A FAST TIME-STEPPING STRATEGY FOR DYNAMICAL SYSTEMS EQUIPPED WITH A SURROGATE MODEL∗

STEVEN ROBERTS†, ANDREY A POPOV†, ARASH SARSHAR†, AND ADRIAN SANDU†

Abstract. Simulation of complex dynamical systems arising in many applications is computationally challenging due to their size and complexity. Model order reduction, machine learning, and other types of surrogate modeling techniques offer cheaper and simpler ways to describe the dynamics of these systems but are inexact and introduce additional approximation errors. In order to overcome the computational difficulties of the full complex models, on one hand, and the limitations of surrogate models, on the other, this work proposes a new accelerated time-stepping strategy that combines information from both. This approach is based on the multirate infinitesimal general-structure additive Runge–Kutta (MRI-GARK) framework. The inexpensive surrogate model is integrated with a small timestep to guide the solution trajectory, and the full model is treated with a large timestep to occasionally correct for the surrogate model error and ensure convergence. We provide a theoretical error analysis, and several numerical experiments, to show that this approach can be significantly more efficient than using only the full or only the surrogate model for the integration.

Key words. Multirate time integration, Surrogate model, General-structure additive Runge–Kutta methods, Reduced-order modeling, Machine learning

AMS subject classifications. 65L05, 65F99

1. Introduction. This paper will consider the system of ordinary differential equations (ODEs)

\begin{equation}
y' = f(t, y), \quad y(t_0) = y_0, \quad y \in \mathbb{C}^N,
\end{equation}

equipped with a surrogate model approximating the dynamics of (1.1):

\begin{equation}
y'_{\text{sur}} = f_{\text{sur}}(t, y_{\text{sur}}), \quad y_{\text{sur}} \in \mathbb{C}^S.
\end{equation}

This surrogate model is assumed to be much less expensive to solve than the full model (1.1), possibly by evolving in a lower dimensional space (S < N). Moreover, we assume that projections between the full and surrogate model spaces are realized by the matrices \(V, W \in \mathbb{C}^{N \times S}\):

\begin{equation}
y_{\text{sur}} = W^*y, \quad y \approx Vy_{\text{sur}}, \quad W^*V = I_{S \times S}.
\end{equation}

Here, the “∗” symbol denotes the conjugate transpose of a matrix.

This paper presents a new technique to supplement the numerical integration of (1.1) using the surrogate model (1.2). For an appropriate choice of surrogate model, our method can be significantly more efficient than using either the full or the surrogate model alone. This is accomplished by applying a multirate time integration scheme to the following ODE with fast dynamics \(f^{(1)}\) and slow dynamics \(f^{(s)}\):

\begin{equation}
y' = \frac{Vf_{\text{sur}}(t, W^*y) + f(t, y) - Vf_{\text{sur}}(t, W^*y)}{f^{(1)}(t, y)}.
\end{equation}

∗Submitted to the editors DATE.

Funding: This work was supported by awards NSF ACI–1709727, NSF CDS&E–MSS 1953113, DE-SC0021313, and by the Computational Science Laboratory at Virginia Tech.

†Computational Science Laboratory, Virginia Polytechnic Institute and State University, Blacksburg, VA 24060 (steven94@vt.edu, apopov@vt.edu, sarshar@vt.edu, sandu@cs.vt.edu).

This manuscript is for review purposes only.
Multirate methods are characterized by using different stepsizes for different parts of an ODE, as opposed to a single, global timestep \([\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots]\). For problems with partitions exhibiting vastly different time scales, stiffnesses, evaluation costs, or amounts of nonlinearity, multirate methods can be more efficient than their single rate counterparts. In (1.4), the fast partition \(f^{(f)}\) is treated with a small timestep, and the slow partition \(f^{(s)}\) is treated with a large timestep. Note that (1.4) has the same solution as (1.1). It is rewritten and partitioned, however, in such a way that \(f^{(f)}\) contains the surrogate model dynamics and \(f^{(s)}\) represents the error in the surrogate model. Ideally, this error will be small, so a large timestep would be acceptable. Moreover, this means expensive evaluations of the full model occur infrequently compared to the inexpensive surrogate model evaluations.

With about 60 years of development \([\cdots\cdots]\), the multirating strategy has been applied to numerous classes of time integration methods. Conceivably any multirate method suitable for additively partitioned systems could be applied to (1.4). This paper builds upon multirate infinitesimal methods \([\cdots\cdots\cdots\cdots\cdots]\), as they offer a particularly flexible and elegant approach to multirate integration. In particular, we use the multirate infinitesimal general-structure additive Runge–Kutta (MRI-GARK) framework proposed in \([\cdots\cdots]\). MRI-GARK is an appealing choice as it generalizes many types of multirate infinitesimal methods, allows for the construction of high order methods, and supports implicit stages. We note, however, that this paper primarily focuses on explicit methods where (1.1) is nonstiff.

There are other instances in the literature where multirate methods and surrogate models are used in conjunction. In \([\cdots\cdots]\), an implicit multirate scheme is used to simulate a fast-evolving electric circuit coupled with a slow-evolving electric field. Model order reduction is then applied to the electric field problem to further reduce the computational cost. A multirate Runge–Kutta–Chebyshev method is proposed in \([\cdots\cdots]\) using a time-averaged right-hand side, which can be viewed as a surrogate model with better stiffness properties. The idea of treating models of different fidelities with different timesteps is also explored in \([\cdots\cdots\cdots\cdots\cdots]\). Surrogate models have been used extensively to speed up optimization problems when objective function evaluations are expensive \([\cdots\cdots\cdots\cdots\cdots]\).

From the multiscale modeling community, there are several related ideas such as high-order/low-order methods \([\cdots\cdots]\). Similar to the methods proposed in this work, the heterogeneous multiscale method and projective integration \([\cdots\cdots]\) utilize different timesteps for “microscopic” and “macroscopic” models as well as operators to convert a state between the two models. They seek to efficiently approximate macroscopic quantities of interest with the help of a microscopic model. In their implementations, the microscopic model is evaluated in small bursts of microsteps to resolve transient effects, while the macroscopic model is evolved with large timesteps. Typically, the coupling between models introduces first order temporal errors. In contrast, the new schemes proposed in this work seek to approximate the full, microscopic model to high orders using large macrosteps.

