Quasi-Newton waveform iteration for partitioned surface-coupled multiphysics applications

Benjamin Rüth | Benjamin Uekermann | Miriam Mehl | Philipp Birken | Azahar Monge | Hans-Joachim Bungartz

1Department of Informatics, Technical University of Munich, Garching, Germany
2Department of Mechanical Engineering, Eindhoven University of Technology, Eindhoven, The Netherlands
3Institute for Parallel and Distributed Systems, University of Stuttgart, Stuttgart, Germany
4Centre for the Mathematical Sciences, Lund University, Lund, Sweden

Correspondence
Benjamin Rüth, Department of Informatics, Technical University of Munich, Boltzmannstr. 3, Garching 85748, Germany.
Email: rueth@in.tum.de

Funding information
Deutsche Forschungsgemeinschaft, Grant/Award Number: SPP1648; H2020 Marie Skłodowska-Curie Actions, Grant/Award Number: 754462

Summary
We present novel coupling schemes for partitioned multiphysics simulation that combine four important aspects for strongly coupled problems: implicit coupling per time step, fast and robust acceleration of the corresponding iterative coupling, support for multirate time stepping, and higher-order convergence in time. To achieve this, we combine waveform relaxation—a known method to achieve higher-order in applications with split time stepping based on continuous representations of coupling variables in time—with interface quasi-Newton coupling, which has been developed throughout the last decade and is generally accepted as a very robust iterative coupling method even for gluing together black-box simulation codes. We show convergence results (in terms of convergence of the iterative solver and in terms of approximation order in time) for two academic testcases—a heat transfer scenario and a fluid-structure interaction simulation. We show that we achieve the expected approximation order and that our iterative method is competitive in terms of iteration counts with those designed for simpler first-order-in-time coupling.

KEYWORDS
conjugate heat transfer, fluid-structure interaction, higher-order, multirate, multiscale, multiphysics, quasi-Newton, waveform iteration

1 INTRODUCTION

We consider time-dependent surface-coupled multiphysics applications. Examples are numerous: flutter of airplanes or wind turbines, blood flow in arteries, gas quenching or cooling of rocket engines, to only name a few. We follow a partitioned coupling approach, which abides the natural wish to reuse established single-physics solvers. Hereby, our goal is to have highest flexibility, while still having a fast time-to-solution in a black-box coupling framework. Only minimal information from the solvers is used in that only boundary conditions at the interface are exchanged, as well as geometric interface information, that can be used to map nonmatching interface meshes.

In this article, we focus on the time integration. The combination of different physics oftentimes implies different time scales. Thus, we would like to be able to use different time steps in the involved solvers. In addition, higher-order time integration is required to efficiently produce sufficiently accurate results. An open problem is to design stable, efficient and...
high-order methods that allow for different time steps in the separate solvers\textsuperscript{1} and for the use of black-box solvers. A class of methods that potentially have these properties is waveform relaxations, also called waveform or dynamic iterations.\textsuperscript{2,3} These integrate over a time window, allowing the subsolvers to use potentially arbitrary methods and time steps in that window, in the spirit of the partitioned approach. To this end, continuous interpolations in time are employed to provide solution values to the other problem. When the time integration over the window is repeated iteratively, high-order can be obtained by sufficiently high order in the subsolvers, as well as the interpolation. An efficient method is obtained, when this iteration converges fast.

The standard option is to add a relaxation step, which is why the terms waveform relaxation and waveform iteration (WI) are used synonymously in most literature. Waveform relaxation was originally introduced for efficient analysis of large digital circuits where components have to be resolved on different timescales.\textsuperscript{4-7} Today it is widely applied in field/circuit coupling.\textsuperscript{8-10} Recently, Neumann-Neumann waveform relaxation was suggested\textsuperscript{11} and applied to conjugate heat transfer problems.\textsuperscript{12,13} Optimal relaxation parameters are determined for the fully discrete version of this specific problem, leading to superlinear convergence. This was later extended to the time adaptive case.\textsuperscript{14} The optimal relaxation parameter is, however, problem dependent. This means that when applied as a black-box method, many iterations are needed.

Different alternatives have been suggested to speed up waveform relaxation, for example, convolution waveform methods,\textsuperscript{15} Krylov methods\textsuperscript{16} and multigrid.\textsuperscript{17} Another alternative are waveform relaxation Newton methods.\textsuperscript{6} There, the Newton method is formulated on the initial value problems using Fréchet derivatives. All of these offer interesting theoretical properties, which are, however, hard to exploit in practice.

Another idea that is more straightforward to exploit in practice is to use a quasi-Newton method instead of relaxation to improve the convergence of WIs. This has been very successfully applied for nonwaveform partitioned black-box FSI. The basic variant is called interface quasi-Newton inverse least-squares (IQN-ILS).\textsuperscript{18,19} As a nonlinear solver, the method is also known as Anderson acceleration.\textsuperscript{20,21} Important recent improvements have been parallel execution of solvers,\textsuperscript{22} filtering,\textsuperscript{23} reducing tuning parameters\textsuperscript{24-26} and the generalization to multiple coupled solvers.\textsuperscript{27} The resulting methods lead to very few coupling iterations, are robust without tuning parameters, and have linear compute and memory complexity in terms of interface degrees of freedom due to matrix-free formulations and due to a mesh independent convergence rate. Implicit coupling together with state-of-the-art IQN methods overcome added mass instability and convergence issues and can therefore also be applied for biomechanical applications.\textsuperscript{28,29}

In this article, we combine WI with IQN, with the goal of getting a novel fast partitioned black-box coupling method that still has high order and allows for different time step sizes in different solvers. We use the idea of time windows similar to References 2, 3, that is, we perform coupling iterations not within single time steps but across all time steps in a given time window. We consider versions of the quasi-Newton method that differ in terms of which subsets of time step data in the so-called coupling windows are used to approximate the system Jacobian. A similar approach has been followed in Reference 30 for a less general setting. There, subcycling, that is, using several time steps of one solver within one time step of the other one, is considered. For a 1D problem and certain time step ratios, the stability of the time integration is analyzed, which is confirmed by numerical results in two dimensions (2D). Our approach allows different numbers of time steps of both solvers within a so-called coupling window such that the time steps of the involved solvers no longer have to be multiples of each other (called multirate time stepping).

For the description and the numerical experiments, we only consider so-called Dirichlet-Neumann type coupling methods. These are a basic building block in FSI that only use boundary conditions that are available in the subsolvers anyhow. To this end, Dirichlet data or Neumann data, respectively, at the interface are exchanged between the solvers. A generalization for other coupling conditions is possible, but requires additional numerical testcases to evaluate the resulting performance. We do not adapt time steps dynamically in time, but we would like to point out that time adaptivity is straightforward to use in WIs. This is important, since it can give dramatic performance improvements.\textsuperscript{31,32}

We demonstrate up to third order convergence in time for the combined method. The iteration numbers do not vary much when varying the time step ratios and are rather small. We test the proposed algorithms by using the coupling library preCICE\textsuperscript{33} with the finite element frameworks FEniCS\textsuperscript{34} and Nutils.\textsuperscript{35} We, therefore, use IQN from preCICE and implement WI in a new middle layer between preCICE and the coupled solvers.

The structure of the article is as follows: In Section 2, we explain the mathematical framework for both concepts—WI for multirate coupled time stepping on the one hand and quasi-Newton on the other hand—separately and then show how we can combine them. We then describe in Section 3 how we realized a prototype implementation in a middle software layer between coupled solvers and preCICE. In Section 4, we show numerical results for a coupled heat equation and an FSI scenario, both in 2D. We conclude in Section 5.
2 COUPLING ALGORITHMS

Before introducing the implementation in the coupling library preCICE, we present the details of our coupling approach allowing for multirate time stepping, higher-order accuracy in time, and fast iterative convergence in several steps: Section 2.1 introduces our notation and the general form of the coupling formulation as used in standard low-order approaches. Section 2.2 explains how to extend this basic coupling with a waveform approach, which includes information from individual time steps to increase the approximation order. Section 2.3 explains how to construct a quasi-Newton scheme for a general setting. Section 2.4 combines both concepts to define quasi-Newton variants for multirate higher-order coupling.

2.1 Basic notation and multirate coupling via single value coupling

We consider a coupled problem on two nonoverlapping domains \(\Omega_D\) and \(\Omega_N\), discretized in space. We use a partitioned approach, that is, we rely on stand-alone solvers for each domain, which we regard as black boxes providing access only to input and output data, but not to discretization details. The coupling of solvers happens at the coupling interface \(\Gamma = \Omega_D \cap \Omega_N\). We couple the solvers via Dirichlet-Neumann coupling. The Dirichlet solver on \(\Omega_D\) takes a Dirichlet boundary condition at the coupling interface \(\Gamma\) from the Neumann solver on \(\Omega_N\), while the Neumann solver takes a Neumann boundary condition at the coupling interface \(\Gamma\) from the Dirichlet solver. For FSI, the natural choice is to use the fluid solver as Dirichlet solver with Dirichlet boundary values being velocities or displacements, and the solid solver as Neumann solver with Neumann boundary values being forces or stresses (see Section 4.2). For other coupled problems such as heat transfer, which we use as a second application testcase in Section 4.1, there is no natural choice, but the roles can be chosen arbitrarily. Still, there is typically a numerically better choice, which, for instance, can depend on the material parameters of the two media. Here, Dirichlet boundary values represent temperature values at the coupling interface and Neumann boundary values the heat flux computed from the internal temperature field.

