
&\partial_t \mathbf{v}_2 = -(\mathbf{v}_2\cdot\nabla)\mathbf{v}_2, && (2.4)\\
&\text{with } t^n \le t \le t^{n+1},\ \mathbf{v}_2(\mathbf{x},t^n) = \mathbf{v}^n(\mathbf{x}),\\
&\frac{\partial T_3(x,t)}{\partial t} = \nabla\ \cdot\ (K\nabla T_3) - \nabla\cdot\mathbf{v}_3 T_3, && (2.5)\\
&\text{with } t^{n+1/2} \le t \le t^{n+1},\ \mathbf{v}_3(\mathbf{x},t^{n+1/2}) = \mathbf{v}_2^{n+1}(\mathbf{x})\ ,\ T_3(x,t_{n+1/2}) = T_1(x,t^{n+1/2})\ ,
\end{aligned}$$

where T^n is the known initial value of the previous solution and $T(t^{n+1}) = T_3(x,t^{n+1})$ is the approximated solution of the full equation.

Here we obtain a coupling method of one order higher than the previous one, i.e., of order $O(\Delta t^2)$.

REMARK 2.1. *With such improved methods, we obtain higher accuracy and faster computations.*

2.3. Iterative splitting method.

2.3.1. Linear Case. The following algorithm is based on the iteration with fixed splitting discretization step-size τ. On the time interval $[t^n, t^{n+1}]$ we solve the following subproblems consecutively for $i = 0, 2, \ldots, 2m$.

The iterative method is given as, see also [3],

$$\begin{aligned}
&\partial_t \mathbf{v}_i = -(\mathbf{v}_i\cdot\nabla)\mathbf{v}_i, && (2.6)\\
&\text{with } t^n \le t \le t^{n+1},\ \mathbf{v}_i(\mathbf{x},t^n) = \mathbf{v}^n(\mathbf{x}),\\
&\frac{\partial T_i}{\partial t} = \nabla\ \cdot\ (K\nabla T_i) - \nabla\cdot\mathbf{v}_{i-1} T_i, && (2.7)\\
&\text{with } t^n \le t \le t^{n+1},\ \mathbf{v}_{i-1}(\mathbf{x},t^n) = \mathbf{v}^n(\mathbf{x})\ ,\ T(\mathbf{x},t^n) = T(\mathbf{x},t^n)
\end{aligned}$$

where $T^n, \mathbf{v}^n$ is the well-known split approximation at time level $t = t^n$ [3].

Here we solve the time-discretization with a BDF4 method.

The higher order is obtained by applying recursively the fixed-point iteration to reconstruct the analytical solution of the coupled operators, see [5].

Generalization to vectorial schemes (systems of ODEs)

We deal with a vectorial iterative scheme, given as an inner and outer iterative scheme.

While the outer iterative scheme is a Waveform-relaxation scheme and could be seen as a coarse iterative scheme, while we iterate over the full system in one step. The inner iterative scheme is a multi-iterative Waveform-relaxation scheme, while it iterates over each ODE in m-steps.

Outer Iteration (Waveform-Relaxation, iteration over the full system):

$$\frac{dU_i}{dt} = \mathcal{A}U_i + \mathcal{B}U_{i-1} + F, \tag{2.8}$$
$$U_i(t^n) = U(t^n), \tag{2.9}$$
$$i = 1, \ldots, I, \tag{2.10}$$

where $U_i = (u_{1,i}, \ldots, u_{m,i})$ and m is the number of ODE's, further

$$\mathcal{A} = \begin{pmatrix} A_{1,1} & A_{1,2} & \ldots & A_{1,m} \\ A_{2,1} & A_{2,2} & \ldots & A_{2,m} \\ \vdots & & & \\ A_{m,1} & A_{m,2} & \ldots & A_{m,m} \end{pmatrix}, \tag{2.11}$$

and

$$\mathcal{B} = \begin{pmatrix} B_{1,1} & B_{1,2} & \ldots & B_{1,m} \\ B_{2,1} & B_{2,2} & \ldots & B_{2,m} \\ \vdots & & & \\ B_{m,1} & B_{m,2} & \ldots & B_{m,m} \end{pmatrix}, \tag{2.12}$$

are matrices of the ODE system, for example the diagonal and outer-diagonal matrices and F is a right hand side (e.g. source term).

Inner Iteration (iterative splitting, relaxation over each sub-equation):

$$\frac{dU_{1,j_1}}{dt} = A_{1,1}U_{1,j_1} + A_{1,2}U_{1,j_1-1} + \ldots + A_{1,m}U_{m,j_1-1}$$
$$+B_{1,1}U_{1,j_1-1} + B_{1,2}U_{1,j_1-1} + \ldots + B_{1,m}U_{m,j_1-1} + f, \tag{2.13}$$
$$U_{1,j_1}(t^n) = U_1(t^n), j_1 = 1, \ldots, J_1,$$
$$\frac{dU_{2,j_2}}{dt} = A_{2,1}U_{1,j_2-1} + A_{2,2}U_{1,j_2} + \ldots + A_{2,m}U_{m,j_2-1} \tag{2.14}$$
$$+B_{2,1}U_{1,j_2-1} + B_{2,2}U_{1,j_2-1} + \ldots + B_{2,m}U_{m,j_2-1} + f, \tag{2.15}$$
$$U_{2,j_2}(t^n) = U_2(t^n), j_2 = 1, \ldots, J_2,$$
$$\ldots$$
$$\frac{dU_{m,j_m}}{dt} = A_{m,1}U_{1,j_m-1} + A_{m,2}U_{1,j_m-1} + \ldots + +A_{m,m}U_{m,j_m} \tag{2.16}$$
$$+B_{m,1}U_{1,j_m-1} + B_{m,2}U_{1,j_m-1} + \ldots + B_{m,m}U_{m,j_m-1} + f,$$
$$U_{m,j_m}(t^n) = U_m(t^n), j_m = 1, \ldots, J_m,$$

where $U_i(t^{n+1}) = (U_{1,J_1}(t^{n+1}), \ldots, U_{m,J_m}(t^{n+1}))$ is the result to the next iterated step i
and $U_j(t^n) = (U_1(t^n), \ldots, U_m(t^n)$ is the initial solution.
$(U_{1,j_1-1}(t^{n+1}), \ldots, U_{m,j_m-1}(t)) = U_{i-1}(t)$ is the approximated starting solution to $i-1$ of the outer iterative step.

We can also iterate much more finer, if we also include operator B into the iterative scheme.

Unifying Analysis:

Application of an alternative waveform-relaxation scheme.

1.) Convergence of the inner iterations

2.) Convergence of the outer iterations

The analysis is based on the convergence results of each inner iteration scheme, that is equal or one order higher than the outer iteration scheme.

If all inner schemes converge and are at least of the same order than the outer schemes, then the full iterative scheme is convergent.

