# *Randomised preconditioning for the forcing formulation of weak constraint 4D-Var*

Received: 18 January 2021 | Revised: 10 August 2021 | Accepted: 12 August 2021

DOI: 10.1002/qj.4151

Quarterly Journal of the Royal Meteorological Society RMetS

RESEARCH ARTICLE

# Randomised preconditioning for the forcing formulation of weak-constraint 4D-Var

**Ieva Daužickaitė**[1] | **Amos S. Lawless**[1,2] | **Jennifer A. Scott**[1,3] | **Peter Jan van Leeuwen**[1,4]

[1]School of Mathematical, Physical and Computational Sciences, University of Reading, Reading, UK

[2]National Centre for Earth Observation, Reading, UK

[3]Scientific Computing Department, STFC Rutherford Appleton Laboratory, Didcot, UK

[4]Department of Atmospheric Science, Colorado State University, Fort Collins, Colorado

**Correspondence**
Ieva Daužickaitė, Department of Mathematics and Statistics, Pepper Lane, Whiteknights, Reading, RG6 6AX, UK
Email: i.dauzickaite@pgr.reading.ac.uk

**Funding information**
Engineering and Physical Sciences Research Council, Grant/Award Number: EP/L016613/1; H2020 European Research Council, Grant/Award Number: 694509; NERC National Centre for Earth Observation

**Abstract**

There is growing awareness that errors in the model equations cannot be ignored in data assimilation methods such as four-dimensional variational assimilation (4D-Var). If allowed for, more information can be extracted from observations, longer time windows are possible, and the minimisation process is easier, at least in principle. Weak-constraint 4D-Var estimates the model error and minimises a series of quadratic cost functions, which can be achieved using the conjugate gradient (CG) method; minimising each cost function is called an inner loop. CG needs preconditioning to improve its performance. In previous work, limited-memory preconditioners (LMPs) have been constructed using approximations of the eigenvalues and eigenvectors of the Hessian in the previous inner loop. If the Hessian changes significantly in consecutive inner loops, the LMP may be of limited usefulness. To circumvent this, we propose using randomised methods for low-rank eigenvalue decomposition and use these approximations to construct LMPs cheaply using information from the current inner loop. Three randomised methods are compared. Numerical experiments in idealized systems show that the resulting LMPs perform better than the existing LMPs. Using these methods may allow more efficient and robust implementations of incremental weak-constraint 4D-Var.

**KEYWORDS**

data assimilation, limited-memory preconditioners, randomised methods, sparse symmetric positive-definite systems, weak-constraint 4D-Var

## 1 | INTRODUCTION

In numerical weather prediction, data assimilation provides the initial conditions for the weather model and hence influences the accuracy of the forecast (Kalnay, 2002). Data assimilation uses observations of a dynamical system to correct a previous estimate (background) of the system's state. Statistical knowledge of the errors in the observations and the background is incorporated in the process. A variational data assimilation method called weak-constraint 4D-Var provides a way to also take the model error into account (Trémolet, 2006),

which can lead to a better analysis (e.g., Trémolet, 2007).

We explore the weak-constraint 4D-Var cost function. In its incremental version, the state is updated by a minimiser of the linearised version of the cost function. The minimiser can be found by solving a large sparse linear system. The process of solving each system is called an inner loop. Because the second derivative of the cost function, the Hessian, is symmetric positive-definite, the systems may be solved with the conjugate gradient (CG) method (Hestenes and Stiefel, 1952), the convergence rate of which depends on the eigenvalue distribution of the Hessian. Limited-memory preconditioners (LMPs) have been shown to improve the convergence of CG when minimising the strong-constraint 4D-Var cost function (Fisher, 1998; Tshimanga *et al.*, 2008). Strong-constraint 4D-Var differs from weak-constraint 4D-Var by making the assumption that the dynamical model has no error.

LMPs can be constructed using approximations to the eigenvalues and eigenvectors (eigenpairs) of the Hessian. The Lanczos and CG connection (section 6.7 of Saad, 2003) can be exploited to obtain approximations to the eigenpairs of the Hessian in one inner loop, and these approximations may then be employed to construct the preconditioner for the next inner loop (Tshimanga *et al.*, 2008). This approach does not describe how to precondition the first inner loop, and the number of CG iterations used on the $i$th inner loop limits the number of vectors available to construct the preconditioner on the $(i+1)$th inner loop. Furthermore, the success of preconditioning relies on the assumption that the Hessians do not change significantly from one inner loop to the next.

In this article, we propose addressing these drawbacks by using easy-to-implement subspace iteration methods (see chapter 5 of Saad, 2011) to obtain approximations of the largest eigenvalues and corresponding eigenvectors of the Hessian in the current inner loop. The subspace iteration method first approximates the range of the Hessian by multiplying it with a start matrix (for approaches to choosing it see, e.g., Duff and Scott, 1993) and the speed of convergence depends on the choice of this matrix (e.g., Gu, 2015). A variant of subspace iteration, which uses a Gaussian random start matrix, is called Randomised Eigenvalue Decomposition (REVD). REVD has been popularised by probabilistic analysis (Halko *et al.*, 2011; Martinsson and Tropp, 2020). It has been shown that REVD, which is equivalent to one iteration of the subspace iteration method, can often generate a satisfactory approximation of the largest eigenpairs of a matrix that has rapidly decreasing eigenvalues. Because the Hessian is symmetric positive-definite, a randomised Nyström method for computing a low-rank eigenvalue decomposition can also be used. It is expected to give a higher quality approximation than REVD (e.g., Halko *et al.*, 2011). We explore these two methods and another implementation of REVD, which is based on the *ritzit* implementation of the subspace method (Rutishauser, 1971). The methods differ in the number of matrix–matrix products with the Hessian. Even though more computations are required to generate the preconditioner in the current inner loop compared with using information from the previous inner loop, the randomised methods are block methods and hence easily parallelisable.

In Section 2, we discuss the weak-constraint 4D-Var method and, in Section 3, we consider LMPs and ways to obtain spectral approximations. The three randomised methods are examined in Section 4. Numerical experiments with linear advection and Lorenz 96 models are presented in Section 5, followed by a concluding discussion in Section 6.

## 2 | WEAK-CONSTRAINT 4D-VAR

We are interested in estimating the state evolution of a dynamical system $\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_N$, with $\mathbf{x}_i \in \mathbb{R}^n$, at times $t_0, t_1, \dots, t_N$. Prior information about the state at $t_0$ is called the background and is denoted by $\mathbf{x}^{\mathrm{b}} \in \mathbb{R}^n$. It is assumed that $\mathbf{x}^{\mathrm{b}}$ has Gaussian errors with zero mean and covariance matrix $\mathbf{B} \in \mathbb{R}^{n\times n}$. Observations of the system at time $t_i$ are denoted by $\mathbf{y}_i \in \mathbb{R}^{q_i}$ and their errors are assumed to be Gaussian with zero mean and covariance matrix $\mathbf{R}_i \in \mathbb{R}^{q_i\times q_i}$ ($q_i \ll n$). An observation operator $\mathcal{H}_i$ maps the model variables into the observed quantities at the correct location, i.e. $\mathbf{y}_i = \mathcal{H}_i(\mathbf{x}_i) + \boldsymbol{\zeta}_i$, where $\boldsymbol{\zeta}_i$ is the observation error. We assume that the observation errors are uncorrelated in time.

The dynamics of the system are described using a nonlinear model $\mathcal{M}_i$ such that

$$\mathbf{x}_{i+1} = \mathcal{M}_i(\mathbf{x}_i) + \boldsymbol{\eta}_{i+1}, \tag{1}$$

where $\boldsymbol{\eta}_{i+1}$ is the model error at time $t_{i+1}$. The model errors are assumed to be uncorrelated in time and to be Gaussian with zero mean and covariance matrix $\mathbf{Q}_i \in \mathbb{R}^{n\times n}$.

The forcing formulation of the nonlinear weak-constraint 4D-Var cost function, in which we solve for the initial state and the model error realizations, is

$$\begin{aligned} J(\mathbf{x}_0, \boldsymbol{\eta}_1, \dots, \boldsymbol{\eta}_N) = {} & \frac{1}{2}(\mathbf{x}_0 - \mathbf{x}^{\mathrm{b}})^{\mathrm{T}}\mathbf{B}^{-1}(\mathbf{x}_0 - \mathbf{x}^{\mathrm{b}}) \\ & + \frac{1}{2}\sum_{i=0}^{N}(\mathbf{y}_i - \mathcal{H}_i(\mathbf{x}_i))^{\mathrm{T}}\mathbf{R}_i^{-1}(\mathbf{y}_i - \mathcal{H}_i(\mathbf{x}_i)) \\ & + \frac{1}{2}\sum_{i=1}^{N}\boldsymbol{\eta}_i^{\mathrm{T}}\mathbf{Q}_i^{-1}\boldsymbol{\eta}_i, \end{aligned} \tag{2}$$

where $\mathbf{x}_i$ satisfies the model constraint in Equation (1) (Trémolet, 2006). The analysis (approximation of the state

evolution over the time window) $\mathbf{x}_0^{\mathrm{a}}, \mathbf{x}_1^{\mathrm{a}}, \dots, \mathbf{x}_N^{\mathrm{a}}$ can be obtained from the minimiser of Equation (2) using the constraints in Equation (1).

## 2.1 | Incremental 4D-Var

One way to compute the analysis is to approximate the minimum of Equation (2) with an inexact Gauss–Newton algorithm (Gratton *et al.*, 2007), where a sequence of quadratic cost functions is minimised. In this approach, we update $\mathbf{x}_0$ and the model error

$$\mathbf{p}^{(j+1)} = \mathbf{p}^{(j)} + \delta\mathbf{p}^{(j)}, \tag{3}$$

where $\mathbf{p}^{(j)} = (\mathbf{x}_0^{(j)\mathrm{T}}, \boldsymbol{\eta}_1^{(j)\mathrm{T}}, \dots, \boldsymbol{\eta}_N^{(j)\mathrm{T}})^{\mathrm{T}}$ is the $j$th approximation and $\delta\mathbf{p}^{(j)} = (\delta\mathbf{x}_0^{(j)\mathrm{T}}, \delta\boldsymbol{\eta}_1^{(j)\mathrm{T}}, \dots, \delta\boldsymbol{\eta}_N^{(j)\mathrm{T}})^{\mathrm{T}}$. The $j$th approximation of the state $\mathbf{x}^{(j)} = (\mathbf{x}_0^{(j)\ \mathrm{T}}, \cdots, \mathbf{x}_N^{(j)\ \mathrm{T}})^{\mathrm{T}}$ is calculated with Equation (1) using $\mathbf{p}^{(j)}$. The update $\delta\mathbf{p}^{(j)}$ is obtained by minimising the following cost function:

$$\begin{aligned} J^{\delta}(\delta\mathbf{p}^{(j)}) = &\frac{1}{2}||\delta\mathbf{p}^{(j)} - \mathbf{b}^{(j)}||^2_{\mathbf{D}^{-1}} \\ &+ \frac{1}{2}||\mathbf{H}^{(j)}(\mathbf{L}^{-1})^{(j)}\delta\mathbf{p}^{(j)} - \mathbf{d}^{(j)}||^2_{\mathbf{R}^{-1}}, \end{aligned} \tag{4}$$

where $||\mathbf{a}||^2_{\mathbf{A}^{-1}} = \mathbf{a}^{\mathrm{T}}\mathbf{A}^{-1}\mathbf{a}$ and the covariance matrices are block-diagonal, that is, $\mathbf{D} = diag(\mathbf{B}, \mathbf{Q}_1, \dots, \mathbf{Q}_N) \in \mathbb{R}^{n(N+1)\times n(N+1)}$ and $\mathbf{R} = diag(\mathbf{R}_0, \dots, \mathbf{R}_N) \in \mathbb{R}^{q\times q}$, where $q = \Sigma_{i=0}^{N} q_i$. We use the notation (following Gratton *et al.*, 2018) $\mathbf{H}^{(j)} = diag(\mathbf{H}_0^{(j)}, \dots, \mathbf{H}_N^{(j)}) \in \mathbb{R}^{q\times n(N+1)}$, where $\mathbf{H}_i^{(j)}$ is the linearised observation operator, and