Both solvers perform time stepping (explicit or implicit) using their respective schemes in a given time window \([t_{ini}, t_{ini} + \Delta t]\) with initial time \(t_{ini}\) and window size \(\Delta t\). Within each window, we assume them to use a fixed, but possibly different time step size. The Dirichlet and the Neumann solver, thus, perform \(n_D\) or \(n_N\) time steps of size \(\Delta t_D\) and \(\Delta t_N\), respectively \((\Delta t = n_D \Delta t_D = n_N \Delta t_N)\). We denote the action of a single time step by

\[
\begin{align*}
    u_{D,i+1} & = \hat{D} \left( c_{D,i+1}, u_{D,i}, \delta t_D \right) \quad \text{and} \quad u_{N,i+1} = \hat{N} \left( c_{N,i+1}, u_{N,i}, \delta t_N \right),
\end{align*}
\]

where \(u_D\) and \(u_N\) are the internal variables and \(c_D\) and \(c_N\) the coupling variables of both solvers. Please note that all variables are discretized in space, meaning they are vectors with (arbitrary) degrees of freedom as entries. Index \(i\) denotes the time level \(t_{ini} + i \cdot \Delta t_D\) or \(t_{ini} + i \cdot \Delta t_N\) within one window. We now define simplified operators,

\[
\begin{align*}
    D_{SC} : c_{D_{end}} \mapsto c_{N_{end}} \quad \text{and} \quad N_{SC} : c_{N_{end}} \mapsto c_{D_{end}},
\end{align*}
\]

as the action of our black-box solvers over the complete window if they use only the values at the end of the coupling window—\(c_{D_{end}}\) or \(c_{N_{end}}\), respectively—as input and output values: Given Dirichlet coupling values \(c_{D_{end}}\), the Dirichlet solver computes \(n_D\) time steps with \(c_{D,i+1} \equiv c_{D_{end}}\) for all \(i = 0, \ldots, n_D - 1\), followed by a postprocessing step returning Neumann coupling values at the end of the coupling window, \(c_{N_{end}} = \phi_D(u_{D,n_D})\). If necessary, the postprocessing also includes a data mapping operation between the discretizations of both solvers. Afterward, the Neumann solver computes \(n_N\) time steps with \(c_{N,i+1} \equiv c_{N_{end}}\) for all \(i = 0, \ldots, n_N - 1\), followed by \(c_{D_{end}} = \phi_N(u_{N,n_N})\). We introduce the simplified notation to express the coupling condition at the interface by means of the fixed-point equation

\[
c_{D_{end}} = N_{SC} \circ D_{SC}(c_{D_{end}})
\]

acting only on the Dirichlet interface data \(c_{D_{end}}\) at the end of the window. We refer to coupling methods solving Equation (2) in this contribution as single value coupling and denote them as SC\((n_D, n_N)\). Furthermore, the execution of \(n_D\) and \(n_N\) time steps in the solvers before coupling at the end of time window is called multirate time stepping. The special case with either \(n_D = 1, n_N \neq 1\) or \(n_D \neq 1, n_N = 1\) is commonly called subcycling.
Equation (2) entails a Gauss-Seidel-type coupling, executing $D_{SC}$ and $N_{SC}$ one after the other, while the roles of $D_{SC}$ and $N_{SC}$ could also be swapped leading then to a fixed-point equation in terms of $c_{N_{end}}$. Furthermore, we could instead also consider a Jacobi-type coupling. For the sake of clarity, we omit both additional variants in this contribution. To achieve a stable coupling also for strongly coupled problems and to increase accuracy, we consider an implicit (or strong) coupling, that is, we repeat $D_{SC}$ and $N_{SC}$ multiple times per time window until the residual falls below a given threshold. This corresponds to a fixed-point iteration for Equation (2) and is a classical Godunov splitting for a single iteration that always yields only first-order in time. We can summarize multirate single value coupling in an algorithmic setting as

1. Initialize $c_{D_{end}}^0$ with value from previous time window, $k = 0$.
2. Solve the Dirichlet problem (with $n_D$ time steps): $c_{N_{end}}^{k+1} = D_{SC} \left(c_{D_{end}}^k\right)$, meaning:
   a. Initialize $u_{D_{end}}^k$ with value from previous time window.
   b. For $i = 0, \ldots, n_D - 1$: $u_{D_{i+1}}^k = \hat{D} \left(c_{D_{end}}^k, u_{D_{i}}, \delta t_D\right)$.
   c. $c_{N_{end}}^{k+1} = \phi_D \left(u_{D_{end}}^k\right)$.
3. Solve the Neumann problem (with $n_N$ time steps): $c_{D_{end}}^{k+1} = N_{SC} \left(c_{N_{end}}^{k+1}\right)$, meaning:
   a. Initialize $u_{N_{end}}^{k+1}$ with value from previous time window.
   b. For $i = 0, \ldots, n_N - 1$: $u_{N_{i+1}}^{k+1} = \hat{N} \left(c_{N_{end}}^{k+1}, u_{N_{i}}, \delta t_N\right)$.
   c. $c_{D_{end}}^{k+1} = \phi_N \left(u_{N_{end}}^{k+1}\right)$.
4. If $\left\| c_D^{k+1} - c_D^k \right\|$ is small enough: Go to the next time window.
5. Else: $k = k + 1$ and go back to item 2.

$k$ denotes the iteration counter. Note that this procedure only converges if $H_{SC}$ is a contraction. Otherwise it requires an additional acceleration step such as underrelaxation or quasi-Newton in item 5 in the algorithm above. We describe details for this acceleration in Section 2.3.

### 2.2 Extension to waveform iteration

WIs are an elegant way to increase the order in time with minimal use of the solvers' internal details, that is, well suited for black-box coupling. By contrast to the method from the previous section, WIs exchange more data than just the solutions at the end points.

The basic idea is to use a time-continuous representation of $c_D$ instead of only the value $c_{D_{end}}$ at the end of the coupling window to connect the two solvers. We can construct representations of the continuous functions $c_D$ and $c_N$ from all time steps in the window $[t_{ini}; t_{ini} + \Delta t]$ by interpolation based on a finite dimensional basis. We, thus, define $c_D$ and $c_N$ as functions in time:

$$c_D : [t_{ini}; t_{ini} + \Delta t] \rightarrow \mathbb{R}^{m_D} \text{ and } c_N : [t_{ini}; t_{ini} + \Delta t] \rightarrow \mathbb{R}^{m_N},$$

where $m_D$ and $m_N$ are the number of degrees of freedom at the coupling interface.

If both solvers execute independent time steps within the time window as introduced above, they can now evaluate the continuous representations $c_D$ and $c_N$ as boundary conditions wherever necessary. If the Dirichlet solver, for example, executes three time steps using the trapezoidal rule (TR) and $\delta t_D = \frac{1}{2} \Delta t$, it samples $c_D$ at four points in time ($t_{ini}, t_{ini} + \frac{1}{2} \Delta t, t_{ini} + \Delta t, t_{ini} + \frac{1}{2} \Delta t$). The Neumann solver uses two implicit Euler (IE) time steps of size $\delta t_N = \frac{1}{2} \Delta t$, it evaluates $c_N$ at two points in time ($t_{ini} + \frac{1}{2} \Delta t$ and $t_{ini} + \Delta t$). If one of the solvers uses higher-order time integration (such as a Runge-Kutta RK4 scheme), necessary substeps can now also be retrieved from $c_D$ or $c_N$. Giving continuous boundary value representations to our solvers yields new operators

$$D_{WI} : c_D \leftrightarrow c_N \text{ and } N_{WI} : c_N \leftrightarrow c_D.$$

These new operators are composed of two parts: First, the actual solver actions,

$$D : c_D \leftrightarrow (c_{N,1}, \ldots, c_{N_{end}}) \text{ and } N : c_N \leftrightarrow (c_{D,1}, \ldots, c_{D_{end}}),$$
that execute Equation (1) \( n_D \) or \( n_N \) times, respectively, sample the coupling conditions from the time-continuous representations, and return all time-discrete values after postprocessing \( c_{N,i} = \phi_D(u_{D,i}) \) and \( c_{D,i} = \phi_N(u_{N,i}) \), respectively. Second, the interpolation operators,\(^1\)

\[ I_p c_0 : (c_1, \ldots, c_n) \mapsto c , \]

that generate an interpolant with an error of order \( \mathcal{O}(\Delta t^{p+1}) \) from \( n \) time-discrete points and an 'initial value' \( c_0 \) at \( t_{\text{ini}} \) (i.e., a total of \( n' = n + 1 \) datapoints). The new operators read

\[ D_{W1} = I_{p,c_{N,0}} \circ D \quad \text{and} \quad N_{W1} = I_{p,c_{D,0}} \circ N . \]

They yield a new fixed-point equation operating on functions in time

\[ c_D = N_{W1} \circ D_{W1}(c_D) \tag{3} \]

or, expressed in terms of the time-discrete steps,

\[ \left( c_{D,1}, \ldots, c_{D,n_N}, c_{D,0} \right) = \underbrace{N \circ I_{p,c_{N,0}} \circ D \circ I_{p,c_{D,0}}} \left( c_{D,1}, \ldots, c_{D,n_N} \right) . \tag{4} \]

We denote fixed-point iterations for Equations (3) or (4) as \( W_1(n_D,n_N,\cdot;\cdot) \) and the corresponding operator as \( H^p_{n_D,n_N} \). Figure 1 compares \( SC(2,5) \) and \( W_1(2,5;2) \) in a schematic way. Note that in the limit of \( \Delta t \) to zero, we would recover a semidiscrete discretization with Equation (3) representing a WI with waveforms in an infinite dimensional function space. This situation is regularly considered in the WI literature. In an algorithmic setting, we can summarize WI as

1. From the previous window, get \( c_{D,0} \) and \( c_{N,0} \). Set the initial approximation as constant extrapolation \( c^0_f \equiv c_{D,0}^+ \), \( k = 0 \).
2. Solve the Dirichlet problem (with \( n_D \) time steps): \( \left( c_{N,1}^{k+1}, \ldots, c_{N,n_D}^{k+1} \right) = D \left( c_{D,k}^{k+1} \right) \).
   a. Initialize \( u_{D,0}^k \) with value from previous time window.
   b. For \( i = 0, \ldots, n_D - 1 \): \( u_{D,i+1}^k = D \left( c_{D,i}^k \right) \).
   c. For \( i = 0, \ldots, n_D - 1 \): \( c_{D,i+1}^{k+1} = \phi_D(u_{D,i+1}^k) \).
3. Interpolate to obtain a continuous waveform: \( c_{N}^{k+1} = I_{p,c_{N,0}} \left( c_{N,1}^{k+1}, \ldots, c_{N,n_D}^{k+1} \right) \).
4. Solve the Neumann problem (with \( n_N \) time steps): \( \left( c_{D,1}^{k+1}, \ldots, c_{D,n_N}^{k+1} \right) = N \left( c_{N}^{k+1} \right) \).
   a. Initialize \( u_{N,0}^{k+1} \) with value from previous time window.
   b. For \( i = 0, \ldots, n_N - 1 \): \( u_{N,i+1}^{k+1} = N \left( c_{D,i}^{k+1} \right) \).
   c. For \( i = 0, \ldots, n_N - 1 \): \( c_{D,i+1}^{k+1} = \phi_N(u_{D,i+1}^{k+1}) \).
5. Interpolate to obtain a continuous waveform: \( c_{D}^{k+1} = I_{p,c_{D,0}} \left( c_{D,1}^{k+1}, \ldots, c_{D,n_N}^{k+1} \right) \).
6. If \( \left\| c_{D}^{k+1}(t) - c_{D}^{k}(t) \right\| \) small enough: Go to the next window.
7. Else: \( k = k + 1 \) and go back to item 2.

