# Accelerating linear system solvers for time domain component separation of cosmic microwave background data

Jan Papež, Laura Grigori, Radek Stompor

*Astronomy & Astrophysics* manuscript no. output
February 7, 2020


# Accelerating linear system solvers for time domain component separation of cosmic microwave background data

J. Papež[1⋆], L. Grigori[2], R. Stompor[3,4]

[1] INRIA Paris, Sorbonne Université, Université Paris-Diderot SPC, CNRS, Laboratoire Jacques-Louis Lions, ALPINES team, France
Currently at Institute of Mathematics, Czech Academy of Sciences, Prague, Czech Republic
[2] INRIA Paris, Sorbonne Université, Université Paris-Diderot SPC, CNRS, Laboratoire Jacques-Louis Lions, ALPINES team, France
[3] Université de Paris, CNRS, AstroParticule et Cosmologie, F-75013 Paris, France
[4] CNRS-UCB International Research Laboratory, "Centre Pierre Binétruy", UMI2007, CPB-IN2P3

February 7, 2020

**ABSTRACT**

Component separation is one of the key stages of any modern, cosmic microwave background (CMB) data analysis pipeline. It is an inherently non-linear procedure and typically involves a series of sequential solutions of linear systems with similar, albeit not identical system matrices, derived for different data models of the same data set. Sequences of this kind arise for instance in the maximization of the data likelihood with respect to foreground parameters or sampling of their posterior distribution. However, they are also common in many other contexts. In this work we consider solving the component separation problem directly in the measurement (time) domain, which can have a number of important advantageous over the more standard pixel-based methods, in particular if non-negligible time-domain noise correlations are present as it is commonly the case. The time-domain based approach implies, however, significant computational effort due to the need to manipulate the full volume of time-domain data set. To address this challenge, we propose and study efficient solvers adapted to solving time-domain-based, component separation systems and their sequences and which are capable of capitalizing on information derived from the previous solutions. This is achieved either via adapting the initial guess of the subsequent system or through a so-called subspace recycling, which allows to construct progressively more efficient, two-level preconditioners. We report an overall speed-up over solving the systems independently of a factor of nearly 7, or 5, in the worked examples inspired respectively by the likelihood maximization and likelihood sampling procedures we consider in this work.

**Key words.** Numerical methods - linear systems solvers - cosmic microwave background data analysis - component separation

## 1. Context and motivation

Measurements registered by the cosmic microwave background (CMB) experiments in addition to the sought-after signal of cosmological origin contain contributions generated by astrophysical sources. These are generically called foregrounds and can be of either galactic or extra-galactic origins, either diffuse or point-source like. An essential step of the CMB data analysis is therefore that of separating the foreground signals from each other and, specifically, from the CMB one, capitalizing on their different electromagnetic frequency dependence and/or statistical properties than those of the CMB (e.g., Planck Collaboration et al. 2016a, and references therein). This process is referred to as component separation. As in polarization the foreground signals tend to dominate the cosmological one over a broad range of angular scales and observational frequencies the next generation of the CMB observatories will be only capable of delivering its science in full if a high precision, statistically sound and reliable component separation techniques and their numerically efficient implementations are available.

The component separation is a non-linear procedure, which from the data measured at multiple different frequency bands aims at estimating simultaneously properties of the foregrounds as well as the signals themselves. This is commonly accomplished in a pixel domain assuming that maps of the sky for each frequency band have been already reconstructed and their statistical uncertainties somehow estimated as part of the initial procedure commonly called a map-making. Then the component separation is typically performed in two steps. On the first step, the foreground parameters are estimated, and, on the second step, they are used to disentangle the frequency maps into maps of sky components. For concreteness, we focus in this work on the so-called parametric component separation approach (e.g., Brandt et al. 1994; Eriksen et al. 2008; Stompor et al. 2009), where the frequency scaling relations for each considered sky component are assumed to be given and known up to a limited number of unknowns, called foreground spectral parameters, which are then estimated from the data on the first step of the component separation. However, the numerical techniques discussed here are more general and should be found useful also in alternative component separation methods.