$$(\mathbf{L}^{-1})^{(j)} = \begin{pmatrix} \mathbf{I} & & & & \\ \mathbf{M}_{0,0}^{(j)} & \mathbf{I} & & & \\ \mathbf{M}_{0,1}^{(j)} & \mathbf{M}_{1,1}^{(j)} & \mathbf{I} & & \\ \vdots & \vdots & \ddots & \ddots & \\ \mathbf{M}_{0,N-1}^{(j)} & \mathbf{M}_{1,N-1}^{(j)} & \dots & \mathbf{M}_{N-1,N-1}^{(j)} & \mathbf{I} \end{pmatrix}, \tag{5}$$

$$\mathbf{b}^{(j)} = \begin{pmatrix} \mathbf{x}^{\mathrm{b}} - \mathbf{x}_0^{(j)} \\ -\boldsymbol{\eta}_1^{(j)} \\ \vdots \\ -\boldsymbol{\eta}_N^{(j)} \end{pmatrix}, \tag{6}$$

$$\mathbf{d}^{(j)} = \begin{pmatrix} \mathbf{y}_0 - \mathcal{H}_0(\mathbf{x}_0^{(j)}) \\ \mathbf{y}_1 - \mathcal{H}_1(\mathbf{x}_1^{(j)}) \\ \vdots \\ \mathbf{y}_N - \mathcal{H}_N(\mathbf{x}_N^{(j)}) \end{pmatrix}, \tag{7}$$

where $\mathbf{M}_{i,l}^{(j)} = \mathbf{M}_l^{(j)} \dots \mathbf{M}_i^{(j)}$ and $\mathbf{M}_i^{(j)}$ is the linearised model, that is, $\mathbf{M}_{i,l}^{(j)}$ denotes the linearised model integration from time $t_i$ to $t_{l+1}$, $(\mathbf{L}^{-1})^{(j)} \in \mathbb{R}^{n(N+1)\times n(N+1)}$, $\mathbf{x}^{(j)}, \delta\mathbf{x}^{(j)}, \mathbf{b}^{(j)} \in \mathbb{R}^{n(N+1)}$, and $\mathbf{d}^{(j)} \in \mathbb{R}^{q}$. The outer loop consists of updating Equation (3), calculating $\mathbf{x}^{(j)}, \mathbf{b}^{(j)}, \mathbf{d}^{(j)}$, and linearising $\mathcal{H}_i$ and $\mathcal{M}_i$ for the next inner loop.

The minimum of the quadratic cost function (Equation 4) can be found by solving a linear system

$$\mathcal{A}^{(j)}\delta\mathbf{p}^{(j)} = \mathbf{D}^{-1}\mathbf{b}^{(j)} + (\mathbf{L}^{-\mathrm{T}})^{(j)}(\mathbf{H}^{\mathrm{T}})^{(j)}\mathbf{R}^{-1}\mathbf{d}^{(\mathbf{j})}, \tag{8}$$

$$\begin{aligned} \mathcal{A}^{(j)} = &(\mathbf{D}^{-1} + (\mathbf{L}^{-\mathrm{T}})^{(j)}(\mathbf{H}^{\mathrm{T}})^{(j)}\mathbf{R}^{-1}(\mathbf{H})^{(j)}(\mathbf{L}^{-1})^{(j)}) \\ &\in \mathbb{R}^{n(N+1)\times n(N+1)}, \end{aligned} \tag{9}$$

where $\mathcal{A}^{(j)}$ is the Hessian of Equation (4), which is symmetric positive-definite. These large sparse systems are usually solved with the conjugate gradient (CG) method, the convergence properties of which depend on the spectrum of $\mathcal{A}^{(j)}$ (see Section 3.1 for a discussion). In general, clustered eigenvalues result in fast convergence. We consider a technique to cluster eigenvalues of $\mathcal{A}^{(j)}$ in the following section. From now on we omit the superscript $(j)$.

## 2.2 | Control-variable transform

A control-variable transform, also called first-level preconditioning, maps the variables $\delta\mathbf{p}$ to $\delta\tilde{\mathbf{p}}$, the errors of which are uncorrelated (see, e.g., Section 3.2 of Lawless, 2013). This can be denoted by the transformation $\mathbf{D}^{1/2}\delta\tilde{\mathbf{p}} = \delta\mathbf{p}$, where $\mathbf{D} = \mathbf{D}^{1/2}\mathbf{D}^{1/2}$ and $\mathbf{D}^{1/2}$ is the symmetric square-root. The update $\delta\tilde{\mathbf{p}}$ is then the solution of

$$\begin{aligned} \mathcal{A}^{pr}\delta\tilde{\mathbf{p}} &= \mathbf{D}^{-1/2}\mathbf{b} + \mathbf{D}^{1/2}\mathbf{L}^{-\mathrm{T}}\mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{d}, \\ \mathcal{A}^{pr} &= \mathbf{I} + \mathbf{D}^{1/2}\mathbf{L}^{-\mathrm{T}}\mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{H}\mathbf{L}^{-1}\mathbf{D}^{1/2}. \end{aligned} \tag{10}$$

Here, $\mathcal{A}^{pr}$ is the sum of the identity matrix and a rank $q$ positive semidefinite matrix. Hence, it has a cluster of $n(N+1)-q$ eigenvalues at one and $q$ eigenvalues that are greater than one. Thus, the spectral condition number $\kappa = \lambda_{\max}/\lambda_{\min}$ (here, $\lambda_{\max}$ and $\lambda_{\min}$ are the largest and smallest eigenvalues of $\mathcal{A}^{pr}$, respectively) is equal to $\lambda_{\max}$. We discuss employing second-level preconditioning to reduce the condition number while also preserving the cluster of eigenvalues at one. In the subsequent sections, we use notation that is common in numerical linear algebra. Namely, we use $\mathbf{A}$ for the Hessian with first-level preconditioning, $\mathbf{x}$ for the unknown, and $\mathbf{b}$ for the right-hand side of the system of linear equations. Thus, we denote Equation (10) by

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \tag{11}$$

where the right-hand side $\mathbf{b}$ is known and $\mathbf{x}$ is the required solution. We assume throughout that $\mathbf{A}$ is symmetric positive-definite.

# 3 | PRECONDITIONING WEAK-CONSTRAINT 4D-VAR

## 3.1 | Preconditioned conjugate gradients

The CG method (see, e.g., Saad, 2003) is a popular Krylov subspace method for solving systems of the form in Equation (11). A well-known bound for the error at the $i$th CG iteration $\boldsymbol{\epsilon}_i = \mathbf{x} - \mathbf{x}_i$ is

$$\frac{||\boldsymbol{\epsilon}_i||_{\mathbf{A}}}{||\boldsymbol{\epsilon}_0||_{\mathbf{A}}} \leq 2\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^i, \tag{12}$$

where $\kappa$ is the spectral condition number and $||\boldsymbol{\epsilon}_i||_{\mathbf{A}}^2 = \boldsymbol{\epsilon}_i^{\mathrm{T}}\mathbf{A}\boldsymbol{\epsilon}_i$ (see, e.g., section 5.1 of Nocedal and Wright, 2006). Note that this bound describes the worst-case convergence and only takes into account the largest and smallest eigenvalues. The convergence of CG also depends on the distribution of the eigenvalues of $\mathbf{A}$ (as well as the right-hand side $\mathbf{b}$ and the initial guess $\mathbf{x}_0$); eigenvalues clustered away from zero suggest rapid convergence (lecture 38 of Trefethen and Bau, 1997). Otherwise, CG can display slow convergence and preconditioning is used to try to tackle this problem (chapter 9 of Saad, 2003). Preconditioning aims to map the system in Equation (11) to another system that has the same solution, but different properties that imply faster convergence. Ideally, the preconditioner $\mathbf{P}$ should be cheap both to construct and to apply, and the preconditioned system should be easy to solve.

If $\mathbf{P}$ is a symmetric positive-definite matrix that approximates $\mathbf{A}^{-1}$ and is available in factored form $\mathbf{P} = \mathbf{C}\mathbf{C}^{\mathrm{T}}$, the following system is solved:

$$\mathbf{C}^{\mathrm{T}}\mathbf{A}\mathbf{C}\hat{\mathbf{x}} = \mathbf{C}^{\mathrm{T}}\mathbf{b}, \tag{13}$$

where $\hat{\mathbf{x}} = \mathbf{C}^{-1}\mathbf{x}$. Split preconditioned CG (PCG) for solving Equation (13) is described in Algorithm 1 (see, for example, algorithm 9.2 of Saad, 2003). Note that every CG iteration involves one matrix–vector product with $\mathbf{A}$ (the product $\mathbf{A}\mathbf{p}_{j-1}$ is stored in step 3 and reused in step 5) and this is expensive in weak-constraint 4D-Var, because it involves running the linearised model throughout the assimilation window through the factor $\mathbf{L}^{-1}$.

## 3.2 | Limited-memory preconditioners

In weak-constraint 4D-Var, the preconditioner $\mathbf{P}$ approximates the inverse Hessian. Hence, $\mathbf{P}$ can be obtained using quasi-Newton methods for unconstrained optimization that construct an approximation of the Hessian matrix, which is updated regularly (see, for example, chapter 6 of Nocedal and Wright, 2006). A popular method to approximate the Hessian is Broyden–Fletcher–Goldfarb–Shanno (BFGS, named after the researchers who proposed it), but it is too expensive in terms of storage and updating the approximation. Instead, the so-called block BFGS method (derived by Schnabel, 1983) uses only a limited number of vectors to build the Hessian, and when new vectors are added older ones are dropped. This is an example of a limited-memory preconditioner (LMP), and the one considered by Tshimanga et al. (see Tshimanga *et al.*, 2008; Gratton *et al.*, 2011; Tshimanga, 2007) in the context of strong-constraint 4D-Var. An LMP for an $n_A \times n_A$ symmetric positive-definite matrix $\mathbf{A}$ is defined as follows:

$$\begin{aligned}\mathbf{P}_k = {} & (\mathbf{I}_{n_A} - \mathbf{S}(\mathbf{S}^{\mathrm{T}}\mathbf{A}\mathbf{S})^{-1}\mathbf{S}^{\mathrm{T}}\mathbf{A})(\mathbf{I}_{n_A} - \mathbf{A}\mathbf{S}(\mathbf{S}^{\mathrm{T}}\mathbf{A}\mathbf{S})^{-1}\mathbf{S}^{\mathrm{T}}) \\ & + \mathbf{S}(\mathbf{S}^{\mathrm{T}}\mathbf{A}\mathbf{S})^{-1}\mathbf{S}^{\mathrm{T}},\end{aligned} \tag{14}$$

where $\mathbf{S}$ is an $n_A \times k$ $(k \leq n_A)$ matrix with linearly independent columns $\mathbf{s}_1, \ldots, \mathbf{s}_k$, and $\mathbf{I}_{n_A}$ is the $n_A \times n_A$ identity matrix (Gratton *et al.*, 2011). $\mathbf{P}_k$ is symmetric positive-definite and if $k = n_A$ then $(\mathbf{S}^{\mathrm{T}}\mathbf{A}\mathbf{S})^{-1} = \mathbf{S}^{-1}\mathbf{A}^{-1}\mathbf{S}^{-\mathrm{T}}$ and $\mathbf{P}_k = \mathbf{A}^{-1}$. In data assimilation, we have $k \ll n_A$, hence the name LMPs. $\mathbf{P}_k$ is called a balancing preconditioner in Tang *et al.* (2009).