\(^1\)We occasionally drop the subscript for the quantities \( n_D, n_N, c_D, c_N \) if we do not refer to a specific side of the coupled problem, but to either.

\(^2\)Other choices for initialization are possible, but are beyond the scope of this article.
Figure 1  Comparison between single value coupling (SC(2,5), left) and waveform iteration (WI(2,5;2), right). For SC(2,5), the Dirichlet solver $D$, depicted in blue, sends the value $c_{N \text{end}}$ to the Neumann solver $N$, depicted in orange. $N$ uses $c_{N \text{end}}$ as boundary condition for all its respective time steps of size $\delta t_N$, exemplary depicted by the blue arrow for the first time step computing $c_{N',1}$. For WI(2,5;2) the Dirichlet solver $D$ provides the interpolant $c_{N'}$. $N$ samples $c_{N'}$ for obtaining boundary conditions for all its respective time steps of size $\delta t_N$, exemplary depicted by the blue arrow for the first time step computing $c_{N',1}$ using the sampled value $c_{N'}(t_{\text{ini}} + \delta t)$. Note that SC(2,5) is not equivalent to WI(2,5;0), that is, waveform iteration with constant interpolation. This is due to the fact that single value coupling does not use piecewise constant interpolation, but globally constant interpolation over the whole time window in time exactly and to (ii) achieve a desired approximation order $p$ in time in general: For (i), we have to use polynomial interpolation of at least degree $p$ to represent this solution exactly at the interface. For time integration, a scheme of at least order $p$ has to be used. For (ii), we have to use a $p$th-order polynomial interpolation, since we can at most afford an error of $O(\Delta t^p)$ per time window, which accumulates to an error of $O(\Delta t^p)$ over $O(\Delta t^{-1})$ time steps. For time integration, again a scheme of at least order $p$ has to be used.

2.3 Interface quasi-Newton methods

After having introduced two specific fixed-point problems with operators $H_{SC}$ and $H_{n_D,n_N}$ in the previous two sections, we now consider a general fixed-point problem and the associated residual:

$$\text{Find } x \in \mathbb{R}^m \text{ s.t. } H(x) = x \Longleftrightarrow R(x) := H(x) - x = 0.$$  

We explain how we can formulate an accelerated iteration

$$x^{k+1} = A(H(x^k)),$$

with $A$ denoting the acceleration operator. We stop the iteration, when the norm of the coupling residual $\|R(x^k)\|$ is small enough. If $A$ is the identity, we get a classical fixed-point iteration (Picard iteration), which only converges if $H$ is a contraction. For FSI, this is typically not true.39 In early FSI days, the acceleration operator $A$ has been realized as underrelaxation,

$$x^{k+1} = \omega H(x^k) + (1 - \omega)x^k, \omega \in ]0;1[,$$

with $\omega$ denoting the relaxation factor.
for example, based on a dynamic Aitken scheme. A numerically more efficient approach is to reuse past iterates to build up an approximate Jacobian of $R$ and use this in quasi-Newton iterations. To avoid linear dependencies between information from previous iterations, we have to perform a modified Newton iteration starting from the result of the pure fixed-point iteration (for details, see References 29):

$$ x^{k+1} = \tilde{x}^k + \Delta \tilde{x}^k \quad \text{with} \quad \Delta \tilde{x}^k = J^{-1}_R(\tilde{x}^k) \tilde{R}(\tilde{x}^k), $$

where we use $\tilde{x} := H(x)$, the modified residual $\tilde{R}(\tilde{x}) := \tilde{x} - H^{-1}(\tilde{x})$, and $J^{-1}_R(\tilde{x}^k)$ as an approximation of the inverse of the Jacobian of $\tilde{R}$. To compute $J^{-1}_R(\tilde{x}^k)$, we collect input-output information from past iterates of $R$ in tall and skinny matrices $V_k, W_k \in \mathbb{R}^{m \times k}$, $m \gg k$,

$$ V_k = \begin{bmatrix} R(\tilde{x}^1) - R(\tilde{x}^0), R(\tilde{x}^2) - R(\tilde{x}^1), \ldots, R(\tilde{x}^k) - R(\tilde{x}^{k-1}) \end{bmatrix}, $$

$$ W_k = \begin{bmatrix} \tilde{x}^1 - \tilde{x}^0, \tilde{x}^2 - \tilde{x}^1, \ldots, \tilde{x}^k - \tilde{x}^{k-1} \end{bmatrix}. $$

For transient coupled problems, we need to solve a fixed-point equation per time window. In this case, it is beneficial for efficiency to also include iterates from past windows in $V_k$ and $W_k$. This variant is however beyond the scope of this article and subject to future work. The matrices $V_k$ and $W_k$ define the multisecant equations for the inverse Jacobian

$$ J^{-1}_R(\tilde{x}^k) V_k = W_k. \tag{5} $$

To get the classical IQN-ILS, we close Equation (5) by

$$ \|J^{-1}_R(\tilde{x}^k)\|_F \rightarrow \min, $$

which gives

$$ J^{-1}_R = W_k(V_k^T V_k)^{-1} V_k^T. $$

In practice, the Jacobian is not stored, but calculated in matrix-free form from via a QR-decomposition of $V_k$,

$$ a = \arg\min_{a \in \mathbb{R}^k} \|V_k a + R_k\|_2 \quad \text{followed by} \quad \Delta \tilde{x}^k = W_k a. \tag{6} $$

### 2.4 Quasi-Newton waveform iteration

Now we combine the two ideas we discussed above: (i) We have used a WI approach dealing with multirate settings, that is, different numbers $n$ and $n'$ of time steps of size $\delta t_D$ and $\delta t_{D'}$ within a window of size $\Delta t$ in the involved solvers, such that $\Delta t = n_D \delta t_D = n_{D'} \delta t_{D'}$, and, at the same time, maintaining higher-order. (ii) We have presented the concept of quasi-Newton methods as an acceleration approach for the resulting interface fixed-point equations. Now, we show how we can combine the two parts toward a robust, efficient, and accurate coupling for partitioned simulations with black-box solvers.

The missing piece here is the answer to the question which residual components we should consider to build our quasi-Newton scheme. For a multirate setting with given $n_D$ and $n_{D'}$ based on the fixed-point equation Equation (2), the choice is canonical, that is, we use $x^k := c_D^k$. If we have $m_D$ degrees of freedom at the interface, $c_{Dend} \in \mathbb{R}^{m_D}$, we get $V_k, W_k \in \mathbb{R}^{m_D \times k}$. We call this variant QN-WI.

For WI, we have formulated our Dirichlet-Neumann coupling in terms of the fixed-point equation Equation (4). We have all $n_D$ time steps of $c_D$ included in the fixed-point equation Equation (4). The canonical quasi-Newton variant for this system concatenates all time steps' data in a large vector for both input and output of the operator $R$. Thus, we get matrices $V_k$ and $W_k$ in $\mathbb{R}^{(m_D \times n_D) \times k}$. We call this variant QN-WI. This method is computationally more expensive as we have to deal with a substantially larger interface system than for single value coupling. Typically, the size of the interface system has, however, a negligible influence on the overall performance.\(^9\)

\(^9\)We use a simple underrelaxation for the first iteration.
A variant to reduce the size of multisecant system Equation (5) is to measure the residuals not at all samples, but only a subset. To test such a setting in the following, we only consider one specific case: we only use the last sample \( c_{Dn_D} \), that is,

\[
V_k = \left[(\bar{R}(x^1))_{n_D} - (\bar{R}(\bar{x}^0))_{n_D}, (\bar{R}(\bar{x}^1))_{n_D} - (\bar{R}(\bar{x}^1))_{n_D}, \ldots, (\bar{R}(\bar{x}^k))_{n_D} - (\bar{R}(\bar{x}^{k-1}))_{n_D}\right] \in \mathbb{R}^{m_D \times k},
\]

\[
W_k = \left[\bar{x}^1 - \bar{x}^0, \bar{x}^2 - \bar{x}^1, \ldots, \bar{x}^k - \bar{x}^{k-1}\right] \in \mathbb{R}^{(m_D \times n_p) \times k}.
\]

In this case, the approximate matrix \( J^{-1} \) is no longer a square matrix but in \( \mathbb{R}^{(m_D \times n_D) \times m_D} \). However, we do not explicitly calculate \( J^{-1} \), but solve a least squares problem as in Equation (6) with \( \alpha \in \mathbb{R}^k \) as before. We call this variant reduced quasi-Newton \( r\text{QN-WI} \).

**Remark 1.** If we do not include all time steps in the input vector \( x \) and, thus, in the matrix \( W_k \), but simply restrict the fixed-point iteration formulation entirely to the last step \( n_D \), we would also only update the entry \( c_{Dn_D} \) and, thus, not generate an appropriate update of the whole waveform representation of \( c_D \).

