Introducing time parallelisation within data assimilation

Rishabh Bhatt, Laurent Debreu, Arthur Vidard

▶ **To cite this version:**

Rishabh Bhatt, Laurent Debreu, Arthur Vidard. Introducing time parallelisation within data assimilation. 2025. hal-03540480v3

HAL Id: hal-03540480

https://inria.hal.science/hal-03540480v3

Preprint submitted on 22 Jan 2025

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License

INTRODUCING TIME PARALLELISATION WITHIN DATA ASSIMILATION*

RISHABH BHATT†, LAURENT DEBREU†, AND ARTHUR VIDARD†

Abstract. Four dimensional variational data assimilation (4DVAR), in its incremental formulation, is based on optimisation algorithms which require the integration of the forward and adjoint versions of the original model in order to compute the gradient. For their use on parallel computers, these models are classically parallelised only in spatial dimension and this is a limiting factor on the maximum number of cores that can be utilised. We present here a novel approach of introducing additional time parallelisation using the Parareal algorithm. This approach is used here for integration of the forward model. We use a modified version of the inexact conjugate gradient method where the matrix-vector multiplication is supplied through Parareal. The use of this inexact conjugate gradient and the associated convergence conditions allows to precisely determine the stopping criterion of the Parareal iterations. The results are demonstrated by considering a one dimensional shallow water model. They are presented in terms of the accuracy (in comparison with the original exact conjugate gradient) and in terms of the number of required iterations of the Parareal algorithm.

Key words. Data assimilation, High performance computing, Optimisation, Numerical analysis, Parareal algorithm

MSC codes. 86A22, 68W10, 65L05, 65K10

1 Introduction

Modern computation emphasises reducing the clock time. With the advancement of massively parallel computers, millions of cores can be utilised at once. However, for solving very large time-dependent problems space parallelisation alone becomes insufficient. This is particularly relevant for applications such as weather prediction where even small gains in computational speed are significant.

Leading meteorological institutes like Météo France, ECMWF, Met Office use variational data assimilation algorithms like 3D or 4D-Var [27, 5, 39, 40, 35] to obtain optimal initial states for short to medium-range weather forecasts. Data assimilation [8, 24, 1] is essentially an initial condition control problem which combines all the information from the model trajectory, the observations, and some error statistics to give an optimal initial state of the atmosphere/ocean. The evolving nature of the numerical models and the observations in terms of quality and size poses a challenge to the data assimilation systems to meet the demands for faster and accurate predictions. In ocean data assimilation, the state vector and the observation vector are usually of order $\mathcal{O}(10^6 - 10^9)$ and $\mathcal{O}(10^6)$ respectively. This significantly increases the overall computational cost and a way to deal with it is to improve the use of massively parallel supercomputers.

For instance, ocean models can typically utilise minimum domain sizes of 20×20 grid points, beyond which the simulation time increases due to the communication time becoming predominant over the computation time within each subdomain. A global simulation of the NEMO model [29] using ORCA-R025 grid, which has a horizontal domain of 1442×1021 grid points and 75 vertical levels, would end up using approximately a maximum of 3672 cores. This means that even if the simulation can have access to more cores, after some point the spatial parallelisation would reach saturation and would not allow to effectively use more cores.

* **Funding:** This work was supported by ANR-ADOM (ANR-18-CE46-0008).

†Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LJK, 38000 Grenoble, France (r.bhatt@reading.ac.uk, laurent.debreu@inria.fr, arthur.vidard@inria.fr).

In large time-dependent problems such as the one described in the above example, it is tempting to use the time dimension as an additional source of parallelism. This is one of the main reasons behind turning our attention to the parallel-in-time (PinT) methods [34, 14] in addition to spatial domain decomposition methods. This observation applies to direct modelling, but also, and this is what will be of interest in this paper, to data assimilation algorithms.

Previous works have explored a "time-parallel 4D-Var", with attempts to parallelise weak-constraint 4D-Var in [12] by solving a saddle point problem. A PDE constrained optimisation problem to solve the coupled system of the forward and backward evolution problems with the help of Parareal (a parallel-in-time algorithm) is proposed in [16]. However, a drawback of this approach is that it is not easy to control the initial condition of the model, which is central to data assimilation. Other works have also investigated Parareal in combination with minimisation iterations to solve optimal control problems [32, 9, 28]. Our main objective here is to study the exploitation of time parallelism by coupling Parareal with incremental 4D-Var. We are going to focus specifically on the inner loop of incremental 4D-Var and so we consider the case where our model is *linear*.

The paper is structured as follows. Section 2 talks about the basics of variational data assimilation and Parareal algorithm. Section 3 delves into the coupling of Parareal with incremental 4D-Var, focusing on a minimisation problem involving inexact matrix-vector products. The development of a modified inexact conjugate gradient method is outlined in Section 4. Numerical experiments based on a 1D shallow water model are presented in Section 5. The conclusion follows in Section 6, offering some perspectives on our work to envision more realistic applications. Throughout this article, we will adopt the notations from Ide et al.[22].

2 Background

2.1 Data Assimilation Four-dimensional variational data assimilation, or 4D-Var [39, 40, 35] minimises a cost function based on discrepancies between the model trajectory and the observations, and between the obtained analysis and the previous known background state. For the model state vector $\mathbf{x}$ we consider the following discrete non-linear dynamical model

$$\begin{aligned} \mathbf{x}_0 &= \mathbf{x}(t_0) \\ \mathbf{x}_i &= F_i(\mathbf{x}_{i-1}) \qquad i = 1, 2, \ldots, N \end{aligned} \tag{2.1}$$

where $F_i : \mathbb{R}^n \to \mathbb{R}^n$ is the model operator describing the state evolution from time t_{i-1} to t_i, and N is the total number of time windows. Given a prior estimate/background state $\mathbf{x}^b \in \mathbb{R}^n$ and a set of observations $\mathbf{y}_i^o \in \mathbb{R}^m$ at time t_i, the 4D-Var cost function is written as

$$J(\mathbf{x}_0) = \frac{1}{2} (\mathbf{x}_0 - \mathbf{x}^b)^T \mathbf{B}^{-1} (\mathbf{x}_0 - \mathbf{x}^b) + \frac{1}{2} \sum_{i=0}^{N} (H_i(\mathbf{x}_i) - \mathbf{y}_i^o)^T \mathbf{R}_i^{-1} (H_i(\mathbf{x}_i) - \mathbf{y}_i^o) \tag{2.2}$$

Here $H_i : \mathbb{R}^n \to \mathbb{R}^m$ is the observation operator mapping the evaluated model state $\mathbf{x}_i$ to the observation $\mathbf{y}_i^o$ at the correct time t_i. The errors in the background and the observations are characterised by the corresponding covariance matrices $\mathbf{B}$ and $\mathbf{R}_i$.

2.2 Incremental 4D-Var formulation Incremental 4D-Var introduced in [6] is an algorithm obtained by linearising the model and observation operators around

the model state. For a sequence of linearisation states $\mathbf{x}^{(0)}, \mathbf{x}^{(1)}, \dots \mathbf{x}^{(l)}$ with corresponding increments $\delta\mathbf{x}^{(0)}, \delta\mathbf{x}^{(1)}, \dots \delta\mathbf{x}^{(l)}$ the following tangent linear hypothesis (a first order Taylor's expansion) is expressed as

$$\begin{aligned} F(\mathbf{x}^{(l)} + \delta\mathbf{x}^{(l)}) &\approx F(\mathbf{x}^{(l)}) + \mathbf{F}\,\delta\mathbf{x}^{(l)} \\ H(\mathbf{x}^{(l)} + \delta\mathbf{x}^{(l)}) &\approx H(\mathbf{x}^{(l)}) + \mathbf{H}\,\delta\mathbf{x}^{(l)} \end{aligned} \tag{2.3}$$

where $\mathbf{F}$ and $\mathbf{H}$ are the linearisation of the model and observation operators respectively around the linearisation state $\mathbf{x}^{(l)}$; that is,

$$\mathbf{F} = \frac{\partial F}{\partial \mathbf{x}}\Big|_{\mathbf{x}^{(l)}} \qquad \mathbf{H} = \frac{\partial H}{\partial \mathbf{x}}\Big|_{F(\mathbf{x}^{(l)})} \tag{2.4}$$

The control variable now changes from the initial model state $\mathbf{x}_0$ to the initial increment $\delta\mathbf{x}_0$ of the model state. Using the above assumption we get our cost function

$$\begin{aligned} J(\delta\mathbf{x}_0) = \frac{1}{2}\{\delta\mathbf{x}_0 - (\mathbf{x}^b - \mathbf{x}_0)\}^T\mathbf{B}^{-1}\{\delta\mathbf{x}_0 - (\mathbf{x}^b - \mathbf{x}_0)\} + \\ \frac{1}{2}\sum_{i=0}^{N}(\mathbf{H}_i\delta\mathbf{x}_i - \mathbf{d}_i)^T\mathbf{R}_i^{-1}(\mathbf{H}_i\delta\mathbf{x}_i - \mathbf{d}_i) \end{aligned} \tag{2.5}$$

where $\mathbf{d}_i = H_i(\mathbf{x}_i) - \mathbf{y}_i$ is called the innovation vector or just departures. Clearly one has to run the non-linear model integration and store the trajectory to compute the departures since the non-linear model M is embedded within $\delta\mathbf{x}_i$. The increment vector evolves in time through the *tangent linear model*, obtained by linearising the discrete non-linear model

$$\delta\mathbf{x}_{i+1} = \frac{\partial F[\mathbf{x}(t_i)]}{\partial \mathbf{x}}\delta\mathbf{x}_i = \mathbf{F}_i\delta\mathbf{x}_i \tag{2.6}$$

In a nutshell, the incremental 4D-Var involves running a high resolution model in the outer loop, which retains all the physics, to compare the model state trajectory with observations for cost function evaluation. The inner loop involves running a low resolution model with simplified physics, to minimise the cost function [35]. For an algorithmic implementation of incremental 4D-Var we refer to [25].

2.3 Parareal Method Parareal initially introduced in [26] and later reformulated [2] is a parallel-in-time method where the solution at a given time step is computed independently of the solution at previous time steps. For a general class of ODE

$$\mathbf{x}'(t) = f(t, \mathbf{x}(t)), \quad t \in (0, T], \quad \mathbf{x}(0) = \mathbf{x}_0 \tag{2.7}$$

the strategy relies on partitioning the time domain $[0, T]$ into N time windows or sub-intervals $[t_i, t_{i+1}]$ of length $\Delta T = T/N$. Within each such sub-interval, a two-level grid system is defined consisting of:

- a cheap coarse propagator $G(t_{i+1}, t_i, \mathbf{x}_i)$ which is fast but gives a rough solution at t_{i+1} from the initial condition $\mathbf{x}_i$.
- a fine propagator $F(t_{i+1}, t_i, \mathbf{x}_i)$ which is computationally expensive but gives very accurate solution (in theory the exact solution).

The Parareal algorithm proceeds iteratively by computing the solution $\mathbf{x}_{i+1}^k \approx \mathbf{x}(t_{i+1})$ at the sub-intervals for given $\mathbf{x}_i^k$ at time t_i and iteration k. An initial configuration is

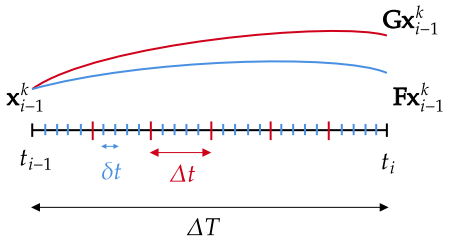

FIG. 1. *Illustration of the Parareal algorithm for a sub-interval* $[t_i, t_{i+1}]$. *For a Parareal iterate* $\mathbf{x}_i^k$ *at iteration* k, $G(t_{i+1}, t_i, \mathbf{x}_i^k)$ *gives solution by coarse propagator* G *using time-step* Δt. $F(t_{i+1}, t_i, \mathbf{x}_i^k)$ *gives solution by fine propagator* F *with time step* δt.

obtained by a serial run of the coarse propagator G over the whole time domain. So, the initial Parareal iteration $k = 0$ looks like

$$\mathbf{x}^0_{i+1} = G(t_{i+1}, t_i, \mathbf{x}^0_i) \tag{2.8}$$

With the solution now available at all the grid points, the fine propagator F is run to provide more accurate solution. As F is expensive, all the heavy fine computations performed by F are done in *parallel* by assigning a core to each sub-interval. The iterates are updated by a correction procedure

$$\begin{aligned} \mathbf{x}^{k+1}_0 &= \mathbf{x}_0 \\ \mathbf{x}^{k+1}_{i+1} &= \underbrace{G(t_{i+1}, t_i, \mathbf{x}^{k+1}_i)}_{\text{Prediction}} + \underbrace{F(t_{i+1}, t_i, \mathbf{x}^k_i) - G(t_{i+1}, t_i, \mathbf{x}^k_i)}_{\text{Correction}} \end{aligned} \tag{2.9}$$

Parareal can be seen as a predictor-corrector algorithm where each iteration predicts with G and corrects with the difference between the solutions from F and G at the previous iteration. An in-depth discussion of the convergence properties of Parareal can be found in [11, 3, 17, 15, 36].