A potential problem for practical applications of Equation (14) is the need for expensive matrix–matrix products with $\mathbf{A}$. Simpler formulations of Equation (14) are obtained by imposing more conditions on the vectors $\mathbf{s}_1, \ldots, \mathbf{s}_k$. Two formulations that Tshimanga *et al.* (2008) calls spectral-LMP and Ritz-LMP have been used, for example, in ocean data assimilation in the Regional Ocean Modeling System (ROMS: Moore *et al.*, 2011) and the variational data assimilation software with the Nucleus for European Modelling of the Ocean (NEMO) ocean model (NEMOVAR: Mogensen *et al.*, 2012), and coupled climate reanalysis in Coupled ECMWF ReAnalysis (CERA: Laloyaux *et al.*, 2018).

**Algorithm 1.** Split preconditioned CG for solving $\mathbf{A}\mathbf{x} = \mathbf{b}$

**Input:** $\mathbf{A} \in \mathbb{R}^{n_A \times n_A}$, $\mathbf{b} \in \mathbb{R}^{n_A}$, preconditioner $\mathbf{P} = \mathbf{C}\mathbf{C}^{\mathrm{T}} \in \mathbb{R}^{n_A \times n_A}$, initial solution $\mathbf{x}_0 \in \mathbb{R}^{n_A}$

**Output:** solution $\mathbf{x}_j \in \mathbb{R}^{n_A}$

1: Compute $\mathbf{r}_0 = \mathbf{C}^{\mathrm{T}}(\mathbf{b} - \mathbf{A}\mathbf{x}_0)$, and $\mathbf{p}_0 = \mathbf{C}\mathbf{r}_0$
2: **for** $j = 1, 2, \ldots,$ until convergence **do**
3: $\quad \alpha_j = (\mathbf{r}_{j-1}^{\mathrm{T}}\mathbf{r}_{j-1})/(\mathbf{p}_{j-1}^{\mathrm{T}}\mathbf{A}\mathbf{p}_{j-1})$
4: $\quad \mathbf{x}_j = \mathbf{x}_{j-1} + \alpha_j\mathbf{p}_{j-1}$
5: $\quad \mathbf{r}_j = \mathbf{r}_{j-1} - \alpha_j\mathbf{C}^{\mathrm{T}}\mathbf{A}\mathbf{p}_{j-1}$
6: $\quad \beta_j = (\mathbf{r}_j^{\mathrm{T}}\mathbf{r}_j)/(\mathbf{r}_{j-1}^{\mathrm{T}}\mathbf{r}_{j-1})$
7: $\quad \mathbf{p}_j = \mathbf{C}\mathbf{r}_j + \beta_j\mathbf{p}_{j-1}$
8: **end for**

To obtain the spectral-LMP, let $\mathbf{v}_1, \ldots, \mathbf{v}_k$ be orthonormal eigenvectors of $\mathbf{A}$ with corresponding eigenvalues $\lambda_1, \ldots, \lambda_k$. Set $\mathbf{V} = (\mathbf{v}_1, \ldots, \mathbf{v}_k)$ and $\mathbf{\Lambda} = diag(\lambda_1, \ldots, \lambda_k)$, so that $\mathbf{AV} = \mathbf{V\Lambda}$ and $\mathbf{V}^{\mathrm{T}}\mathbf{V} = \mathbf{I}_k$. If $\mathbf{s}_i = \mathbf{v}_i$, $i = 1, \ldots, k$, then the LMP in Equation (14) is the spectral-LMP $\mathbf{P}_k^{\mathrm{sp}}$ (it is called a deflation preconditioner in Giraud and Gratton, 2006), which can be simplified to

$$\mathbf{P}_k^{\mathrm{sp}} = \mathbf{I}_{n_A} - \sum_{i=1}^{k} (1 - \lambda_i^{-1}) \mathbf{v}_i \mathbf{v}_i^{\mathrm{T}}. \tag{15}$$

Then $\mathbf{P}_k^{\mathrm{sp}} = \mathbf{C}_k^{\mathrm{sp}} (\mathbf{C}_k^{\mathrm{sp}})^{\mathrm{T}}$ with (presented in section 2.3.1 of Tshimanga, 2007)

$$\mathbf{C}_k^{\mathrm{sp}} = \prod_{i=1}^{k} \left( \mathbf{I}_{n_A} - \left( 1 - \left( \sqrt{\lambda_i} \right)^{-1} \right) \mathbf{v}_i \mathbf{v}_i^{\mathrm{T}} \right). \tag{16}$$

In many applications, including data assimilation, exact eigenpairs are not known, and their approximations, called Ritz values and vectors, are used (we discuss these in the following section). If $\mathbf{u}_1, \ldots, \mathbf{u}_k$ are orthogonal Ritz vectors, then the following relation holds: $\mathbf{U}^{\mathrm{T}}\mathbf{AU} = \mathbf{\Theta}$, where $\mathbf{U} = (\mathbf{u}_1, \ldots, \mathbf{u}_k)$, $\mathbf{\Theta} = diag(\theta_1, \ldots, \theta_k)$ and $\theta_i$ is a Ritz value. Setting $\mathbf{s}_i = \mathbf{u}_i$, $i = 1, \ldots, k$, the Ritz-LMP $\mathbf{P}_k^{\mathrm{Rt}}$ is

$$\mathbf{P}_k^{\mathrm{Rt}} = (\mathbf{I}_{n_A} - \mathbf{U\Theta}^{-1}\mathbf{U}^{\mathrm{T}}\mathbf{A})(\mathbf{I}_{n_A} - \mathbf{AU\Theta}^{-1}\mathbf{U}^{\mathrm{T}}) + \mathbf{U\Theta}^{-1}\mathbf{U}^{\mathrm{T}}. \tag{17}$$

Each application of $\mathbf{P}_k^{\mathrm{Rt}}$ requires a matrix–matrix product with $\mathbf{A}$. If the Ritz vectors are obtained by the Lanczos process (described in Section 3.4 below), then Equation (17) can be simplified further, so that no matrix–matrix products with $\mathbf{A}$ are needed (see section 4.2.2 of Gratton *et al.*, 2011 for details).

An important property is that if an LMP is constructed using $k$ vectors then at least $k$ eigenvalues of the preconditioned matrix $\mathbf{C}^{\mathrm{T}}\mathbf{AC}$ will be equal to one, and the remaining eigenvalues will lie between the smallest and largest eigenvalues of $\mathbf{A}$ (see theorem 3.4 of Gratton *et al.*, 2011). Moreover, if $\mathbf{A}$ has a cluster of eigenvalues at one, then LMPs preserve this cluster. This is crucial when preconditioning Equation (10): because the LMPs preserve the $n(N+1) - q$ smallest eigenvalues of $\mathcal{A}^{pr}$ that are equal to one, the CG convergence can be improved by decreasing the largest eigenvalues. Hence, it is preferable to use the largest eigenpairs or their approximations.

In practice, both spectral-LMP and Ritz-LMP use Ritz vectors and values to construct the LMPs. It has been shown that the Ritz-LMP can perform better than spectral-LMP in a strong-constraint 4D-Var setting by correcting for the inaccuracies in the estimates of eigenpairs (Tshimanga *et al.*, 2008). However, Gratton *et al.* (2011) (their theorem 4.5) have shown that, if the preconditioners are constructed with Ritz vectors and values that have converged, then the spectral-LMP acts like the Ritz-LMP.

**Algorithm 2.** Rayleigh–Ritz procedure for computing approximations of eigenpairs of symmetric $\mathbf{A}$

**Input:** symmetric matrix $\mathbf{A} \in \mathbb{R}^{n_A \times n_A}$, orthogonal matrix $\tilde{\mathbf{Z}} \in \mathbb{R}^{n_A \times m}$, $m < n_A$

**Output:** orthogonal $\mathbf{U} \in \mathbb{R}^{n_A \times m}$ with approximations to eigenvectors of $\mathbf{A}$ as its columns, and diagonal $\mathbf{\Theta} \in \mathbb{R}^{m \times m}$ with approximations to eigenvalues of $\mathbf{A}$ on the diagonal

1: Form $\tilde{\mathbf{K}} = \tilde{\mathbf{Z}}^{\mathrm{T}}\mathbf{A}\tilde{\mathbf{Z}} \in \mathbb{R}^{m \times m}$
2: Form EVD of $\tilde{\mathbf{K}}$ : $\tilde{\mathbf{K}} = \mathbf{W\Theta W}^{\mathrm{T}}$, where $\mathbf{W}$, $\mathbf{\Theta} \in \mathbb{R}^{m \times m}$
3: Form Ritz vectors $\mathbf{U} = \tilde{\mathbf{Z}}\mathbf{W} \in \mathbb{R}^{n_A \times m}$

## 3.3 | Ritz information

Calculating or approximating all the eigenpairs of a large sparse matrix is impractical. Hence, only a subset is approximated to construct the LMPs. This is often done by extracting approximations from a subspace, and the Rayleigh–Ritz (RR) procedure is a popular method for doing this.

Assume that $\mathcal{Z} \subset \mathbb{R}^{n_A}$ is an invariant subspace of $\mathbf{A}$, that is, $\mathbf{Az} \in \mathcal{Z}$ for every $\mathbf{z} \in \mathcal{Z}$, and the columns of $\mathbf{Z} \in \mathbb{R}^{n_A \times m}$, $m < n_A$, form an orthonormal basis for $\mathcal{Z}$. If $(\lambda, \hat{\mathbf{y}})$ is an eigenpair of $\mathbf{K} = \mathbf{Z}^{\mathrm{T}}\mathbf{AZ} \in \mathbb{R}^{m \times m}$, then $(\lambda, \mathbf{v})$, where $\mathbf{v} = \mathbf{Z}\hat{\mathbf{y}}$, is an eigenpair of $\mathbf{A}$ (see, e.g., theorem 1.2 in chapter 4 of Stewart, 2001). Hence, eigenvalues of $\mathbf{A}$ that lie in the subspace $\mathcal{Z}$ can be extracted by solving a small eigenvalue problem.

However, generally the computed subspace $\tilde{\mathcal{Z}}$ with orthonormal basis as columns of $\tilde{\mathbf{Z}}$ is not invariant. Hence, only approximations $\tilde{\mathbf{v}}$ to the eigenvectors $\mathbf{v}$ belong to $\tilde{\mathcal{Z}}$. The RR procedure computes approximations $\mathbf{u}$ to $\tilde{\mathbf{v}}$. We give the RR procedure in Algorithm 2, where the eigenvalue decomposition is abbreviated as EVD. Approximations to eigenvalues $\lambda$ are called Ritz values $\theta$, and $\mathbf{u}$ are the Ritz vectors. Eigenvectors of $\tilde{\mathbf{K}} = \tilde{\mathbf{Z}}^{\mathrm{T}}\mathbf{A}\tilde{\mathbf{Z}}$, which is the projection of $\mathbf{A}$ on to $\tilde{\mathcal{Z}}$, are denoted by $\mathbf{w}$ and are called primitive Ritz vectors.

## 3.4 | Spectral information from CG

Tshimanga *et al.* (2008) use Ritz pairs of the Hessian in one inner loop to construct LMPs for the following inner loop, i.e. information on $\mathbf{A}^{(0)}$ is used to precondition $\mathbf{A}^{(1)}$, and so on. Success relies on the Hessians not changing

significantly from one inner loop to the next. Ritz information can be obtained from the Lanczos process that is connected to CG, hence information for the preconditioner can be gathered at a negligible cost.

The Lanczos process is used to obtain estimates of a few extremal eigenvalues and corresponding eigenvectors of a symmetric matrix $\mathbf{A}$ (section 10.1 of Golub and Van Loan, 2013). It produces a sequence of tridiagonal matrices $\mathbf{T}_j \in \mathbb{R}^{j\times j}$, the largest and smallest eigenvalues of which converge to the largest and smallest eigenvalues of $\mathbf{A}$. Given a starting vector $\mathbf{f}_0$, it also computes an orthonormal basis $\mathbf{f}_0, \ldots, \mathbf{f}_{j-1}$ for the Krylov subspace $\mathcal{K}_j = span\{\mathbf{f}_0, \mathbf{A}\mathbf{f}_0, \ldots, \mathbf{A}^{j-1}\mathbf{f}_0\}$. Ritz values $\theta_i$ are obtained as eigenvalues of a tridiagonal matrix, which has the following structure:

$$\mathbf{T}_j = \begin{pmatrix} \gamma_1 & \tau_1 & & \\ \tau_1 & \gamma_2 & \tau_2 & \\ & \ddots & \ddots & \ddots \\ & & \tau_{j-1} & \gamma_j \end{pmatrix}. \tag{18}$$

The Ritz vectors of $\mathbf{A}$ are $\mathbf{u}_i = \mathbf{F}_j\mathbf{w}_i$, where $\mathbf{F}_j = (\mathbf{f}_0, \ldots, \mathbf{f}_{j-1})$ and the eigenvector $\mathbf{w}_i$ of $\mathbf{T}_j$ is a primitive Ritz vector. Eigenpairs of $\mathbf{T}_j$ can be obtained using a symmetric tridiagonal QR algorithm or Jacobi procedures (see, e.g., section 8.5 of Golub and Van Loan, 2013).