The two-step, pixel-domain based approach outlined above is conceptually simple and computationally potentially very efficient as the maps conveniently compress the information contained in a typically much larger initial raw data set, referred to hereafter as the time domain data set, and do so in a nearly lossless fashion. Indeed, for the next generation of the CMB experiments we expect to have as many as $n_{\rm t} \sim O(10^{13} - 10^{15})$ raw measurements but only $n_{\rm pix} \sim O(10^5 - 10^8)$ sky pixels. This compression however can only work satisfactorily if a manageable, but sufficiently precise, statistical description of these maps

⋆ e-mail: jan@papez.org

can be derived. In practice, this is often difficult due to storage and computational cycles limitations. A general, full covariance matrix of a single frequency map contains ${n_{\rm pix}}^2 \sim O(10^{10} - 10^{16})$ elements, all of which need to be first computed, at the cost of at least of $O(\lambda)\,O(n_{\rm t})$ floating point operations (flops), and later stored in a computer memory. (Here $\lambda$ is a time-domain noise correlation length and can reach many thousands of samples.) The full computations therefore quickly become prohibitively expensive even if an explicit inversion of the covariance matrix is replaced by some iterative procedure which typically requires $O(n_{\rm iter}\,{n_{\rm pix}}^2)$ flops, where the number of iterations, $n_{\rm iter}$ is usually on order of $10^2$. Consequently, in practice the best way forward may be to invoke some approximations. This however is often problematic as well, as a successful approximation needs to ensure sufficient accuracy to avoid introducing systematic biases in the estimated foreground parameters and later also in the component maps, while being computationally feasible.

A general solution to the problem would be to avoid resorting to the frequency maps at all and to perform all the calculation in the time data domain. This would typically require $O(n_{\rm iter}\,n_{\rm t}\,\ln\,\lambda)$ flops and the memory on order of $O(n_{\rm t})$ and would not only be more robust but also more manageable whenever significant time-domain noise correlations are present, $\lambda \gg 1$. Nevertheless, this approach also faces significant numerical challenges. In this work, we explore some of the avenues which could render this approach feasible. We first identify the major, computation-heavy step which unavoidably appears in any implementation of this technique and then investigate how it could be accelerated by employing better, more advanced methods and their implementations.

The plan of this paper is as follows. In Section 2 we present the component separation problem and the numerical challenges it poses. In Section 3 we describe the proposed solution and in Section 4 results of the numerical tests. Section 5 provides a brief summary and outlines prospects for the future. Material which is more technical in nature or which is added for completeness is as usually deferred to the Appendices.

## 2. Problem description and setting

### 2.1. Preliminaries.

Hereafter, we consider polarized signals and assume, for simplicity and definiteness, that in every pixel on the sky the signal is characterized by two Stokes parameters, $Q$ and $U$. Extensions to include total intensity, $I$, are straightforward. Consequently, hereafter every considered sky map consists of two maps each corresponding to one of the two Stokes parameters and which are concatenated together in a single map vector,

$$\mathsf{v} \;=\; \left\lceil v_q, v_u \right\rfloor \equiv \begin{bmatrix} v_q \\ v_u \end{bmatrix}, \qquad v_q, v_u \in \mathbb{R}^{n_{\rm pix}}. \tag{2.1}$$

Hereafter, partial brackets, $\lceil \ldots \rfloor$, denote a vertical object. Examples of the sky maps as discussed in the following are single frequency maps storing information about the sky signal as observed at a given frequency or single sky component maps containing information about a sky signal of some specific physical origin. We refer to the ordering defined above as Stokes-wise, as a complete sky map of one Stokes parameter is followed up by another. In addition, we will also consider a pixel-wise ordering which for single maps reads,

$$\mathsf{v} \;\equiv\; \left\lceil v_q(1), v_u(1), \ldots, v_q(n_{\rm pix}), v_u(n_{\rm pix}) \right\rfloor, \tag{2.2}$$

where $Q$ and $U$ parameters of a signal in one pixel are stored consecutively and followed by those in another.