THEOREM 2.1. *Let us consider the abstract Cauchy problem in a Banach space* **X**

$$\frac{dU}{dt} = \mathcal{A}U + \mathcal{B}U + F, \tag{2.17}$$
$$U(t^n) = U_n, \tag{2.18}$$

where $U(t^{n+1}) = (u_1(t^{n+1}), \ldots, u_m(t^{n+1}))$ is the solution at time t^{n+1} and m is the number of ODE's, further the matrices are given in (2.11) and (2.12) and F is a right hand side (e.g. source term).

Then the iteration process (2.8) is convergent and the and the rate of the convergence is of higher order, depending of the iterative step i.

Proof.

The outer iterative process is given with the convergence:

$$\|E_i(t)\| \leq K\tau_n\|E_{i-1}(t)\|, \tag{2.19}$$

while E_i is i-th error $E_i(t) := U(t) - U_i(t)$ and $U(t)$ is the given exact solution of the ODE.

The proof is given in [5].

Further the inner iterative process is given with the convergence:

$$\|E_i(t)\| \leq K\tau_n \|E_{\tilde{j}}(t)\|, \tag{2.20}$$

while E_i is i-th error $E_i(t) := U(t) - U_i(t)$ and $U(t)$ is the given exact solution of the ODE. $\tilde{j} = \min\{J_1, \ldots, J_m\}$.

So we need at least the minimum of 1 iterative steps over each single equation to gain at least one order of accuracy for the full system.
□

2.4. Application to the Heat-transfer and Heat-flow equations. We have the following equation schemes:

$$\begin{aligned}
\partial_t T_i &= \nabla \cdot (K \nabla T_i) - \nabla \cdot \mathbf{v}_{i-1} T_{i-1}, && (2.21)\\
\partial_t \mathbf{v}_i &= -(\mathbf{v}_{i-1} \cdot \nabla)\mathbf{v}_i - \nabla p(T_{i-1}), && (2.22)\\
T_i(\mathbf{x}, t^n) &= T(\mathbf{x}, \mathbf{t^n}),\\
\mathbf{v}_i(\mathbf{x}, t^n) &= \mathbf{v}(\mathbf{x}, t^n),\\
&\quad \text{for } t \in [t^n, t^{n+1}], n = 0, 1, 2, \ldots, N, \text{ with } i = 1, 2, \ldots, I,
\end{aligned}$$

where the initialization (starting condition for $i = 0$) is $T_0(\mathbf{x}, t) = T(\mathbf{x}, \mathbf{t^n})$ and $\mathbf{v}_0(\mathbf{x}, t) = \mathbf{v}(\mathbf{x}, \mathbf{t^n})$ with $t \in [t^n, t^{n+1}]$, means the solution at the last time-point.

2.4.1. Nonlinear Case: Modified Jacobian-Newton Methods and Fixpoint-iteration Methods. We describe the modified Jacobian-Newton methods and Fixpoint-iteration methods.

We propose for weak nonlinearities, e.g. quadratic nonlinearity, the fixpoint iteration method, where our iterative operator splitting method is one, see [7]. For stronger nonlinearities, e.g. cubic or higher order polynomial nonlinearities, the modified Jacobian method with embedded iterative-splitting methods is suggested.

The point of embedding the splitting methods into the Newton methods is to decouple the equation systems into simpler equations. Such simple equation systems can be solved with scalar Newton methods.

The altered Jacobian-Newton iterative methods with embedded sequential splitting methods

We confine our attention to time-dependent partial differential equations of the form

$$\frac{dc}{dt} = A(c(t))c(t) + B(c(t))c(t), \text{with } c(t^n) = c^n, \tag{2.23}$$

where $A(c), B(c) : \mathbf{X} \to \mathbf{X}$ are linear and densely defined in the real Banach space $\mathbf{X}$, involving only spatial derivatives of c, see [15]. We assume also that we have a weak nonlinear operator with $A(c)c = \lambda_1 c$ and $B(c)c = \lambda_2 c$, where λ_1 and λ_2 are constant factors.

In the following we discuss the embedding of a sequential splitting method into the Newton method.

The altered Jacobian-Newton iterative method with an embedded iterative splitting method is given as:

Newton's method:

$F(c) = \frac{dc}{dt} - A(c(t))c(t) - B(c(t))c(t)$ and we can compute $c^{(k+1)} = c^{(k)} - D(F(c^{(k)}))^{-1}F(c^{(k)})$, where $D(F(c))$ is the Jacobian matrix and $k = 0, 1, \ldots$.

We stop the iterations when we obtain : $|c^{(k+1)} - c^{(k)}| \leq err$, where err is an error bound, e.g. $err = 10^{-4}$.

We assume the spatial discretization, with spatial grid points, $i = 1, \ldots, m$ and obtain the differential equation system:

$$F(c) = \begin{pmatrix} F(c_1) \\ F(c_2) \\ \vdots \\ F(c_m) \end{pmatrix}, \tag{2.24}$$

where $c = (c_1, \ldots, c_m)T$ and m is the number of spatial grid points.

The Jacobian matrix for the equation system is given as :

$$DF(c) = \begin{pmatrix} \frac{\partial F(c_1)}{c_1} & \frac{\partial F(c_1)}{c_2} & \cdots & \frac{\partial F(c_1)}{c_m} \\ \frac{\partial F(c_2)}{c_1} & \frac{\partial F(c_2)}{c_2} & \cdots & \frac{\partial F(c_2)}{c_m} \\ \vdots & & & \\ \frac{\partial F(c_m)}{c_1} & \frac{\partial F(c_m)}{c_2} & \cdots & \frac{\partial F(c_m)}{c_m} \end{pmatrix},$$

where $c = (c_1, \ldots, c_m)$.

The modified Jacobian is given as:

$$DF(c) = \begin{pmatrix} \frac{\partial F(c_1)}{c_1} + F(c_1) & \frac{\partial F(c_1)}{c_2} & \cdots & \frac{\partial F(c_1)}{c_m} \\ \frac{\partial F(c_2)}{c_1} & \frac{\partial F(c_2)}{c_2}F(c_2) & \cdots & \frac{\partial F(c_2)}{c_m} \\ \vdots & & & \\ \frac{\partial F(c_m)}{c_1} & \frac{\partial F(c_m)}{c_2} & \cdots & \frac{\partial F(c_m)}{c_m} + F(c_m) \end{pmatrix},$$

where $c = (c_1, \ldots, c_n)$.

By embedding the sequential splitting method we obtain the following algorithm. We decouple into two equation systems:

$$F_1(u_1) = \partial_t u_1 - A(u_1)u_1 = 0 \quad \text{with} \quad u_1(t^n) = c^n, \tag{2.25}$$

$$F_2(u_2) = \partial_t u_2 - B(u_2)u_2 = 0 \quad \text{with} \quad u_2(t^n) = u_1(t^{n+1}), \tag{2.26}$$

where the results of the methods are $c(t^{n+1}) = u_2(t^{n+1})$. and $u_1 = (u_{11}, \ldots, u_{1n})$, $u_2 = (u_{21}, \ldots, u_{2n})$.