Saad (see section 6.7.3 of Saad, 2003) discusses obtaining entries of $\mathbf{T}_j$ when solving $\mathbf{A}\mathbf{x} = \mathbf{b}$ with CG. At the $j$th iteration of CG, new entries of $\mathbf{T}_j$ are calculated as follows:

$$\gamma_j = \begin{cases} \frac{1}{\alpha_j} & \text{for } j = 1, \\ \frac{1}{\alpha_j} + \frac{\beta_{j-1}}{\alpha_{j-1}} & \text{for } j > 1, \end{cases} \tag{19}$$

$$\tau_j = \frac{\sqrt{\beta_j}}{\alpha_j}, \tag{20}$$

and the vector $\mathbf{f}_j = \mathbf{r}_j/||\mathbf{r}_j||$, where $||\mathbf{r}_j||^2 = \mathbf{r}_j^{\mathrm{T}}\mathbf{r}_j$ and $\alpha_j, \beta_j$, and $\mathbf{r}_j$ are as in Algorithm 1. Hence, obtaining eigenvalue information requires normalizing the residual vectors and finding eigenpairs of the tridiagonal matrix $\mathbf{T}_j$. In data assimilation, the dimension of $\mathbf{T}_j$ is small, because the cost of matrix–vector products restricts the number of CG iterations in the previous inner loop. Hence its eigenpairs can be calculated cheaply. However, caution has to be taken to avoid 'ghost' values, that is, repeated Ritz values, due to the loss of orthogonality in CG (section 10.3.5 of Golub and Van Loan, 2013). This can be addressed using a complete reorthogonalization in every CG iteration, which is done in the CONGRAD routine used at the European Centre for Medium Range Weather Forecasts (ECMWF, 2020). This makes every CG iteration more expensive, but CG may converge in fewer iterations (Fisher, 1998).

## 4 | RANDOMISED EIGENVALUE DECOMPOSITION

If the Hessian in one inner loop differs significantly from the Hessian in the previous inner loop, then it may not be useful to precondition the former with an LMP that is constructed with information from the latter. Employing the Lanczos process to obtain eigenpair estimates and use them to construct an LMP in the same inner loop is too computationally expensive, because each iteration of the Lanczos process requires a matrix–vector product with the Hessian, thus the cost is similar to the cost of CG. Hence, we explore a different approach.

Subspace iteration is a simple procedure to obtain approximations to the largest eigenpairs (see, e.g., chapter 5 of Saad, 2011). It is easily understandable and can be implemented in a straightforward manner, although its convergence can be very slow if the largest eigenvalues are not well separated from the rest of the spectrum. The accuracy of subspace iteration may be enhanced by using a RR projection.

Such an approach is used in Randomised Eigenvalue Decomposition (REVD: see, e.g., Halko *et al.*, 2011). This takes a Gaussian random matrix, that is, a matrix with independent standard normal random variables with zero mean and variance equal to one as its entries, and applies one iteration of the subspace iteration method with RR projection, hence obtaining a rank $m$ approximation $\mathbf{A} \approx \mathbf{Z}_1(\mathbf{Z}_1^{\mathrm{T}}\mathbf{A}\mathbf{Z}_1)\mathbf{Z}_1^{\mathrm{T}}$, where $\mathbf{Z}_1 \in \mathbb{R}^{n_A\times m}$ is orthogonal. We present REVD in Algorithm 3. An important feature of REVD is the observation that the accuracy of the approximation is enhanced with oversampling (which is also called "using guard vectors" in Duff and Scott, 1993), that is, working on a larger space than the required number of Ritz vectors. Halko *et al.* (2011) claim that setting the oversampling parameter to 5 or 10 is often sufficient.

Randomised algorithms are designed to minimise the communication instead of the flop count. The expensive parts of Algorithm 3 are the two matrix–matrix products $\mathbf{A}\mathbf{G}$ and $\mathbf{A}\mathbf{Z}_1$ in steps 2 and 4, that is, in each of these steps, matrix $\mathbf{A}$ has to be multiplied with $(k + l)$ vectors, which in serial computations would be essentially the cost of $2(k + l)$ iterations of unpreconditioned CG. However, note that these matrix–matrix products can be parallelised.

In weak-constraint 4D-Var, $\mathbf{A}$ is the Hessian, hence it is symmetric positive-definite and its eigenpairs can also be approximated using a randomised Nyström method (algorithm 5.5 of Halko *et al.*, 2011), which is expected to give much more accurate results than REVD (Halko *et al.*,

**Algorithm 3.** Randomised eigenvalue decomposition, REVD

**Input:** symmetric matrix $\mathbf{A} \in \mathbb{R}^{n_A \times n_A}$, target rank $k$, an oversampling parameter $l$

**Output:** orthogonal $\mathbf{U}_1 \in \mathbb{R}^{n_A \times k}$ with approximations to eigenvectors of $\mathbf{A}$ as its columns, and diagonal $\mathbf{\Theta}_1 \in \mathbb{R}^{k \times k}$ with approximations to the largest eigenvalues of $\mathbf{A}$ on the diagonal

1: Form a Gaussian random matrix $\mathbf{G} \in \mathbb{R}^{n_A \times (k+l)}$
2: Form a sample matrix $\mathbf{Y} = \mathbf{AG} \in \mathbb{R}^{n_A \times (k+l)}$
3: Orthonormalize the columns of $\mathbf{Y}$ to obtain orthonormal $\mathbf{Z}_1 \in \mathbb{R}^{n_A \times (k+l)}$
4: Form $\mathbf{K}_1 = \mathbf{Z}_1^{\mathrm{T}}\mathbf{A}\mathbf{Z}_1 \in \mathbb{R}^{(k+l)\times(k+l)}$
5: Form EVD of $\mathbf{K}$ : $\mathbf{K} = \mathbf{W}_1\mathbf{\Theta}_1\mathbf{W}_1^{\mathrm{T}}$, where $\mathbf{W}_1$, $\mathbf{\Theta}_1 \in \mathbb{R}^{(k+l)\times(k+l)}$, elements of $\mathbf{\Theta}_1$ are sorted in decreasing order
6: Remove last $l$ columns and rows of $\mathbf{\Theta}_1$, so that $\mathbf{\Theta}_1 \in \mathbb{R}^{k\times k}$
7: Remove last $l$ columns of $\mathbf{W}_1$, so that $\mathbf{W}_1 \in \mathbb{R}^{(k+l)\times k}$
8: Form $\mathbf{U}_1 = \mathbf{Z}_1\mathbf{W}_1 \in \mathbb{R}^{n_A \times k}$

**Algorithm 4.** Randomised eigenvalue decomposition for symmetric positive semidefinite $\mathbf{A}$, Nyström

**Input:** symmetric positive semidefinite matrix $\mathbf{A} \in \mathbb{R}^{n_A \times n_A}$, target rank $k$, an oversampling parameter $l$

**Output:** orthogonal $\mathbf{U}_2 \in \mathbb{R}^{n_A \times k}$ with approximations to eigenvectors of $\mathbf{A}$ as its columns, and diagonal $\mathbf{\Theta}_2 \in \mathbb{R}^{k\times k}$ with approximations to the largest eigenvalues of $\mathbf{A}$ on the diagonal

1: Form a Gaussian random matrix $\mathbf{G} \in \mathbb{R}^{n_A \times (k+l)}$
2: Form a sample matrix $\mathbf{Y} = \mathbf{AG} \in \mathbb{R}^{n_A \times (k+l)}$
3: Orthonormalize the columns of $\mathbf{Y}$ to obtain orthonormal $\mathbf{Z}_1 \in \mathbb{R}^{n_A \times (k+l)}$
4: Form matrices $\mathbf{E}^{(1)} = \mathbf{AZ}_1 \in \mathbb{R}^{n_A \times (k+l)}$ and $\mathbf{E}^{(2)} = \mathbf{Z}_1^{\mathrm{T}}\mathbf{E}^{(1)} \in \mathbb{R}^{(k+l)\times(k+l)}$
5: Form a Cholesky factorization $\mathbf{E}^{(2)} = \mathbf{C}^{\mathrm{T}}\mathbf{C}$
6: Solve $\mathbf{FC} = \mathbf{E}^{(1)}$ for $\mathbf{F} \in \mathbb{R}^{n_A \times (k+l)}$
7: Form SVD of $\mathbf{F}$ : $\mathbf{F} = \mathbf{U}_2\mathbf{\Sigma}\mathbf{V}^{\mathrm{T}}$, where $\mathbf{U}_2$, $\mathbf{V} \in \mathbb{R}^{n_A \times (k+l)}$, $\mathbf{\Sigma} \in \mathbb{R}^{(k+l)\times(k+l)}$, elements of $\mathbf{\Sigma}$ are sorted in decreasing order
8: Remove last $l$ columns of $\mathbf{U}_2$, so that $\mathbf{U}_2 \in \mathbb{R}^{n_A \times k}$
9: Remove last $l$ columns and rows of $\mathbf{\Sigma}$, so that $\mathbf{\Sigma} \in \mathbb{R}^{k\times k}$, and set $\mathbf{\Theta}_2 = \mathbf{\Sigma}^2$

2011). We present the Nyström method in Algorithm 4, where singular-value decomposition is abbreviated as SVD. It considers a more elaborate rank $m$ approximation than in REVD: $\mathbf{A} \approx (\mathbf{AZ}_1)(\mathbf{Z}_1^{\mathrm{T}}\mathbf{AZ}_1)^{-1}(\mathbf{AZ}_1)^{\mathrm{T}} = \mathbf{FF}^{\mathrm{T}}$, where $\mathbf{Z}_1 \in \mathbb{R}^{n_A \times m}$ is orthogonal (obtained in the same way as in REVD, e.g. using a tall skinny QR (TSQR) decomposition: Demmel *et al.*, 2012) and $\mathbf{F} = (\mathbf{AZ}_1)(\mathbf{Z}_1^{\mathrm{T}}\mathbf{AZ}_1)^{-1/2} \in \mathbb{R}^{n_A \times m}$ is an approximate Cholesky factor of $\mathbf{A}$, which is found in step 6. The eigenvalues of $\mathbf{FF}^{\mathrm{T}}$ are the squares of the singular values of $\mathbf{F}$ (see section 2.4.2 of Golub and Van Loan, 2013). In numerical computations we store matrices $\mathbf{E}^{(1)} = \mathbf{AZ}_1$ and $\mathbf{E}^{(2)} = \mathbf{Z}_1^{\mathrm{T}}\mathbf{E}^{(1)} = \mathbf{Z}_1^{\mathrm{T}}\mathbf{AZ}_1$ (step 4), perform the Cholesky factorization of $\mathbf{E}^{(2)} = \mathbf{C}^{\mathrm{T}}\mathbf{C}$ (step 5), and obtain $\mathbf{F}$ by solving the triangular system $\mathbf{FC} = \mathbf{E}^{(1)}$.