**Remark 2.** The method of De Moerloose et al. \(^{30} \) corresponds to \( WI(1,n;1) \) or \( WI(n,1;1) \), that is, one solver executes \( n \) time steps \( \delta t \) within one window, while the time step of the other solver is equal to the window size \( \Delta t \). Linear interpolation is used to transfer information from the larger time step (\( \Delta t \)) solver to the other solver (\( \delta t = \Delta t / n \)). In this case, \( QN-WI \) and \( r\text{QN-WI} \) are equivalent. In our general setting, we impose no restrictions on time steps of the involved solvers except that they meet at a common time at the end of the window, that is, the window size \( \Delta t \) is a multiple of the solver’s time steps \( \delta t \). Therefore, our generalization also allows to naturally handle more than two coupled solvers.

**Remark 3.** In literature, WI, that is, \( H^p_{n_D,n_D'} \), is usually combined with (constant) underrelaxation. We compare our quasi-Newton variants to this standard technique in Section 4 and call the latter \( rel-WI \).

**Remark 4.** In the case \( \Delta t \) to zero, the method would be a quasi-Newton method for Equation (3) with waveforms in an infinite dimensional function space, formulated using Fréchet derivatives.

Summarizing, we can state that \( QN-WI \) and \( r\text{QN-WI} \) can be considered as quasi-Newton variants in space-time, whereas \( QN-SC \) only acts in space.

### 3 SOFTWARE

We now describe how we realize the aforementioned multirate coupling algorithms in software. In fact, the bigger picture of our research is to implement an elaborate and sophisticated treatment of the temporal dimension into the widespread open-source coupling software preCICE. \(^{33} \) This article is an important milestone for this endeavor. The purpose of this section is manifold: to motivate and justify software design choices, to make our research reproducible, also in different setups, and to allow existing preCICE users to quickly pick up, use, and further validate the novel multirate coupling algorithms. It is, however, also safe to directly jump to the numerical results in Section 4.

preCICE is a library that allows to couple existing (legacy) simulation codes to complete multiphysics simulations in a minimally invasive way. preCICE is written in C++, but also offers bindings for C, Fortran and Python. For many community codes (such as OpenFOAM, CalculiX, or SU2), ready-to-use adapters are provided. \(^{41} \) The setup of coupled codes and of the data to be exchanged is configured at run-time through a global XML file. Various quasi-Newton algorithms are already implemented in preCICE including IQN-ILS\(^{18} \) as introduced in Section 2.3.

Whereas single value coupling, as introduced in Section 2.1, is based on exchanging the values of the coupling variables at window end points, WI, as introduced in Section 2.2 needs time-continuous representation of the coupling variables over complete windows. This important mathematical difference is also reflected in the software requirements. preCICE currently only offers interface methods such as \( \text{data} = \text{read\_data}(...) \) and \( \text{write\_data(data,...)} \) to read and write values at a single point in time, which can be interpreted in most cases as the window end \( t_{\text{ini}} + \Delta t \). Afterward, the interface method \( \text{advance()} \) triggers the exchange of these values between both solvers and the computation of the quasi-Newton acceleration. This set of methods provides everything needed to realize single value coupling, which, in fact, is already fully supported\(^{42} \) by the preCICE version we use in this contribution, v1.6.1. WI, however, requires some notion of time in the application programming interface (API) of preCICE, which would make a fundamental redesign of the coupling logic necessary.
Instead of directly calling the preCICE API from the solver adapter, the waveform bindings are called which in turn call preCICE in addition to executing new components such as interpolation in time. Data access methods need to be augmented by time as input argument.

3.1 Waveform bindings

As the choice of the most suitable WI algorithm for preCICE is still an open research question, we instead aim for a prototype software design, which allows to easily test various multirate approaches without requiring immature refactoring of preCICE. A mature implementation of WI in preCICE is subject to future work. We, therefore, decided that the prototype should implement WI in a new middle layer called waveform bindings, sitting between the adapter code and the preCICE library. Whereas preCICE does the communication and the quasi-Newton acceleration of the coupling data, the waveform bindings handle the time-continuous representation. With this solution, we do not require any alterations to preCICE itself. Figure 2 sketches the layered architecture.

The essential trick to realize this separation of concerns is to treat a complete time window like a single time step in preCICE. To this end, the data fields of preCICE need to be altered: instead of one data field, such as Displacements, we now have one data field per time step of a window, Displacements1, Displacements2, and so on. preCICE can be configured to handle and communicate multiple fields, but has no notion of time for these fields, they are just several data fields for a single preCICE time step. The waveform bindings, on the other hand, have this notion of time: Displacements1 are the values after the first time step within a window, Displacements2 after the second time step, and so on. The API of the waveform bindings now offers methods such as write_data(data,..., time) to write timestamped data at a specific time step \( t \in [t_{\text{ini}} + \delta t, t_{\text{ini}} + 2\delta t, ..., t_{\text{ini}} + \Delta t] \) and data = read_data(..., time) to read data at an arbitrary \( t \in [t_{\text{ini}}, t_{\text{ini}} + \Delta t] \). If the latter does not coincide with a given time step, an interpolated value is returned. For data interpolation, we use a B-Spline curve. The corresponding implementation is provided in the SciPy package via the functions interpolate.splrep and interpolate.splev, which are used to compute interpolating B-Spline curves of degree one to four.

As we only use \( n + 1 \) data points in time and, in particular, neither use values from previous windows nor derivatives in time for B-Spline interpolation, we have the restriction \( p \leq n \) for \( L_{p,q} \). We use a \( p \)-th-order B-Spline, meaning piecewise linear (\( p = 1 \)), quadratic (\( p = 2 \)), or cubic (\( p = 3 \)) interpolation. A first-order B-Spline requires two data points, that is, \( n \geq 1 \), and is, thus, always possible, a second-order B-Spline requires at least three samples, that is, \( n \geq 2 \), and a third-order B-Spline at least four samples, that is, \( n \geq 3 \). Therefore, accuracy is limited by the number of time steps \( n \) that defines the maximal polynomial degree \( p \) of the interpolation scheme. In future work, we intend to remove this restriction by using data from previous windows.

The waveform bindings also offer a function advance() which in turn calls the preCICE advance() upon completion of a window to communicate the discrete values (such as Displacements1, Displacements2, ...) between both solvers. In the preCICE configuration, the quasi-Newton acceleration can now be configured to be computed on the full set of data fields or only a reduced subset as explained in Section 2.4.

\(^{8}\text{SciPy version 1.3.1}\)
3.2 Application programming interface

To only require minimal modification to existing adapters and testcases, the main API design goal of the waveform bindings is to mimic the preCICE API as closely as possible. In fact, solely the data access functions are altered as explained above to include time as an additional input argument. Therefore, rewriting an existing coupled code to use the waveform bindings instead of preCICE directly requires modifying less than ten lines of code, which makes alongside testing of single value coupling and WI easy.

Listing 1 exemplarily shows how the waveform bindings are used to couple a structure solver. The code is nearly identical to a coupled code directly using the preCICE API. In the following, we explain the slight differences. For a detailed explanation of the preCICE API itself, we refer to the respective documentation. The obvious change in line 1 is to import waveform_bindings instead of precice. Afterward, the waveform bindings need additional configuration in lines 11 to 13. The number of time steps per window on both sides as well as the interpolation strategy need to be defined. For the data access in lines 23 and 31, the preCICE API is extended as mentioned previously. In the example, the structure solver only samples force values at the end of each time step \( t + \delta t \) in line 23, as we use an implicit Newmark-beta method and, thus, only require these values. For other time stepping methods, forces could be read from the waveform bindings wherever necessary.

3.3 Limitations

The layered architecture, however, also comes with technical shortcomings. First, as the waveform bindings build upon the Python bindings of preCICE, currently only codes in Python can be coupled. Besides this language restriction, the software concept comes with a few additional limitations as expected for a prototype. For quasi-Newton acceleration, preCICE allows to reuse iterations from previous time steps to achieve faster convergence. As preCICE has no notion of time for the samples \( \text{Displacements}_1, \text{Displacements}_2, \ldots \), the reuse strategy would give wrong values. We, therefore, simply switch off this functionality for all experiments in the following section. For the same reason, we cannot use underrelaxation in preCICE, neither directly nor as starting value for the quasi-Newton methods. Otherwise the first relaxed iteration would be a weighted combination between the current solution and the waveform of the last window. Waveform relaxation rel-WI as introduced in Section 2.4, however, needs a weighted combination between the current solution and a constant waveform extrapolated from the end value of the previous window. As a remedy, we realized underrelaxation directly in the advance() function of waveform bindings and switched it off in preCICE. Last, each coupled solver needs to know the number of time steps per window for both sides at configuration, see Listing 1, lines 11 to 13, contradicting the flexibility argument of black-box coupling. We aim to eliminate all these limitations with a mature implementation of WI within preCICE in future work. Within preCICE in particular means to maintaining backward compatibility. To which extent this is possible is an open research question.

4 NUMERICAL RESULTS

We consider two testcases to evaluate the performance of the proposed algorithms both in terms of convergence properties of the quasi-Newton iterations and in terms of the achieved approximation order in time: (i) a simple 2D heat equation (diffusion equation) with known analytic solution, where the computational domain has been divided artificially into two partitions, (ii) a fluid-structure interaction scenario, where a solid truss is deformed by the surrounding fluid flow. Both examples are discretized in space using finite elements. To comply with the restriction of the waveform bindings to Python, we use the finite element frameworks FEniCS and Nutils. We choose two different codes to show-case the black-box methodology, for which testcases and adapters were already available. For interfacing between FEniCS
and preCICE, we make use of the FEniCS adapter, which converts between FEniCS and preCICE data structures and realizes the rewinding of time steps for implicit (or strong) coupling. Nutils, on the other hand, does not use an adapter, but directly accesses preCICE or the waveform bindings. Code and instructions for running the examples can be found on https://gitlab.lrz.de/precice/ijnme2019-experiments.