2.3.1 Speedup We briefly discuss about the speedup of Parareal algorithm which we will be used as a criteria to check the performance of our experiments. The theoretical speedup S is defined as the ratio of the time taken to perform a task sequentially to the time to do the same task in parallel,

$$S = \frac{\text{Serial computation time}}{\text{Parallel computation time}} \tag{2.10}$$

Assuming N cores are available, each assigned to one time window, and negligible communication time between any two cores, the speedup is bounded by [33, 37]

$$S \leq \min\left(\frac{N}{k}, \frac{\text{fine time}}{\text{coarse time}}\right) \tag{2.11}$$

The bound (2.11) shows that the coarse propagator should be really cheap and fast in order to get a good speed up but at the same time there will be more Parareal iterations which will reduce the speedup according to the first bound. Thus the optimal speedup must balance the two bounds in the best possible way.

3 Introducing time parallelisation The inner loop of the incremental 4D-Var remains the least scalable component due to the use of lower resolution models [23]. This limitation arises from having fewer grid points available for distributing across cores when producing the analysis increment. In fact, parallelising the inner loop is by no means trivial. The most obvious choice for minimisation within the inner loop is the conjugate gradient method, one of the Krylov-subspace based methods. It is constructed on the principle of finding higher level Krylov subspaces at each iteration, relying on the initial gradient (or residual) of the cost function. These gradients are updated at the end of the iteration and are used to build the next Krylov subspace [13]. However these iterations are inherently sequential, leaving parallelising the computations within the iteration as the only alternative to parallelising the minimisation iterations themselves.

Our approach is as follows. Each minimisation iteration of an incremental 4D-Var (or inner loop) cycle begins with a forward integration of the linear model (2.6) to calculate the quadratic cost function value. An immediate way to introduce time parallelisation is to use Parareal algorithm for the forward model integration. To see how this could be done, let us suppose the length of one data assimilation cycle to be T which is made up of N time windows. Consider the case of a linear forward model for the state vector $\delta\mathbf{x} \in \mathbb{R}^n$

$$\begin{aligned} \delta\mathbf{x}_0 &= \delta\mathbf{x}(t_0) \\ \delta\mathbf{x}_{i+1} &= \mathbf{F}_i \delta\mathbf{x}_i, \quad i = 1, \dots N \end{aligned} \tag{3.1}$$

The model operator $\mathbf{F}_i : \mathbb{R}^n \to \mathbb{R}^n$ is the same as the one used in (2.6) and so

$$\mathbf{F}_i = \frac{\partial F}{\delta \mathbf{x}}\Big|_{\mathbf{x}=\mathbf{x}_i} \tag{3.2}$$

For the sake of simplicity we remove the background term $\mathbf{x}^b$ from the data assimilation cost function since it is not affected by time integration. We also assume that there is only one observation $\mathbf{y} \in \mathbb{R}^n$ of the whole state vector at the end of integration time $t_N = T$. Doing so the observation operator H is an identity map for each time t_i and the minimisation problem is well posed. As a further simplification, the covariance matrix $\mathbf{R}_i$ is set to be equal to the identity matrix. Our cost function can be written as

$$J(\delta\mathbf{x}_0) = \frac{1}{2} \left\| \left(\prod_{i=1}^{N} \mathbf{F}_i \right) \delta\mathbf{x}_0 - \mathbf{y} \right\|_2^2 \tag{3.3}$$

Remark 3.1. In the rest of this manuscript instead of writing the composition of the discrete model operators $\mathbf{F}_i$ at a model state $\delta\mathbf{x}_i$ as $\prod_{i=1}^{N} \mathbf{F}_i$ each time we are going to use the block $\mathbf{F}^N$ for $\mathbf{F}_1\mathbf{F}_2 \cdots \mathbf{F}_N$ for convenience and it is just an abuse of the notation.

Thus we have,

$$J(\delta\mathbf{x}_0) = \frac{1}{2} \|\mathbf{F}^N \delta\mathbf{x}_0 - \mathbf{y}\|_2^2 \tag{3.4}$$

with gradient,

$$\nabla J(\delta\mathbf{x}_0) = (\mathbf{F}^N)^T (\mathbf{F}^N \delta\mathbf{x}_0 - \mathbf{y}) \tag{3.5}$$

where $(\mathbf{F}^N)^T$ is the adjoint of the operator $\mathbf{F}^N$. To carry out computation of the forward model by a Parareal based operator, we accordingly define a Parareal based integrator $\mathbf{P}(k)$ over $[0, T]$ such that

$$\mathbf{P}(k)\,\delta\mathbf{x}_0 = \delta\mathbf{x}_N^k \tag{3.6}$$

which acts on an initial condition $\delta\mathbf{x}_0$ and gives Parareal solution at the last time window N after k iterations. The gradient of the cost function is then approximated by

$$\begin{aligned}\nabla J(\delta\mathbf{x}_0) &\approx (\mathbf{F}^N)^T\left(\mathbf{P}(k)\,\delta\mathbf{x}_0 - \mathbf{y}\right)\\ &\stackrel{\text{def}}{=} \widetilde{\nabla J}(\delta\mathbf{x}_0)\end{aligned} \tag{3.7}$$

Note that we kept the same adjoint for the gradient i.e. $(\mathbf{F}^N)^T$ and not the adjoint of $\mathbf{P}(k)$. Now minimising the original cost function in (3.4) requires finding $\delta\mathbf{x}_0$ such that gradient $\nabla J(\delta\mathbf{x}_0)$ is 0. This equates to solving the linear system

$$\mathbf{A}\delta\mathbf{x} = \mathbf{b} \tag{3.8}$$

where $\mathbf{A} = (\mathbf{F}^N)^T\mathbf{F}^N,\ \mathbf{b} = (\mathbf{F}^N)^T\mathbf{y}$. Similarly, the corresponding approximate linear system writes

$$\mathbf{A}_{\text{para}}\delta\mathbf{x} = \mathbf{b} \tag{3.9}$$

where $\mathbf{A}_{\text{para}} = (\mathbf{F}^N)^T\ \mathbf{P}(k)$.

Remark 3.2. In realistic applications, matrix $\mathbf{H}$ is typically low-rank rather than the identity, meaning we observe only a subset of the full state variable. This affects the conditioning of the minimisation which can be improved by adding more observations at different times. Additionally, we need to introduce observation-error covariance matrix $\mathbf{R}$ to account for uncertainties. If observations $\mathbf{y}_i$ are available at times t_i, our approach can be extended to this general case by solving the linear system $\mathbf{A}\mathbf{x} = \mathbf{b}$ where

$$\mathbf{A} = \frac{1}{p}\sum_{i=1}^{p}(\mathbf{H}_i\mathbf{F}^{N_i})^T\mathbf{R}_i^{-1}\mathbf{H}_i\mathbf{P}^i(k), \qquad \mathbf{b} = \frac{1}{p}\sum_{i=1}^{p}(\mathbf{H}_i\mathbf{F}^{N_i})^T\mathbf{R}_i^{-1}\mathbf{y}_i \tag{3.10}$$

and $\mathbf{F}^{N_i}, \mathbf{P}^i(k)$, $\mathbf{R}_i$, and $\mathbf{H}_i$ are evaluated at time t_i.

The exact linear system $\mathbf{A}\mathbf{x} = \mathbf{b}$ is usually solved using the conjugate gradient (CG) method since the matrix $\mathbf{A}$ is symmetric. But as the corresponding approximate matrix $\mathbf{A}_{\text{para}} = (\mathbf{F}^N)^T\mathbf{P}(k)$ is non-symmetric, there is a big possibility that CG will not give the desired solution and thus the correct gradient value of our cost function. We address this challenge by using an *inexact Krylov subspace method* [19, 4] which is based on a counter-intuitive principle that the error in the matrix-vector multiplication is allowed to grow as iterations progress [38, 42]. By doing so, same accuracy levels for the minimisation process can be maintained as expected with the usual (exact) Krylov subspace methods. In the next section we discuss about the inexact conjugate gradient method, a kind of inexact Krylov subspace method.

4 Inexact conjugate gradient method

In this section we talk about the inexact conjugate gradient method proposed by Gratton et al. [20] and then introduce our own modified version of this method with the Parareal operator $\mathbf{P}(k)$ embedded in it.

4.1 Formulation The inexact conjugate gradient method (hereafter referred to as *inexact CG*) provides a way of allowing inaccuracies in matrix-vector multiplications when solving convex optimisation problem. The method monitors the decrease in the quadratic $q(\delta\mathbf{x}) = \frac{1}{2}\,\delta\mathbf{x}^T\mathbf{A}\delta\mathbf{x} - \mathbf{b}^T\delta\mathbf{x}$. The quadratic change is directly linked to the energy norm of the residual $\mathbf{r}(\delta\mathbf{x}_j) = \mathbf{A}\delta\mathbf{x}_j - \mathbf{b}$ [20, §2], offering an improved stopping minimisation criterion in terms of avoiding under- or over-solving.

Let $\mathbf{E}_j$ denote the error in the matrix $\mathbf{A}$ at inexact CG iteration (hereafter referred to as *icg-iteration*) j such that for corresponding conjugate direction vector $\mathbf{p}_j$ we have access to the matrix-vector product $(\mathbf{A} + \mathbf{E}_j)\,\mathbf{p}_j$. The presence of error $\mathbf{E}_j$ results in the computed residual $\mathbf{r}_j$ being different from $\mathbf{r}(\delta\mathbf{x}_j)$. Let the size of perturbation matrix size $\mathbf{E}_j$ be measured in the primal-dual norm defined as

$$\|\mathbf{E}_j\|_{\mathbf{A}^{-1},\mathbf{A}} = \sup_{\delta\mathbf{x}\neq 0} \frac{\|\mathbf{E}_j\delta\mathbf{x}\|_{\mathbf{A}^{-1}}}{\|\delta\mathbf{x}\|_{\mathbf{A}}} = \|\mathbf{A}^{-1/2}\mathbf{E}_j\mathbf{A}^{-1/2}\|_2 \tag{4.1}$$

For some permissible bound on the error norm $\|\mathbf{E}_j\|_{\mathbf{A}^{-1},\mathbf{A}}$ at icg-iteration j, if the inexact residual norm $\|\mathbf{r}_j\|_{\mathbf{A}^{-1}}$ and the residual gap norm $\|\mathbf{r}(\delta\mathbf{x}_j) - \mathbf{r}_j\|_{\mathbf{A}^{-1}}$ can be suitably bounded, then the change in the quadratic can be controlled. The following theorem captures the essence of this idea.