The goal of the component separation procedure is to estimate all assumed sky component signals given multiple frequency data. Therefore, commonly, we will be dealing with multiple maps of the same type, say, multiple single frequency maps or multiple single component maps, and will concatenate them in a single, multi-frequency or multi-component vector. For definiteness, in this work we fix the number of components to $n_{\rm comp} = 3$ and consider three different sky components, CMB, dust, and synchrotron. A multi-component vector, $\mathbb{s}$, will therefore contain information about $Q$ and $U$ Stokes parameters of all three components. Such a vector can be ordered in multiple ways and most commonly we will assume to be ordered either in a component-wise way, when,

$$\begin{aligned} \mathbb{s} \;&\equiv\; \left\lceil \mathsf{s}_{\rm cmb}, \mathsf{s}_{\rm dust}, \mathsf{s}_{\rm sync} \right\rfloor \\ &=\; \left\lceil s_{{\rm cmb},q}, s_{{\rm cmb},u}, s_{{\rm dust},q}, s_{{\rm dust},u}, s_{{\rm sync},q}, s_{{\rm sync},u} \right\rfloor \in \mathbb{R}^{6n_{\rm pix}}, \end{aligned} \tag{2.3}$$

or a pixel-wise way, where for each pixel all Stokes parameters follow consecutively for all considered components, i.e.,

$$\begin{aligned} \mathbb{s} \;\equiv\; \lceil & s_{{\rm cmb},q}(1), s_{{\rm cmb},u}(1), \ldots, s_{{\rm sync},q}(1), s_{{\rm sync},u}(1), \ldots \\ & s_{{\rm cmb},q}(n_{\rm pix}), s_{{\rm cmb},u}(n_{\rm pix}), \ldots, s_{{\rm sync},q}(n_{\rm pix}), s_{{\rm sync},u}(n_{\rm pix}) \rfloor. \end{aligned} \tag{2.4}$$

Multi-frequency vectors can be ordered in analogous manners.

The choice of the ordering will in general depend on the specific context and is obviously of a key importance for a numerical implementation of the map-making or component separation procedures. Nonetheless, mathematically, switching the ordering from one to another, is described by a linear, orthonormal, full-rank operator, $U$, which is conceptually trivial to apply and an application of which commutes with other matrix operations such as a matrix inversion, given that,

$$(U\,M\,U^t)^{-1} \;=\; U\,M^{-1}\,U^t, \tag{2.5}$$

for any invertible matrix, $M$. Consequently, one can perform a matrix inversion using one ordering, say, computing $M^{-1}$, and reorder the result afterwards to obtain the inverse in the other ordering scheme, i.e., $(U\,M\,U^t)^{-1}$. For this reason, in the following we will freely switch between the different orderings depending on the context in order to highlight specific structures of the matrices, which may be more apparent for one choice than the other.

### 2.2. Data model.

As mentioned earlier we consider a component separation procedure performed directly on the time-domain data as measured by the instrument thus avoiding performing any prior, explicit map-making procedure. For this we need to relate the data directly to amplitudes of the component maps in the predefined sky pixels as these maps are the targeted outcome of the component separation procedure. We assume that for each frequency the time-domain data are made of sequences of consecutive observations registered by all detectors operating at this frequency and concatenated together, we can write,

$$d_f = P_{\beta^\star,f}\,\mathbb{s}_\star + n_f, \qquad d_f, n_f \in \mathbb{R}^{n_t}, \qquad f = 1, \ldots, n_{\rm freq}, \tag{2.6}$$

where $\mathbb{s}_\star$ is the unknown vector of the component amplitudes and the star indicates that those are their actual values, $n_f$ is an (unknown) noise vector, and the number of the frequency channels, $n_{\rm freq}$, is assumed to be larger than that of the components,

$n_{\rm comp}$, set to 3 in this work, to ensure that the problem is well-defined.

The matrix, $P_{\beta^\star,f}$, appearing in Eq. (2.6) combines the information about the instrument operations and the sky properties, and can be expressed as,

$$P_{\beta^\star,f} = P_f \cdot M_{\beta^\star,f}. \tag{2.7}$$

where $M_{\beta^\star,f} \in \mathbb{R}^{2n_{\rm pix}\times 6n_{\rm pix}}$ is a so-called *mixing matrix*, which determines how different sky components mix together at all observed frequencies to yield the observed signal. The mixing matrix depends explicitly on the foreground scaling parameters, which we denote as $\beta^\star$, and frequency of the observation, $f$. $P_f \in \mathbb{R}^{n_t\times 2n_{\rm pix}}$ is in turn a *pointing matrix* defining which pixel of the sky each detector operating at a given frequency observed at every time. So while it does not explicitly depend on frequency or scaling parameters it is in principle different for different frequencies as it encodes pointing of detectors specific to this frequency, the fact which we highlight by the subscript, $f$. We have,

$$P_f : \left\lceil s^\star_{f,q}, s^\star_{f,u}\right\rfloor \mapsto d_f, \qquad M_{\beta^\star,f} : \mathtt{s}_\star \mapsto \mathtt{s}_f \equiv \left\lceil s^\star_{f,q}, s^\star_{f,u}\right\rfloor, \tag{2.8}$$

where $\mathtt{s}^\star_f$ is a single frequency map expressing the combined sky signal at frequency $f$. The data vector, $d_f$, is time-ordered as its elements are indexed by the time at which the measurement was taken.