The matrix–matrix product with $\mathbf{A}$ at step 4 of Algorithms 3 and 4 is removed in Rutishauser's implementation of subspace iteration with RR projection called *ritzit* (Rutishauser, 1971). It can be derived in the following manner (see chapter 14 of Parlett, 1998). Assume that $\mathbf{G}_3 \in \mathbb{R}^{n_A \times m}$ is an orthogonal matrix and the sample matrix is $\mathbf{Y}_3 = \mathbf{AG}_3 = \mathbf{Z}_3\mathbf{R}_3$, where $\mathbf{Z}_3 \in \mathbb{R}^{n_A \times m}$ is orthogonal and $\mathbf{R}_3 \in \mathbb{R}^{m \times m}$ is upper triangular. Then a projection of $\mathbf{A}^2$ onto the column space of $\mathbf{G}_3$ is $\hat{\mathbf{K}} = \mathbf{Y}_3^{\mathrm{T}}\mathbf{Y}_3 = \mathbf{R}_3^{\mathrm{T}}\mathbf{Z}_3^{\mathrm{T}}\mathbf{Z}_3\mathbf{R}_3 = \mathbf{R}_3^{\mathrm{T}}\mathbf{R}_3$. Then $\mathbf{K}_3 = \mathbf{R}_3\mathbf{R}_3^{\mathrm{T}} = \mathbf{R}_3\mathbf{R}_3^{\mathrm{T}}\mathbf{R}_3\mathbf{R}_3^{-1} = \mathbf{R}_3\hat{\mathbf{K}}\mathbf{R}_3^{-1}$, which is similar to $\hat{\mathbf{K}}$ and hence has the same eigenvalues. This leads to another implementation of REVD presented in Algorithm 5. This is a single-pass algorithm, meaning that $\mathbf{A}$ has to be accessed just once, and, to the best of our knowledge, this method has not been considered in the context of randomised eigenvalue approximations.

Note that the Ritz vectors given by Algorithms 3, 4, and 5 are different. Although Algorithm 5 accesses the matrix $\mathbf{A}$ only once, it requires an additional orthogonalisation of a matrix of size $n_A \times (k+l)$.

In Table 1, we summarise some properties of the Lanczos, REVD, Nyström, and REVD_*ritzit* methods when they are used to compute Ritz values and vectors to generate a preconditioner for $\mathbf{A}$ in incremental data assimilation. Note that the cost of applying spectral-LMP depends on the number of vectors $k$ used in its construction and is independent of which method is used to obtain them. The additional cost of using randomised algorithms arises only once per inner loop when the preconditioner is generated. We recall that in these algorithms the required EVD or SVD of the small matrix can be obtained cheaply and the most expensive parts of the algorithms are the matrix–matrix products of $\mathbf{A}$ and the $n_A \times (k+l)$ matrices. If enough computational resources are available, these can be parallelised. In the best-case scenario, all $k+l$ matrix–vector products can be performed at the same time, making the cost of the matrix–matrix product equivalent to the cost of one iteration of CG plus communication between the processors.

When a randomised method is used to generate the preconditioner, an inner loop is performed as follows. Estimates of the Ritz values of the Hessian and the corresponding Ritz vectors are obtained with a randomised

**TABLE 1** A summary of the properties of the different methods of obtaining $k$ Ritz vectors and values to generate the preconditioner for a $n_A \times n_A$ matrix $\mathbf{A}$ in the $i$th inner loop; here $l$ is the oversampling parameter, while * applies for CG with reorthogonalization

| | **Lanczos** | **REVD** | **Nyström** | **REVD_*ritzit*** |
|---|---|---|---|---|
| Information source | Previous inner loop | Current inner loop | Current inner loop | Current inner loop |
| Preconditioner for the first inner loop | No | Yes | Yes | Yes |
| $k$ dependence on the previous inner loop | Bounded by the number of CG iterations | Independent | Independent | Independent |
| Matrix-matrix products with $\mathbf{A}$ | None | 2 products with $n_A \times (k+l)$ matrices | 2 products with $n_A \times (k+l)$ matrices | 1 product with $n_A \times (k+l)$ matrix |
| QR decomposition | None | None | None | $\mathbf{Y}_3 \in \mathbb{R}^{n_A \times (k+l)}$ |
| Orthogonalisation | None | $\mathbf{Y} \in \mathbb{R}^{n_A \times (k+l)}$ | $\mathbf{Y} \in \mathbb{R}^{n_A \times (k+l)}$ | $\mathbf{G} \in \mathbb{R}^{n_A \times (k+l)}$ |
| Cholesky factorization | None | None | $\mathbf{E}^{(2)} \in \mathbb{R}^{(k+l)\times(k+l)}$ | None |
| Triangular solve | None | None | $\mathbf{FC} = \mathbf{E}^{(1)}$ for $\mathbf{F} \in \mathbb{R}^{n_A \times (k+l)}$ | None |
| Deterministic EVD | $\mathbf{T}_k \in \mathbb{R}^{k\times k}$ * | $\mathbf{K} \in \mathbb{R}^{(k+l)\times(k+l)}$ | None | $\mathbf{K}_3 \in \mathbb{R}^{(k+l)\times(k+l)}$ |
| Deterministic SVD | None | None | $\mathbf{F} \in \mathbb{R}^{n_A \times (k+l)}$ | None |

method (Algorithm 3, 4, or 5) and used to construct an LMP. Then the system of Equation (11) with the exact Hessian $\mathbf{A}$ is solved with PCG (Algorithm 1) using the LMP. The state is updated in the outer loop using the PCG solution.

## 5 | NUMERICAL EXPERIMENTS

We demonstrate our proposed preconditioning strategies using two models: a simple linear advection model to explore the spectra of the preconditioned Hessian and the nonlinear Lorenz 96 model (Lorenz, 1996) to explore the convergence of split preconditioned CG (PCG). We perform identical twin experiments, where $\mathbf{x}^t = ((\mathbf{x}_0^t)^{\mathrm{T}}, \ldots, (\mathbf{x}_N^t)^{\mathrm{T}})^{\mathrm{T}}$ denotes the reference trajectory. The observations and background state are generated by adding noise to $\mathcal{H}_i(\mathbf{x}_i^t)$ and $\mathbf{x}_0^t$ with covariance matrices $\mathbf{R}$ and $\mathbf{B}$, respectively. We use direct observations, thus the observation operator $\mathcal{H}_i$ is linear.

We use covariance matrices $\mathbf{R}_i = \sigma_{\mathrm{o}}^2 \mathbf{I}_{q_i}$, where $q_i$ is the number of observations at time $t_i$, $\mathbf{Q}_i = \sigma_q^2 \mathbf{C}_q$, where $\mathbf{C}_q$ is a Laplacian correlation matrix (Johnson *et al.*, 2005), and $\mathbf{B} = \sigma_{\mathrm{b}}^2 \mathbf{C}_{\mathrm{b}}$, where $\mathbf{C}_{\mathrm{b}}$ is a second-order auto-regressive correlation matrix (Daley, 1993).

We assume that first-level preconditioning has already been applied (recall Equation 10). In data assimilation, using Ritz-LMP as formulated in Equation (17) is impractical because of the matrix products with $\mathbf{A}$ and we cannot use a simple formulation of Ritz-LMP when the Ritz values and vectors are obtained with randomised methods. Hence, we use spectral-LMP. However, as we mentioned in Section 3.2, the spectral-LMP that is constructed with well-converged Ritz values and vectors acts like Ritz-LMP. When we consider the second inner loop, we compare the spectral-LMPs using information from the randomised methods with the spectral-LMP constructed using information obtained with the Matlab function *eigs* in the previous inner loop. *eigs* returns a highly accurate estimate of a few largest or smallest eigenvalues and corresponding eigenvectors. We will use the term randomised LMP to refer to the spectral-LMPs that are constructed with information from the randomised methods, and deterministic LMP to refer to the spectral-LMP that is constructed with information from *eigs*.

The computations are performed with Matlab R2017b. Linear systems are solved using the Matlab implementation of PCG (function *pcg*), which was modified to allow split preconditioning to maintain the symmetric coefficient matrix at every loop.

**Algorithm 5.** Randomised eigenvalue decomposition based on *ritzit*, REVD_*ritzit*

**Input:** symmetric matrix $\mathbf{A} \in \mathbb{R}^{n_A \times n_A}$, target rank $k$, an oversampling parameter $l$

**Output:** orthogonal $\mathbf{U}_3 \in \mathbb{R}^{n_A \times k}$ with approximations to eigenvectors of $\mathbf{A}$ as its columns, and diagonal $\mathbf{\Theta}_3 \in \mathbb{R}^{k \times k}$ with approximations to the largest eigenvalues of $\mathbf{A}$ on the diagonal

1: Form a Gaussian random matrix $\mathbf{G} \in \mathbb{R}^{n_A \times (k+l)}$
2: Orthonormalize the columns of $\mathbf{G}$ to obtain orthonormal $\mathbf{G}_3$
3: Form a sample matrix $\mathbf{Y}_3 = \mathbf{A}\mathbf{G}_3 \in \mathbb{R}^{n_A \times (k+l)}$
4: Compute QR decomposition $\mathbf{Y}_3 = \mathbf{Z}_3\mathbf{R}_3$ to obtain orthogonal $\mathbf{Z}_3 \in \mathbb{R}^{n_A \times (k+l)}$ and upper triangular $\mathbf{R}_3 \in \mathbb{R}^{(k+l)\times(k+l)}$
5: Form $\mathbf{K}_3 = \mathbf{R}_3\mathbf{R}_3^{\mathrm{T}} \in \mathbb{R}^{(k+l)\times(k+l)}$
6: Form EVD of $\mathbf{K}_3$ : $\mathbf{K}_3 = \mathbf{W}_3\mathbf{\Theta}_3^2\mathbf{W}_3^{\mathrm{T}}$, where $\mathbf{W}_3$, $\mathbf{\Theta}_3^2 \in \mathbb{R}^{(k+l)\times(k+l)}$, elements of $\mathbf{\Theta}_3$ are sorted in decreasing order
7: Remove last $l$ columns and rows of $\mathbf{\Theta}_3^2$, so that $\mathbf{\Theta}_3^2 \in \mathbb{R}^{k\times k}$
8: Remove last $l$ columns of $\mathbf{W}_3$, so that $\mathbf{W}_3 \in \mathbb{R}^{(k+l)\times k}$
9: Form $\mathbf{U}_3 = \mathbf{Z}_3\mathbf{W}_3 \in \mathbb{R}^{n_A \times k}$

## 5.1 | Advection model

First, we consider the linear advection model:

$$\frac{\partial u(z,t)}{\partial t} + \frac{\partial u(z,t)}{\partial z} = 0, \tag{21}$$

where $z \in [0,1]$ and $t \in (0,T)$. An upwind numerical scheme is used to discretise Equation (21) (see, e.g., chapter 4 of Morton and Mayers, 1994). To allow us to compute all the eigenvalues (described in the following section), we consider a small system with the linear advection model. The domain is divided into $n = 40$ equally spaced grid points, with grid spacing $\Delta z = 1/n$. We run the model for 51 time steps ($N = 50$) with time-step size $\Delta t = 1/N$, hence $\mathbf{A}$ is a 2,040 × 2,040 matrix. The Courant number is $C = 0.8$ (the upwind scheme is stable with $C \in [0,1]$). The initial conditions are Gaussian,

$$u(z,0) = 6\exp\left(-\frac{(z-0.5)^2}{2\times 0.1^2}\right),$$

and the boundary conditions are periodic, $u(1,t) = u(0,t)$.

We set $\sigma_o = 0.05$, $\sigma_q = 0.05$, and $\sigma_b = 0.1$. $\mathbf{C}_q$ and $\mathbf{C}_b$ have length-scales equal to $10\Delta z$. Every fourth model variable is observed at every fifth time step, ensuring that there is an observation at the final time step (100 observations in total). Because the model and the observational operator $\mathcal{H}_i$ are linear, the cost function (Equation 2) is quadratic and its minimiser is found in the first loop of the incremental method.

### 5.1.1 | Eigenvalues of the preconditioned matrix

We apply the randomised LMPs in the first inner loop. Note that if the deterministic LMP is used, it is unclear how to precondition the first inner loop. We explore what effect the randomised LMPs have on the eigenvalues of $\mathbf{A}$. The oversampling parameter $l$ is set to 5 and the randomised LMPs are constructed with $k = 25$ vectors.