THEOREM 4.1 (From [20]). *Let $\epsilon > 0$ and let $\boldsymbol{\phi} = (\phi_0, \phi_1, \cdots, \phi_{j-1})^T \in \mathbb{R}^j$ be a positive vector satisfying*

$$\sum_{i=1}^{j} \frac{1}{\phi_i} \leq 1 \tag{4.2}$$

Suppose that

$$\|\mathbf{E}_i\|_{\mathbf{A}^{-1},\mathbf{A}} \leq \omega_i = \frac{\sqrt{\epsilon}\,\|\mathbf{b}\|_{\mathbf{A}^{-1}}\|\mathbf{p}_i\|_{\mathbf{A}}}{2\,\phi_{i+1}\|\mathbf{r}_i\|_2^2 + \sqrt{\epsilon}\,\|\mathbf{b}\|_{\mathbf{A}^{-1}}\|\mathbf{p}_i\|_{\mathbf{A}}} \tag{4.3}$$

for all $i \in \{0, ..., j-1\}$, with $\mathbf{p}_i$ being the conjugate direction at icg-iteration i. Then

$$\|\mathbf{r}(\delta\mathbf{x}_j) - \mathbf{r}_j\|_{\mathbf{A}^{-1}} \leq \frac{\sqrt{\epsilon}}{2}\|\mathbf{b}\|_{\mathbf{A}^{-1}} \tag{4.4}$$

Additionally if

$$\|\mathbf{r}_j\|_{\mathbf{A}^{-1}} \leq \frac{\sqrt{\epsilon}}{2}\|\mathbf{b}\|_{\mathbf{A}^{-1}} \tag{4.5}$$

then

$$|q(\delta\mathbf{x}_j) - q(\delta\mathbf{x}_*)| \leq \epsilon|q(\delta\mathbf{x}_*)|. \tag{4.6}$$

Remark 4.2. Due to increasing error in the matrix-vector product, search directions $\mathbf{p}_j$ are no longer conjugate to each other and the (inexact) residuals lose their orthogonality. While the theorem assumes that the inexact residual norm $\|\mathbf{r}_j\|_{\mathbf{A}^{-1}}$ eventually becomes smaller, it is not guaranteed. Adding a reorthogonalisation step could ensure that the residuals converge to zero after at most n steps.

4.2 Controlling error with Parareal stopping criteria In Theorem 4.1, the residual gap is kept under controlled precision provided that $\|\mathbf{E}_j\|_{\mathbf{A}^{-1},\mathbf{A}}$ is bounded by the quantity ω_j. In our case the $\mathbf{E}$ matrix, by construction, at any Parareal iteration k is given as

$$\begin{aligned}\mathbf{E}(k) &= \mathbf{A}_{\text{para}} - \mathbf{A} \\ &= (\mathbf{F}^N)^T\mathbf{P}(k) - (\mathbf{F}^N)^T\mathbf{F}^N \\ &= -(\mathbf{F}^N)^T(\mathbf{F}^N - \mathbf{P}(k))\end{aligned} \tag{4.7}$$

It can be said that ω_j acts as a threshold for how large an error can be allowed at a particular icg-iteration j by controlling the number of Parareal iterations k. As it will be seen in our numerical experiments (section 5), the use of $\|\mathbf{E}_j(k)\|_{\mathbf{A}^{-1},\mathbf{A}}$ does not provide a refined enough stopping criterion for our application. We show in the following that it is preferable to bound the product of the error matrix $\mathbf{E}$ with the vector $\mathbf{p}_j$ and that an estimate for this product can be easily obtained in the context of Parareal. In the proof of Theorem 4.1, the bound (4.4) is obtained by using the inequality

$$\|\mathbf{E}_j\mathbf{p}_j\|_{\mathbf{A}^{-1}} \leq \|\mathbf{E}_j\|_{\mathbf{A}^{-1},\mathbf{A}}\|\mathbf{p}_j\|_{\mathbf{A}} \leq \omega_j\|\mathbf{p}_j\|_{\mathbf{A}} \tag{4.8}$$

Upon observation, we found that utilising $\|\mathbf{E}_j\mathbf{p}_j\|_{\mathbf{A}^{-1}}$ instead of $\|\mathbf{E}_j\|_{\mathbf{A}^{-1},\mathbf{A}}$ for the Parareal stopping criterion gives much better results. To bound our new norm we introduce a new quantity ξ_j which satisfies,

$$\|\mathbf{E}_j\mathbf{p}_j\|_{\mathbf{A}^{-1}} \leq \omega_j\|\mathbf{p}_j\|_{\mathbf{A}} = \xi_j \tag{4.9}$$

Thus from (4.3) the value of ξ_j can be obtained as

$$\xi_j = \frac{\sqrt{\epsilon}\,\|\mathbf{b}\|_{\mathbf{A}^{-1}}\|\mathbf{p}_j\|_{\mathbf{A}}^2}{2\boldsymbol{\phi}_{j+1}\|\mathbf{r}_j\|_2^2 + \sqrt{\epsilon}\,\|\mathbf{b}\|_{\mathbf{A}^{-1}}\|\mathbf{p}_j\|_{\mathbf{A}}} \tag{4.10}$$

The following theorem encapsulates the modifications discussed above. The proof is almost identical to the proof of theorem 1 in [20], and uses lemma 2 of the same paper.

THEOREM 4.3. *In Theorem* 4.1*, condition* (4.3) *is replaced by*

$$\|\mathbf{E}_j(k)\|_{\mathbf{A}^{-1},\mathbf{A}} = \sup_{\delta\mathbf{x}\neq 0} \frac{\|\mathbf{E}_j(k)\,\delta\mathbf{x}\|_{\mathbf{A}^{-1}}}{\|\delta\mathbf{x}\|_{\mathbf{A}}} \leq \xi_j/\|\mathbf{p}_j\|_{\mathbf{A}} \tag{4.11}$$

However, using the fact that

$$\|\mathbf{E}_j(k)\mathbf{p}_j\|_{\mathbf{A}^{-1}} \leq \|\mathbf{E}_j(k)\|_{\mathbf{A}^{-1},\mathbf{A}}\|\mathbf{p}_j\|_{\mathbf{A}} \tag{4.12}$$

we can get a tighter bound, while retaining the validity of the theorem.

4.3 Other feasible approximations So far the method has been discussed from a theoretical point of view with an assumption that the quantities or norms concerning the matrix $\mathbf{A}$ and its inverse are accessible. But realistically one might not even have access to the matrix $\mathbf{A}$, let alone its inverse. Thus in practice Gratton et al. [20, §3.2-3.4] proposed the following approximations:

- $\|\mathbf{p}_j\|_{\mathbf{A}} \approx \sqrt{\frac{1}{n}\operatorname{Tr}(\mathbf{A})}\,\|\mathbf{p}_j\|_2$
- $q(\delta\mathbf{x}_j) \approx q_j \overset{\text{def}}{=} -\frac{1}{2}\mathbf{b}^T\delta\mathbf{x}_j$

- $\|\mathbf{b}\|_{\mathbf{A}^{-1}} = \sqrt{2|q(\delta\mathbf{x}_j)|} \approx \begin{cases} \dfrac{\|\mathbf{b}\|_2}{\mu_{\max}(\mathbf{A})}, & j = 0 \\ \sqrt{2\,|q_j|}, & j = 1, \cdots, j_{\max} \end{cases}$
- Termination criterion (4.5) by $q_{j-d} - q_j \leq \frac{1}{4}\,\epsilon|q_j|$ for some integer d

Remark 4.4. The quality of approximation for q has been discussed in Sec 3.3 of Gratton et al. In the presence of matrix-vector product errors, the equality

$$q(\delta\mathbf{x}_j) = -\tfrac{1}{2}\mathbf{b}^T\delta\mathbf{x}_j \tag{4.13}$$

may no longer hold. However, $q(\delta\mathbf{x}_j)$ remains close to q_j if the inexactness of the matrix-vector products are controlled in the same way as in Theorem 4.1 (see Sec 3.3, Theorem 2).

Since we have a modified stopping criterion involving $\mathbf{A}^{-1}$ norm, we also require an approximation for $\|\mathbf{E}_j\mathbf{p}_j\|_{\mathbf{A}^{-1}}$

4.4 Approximation to $\|\mathbf{E}_j\mathbf{p}_j\|_{\mathbf{A}^{-1}}$ We found that we can replace $\|\mathbf{E}_j\mathbf{p}_j\|_{\mathbf{A}^{-1}}$ by another equivalent and computable quantity as proved in the theorem below.

THEOREM 4.5. *If* $\mathbf{F}$ *is invertible,* $\|\mathbf{E}_j(k)\mathbf{p}_j\|_{\mathbf{A}^{-1}} = \|\mathbf{P}(k)\mathbf{p}_j - \mathbf{F}^N\mathbf{p}_j\|_2$ *irrespective of the choice of Parareal approximation.* $\mathbf{P}(k)\mathbf{p}_j$ *is the Parareal approximation at iteration* k *with* $\mathbf{p}_j$ *as the initial condition and* $\mathbf{F}^N\mathbf{p}_j$ *is the corresponding exact solution.*

Proof. We have,

$$\begin{aligned}
\|\mathbf{E}_j(k)\mathbf{p}_j\|_{\mathbf{A}^{-1}} &= \left\langle \mathbf{E}_j(k)\mathbf{p}_j, \mathbf{A}^{-1}\mathbf{E}_j(k)\mathbf{p}_j \right\rangle \\
&= \Big\langle (\mathbf{F}^N)^T\mathbf{F}^N - (\mathbf{F}^N)^T\mathbf{P}(k)]\,\mathbf{p}_j,\, [(\mathbf{F}^N)^T\mathbf{F}^N]^{-1} \\
&\qquad\qquad [(\mathbf{F}^N)^T\mathbf{F}^N - (\mathbf{F}^N)^T\mathbf{P}(k)]\,\mathbf{p}_j \Big\rangle \\
&= \left\langle (\mathbf{F}^N)^T[(\mathbf{F}^N - \mathbf{P}(k))\,\mathbf{p}_j\,],\, [(\mathbf{F}^N)^T\mathbf{F}^N]^{-1}\,(\mathbf{F}^N)^T[(\mathbf{F}^N - \mathbf{P}(k))\,\mathbf{p}_j\,] \right\rangle \\
&= \left\langle (\mathbf{F}^N - \mathbf{P}(k))\,\mathbf{p}_j\,,\, \mathbf{F}^N[(\mathbf{F}^N)^T\mathbf{F}^N]^{-1}\,(\mathbf{F}^N)^T[(\mathbf{F}^N - \mathbf{P}(k))\,\mathbf{p}_j\,] \right\rangle \\
&= \left\langle (\mathbf{F}^N - \mathbf{P}(k))\,\mathbf{p}_j\,,\, [\mathbf{F}^N(\mathbf{F}^N)^\dagger]\,[(\mathbf{F}^N - \mathbf{P}(k))\,\mathbf{p}_j\,] \right\rangle
\end{aligned}$$

where $(\mathbf{F}^N)^\dagger$ is the Moore-Penrose generalised inverse or pseudo-inverse of $\mathbf{F}^N$. Thus when $\mathbf{F}$ is invertible (i.e. $(\mathbf{F}^N)^\dagger = (\mathbf{F}^N)^{-1}$) we have

$$\begin{aligned}
\|\mathbf{E}_j(k)\mathbf{p}_j\|_{\mathbf{A}^{-1}} &= \left\langle (\mathbf{F}^N - \mathbf{P}(k))\,\mathbf{p}_j\,,\, (\mathbf{F}^N - \mathbf{P}(k))\,\mathbf{p}_j\,) \right\rangle \\
&= \left\langle \mathbf{F}^N\mathbf{p}_j - \mathbf{P}(k)\mathbf{p}_j\,,\, \mathbf{F}^N\mathbf{p}_j - \mathbf{P}(k)\mathbf{p}_j \right\rangle \\
&= \|\mathbf{P}(k)\mathbf{p}_j - \mathbf{F}^N\mathbf{p}_j\|_2
\end{aligned}$$

□

In the general case there is no guarantee that $\mathbf{F}$ is invertible so we can only say that $\|\mathbf{P}(k)\mathbf{p}_j - \mathbf{F}^N\mathbf{p}_j\|_2$ may be a good approximation for $\|\mathbf{E}_j\mathbf{p}_j\|_{\mathbf{A}^{-1}}$.