Thus we have to solve two Newton methods, each in one equations system. The contribution is to reduce the Jacobian matrix into a diagonal entry, e.g. with a weighted Newton method, see [9]. The splitting method with embedded Newton method is given as:

ALGORITHM 2.1. *We assume the spatial operators A and B are discretized, e.g. finite difference or finite element methods; further all initial conditions and boundary*

conditions are discrete given. Then we can apply the Newton's method in its discrete form as:

$$u_1^{(k+1)} = u_1^{(k)} - D(F_1(u_1^{(k)}))^{-1}(\partial_t u_1^{(k)} - A(u_1^{(k)})u_1^{(k)}), \tag{2.27}$$

$$\text{with } D(F_1(u_1^{(k)})) = \frac{\partial}{\partial u_1^{(k)}}(\partial_t u_1^{(k)} - A(u_1^{(k)}) - \frac{\partial A(u_1^{(k)})}{\partial u_1^{(k)}}u_1^{(k)}), \tag{2.28}$$

$$u_1^{(k)}(t^n) = c^n \text{ and } k = 0, 1, 2, \ldots, K, \tag{2.29}$$

$$u_2^{(l+1)} = u_2^{(l)} - D(F_2(u_2^{(l)}))^{-1}(\partial_t u_2^{(l)} - B(u_2^{(l)})u_2^{(l)}), \tag{2.30}$$

$$\text{with } D(F_2(u_2^{(l)})) = \frac{\partial}{\partial u_1^{(k)}}(\partial_t u_2^{(k)} - B(u_2^{(l)}) - \frac{\partial B(u_2^{(l)})}{\partial u_2^{(l)}}u_2^{(l)}), \tag{2.31}$$

$$u_2^{(l)}(t^n) = u_1^K(t^{n+1}) \text{ and } l = 0, 1, 2, \ldots, L. \tag{2.32}$$

where k and l are the iteration indices, K and L the maximal iterative steps for each part of the Newton's method. The maximal iterative steps allow us to have at least an error of: $|u_1^{(K)(t^{n+1})} - u_1^{(K-1)(t^{n+1})}| \leq err$,

and

$|u_2^{(L)(t^{n+1})} - u_2^{(L-1)(t^{n+1})}| \leq err$,

where err is the error bound, e.g. $err = 10^{-6}$.

The approximated solution is given as :

$u(t^{n+1}) = u_2^{(L)(t^{n+1})}$.

For the improvement method, we can apply the weighted Newton method. We try to skip the delicate outer diagonals in the Jacobian matrix and apply:

$$u_1^{(k+1)} = u_1^{(k)} - (D(F_1(u_1^{(k)})) + \delta_1(u_1^{(k)}))^{-1}(F_1(u_1^{(k)}) + \epsilon\, u_1^{(k)}), \tag{2.33}$$

where the function δ can be applied as a scalar, e.g. $\delta = 10^{-6}$, and the same with ϵ. It is important to ensure that δ is small enough to preserve the convergence.

REMARK 2.2. *If we assume that we discretize the equation (2.25) and (2.26) with the backward-Euler method, e.g.:*

$$F_1(u_1(t^{n+1})) = u_1(t^{n+1}) - u_1(t^n) - \Delta t A(u_1(t^{n+1}))u_1(t^{n+1}) = 0 \quad \text{with} \quad u_1(t^n) = c^n,$$
$$F_2(u_2) = u_2(t^{n+1}) - u_2(t^n) - \Delta t B(u_2(t^{n+1}))u_2(t^{n+1}) = 0 \quad \text{with} \quad u_2(t^n) = u_1(t^{n+1}),$$

then we obtain the derivations $D(F_1(u_1(t^{n+1})))$ and $D(F_2(u_2(t^{n+1})))$

$$D(F_1(u_1(t^{n+1}))) = 1 - \Delta t(A(u_1(t^{n+1})) + \frac{\partial A(u_1(t^{n+1}))}{\partial u_1(t^{n+1})}u_1(t^{n+1})),$$
$$D(F_2(u_2)) = 1 - \Delta t(B(u_2(t^{n+1})) + \frac{\partial B(u_2(t^{n+1}))}{\partial u_2(t^{n+1})}u_2(t^{n+1})).$$

We can apply the equation (2.33) analogously $u_2^{(l+1)}$.

Iterative operator-splitting method as a fixpoint scheme

The iterative operator-splitting method is used as a fixpoint scheme to linearize the nonlinear operators, see [4] and [7].

We confine our attention to time-dependent partial differential equations of the form:

$$\frac{du}{dt} = A(u(t))u(t) + B(u(t))u(t), \text{with } u(t^n) = c^n, \tag{2.34}$$

where $A(u), B(u) : \mathbf{X} \to \mathbf{X}$ are linear and densely defined in the real Banach space $\mathbf{X}$, involving only spatial derivatives of c, see [15]. In the following we discuss the standard iterative operator-splitting methods as a fixpoint iteration method to linearize the operators.

We split our nonlinear differential equation (2.23) by applying:

$$\frac{du_i(t)}{dt} = A(u_{i-1}(t))u_i(t) \; + \; B(u_{i-1}(t))u_{i-1}(t), \text{ with } \; u_i(t^n) = c^n, \tag{2.35}$$

$$\frac{du_{i+1}(t)}{dt} = A(u_{i-1}(t))u_i(t) \; + \; B(u_{i-1}(t))u_{i+1}, \text{ with } \; u_{i+1}(t^n) = c^n, \tag{2.36}$$

where the time-step is $\tau = t^{n+1} - t^n$. The iterations are $i = 1, 3, \ldots, 2m+1$. $u_0(t) = c_n$ is the starting solution, where we assume the solution c^{n+1} is near c^n, or $u_0(t) = 0$. So we have to solve the local fixpoint problem. c^n is the known split approximation at the time level $t = t^n$.
The split approximation at time level $t = t^{n+1}$ is defined as $c^{n+1} = u_{2m+2}(t^{n+1})$. We assume the operators $A(u_{i-1}), B(u_{i-1}) : \mathbf{X} \to \mathbf{X}$ to be linear and densely defined on the real Banach space $\mathbf{X}$, for $i = 1, 3, \ldots, 2m + 1$.
Here the linearization is done with respect to the iterations, such that $A(u_{i-1}), B(u_{i-1})$ are at least non-dependent operators in the iterative equations, and we can apply the linear theory.
The linearization is at least in the first equation $A(u_{i-1}) \approx A(u_i)$, and in the second equation $B(u_{i-1}) \approx B(u_{i+1})$
We have

$$||A(u_{i-1}(t^{n+1}))u_i(t^{n+1}) - A(u^{n+1})u(t^{n+1})|| \leq \epsilon,$$

with sufficient iterations $i = \{1, 3, \ldots, 2m + 1\}$.

REMARK 2.3. *The linearization with the fixpoint scheme can be used for smooth or weak nonlinear operators, otherwise we lose the convergence behavior, while we did not converge to the local fixpoint, see [7].*

The second idea is based on the Newton method.