### 2.3. Component separation.

The goal of the component separation procedure is to solve an inverse problem, Eq. (2.6), and estimate the components, $\mathtt{s}_\star$, given the full data set, $d$ (:= $\{d_f\}$), made of data taken at all observational frequencies. This is typically solved by assuming that the noise, $n_f$, is Gaussian, with a vanishing mean and a known variance, $N_f$, and writing a data likelihood,

$$-2\ln\mathcal{L}(\beta,\mathtt{s};d) \;=\; (\widetilde{d}-\widetilde{P}_\beta\,\mathtt{s})^\top N^{-1}\,(\widetilde{d}-\widetilde{P}_\beta\,\mathtt{s}) \;+\; \textsc{const}, \tag{2.9}$$

where we have dropped $\star$ to distinguish between an estimate and the true value and we use tilde to denote multifrequency objects. We have,

$$\widetilde{P}_\beta = \begin{bmatrix} P_{\beta,1}\\ \vdots \\ P_{\beta,n_{\rm freq}}\end{bmatrix} = \begin{bmatrix} P_1\cdot M_{\beta,1}\\ \vdots \\ P_{n_{\rm freq}}\cdot M_{\beta,n_{\rm freq}}\end{bmatrix}, \tag{2.10}$$

which follows from Eq. (2.7), and,

$$\widetilde{N} = \begin{bmatrix} N_1 & & 0\\ & \ddots & \\ 0 & & N_{n_{\rm freq}}\end{bmatrix}, \quad \widetilde{d} = \begin{bmatrix} d_1\\ \vdots \\ d_{n_{\rm freq}}\end{bmatrix}, \tag{2.11}$$

which assumes no noise correlations between different frequency channels. In addition, throughout this work we also assume that while the component mixing represented by $M_\beta$ may involve (potentially) all components, it is always done on a pixel-by-pixel basis so all the elements of $M_\beta$ which corresponds to different pixels vanish. Similarly, and in agreement with assumptions made in map-making procedures, we will assume that the noise matrices, $N_f$, are block diagonal with each block representing a banded Toeplitz matrix.

The standard, two-step component separation procedure proceeds by first estimating for each frequency band, $f$, a single frequency map, $m_f$, and its covariance, $\hat{N}_f$, given by,

$$m_f \;=\; (P_f^\top N_f^{-1} P_f)^{-1} P_f^\top N_f^{-1}\, d_f, \tag{2.12}$$

$$\hat{N}_f \;=\; (P_f^\top N_f^{-1} P_f)^{-1}, \tag{2.13}$$

and then by performing component separation by modeling the single frequency maps yielded by the first step as,

$$m_f \;=\; M_{\beta^\star,f}\,\mathtt{s}_\star \;+\; \hat{n}_f, \tag{2.14}$$

where $\hat{n}_f$ stands for a pixel-domain noise and is a Gaussian variable with variance $\hat{N}_f$. We can therefore write the corresponding likelihood as,

$$-2\ln\mathcal{L}(\beta,\mathtt{s};\{m_f\}) = \\ = \sum_f (m_f - M_{\beta,f}\,\mathtt{s})^\top\,\hat{N}_f^{-1}\,(m_f - M_{\beta,f}\,\mathtt{s}) \;+\; \textsc{const}. \tag{2.15}$$

This procedure is equivalent to solving the maximum likelihood problem defined by Eq. (2.9) directly. However it requires an explicit calculation of $\hat{N}_f^{-1}$, which for the current and forthcoming experiment is typically prohibitive due to restrictions on both the available computer memory and computational cycles. An alternative could be to solve the original problem directly without invoking explicitly any pixel-domain objects. This is the option we study in this work. We note here in passing that intermediate approaches are also possible as for instance one which relies on the likelihood in Eq. (2.15), but does not assume that $\hat{N}_f^{-1}$ is given explicitly. Instead, it computes a product of the covariance and a vector using an iterative procedure which only requires applying the inverse covariance to a vector, what is performed using its implicit representation, Eq. (2.13), as it is done in the map-making solvers. On the algorithmic level such approaches are equivalent to solving the problem in the time domain and the methods considered hereafter would be applicable to that approach as well.