Listing 1 A simple structure solver in Python using the waveform bindings for WI(2, 5; 1)

```python
import waveform.bindings as precice

solver_name = "StructureSolver"  # Structure solver is the Neumann solver \mathcal{N}
interface = precice.Interface(solver_name)  # create handle to waveform bindings API
interface.configure("precice-config.xml")  # usual preCICE configuration

mesh_ID, vertex_IDS = ...  # define coupling mesh on \Gamma
u = setup_solver()  # initialize displacement field \mathbf{u}_N

# configuration for computing WI(2, 5; 1)
n_this, n_other = 5, 2  # define n_N and n_D
interpolation_strategy = "linear"  # use piecewise linear interpolation with p = 1
interface.configure_waveform_iteration(n_this, n_other, interpolation_strategy)

window_size = interface.initialize()  # preCICE couples at window size \Delta t
dt = window_size / n_this  # Neumann solver \mathcal{N} performs time step of size \Delta t_N = \Delta t / n_N

while interface.is_coupling_ongoing():
    if interface.writing_checkpoint_is_required():
        write_checkpoint(t, u)  # at window start \xi_{ini}

    # read \mathcal{C}_N at \Gamma + \Delta t_N needed for this time step from coupling interface
    forces = interface.read_block_vector_data("Forces", mesh_ID, vertex_IDS, t + dt)
    coupling_boundary_condition = set_forces(forces)  # apply \mathcal{C}_N

    # perform time step of size \Delta t_N
    u_new = do_time_step(u, coupling_boundary_condition, t, dt)

    # write values of \mathcal{C}_D at \Gamma + \Delta t_N to coupling interface
    displacements = compute_displacement(new_u)  # extract \mathcal{C}_D
    interface.write_block_vector_data("Displacements", mesh_ID, vertex_IDS, displacements, t + dt)

    # advance coupling interface by \Delta t_N
    interface.advance(dt)

    if interface.reading_checkpoint_is_required():  # rollback to \xi_{ini}
        t, u = read_checkpoint()
    else:  # continue to next time step \Gamma + \Delta t_N
        t, u = t + dt, u_new

interface.finalize()
```

\textsuperscript{11}Source code at https://github.com/precice/fenics-adapter. For directly accessing preCICE, we use version https://github.com/precice/fenics-adapter/commit/1ccab476e3293b34dafa274e365ad777f9411569; for using the waveform bindings, we use version https://github.com/precice/fenics-adapter/commit/61c82a613f47d79d3eb786f7446f691d9da2bb
4.1 Conjugate heat transfer

With this academic testcase, we want to test two things: (i) How well does QN converge for WI, which combination works best? (ii) Can we recover higher-order in time? We look at both in this order. As most of our conclusions are based on a known exact solution $u_{\text{exact}}$, the numerical methods in the solvers have been chosen accordingly. In particular, the solvers use finite elements of sufficiently high order to be able to solve the problem without spatial discretization error.

4.1.1 Scenario setup

We solve the dimensionless 2D time-dependent heat equation

$$\frac{\partial u}{\partial t} = \Delta u + f \text{ on } \Omega = [0, 2] \times [0, 1] \text{ for time } t \in [0; T]$$

as described in the FEniCS tutorials book. In the following, we extend this example, such that it can be used to assess the quality of the previously discussed algorithms. For details on finite element discretization and solution strategies in FEniCS, we refer the interested reader to the FEniCS tutorials book.

The right-hand side $f$ as well as the boundary conditions in Equation (7) are manufactured such that we get the analytical solution

$$u_{\text{exact}}(x, y, t) = 1 + g(t)x^2 + 3y^2 + 1.2t.$$ 

with an arbitrary continuous function $g$. We consider the following choices for $g(t)$:

- polynomial: $g_{\text{pol}}(t) = (1 + t)^a$, 
- trigonometric: $g_{\text{tri}}(t) = \sin(t)$.

Using $P^2$ finite elements and a sufficient approximation order in time allows us to recover $u_{\text{exact}}$ and to compute the exact heat flux $\bar{q}_{\text{exact}} = \vec{\nabla} u_{\text{exact}}$ via finite element projection.

To use the heat equation as a benchmark for partitioned solution strategies, we divide the domain $\Omega$ into two independent subdomains $\Omega_D = [0, 1] \times [0, 1]$ (handled by the Dirichlet solver) and $\Omega_N = [1, 2] \times [0, 1]$ (handled by the Neumann solver) such that $\Omega = \Omega_D \cup \Omega_N$ and the coupling boundary $\Gamma = \Omega_D \cap \Omega_N$, compare Figure 3. This setup will be referred to as the partitioned heat equation. We use a triangular equidistant mesh with $20 \times 20 \times 2$ triangles in $\Omega_D$ and $\Omega_N$. We extract the temperature $u$ or the normal component of the heat flux $q_\nu$ on $\Gamma$ from one participant to be provided for the other as Dirichlet boundary condition $u = c_D$ or Neumann boundary condition $\vec{n} \cdot \vec{\nabla} u = q_\nu = c_N$, respectively.

For time stepping, we use IE, TR, or a fourth-order spectral deferred correction (SDC) scheme, depending on the choice of $\alpha$ and the goal in terms of approximation order in time. We construct the SDC scheme from an IE time integration scheme with three Gauss-Lobatto collocation nodes at $t_{\text{ini}} + i\delta t, t_{\text{ini}} + (i + 0.5)\delta t$ and $t_{\text{ini}} + (i + 1)\delta t, i = 1 \ldots n$ and $K$ correction sweeps per time step of size $\delta t$.

4.1.2 Quasi-Newton convergence

In Section 2.4, we introduced two variants combining WI coupling with quasi-Newton acceleration, QN-WI and rQN-WI. In addition, we consider classical underrelaxation for rel-WI and the plain fixed-point iteration full-WI. For rel-WI, we apply an underrelaxation of 0.5, which is optimal for the semidiscrete case. We compare these methods concerning the speed of convergence. As a reference, we compare the convergence of all WI coupling acceleration methods to the convergence of QN-SC for SC($n_D, n_N$). The latter quantifies the price we pay for higher-order in time in terms of iteration counts per time step.

As we are not considering the approximation order of our time stepping scheme in this paragraph, we use IE time integration combined with piecewise linear interpolation for WI in both domains (WI($n_D, n_N$)). Note that for the setups in which WI is used (QN-WI, rQN-WI, rel-WI, full-WI), the termination criterion considers interface values at every time step. For the basic variant with plain single value coupling (QN-SC), only the values at the end of the time window are considered. We use a moderate relative coupling tolerance of $10^{-5}$ to reach reasonable runtimes. For all QN
FIGURE 3  Heat-heat coupling: Partitioning and mesh. Contour lines visualizing temperature are continuous across the coupling interface. This indicates that temperature and heat flux are continuous at the coupling interface.

FIGURE 4  Heat-heat coupling: Number of coupling iterations per time step for QN-WI, WI(1,1;1), implicit Euler for time stepping with time step/window sizes $\Delta t_d = \Delta t_N = \Delta t = 1$ for $T = 10$, different choices for $g(t)$, and a coupling tolerance of $10^{-12}$.

acceleration schemes, we use a $QR2$ filter$^{23}$ for the QR decomposition with limit $\epsilon = 10^{-3}$ and a residual-sum weighting of the individual sub vectors.$^{29}$

As testcase, we use $g_{uni}(t)$, since for $g_{pol}(t)$, the $g(t)x^2$ term quickly dominates the equation leading to a much simpler coupled problem. See Figure 4 for the iteration per time step for QN-WI for different choices of $g(t)$. Only the periodic $g_{uni}(t)$ shows a rather constant behavior of the iterations per time step.

We measure the average number of iterations required to terminate in every window until $T = 10$ and present the respective results in Table 1. We evaluate the performance for different window sizes $\Delta t = 0.1, 0.2, 0.5, 1.0, 2.0, 5.0$ and different multirate settings with $n_{D}, n_{N} \in \{1,3,5\}$. Our study showed that full-WI, that is, WI without acceleration, does not converge within 100 iterations per time step. Thus, we present only iteration counts for the other four variants. The most important findings are summarized in the following:

1. For rel-WI, the number of coupling iterations is almost independent of the multirate setup. All quasi-Newton variants need two to three times fewer iterations than rel-WI.
2. Overall, QN-WI is more robust, while rQN-WI is more efficient for special cases. In general, rQN-WI seems to have difficulties for setups with many substeps on both sides, while QN-WI can better cope with these setups. rQN-WI and QN-WI perform equally well for WI($n_{D},1;1$) as both methods are equivalent for these cases.
3. For all variants, the number of iterations slowly decreases with decreasing window size $\Delta t$. This effect is observable, but not very significant. Thus, there is potential to make coupling more asynchronous by applying WI on larger windows.
4. In comparison to existing literature,$^{29}$ the number of iterations needed in our experiments is relatively high although the application is not particularly challenging. We explain this with the fact that our current WI algorithm does not
FIGURE 5 Heat-heat coupling: visualization of coupling quantities temperature $u$ and heat flux $q_x$ and their time derivatives $\partial_t u$ and $\partial_t q_x$ at interface point $(x, y) = (1, 0.5)$ over time $t$ (solid lines) together with the respective exact solutions (dotted lines). We show the quantities for three windows of size $\Delta t = 2$ and a total simulation time $T = 6$. We use $ WI(5, 3; 2)$ and $g = g_{\text{tri}}$. Note that the coupling quantities are $C^0$ continuous across coupling windows, but not $C^1$ continuous, which can be seen from the discontinuity in the first derivative in time at the window boundaries $t = 2, 4$. The quantities $u$ and $\partial_t u$ are scaled for better readability.

yet apply further optimization of quasi-Newton that reduces the number of iteration dramatically in Reference 29. In particular, we do not reuse data from previous time windows. Implementing this reuse for WI is subject to future work.

5. **QN-SC** performs well and does not suffer from multirate, even though the time steps ($\delta t$) are ignored by the acceleration scheme. Note, that **QN-SC** uses an easier convergence criterion than the other two quasi-Newton variants, since only the last sample is considered at the end of the window and recall that this method does not allow to achieve more than first-order accuracy in time.

From these findings, we conclude that the convergence speed of the quasi-Newton based variants **QN-SC**, **QN-WI**, and **rQN-SC** is relatively similar. The underrelaxation based variant **rel-WI** performs significantly worse. In the remainder of this article, we will therefore focus on the quasi-Newton based variants.