This remainder of this paper is structured as follows. In section 2, we review the MRI-GARK framework and extensions. Section 3 then specializes MRI-GARK-type methods to the special ODE (1.4). An error analysis is performed in section 4, and various approaches for constructing surrogate models are discussed in section 5. Convergence and performance experiments can be found in section 6. Finally, section 7 summarizes the results of the paper and proposes future extensions.
2. Multirate infinitesimal general-structure additive Runge-Kutta methods. In this section, we will briefly review the MRI-GARK framework, as it serves as the foundation for the remainder of the paper. These methods numerically solve ODEs that are additively partitioned into fast and slow dynamics:

\[ y' = f(t, y) = f^{(f)}(t, y) + f^{(s)}(t, y). \]

The defining characteristic of multirate infinitesimal methods is that the slow dynamics \( f^{(s)} \) is propagated with a discrete method, most commonly a Runge–Kutta method, while the fast dynamics \( f^{(f)} \) is propagated continuously through modified fast ODEs. An MRI-GARK scheme \([?]\) advances the solution of (2.1) from time \( t_n \) to \( t_{n+1} = t_n + H \) via the following computational process:

\[
\begin{align*}
(2.2a) \quad & Y^{(s)}_1 = y_n, \\
& v_i(0) = Y^{(s)}_i, \\
& T_i = t_n + \epsilon^{(s)}_i H, \\
(2.2b) \quad & v'_i(\theta) = \Delta c^{(s)}_i f^{(f)}(T_i + \Delta c^{(s)}_i \theta, v_i(\theta)) + \sum_{j=1}^{i+1} \gamma_{i,j}(\frac{\theta}{H}) f^{(s)}(T_j, Y^{(s)}_j), \\
& \text{for } \theta \in [0, H), \\
& Y^{(s)}_{i+1} = v_i(H), \quad i = 1, \ldots, s^{(s)}, \\
(2.2c) \quad & y_{n+1} = Y^{(s)}_{s^{(s)}+1}.
\end{align*}
\]

There are \( s^{(s)} \) modified fast ODEs (2.2b) solved between the abscissae of a Runge–Kutta method. The distances between consecutive abscissae are

\[
\Delta c^{(s)}_i = \begin{cases} 
\epsilon^{(s)}_{i+1} - \epsilon^{(s)}_i, & i = 1, \ldots, s^{(s)} - 1, \\
1 - \epsilon^{(s)}_{s^{(s)}}, & i = s^{(s)}.
\end{cases}
\]

Equation (2.2b) also contains time-dependent, polynomial forcing terms for the slow dynamics which satisfy

\[
(2.3) \quad \gamma_{i,j}(t) := \sum_{k \geq 0} \gamma_{i,j}^k t^k, \quad \gamma_{i,j}(t) := \int_0^t \gamma_{i,j}(\tau)d\tau = \sum_{k \geq 0} \gamma_{i,j}^k \frac{t^{k+1}}{k+1}, \quad \gamma_{i,j} := \gamma_{i,j}(1).
\]

Continuing with the notation of \([?]\), capitalized versions of (2.3) denote matrices of coefficients. If \( \Gamma \) is lower triangular, (2.2) is explicit, and otherwise, it is implicit.

For general \( f^{(f)} \), one cannot expect to find a closed-form solution to the modified fast ODE (2.2b); however, one can determine \( v_i(H) \) by time-stepping with “infinitesimally” small timesteps. For the sake of simplifying the analysis, we assume this integration is exact, but computationally, it is not possible to take infinitely many steps. In practice, (2.2b) is integrated to an accuracy that is negligible compared to the other source of error: treating \( f^{(s)} \) with a discrete Runge–Kutta method. This is analogous to assuming integrations within a Strang splitting are exact, but using sufficiently accurate approximations in practice. Multirate infinitesimal methods enjoy great flexibility in that any consistent time integration method can be used to solve the modified fast ODEs. This offers the freedom to choose implicit or explicit methods, as well as methods of any order. Equation (2.2) is multirate in the sense that there are \( s^{(s)} \) evaluations of \( f^{(s)} \) over a timestep \( H \), while (2.2b) is generally integrated with a smaller timestep and requires many more evaluations of \( f^{(f)} \).
2.1. Coupled MRI-GARK schemes. In [2] two new types of MRI-GARK methods were proposed: step predictor-corrector MRI-GARK (SPC-MRI-GARK) and internal predictor-corrector MRI-GARK (IPC-MRI-GARK). In order to improve the linear stability of MRI-GARK, both include coupled, discrete prediction stages that are subsequently corrected by solving modified fast ODEs. In comparing the two MRI-GARK variants, SPC-MRI-GARK was identified to have simpler order conditions, better stability, and a simpler implementation. For these reasons, we will only consider SPC-MRI-GARK in the construction of surrogate model timestepping techniques.

An SPC-MRI-GARK method based on the “slow” Runge–Kutta method \((s^{(s)}, c^{(s)})\) solves (2.1) with steps of the form:

\[
\begin{align}
(2.4a) \quad Y_i &= y_n + H \sum_{j=1}^{s^{(s)}} a_{i,j}^{(s)} f(t_n + c_j^{(s)} H, Y_j), \quad i = 1, \ldots, s^{(s)}, \\
(2.4b) \quad \begin{cases} 
  v(0) = y_n, \\
  v' = f(t_n + \theta, v) + \sum_{j=1}^{s^{(s)}} \gamma_j f_j(t_n + c_j^{(s)} H, Y_j), \quad &\text{for } \theta \in [0, H], \\
  y_{n+1} = v(H).
\end{cases}
\end{align}
\]

Similar to a traditional Runge–Kutta scheme, SPC-MRI-GARK starts by computing stage values that approximate the solution at the times \(t_n + c_j^{(s)} H\). In this prediction step (2.4a), the fast and slow dynamics are treated together with the timestep \(H\). The fast part of the stage values is inaccurate from this large timestep and is discarded. A single ODE (2.4b) is solved from \(t_n\) to \(t_{n+1}\) to correct the fast dynamics and produce the next step \(y_{n+1}\). Like MRI-GARK, there are time-dependent, polynomial forcing terms for the slow tendencies. Following [2, Definition 2.2], the definitions of \(\gamma_i(t)\), \(\tilde{\gamma}_i(t)\), \(\tau_i\) are identical to (2.3), except they are vector quantities with a single subscript index. While base methods of any structure are admitted in the SPC-MRI-GARK framework, only diagonally implicit methods were considered in [2]. As the focus of this paper is explicit methods, we have derived new explicit methods of orders two and three and present their coefficients in Appendix A.

3. Method formulation. Following the process outlined in the introduction, we seek to apply the MRI-GARK method (2.2) to the partitioned problem (1.4) in order to construct a solution process that can incorporate surrogate model information. We start with the modified fast ODE (2.2b):

\[
(3.1) \quad v'_i(\theta) = \Delta c_i^{(s)} \mathcal{V}_{\text{sur}}(T_i + \Delta c_i^{(s)} \theta, \mathbf{W}^{*} v_i(\theta)) \\
+ \sum_{j=1}^{i+1} \gamma_{i,j}(\frac{\theta}{H})(f(T_j, Y_j) - \mathcal{V}_{\text{sur}}(T_j, \mathbf{W}^{*} Y_j)).
\]

In (3.1), the full model \(f\) only appears in the forcing terms, and for explicit methods, it is evaluated at previously computed stage values. Only the surrogate model \(f_{\text{sur}}\) needs to be evaluated at each of the infinitesimal steps to compute \(v_i(H)\).