Jacobian-Newton iterative method with embedded operator-splitting method

The Newton method is used to solve the nonlinear parts of the iterative operator-splitting method (see the linearization techniques in [7],[8]).
Newton method:
The function is given as:
$F(c) = \frac{\partial c}{\partial t} - A(c(t))c(t) - B(c(t))c(t) = 0,$

The iteration can be computed as:
$c^{(k+1)} = c^{(k)} - D(F(c^{(k)}))^{-1}F(c^{(k)}),$
where $D(F(c))$ is the Jacobian matrix and $k = 0, 1, \ldots.$ and $c = (c_1, \ldots, c_m)$ is the solution vector of the spatial discretized nonlinear equation.
We then have to apply the iterative operator-splitting method and obtain:

$$F_1(u_i) = \partial_t u_i - A(u_i)u_i - B(u_{i-1})u_{i-1} = 0, \tag{2.37}$$

$$\text{with} \quad u_i(t^n) = c^n, \tag{2.38}$$

$$F_2(u_{i+2}) = \partial_t u_{i+1} - A(u_i)u_i - B(u_{i+1})u_{i+1} = 0, \tag{2.39}$$

$$\text{with} \quad u_{i+1}(t^n) = c^n, \tag{2.40}$$

where the time-step is $\tau = t^{n+1} - t^n$. The iterations are $i = 1, 3, \ldots, 2m+1$. $c_0(t) = 0$ is the starting solution and c^n is the known split approximation at the time-level $t = t^n$. The results of the methods are $c(t^{n+1}) = u_{2m+2}(t^{n+1})$.
Thus we have to solve two Newton methods and the contribution will be to reduce the Jacobian matrix, e.g. skip the diagonal entries. The splitting method with the embedded Newton method is given as:

$$
\begin{aligned}
& u_i^{(k+1)} = u_i^{(k)} - D(F_1(u_i^{(k)}))^{-1}(\partial_t u_i^{(k)} - A(u_i^{(k)})u_i^{(k)} - B(u_{i-1}^{(k)})u_{i-1}^{(k)}),\\
& \text{with } D(F_1(u_i^{(k)})) = -(A(u_i^{(k)}) + \frac{\partial A(u_i^{(k)})}{\partial u_i^{(k)}} u_i^{(k)}),\\
& \text{and } k = 0, 1, 2, \ldots, K,\\
& \text{with} \quad u_i(t^n) = c^n,
\end{aligned}
$$

$$
\begin{aligned}
& u_{i+1}^{(l+1)} = u_{i+1}^{(l)} - D(F_2(u_{i+1}^{(l)}))^{-1}(\partial_t u_{i+1}^{(l)} - A(u_i^{(k)})u_i^{(k)} - B(u_{i+1}^{(k)})u_{i+1}^{(k)})c_2^{(l)}),\\
& \text{with } D(F_2(u_{i+1}^{(l)})) = -(B(u_{i+1}^{(l)}) + \frac{\partial B(u_{i+1}^{(l)})}{\partial u_{i+1}^{(l)}} u_{i+1}^{(l)}),\\
& \text{and } l = 0, 1, 2, \ldots, L,\\
& \text{with} \quad u_{i+1}(t^n) = c^n,
\end{aligned}
$$

where the time-step is $\tau = t^{n+1} - t^n$. The iterations are: $i = 1, 3, \ldots, 2m+1$. $c_0(t) = 0$ is the starting solution and c^n is the known split approximation at the time-level $t = t^n$. The results of the methods are $c(t^{n+1}) = u_{2m+2}(t^{n+1})$.

For the improvement by skipping the delicate outer diagonals in the Jacobian matrix, we apply $u_i^{(k+1)} = u_i^{(k)} - (D(F_1(u_i^{(k)})) + \delta_1(u_i^{(k)}))^{-1}(F_1(u_i^{(k)}) + \epsilon\ u_i^{(k)})$, and analogously $u_{i+1}^{(l+1)}$.

REMARK 2.4. *For the iterative operator-splitting method with the Newton iteration we have two iteration procedures. The first iteration is the Newton method to compute the solution of the nonlinear equations, and the second iteration is the iterative splitting method, which computes the resulting solution of the coupled equation systems. The embedded method is used for strong nonlinearities.*

3. Parallelization : Parareal. To improve the parallel scaling properties of our algorithm, we employ the so-called parareal algorithm invented by Lions, Maday & Turincini [10]. This algorithm is a means of time-parallelization and can be viewed as a multiple shooting method.

We assume to have a partitioning of the time interval $\Omega = [0, T]$ divided into N subdomains:

$$\Omega_n = [T_{n-1}, T_n], \quad n = 1, 2, \ldots, N. \tag{3.1}$$

In parareal one makes use of two solvers, a coarse propagator $G(T_n, T_{n-1}, x)$ and a fine propagator $F(T_n, T_{n-1}, x)$, each of which compute coarse and fine approximations of the solution U_n of the equation

$$\frac{dU}{dt} = f(t, U(t)), \text{ with } U(T_{n-1}) = x. \tag{3.2}$$

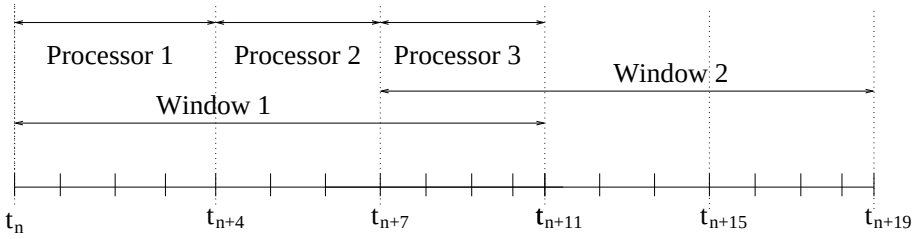

FIG. 3.1. *Parallelization with Parareal, windowing of the parallel process.*

It is crucial that the coarse integrator is computationally much faster than the fine integrator, the latter of which is more accurate. The first iteration of parareal uses the coarse integrator in a serial fashion to provide initial conditions to each time slice Ω_n:

$$U_n^1 = G(T_n, T_{n-1}, U_{n-1}^1), \quad n = 1, 2, \ldots, N.$$

After this initialization the fine propagator can be used to integrate independently (i.e., in parallel) N initial-value problems $F(T_n, T_{n-1}, U_{n-1}^k)$ $(n = 1, 2, \ldots, N)$, yielding new approximations for the initial conditions on the following time slices. In each iteration k the corrections are then again quickly propagated using the coarse integrator:

$$U_n^{k+1} = F(T_n, T_{n-1}, U_{n-1}^k) + G(T_n, T_{n-1}, U_{n-1}^{k+1}) - G(T_n, T_{n-1}, U_{n-1}^k), \quad (3.3)$$

EXAMPLE 3.1. *We assume to have F as the iterative splitting propagator and G as the A-B splitting propagator.*

Further the iterative splitting scheme include additionally a fixpoint scheme for nonlinear problems.