**4.1.3 Approximation order in time**

Above, we have examined the convergence speed of different iterative coupling algorithms and concluded that only quasi-Newton accelerated algorithms are able to reach a competitive number of iterations. In the following, we concentrate on the accuracy of the quasi-Newton based multirate setups $ WI(n_D, n_X; p)$ and $ SC(n_D, n_X)$. Note that we do not apply any kind of additional conditions to enforce continuity across windows (see Figure 5). In a first step, we use $g_{\text{pol}}(t)$ and study for which setups and $p$ we cannot observe any discretization error in the solution $u$ with respect to the exact solution $u_{\text{exact}}$. As stated in Section 2.2, a time stepping scheme of at least order $p$ and waveform interpolation of at least degree $p$ are required to achieve this. We consider the numerical solution $u$ of the partitioned heat equation to be equal to the exact solution $u_{\text{exact}}$ and thus free of any discretization error in time and space, if the $L^2(\Omega)$ error is smaller than $10^{-12}$ in every time step. To avoid errors being introduced by the iterative coupling procedure, we use a very strict relative coupling tolerance of $10^{-12}$. As explained above, our discretization uses piecewise quadratic finite elements for the computation of $u$ to be able to represent $u_{\text{exact}}$ without discretization error in space.

Using **QN-WI** for $ WI(n_D, n_X; p)$, we were able to recover the exact solution for all multirate settings ($n_D, n_X \in \{1, 2, 3, 5\}$) and all window sizes ($\Delta t \in \{0.0125, 0.025, 0.05, 0.1, 0.2, 0.5, 1.0\}$) for the following cases:

1. $\alpha = 1$, IE, and piecewise linear interpolation,
2. $\alpha = 1$, TR, and piecewise linear interpolation,
3. $\alpha = 2$, TR, and piecewise quadratic interpolation,
4. $\alpha = 3$, SDC, and piecewise cubic interpolation,

For 3, we also observed that, as expected, piecewise linear interpolation is not sufficient, except for cases where interpolation is not required at all (such as $ WI(5,5;1)$). For 4, we additionally tested that piecewise linear and piecewise
### TABLE 1  Heat-heat coupling: Average number of coupling iterations per time step for four different coupling schemes and varying multirate setups, SC\((n,n)\) and WI\((n,n,1)\), and window sizes \(\Delta t\). As time integration method, implicit Euler is used for both solvers in all examples. The highest and the lowest number per column are always set in bold face. The number of iterations remains almost constant over the various multirate setups and slowly decreases with the window size.

|       | \(\Delta t\) | 5.0  | 2.0  | 1.0  | 0.5  | 0.2  | 0.1  |
|-------|--------------|------|------|------|------|------|------|
|       |              |      |      |      |      |      |      |
| rQN-WI | \(\Delta t\) | 5.0  | 2.0  | 1.0  | 0.5  | 0.2  | 0.1  |
| WI(1,1) |              |      |      |      |      |      |      |
| WI(1,3) |              |      |      |      |      |      |      |
| WI(1,5) |              |      |      |      |      |      |      |
| WI(3,1) |              |      |      |      |      |      |      |
| WI(3,3) |              |      |      |      |      |      |      |
| WI(3,5) |              |      |      |      |      |      |      |
| WI(5,1) |              |      |      |      |      |      |      |
| WI(5,3) |              |      |      |      |      |      |      |
| WI(5,5) |              |      |      |      |      |      |      |
| QN-SC | \(\Delta t\) | 5.0  | 2.0  | 1.0  | 0.5  | 0.2  | 0.1  |
| SC(1,1) |              |      |      |      |      |      |      |
| SC(1,3) |              |      |      |      |      |      |      |
| SC(1,5) |              |      |      |      |      |      |      |
| SC(3,1) |              |      |      |      |      |      |      |
| SC(3,3) |              |      |      |      |      |      |      |
| SC(3,5) |              |      |      |      |      |      |      |
| SC(5,1) |              |      |      |      |      |      |      |
| SC(5,3) |              |      |      |      |      |      |      |
| SC(5,5) |              |      |      |      |      |      |      |
| rQN-WI | \(\Delta t\) | 5.0  | 2.0  | 1.0  | 0.5  | 0.2  | 0.1  |
| WI(1,1) |              |      |      |      |      |      |      |
| WI(1,3) |              |      |      |      |      |      |      |
| WI(1,5) |              |      |      |      |      |      |      |
| WI(3,1) |              |      |      |      |      |      |      |
| WI(3,3) |              |      |      |      |      |      |      |
| WI(3,5) |              |      |      |      |      |      |      |
| WI(5,1) |              |      |      |      |      |      |      |
| WI(5,3) |              |      |      |      |      |      |      |
| WI(5,5) |              |      |      |      |      |      |      |
Figure 6: Heat-heat coupling: convergence of the $L^2(\Omega)$-error at $T = 1$ for the multirate setups $SC(5,3)$ and $WI(5,3;p)$ and various coupling, time integration and interpolation methods.

Quadratic interpolation is never sufficient (not even for $WI(5,5;1)$), since SDC internally performs two IE substeps, where a sufficiently accurate interpolation routine is required.

Using QN-SC for $SC(n_D,n_N)$, we observed that only for the trivial nonmultirate cases with $n_D, n_N = 1$ using IE and $\alpha = 1$ or TR and $\alpha = 2$, where no interpolation is necessary, the exact solution can be recovered. The only multirate settings, where the exact solution could be obtained are the edge cases $SC(1,n_N), n_N \in \{1,2,3,5\}$ with $\alpha = 1$ and IE. Here, the starting Dirichlet solver only requiring data at the end of the time window computes the correct flux for the end of the Neumann window, which is the only place where we measure the error. We expect the single time steps on Neumann side to not give the exact solution. For other multirate setups (such as $SC(5,2)$), we could not obtain the exact solution, which was to be expected.

In a second step, we determine the convergence order of time stepping for the setup using $g_{\text{tri}}$. For the convergence study, we determine the $L^2(\Omega)$ error of the numerical solution $u$ with respect to the exact solution $u_{\text{exact}}$. We consider the error at final simulation time $T = 1$ and decrease the window size $\Delta t$. We use a moderate coupling tolerance of $10^{-5}$. Figure 6 shows a convergence study for $WI(5,3;p)$ and $SC(5,3)$. Note, that we intentionally pick a setup where interpolated values are required by both participants of the coupled simulation at every time step. If $WI(5,3;p)$ is used for coupling in combination with a time stepping method of order $\geq p$, we expect convergence order $p$ as detailed in Section 2.2. We observe this for IE and TR, combined with linear and quadratic interpolation, respectively. For the fourth-order SDC and cubic interpolation, we observe a convergence order of approximately 3.5. In addition, we observe that the accuracy of SDC is bounded by the coupling tolerance for small $\Delta t$ and the error does not scale with the time step anymore. $SC(5,3)$, on the other hand, gives slow to no convergence and significantly higher errors for IE and TR time integration methods.

To investigate the effect of multirate schemes being used, we perform another convergence study, where TR with $WI(2,2;p), WI(5,2;p), WI(2,5;p)$, and $WI(5,5;p)$ is used (see Figure 7). The degree of the interpolating polynomials is irrelevant for cases, where no interpolation is needed, that is, $WI(2,2;p)$ and $WI(5,5;p)$ reach second-order regardless of the interpolation scheme being used. Note that this effect is only expected to occur for time stepping schemes such as TR or IE, where no additional function evaluations in time are required. For SDC, we did not observe this effect. For $WI(5,2;p)$ and $WI(2,5;p)$, piecewise quadratic interpolation ($p = 2$) allows us to reach second order of the time stepping scheme, as expected. Here, the benefit of multirate is clearly visible: $WI(2,5;2)$ shows a significantly lower error than $WI(5,2;2)$ and almost the same as $WI(5,5;2)$, which means that higher resolution in time for $N$ is more beneficial than for $D$. Presumably due to the term $x^2 \sin(t)$ in $u_{\text{exact}}$, which is significantly larger in $\Omega_N$.

If only piecewise linear interpolation ($p = 1$) is applied, second order can be reached for certain cases (eg, $WI(2,5;1)$). This might be due to the smaller time steps on the Neumann side, from which the Dirichlet solver reads the interpolated data. In other words, it can be assumed that what we observe is the convergence of the dominating spatial error of the Dirichlet solver (using a 2.5 times larger time step). $WI(5,2;1)$ shows the expected deterioration of the order below second order. We observe that, even though reaching second-order convergence, the accuracy of $WI(2,5;1)$ is significantly lower than for the equivalent case with higher-order interpolation $WI(2,5;1)$.

To summarize, we conclude that the converged solutions of QN-WI for $WI(n_N,n_D;p)$ fulfill the expectations in terms of accuracy. $SC(n,n)$, on the other hand, yields only first-order accuracy. We observe that interpolation plays an important role for WI. If we want to avoid order degradation, we have to use an interpolation with a sufficiently high polynomial degree. The required degree depends on the order of the time integration method, usually we require the polynomial...
degree $p$ to be at least equal to the order of the time stepping scheme. However, we notice that, for exceptional cases, $p$ lower than the order of the time stepping scheme can be sufficient.

### 4.2 Fluid-structure interaction

We now study whether we can generalize our observations from the relatively simple heat transfer problem to a more involved fluid-structure interaction problem. We show that QN-WI coupling allows us to couple two different second-order time integration methods, a TR used in the fluid solver and a Newmark-$\beta$ method used in the solid solver to an overall second-order coupled simulation even when combining different time step sizes in a multirate setup.

As FSI scenario, we study the Perpendicular Flap testcase from the preCICE tutorials§§. An elastic flap with a width of 0.1 m and a height of 1.0 m is mounted in crossflow at the center of a channel with a length of 6.0 m and a height of 4.0 m. Figure 8 left shows the setup at maximal deflection. This testcase is not experimentally validated, but cross-validated with different solver combinations. Compared with the preCICE tutorial, we slightly adjust the physical parameters: we use a higher fluid viscosity and a stiffer solid material to avoid requiring a finite element stabilization in the fluid solver and to use an only linear model in the structure solver, respectively. We set the fluid density to $\rho_F = 1.0 \text{ kg/m}^3$, the kinematic viscosity to $\nu_F = 1.0 \text{ m}^2/\text{s}$, the solid density to $\rho_S = 3.0 \cdot 10^3 \text{ kg/m}^3$, the Young's modulus to $E = 4.0 \cdot 10^7 \text{ kg/ms}^2$ and the Poisson ratio to $\nu = 0.3$. On the left boundary a constant inflow profile in $x$-direction of 10 m/s is prescribed. The right boundary is an outflow and the top and bottom of the channel as well as the surface of the flap are no-slip walls.