Note that in formulation (3.1), \(v_i \in \mathbb{C}^N\) evolves in the full space, and there is no benefit from choosing a surrogate model that evolves in a lower dimensional space. To resolve this, we first split \(v_i\) into the parts residing inside and outside the surrogate
model space:
\[ v_i(\theta) = \mathbf{VW}^* v_i(\theta) + (\mathbf{I}_{N \times N} - \mathbf{VW}^*) v_i(\theta). \]

More formally, \( \mathbf{VW}^* \) and \( (\mathbf{I}_{N \times N} - \mathbf{VW}^*) \) define oblique projections onto the range of \( \mathbf{V} \) and the nullspace of \( \mathbf{W}^* \), respectively. We now consider the solution to (3.1) from the perspective of these two, complementary subspaces.

Starting outside the surrogate model space, we have that
\[
(I_{N \times N} - \mathbf{VW}^*) v_i(\theta) = (I_{N \times N} - \mathbf{VW}^*) \left( Y_i + \int_0^H v_i'(\theta) d\theta \right)
\]
\[
= (I_{N \times N} - \mathbf{VW}^*) \left( Y_i + H \sum_{j=1}^i \gamma_{i,j} f(T_j, Y_j) \right),
\]
where we have used (2.3) and the fact that \( (I_{N \times N} - \mathbf{VW}^*) \mathbf{V} = 0_{N \times N} \) by (1.3).

Equation (3.2) reveals that inside the nullspace of \( \mathbf{W}^* \), the solution to equation (3.1) becomes a Runge–Kutta stage. Therefore, we exclude the components in this subspace during the infinitesimal step integration of (3.1).

We now consider the dynamics (3.1) projected onto the surrogate model space:
\[
z_i(\theta) := \mathbf{W}^* v_i(\theta) \in \mathbb{C}^S,
\]
\[
z_i'(\theta) = c_i \{ s \} f_{\text{sur}}(T_i + c_i \{ s \} \theta, z_i(\theta)) + \sum_{j=1}^{i+1} \gamma_{i,j} f(\theta) \left( \mathbf{W}^* f(T_j, Y_j) - f_{\text{sur}}(T_j, \mathbf{W}^* Y_j) \right).
\]

The solution to (3.1) can be expressed as the sum of its two components:
\[
v_i(H) = \mathbf{V} z_i(H) + (I_{N \times N} - \mathbf{VW}^*) \left( Y_i + H \sum_{j=1}^{i+1} \gamma_{i,j} f(T_j, Y_j) \right).
\]

Thus to compute an MRI-GARK stage, we only need to solve an ODE of dimension \( S \) instead of \( N \). We summarize the simplifications of (2.2) in Definition 3.1 and provide a graphical illustration in Figure 1.

**Definition 3.1 (SM-MRI-GARK).** A surrogate model MRI-GARK (SM-MRI-GARK) method solves the ODE (1.1) with the help of the surrogate model (1.2) using

---

This manuscript is for review purposes only.
**Modified surrogate model integrated exactly**

**Runge–Kutta stages to propagate full model**

**Range of V**

**Nullspace of W**

**Modified surrogate model integrated exactly**

**Fig. 1: Illustration of SM-MRI-GARK for a two-variable ODE and a surrogate model that evolves in a one-dimensional subspace.**

---

**steps of the form**

176 (3.4a) \( Y_1 = y_n, \)

\[
\begin{aligned}
  z_i(0) &= W^*Y_i, \\
  T_i &= t_n + c_i \{ s \} H, \\
  z_i'(\theta) &= \Delta \epsilon_i^{(s)} f_{\text{sur}}(T_i + \Delta \epsilon_i^{(s)} \theta, z_i(\theta)), \\
  &+ \sum_{j=1}^{i+1} \gamma_{i,j} (\theta/\Delta \epsilon_i^{(s)}) (W^*(T_j, Y_j) - f_{\text{sur}}(T_j, W^*Y_j)), \quad \text{for } \theta \in [0, H], \\
  Y_{i+1} &= Vz_i(H) + (I_{d \times d} - VW^*) \left( Y_i + H \sum_{j=1}^{i+1} \gamma_{i,j} f(T_j, Y_j) \right), \\
  i &= 1, \ldots, s^{(s)},
\end{aligned}
\]

178 (3.4b) \( y_{n+1} = Y_{s^{(s)}+1}. \)

---

**Remark 3.2** (SM-MRI-GARK for hierarchical surrogate models). For certain applications, an entire hierarchy of surrogate models \( f, f^{[1]}_{\text{sur}}, f^{[2]}_{\text{sur}}, \ldots, f^{[m]}_{\text{sur}} \) may be available. In such scenarios, one can take advantage of all models by applying SM-MRI-GARK recursively. First, the highest fidelity models \( f \) and \( f^{[1]}_{\text{sur}} \) are applied to (3.4). The ODE (3.4b), is then solved using an SM-MRI-GARK method with \( \Delta \epsilon_i^{(s)} f^{[2]}_{\text{sur}} \) as the surrogate model. Its ODEs are solved using an SM-MRI-GARK method with \( \Delta \epsilon_i^{(s)} f^{[m]}_{\text{sur}} \) as the lowest fidelity surrogate model. Note the power of \( \Delta \epsilon_i^{(s)} \) scaling the surrogate models can be eliminated by applying the change of variable \( \theta \rightarrow \theta/\Delta \epsilon_i^{(s)} \) to (3.4b).

**Remark 3.3** (Connection to EPIRK-K methods). In [?], Sandu showed that exponential Runge–Kutta methods are closely related to MRI-GARK methods when the linear-nonlinear partitioning

\[ y' = \underbrace{L_y}_{f^{(1)}(t,y)} + \underbrace{f(t,y) - L_y}_{f^{(s)}(t,y)} \]

is used. MRI-GARK can be viewed as a generalization of exponential Runge–Kutta
methods where the piece integrated exactly can be linear or nonlinear.

An analogous connection can be made between EPIRK-K [?] and SM-MRI-GARK. EPIRK-K is an extension of exponential propagation iterative methods of Runge-Kutta type (EPIRK) [?] that approximates the Jacobian $J_n = \frac{\partial f}{\partial y}(t_n, y_n)$ appearing in matrix exponentials and related matrix functions with a Krylov-subspace approximation $U^*J_nUU^*$. SM-MRI-GARK with $V = W = U$ and $f_{\text{sur}}(t, y_{\text{sur}}) = U^*J_nUy_{\text{sur}}$ closely resembles the class of EPIRK-K schemes.