So we step by each window to the next time interval, see Figure 3.1.

4. Benchmark Problems. In the following, we deal with simple examples to test the convergence of parareal with embedded higher order splitting schemes.

4.1. Test 1: One-directional coupling. We start with a scalar test problem for the coupling between a heat equation with convection term and a fluid flow given by a convection equation:

$$\partial_t T = \nabla \cdot (K \nabla T) - \nabla \cdot vT, \quad (4.1)$$
$$\partial_t v = -(v \cdot \nabla)v - \nabla p, \quad (4.2)$$
$$T(x, t_0) = T_0(x), \quad (4.3)$$
$$v(x, t_0) = v_0(x), \quad (4.4)$$

where the unknown temperature is T, v is the flow field of the temperature and p is a given pressure. The spacial domain of this problem is the interval $[0, 1]$ with thermal conductivity $K = 0.01$, which we discretize by finite differences at $n + 1$, equidistant nodes $x_j = j/n$ $(j = 0, 1, \ldots, n,\ n = 100)$. We assume to have homogeneous Neumann conditions at the interval endpoints. This results in the following system

$$\partial_t \mathbf{T} = D_2 \mathbf{T} - D_1(\mathbf{T} \circ \mathbf{v}) + \mathbf{b} \quad (4.5)$$
$$\partial_t \mathbf{v} = -\mathbf{v} \circ D_1 \mathbf{v} - D_1 \mathbf{p}, \quad (4.6)$$

where the matrices D_1, D_2 discretize first and second order differential operators and $\mathbf{b}$ is a vector representing the boundary data. In a real-world problem the pressure p will depend on the temperature vector T, but in this test we assume that the pressure is given by the function

$$p(t,x) = \mathbf{1}_{[0,1/3]}(t) \cdot 3\text{e}3 \cdot tx^6(1-x)^6,$$

where $\mathbf{1}_{[0,1/3]}(t)$ is an indicator function for the time-interval $[0, 1/3]$. This function creates a transport wave (a flow field) traveling in both directions from the midpoint to the interval towards the interval endpoints. The initial flow field is taken as $v_0(x) = 0$ and for the initial temperature we have taken $T_0(x) = 16x^2(1-x)^2$. The whole problem is integrated over the time domain $\Omega_T = [0, 1]$, which we have divided into 10 subdomains of equal length.

In order to evaluate the accuracy and performance of the coarse and fine integrator over one time slice $[0, 0.1]$, we compare 3 different integrators for a general initial-value problem $u' = f(t, u)$, $u(0) = u_0$, namely

- the implicit Euler method of order 1 (Euler1),
- a classical explicit Runge–Kutta method of order 2 (RK2 or midpoint method),
- the backward differentiation formula of order 4 (BDF4).

Note that the equation for the temperature is semi-linear. For semi-linear initial-value problems $u' = Au + f(t, u)$, $u(0) = u_0$ we also test

- an exponential Runge–Kutta method of stiff order 2 (expRK2 or exponential midpoint method),

which treats the stiff linear term Au exactly and hence the stability condition is only possed by the mildly stiff nonlinear term. This method requires the evaluation of matrix exponentials $\exp(tA)$, or more precisely, the action of these exponentials onto vectors $\exp(tA)v$. We have used a rational Chebyshev method for these evaluations.

We apply the above integrators to each of the two differential equations for T and v, where we have used three different coupling techniques:

- A-B coupling, which is of order 1 and hence will only be tested in combination with Euler1,
- Strang coupling, which is of order 2 and hence will be tested in combination with the Runge–Kutta methods (RK2 and expRK2),
- iterative coupling in combination with the BDF4 integrator (note that in this example the coupling is only one-directional and hence there is need for exactly one splitting iteration).

In Figure 4.1 we show the accuracy of the tested combinations as a function of the number of time steps (top) and the number of function evaluations (below), respectively. In the number of function evaluations we have included for the implicit solvers (Euler1 and BDF4) the evaluations required by the Newton solver. We have used the m-file `nsoli.m`, a globally convergent Newton-Krylov solver given in [9], with an absolute and relative error tolerance tol proportional to tsteps^{-q}, written $\text{tol} \sim \text{tsteps}^{-q}$, where tsteps is the number of time steps and q is the order of the integrator. For BDF4 ($q = 4$) we have also included the function evaluations for the higher-accuracy initialization of the 4 initial time steps by expRK2 with a refined step-size h. Since expRK2 is only of order 2 we have chosen $h \sim \sqrt{\text{tol}}$. Note that the number of function evaluations for the second equation (velocity) with the explicit methods and Strang-splitting is twice as large as the number of function evaluations for the first equation (temperature), because in each time step we make 2 half-steps with v and 1 full-step with T. Note also that the convergence of RK2 for the temperature

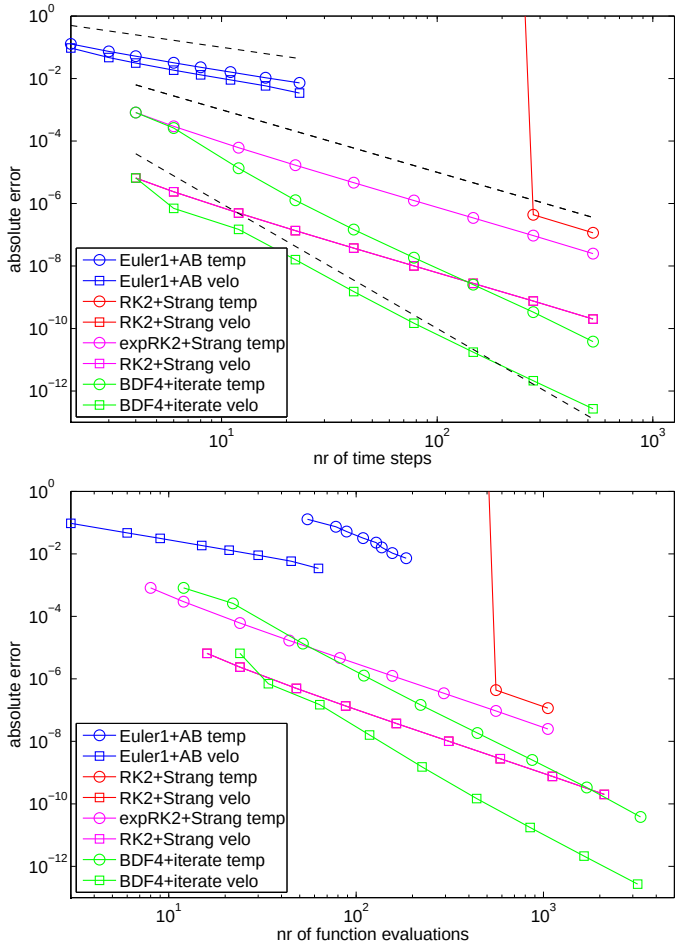

FIG. 4.1. *Convergence of combinations of different time-stepping schemes with various coupling techniques for integration of the model problem over a single time slice* $[0, 0.1]$. *The coupling of the two equations is one-directional here. The dashed lines on the left indicate convergence of orders* 1, 2, *and* 4.

equation is subject to stringent stability constraints and the convergence curve comes reasonable low only for a large number of time steps (or function evaluations). The expRK2 method does not have this stability constraints and shows order 2 convergence already for very small numbers of time steps (functions evaluations).