In the following, we shortly describe the solid and the fluid solver we use. Note that most details, in particular the spatial discretization, can be chosen arbitrarily for the purpose of this article. Thus, we only mention the respective (state-of-the-art) methods and put more emphasis on additional features that must be offered by a solver to allow higher-order time stepping in the coupled simulation. This, in particular, concerns the fluid solver: Whereas a stand-alone second-order fluid solver only needs to provide velocity and pressure with second-order accuracy in time, a fluid solver in a coupled simulation also needs to provide mesh displacements and forces with suitable accuracy.

For the solid domain, we use the linear elasticity solver developed in FEniCS and validated against CalculiX as part of the Bachelor thesis of Hertrich,§§ to which we also refer for details. We use a structured mesh with $5 \times 50$ cells, where each cell is cut into two triangles and $P_2$ finite elements for discretization. For time integration, as mentioned previously, we use a second-order Newmark-$\beta$ method. The resulting linear system is solved by a direct solver.

For the fluid domain, we implemented a new incompressible Navier-Stokes solver in Nutilis specifically for this testcase. We use a structured adaptive mesh as depicted in Figure 8 left. We formulate the fluid equations fully coupled and fully nonlinearly in an arbitrary-Lagrangian-Eulerian framework and discretize with $Q^2 - Q^1$ Taylor-Hood elements. The nonlinearity is resolved by Newton's method with line-search up to an absolute residual of $10^{-9}$. A direct solver is used for the resulting linear systems. The mesh displacement is modeled by a simple Laplace equation, discretized by $Q^1$ elements.

§§https://github.com/precice/tutorials
and again solved with a direct solver. For time integration, we make use of the second-order TR, meaning we replace the 
time derivative of the velocity by the finite difference $(u_{i+1} - u_i)/\delta t$ and use a linear average between the variables of the 
current and the previous time step for the diffusive and the convective term. To avoid pressure oscillations, however, we 
take the pressure gradient and the continuity term only at the new time level.

To get to second order in a coupled FSI setup, a few enhancements of the fluid time integration are necessary, as 
mentioned above. First, a second-order backward difference scheme is needed to compute the mesh velocity from the 

$$u_{\text{mesh}} = \frac{3 \cdot d_{i+1} - 4 \cdot d_i + d_{i-1}}{2 \cdot \delta t},$$

where $u_{\text{mesh}}$ is the mesh velocity, $d$ the mesh displacement, and the index $i$ the time step. Second, the computation of 
forces acting on the solid needs to be constructed carefully. We compute fluxes over the fluid boundaries by evaluating the 
residual of the weak formulation at the current solution and extracting the coupling boundary. The velocity, $u_{\text{mesh}}$, is the 
time derivative of the velocity. Furthermore, the TR with a fully implicit pressure gradient as explained above only results in 
second-order pressure values at the midpoint $p_{i+0.5}$. Using these values at the time level $i + 1$, as often done, is, however, 
only first-order accurate. To get second-order-accurate values at this time level, we extrapolate

$$p_{i+1} = \frac{3}{2} \cdot p_{i+0.5} - \frac{1}{2} \cdot p_{i-0.5}$$

in the pressure gradient term.

As the fluid solver and the solid solver use nonmatching meshes at the coupling boundary, we require data mapping 
methods between both meshes. We, therefore, use radial-basis function interpolation with thin-plates splines as 
provided by preCICE. In the fluid solver (Nutils), we directly read the displacement values at the Gauss points. 
FEniCS, however, requires a functional description of boundary conditions. We, therefore, read force values at the 
mes vertices and construct the boundary function again by a radial-basis function interpolation; this time in the 
FEniCS adapter.
To validate the new fluid solver, its mesh movement, and its force computation, we compare the Nutils-FEniCS results against OpenFOAM-FEniCS results from the preCICE tutorials. For both simulations, we use $\Delta t = 0.01$ s and $QN-SC(1,1)$ as coupling algorithm. Figure 8 shows a close agreement for the $x$-displacement of the tip until $T = 5.0$ s for an initial fluid at rest.

### 4.2.1 Convergence study

To study the temporal convergence order of a coupled simulation, we compute the Perpendicular Flap testcase until $T = 0.01$ s with decreasing window sizes $\Delta t_l = 0.0025 \cdot 2^{-l}$ s for time levels $l = 0, 1, \ldots, 4$. For all levels, we compute $QN$-WI with $WI(1,1;1)$, $WI(2,3;2)$, and $WI(3,2;2)$. As reference solution for all methods, we use $WI(1,1;1)$ with time level $l = 6$.

A crucial ingredient to see second order are suitable initial conditions. We start the fluid solver from an initial Stokes solution and then scale the forces on the structure until $T/2$ with a $C^1$-continuous $\sin^2(\pi(t + \delta t)/T)$ ramp. We measure the error in the displacement values at the coupling boundary to the reference solution $d_{ref}$. To smooth out variations, we consider the error over the complete time interval $[0, T]$, 

$$|||d_l - d_{ref}||| := \left( \frac{1}{N_l} \sum_{j=1}^{N_l} ||d_l(\Delta t_l \cdot j) - d_{ref}(\Delta t_l \cdot j)||_{L^2(\Gamma)}^2 \right)^{1/2} \quad l = 0, 1, \ldots, 4,$$

where $N_l = \frac{T}{\Delta t_l}$ denotes the number of windows. For every time slice, we use 

$$||d||_{L^2(\Gamma)} = \left( \sum_{\nu=1}^{m} d(x_\nu)^2 \right)^{1/2},$$

where $x_\nu$ denotes the coordinates at the coupling interface and $m$ the number of vertices. Thus, $d(x_\nu)$ are nodal evaluations.

Figure 9 compares the errors for all time levels. QNWI coupling achieves a clear second-order convergence for all three considered multirate setups. The errors of $WI(1,1;1)$ appear to be nearly the same as those of $WI(3,2;2)$, but for two times smaller time window sizes, which indicates that the solid time step size dominates the overall error. This again shows the great potential of multirate time stepping as the fluid problem is the far more expensive one. Thus, using a smaller time step size for the structure solver is almost for free. We compute the coupling iterations up to a relative residual error of $10^{-6}$ and list the average number of iterations in Table 2. We can observe a slight increase of required iterations with decreasing window size, contrary to our observations for the heat transfer testcase in the last section. This does not come as a surprise as, for incompressible FSI and vanishing structural mass, a similar behavior is reported in literature. It opens, however, the possibility for optimization as increasing the window size while keeping the time step size constant might lead to a more efficient numerical method, besides also having all the computing advantages of increased asynchronicity (such as reduced communication and better balancing of varying computational load). A detailed study is, however, beyond the scope of this article. Finally, comparing the required iterations of $WI(2,3;2)$ with $WI(3,2;2)$ could indicate that the structural time step size is the critical part.
5 | CONCLUSIONS AND OUTLOOK

This contribution shows that using WI in a black-box approach for partitioned multiphysics simulations is possible. Black-box means here in particular that the required interpolation in time can be computed without using discretization details of the coupled codes. An elegant prototype software design using a middle-layer called waveform bindings allows us to test arbitrary multirate settings with the coupling library preCICE without any alterations to the latter. Arbitrary multirate settings means that we allow any (and different) numbers of time steps in the involved solvers within a so-called coupling window. Our results show that (i) the implementation actually achieves the predicted order of consistency in time and that (ii) we can generalize interface quasi-Newton methods to the space-time domain. For more complex scenarios such as fluid-structure interaction, we observed that actually achieving higher-order requires careful consideration of all solver components, of mesh moving, and of evaluations of coupling variables.

Our current implementation has a couple of limitations, which we intend to tackle in future work: (i) We have to move the functionality of the waveform bindings to preCICE and decide how to do this while maintaining backward compatibility of the application user interface as far as possible. This also removes the technical restriction to Python. (ii) Features that optimize the performance of quasi-Newton schemes need to be provided also for WIs. This holds in particular for reuse of previous time step data and improved initial guesses for the initial approximation (currently constant extrapolation) of interface data before the first iteration in a new time window. (iii) Our methods should be generalized to adaptive time stepping, which is less of a mathematically unsolved problem, but a software issue as it might require varying numbers of time steps and, thus, varying numbers of coupling data structures in preCICE from one time window to the next. In addition to these enhancements, further studies for arbitrary high order (as in Reference 50) will be necessary.

ACKNOWLEDGEMENTS

We want to thank Harald van Brummelen for the fruitful discussions concerning the results of our research, and in particular concerning the second-order accuracy of the fluid solver in Section 4.2, Gertjan van Zwieten for the high-quality Nutils support and Richard Hertrich for the implementation and validation of the structural solver in FEniCS. Furthermore, we want to thank the reviewers of this article for their detailed and very valuable feedback. Last, we thankfully acknowledge the funding from the European Union’s Horizon 2020 research and innovation program under the Marie Skłodowska-Curie grant agreement No 754462 as well as funding from SPPEXA, the German Science Foundation Priority Program 1648—Software for Exascale Computing.

ORCID

Benjamin Rüth https://orcid.org/0000-0002-3116-0133
Benjamin Uekermann https://orcid.org/0000-0002-1314-9969
Miriam Mehl https://orcid.org/0000-0003-4594-6142
Philipp Birken https://orcid.org/0000-0002-6706-3634
Hans-Joachim Bungartz https://orcid.org/0000-0002-0171-0712

REFERENCES

1. Keyes DE, McInnes LC, Woodward C, et al. Multiphysics simulations: challenges and opportunities. Int J High Perform Comput Appl. 2013;27(1):4-83. https://doi.org/10.1177/1094342012468181.
2. Gander MJ. Waveform relaxation. In: Engquist B, ed. Encyclopedia of Applied and Computational Mathematics. Berlin, Heidelberg/Germany: Springer; 2015:1549-1552.
3. Burrage K. Parallel and sequential methods for ordinary differential equations. Numerical Mathematics and Scientific Computation. Oxford, UK: Clarendon; 1995.
4. Lelarasmee E, Ruehli A, Sangiovanni-Vincentelli A. The waveform relaxation method for time-domain analysis of large scale integrated circuits. *IEEE Trans Comput-Aided Des Integr Circ Syst.* 1982;CAD-1(3):131-145. https://doi.org/10.1109/TCAD.1982.1270004.