We managed to replace a quantity which involves the $\mathbf{A}^{-1}$ norm to a quantity which needs the 2-norm. Obviously, what remains is to find an approximation for $\mathbf{F}^N\mathbf{p}_j$ since running the fine solver $\mathbf{F}$ each time is computationally not feasible. The idea is to approximate $\|\mathbf{E}_j\mathbf{p}_j\|_{\mathbf{A}^{-1}}$ as

$$\|\mathbf{E}_j\mathbf{p}_j\|_{\mathbf{A}^{-1}} = \|\mathbf{P}(k)\mathbf{p}_j - \mathbf{F}^N\mathbf{p}_j\|_2 \approx \|\mathbf{P}(k)\mathbf{p}_j - \mathbf{P}(k+1)\mathbf{p}_j\|_2 \tag{4.14}$$

This is achieved by computing the difference between successive Parareal iterates in the 2-norm. As the Parareal iteration k increases the iterate $\mathbf{P}(k+1)\mathbf{p}_j$ gets closer and closer to $\mathbf{F}^N\mathbf{p}_j$. Consequently, we only have to run Parareal to a point where the Parareal stopping criterion is met (bounded by the value of ξ_j). That is, at each icg-iteration j we increment the Parareal iterations k by 1 until the condition $\|\mathbf{P}(k)\mathbf{p}_j - \mathbf{P}(k+1)\mathbf{p}_j\|_2 \leq \xi_j$ is satisfied. However, as a trade-off for using an approximation, we end up using one more Parareal iteration than needed since the latest Parareal iterate $\mathbf{P}(k+1)\mathbf{p}_j$ replaces $\mathbf{F}^N\mathbf{p}_j$ to compute the difference with the last Parareal iterate $\mathbf{P}(k)\mathbf{p}_j$ (see Algorithm 4.1).

Algorithm 4.1 Approximation for $\mathbf{F}^N\mathbf{p}_j$

Given: icg-iteration j, direction vector $\mathbf{p}_j$
Run 2 Parareal iterations for initial state $\mathbf{p}_j$
Set $k = 1$
while true **do**
  Compute $e = \|\mathbf{P}(k)\mathbf{p}_j - \mathbf{P}(k+1)\mathbf{p}_j\|_2$
  if $e > \xi_j$ **then**
    Run one more Parareal iteration and set $k = k + 1$
  else
    break
  end if
end while
Store the Parareal iterate $\mathbf{P}(k)\mathbf{p}_j$ for computing the matrix-vector product

4.5 Inaccuracy budget Until now we have not talked about the role of ϕ_j in Theorem 4.1, which can be used to manage the inaccuracy budget [20, §3.1]. By inaccuracy budget we mean adjusting the error tolerance ω_j for icg-iteration j by distributing the *unused inaccuracy* between the error bound ξ_j and the actual error norm $\|\mathbf{E}_j\mathbf{p}_j\|_{\mathbf{A}^{-1}}$ to subsequent icg-iterations. This allows for a larger error in the matrix-vector product and thus using fewer number of Parareal iterations.

Suppose for a given ϕ_{j+1} we obtain $\|\mathbf{E}_j\mathbf{p}_j\|_{\mathbf{A}^{-1}} = \hat{\xi}_j \leq \xi_j$. The corresponding ξ_j can be obtained from $\hat{\xi}_j(\hat{\phi}_{j+1})$ in (4.10) as

$$\hat{\phi}_{j+1} = \frac{(\|\mathbf{p}_j\|_{\mathbf{A}} - \hat{\xi}_j)}{\hat{\xi}_j} \frac{\sqrt{\epsilon}\|\mathbf{b}\|_{\mathbf{A}^{-1}}\|\mathbf{p}_j\|_{\mathbf{A}}}{2\|\mathbf{r}_j\|_2^2} > \phi_{j+1} \tag{4.15}$$

We distribute the inaccuracy evenly in the remaining $j_{\max} - j - 1$ icg-iterations the same way as in the original method [20] with $j_{\max}$ being the maximum icg-iterations allowed. Algorithm 4.2 implements inexact CG using all the manipulations done by including the inaccuracy budget, other feasible approximations, and Parareal stopping criterion involving $\|\mathbf{E}_j\mathbf{p}_j\|_{\mathbf{A}^{-1}}$. In the next section we are going to analyse the results of the Parareal and data assimilation coupling using various settings of the minimisation algorithm.

5 Numerical experiments

We remind the reader that the primary goal of our approach is to improve the theoretical speedup (as introduced in subsection 2.3.1) of the inner loop in incremental 4D-Var using Parareal. This is accomplished by using a modified Parareal stopping criterion within inexact CG.

In this section, we numerically demonstrate how the theoretical speedup is improved by running a series of minimisation experiments with different versions of CG and inexact CG. Their performance is compared in terms of the total number of

Algorithm 4.2 Inexact CG with all approximations

Given: Symmetric positive definite matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$, right hand side vector $\mathbf{b} \in \mathbb{R}^n$, tolerance ϵ_{icg}
Set $\delta\mathbf{x}_0 = 0$, $\mathbf{r}_0 = -\mathbf{b}$, $\mathbf{p}_0 = \mathbf{r}$, $\beta_0 = \|\mathbf{b}\|_2^2$, $\mathbf{u}_1 = \mathbf{b}/\beta_0$, $\phi_0 = j_{\max}$, $\Phi_0 = 1$
for $j = 0, \cdots j_{\max}$ **do**
  Compute the approximation to $\|\mathbf{p}_j\|_{\mathbf{A}}$
  Determine ξ_j from the equation (4.10)
  for $k = 1, \cdots, N-1$ **do**
    Run Parareal with $k+1$ iterations
    Calculate $e = \|\mathbf{P}(k+1)\mathbf{p}_j - \mathbf{P}(k)\mathbf{p}_j\|_2$
    if $e < \xi_j$ **then**
      Set $\hat{\xi}_j = e$
      break
    end if
  end for
  Compute the product $\mathbf{c}_j = (\mathbf{F}^N)^T\mathbf{P}(k)\mathbf{p}_j$
  Find $\hat{\phi}_j$ from $\hat{\xi}_j$
  $\Phi_{j+1} = \Phi_j - \hat{\phi}_j^{-1}$
  if $j < j_{\max}$ **then**
    $\phi_{j+1} = (j_{\max} - j)/\Phi_{j+1}$
  else
    $\phi_{j+1} = \phi_j$
  end if
  $\alpha_j = \beta_j/\mathbf{p}_j^T\mathbf{c}_j$
  $\delta\mathbf{x}_{j+1} = \delta\mathbf{x}_j + \alpha_j\mathbf{p}_j$
  $\mathbf{r}_{j+1} = \mathbf{r}_j + \alpha_j\mathbf{c}_j$
  $J_{j+1} = \frac{1}{2}\mathbf{b}^T\delta\mathbf{x}_{j+1}$
  if $(J_{j+1-d} - J_{j+1}) \le \frac{1}{4}\,\epsilon_{\text{icg}}|J_{j+1}|$ **then**
    break
  end if
  if (reorth) **then**
    for $i = 1, \cdots, j$ **do**
      $\mathbf{r}_{j+1} = \mathbf{r}_{j+1} - (\mathbf{u}_i^T\mathbf{r}_{j+1})\mathbf{u}_i$
    end for
    $\beta_{j+1} = \mathbf{r}_{j+1}^T\mathbf{r}_{j+1}$
    $\mathbf{u}_{j+1} = \mathbf{r}_{j+1}/\sqrt{\beta_{j+1}}$
  else
    $\beta_{j+1} = \mathbf{r}_{j+1}^T\mathbf{r}_{j+1}$
  end if
  $\mathbf{p}_{j+1} = -\mathbf{r}_{j+1} + (\beta_{j+1}/\beta_j)\mathbf{p}_j$
end for

Parareal iterations and the average Parareal iterations per CG/inexact CG iterations. The results are presented for the following versions:

- CG with exact matrix-vector products.
- CG with inexact matrix-vector products (through Parareal).
- Inexact CG with original Parareal stopping criterion.
- Inexact CG with modified Parareal stopping criterion.
- Inexact CG with all approximations, modified Parareal stopping criterion, (subsections 4.3 and 4.4), and inaccuracy budget (subsection 4.5).

To carry out these experiments, we first set up our numerical model, construct the Parareal propagators $\mathbf{F}$ and $\mathbf{G}$, and define the data assimilation cost function.

5.1 Numerical setup Before proceeding, we make the reader aware that we are going to use the notation "$\mathbf{x}$" for the increment "$\delta\mathbf{x}$" throughout this section for simplicity. For illustrations we use the linearised one-dimensional shallow water equa-

tions [21, 18, 41], a simple model describing some important geophysical phenomena. The equations are given by

$$\begin{aligned} \frac{\partial \eta}{\partial t} &= -H \frac{\partial u}{\partial x} \\ \frac{\partial u}{\partial t} &= -g \frac{\partial \eta}{\partial x} + \mu \frac{\partial^2 u}{\partial x^2} \end{aligned} \tag{5.1}$$

Here u is horizontal velocity, η is free surface elevation, g is gravity, H is a constant reference depth, and μ is the diffusion constant. An explicit diffusion term is added to account for the unresolved small scales in terms of the resolved large scales also known as the closure of the system [7, 43]. The physical domain $\Lambda = [0, L]$ is closed and the physical values are provided in Table 1.

TABLE 1
Domain and physical parameters for the linearised 1D shallow water model

Shallow water parameters	**values**
Length of the basin	$L = 120\text{m}$
Reference depth	$H = 0.9\text{m}$
Gravity	$g = 10\text{m.s}^{-2}$
Diffusion constant	$\mu = 0.15\text{m}^2\text{.s}^{-1}$

5.1.1 Spatial discretisation For our model (5.1) we consider a staggered grid discretisation [10] with Δx as the spatial stepping size and N_x as the total number of spatial grid points. Using a second-order centered finite difference scheme in space leads to an ODE system

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{dt}} = \mathbf{C}\,\mathbf{x} \tag{5.2}$$

where

(5.3)
$$\mathbf{C} = \left(\begin{array}{cccc|cccc} & & & & -H/\Delta x & & & \\ & & & & H/\Delta x & -H/\Delta x & & \\ & & \mathbf{0} & & & \ddots & \ddots & \\ & & & & & & H/\Delta x & -H/\Delta x \\ & & & & & & & H/\Delta x \\ \hline g/\Delta x & -g/\Delta x & & & -2\mu/\Delta x^2 & \mu/\Delta x^2 & & \\ & g/\Delta x & -g/\Delta x & & \mu/\Delta x^2 & -2\mu/\Delta x^2 & \mu/\Delta x^2 & \\ & & \ddots & \ddots & & \ddots & \ddots & \ddots \\ & & g/\Delta x & -g/\Delta x & & & \mu/\Delta x^2 & -2\mu/\Delta x^2 \end{array}\right)$$

and $\mathbf{x} = \left(\eta_{1/2}, \ldots \eta_{N_x-1/2}, u_1, \ldots u_{N_x-1}\right)^T$ is the state vector. In the numerical experiments, we will use number of grid points N_x equal to 120 ($\Delta x = L/N_x = 1\text{m}$). The dimension of the state vector is thus equal to $n = 2N_x - 1 = 239$.

5.1.2 Temporal discretisation and Parareal propagators Now to solve the linear ODE system (5.2) numerically in time, we use an implicit theta scheme. Let q be the index for the temporal grid and ΔT be the time-stepping size. We have

$$\mathbf{x}^{q+1} = [\,\mathbf{I} - \theta \Delta T \mathbf{C}\,]^{-1}\,[\,\mathbf{I} + (1-\theta)\Delta T \mathbf{C}\,]\,\mathbf{x}^q, \qquad q = 0, 1, \cdots \tag{5.4}$$

where $0 \leq \theta \leq 1$. Next, we construct the fine and coarse propagators $\mathbf{F}$ and $\mathbf{G}$ respectively for implementing the Parareal algorithm. Let $\Omega = [0, T]$ be the time

domain such that T is the total time period of integration, N is the total number of time windows and ΔT is the length of one time window. For each time window, let N_{fine} and N_{coarse} be the number of fine and coarse time steps respectively. Clearly N_{coarse} has to divide N_{fine}. If δt is the fine step length, then the coarse step length Δt can be calculated as

$$\Delta t = \frac{\Delta T}{N_{\text{coarse}}} = \frac{\delta t \times N_{\text{fine}}}{N_{\text{coarse}}} \tag{5.5}$$

Also, $T = N_{\text{coarse}}\Delta t N = N_{\text{fine}}\delta t N$. The fine and coarse propagators are defined as

$$\begin{aligned} \mathbf{F} &= \Big\{[\mathbf{I} - \theta\delta t\mathbf{C}]^{-1}[\mathbf{I} + (1-\theta)\delta t\mathbf{C}]\Big\}^{N_{\text{fine}}} \\ \mathbf{G} &= \Big\{[\mathbf{I} - \theta\Delta t\mathbf{C}]^{-1}[\mathbf{I} + (1-\theta)\Delta t\mathbf{C}]\Big\}^{N_{\text{coarse}}} \end{aligned} \tag{5.6}$$

In our experiments the fine time step δt is defined based on the criterion:

$$\delta t \max|\lambda_{\mathbf{C}}| \leq \omega_{\max} \tag{5.7}$$

where $\lambda_{\mathbf{C}}$ is an eigenvalue of $\mathbf{C}$ and $\omega_{\max}$ is some constant. The parameter values used for Parareal are put in Table 2.