To estimate $\beta$ and $\mathtt{s}$ directly from Eq. (2.9) we may either maximize this likelihood or sample from a posterior derived from it assuming some priors on the spectral parameters[1]. Alternately, one may resort to a so-called spectral likelihood (Stompor et al. 2009), where $\mathtt{s}$ is already either marginalized or maximized over, i.e.,

$$2\ln\mathcal{L}_{spec}(\beta;\widetilde{d}) = \\ = \widetilde{d}^\top\,\widetilde{N}^{-1}\,\widetilde{P}_\beta\,(\widetilde{P}_\beta^\top\,\widetilde{N}^{-1}\,\widetilde{P}_\beta)^{-1}\,\widetilde{P}_\beta^\top\,\widetilde{N}^{-1}\,\widetilde{d} \;+\; \textsc{const}, \tag{2.16}$$

which can be either minimized or sampled from.

In both these cases, a key operation is a solution of a linear set of equations given by,

$$\widetilde{P}_{\beta_i}^\top\widetilde{N}^{-1}\widetilde{P}_{\beta_i}\,\mathtt{s}_{\beta_i} = \widetilde{P}_{\beta_i}^\top\widetilde{N}^{-1}\widetilde{d}, \tag{2.17}$$

for a sequence of tentative values of the spectral parameters, $\beta_i$. Those can either constitute a chain produced as a result of sampling or a sequence of values obtained in the course of a minimization. We note that Eq. (2.17) is essentially a so-called map-making equation (e.g., Natoli et al. 2001; Szydlarski et al. 2014; Puglisi et al. 2018) but with a pointing matrix now replaced by $\widetilde{P}_{\beta_i}$. We can thus hope that, as in the map-making problem, we can capitalize on special structures of the involved matrices and very efficient iterative solvers of linear system to render the problem feasible. We point out that in the applications considered here a subsequent value of the parameter $\beta$, i.e., $\beta_{i+1}$, can be only known once the system for the current value, $\beta_i$, is fully resolved. A simultaneous computation of all systems for all values of $\beta_i$

[1] We note that in sampling from the posterior one would typically also postulate some priors for the signal, what would lead to a different system of equations than the one studied in this work. We leave this case to follow-up work.

is not therefore possible and any computational speedup has to come from using better solvers for the linear systems and/or their numerical implementations.

Separating parts dependent and independent on $\beta$, the system in Eq. (2.17) can also be written as,

$$\begin{bmatrix} M_{\beta,1} \\ \vdots \\ M_{\beta,n_{\rm freq.}} \end{bmatrix}^\top \overbrace{\begin{bmatrix} P_1^\top N_1^{-1} P_1 & & 0 \\ & \ddots & \\ 0 & & P_{n_{\rm freq}}^\top N_{n_{\rm freq}}^{-1} P_{n_{\rm freq}} \end{bmatrix}}^{\equiv \widetilde{A}} \overbrace{\begin{bmatrix} M_{\beta,1} \\ \vdots \\ M_{\beta,n_{\rm freq.}} \end{bmatrix}}^{\equiv \widetilde{M}_\beta} \mathfrak{s}_\beta = \\ = \begin{bmatrix} M_{\beta,1} \\ \vdots \\ M_{\beta,n_{\rm freq.}} \end{bmatrix}^\top \underbrace{\begin{bmatrix} P_1^\top N_1^{-1} d_1 \\ \vdots \\ P_{n_{\rm freq}}^\top N_{n_{\rm freq}}^{-1} d_{n_{\rm freq.}} \end{bmatrix}}_{= \widetilde{P}^\top \widetilde{N}^{-1} \widetilde{d}} \quad (2.18)$$

The approach we propose here is based on two observations. First, our system has some essential similarities to that of the map-making problem, we should be therefore able to capitalize on novel iterative techniques proposed in that case. Second, we expect that consecutive values of $\beta_i$ in realistic sequences should not vary arbitrarily and therefore subsequent linear systems (2.17) should show some resemblance. Consequently, it should be possible to shorten time to solution for the next value of $\beta_{i+1}$ capitalizing on the solution for the current one, $\beta_i$.