The fact that the favorable stability properties of expRK2 allow for larger step sizes makes this method in conjunction with Strang-coupling predestined as a coarse integrator for parareal. In order to achieve a high accuracy with few function evaluations we will use BDF4 with iterative coupling as the fine integrator.

The convergence of parareal is depicted in Figure 4.2. Each convergence curve corresponds to the error of the computed temperature T (left) and velocity v (right) measure at each of the 11 coarse time points $T_0, T_1, \ldots, T_N$, where the abscissae indicate the parareal iteration index k. Note that after $k = 11$ iterations the algorithm reaches the accuracy of the fine propagator, the result of which was also used as the reference solution for computing the error. We also note that the convergence for the

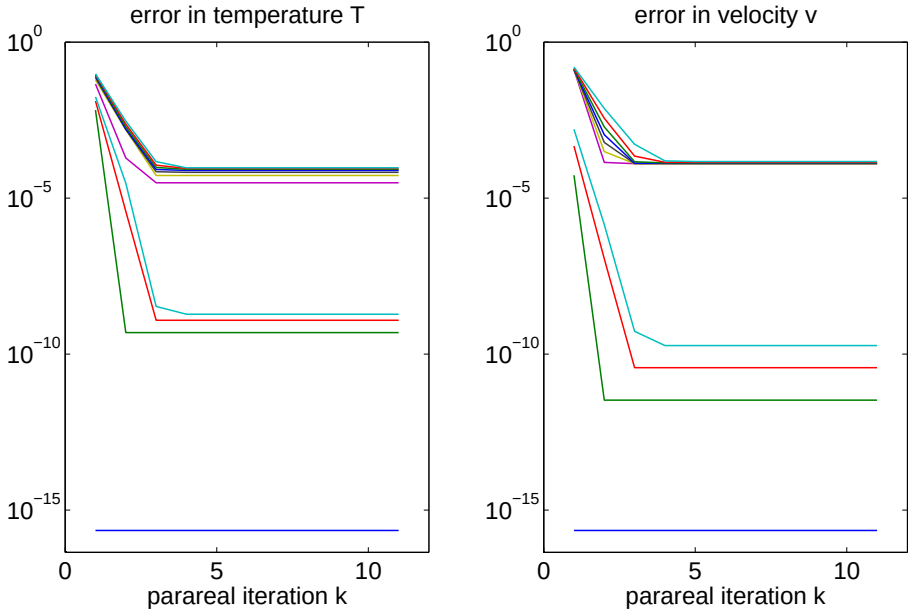

FIG. 4.2. *Convergence of parareal applied to the model problem. The left picture shows the error of the temperature variable at each of the* 11 *points of the time grid (the lowest constant curve corresponds to time* $t = 0$*, where the initial value is given) after* $k = 1, 2, \ldots$ *parareal iterations. On the right we show the error of the velocity variable.*

velocity field is much more rapid than for the temperature, see Figure 4.2.

4.2. Test 2: Bidirectional coupling. We consider the previous test problem but now with a pressure function depending on the temperature T,

$$p(T) = 0.2 \cdot T + 0.2.$$

Therefore we now have a bidirectional coupling between the equations (4.1) and (4.2). For the iterative splitting method this results in the system

$$\partial_t \mathbf{T}_i = D_2 \mathbf{T}_i - D_1(\mathbf{T}_i \circ \mathbf{v}_{i-1}) + \mathbf{b}\ , \mathbf{T}_i t^n = \mathbf{T}(t^n)\ ,\ t \in [t^n, t^{n+1}], \tag{4.7}$$

$$\partial_t \mathbf{v}_i = -\mathbf{v}_{i-1} \circ D_1 \mathbf{v}_i - D_1 \mathbf{p}(T_i)\ , \mathbf{v}_i t^n = \mathbf{v}(t^n)\ ,\ t \in [t^n, t^{n+1}], \tag{4.8}$$

where $i = 1, 2, \ldots, I$ and $I \in \mathbb{N}^+$ is a fixed number. Further we assume the starting value $v_0(t) = v(t^n)$ or an initial guess $v_0(t) = v_{initial}(t)$. The approximated solutions are given as $\mathbf{T}(t^{n+1}) = \mathbf{T}_I t^{n+1}$, $\mathbf{v}(t^{n+1}) = \mathbf{v}_I t^{n+1}$ with I iterative steps.

As in the previous test, we have used a Newton iteration `nsoli` for the implicit time-stepping methods and we have chosen the stopping tolerance tol depending the number of time-steps tsteps as tol $\sim$ tsteps^{-q}, as before. The achieved accuracy as a function of the number of time-steps is shown in the right plot of Figure 4.3.

It turns out that for every time-step only 2–3 iterative splitting-scheme iterations are required to satisfy the imposed error criterion. However, each call to the Newton solver `nsoli` requires 3–7 function calls for solving the implicit equation for the new temperature or velocity iterate to prescribed error tolerance. The average number of function calls per time-step for the temperature equation is therefore above 4, which results in a worse work-precision curve for the iterative splitting method when the

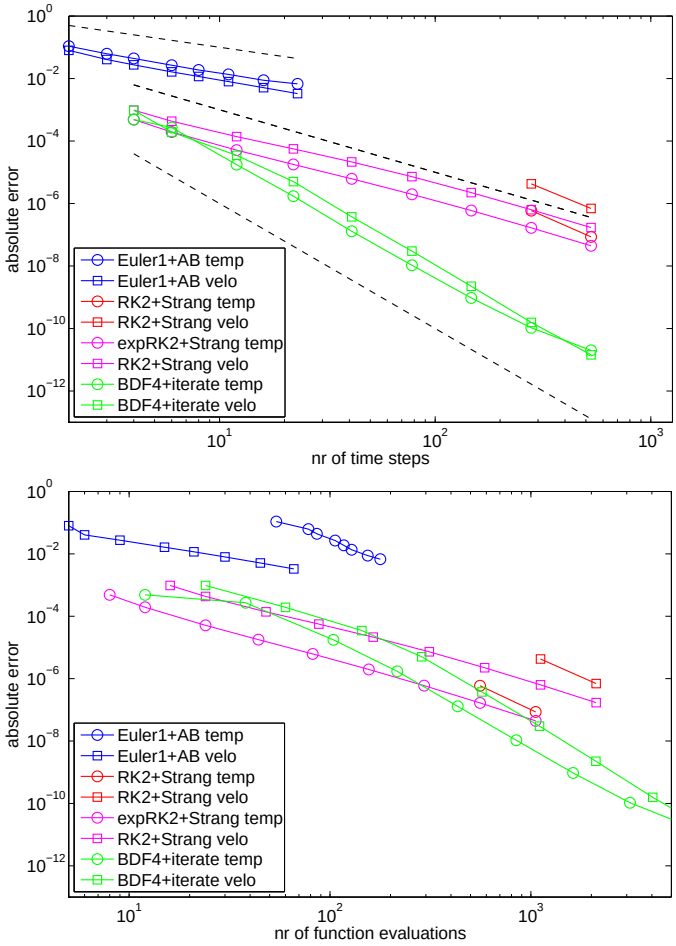

FIG. 4.3. *Convergence of combinations of different time-stepping schemes with various coupling techniques for integration of the model problem over a single time slice* $[0, 0.1]$. *The coupling of the two equations is bidirectional here. The dashed lines on the left indicate convergence of orders* 1, 2, *and* 4.