Remark 3.4 (Stepsize adaptivity). Even with fixed timesteps $H$, adaptive time-steppers can be used to integrate (3.4b) so that $f_{\text{sur}}$ is evaluated just enough to meet tolerances. It is also possible to independently and adaptively select $H$, and thus, the frequency at which $f$ is evaluated. Following [?, Remark 2.5], (3.4) can be equipped with an embedded method to estimate the local truncation error, and then standard error controllers can be used [?, Section II.4]. Each method in Appendix A includes an embedded method that is one order lower than the main method.

As a simple example of an SM-MRI-GARK method, consider forward Euler as the base method and the internally consistent [?, Section 3.1.1] coupling $\gamma_{1,1}(t) = 1$:

$$
(3.5) \quad z(t) = f_{\text{sur}}(t + \theta, z(\theta)) + W^*f(t_n, y_n) - f_{\text{sur}}(t_n, W^*y_n), \quad \text{for } \theta \in [0, H],
$$

$$
y_{n+1} = Vz(H) + (I_{d \times d} - VW^*) (y_n + Hf(t_n, y_n)).
$$

Equation (3.5) is only first order accurate; however, second and third order SM-MRI-GARK schemes can be constructed using the coefficients provided in Appendix A. In order to implement these efficiently, we have provided pseudocode in Algorithm 3.1. It minimizes the number of full and surrogate model evaluations, as well as the number of matrix-vector products involving $V$ and $W^*$.

3.1. Formulation for SPC-MRI-GARK. A similar process can be used to apply SPC-MRI-GARK (2.4) to the ODE (1.4). For brevity, we avoid replicating the simplifications (3.2) and (3.3) and directly present the method formulation in the following definition.

Definition 3.5 (SM-SPC-MRI-GARK). A surrogate model SPC-MRI-GARK (SM-SPC-MRI-GARK) method solves the ODE (1.1) with the help of the surrogate model (1.2) using steps of the form

$$(3.6a) \quad Y_i = y_n + H \sum_{j=1}^{s^{(s)}} a_{ij}^{(s)} f(t_n + \zeta_j^{(s)}H, Y_j), \quad i = 1, \ldots, s^{(s)},$$

$$(3.6b) \quad \begin{cases}
z(0) = W^*y_n, \\
z(\theta)' = f_{\text{sur}}(t_n + \theta, z(\theta)) \\
+ \sum_{j=1}^{s^{(s)}} \gamma_j(\theta) \left( W^*f(t_n + \zeta_j^{(s)}H, Y_j) - f_{\text{sur}}(t_n + \zeta_j^{(s)}H, W^*Y_j) \right), \\
\text{for } \theta \in [0, H], \\
y_{n+1} = Vz(H) + (I_{d \times d} - VW^*) \left( y_n + H \sum_{j=1}^{s^{(s)}} b_{ij} f(t_n + \zeta_j^{(s)}H, Y_j) \right).
\end{cases}$$
Algorithm 3.1 Pseudocode for explicit SM-MRI-GARK (3.4).

1: procedure SM-MRI-GARK($f, f_{\text{sur}}, V, W^*, t_0, t_{\text{end}}, y_0, N_{\text{steps}}$)
2: \hspace{1em} $y = y_0$
3: \hspace{1em} $\hat{y} = W^* y_0$
4: \hspace{1em} $H = (t_{\text{end}} - t_0)/N_{\text{steps}}$
5: \hspace{1em} for $n = 1, \ldots, N_{\text{steps}}$ do
6: \hspace{2em} for $i = 1, \ldots, s^{(s)}$ do
7: \hspace{3em} $T_i = t_0 + (n + c_i^{(s)})H$
8: \hspace{3em} $k_i = f(T_i, y)$
9: \hspace{3em} $\hat{k}_i = W^* k_i$
10: \hspace{3em} $\hat{\ell}_i = \hat{k}_i - f_{\text{sur}}(T_i, \hat{y})$
11: \hspace{3em} $\hat{w} = \hat{y} + H \sum_{j=1}^{i} \gamma_{i,j} \hat{\ell}_j$
12: \hspace{3em} $y = y + H \sum_{j=1}^{i} \gamma_{i,j} k_j$
13: \hspace{3em} $\hat{y} = \text{OdeSolve} \left( z_i'(\theta) = \Delta c_i^{(s)} f_{\text{sur}}(T_i + \Delta c_i^{(s)} \theta, z_i(\theta)) + \sum_{j=1}^{i} \gamma_{i,j} (\theta \hat{\ell}_j), \text{timespan} = [0, H], \text{initial condition} = \hat{y} \right)$
14: \hspace{3em} $y = y + V (\hat{y} - \hat{w})$
15: \hspace{2em} end for
16: \hspace{1em} end for
17: \hspace{1em} return $y$
18: end procedure

The SM-MRI-GARK Euler method from (3.5) also happens to be an SM-SPC-MRI-GARK method. The base method is again forward Euler, and the coupling is $\gamma_1(t) = 1$. Methods with more than one stage will not coincide, however, as SM-SPC-MRI-GARK has one modified fast ODE per step while SM-MRI-GARK has $s^{(s)}$. Another distinction, which can be seen in Algorithm 3.2, is that SM-SPC-MRI-GARK only requires one matrix-vector product with $V$ per step, while SM-MRI-GARK requires $s^{(s)}$.

4. Error analysis. Consider the edge case where the surrogate model is taken to be the full model and $V = W^* = I_{N \times N}$. The stages of (3.4) simplify to

\[
\begin{align*}
Y_1 &= y_n, \\
\begin{cases}
z_i(0) = Y_i, \\
z_i' = \Delta c_i^{(s)} f_{\text{sur}}(t_n + c_i^{(s)} H + \Delta c_i^{(s)} \theta, z_i), & \text{for } \theta \in [0, H], \\
Y_{i+1} = z_i(H), & i = 1, \ldots, s^{(s)}, \\
y_{n+1} = Y_{s^{(s)}+1}.
\end{cases}
\end{align*}
\]

This is simply the original ODE (1.1) solved between the abscissae of a Runge–Kutta method. With the infinitesimal step assumption, the error is zero. At the other extreme, take the surrogate model to be $f_{\text{sur}}(t, y_{\text{sur}}) = 0_N$. Again, (3.4) can be
Algorithm 3.2 Pseudocode for explicit SM-SPC-MRI-GARK (3.6).