### 2.4. Block-diagonal preconditioner

The block-diagonal preconditioner is the most common preconditioner used in the preconditioned conjugate gradient solvers applied in the context of the CMB map-making problem (Natoli et al. 2001), which has demonstrated a very good performance in the number of applications. It is also a basis for a construction of more advanced preconditioners (e.g. Szydlarski et al. 2014). The block-diagonal preconditioner is derived by replacing the noise covariance $\hat{N}_f^{-1}$ in Eq. (2.12) by its diagonal. In the map making case, assuming the pixel-wise ordering, this leads to a block-diagonal matrix with the blocksize defined by the number of the considered Stokes parameters. The generalization of this preconditioner to the component separation case is straightforward and it is given by $\widetilde{P}_\beta^\top \,\mathrm{diag}(\widetilde{N}^{-1})\,\widetilde{P}_\beta$. Again in the pixel-wise ordering the matrix constructed in such a way is block-diagonal with the block size now equal to the product of the number of the Stokes parameters and the number of the sky components, i.e., $6\times6$ in the specific case considered here. Consequently, the preconditioner is easily invertible in any ordering scheme adapted.

Hereafter, we denote the $\beta$-independent part of the preconditioner as $\widetilde{B} := \widetilde{P}^\top \,\mathrm{diag}(\widetilde{N}^{-1})\,\widetilde{P}$ so,

$$\widetilde{P}_\beta^\top \,\mathrm{diag}(\widetilde{N}^{-1})\,\widetilde{P}_\beta = \widetilde{M}_\beta^\top\, \widetilde{B}\, \widetilde{M}_\beta.$$

By preconditioning the system (2.17) from the left, we get

$$(\widetilde{M}_\beta^\top\, \widetilde{B}\widetilde{M}_\beta)^{-1} \widetilde{M}_\beta^\top\, \widetilde{A}\widetilde{M}_\beta\, \mathfrak{s}_\beta = (\widetilde{M}_\beta^\top\, \widetilde{B}\widetilde{M}_\beta)^{-1} \widetilde{M}_\beta^\top\, \widetilde{P}^\top \widetilde{N}\widetilde{d}. \quad (2.19)$$

To simplify the notation in the following, we define,

$$\mathbb{A} := \widetilde{M}_\beta^\top\, \widetilde{A}\, \widetilde{M}_\beta, \qquad \mathbb{B} := \widetilde{M}_\beta^\top\, \widetilde{B}\, \widetilde{M}_\beta, \qquad \mathbb{b} := \widetilde{M}_\beta^\top\, \widetilde{P}^\top \widetilde{N}\, \widetilde{d}. \quad (2.20)$$

### 2.5. Component mixing

For concreteness throughout the paper we assume the following component mixing scheme,

$$\begin{aligned} s_{f,q} &= \alpha_{f,1}\, s_{{\rm cmb},q} + \alpha_{f,2}(\beta_d)\, s_{{\rm dust},q} + \alpha_{f,3}(\beta_s)\, s_{{\rm sync},q}\,, \\ s_{f,u} &= \alpha_{f,1}\, s_{{\rm cmb},u} + \alpha_{f,2}(\beta_d)\, s_{{\rm dust},u} + \alpha_{f,3}(\beta_s)\, s_{{\rm sync},u}\,, \end{aligned} \quad (2.21)$$

which follows the standard assumptions that there is no $Q$ and $U$ mixing and that the scaling laws for the Stokes parameters $Q$ and $U$ of each components are the same. In the component-wise ordering such mixing corresponds to the mixing matrix of the form, ($I$ is the identity matrix, $2\times2$ in this case),

$$M_{\beta,f} = \begin{bmatrix} \alpha_{f,1} I & 0 & \alpha_{f,2}(\beta_d) I & 0 & \alpha_{f,3}(\beta_s) I & 0 \\ 0 & \alpha_{f,1} I & 0 & \alpha_{f,2}(\beta_d) I & 0 & \alpha_{f,3}(\beta_s) I \end{bmatrix}.$$

The coefficients $\alpha_{f,i}$ encode the assumed scaling laws of the CMB, $i = 1$, dust, $i = 2$, and synchrotron, $i = 3$ where the last two depend on unknown scaling parameters, $\beta_d$ and $\beta_s$. This matrix can be rewritten with help of the Kronecker product as,

$$M_{\beta,f} = \begin{bmatrix} \alpha_{f,1} & 0 & \alpha_{f,2}(\beta_d) & 0 & \alpha_{f,3}(\beta_s) & 0 \\ 0 & \alpha_{f,1} & 0 & \alpha_{f,2}(\beta_d) & 0 & \alpha_{f,3}(\beta_s) \end{bmatrix} \otimes I. \quad (2.22)$$

Hereafter, we will drop the explicit dependence of the mixing coefficients on $\beta$ denoting them simply as $\alpha_{f,k}$.