TABLE 2
Parareal parameters for the linearised 1D shallow water model

Parareal parameters	**values**
Number of time windows, N	20
Theta parameter, θ	0.51
Number of fine time steps per time window, N_{fine}	100
Number of coarse time steps per time window, N_{coarse}	20
Fine time step, δt	0.05s
Coarse time step, Δt	0.25s
Courant number, $\omega_{\max}$	0.29
Total time period of integration, T	100s

Since an explicit diffusion term ($\mu\frac{\partial^2 u}{\partial x^2}$) is included in our 1D model, it is important to examine how varying the diffusion constant μ affects Parareal's convergence. Figure 2 shows the impact of μ on the total Parareal iterations required to solve system (5.2). Note that μ must be chosen such that

$$\mu\frac{\delta t}{\delta x} \leq \frac{1}{2} \tag{5.8}$$

where the quantity on the left is the parabolic Courant number. We find that Parareal requires 15 iterations when no diffusion is applied ($\mu = 0$), compared to 6 Parareal iterations when $\mu = 1$. Figure 3 further shows the impact of diffusion values on the solution at the end of integration time. As expected, increasing μ reduces the Parareal iterations but also strongly damps the solution. For our experiments, $\mu = 0.15$ strikes a balance between reducing Parareal iterations and introducing only a slight damping (dotted line in Figure 3). This choice corresponds to the parabolic Courant number equal to 0.0075.

5.1.3 Data assimilation cost function Having defined $\mathbf{x} = (\boldsymbol{\eta}, \mathbf{u})^T$ and $\mathbf{F}$, we can finally write our data assimilation cost function as

$$\min\ J(\mathbf{x}_0) = \frac{1}{2}\left\|\mathbf{F}^N\mathbf{x}_0 - \mathbf{y}\right\|_2^2 + \frac{\alpha}{2}\left\|\frac{\mathrm{d}\mathbf{x}_0}{\mathrm{d}x}\right\|_2^2 \tag{5.9}$$

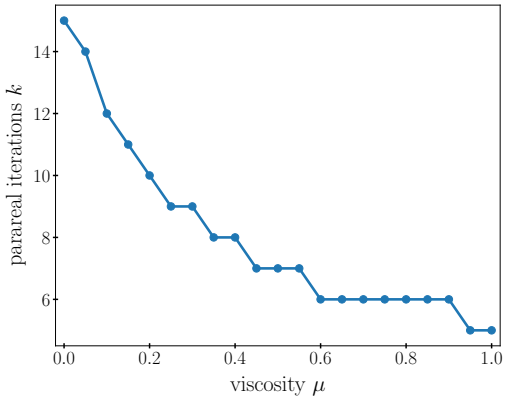

FIG. 2. *Total Parareal iterations when the diffusion constant μ varies in the 1d model. A sharp decline is seen in the Parareal iterations when μ increases before reaching a point where the Parareal iterations almost remain unchanged.*

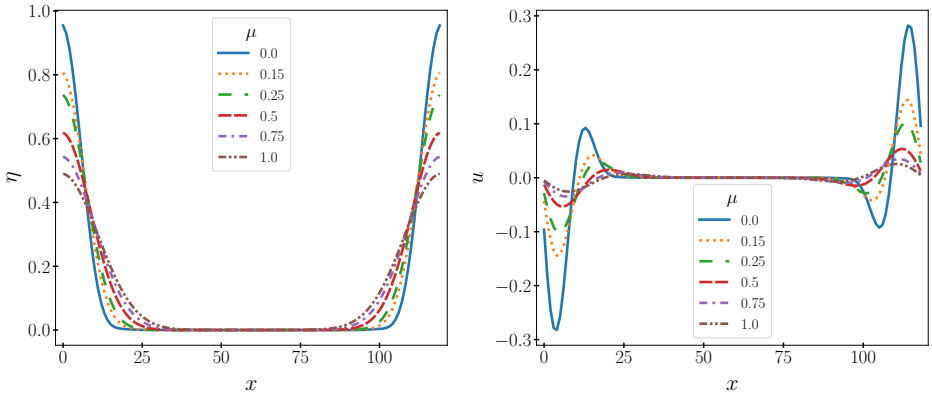

FIG. 3. *Effect of varying μ on the model solution at the end of integration time for free surface η (left) and velocity u (right). As expected, a significant damping is observed when a higher μ is used on the model.*

where we use a regularisation term with regularisation constant α to improve the conditioning of $\mathbf{A} = (\mathbf{F}^N)^T\mathbf{F}^N$. Without regularisation and with the computational parameters given in Table 1 and Table 2, $\kappa(\mathbf{A}) \approx 2.98 \times 10^{18}$ which is extremely large. The following parameters values are used for data assimilation as shown in Table 3.

TABLE 3
Data assimilation parameters for the linearised 1D shallow water model

Data assimilation parameters	**values**
Regularisation constant	$\alpha = 10^{-5}$
Initial free surface value	$\eta_0(x) = \exp\left\{\left[(x - L/2)/(L/15)\right]^2\right\}$
Initial velocity value	$u_0(x) = 0\ \forall x$

Remark 5.1. A standard regularisation term could have been $\alpha\|\mathbf{x}_0\|$, but our pri-

mary interest is to explore the model dynamics using Parareal and how it impacts the minimisation without constraining the initial state. Instead we use a regularisation term based on the spatial second derivative of the initial state to improve the conditioning of the Hessian. It also ensures the smoothness or stability of the numerical experiments by controlling high-frequency variation or abrupt spatial or temporal changes in the solution.

The conditioning of the regularised cost function after using the regularisation constant given in Table 3 is $\kappa(\mathbf{A}) \simeq 10^6$. The gradient also changes to

$$\nabla J(\mathbf{x}_0) = (\mathbf{F}^N)^T(\mathbf{F}^N\mathbf{x}_0 - \mathbf{y}) + \alpha\mathbf{Q}_2\mathbf{x}_0 \tag{5.10}$$

where $\mathbf{Q}_2 = \dfrac{1}{\Delta x^2}\mathbf{Q}_1^T\mathbf{Q}_1$ and

$$\mathbf{Q}_1 = \left(\begin{array}{cc|cc} -1 & 1 & & \\ \ddots & \ddots & & \\ & -1 & 1 & \\ & & -1 & \end{array} \;\middle|\; \mathbf{0} \right) \left(\begin{array}{cc|cc} & & -1 & 1 \\ & & \ddots & \ddots \\ & & & -1 \\ \mathbf{0} & & & \end{array}\right) \tag{5.11}$$

Each diagonal block matrix in $\mathbf{Q}_1$ is of size $N_x - 2 \times N_x - 2$ and so $\mathbf{Q}_1, \mathbf{Q}_2 \in \mathbb{R}^{2(N_x-2)\times 2(N_x-2)}$. Therefore, $\mathbf{A} = (\mathbf{F}^N)^T\mathbf{F}^N + \alpha\mathbf{Q}_2$.

In the subsequent subsections, we demonstrate how the performance of minimisation changes when transitioning from CG without Parareal to CG with Parareal, and finally to inexact CG with Parareal. We also show how the incorporation of feasible approximations and the adoption of a modified Parareal stopping criterion within inexact CG significantly enhance Parareal's performance. The results for inexact CG follow from Theorem 4.1. Specifically once the quadratic change $q(\mathbf{x}_j) - q(\mathbf{x}_*)$ is bounded by the required precision level (4.6), the corresponding $\mathbf{x}_j$ represents the retrieved optimal initial state.

5.2 Conjugate gradient (CG) We begin by examining the performance of CG using the 1D shallow water model with exact matrix-vector multiplication is used. Note that this corresponds to using CG *without Parareal*. For a given CG tolerance ϵ_{cg}, the minimisation ends when

$$\|\mathbf{r}(\mathbf{x}_j)\|_2 \leq \epsilon_{\text{cg}}. \tag{5.12}$$

Figure 4 below shows the evolution of the residual $\|\mathbf{r}(\mathbf{x}_j)\|_2$ (solid line) with respect to the CG iterations for $\epsilon_{\text{cg}} = 10^{-4}$ (dashed line). It can be seen that the minimisation terminates after 24 iterations.

5.3 Conjugate gradient with Parareal Our next step is to evaluate the performance of CG when Parareal is introduced. In this case, the matrix-vector products are computed using Parareal, with their accuracy depending on the *fixed* Parareal stopping tolerance ϵ_{p}. It is important to note that the exact Parareal precision required to match the performance of CG without Parareal (exact matrix-vector product) is not known beforehand.

To determine the optimal ϵ_{p}, a hit-and-trial strategy is followed by running CG with arbitrary Parareal tolerances. For simplicity, we choose $\epsilon_{\text{p}} = 10^{-6}$, which corresponds to the minimum of the $\{\xi_j\}_j$ values produced by inexact CG (as discussed in

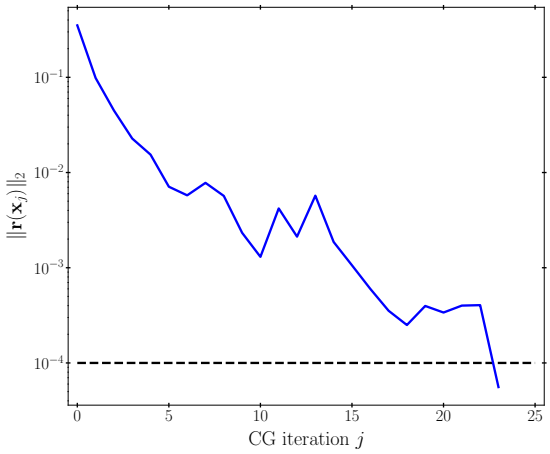

FIG. 4. *CG when matrix-vector products are exact and Parareal is not used. Minimisation terminates after 24 iterations when the residual norm* $\|\boldsymbol{r}(\mathbf{x}_j)\|_2$ *(solid) becomes less than the chosen tolerance* $\epsilon_{\text{cg}} = 10^{-4}$ *(dashed).*

subsection 5.3.3). It is important to emphasise note that this choice is used solely for comparison purposes and is not practically available, as it requires a prior run of inexact CG. Figure 5 shows the result of minimisation when $\epsilon_{\text{cg}} = 10^{-4}$ and $\epsilon_{\text{p}} = 10^{-6}$. The minimisation ends after 24 CG iterations with a total of 191 Parareal iterations. This corresponds to an average of 7.96 Parareal iterations per CG iteration, which will

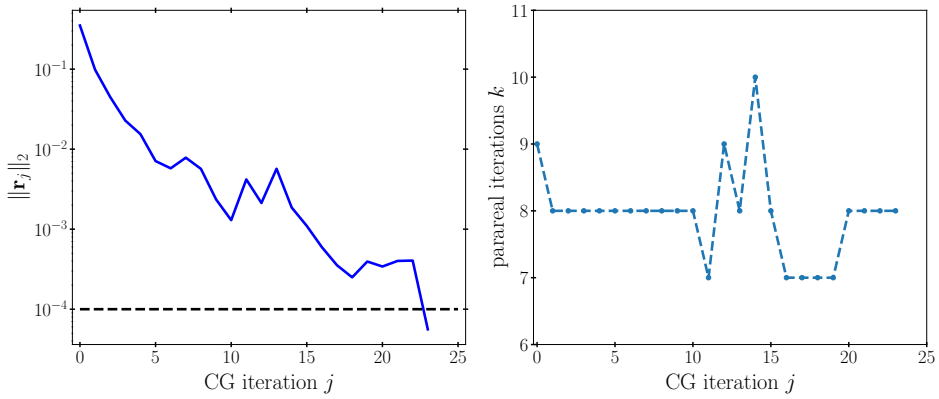

FIG. 5. *CG when matrix-vector products come from Parareal prescribed a fixed Parareal tolerance* ϵ_{p}*. Left image shows identical residual norm values as in Figure 4 when CG tolerance* $\epsilon_{\text{cg}} = 10^{-4}$ *(dashed) and* $\epsilon_{\text{p}} = 10^{-6}$ *are used. Right image shows the evolution of Parareal iterations with CG iterations.*

be shown to be slightly higher than the average of 6.25 Parareal iterations (as in subsection 5.3.3) when using inexact CG. This result highlights the need for an *adaptive* Parareal within the inexact CG framework, demonstrating its advantage over fixed ϵ_{p} obtained through hit-and-trial.