number of function evaluations is taken as measure (see the right plot of Figure 4.3). In this example, only for very high accuracies (below 10^{-7}, which is far below our spacial discretization error) we can expect that the iterative splitting approach outperforms the expRK2 scheme with Strang coupling.

The convergence of parareal for the temperature and velocity components is illustrated in Figure 4.4. As before we have partitioned the time interval $[0, 1]$ into 10 time-slices of equal length. Note that after about 8 iterations of parareal we reach the final stagnation level of the error, which means that we can expect a parallel speedup of 10/8, provided that the cost for the coarse integrator is neglectable. As coarse integrator we have used 1 time-step per time-slice of the expRK2 method with Strang coupling. The fine integrator is 200 time-steps of BDF4 with iterative coupling.

5. Conclusion. We have presented a model for the heat transport and flow field of a technical application in car body heating. We discuss an improvement of existing coupling methods with higher order splitting schemes. Numerical accerlerations are

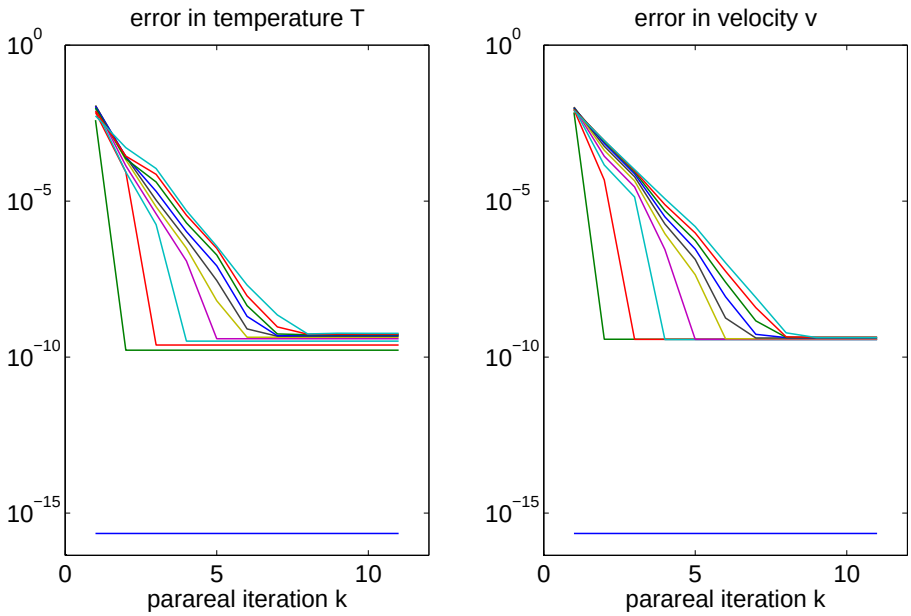

FIG. 4.4. *Convergence of parareal applied to the model problem. The left picture shows the error of the temperature variable at each of the* 11 *points of the time grid (the lowest constant curve corresponds to time* $t = 0$*, where the initial value is given) after* $k = 1, 2, \ldots$ *parareal iterations. On the right we show the error of the velocity variable.*

obtained with the time-parallelization with Parareal.

In numerical examples, we could show the efficiency of the higher order schemes embedded into Parareal, while producing more accurate solutions with larger time steps. Such accelerations helps to achieve realistic computational times.

6. Appendix. In the following, we present the construction of the BDF (Backward Differentiation Formula) and SBDF (Stiff Backward Differentiation Formula) methods.

6.1. Construction of the BDF methods. The BDF methods are one class of very effective methods for solving stiff problem. These methods are refereed to as Gear's methods since one of the first software packages for stiff problem is written by him although Curtis and Hirschfelder [**?**] first introduced the methods. These methods are of the form

$$\sum_{r=1}^{k+1} \alpha_r u^{n-r+2} = \tau \beta f(u^{n+1}). \tag{6.1}$$

where the coefficients α_r and β are obtained based on the Taylor expansion and difference operator. For example, for $k = 0$, $\alpha_1 = 1, \alpha_1 = -1, \beta = 1$, this corresponds to the classical implicit Euler method of order 1. For $k = 2$, the coefficients are $\alpha_1 = 3, \alpha_2 = -4, \alpha_3 = 1, \beta = 2$. This method is of order 3. We reconstruct the BDF for iterative splitting method as follows,

$$\sum_{r=1}^{k+1} \alpha_r u^{n-r+2} = \tau \beta A_1 u^{n+1} + \tau \beta A_2 u^{n+1}, \tag{6.2}$$

and the following conditions must satisfy:

- $k < 7$, otherwise the method is not zero stable,
- $\sum_{r=1}^{k+1} \alpha_r = 0$, otherwise the method is not consistent.

In the next section, we will construct stiff backward differentiation formulas (SBDF) with order k.

6.2. Construction of the SBDF methods. The implicit-explicit (IMEx) schemes have been widely for time integration of spatial discretization of the partial differential equations of diffusion-convection type. We propose SBDF methods as the one class of IMEx methods for iterative splitting. These methods are applied to decouple the implicit and explicit terms. So for example the convection-diffusion equation, one use the explicit part for the convection term and the implicit part for the diffusion. In our application we divide between the stiff and nonstiff term, so we apply the implicit part for the stiff operators and the explicit part for the nonstiff operators. We would like to design k+1-th order SBDF method as follows: We would like to design k-th order SBDF method as follows:

$$\sum_{r=1}^{k+1} \alpha_r u^{n-r+2} = \tau \beta_1 A_2 u^{n+1} + \tau \sum_{r=2}^{k+1} \beta_r A_1 u^{n-r+2}, \tag{6.3}$$

the following conditions must be satisfy:

1. $k < 7$, otherwise the method is not zero stable,
2. $\sum_{r=1}^{k+1} \alpha_r = 0$, otherwise the method is not consistent,
3. $\beta_1 = \sum_{r=2}^{k+1} \beta_r$, otherwise the method is not consistent.