1: procedure SM-SPC-MRI-GARK($f, f_{\text{sur}}, V, W^*, t_0, t_{\text{end}}, y_0, N_{\text{steps}}$)
2: \hspace{1em} $y = y_0$
3: \hspace{1em} $\tilde{y} = W^* y_0$
4: \hspace{1em} $H = (t_{\text{end}} - t_0)/N_{\text{steps}}$
5: \hspace{1em} for $n = 1, \ldots, N_{\text{steps}}$ do
6: \hspace{2em} $t_n = t_0 + nH$
7: \hspace{2em} for $i = 1, \ldots, s^{(s)}$ do
8: \hspace{3em} $k_i = f(t_n + \epsilon_i^{(s)} H, y + H \sum_{j=1}^{i-1} \alpha_{i,j}^{(s)} k_j)$
9: \hspace{3em} $\tilde{k}_i = W^* k_i$
10: \hspace{3em} $\ell_i = k_i - f_{\text{sur}}(t_n + \epsilon_i^{(s)} H, \tilde{y} + H \sum_{j=1}^{i-1} \alpha_{i,j}^{(s)} \tilde{k}_j)$
11: \hspace{2em} end for
12: \hspace{1em} $\hat{w} = \hat{y} + H \sum_{j=1}^{s^{(s)}} \beta_j^{(s)} \tilde{k}_j$
13: \hspace{1em} $\hat{y} = \text{OdeSolve}\left(z'(\theta) = f_{\text{sur}}(t_n + \theta, z(\theta)) + \sum_{j=1}^{s^{(s)}} \gamma_j^{(s)} \hat{\ell}_j,\right.$
14: \hspace{1em} timespan = $[0, H], \text{initial condition} = \hat{y}$
15: \hspace{1em} $y = y + V(\hat{y} - \hat{w})$
16: end for
17: return $y$
18: end procedure

This manuscript is for review purposes only.
we identically recover (2.2) and (2.4). Thanks to these properties, MRI-GARK and SPC-MRI-GARK order conditions are both necessary and sufficient for the surrogate model versions. Interested readers can refer to [?], Section 3.1 and [?], Section 2.2 for these order conditions, which are derived using N-tree theory [?]. Thus, the order of the surrogate-model-based integration only depends on the coefficients of the underlying MRI-GARK method. Notably, the order does not depend on the accuracy or quality of the surrogate model.

We note that these results are based on nonstiff order condition theory which requires \( f \) and \( f_{\text{sur}} \) to be \( p + 1 \) times continuously differentiable and have moderate Lipschitz constants with respect to \( H \). In cases where these assumptions do not hold, e.g., nonsmooth surrogate models or stiff, infinite-dimensional PDEs, order reduction can occur.

We now discuss the leading error constant \( C \) in (4.1). N-Trees provide an elegant way to quantify the terms of which it is composed. For our multirate methods, there are two partitions, so we use the set of two-trees \( T_2 \). Tree vertices are “colored” either fast (black circle) or slow (white circle). Let \( T^{(f)}_2 \) be the set of trees in \( T_2 \) with all vertices fast-colored (including the empty tree), and \( T_2 \setminus T^{(f)}_2 \) the set of trees with at least one slow-colored node. We have that

\[
C = \sum_{\substack{u \in T_2 \setminus T^{(f)}_2 \setminus \{\emptyset\} \\ \rho(u) = p+1}} \frac{1}{\sigma(u)} \left( \Phi(u) - \frac{1}{\gamma(u)} \right) F(u)(y),
\]

(4.2)

\[
T^{(f)}_2 = \left\{ \emptyset, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o,}
its derivatives appearing in an elementary differential. In order to quantify the error in the surrogate model, suppose there exists a (small) constant $\varepsilon$ such that:

$$
\|f(y) - V_{\text{sur}}(W^* y)\| \leq \varepsilon,
$$

for $m = 1, \ldots, p$. Here $\|\|$ denotes an arbitrary norm on $\mathbb{C}^N$. This yields the bounds

$$
\|F(\bullet)(y_n)\| \leq d_0 + \varepsilon, \quad \|F([u_1, \ldots, u_m]^{(1)})\| \leq (d_m + \varepsilon) \prod_{i=1}^m \|F(u_i)(y_n)\|,
$$

$$
\|F(\circ)(y_n)\| \leq \varepsilon, \quad \|F([u_1, \ldots, u_m]^{(s)})\| \leq \varepsilon \prod_{i=1}^m \|F(u_i)(y_n)\|,
$$

where $d_0 = ||f(y_n)||$ and $d_m$ is the operator norm of $f^{(m)}(y_n)$. Now we have that

$$
\|C\| \leq \sum_{u \in \mathbb{T}_1 \setminus \mathbb{T}_2^{(1)}} \frac{1}{\rho(u)} \left( \Phi(u) - \frac{1}{\gamma(u)} \right) \varepsilon^{\rho^{(s)}(u)} (\varepsilon + \max_{i=0,\ldots,p} d_i)^{\rho^{(t)}(u)} = \sum_{i=1}^{p+1} c_i \varepsilon^i,
$$

where $\rho^{(s)}(u)$ is the number of $\sigma$-colored vertices in $u$. The constants $c_i$ only depend on $f$ and the order condition residuals; they are independent of $f_{\text{sur}}, V, W, H$ and $\varepsilon$. As one might expect, the local truncation error decreases as the accuracy of the surrogate model and its derivatives increase.

5. Construction of surrogate models for accelerating time integration. This sections discusses several techniques to construct a surrogate model $f_{\text{sur}}$ and the associated linear operators $V$ and $W^*$ with a computationally favorable balance of accuracy and evaluation cost.

5.1. Reduced-order models (ROMs). There are numerous techniques from the reduced-order modeling community that may be used to generate the surrogate models including proper orthogonal decomposition (POD) [?], Krylov-subspace methods [?], reduced basis methods [?], discrete empirical interpolation method [?], and dynamic mode decomposition [?, ?]. Other multi-resolution methods based on Fourier or wavelet transformation [?] could be used to only capture coarse information about the model.