To further assess CG performance in terms of the relative error in the 2-norm of

the residual and the total number of Parareal iterations, Figure 6 presents the results for CG with $\epsilon_p = 10^{-5}, 10^{-6}$ and 10^{-7} are used. The relative error in the 2-norm of the residual is defined as

$$\text{Relative error in the 2-norm of residual} = \frac{\left| \|\mathbf{r}_j\|_2 - \|\mathbf{r}(\mathbf{x}_j)\|_2 \right|}{\left| \|\mathbf{r}(\mathbf{x}_j)\|_2 \right|}$$

Initially, the relative error is low and nearly identical for all three cases. However, after a few CG iterations it begins to vary, becoming smaller for smaller ϵ_{p}. The relative error also gradually increases with each CG iteration, suggesting that a lower Parareal precision is needed near the end of the minimisation. As expected, a smaller ϵ_{p} implies more Parareal iterations and vice-versa.

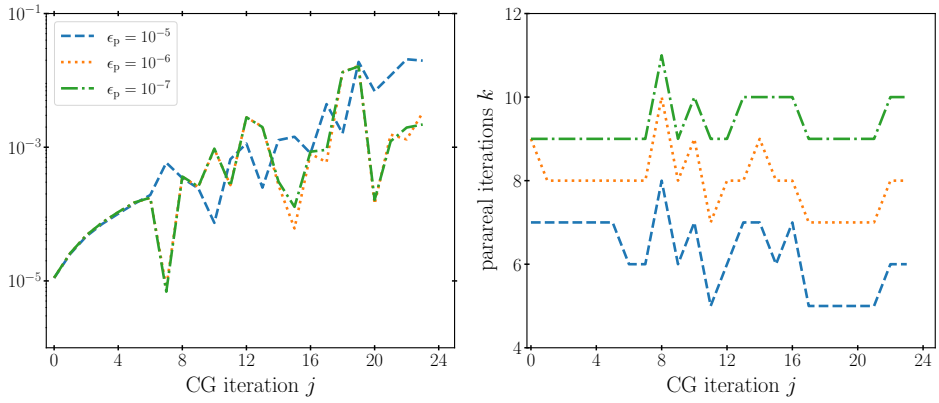

FIG. 6. *Relative error in the 2-norm of the residual (left) and the corresponding Parareal iterations k (right) for fixed values of $\epsilon_{\text{p}} = 10^{-5}$ (dashed), $\epsilon_{\text{p}} = 10^{-6}$ (dotted), and $\epsilon_{\text{p}} = 10^{-7}$ (dashdot) when using CG.*

5.3.1 Inexact conjugate gradient using Parareal We now present the results of our experiments using inexact CG, with the CG results serving as a reference for comparison. As noted earlier, CG and inexact CG use different stopping criteria: CG uses the 2-norm of the residual, while inexact CG relies on the $\mathbf{A}^{-1}$ norm (see Theorem 4.1). To ensure both methods follow the same stopping criterion we see from (4.5) that inexact CG terminates when, for a given value of ϵ_{icg},

$$\|\mathbf{r}_j\|_{\mathbf{A}^{-1}} \leq \frac{\sqrt{\epsilon_{\text{icg}}}}{2} \|\mathbf{b}\|_{\mathbf{A}^{-1}}. \tag{5.13}$$

Rearranging the terms for equality, we find

$$\epsilon_{\text{icg}} = \left(\frac{2\,\|\mathbf{r}_j\|_{\mathbf{A}^{-1}}}{\|\mathbf{b}\|_{\mathbf{A}^{-1}}} \right)^2. \tag{5.14}$$

To determine the value of ϵ_{icg} corresponding to the CG stopping tolerance, we use the value of $\|\mathbf{r}_j\|_{\mathbf{A}^{-1}}$ at convergence (last CG iteration) (from Figure 5) and deduce ϵ_{icg} using equation (5.14). In this case, $\epsilon_{\text{icg}} = 1.12 \times 10^{-7}$.

We first show the results of inexact CG when the original Parareal stopping criterion ($\|\mathbf{E}_j\|_{\mathbf{A}^{-1},\mathbf{A}}$ bounded by ω_j in (4.3)) is used in Figure 7.

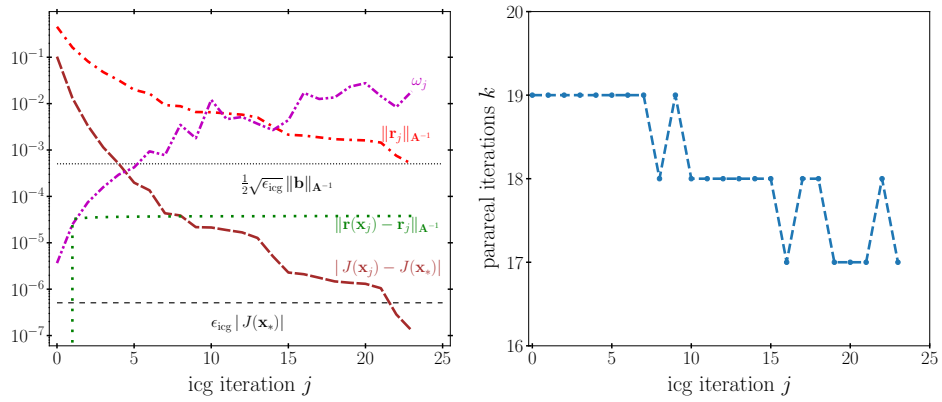

FIG. 7. *Inexact CG following Gratton et al. with original Parareal stopping criterion* $\|\boldsymbol{E}_j\|_{\boldsymbol{A}^{-1},\boldsymbol{A}} \leq \omega_j$. *The left image shows the residual norm* $\|\boldsymbol{r}_j\|_{\boldsymbol{A}^{-1}}$ *(dashdotted), inexact CG stopping tolerance* $1/2\sqrt{\epsilon_{icg}}\|\boldsymbol{b}\|_{\boldsymbol{A}^{-1}}$ *(dotted), bound for the error norm* ω_j *(densely dashdotted), residual gap norm* $\|\boldsymbol{r}(\mathbf{x}_j)-\boldsymbol{r}_j\|_{\boldsymbol{A}^{-1}}$ *(loosely dotted), and the cost function change* $|J(\mathbf{x}_j)-J(\mathbf{x}_*)|$ *(densely dashed) which is bounded by the quantity* $\epsilon_{\text{icg}}|J(\mathbf{x}_*)|$ *(dashed). The right image shows the evolution of corresponding Parareal iterations* k *with icg-iterations* j. *No approximations are used.*

Remark 5.2. Throughout the manuscript we will adhere to the same line style coding for quantities used in Figure 7 across different inexact CG configurations.

As a reminder from Theorem 4.1 inexact CG terminates when both

$$\|\mathbf{r}_j\|_{\mathbf{A}^{-1}} \leq \frac{\sqrt{\epsilon_{\text{icg}}}}{2}\|\mathbf{b}\|_{\mathbf{A}^{-1}} \quad \text{and} \quad \|\mathbf{r}(\mathbf{x}_j)-\mathbf{r}_j\|_{\mathbf{A}^{-1}} \leq \frac{\sqrt{\epsilon_{\text{icg}}}}{2}\|\mathbf{b}\|_{\mathbf{A}^{-1}} \tag{5.15}$$

and we have $|J(\mathbf{x}_j)-J(\mathbf{x}_*)| \leq \epsilon_{\text{icg}}|J(\mathbf{x}_*)|$. From Figure 7 that at the end of the minimisation the conditions of Theorem 4.1 are well satisfied. The minimisation ends after 24 icg-iterations involving a total of 436 Parareal iterations. On average, 18.2 Parareal iterations per icg-iteration are used which is almost equal to the total number of time windows ($N = 20$). This implies negligible speedup since convergence is achieved only during the second last or last iteration. The use of $\|\mathbf{E}_j\|_{\mathbf{A}^{-1},\mathbf{A}}$ for deriving the Parareal stopping criterion leads to a suboptimal algorithm.

5.3.2 Illustration of the modified Parareal criterion Here we demonstrate the significant reduction in the number of Parareal iterations when switching from the original to the modified Parareal stopping criterion for inexact CG. Figure 8 below compares the two criteria for a particular icg-iteration j. The left image on uses the original criterion (involving $\|\mathbf{E}_j\|_{\mathbf{A}^{-1},\mathbf{A}}$), while the right image uses the modified criterion (involving $\|\mathbf{E}_j\mathbf{p}_j\|_{\mathbf{A}^{-1}}$). A substantial difference is evident: only 6 Parareal iterations are required with the modified criterion compared to 19 Parareal iterations with the original criterion. If we look at the definition,

$$\|\mathbf{E}_j\|_{\mathbf{A}^{-1},\mathbf{A}} = \sup_{\delta\mathbf{x}\neq 0} \frac{\|\mathbf{E}_j\delta\mathbf{x}\|_{\mathbf{A}^{-1}}}{\|\delta\mathbf{x}\|_{\mathbf{A}}} \tag{5.16}$$

we observe that $\|\mathbf{E}_j\|_{\mathbf{A}^{-1},\mathbf{A}}$ does not involve the conjugate direction vector $\mathbf{p}_j$ as a product with $\mathbf{E}_j$. Theoretically, as j increases we would expect $\|\mathbf{p}_j\|_{\mathbf{A}} \to 0$, which should lead to decreasing $\|\mathbf{E}_j\mathbf{p}_j\|_{\mathbf{A}^{-1}}$ values. However, since $\mathbf{p}_j$ is absent in the original criterion, the relatively high values present the most pessimistic case.

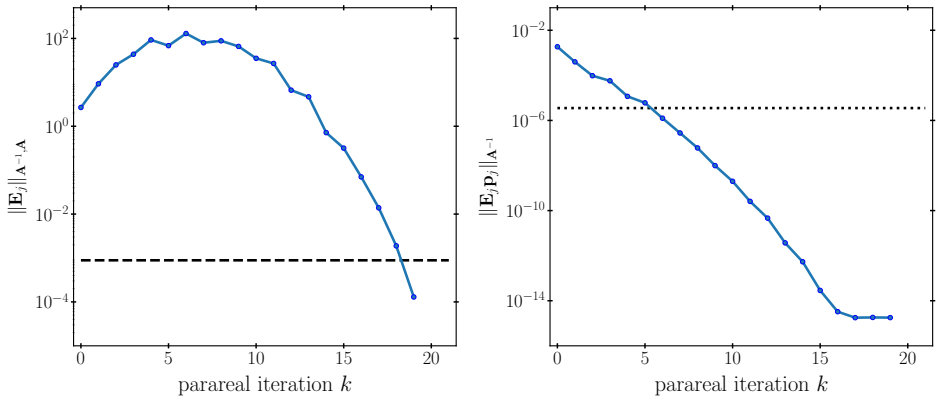

FIG. 8. *Comparison of original (left) and modified (right) Parareal stopping criterion. The norm values (solid) satisfy the corresponding tolerances ω_j (dashed) and ξ_j (dotted) for a given icg-iteration j. A big difference can be seen with the original criterion taking a lot more Parareal iterations than with the modified criterion.*

The benefits of using the modified Parareal stopping criterion for inexact CG are shown in Figure 9. While the minimisation again ends in 24 iterations, the total number of Parareal iterations is significantly reduced to 132, compared to 436 Parareal iterations in subsection 5.3.1). This corresponds to an average of 5.5 Parareal iterations per icg-iteration, a big improvement.