5. White JK. Multirate Integration Properties of Waveform Relaxation with Applications to Circuit Simulation and Parallel Computation (Phd thesis). University of California, Berkeley; 1985.

6. White JK, Sangiovanni-Vincentelli A, Odeh F, Ruehli A. *Waveform Relaxation: Theory and Practice.* Berkeley, CA: Electronics Research Laboratory, College of Engineering, UCB; 1985.

7. Reichelt M, White JK, Allen J, Odeh F. Waveform relaxation applied to transient device simulation. Paper presented at: Proceedings of the IEEE International Symposium on Circuit and Systems; 1988:1647-1650; Espoo, Finland.

8. Schöps S. Multiscale Modeling and Multirate Time-Integration of Field/Circuit Coupled Problems (Phd thesis). Katholieke Universiteit Leuven, Leuven; 2011.

9. Schöps S, De Gersem H, Bartel A. Higher-order cosimulation of field/circuit coupled problems. *IEEE Trans Mag.* 2012;48(2):535-538. https://doi.org/10.1109/TMAG.2011.2174039.

10. Maciejewski M, Garcia IC, Schöps S, et al. Application of the waveform relaxation technique to the co-simulation of power converter controller and electrical circuit models. Paper presented at: Proceedings of the 22nd International Conference on Methods and Models in Automation and Robotics (MMAR); 2017:837-842.

11. Gander MJ, Kwok F, Mandal BC. Dirichlet-Neumann and Neumann-Neumann waveform relaxation algorithms for parabolic problems. *Electron Trans Numer Anal.* 2016;45:424-456.

12. Monge A, Birken P. A multirate Neumann–Neumann waveform relaxation method for heterogeneous coupled heat equations. *SIAM J Sci Comput.* 2019;41(5):S86-S105. https://doi.org/10.1137/18M1187878.

13. Monge A. Partitioned methods for time-dependent thermal fluid-structure interaction (Phd thesis). Lund University, Lund; 2018.

14. Monge A, Birken P. A time adaptive Neumann-Neumann waveform relaxation method for thermal fluid-structure interaction. Paper presented at: Proceedings of the 25th Domain Decomposition Conference; 2019; Newfoundland, Canada.

15. Reichelt MW, White JK, Allen J. Optimal convolution SOR acceleration of waveform relaxation with application to parallel simulation of semiconductor devices. *SIAM J Sci Comput.* 1995;16(5):1137-1158. https://doi.org/10.1137/0916066.

16. Lumsdaine A, Wu D. Krylov subspace acceleration of waveform relaxation. *SIAM J Numer Anal.* 2003;41(1):90-111. https://doi.org/10.1137/S0036142996313142.

17. Horton G, Vandewalle S. A space-time multigrid method for parabolic partial differential equations. *SIAM J Sci Comput.* 1995;16(4):848-864. https://doi.org/10.1137/0916050.

18. Degroote J, Bathe KJ, Vierendeels J. Performance of a new partitioned procedure versus a monolithic procedure in fluid-structure interaction. *Comput Struct.* 2009;87(11-12):793-801. https://doi.org/10.1016/j.compstruc.2008.11.013.

19. Haeltlman R, Degroote J, Heule VD, Vierendeels J. The quasi-Newton least squares method: a new and fast secant method analyzed for linear systems. *SIAM J Numer Anal.* 2009;47(3):2347-2368. https://doi.org/10.1137/070710469.

20. Anderson DG. Iterative procedures for nonlinear integral equations. *J ACM.* 1965;12(4):547-560. https://doi.org/10.1145/321296.321305.

21. Srhöps S, Vierendeels J. Parallel coupling numerics for partitioned fluid-structure interaction. Paper presented at: Proceedings of 6th International Conference on Computational Methods for Coupled Problems in Science and Engineering, Venice, May 18 – 20 Barcelona, Spain: International Center for Numerical Methods in Engineering (CIMNE); 2015.

22. Reichelt MW, White JK, Allen J, Odeh F. Waveform relaxation applied to transient devices simulation. Paper presented at: Proceedingsof6thInternationalConferenceonComputationalMethodsforCoupledProblemsinScienceandEngineering,Venice,May18–20Barcelona,Spain:InternationalCenterforNumericalMethodsinEngineering(CIMNE);2015.

23. Schöps S, De Gersem H, Bartel A. Higher-order cosimulation of field/circuit coupled problems. Paper presented at: Proceedings of 6th International Conference on Computational Methods for Coupled Problems in Science and Engineering, Venice, May 18 – 20 Barcelona, Spain: International Center for Numerical Methods in Engineering (CIMNE); 2015.

24. Mehl M, Uekermann B, Bijl H, Blom D, Gatzhammer B, Van Zuijlen A. Parallel coupling numerics for partitioned fluid-structure interaction simulations. *Comput Math Appl.* 2016;71(4):869-891. https://doi.org/10.1016/j.camwa.2015.12.025.

25. Haeltlman R, Bogaers A, Uekermann B, Scheufele K, Meh M. Improving the performance of the partitioned QN-ILS procedure for strongly coupled fluid-structure interaction. Paper presented at: 6th International Conference on Computational Methods for Coupled Problems in Science and Engineering, Munich, Munich, Germany; 2016.
34. Logg A, Mardal KA, Wells G. Automated Solution of Differential Equations by the Finite Element Method: The FEniCS Book. Vol 84. Berlin, Heidelberg: Springer Science & Business Media; 2012.
35. Zwieten BG, Zwieten BJ, Verhoosel C, Fonn E, Opstal BT, Hoitinga W. Nutils (Version 5.0). Zenodo. 2019. https://doi.org/10.5281/zenodo.3243447.
36. Monge A, Birken P. On the convergence rate of the Dirichlet–Neumann iteration for unsteady thermal fluid–structure interaction. Comput Mech. 2018;62(3):525-541. https://doi.org/10.1007/s00466-017-1511-3.
37. Farhat C, Lesoinne M. Two efficient staggered algorithms for the serial and parallel solution of three-dimensional nonlinear transient aeroelastic problems. Comput Methods Appl Mech Eng. 2000;182(3-4):499-515. https://doi.org/10.1016/S0045-7825(99)00206-6.
38. Felippa CA, Park KC. Synthesis tools for structural dynamics and partitioned analysis of coupled systems. In: Ibrahimbegović A, Blank B, eds. Engineering Structures Under Extreme Conditions: Multi-Physics and Multi-Scale Computer Models in Non-Linear Analysis and Optimal Design. Amsterdam, Netherlands: IOS Press; 2004:50-110.
39. Brummelen vEH. Added mass effects of compressible and incompressible flows in fluid-structure interaction. J Appl Mech 2009; 76(2): 1–7. https://doi.org/10.1115/1.3059565
40. Küttler U, Wall WA. Fixed-point fluid–structure interaction solvers with dynamic relaxation. Comput Mech. 2008;43(1):61-72. https://doi.org/10.1007/s00466-008-0255-5.
41. Uekermann B, Bungartz HJ, Cheung Yau L, Choudakis G, Rusch A. Official preCICE adapters for standard open-source solvers. GACM Colloquium on Computational Mechanics for Young Scientists from Academia and Industry. Stuttgart, Germany: Institute for Structural Mechanics, University of Stuttgart; 2017:210-213.
42. Gatzhammer B. Efficient and Flexible Partitioned Simulation of Fluid-Structure Interactions (Phd thesis). Department of Informatics, Technical University of Munich, Munich, Germany; 2015.
43. Dierckx P. Curve and Surface Fitting with Splines: Monographs on Numerical Analysis. Oxford, UK: Oxford Science Publications, Clarendon Press; 1993.
44. Alnæs M, Blechta J, Hake J, et al. The FEniCS project version 1.5. Arch Numer Softw. 2015;3(100):9-23. https://doi.org/10.11588/ans.2015.100.20553.
45. Langtangen HP, Logg A. Solving PDEs in Python - The FEniCS Tutorial I. 1st ed. New York, NY: Springer International Publishing.; 2016.
46. Dutt A, Greengard L, Rokhlin V. Spectral deferred correction methods for ordinary differential equations. BIT Numer Math. 2000;40(2):241-266. https://doi.org/10.1023/A:1022338906956.
47. Hertrich R. Partitioned Fluid Structure Interaction: Coupling FEniCS and OpenFOAM via preCICE (Bachelor’s thesis). Technical University of Munich, Munich; 2019. https://mediatum.ub.tum.de/doc/1520579/.
48. Van Brummelen E, Van Der Zee K, Garg V, Prudhomme S. Flux evaluation in primal and dual boundary-coupled problems. J Appl Mech. 2012;79(1):010904-1-010904-8. https://doi.org/10.1115/1.4005187.
49. Lindner F, Mehl M, Uekermann B. Radial basis function interpolation for black-box multi-physics simulations. Paper presented at: Proceedings of the 7th International Conference on Computational Methods for Coupled Problems in Science and Engineering; 2017:1-12.
50. Huang DZ, Pazner W, Persson PO, Zahr MJ. High-order partitioned spectral deferred correction solvers for multiphysics problems. 2019:1-25. arXiv:1909.01977.

SUPPORTING INFORMATION
Additional supporting information may be found online in the Supporting Information section at the end of this article.

How to cite this article: Rüth B, Uekermann B, Mehl M, Birken P, Monge A, Bungartz H-J. Quasi-Newton waveform iteration for partitioned surface-coupled multiphysics applications. Int J Numer Methods Eng. 2021;122:5236–5257. https://doi.org/10.1002/nme.6443