In the Eq. (6.3), the parameters α_r, where $r = 1...., k+1$, are the same as BDFk method. One can rescale the the Eq. (6.3) by taking $\beta_1 = 1$, then unknown parameters β_r, where $r = 2...., k+1$, can be found by solving linear system of equation, obtained by the Taylor expansions of u^{n-r+2} around u^n.
For each k (order of SBDF method), we have the following $k \times k$ equation systems:

For $k = 2$, after solving the system

$$\begin{pmatrix} k & \binom{k}{k-2} \\ 0 & -\binom{k}{k-2} \end{pmatrix} \begin{pmatrix} t_1 \\ t_2 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix} \tag{6.4}$$

we have $\beta_2 = t_1 \binom{k}{k-1}$ and $\beta_3 = t_2 \binom{k}{k-2}$.

For $k = 3$, after solving the system

$$\begin{pmatrix} k & \binom{k}{k-2} & 1 \\ 0 & -\binom{k}{k-2} & -(k-1) \\ 0 & \binom{k}{k-2} & (k-1)^2 \end{pmatrix} \begin{pmatrix} t_1 \\ t_2 \\ t_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}. \tag{6.5}$$

we have $\beta_2 = t_1 \binom{k}{k-1}$, $\beta_3 = t_2 \binom{k}{k-2}$ and $\beta_4 = t_3 \binom{k}{k-3}$. We can solve for $k = 4, 5, 6$ in the same way.

These linear systems can be written in a closed form as follows:

$$\sum_{s=1}^{k} \left(\begin{array}{c} k \\ s \end{array} \right) (-1)^{r+1} t_s = 1 \text{ for } \ r = 1 \tag{6.6}$$

$$\sum_{s=2}^{k} \left(\begin{array}{c} k \\ s \end{array} \right) (-1)^{r+1} t_s (s-1)^{r-1} = 1 \text{ for } \ r = 2, \ldots, k \tag{6.7}$$

where $\beta_r = t_{r-1} \left(\begin{array}{c} k \\ r-1 \end{array} \right)$.

As a result we have general form of SBDFk methods as follows:

For k is even:

$$\sum_{r=1}^{k+1} \alpha_r u^{n-r+2} = \tau A_2 u^{n+1} + \tau \sum_{r=2}^{k+1} (-1)^{k+r-1} \left(\begin{array}{c} k \\ r-1 \end{array} \right) A_1 u^{n-r+2} \tag{6.8}$$

For k is odd:

$$\sum_{r=1}^{k+1} \alpha_r u^{n-r+2} = \tau A_2 u^{n+1} + \tau \sum_{r=2}^{k+1} (-1)^{k+r-2} \left(\begin{array}{c} k \\ r-1 \end{array} \right) A_1 u^{n-r+2} \tag{6.9}$$

where k is the order of the SBDF method. The coefficients for SBDF method is presented in Table 1.

order	α_7	α_6	α_5	α_4	α_3	α_2	α_1	β_1	β_2	β_3	β_4	β_5	β_6	β_7
1						-1	1	1	1					
2					1/2	-2	3/2	1	2	-1				
3				-1/3	3/2	-3	11/6	1	3	-3	1			
4			1/4	-4/3	3	-4	25/12	1	4	-6	4	-1		
5		-1/5	5/4	-10/3	5	-5	137/60	1	5	-10	10	-5	1	
6	1/6	-6/5	15/4	-20/3	15/2	-6	49/20	1	6	-15	20	-15	6	1

TABLE 6.1
SBDF coefficients

For example, 2nd order SBDF (SBDF2) reads as

$$(3u^{n+1} - 4u^n + u^{n-1}) = 4\tau A_1 u^n - 2\tau A_1 u^{n-1} + 2\tau A_2 u^{n+1}, \tag{6.10}$$

and 3rd order SBDF (SBDF3) reads as

$$\begin{aligned} (11u^{n+1} - 18u^n + 9u^{n-1} - 2u^{n-2}) = {} & 18\tau A_1 u^n - 18\tau A_1 u^{n-1} \\ & + 6\tau A_1 u^{n-2} + 6\tau A_2 u^{n+1}. \end{aligned} \tag{6.11}$$

REFERENCES

[1] Z. Chen, G. Ji. *Sharp L^1 a posteriori error analysis for nonlinear convection-diffusion problems.* to appear, 2004.

[2] S. Descombes. *Convergence of a splitting method of high order for reaction-diffusion systems.* Mathematics of Computations, vol. 70, 1481–1501, 2001.

[3] I. Farago and J. Geiser. *Iterative operator-splitting methods for linear problems.* International Journal of Computational Science and Engineering , 3(4): 255–263, 2007.

[4] J. Geiser. *Iterative Operator-Splitting Methods with higher order Time-Integration Methods and Applications for Parabolic Partial Differential Equations.* Journal of Computational and Applied Mathematics, Elsevier, accepted, June 2007.

[5] J. Geiser. *Iterative Operator-Splitting Methods with higher order Time-Integration Methods and Applications for Parabolic Partial Differential Equations.* Journal of Computational and Applied Mathematics, Elsevier, Amsterdam, The Netherlands, 217, 227–242, 2008.

[6] R. Herbin, M. Ohlberger. *A posteriori error estimate for finite volume approximations of convection diffusion problems.* Proceedings of the Third International Symposium on: FINITE VOLUMES FOR COMPLEX APPLICATIONS - PROBLEMS AND PERSPECTIVES, Porquerolles (2002), 753–760. Hermes Penton Ltd, London, 2002.

[7] J. Kanney, C. Miller, and C.T. Kelley, Convergence of iterative split-operator approaches for approximating nonlinear reactive transport problems, Advances in Water Resources 26 (2003) 247-261.

[8] K.H. Karlsen and N. Risebro. *An Operator Splitting method for nonlinear convection-diffusion equation.* Numer. Math., 77, 3 , 365–382, 1997.

[9] C.T. Kelly. *Iterative Methods for Linear and Nonlinear Equations.* Frontiers in Applied Mathematics, SIAM, Philadelphia, PA, 1995.

[10] J. L. Lions, Y. Maday, G. Turincini. *A "parareal" in time discretization of PDEs.* C. R. Acad. Sci. Paris Sér. I Math., 332, 661–668, 2001.

[11] M. Ohlberger. *A posteriori error estimates for vertex centered finite volume approximations of convection-diffusion-reaction equations.* M2AN Math. Model. Numer. Anal. 35 (2001) 2, 355–387.

[12] J. Salcedo Rulz and F.J. Sanchez Bernabe. *A Numerical Study of Stiffness Effects on some Higher Order Splitting Methods.* Revista Mexicana de Fisica, vol. 52, no. 2, 129–134, 2006.

[13] G. Strang. *On the construction and comparision of difference schemes.* SIAM J. Numer. Anal., 5:506–517, 1968.

[14] Ricardo Software. *VECTIS, three-dimensional fluid dynamics program.* http://www.ricardo.com/What-we-do/Software/Products/VECTIS/

[15] E. Zeidler. Nonlinear Functional Analysis and its Applications. II/B Nonlinear montone operators Springer-Verlag, Berlin-Heidelberg-New York, 1990.