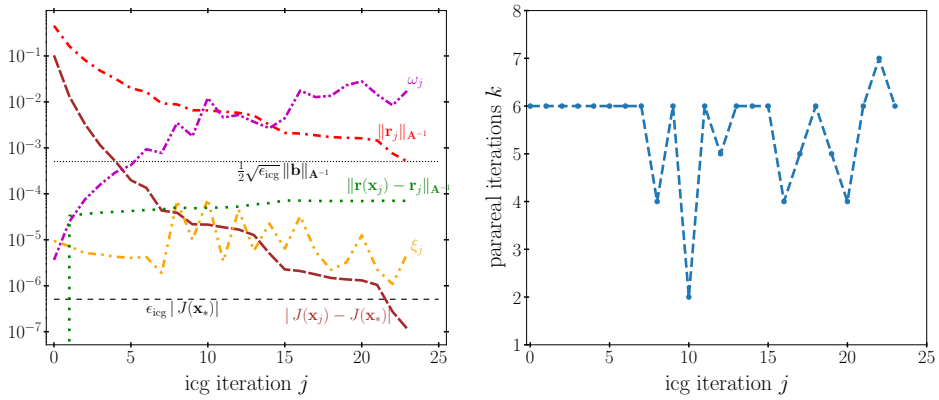

FIG. 9. *Inexact CG following Gratton et al. but now with modified Parareal stopping criterion $\|\boldsymbol{E}_j\boldsymbol{p}_j\|_{\boldsymbol{A}^{-1}} \leq \xi_j$. The new quantity ξ_j (dashdotdotted) on the left image is the bound for $\|\boldsymbol{E}_j\boldsymbol{p}_j\|_{\boldsymbol{A}^{-1}}$. Still no approximations are used. The number of Parareal iterations k required at a given icg-iteration j is plotted on the right image.*

Note that we no longer have to fix the Parareal tolerance ϵ_{p} a priori. The modified Parareal criterion allows inexact CG to use Parareal adaptively since its tolerance (ξ_j) adjusts with every icg-iteration. Additionally, Figure 9 includes the quantity ω_j to show that while the permissible matrix-vector product error $(\|\mathbf{E}_j\|_{\mathbf{A}^{-1},\mathbf{A}})$ gradually increases with icg-iterations, them method still converges and maintains the same level of accuracy as CG.

5.3.3 Inexact conjugate gradient with practical estimates In this subsection we are going to use the derived estimate of $\|\mathbf{E}_j\mathbf{p}_j\|_{\mathbf{A}^{-1}}$ for the modified Parareal stopping criterion. Recall that the approximation is given as

$$\|\mathbf{E}_j\mathbf{p}_j\|_{\mathbf{A}^{-1}} \approx \|\mathbf{P}(k+1)\mathbf{p}_j - \mathbf{P}(k)\mathbf{p}_j\|_2 \tag{5.17}$$

To assess the accuracy of our approximation, Figure 10 compares the exact and approximate values of $\|\mathbf{E}_j\mathbf{p}_j\|_{\mathbf{A}^{-1}}$ for some icg-iterations. It can be seen that both exact and approximate values align, providing a nice approximation of $\|\mathbf{E}_j\mathbf{p}_j\|_{\mathbf{A}^{-1}}$ by the end of the Parareal iteration (where the Parareal stopping criterion is satisfied by ξ_j). Figure 11 further illustrates the results of using approximate $\|\mathbf{E}_j\mathbf{p}_j\|_{\mathbf{A}^{-1}}$ in the mod-

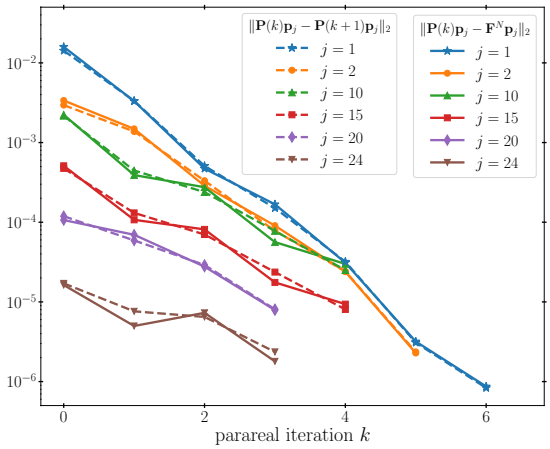

FIG. 10. *Comparison of the exact (solid) and approximate (dashed) values of* $\|\boldsymbol{E}_j\boldsymbol{p}_j\|_{\boldsymbol{A}^{-1}}$ *for icg-iterations* $1, 2, 10, 15, 20, 24$. *The relatively close values at each Parareal iteration suggest that the approximation is a good one.*

ified Parareal criterion for inexact CG. The minimisation performs 24 icg-iterations and a total of 153 Parareal iterations, resulting in an average of 6.375 Parareal iterations per icg-iterations. If we compare Figure 11 and Figure 9 (result with exact $\|\mathbf{E}_j\mathbf{p}_j\|_{\mathbf{A}^{-1}}$), minimal differences can be observed on the left images among various quantities. The only difference is noticed on the right images where inexact CG with approximated $\|\mathbf{E}_j\mathbf{p}_j\|_{\mathbf{A}^{-1}}$ requires one more Parareal iteration per icg-iteration due to the construction of our approximation (the use of $\mathbf{P}(k+1)$).

Finally we present the main result of our work in Figure 12 which uses practical estimates for all the remaining norms (from subsection 4.3), approximate $\|\mathbf{E}_j\mathbf{p}_j\|_{\mathbf{A}^{-1}}$, and the inaccuracy budget within inexact CG. This result follows from Algorithm 4.2 which also requires value of an integer d ($d = 2$ in this case) used for the inexact CG termination criterion (4.5). Even after using all approximations, we see that inexact CG gives similar results to the other variants of inexact CG discussed above. The minimisation ends after 24 iterations and needing a total of 154 Parareal iterations. On average, the number of Parareal iterations per icg-iteration remains 6.4. The *maximum possible speedup* is bounded by a factor of

$$S = \frac{\text{number of time windows}}{\text{average Parareal iterations per icg-iteration}} = \frac{20}{6.4} \approx 3.12 \tag{5.18}$$

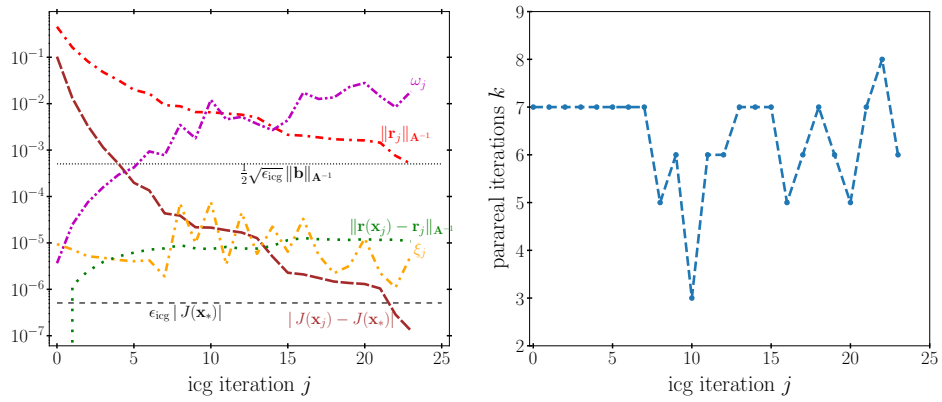

FIG. 11. *Inexact CG using modified Parareal stopping criterion but now with approximation for* $\|\boldsymbol{E}_j\boldsymbol{p}_j\|_{\boldsymbol{A}^{-1}}$ *obtained in subsection* 4.4. *The minimisation result and the corresponding required Parareal iterations* k *are shown on left and right images respectively. We end up using one more Parareal iteration per icg-iteration due to the approximation used.*

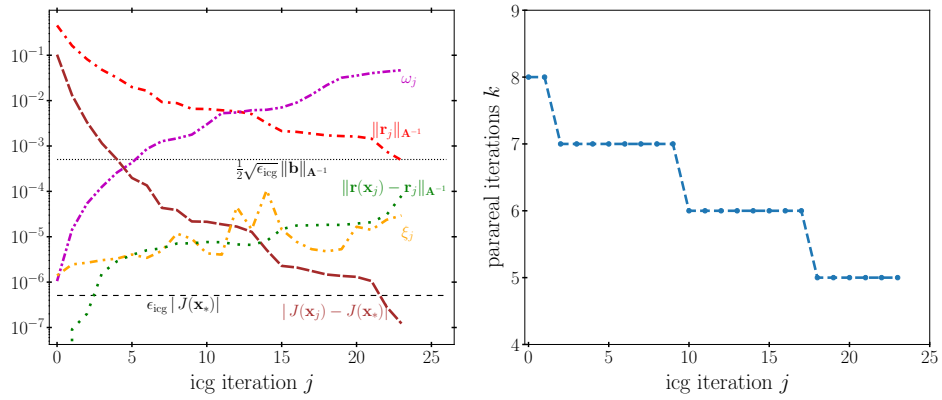

FIG. 12. *Inexact CG using: approximation for* $\|\boldsymbol{E}_j\boldsymbol{p}_j\|_{\boldsymbol{A}^{-1}}$*, practical estimates from subsection* 4.3*, and the inaccuracy budget from subsection* 4.5. *The results are almost identical to Figure* 11 *showing that even after all the approximations the algorithm performs well.*

To check the accuracy of the obtained initial state from inexact CG, we compare the contour plots of the solution from inexact CG with the exact solution for the whole time domain as shown in Figure 13. By inexact CG solution and exact solution , we mean the solutions obtained by applying the fine solver $\mathbf{F}$ to the initial state retrieved from inexact CG and to the given initial state respectively. The free surface values η and velocity values u are plotted separately on the top and bottom row of Figure 13 respectively. Visual inspection of Figure 13 shows minimal differences, indicating that the retrieved initial state is sufficiently accurate. Additionally, Figure 14 presents a contour plot of the error between the exact solution and the inexact CG solution, confirming that the errors are indeed small.

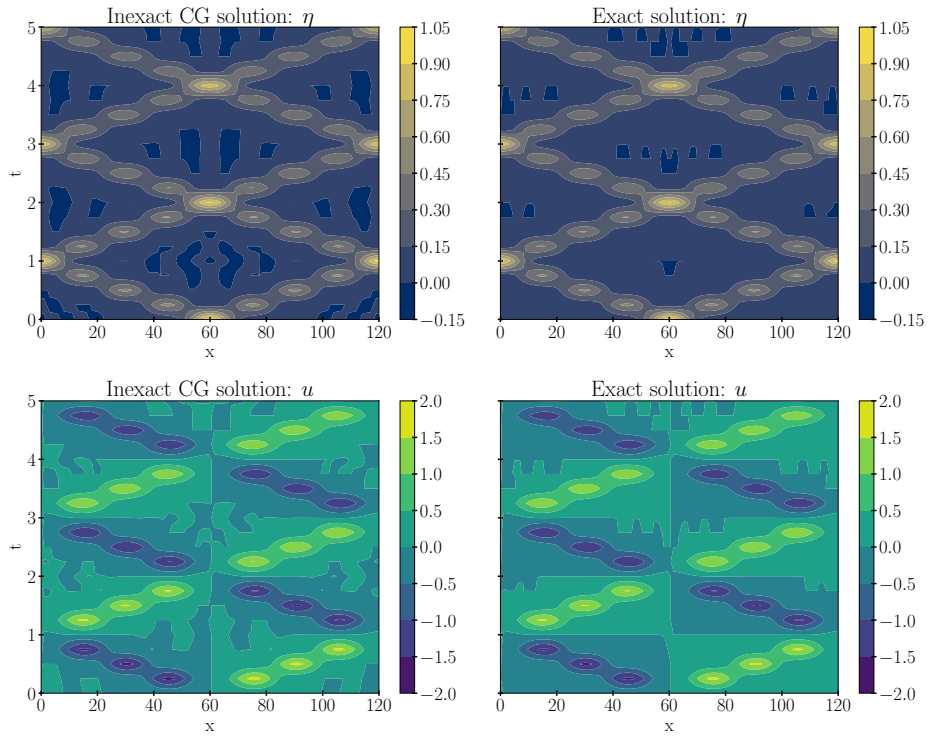

FIG. 13. *Contour plots of the iterative solution (left column) and exact solution (right column) for free surface (first row) and velocity (second row). The iterative and exact solution are obtained by running the fine solver on the initial state retrieved by inexact CG and on the given (exact) initial state respectively. A first glance suggests that the retrieved initial state is very close to the exact initial state.*

6 Conclusion and Future Work In this paper we focused on time parallelising the inner loop of the incremental 4D-Var by utilising Parareal for the tangent-linear model integration. This led to solving an approximate linear system using an inexact version of the conjugate gradient method. The advantage of inexact CG over the usual CG is that it allowed us to adaptively control the required Parareal's accuracy. We also proposed a modified precision criterion based on the original criterion which significantly reduced the required number of required Parareal iterations. Numerical results on the 1D linearised shallow water equations showed that inexact CG with all approximations outperformed CG with Parareal (with a fixed Parareal tolerance) by giving a speedup factor of 3.12.