5.2. Multimesh models. Consider a numerical approach to discretize a partial differential equation (PDE) in space over a computational mesh using, for example, the finite element, difference, or volume method. A surrogate model can come from solving the PDE on a coarser mesh or using a lower order spatial discretization. This provides

| $u$ | $F(u)(y)$ | $\rho(u)$ | $\sigma(u)$ | $\gamma(u)$ |
|-----|-----------|----------|------------|------------|
| $\circ$ | $f(y) - V_{\text{sur}}(W^* y)$ | 1 | 1 | 1 |
| $\bullet$ | $f'(y) V_{\text{sur}}(W^* y) - V f'_{\text{sur}}(W^* y) f_{\text{sur}}(W^* y)$ | 2 | 1 | 2 |
| $\bullet \rho$ | $V f''_{\text{sur}}(W^* y) V f_{\text{sur}}(W^* y), f(y) - V f_{\text{sur}}(W^* y)$ | 3 | 1 | 3 |

Table 1: Examples of trees, elementary differentials, and other tree properties.
an approximation to capture the “shape” of the solution that is cheaper and also enjoys
a less strict Courant–Friedrichs–Lewy (CFL) condition. If the meshes are nested, $W$
is a subset of columns of the identity matrix, and $V$ is a sparse interpolation matrix.
We note that this strategy closely resembles multigrid methods [?]. The relaxation
and prolongation operators, however, have to be chosen such as $W^*V = I_{N \times N}$, such
as the 1D relaxation stencil $[0 \ 1 \ 0]$ and the 1D prolongation stencil $[1/2 \ 1 \ 1/2]$.

5.3. Applying simplifying approximations to the full model. By lineariz-
ing, filtering, averaging, simply ignoring, or otherwise approximating certain terms in
$f$, a computationally inexpensive $f_{\text{sur}}$ can be produced. In [?, Section 6], for exam-
ple, a surrogate model of $z$-level ocean model is produced by a vertical averaging of
barotropic velocities. Consider also a direct N-body simulation with a Barnes-Hut or
fast multipole method used as the surrogate model. For this example, the surrogate
models happen to have the same state representation as the full model, and therefore,
$S = N$ and $V = W = I_{N \times N}$.

5.4. Data-driven surrogate models. When sufficient input and output data is
available, supervised learning approaches are a viable option to generate approximate
models for a system. A variety of techniques have been developed, some depending on
system identification and sparse dictionary learning [?], others using neural networks
to discover operators and right-hand side functions [?]. Patch data, both in time and
space [?] have been used to train machine learning models that can reproduce crude
or partial dynamics of the full system. In some cases, the data driven dynamics reside
in the full space, so $S = N$ and $V = W = I_{N \times N}$. In other cases, the dynamics could
reside in the dominant modes of the data, in which case $S < N$ and $V$ and $W$ are
determined by the method.

6. Numerical experiments. In order to evaluate the new surrogate model
time-steppers, we apply them to a diverse set of ODE test problems equipped with
surrogate models. Two of the test problems are used to verify convergence properties
(error versus steps) and another two are used to measure the integrators’ performance
(error versus runtime). The methods considered in the experiments are SM-MRI-
GARK Euler from (3.5) and the four methods from Appendix A. We compare these
with their base methods, which are traditional Runge–Kutta methods, when only the
full or only the surrogate model is used. In all the experiments, the error is computed
by comparing a numerical solution to a high-accuracy reference solution at the final
time. The solution using only the surrogate model resides in the surrogate model
subspace, so it is multiplied by $V$ before computing error.

6.1. Quasi-geostrophic model and quadratic ROM. The quasi-geostrophic
(QG) equations [?, ?], are a well-studied set of approximations to both atmospheric
and ocean flow. The QG equations have a wide range of well-studied lower dimensional
approximations. For our use case, a quadratic POD-Galerkin ROM makes an excellent
surrogate model.

We follow the formulations of [?, ?], and the same setup as utilized in [?):
\[
\frac{\partial \omega}{\partial t} + J(\psi, \omega) - \text{Ro}^{-1} \frac{\partial \psi}{\partial x} = \text{Re}^{-1} \nabla^2 \omega + \text{Ro}^{-1} F,
\]
\[
\omega = -\nabla^2 \psi,
\]
\[
J(\psi, \omega) = \frac{\partial \psi \partial \omega}{\partial y \partial x} - \frac{\partial \psi \partial \omega}{\partial x \partial y}.
\]
Here $\omega$ represents the vorticity, $\psi$ the streamfunction, $\text{Re} = 450$ the Reynolds number,
and Ro = 0.0036 the Rossby number. The forcing term is selected to be a symmetric double Gyre to simulate flow in the ocean [?]:

\[ F = \sin(\pi(y - 1)) \].

The domain for the problem is \( \Omega = [0, 1] \times [0, 2] \). Homogeneous Dirichlet, boundary conditions are used for all time.

For the spatial discretization, a second order finite difference is performed, with the canonical Arakawa approximation [?] performed on the Jacobian term, \( J \). The domain is discretized using 63 points in the \( x \) direction and 127 points in the \( y \) direction. The discretization begets the discrete stream function variable \( \psi \). The relation between the streamfunction and vorticity is used to define the time derivative of the streamfunction variable \( \partial \psi / \partial t \), which we will take as the PDE of interest.

Following [?], and using the method of snapshots [?], we construct basis transformation operators that capture the time-averaged dominant linear modes of the system. The basis transformations are represented by the orthogonal, in \( L^2 \), linear operators \( V \), and \( W^* \), where \( y_{\text{sur}} = W^* \psi \) represents the basis transformation from streamfunction space into ROM space. Details as to how these are derived are found in [?].

In order to construct a reduced-order model we have to optimally approximate the time derivative of the reduced-order state. As (6.1) is quadratic in nature, we can construct a quadratic POD-Galerkin reduced-order model, of the form

\[ y'_{\text{sur}} = f_{\text{sur}}(t, y_{\text{sur}}) = W^* f(t, V y_{\text{sur}}) = b + A y_{\text{sur}} + y_{\text{sur}}^T B y_{\text{sur}}, \]

where \( y_{\text{sur}} \) is the state in the POD basis, \( f \) is the solution to the Poisson equation in (6.1) for the streamfunction derivative \( \partial \psi / \partial t \), \( b \in \mathbb{R}^S \) corresponds to the forcing term \( Ro^{-1} F \), \( A \in \mathbb{R}^{S \times S} \) corresponds to the linear term \( Ro^{-1} \partial \psi / \partial x + Re^{-1} \nabla^2 \omega \), and \( B \in \mathbb{R}^{S \times S \times S} \) corresponds to the Jacobian term \( -J(\psi, \omega) \). We take \( S \), the dimension of the reduced-order state space, to be significantly smaller than the dimension of the space of the discretized streamfunction, \( N \). It is known from [?] that for this specific spatial discretization, utilizing \( N = 8001 \) points, \( S = 40 \) modes corresponds to 98.1\% of the total kinetic energy of the system, while \( S = 80 \) modes corresponds to 99.4\% of the total energy, meaning that this problem should be extremely well-suited to dimensional reduction.

The timespan for which we run the system is a ten day forecast, which for the given parameters is approximately \( t \in [0, 0.109] \). The ROM was built in the interpolatory regime on the test timespan using 2000 evenly-spaced snapshots. The model and the reduced-order model implementations for the experiments have been taken from [?]. As they are written in MATLAB, we use \texttt{ode45} with absolute and relative tolerances of \( 10^{-11} \) to accurately solve the modified fast ODEs (3.4b) and (3.6b).