The Ritz values of $\mathbf{A}$ given by the randomised methods are compared with those computed using *eigs* (Figure 1a). The Nyström method produces a good approximation of the largest eigenvalues, while REVD gives a slightly worse approximation, except for the largest five eigenvalues. The REVD_*ritzit* method underestimates the largest eigenvalues significantly. The largest eigenvalues of the preconditioned matrices are smaller than the largest eigenvalue of $\mathbf{A}$ (Figure 1b). However, the smallest eigenvalues of the preconditioned matrices are less than one and hence applying the preconditioner expands the spectrum of $\mathbf{A}$ at the lower boundary (Figure 1c), so that theorem 3.4 of Gratton *et al.* (2011), which considers the non-expansiveness of the spectrum of the Hessian after preconditioning with an LMP, does not hold. This happens because the formulation of spectral-LMP is derived assuming that the eigenvalues and eigenvectors are exact, while the randomized methods provide only approximations. Note that, even though REVD_*ritzit* gives the worst approximations of the largest eigenvalues of the Hessian, using the randomised LMP with information from REVD_*ritzit* reduces the largest eigenvalues of the preconditioned matrix the most and the smallest eigenvalues are close to one. Using the randomised LMP with estimates from Nyström gives similar results. Hence, the condition number of the preconditioned matrix is lower when the preconditioners are constructed with REVD_*ritzit* or Nyström compared with REVD.

The values of the quadratic cost function at the first ten iterations of PCG are shown in Figure 1d. Using the randomised LMP that is constructed with information from REVD is detrimental to the PCG convergence compared with using no preconditioning. Using information from the Nyström and REVD_*ritzit* methods results in similar PCG convergence and low values of the quadratic cost

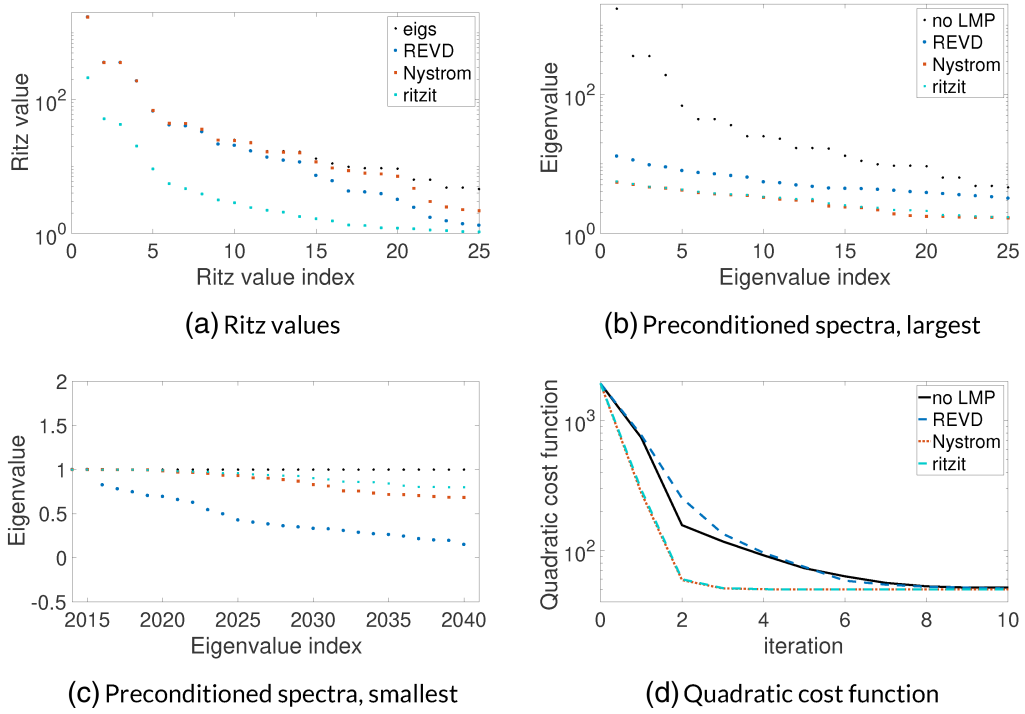


**FIGURE 1** Advection problem. (a) The 25 largest eigenvalues of $\mathbf{A}$ (*eigs*) and their estimates given by randomised methods; the largest eigenvalues and their estimates given by REVD and Nyström coincide. (b) The largest eigenvalues of $\mathbf{A}$ (no LMP, the same as *eigs* in (a)) and $(\mathbf{C}_{25}^{\text{sp}})^{\text{T}}\mathbf{A}\mathbf{C}_{25}^{\text{sp}}$, where $\mathbf{C}_{25}^{\text{sp}}$ is constructed with Ritz values in (a) and corresponding Ritz vectors. (c) The smallest eigenvalues of $\mathbf{A}$ and $(\mathbf{C}_{25}^{\text{sp}})^{\text{T}}\mathbf{A}\mathbf{C}_{25}^{\text{sp}}$. (d) Quadratic cost function value versus PCG iteration when solving systems with $\mathbf{A}$ and $(\mathbf{C}_{25}^{\text{sp}})^{\text{T}}\mathbf{A}\mathbf{C}_{25}^{\text{sp}}$ [Colour figure can be viewed at wileyonlinelibrary.com]

function are reached in fewer iterations than without preconditioning. The PCG convergence may be explained by the favourable distribution of the eigenvalues after preconditioning using Nyström and REVD_*ritzit*, and the smaller-than-one eigenvalues when using REVD. These results, however, do not necessarily generalize to an operational setting, as this system is well conditioned while operational settings are not. This will be investigated further in the next section.

## 5.2 | Lorenz 96 model

We next use the Lorenz 96 model to examine what effect the randomised LMPs have on PCG performance. In the Lorenz 96 model, the evolution of the $n$ components $X^j$, $j \in \{1, 2, \ldots, n\}$ of $\mathbf{x}_i$ is governed by a set of $n$ coupled ODEs:

$$\frac{dX^j}{dt} = -X^{j-2}X^{j-1} + X^{j-1}X^{j+1} - X^j + F, \qquad (22)$$

where $X^{-1} = X^{n-1}$, $X^0 = X^n$, $X^{n+1} = X^1$, and $F = 8$. The equations are integrated using a fourth-order Runge–Kutta scheme (Butcher, 1987). We set $n = 80$ and $N = 150$ (the size of $\mathbf{A}$ is $12{,}080 \times 12{,}080$) and observe every tenth model variable at every tenth time step (120 observations in total), ensuring that there are observations at the final time step. The grid-point distance is $\Delta X = 1/n$ and the time step is set to $\Delta t = 2.5 \times 10^{-2}$.

For the covariance matrices we use $\sigma_{\text{o}} = 0.15$ and $\sigma_{\text{b}} = 0.2$. $\mathbf{C}_{\text{b}}$ has length-scale equal to $2\Delta X$. Two setups are used for the model-error covariance matrix:

- $\sigma_q = 0.1$ and $\mathbf{C}_q$ has length-scale $L_q = 2\Delta X$ (the same as $\mathbf{C}_{\text{b}}$);
- $\sigma_q = 0.05$ and $\mathbf{C}_q$ has length-scale $L_q = 0.25\Delta X$.

In our numerical experiments, the preconditioners have very similar effect using both setups. Hence, we present results for the case $\sigma_q = 0.1$ and $L_q = 2\Delta X$ in Figures 2–5 in the following sections, except Figure 3.

The first outer loop is performed and no second-level preconditioning is used in the first inner loop, where PCG is run for 100 iterations or until the relative residual norm reaches $10^{-6}$. In the following sections, we use randomised and deterministic LMPs in the second inner loop. PCG has the same stopping criteria as in the first inner loop.

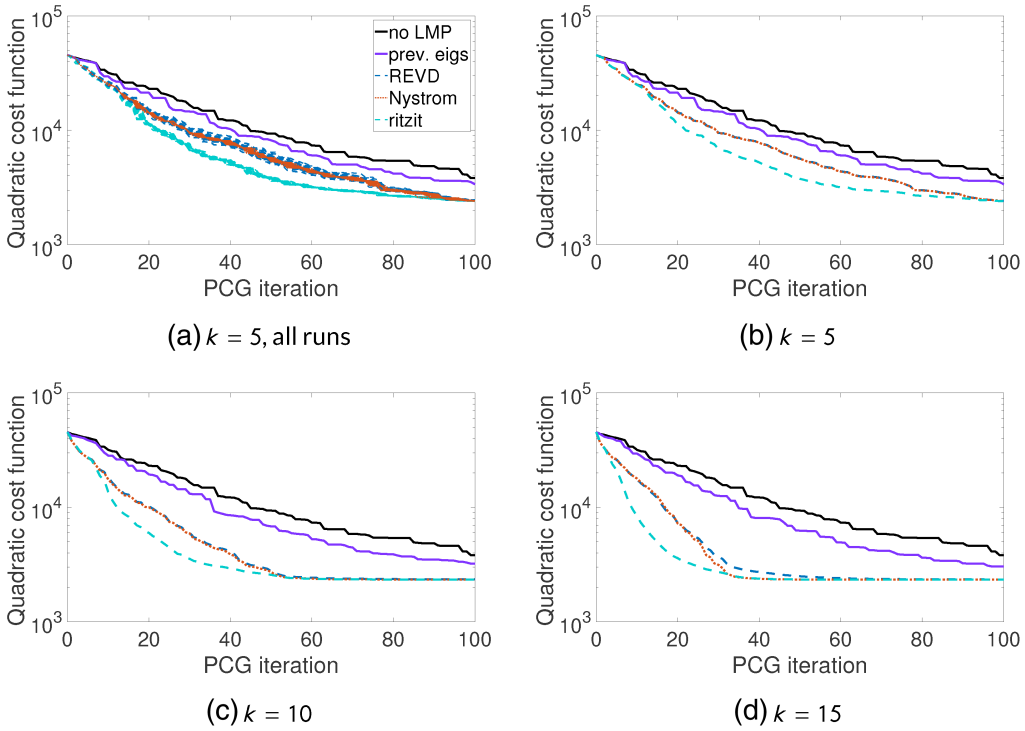


**FIGURE 2** A comparison of the value of the quadratic cost function at every PCG iteration when spectral-LMP is constructed with $k \in \{5, 10, 15\}$ Ritz values and vectors obtained with the randomised methods in the current inner loop, and function *eigs* in the previous inner loop. We also show no second-level preconditioning (no LMP), which is the same in all four panels. For the randomised methods, (a) shows 50 experiments for $k = 5$ and the remaining panels display means over 50 experiments. Here $\sigma_q = 0.1$ and $L_q = 2\Delta X$ [Colour figure can be viewed at wileyonlinelibrary.com]

### 5.2.1 | Minimising the inner loop cost function

In Figure 2, we compare the performance of the randomised LMPs with the deterministic LMP. We also consider the effect of varying $k$, the number of vectors used to construct the preconditioner. We set the oversampling parameter to $l = 5$. Because results from the randomized methods depend on the random matrix used, we perform 50 experiments with different realizations for the random matrix. We find that the different realizations lead to very similar results (see Figure 2a).

Independently of the $k$ value, there is an advantage in using second-level preconditioning. The reduction in the value of the quadratic cost function is faster using randomised LMPs compared with deterministic LMPs, with REVD_*ritzit* performing the best after the first few iterations. The more information we use in the preconditioner (i.e., the higher the $k$ value), the faster REVD_*ritzit* shows better results than the other methods. The performance of the REVD and Nyström methods is similar. Note that, as $k$ increases, the storage (see Table 1) and work per PCG iteration increase. Examination of the Ritz values given by the randomised methods shows that REVD_*ritzit* gives the worse estimate of the largest eigenvalues, as was the case when using the advection model. We calculated the smallest eigenvalue of the preconditioned matrix $(\mathbf{C}_5^{\text{sp}})^{\text{T}}\mathbf{A}\mathbf{C}_5^{\text{sp}}$ using *eigs*. When $\mathbf{C}_5^{\text{sp}}$ is constructed using REVD_*ritzit* or Nyström, the smallest eigenvalue of $(\mathbf{C}_5^{\text{sp}})^{\text{T}}\mathbf{A}\mathbf{C}_5^{\text{sp}}$ is equal to one, whereas using REVD it is approximately 0.94. This may explain why the preconditioner constructed using REVD does not perform as well as other randomised preconditioners, but it is not entirely clear why the preconditioner that uses REVD_*ritzit* shows the best performance.