Areas of future work include extending these developments to a more realistic 2D shallow water model. In that case the inclusion of parameters such as Coriolis force and additional dimension will make the problem more challenging in terms of size and complexity. Another point of interest is the introduction of potential time parallelism in the adjoint direction, i.e. time parallelism for the backward model $(\mathbf{F}^N)^T$ in equation (3.5). This aspect has not been explored in our work since the existing approximation for $\|\mathbf{E}_j\mathbf{p}_j\|_{\mathbf{A}^{-1}}$ is only valid when exact adjoint is used. If the adjoint could be parallelised, a natural extension of this idea would involve using asynchronous Parareal iterations [31, 30] where the algorithm runs without a synchronisation

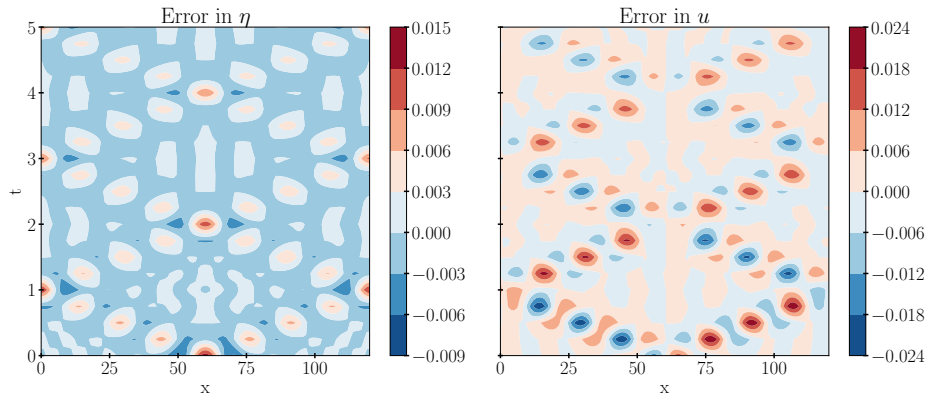

FIG. 14. *Contour plots depicting the error between the inexact CG solution and exact solution for free surface (left) and velocity (right).*

point. In that case the forward and adjoint parallel runs could be done simultaneously without needing to wait for any exchange of information.

Finally, some open questions remain which are worth investigating in the future. One question is whether the modified modified Parareal stopping criterion derived here is also suitable for other types of PDEs or data assimilation setup. In the numerical experiments, the residual mismatch norm $\|\mathbf{r}(\mathbf{x}_j)-\mathbf{r}_j\|_{\mathbf{A}^{-1}}$ in Figure 9 is noticeably larger than other examples. This may explain the better Parareal performance, as less computational effort is spent on making the mismatch unnecessarily small. Additionally, in Figure 10 the exact and approximate values of $\|\mathbf{E}_j\mathbf{p}_j\|_{\mathbf{A}^{-1}}$ agree very closely. Is this level of agreement generally expected or is there a specific reason that makes experiments considered particularly amenable.

Acknowledgement We would like to thank the two anonymous reviewers for their valuable comments which greatly improved our paper.

Data availability statement The python code for the numerical experiments shown in Section 5 can be found in a public GitHub repository. The link is as follows: https://github.com/rbcfc/parareal_data_assimilation.

REFERENCES

[1] M. Asch, M. Bocquet, and M. Nodet, *Data assimilation: methods, algorithms, and applications*, SIAM, 2016.

[2] L. Baffico, S. Bernard, Y. Maday, G. Turinici, and G. Zérah, *Parallel-in-time molecular-dynamics simulations*, Physical Review E, 66 (2002), p. 057701.

[3] G. Bal, *On the convergence and the stability of the parareal algorithm to solve partial differential equations*, in Domain decomposition methods in science and engineering, Springer, 2005, pp. 425–432.

[4] A. Bouras and V. Frayssé, *Inexact matrix-vector products in krylov methods for solving linear systems: a relaxation strategy*, SIAM J. Matrix Anal. Appl., 26 (2005), pp. 660–678.

[5] P. Courtier, E. Andersson, W. Heckley, D. Vasiljevic, M. Hamrud, A. Hollingsworth, F. Rabier, M. Fisher, and J. Pailleux, *The ecmwf implementation of three-dimensional variational assimilation (3d-var). i: Formulation*, Q. J. R. Meteorol. Soc., 124 (1998), pp. 1783–1807.

[6] P. Courtier, J.-N. Thépaut, and A. Hollingsworth, *A strategy for operational implementation of 4d-var, using an incremental approach*, Q. J. R. Meteorol. Soc., 120 (1994),

pp. 1367–1387.
[7] B. Cushman-Roisin and J.-M. Beckers, *Introduction to geophysical fluid dynamics: physical and numerical aspects*, Academic press, 2011.
[8] R. Daley, *Atmospheric data analysis*, no. 2, Cambridge university press, 1993.
[9] X. Du, M. Sarkis, C. E. Schaerer, and D. B. Szyld, *Inexact and truncated parareal-in-time krylov subspace methods for parabolic optimal control problems*, Electron. Trans. Numer. Anal., 40 (2013), pp. 36–57.
[10] D. R. Durran, *Numerical methods for wave equations in geophysical fluid dynamics*, vol. 32, Springer Science & Business Media, 2013.
[11] C. Farhat and M. Chandesris, *Time-decomposed parallel time-integrators: theory and feasibility studies for fluid, structure, and fluid–structure applications*, Internat. J. Numer. Methods Engrg., 58 (2003), pp. 1397–1434.
[12] M. Fisher and S. Gürol, *Parallelization in the time dimension of four-dimensional variational data assimilation*, Q. J. R. Meteorol. Soc., 143 (2017), pp. 1136–1147.
[13] M. Fisher, Y. Tremolet, H. Auvinen, D. Tan, and P. Poli, *Weak-constraint and long-window 4D-Var*, ECMWF Reading, UK, 2012.
[14] M. J. Gander, *50 years of time parallel time integration*, in Multiple shooting and time domain decomposition methods, Springer, 2015, pp. 69–113.
[15] M. J. Gander and E. Hairer, *Nonlinear convergence analysis for the parareal algorithm*, in Domain decomposition methods in science and engineering XVII, Springer, 2008, pp. 45–56.
[16] M. J. Gander, F. Kwok, and J. Salomon, *Paraopt: A parareal algorithm for optimality systems*, SIAM J. Sci. Comput., 42 (2020), pp. A2773–A2802.
[17] M. J. Gander and S. Vandewalle, *Analysis of the parareal time-parallel time-integration method*, SIAM J. Sci. Comput., 29 (2007), pp. 556–578.
[18] A. E. Gill, *Atmosphere-ocean dynamics*, vol. 30, Academic press, 1982.
[19] G. H. Golub and Q. Ye, *Inexact preconditioned conjugate gradient method with inner-outer iteration*, SIAM J. Sci. Comput., 21 (1999), pp. 1305–1320.
[20] S. Gratton, E. Simon, D. Titley-Peloquin, and P. L. Toint, *Minimizing convex quadratics with variable precision conjugate gradients*, Numer. Linear Algebra Appl., 28 (2021), p. e2337.
[21] J. R. Holton, *An introduction to dynamic meteorology*, American Journal of Physics, 41 (1973), pp. 752–754.
[22] K. Ide, P. Courtier, M. Ghil, and A. C. Lorenc, *Unified notation for data assimilation: Operational, sequential and variational (gtspecial issueltdata assimilation in meteology and oceanography: Theory and practice)*, Journal of the Meteorological Society of Japan. Ser. II, 75 (1997), pp. 181–189.
[23] L. Isaksen, *Data assimilation on future computer architectures*, in Proc. ECMWF Seminar on Data Assimilation for Atmosphere and Ocean, 2012, pp. 301–322.
[24] E. Kalnay, *Atmospheric modeling, data assimilation and predictability*, Cambridge university press, 2003.
[25] A. Lawless, S. Gratton, and N. Nichols, *An investigation of incremental 4d-var using non-tangent linear models*, Q. J. R. Meteorol. Soc., 131 (2005), pp. 459–476.
[26] J.-L. Lions, Y. Maday, and G. Turinici, *Résolution d'edp par un schéma en temps pararéel*, Comptes Rendus de l'Académie des Sciences-Series I-Mathematics, 332 (2001), pp. 661–668.
[27] A. Lorenc, S. Ballard, R. Bell, N. Ingleby, P. Andrews, D. Barker, J. Bray, A. Clayton, T. Dalby, D. Li, et al., *The met. office global three-dimensional variational data assimilation scheme*, Q. J. R. Meteorol. Soc., 126 (2000), pp. 2991–3012.
[28] Y. Maday, M.-K. Riahi, and J. Salomon, *Parareal in time intermediate targets methods for optimal control problems*, Control and optimization with PDE constraints, (2013), pp. 79–92.
[29] G. Madec, R. Bourdallé-Badie, P.-A. Bouttier, C. Bricaud, D. Bruciaferri, D. Calvert, J. Chanut, E. Clementi, A. Coward, D. Delrosso, et al., *Nemo ocean engine*, (2017).
[30] F. Magoulès and G. Gbikpi-Benissan, *Asynchronous parareal time discretization for partial differential equations*, SIAM J. Sci. Comput., 40 (2018), pp. C704–C725.
[31] F. Magoulès, G. Gbikpi-Benissan, and Q. Zou, *Asynchronous iterations of parareal algorithm for option pricing models*, Mathematics, 6 (2018), p. 45.
[32] T. P. Mathew, M. Sarkis, and C. E. Schaerer, *Analysis of block parareal preconditioners for parabolic optimal control problems*, SIAM J. Sci. Comput., 32 (2010), pp. 1180–1200.
[33] M. Minion, *A hybrid parareal spectral deferred corrections method*, Commun. Appl. Math.

Comput. Sci., 5 (2011), pp. 265–301.
[34] B. W. Ong and J. B. Schroder, *Applications of time parallelization*, Comput. Vis. Sci., 23 (2020), pp. 1–15.
[35] F. Rabier, H. Järvinen, E. Klinker, J.-F. Mahfouf, and A. Simmons, *The ecmwf operational implementation of four-dimensional variational assimilation. i: Experimental results with simplified physics*, Q. J. R. Meteorol. Soc., 126 (2000), pp. 1143–1170.
[36] D. Ruprecht, *Wave propagation characteristics of parareal*, Comput. Vis. Sci., 19 (2018), pp. 1–17.
[37] D. Ruprecht and R. Krause, *Explicit parallel-in-time integration of a linear acoustic-advection system*, Comput. & Fluids, 59 (2012), pp. 72–83.
[38] V. Simoncini and D. B. Szyld, *Theory of inexact krylov subspace methods and applications to scientific computing*, SIAM J. Sci. Comput., 25 (2003), pp. 454–477.
[39] J.-N. Thepaut and P. Courtier, *Four-dimensional variational data assimilation using the adjoint of a multilevel primitive-equation model*, Q. J. R. Meteorol. Soc., 117 (1991), pp. 1225–1254.
[40] J.-N. Thépaut, R. N. Hoffman, and P. Courtier, *Interactions of dynamics and observations in a four-dimensional variational assimilation*, Mon. Weather Rev., 121 (1993), pp. 3393–3414.
[41] G. K. Vallis, *Atmospheric and oceanic fluid dynamics*, Cambridge University Press, 2017.
[42] J. Van Den Eshof and G. L. Sleijpen, *Inexact krylov subspace methods for linear systems*, SIAM J. Matrix Anal. Appl., 26 (2004), pp. 125–153.
[43] D. C. Wilcox et al., *Turbulence modeling for CFD*, vol. 2, DCW industries La Canada, CA, 1998.