Figure 2 shows the convergence of the QG equations with respect to a ROM. We have not plotted the results of the Runge–Kutta methods using only the surrogate model as they produced errors orders of magnitude larger than the other methods. For a fixed number of timesteps, SM-MRI-GARK and SM-SPC-MRI-GARK consistently have less error than Runge–Kutta methods using only the full model. At order three, however, the more accurate ROM of size \( S = 80 \) is needed to achieve a substantial decrease in error.
SM-MRI-GARK using ROM of size 80
SM-MRI-GARK using ROM of size 40
SM-SPC-MRI-GARK using ROM of size 80
SM-SPC-MRI-GARK using ROM of size 40
Runge–Kutta using full model

Fig. 2: Convergence plots for the Quasi-Geostrophic equations (6.1).

6.2. Lorenz ’96 with a machine learning surrogate model. The Lorenz ’96 problem \cite{lorenz1996} is given by

\begin{equation}
X'_k = -X_{k-2}X_{k-1} + X_{k-1}X_{k+1} - X_k + F, \quad k = 1, 2, \ldots, 40,
\end{equation}
with periodic boundary conditions, and a forcing term $F = 8$. Due to its chaotic dynamics, this problem is prominently used in data assimilation and atmospheric research. Readers interested in current literature on machine learning surrogate models based on Lorenz ’96 may refer to [?, ?].

In the following experiments, a neural network model with fully-connected layers was trained to approximate the right-hand side function in (6.2). Therefore the projections are $V = W^* = I_{d \times d}$ and the trained network acts as $f_{\text{sur}}$. The network consists of input and output layers and a hidden layer of dimension 80. The input and hidden layers use the softplus activation function while the output layer does not have any activations. Again, this experiment is implemented in MATLAB and uses \texttt{ode45} with tight tolerances to solve ODEs involving the surrogate model.

Similar to [?], the initial condition $X_{20}(0) = 8.008$ and $X_i(0) = 8$ for $i \neq 20$ is propagated for 2 units of time to expel transient effects and the developed solution is used as the true initial condition. The training data is taken from 5000 equally-spaced solutions of the full model within the interval $t \in [2, 10]$ and, the convergence tests are performed over timespan $t \in [4, 8]$.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{plots/lorenz_convergence.png}
\caption{Convergence plots for the Lorenz ’96 problem (6.2).}
\end{figure}

The convergence results are shown in Figure 3. We note that merely integrating the surrogate model did not produce a stable solution in any of the experiments.
On the other hand, when used together with the full model in SM-MRI-GARK and SM-SPC-MRI-GARK methods, the accuracy of the solution increases significantly compared to the full model solution.

6.3. Brusselator PDE. The test problem BRUS from [7, Section II.10] is based on the Brusselator reaction-diffusion PDE

$$\begin{align*}
\frac{\partial u}{\partial t} &= \alpha \nabla^2 u + 1 + u^2 v - 4.4u, \\
\frac{\partial v}{\partial t} &= \alpha \nabla^2 v + 3.4u - u^2 v,
\end{align*}$$

(6.3)

where \( \alpha = 0.002 \). The spatial domain is the unit square \( 0 \leq x, y \leq 1 \) with zero, Neumann boundary conditions for both \( u \) and \( v \). Using the method of lines, this is discretized with second order central finite difference on a uniform \( P \times P \) grid. Starting at the initial conditions

$$u(t = 0, x, y) = 0.5 + y, \quad v(t = 0, x, y) = 1 + 5x,$$

we seek the solution at \( t = 7.5 \).

In our experiments, \( P = 257 \), and thus, \( N = 2P^2 = 132098 \). For the surrogate model, we take the approach described in subsection 5.2 where we use the same finite difference discretization of (6.3) but on a coarser mesh. In particular, coarse meshes of size \( P = 129 \) and \( P = 65 \) are used, which nest inside the fine mesh. The modified fast ODEs (3.4b) and (3.6b) are solved with one step of a Runge–Kutta method one or two orders higher than the base method. We found that this uses the fewest evaluations of \( f_{\text{sur}} \) while keeping the ODE solution errors negligible.

Comparing the performance of the integrators is our primary goal with the Brus- selator problem, and we implemented the tests in C. For each integrator and surrogate model size, the runtime and error was recorded for a range of eight stepsizes. The results are plotted in Figure 4. At orders one and two, SM-MRI-GARK and SM-SPC-MRI-GARK show clear speedups over Runge–Kutta solutions using only the full model. In addition, the performance increases as the surrogate model mesh becomes finer. Unfortunately, the results are reversed at order three.

Profiling of the code helped to identify why SM-MRI-GARK and SM-SPC-MRI- GARK performed poorly. Table 2 provides a breakdown of the relative costs of four primary operations. When a \( 127 \times 127 \) surrogate model is used, about one third of the runtime is devoted to solving modified fast ODEs, and an evaluation of \( f_{\text{sur}} \) is about 25% as expensive as \( f \). The linear operators \( \mathbf{V} \) and \( \mathbf{W}^* \) have efficient, matrix-free implementations, but are still 75% as expensive as evaluating \( f \). These factors caused one step of SM-MRI-GARK and SM-SPC-MRI-GARK to take approximately twice as long as a traditional Runge–Kutta step. At order three, the reduction in error is not enough to overcome the overheads.

6.4. Advection PDE. Finally, we consider the following 2D advection problem with zero, Dirichlet boundary conditions:

$$\begin{align*}
\frac{\partial u}{\partial t} + a \cdot \nabla u &= 0, \quad \text{on } \Omega = [0, 1] \times [0, 1], \\
u(t, x, y) &= 0, \quad \text{on } \partial \Omega.
\end{align*}$$

The velocity field corresponds to the Molenkamp-Crowley problem:

$$a(x, y) = \begin{bmatrix} 2\pi \left( y - \frac{1}{2} \right), -2\pi \left( x - \frac{1}{2} \right) \end{bmatrix}.$$

This manuscript is for review purposes only.
We start with the initial condition
\[
  u(t = 0, x, y) = \exp\left(-100 \left((x - 0.35)^2 + (y - 0.35)^2\right)\right)
\]
and stop at \( t = 2 \). This allows the initial profile to rotate clockwise twice about the center of the domain.

For the method of lines discretization in space, we represent \( \Omega \) with a uniform, triangular mesh and apply a nodal discontinuous Galerkin (DG) method. The order in space is chosen to match the order in time. This yields the linear ODE
\[
y' = M^{-1} K y,
\]
where \( M \) and \( K \) are mass and advection matrices, respectively. We use the C++ library MFEM \(^1\) for this DG discretization, and in particular, base it on the serial version of example 9 provided in MFEM version 4.1.