The PCG convergence when using the deterministic LMP and the randomised LMP with information from REVD_*ritzit* with different $k$ values is compared in Figure 3 for both setups of the model-error covariance matrix. We also show an additional case where the model-error covariance matrix is constructed setting $\sigma_q = \sigma_b/100 = 0.002$ and $L_q = 0.25\Delta X$. In this case, the performance of the REVD and Nyström methods is very similar, outperforming no preconditioning after the first 10–15 iterations, with better performance for higher $k$ values (results not shown). Moreover, REVD_*ritzit* again outperforms the deterministic LMP from the first PCG iterations. For the deterministic LMP in Figure 3, varying $k$ has little effect, especially in the initial iterations. However, for REVD_*ritzit*, in general increasing $k$ results in a greater decrease of the cost function. Setting $k = 5$ gives better

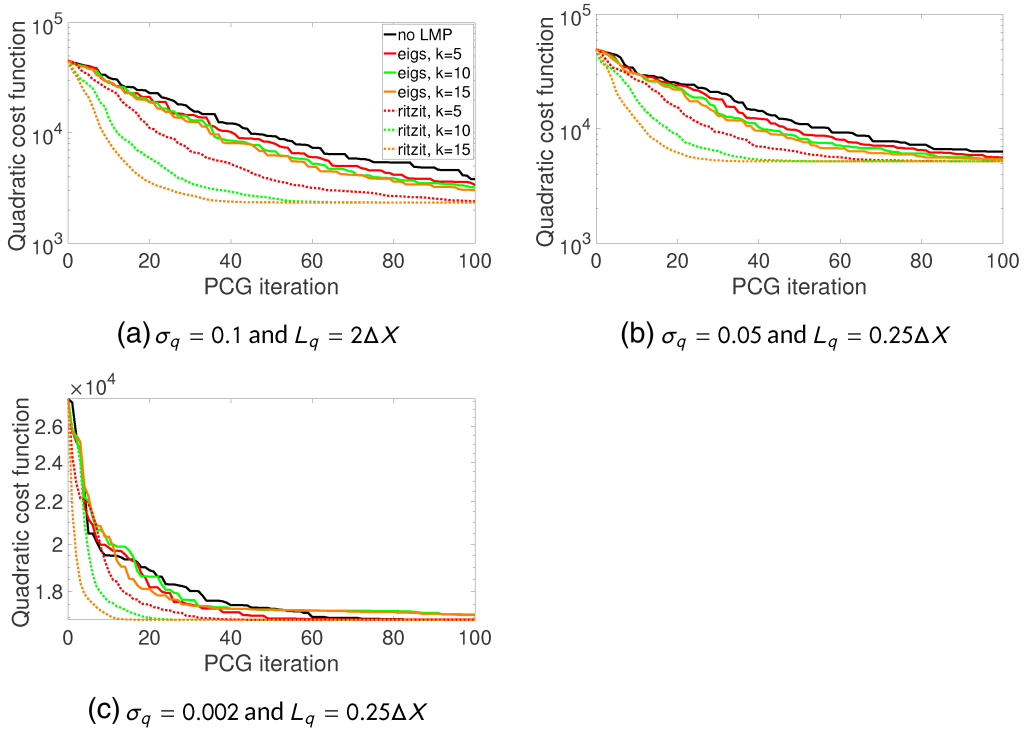


**FIGURE 3** A comparison of the values of the quadratic cost function at every PCG iteration when using deterministic LMP with information from the previous loop (eigs) and randomised LMP with information from REVD_*ritzit* for different $k$ values (5, 10, and 15). No second-level preconditioning is also shown (case (a) is the same as in Figure 2). In cases (a), (b), and (c) the model-error covariance matrices are constructed using parameters $\sigma_q$ and $L_q$ [Colour figure can be viewed at wileyonlinelibrary.com]

initial results compared with $k = 10$ in the $\sigma_q = 0.002$ case, but the larger $k$ value performs better after that. Also, at any iteration of PCG we obtain a lower value of the quadratic cost function using the randomised LMP with $k = 5$ compared with the deterministic LMP with $k = 15$, which uses exact eigenpair information from the Hessian of the previous loop.

### 5.2.2 | Effect of the observation network

To understand the sensitivities of results from the different LMPs to the observation network, we consider a system with the same parameters as in the previous section, where we had 120 observations, but we now observe the following:

- every fifth model variable at every fifth time step (480 observations in total);
- every second variable at every second time step (3,000 observations in total).

The oversampling parameter is again set to $l = 5$ and we set $k = 5$ and $k = 15$ for both observation networks. Since the number of observations is equal to the number of eigenvalues that are larger than one and there are more observations than in the previous section, there are more eigenvalues that are larger than one after first-level preconditioning. Because all 50 experiments with different Gaussian matrices in the previous section were close to the mean, we perform 10 experiments for each randomised method, solve the systems, and report the means of the quadratic cost function.

The results are presented in Figure 4. Again, the randomised LMPs perform better than the deterministic LMP. However, if the preconditioner is constructed with a small amount of information about the system ($k = 5$ for both systems and $k = 15$ for the system with 3,000 observations), then there is little difference in the performance of different randomised LMPs. Also, when the number of observations is increased, more iterations of PCG are needed to get any improvement in the minimisation of the quadratic cost function when using the deterministic LMP over using no second-level preconditioning.

When comparing the randomised and deterministic LMPs with different values of $k$ for these systems, we obtain similar results to those in Figure 3a, that is, it is more advantageous to use the randomised LMP constructed with $k = 5$ than the deterministic LMP constructed with $k = 15$.

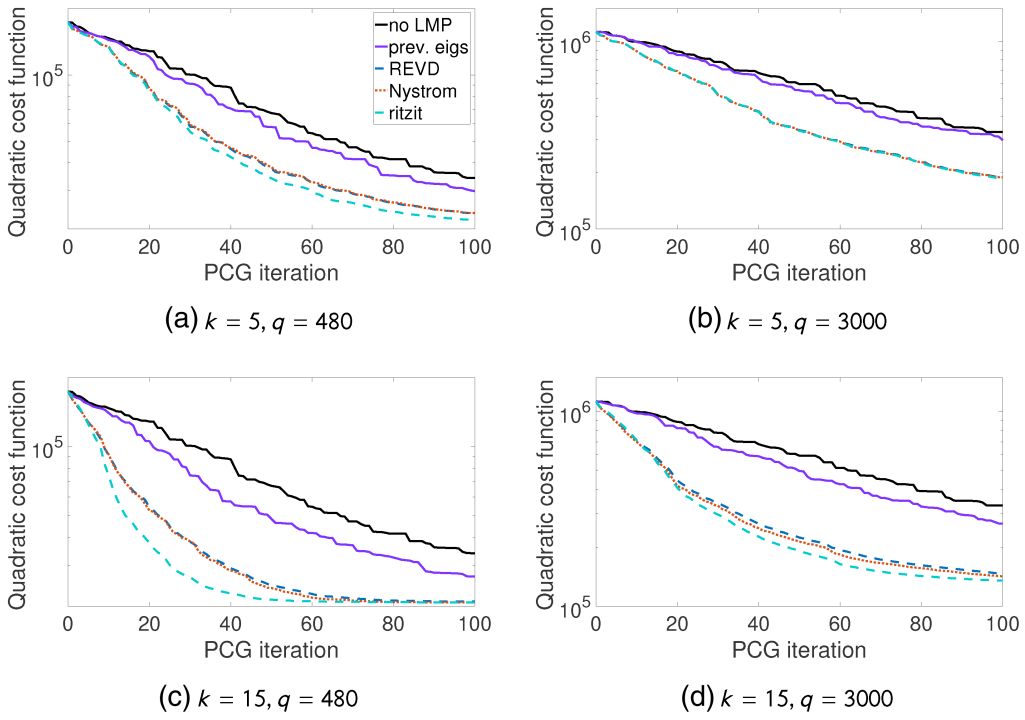


**FIGURE 4** As in Figure 2, but for two systems with $q$ observations; 10 experiments are done for each randomised method and the mean values plotted [Colour figure can be viewed at wileyonlinelibrary.com]

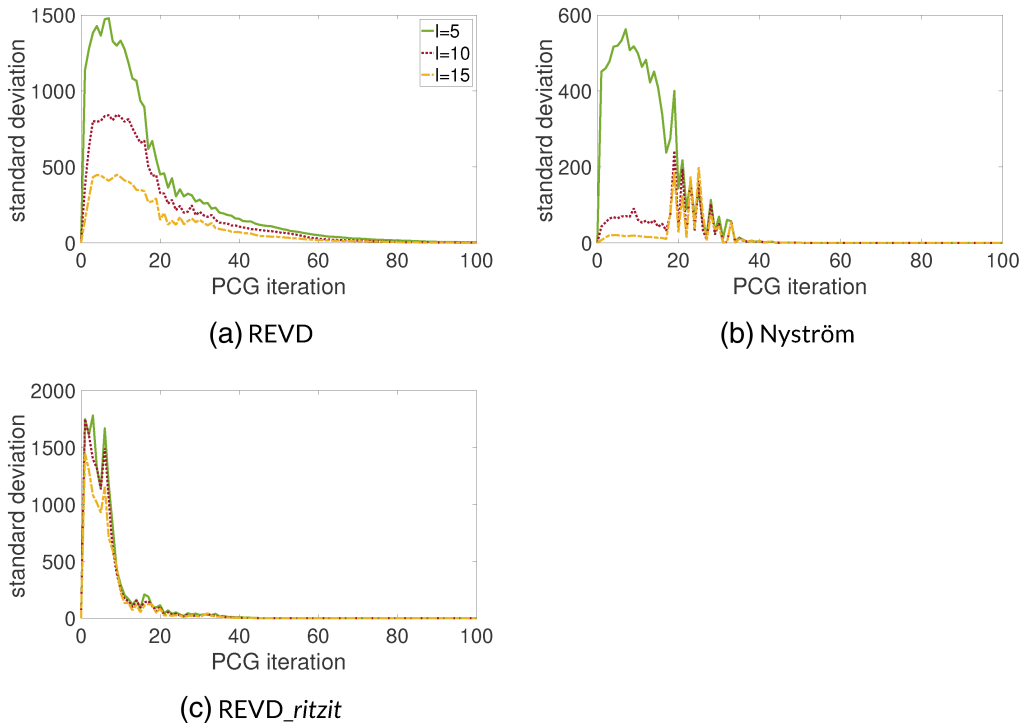


**FIGURE 5** Standard deviation of the quadratic cost function at every iteration of PCG when spectral-LMP is constructed with different randomised methods. For every randomised method we perform 50 experiments. Here $\sigma_q = 0.1$ and $L_q = 2\Delta X$ [Colour figure can be viewed at wileyonlinelibrary.com]

### 5.2.3 | Effect of oversampling

We next consider the effect of increasing the value of the oversampling parameter $l$. The observation network is as in Section 5.2.1 (120 observations in total). We set $k = 15$ and perform the second inner loop 50 times for every value of $l \in \{5, 10, 15\}$ with all three randomised methods. The standard deviation of the value of the quadratic cost function at every iteration is presented in Figure 5.

For all the methods, the standard deviation is greatest in the first iterations of PCG. It is reduced when the value of $l$ is increased and the largest reduction happens in the first iterations. However, REVD_*ritzit* is the least sensitive to the increase in oversampling. With all values of $l$, REVD_*ritzit* has the largest standard deviation in the first few iterations, but it stills gives the largest reduction of the quadratic cost function. Hence, large oversampling is not necessary if REVD_*ritzit* is used.