## 3. Solution procedure for parametric component separation problem

A complete solution to the component separation problem will have to address two aspects successfully. First, it will need to propose an efficient approach to solving the sequences of linear systems as in Eq. (2.19) and, second, it will have to combine it with an optimized procedure for an efficient determination of the new values of the parameters $\beta$. This study aims at addressing the former problem and focuses on the solution of a sequence of the linear systems obtained for some sequences of the spectral parameters. In order to provide a fair comparison of various solution techniques for the linear systems, we generate a sequence $\{\beta_i\}$ a priori, i.e., in our experiments $\beta_{i+1}$ is in fact independent of the computed approximation $\mathfrak{s}_{\beta_i}^{(final)}$, what ensures that performance evaluation of all the considered solvers is based on solving identical sequence of linear systems.
The overall solution scheme we adapt here is then as follows,

0) Initialize $\beta_0$ and $\mathfrak{s}_{\beta_0}^{(0)}$ (typically $\mathfrak{s}_{\beta_0}^{(0)} := 0$), set $i := 0$.
1) Given $\beta_i$ and the initial guess $\mathfrak{s}_{\beta_i}^{(0)}$, solve the preconditioned problem (2.19) for the approximation $\mathfrak{s}_{\beta_i}^{(final)}$.
2a) Determine the new parameters $\beta_{i+1}$; in practice from $\mathfrak{s}_{\beta_i}^{(final)}$, in our experiments $\beta_{i+1}$ is given a priori.
2b) Compute a new deflation space for the system associated with $\beta_{i+1}$ using a recycling technique (see details below). This should not involve the value of $\beta_{i+1}$ so that this step can be done in parallel with **2a)**.
3) Compute the initial guess $\mathfrak{s}_{\beta_{i+1}}^{(0)}$.
4) Set $i := i + 1$ and go to **1)**.

In the subsections below, we discuss the steps **1)**, **2b)**, and **3)** in more details.

### 3.1. PCG with deflation and two-level preconditioners

Though the block-diagonal preconditioner has been shown to ensure good performance in the map-making experience, it has been pointed out that even better performance can be often achieved by employing so-called two-level preconditioners (Szydlarski et al. 2014; Puglisi et al. 2018). Such preconditioners are built from the block-diagonal preconditioner (referred to as the first level) and the second level based on several vectors that are *deflated* (i.e., suppressed in the operator) in order to accelerate the convergence. These vectors are typically taken to be approximate eigenvectors of the system matrix corresponding to its smallest eigenvalues, which often hamper the convergence of PCG with the block-diagonal preconditioner.

We start from the case of *deflation* for (unpreconditioned) conjugate gradient (CG) method. CG applied to a linear system $\mathbb{A}\mathfrak{s} = \mathbb{b}$ with a given initial vector $\mathfrak{s}^{(0)}$ and an initial residual $\mathfrak{r}^{(0)} := \mathbb{b} - \mathbb{A}\mathfrak{s}^{(0)}$, builds implicitly an orthogonal (residuals) and an $\mathbb{A}$-orthogonal (search directions) basis of the Krylov subspace,

$$\mathcal{K}_j(\mathbb{A}, \mathfrak{r}^{(0)}) = \mathrm{span}\{\mathfrak{r}^{(0)}, \mathbb{A}\mathfrak{r}^{(0)}, \mathbb{A}^2\mathfrak{r}^{(0)}, \ldots, \mathbb{A}^{j-1}\mathfrak{r}^{(0)}\},$$

and the $j$th CG approximation $\mathfrak{s}^{(j)} \in \mathfrak{s}^{(0)} + \mathcal{K}_j(\mathbb{A}, \mathfrak{r}^{(0)})$ is determined by the orthogonality condition on the $j$th residual $\mathfrak{r}^{(j)} := \mathbb{b} - \mathbb{A}\mathfrak{r}^{(j)}$,