Again, we try the multimesh approach by using a mesh size of \( P = 100 \) for \( f \) and a mesh size of \( P = 50 \) for \( f_{\text{sur}} \). We use the same technique discussed in subsection 6.3 for integrating modified fast ODEs involving \( f_{\text{sur}} \). The size of the ODE (6.4) for each experiment is listed in Table 3.

| Order of time and space discretization | Dimension of full model \( N \) | Dimension of surrogate model \( S \) |
|----------------------------------------|---------------------------------|----------------------------------|
| 1                                      | \( 6 \times 10^4 \)             | \( 1.5 \times 10^4 \)             |
| 2                                      | \( 1.2 \times 10^5 \)           | \( 3 \times 10^4 \)              |
| 3                                      | \( 2 \times 10^5 \)             | \( 5 \times 10^4 \)              |

Table 3: Dimensions of the full and surrogate models used in the advection experiment.

In contrast to the Brusselator problem (6.3), the advection problem is linear and hyperbolic. Moreover, profiling reveals an evaluation of the RHS of (6.4) is much more expensive than an interpolation between meshes due, in part, to the linear solve with the mass matrix. Based on the discussion in the previous subsection, we may expect better performance results at order three, and indeed, that is the case.

Figure 5 shows the error and timing for the integrators over a range of eight step-sizes. All SM-MRI-GARK and SM-SPC-MRI-GARK methods outperform Runge–Kutta methods with speedups ranging from approximately 3 to 725. The error for
the Runge–Kutta methods using the surrogate model remains nearly constant, which indicates that spatial errors dominate temporal errors.
TIME-STEPPING WITH A SURROGATE MODEL

7. Conclusions. This work develops a new methodology to accelerate the time integration of large ODEs using surrogate models. Specifically, surrogate information is incorporated into the numerical solution of the original ODEs using multirate methods. We derive and analyze two implementations of this new time-stepping technique: SM-MRI-GARK and SM-SPC-MRI-GARK. Both combine continuous evolution of the surrogate model with discrete Runge–Kutta steps of the full ODE. There are numerous ways to generate surrogate models that offer flexible trade-offs among accuracy, computational cost, and stability. The new methods are designed such that the overall order of accuracy is independent of the surrogate model. The more accurate the surrogate models are, the smaller the local error constants are, which leads to a smaller global error.

Numerical experiments reveal that, at low orders of accuracy, it is possible to achieve orders of magnitude speedups over traditional Runge–Kutta methods. As the order increases, overheads associated with evaluations of $V$, $W^*$, and the surrogate model are penalized more when measuring efficiency, and speedups tend to decrease. There is not a clear winner between SM-MRI-GARK and SM-SPC-MRI-GARK; their relative performance appears to depend heavily on the properties of the full and surrogate models.

Fig. 5: Work precision diagrams for advection problem (6.4).
The methods considered in this paper are based on explicit multirate infinitesimal methods, although the surrogate integration can be done implicitly. Even so, the explicit treatment of the surrogate model error can lead to stepsize restrictions. Instability can occur if a surrogate model does not capture the stiffness of the full model. The authors plan to explore implicit and implicit-explicit methods [?, ?, ?] for stiff problems and differential-algebraic equations. We note that stiffness can appear in the full model, surrogate model, and even the surrogate model error, and each scenario brings different considerations.

Appendix A. Explicit MRI-GARK and SPC-MRI-GARK methods of orders two and three.

Table 4 provides coefficients for new MRI-GARK and SPC-MRI-GARK schemes based on Ralston’s optimal second and third order Runge–Kutta methods [?]. Figure 6 plots their scalar slow stability regions [?, Definition 4.1]

\[ \mathcal{S}^{10}_{\infty, \alpha} = \{ z^{(s)} \in \mathbb{C} \mid |R(z^{(f)}, z^{(s)})| \leq 1, \forall z^{(f)} \in \mathbb{C}^- : |\arg(z^{(f)}) - \pi| \leq \alpha \}, \]

where \( R(z^{(f)}, z^{(s)}) \) is the scalar linear stability function. All methods in Table 4 satisfy \( \lim_{z^{(f)} \to -\infty} R(z^{(f)}, z^{(s)}) = 0 \), so they are suitable for problems where the fast dynamics are stiff, but the slow dynamics are nonstiff. For readers interested in using different base methods, we have provided a Mathematica notebook in the supplementary materials with general, parameterized families of methods.

| Order | Base Method | MRI-GARK \( \Gamma(t), \hat{\gamma}(t) \) | SPC-MRI-GARK \( \gamma(t), \hat{\gamma}(t) \) |
|-------|-------------|---------------------------------|---------------------------------|
| 2     | \begin{bmatrix} 0 & 0 & 0 \\ \frac{3}{4} & \frac{3}{4} & 0 \\ \frac{1}{2} & \frac{3}{4} & \frac{1}{4} \end{bmatrix} | \begin{bmatrix} \frac{3}{4} & 0 \\ -\frac{5}{4} & \frac{1}{4} \\ \frac{1}{2} & \frac{1}{4} \end{bmatrix} | \begin{bmatrix} -\frac{1}{2} + \frac{3t}{2} \\ \frac{3}{4} - \frac{3t}{2} \end{bmatrix} | 1 |
| 3     | \begin{bmatrix} 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{bmatrix} | \begin{bmatrix} \frac{1}{2} & 0 & 0 \\ \frac{11}{36} + \frac{3t}{2} & 3 - \frac{9t}{2} & 0 \\ \frac{47}{36} - \frac{133}{6} & -\frac{1}{6} - \frac{t}{2} & -\frac{8}{9} + \frac{6t}{3} \end{bmatrix} | \begin{bmatrix} 1 - \frac{2t}{3} - \frac{4t^2}{3} \\ -2t + 4t^2 \\ \frac{8t}{3} - \frac{8t^2}{3} \end{bmatrix} | \begin{bmatrix} -\frac{7}{8} + \frac{9t}{5} \\ \frac{71}{40} - \frac{12t}{5} \end{bmatrix} | \begin{bmatrix} \frac{1}{10} - \frac{4t}{10} \end{bmatrix} |

Table 4: Second and third order MRI-GARK and SPC-MRI-GARK coefficients.
(a) Second order MRI-GARK

(b) Third order MRI-GARK

(c) Second order SPC-MRI-GARK

(d) Third order SPC-MRI-GARK

Fig. 6: Stability regions for new methods in Table 4 including the base Runge–Kutta stability region and $S_{\infty,\alpha}$ for $\alpha = 45^\circ, 65^\circ, 85^\circ$.