## 6 | CONCLUSIONS AND FUTURE WORK

We have proposed a new randomised approach to second-level preconditioning of the incremental weak-constraint 4D-Var forcing formulation. It can be preconditioned with an LMP that is constructed using approximations of eigenpairs of the Hessian. Previously, by using the Lanzcos and CG connection these approximations were obtained at a very low cost in one inner loop and then used to construct the LMP in the following inner loop. We have considered three methods (REVD, Nyström, and REVD_*ritzit*) that employ randomisation to compute the approximations. These methods can be used to construct the preconditioner cheaply in the current inner loop, with no dependence on the previous inner loop, and are parallelisable.

Numerical experiments with the linear advection and Lorenz-96 models have shown that the randomised LMPs constructed with approximate eigenpairs improve the convergence of PCG more than deterministic LMPs with information from the previous loop, especially after the initial PCG iterations. The quadratic cost function reduces more rapidly when using a randomised LMP rather than a deterministic LMP, even if the randomised LMP is constructed with fewer vectors than the deterministic LMP. Also, for randomised LMPs, the more information about the system we use (i.e., the more approximations of eigenpairs are used to construct the preconditioner), the greater the reduction in the quadratic cost function, with a possible exception in the first PCG iterations for low $k$ values and very small model error. Using more information to construct a deterministic LMP may not result in larger reduction of the quadratic cost function, especially in the first iterations of PCG, which is in line with the results in Tshimanga *et al.* (2008). However, if not enough information is included in the randomised LMP, then preconditioning may have no effect on the first few iterations of PCG.

Of the randomised methods considered, the best overall performance was for REVD_*ritzit*. However, if we run a small number of PCG iterations, the preconditioners obtained with different randomised methods give similar results. In the case of a very small model error, using REVD and Nyström is useful after the initial iterations of PCG, whereas REVD_*ritzit* improves the reduction of the quadratic cost function from the start. The performance is independent of the choice of random Gaussian start matrix and it may be improved with oversampling.

In this work we apply randomised methods to generate a preconditioner, which is then used to accelerate the solution of the exact inner loop problem (Equation 11) with the PCG method (as discussed in Section 4). A different approach has been explored by Bousserez and Henze (2018) and Bousserez *et al.* (2020), who presented and tested a randomised solution algorithm called the Randomized Incremental Optimal Technique (RIOT) in data assimilation. RIOT is designed to be used instead of PCG and employs a randomised eigenvalue decomposition of the Hessian (using a different method from the ones presented in this article) to construct directly the solution $\mathbf{x}$ in Equation (11), which approximates the solution given by PCG.

The randomised preconditioning approach can also be employed to minimise other quadratic cost functions, including the strong-constraint 4D-Var formulation. Further exploration of other single-pass versions of randomised methods for eigenvalue decomposition, which are discussed in Halko *et al.* (2011), may be useful. In particular, the single-pass version of the Nyström method is potentially attractive. If a large number of Ritz vectors are used to construct the preconditioner, more attention can be paid to choosing the value of the oversampling parameter $l$ in the randomised methods. In some cases, a better approximation may be obtained if $l$ depends linearly on the target rank of the approximation (Nakatsukasa, 2020).

## AUTHOR CONTRIBUTIONS

**Ieva Daužickaitė:** conceptualization, formal analysis, investigation, methodology, software, validation, visualization, writing-original draft, writing-review and editing. **Amos S. Lawless:** investigation; methodology; supervision; validation; writing – review and editing. **Jennifer A. Scott:** funding acquisition; investigation; methodology; project administration; resources; supervision; validation;

writing – review and editing. **Peter Jan van Leeuwen:** investigation; methodology; supervision; validation; writing – review and editing.

## ACKNOWLEDGEMENTS

We are grateful to Dr Adam El-Said for his code for the weak-constraint 4D-Var assimilation system. We also thank two anonymous reviewers, whose comments helped us to improve the article.

## CONFLICT OF INTEREST

The authors declare no conflict of interest.

## ORCID

*Ieva Daužickaitė* https://orcid.org/0000-0002-1285-1764
*Amos S. Lawless* https://orcid.org/0000-0002-3016-6568
*Jennifer A. Scott* https://orcid.org/0000-0003-2130-1091
*Peter Jan van Leeuwen* https://orcid.org/0000-0003-2325-5340

## REFERENCES

Bousserez, N., Guerrette, J.J. and Henze, D.K. (2020) Enhanced parallelization of the incremental 4D-Var data assimilation algorithm using the randomized incremental optimal technique. *Quarterly Journal of the Royal Meteorological Society*, 146, 1351–1371.

Bousserez, N. and Henze, D.K. (2018) Optimal and scalable methods to approximate the solutions of large-scale Bayesian problems: theory and application to atmospheric inversion and data assimilation. *Quarterly Journal of the Royal Meteorological Society*, 144, 365–390.

Butcher, J.C. (1987) *The Numerical Analysis of Ordinary Differential Equations: Runge–Kutta and General Linear Methods*. Chichester, UK: Wiley-Interscience.

Daley, R. (1993) *Atmospheric Data Analysis* Vol. 2. Cambridge, UK: Cambridge University Press.

Demmel, J., Grigori, L., Hoemmen, M. and Langou, J. (2012) Communication-optimal parallel and sequential QR and LU factorizations. *SIAM Journal on Scientific Computing*, 34, A206–A239.

Duff, I.S. and Scott, J.A. (1993) Computing selected eigenvalues of sparse unsymmetric matrices using subspace iteration. *ACM Transactions on Mathematical Software*, 19, 137–159.

ECMWF (2020). *Part II: Data Assimilation.* No. 2 in IFS Documentation. European Centre for Medium Range Weather Forecasts. Available at.

Fisher, M. (1998). Minimization algorithms for variational data assimilation, *Proceedings of the Seminar on Recent Developments in Numerical Methods for Atmospheric Modelling*, pp. 364–385: Reading, UK: European Centre for Medium Range Weather Forecasts.

Giraud, L. and Gratton, S. (2006) On the sensitivity of some spectral preconditioners. *SIAM Journal on Matrix Analysis and Applications*, 27, 1089–1105.

Golub, G.H. and Van Loan, C.F. (2013) *Matrix Computations*. 4th, Baltimore: JHU Press. 4th edn.

Gratton, S., Gürol, S., Simon, E. and Toint, P.L. (2018) A note on preconditioning weighted linear least-squares, with consequences for weakly constrained variational data assimilation. *Quarterly Journal of the Royal Meteorological Society*, 144, 934–940.

Gratton, S., Lawless, A.S. and Nichols, N.K. (2007) Approximate Gauss–Newton methods for nonlinear least squares problems. *SIAM Journal on Optimization*, 18, 106–132.

Gratton, S., Sartenaer, A. and Tshimanga, J. (2011) On a class of limited-memory preconditioners for large scale linear systems with multiple right-hand sides. *SIAM Journal on Optimization*, 21, 912–935.

Gu, M. (2015) Subspace iteration randomization and singular value problems. *SIAM Journal on Scientific Computing*, 37, A1139–A1173.

Halko, N., Martinsson, P. and Tropp, J. (2011) Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, 53, 217–288.

Hestenes, M.R. and Stiefel, E. (1952) Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards*, 49, 409–436.

Johnson, C., Hoskins, B.J. and Nichols, N.K. (2005) A singular vector perspective of 4D-Var: filtering and interpolation. *Quarterly Journal of the Royal Meteorological Society*, 131, 1–19.

Kalnay, E. (2002) *Atmospheric Modeling. Data Assimilation and Predictability*. Cambridge, UK: Cambridge University Press.

Laloyaux, P., Frolov, S., Ménétrier, B. and Bonavita, M. (2018) Implicit and explicit cross-correlations in coupled data assimilation. *Quarterly Journal of the Royal Meteorological Society*, 144, 1851–1863.

Lawless, A.S. (2013). Variational data assimilation for very large environmental problems. In M. Cullen, M.A. Freitag, S. Kindermann, and R. Scheichl (Eds.), *Large Scale Inverse Problems: Computational Methods and Applications in the Earth Sciences. Radon Series on Computational and Applied Mathematics*, pp. 55–90: 13 Berlin, Boston: De Gruyter.

Lorenz, E. (1996). Predictability – a problem partly solved. In:, *Proceedings of the Seminar on Predictability* Vol. 1, Reading, UK: European Centre for Medium RangeWeather Forecasts, pp. 1–18.

Martinsson, P.G. and Tropp, J.A. (2020) Randomized numerical linear algebra: foundations and algorithms. *Acta Numerica*, 29, 403–572.

Mogensen, K., Alonso Balmaseda, M. and Weaver, A. (2012). The NEMOVAR ocean data assimilation system as implemented in the ECMWF ocean analysis for System 4. ECMWF Technical Memorandum. Reading, UK: European Centre for Medium Range Weather Forecasts.

Moore, A.M., Arango, H.G., Broquet, G., Powell, B.S., Weaver, A.T. and Zavala-Garay, J. (2011) The regional ocean modeling system (ROMS) 4-dimensional variational data assimilation systems: part I – system overview and formulation. *Progress in Oceanography*, 91, 34–49.

Morton, K.W. and Mayers, D. (1994) *Numerical Solution of Partial Differential Equations* Vol. 2. Cambridge, UK: Cambridge University Press.

Nakatsukasa, Y. (2020). Fast and stable randomized low-rank matrix approximation.

Nocedal, J. and Wright, S.J. (2006) *Numerical Optimization* (2nd edn). New York: World Scientific.

Parlett, B.N. (1998) *The Symmetric Eigenvalue Problem*. Classics in Applied Mathematics, Philadelphia: SIAM.

Rutishauser, H. (1971). Contribution II/9: Simultaneous iteration method for symmetric matrices. In J. H. Wilkinson and C. Reinsch (Eds.), *Handbook for Automatic Computation: Volume II: Linear Algebra. Die Grundlehren der mathematischen Wissenschaften*, pp. 284–302: Berlin, Heidelberg/Germany: Springer-Verlag.

Saad, Y. (2003) *Iterative Methods for Sparse Linear Systems* (2nd edn). Philadelphia: SIAM.

Saad, Y. (2011) *Numerical Methods for Large Eigenvalue Problems* (Revised edition). Philadelphia: SIAM.

Schnabel, R. (1983). Quasi-Newton methods using multiple secant equations. Technical Report CU-CS-247-83, Boulder, CO.

Stewart, G.W. (2001) *Matrix Algorithms: Volume II: Eigensystems*. Philadelphia: SIAM.

Tang, J.M., Nabben, R., Vuik, C. and Erlangga, Y.A. (2009) Comparison of two-level preconditioners derived from deflation, domain decomposition and multigrid methods. *Journal of Scientific Computing*, 39, 340–370.

Trefethen, L.N. and Bau, D. (1997) *Numerical Linear Algebra*. Philadelphia: SIAM.

Trémolet, Y. (2006) Accounting for an imperfect model in 4D-Var. *Quarterly Journal of the Royal Meteorological Society*, 132, 2483–2504.

Trémolet, Y. (2007) Model-error estimation in 4D-Var. *Quarterly Journal of the Royal Meteorological Society*, 133, 1267–1280.

Tshimanga, J. (2007). On a class of limited memory preconditioners for large-scale nonlinear least-squares problems (with application to variational ocean data assimilation). PhD Thesis, Department of Mathematics, University of Namur, Belgium.

Tshimanga, J., Gratton, S., Weaver, A.T. and Sartenaer, A. (2008) Limited-memory preconditioners, with application to incremental four-dimensional variational data assimilation. *Quarterly Journal of the Royal Meteorological Society*, 134, 751–769.

**How to cite this article:** Daužickaitė, I., Lawless, A.S., Scott, J.A. & van Leeuwen, P.J. (2021) Randomised preconditioning for the forcing formulation of weak-constraint 4D-Var. *Quarterly Journal of the Royal Meteorological Society*, 1–16. Available from: https://doi.org/10.1002/qj.4151