$$\mathfrak{r}^{(j)} \perp \mathcal{K}_j(\mathbb{A}, \mathfrak{r}^{(0)}).$$

Given a set of deflation vectors, i.e., the vectors to be suppressed, denote by $\mathcal{U}$ the subspace spanned by these vectors. The deflation techniques replace the original operator $\mathbb{A} : \mathbb{R}^n \to \mathbb{R}^n$ by a deflated operator $\widehat{\mathbb{A}} : (\mathbb{R}^n \setminus \mathcal{U}) \to (\mathbb{R}^n \setminus \mathcal{U})$. The approximation is then sought over the augmented subspace (see, e.g., Gaul et al. (2013))

$$\mathfrak{s}^{(j)} \in \widehat{\mathfrak{s}}_0 + \mathcal{K}_j(\widehat{\mathbb{A}}, \widehat{\mathfrak{r}}^{(0)}) \cup \mathcal{U}$$

and the $j$th residual is required to be orthogonal to $\mathcal{K}_j(\widehat{\mathbb{A}}, \widehat{\mathfrak{r}}^{(0)}) \cup \mathcal{U}$. This effectively prevents the solver from exploring the subspace $\mathcal{U}$.

Extension of the above discussion for PCG with a (first-level) preconditioner $\mathbb{B}$ is straightforward as we can use PCG to build implicitly the Krylov subspace $\mathcal{K}_j(\mathbb{B}^{-1}\mathbb{A}, \mathbb{B}^{-1}\mathfrak{r}^{(0)})$. In the considered application, the preconditioner $\mathbb{B}$ is the block diagonal preconditioner. There are many variants of two-level preconditioners and we summarize them briefly in Appendix B.2, a more thorough survey can be found, e.g., in Tang et al. (2009).

Each iteration of a deflated (P)CG, i.e., with or without the first level, is more costly than a single iteration of standard (P)CG. The additional cost depends, primarily, on the number of deflated vectors, i.e., the dimension of $\mathcal{U}$, but also on a deflation variant. Building the subspace $\mathcal{U}$ typically requires some preliminary computations, which can be as costly as solving the system (see e.g., Szydlarski et al. 2014; Puglisi et al. 2018). Another approach, applicable to the cases when multiple systems need to be solved, is to construct the vectors "on the fly" during the solution of the systems themselves thus hiding the additional cost. This is the approach we detail in the follow-up sections.

### 3.2. Subspace recycling

There already exist several constructions of the deflation space adapted to solving a sequence of linear systems such as Saad et al. (2000); Parks et al. (2006); Kilmer & de Sturler (2006); O'Connell et al. (2017); Jolivet & Tournier (2016). In this work, where the system matrix is SPD, we follow Saad et al. (2000) and build a subspace $\mathcal{Z} \subset \mathcal{K}_j(\widehat{\mathbb{A}}, \widehat{\mathfrak{r}}^{(0)})$ by storing some of the vectors computed during the previous run of (P)CG solver and determine the slowest eigenvectors of the operator $\mathbb{A}$ restricted on the subspace $\mathcal{U} \cup \mathcal{Z}$. These are taken as the deflation vectors for the next solution. The resulting algorithm is given in Appendix C.

We can determine the subspace $\mathcal{Z}$ by using either the residual or the search direction vectors forming (assuming the exact arithmetic) the orthogonal or, respectively, an $\widehat{\mathbb{A}}$-orthogonal basis of $\mathcal{K}_j(\widehat{\mathbb{A}}, \widehat{\mathfrak{r}}^{(0)})$. Following Saad et al. (2000), we choose here to use the search direction vectors. We retain the first $\dim_p$ search direction vectors, where $\dim_p$ defines the dimension of the so-called recycle subspace. Using the first vectors is motivated by the fact that the orthogonality among the computed vectors is gradually lost in CG; it is therefore preserved better in the initial iterations.

The techniques for approximating $k$ eigenvectors of the operator over a given subspace are well-established. Among them, we note the Ritz and harmonic Ritz projections, which are described in detail in Appendix B.1.1. They lead to solution of (generalized) eigenvalue problem of a small dimension – in our case of $\dim(\mathcal{U}) + \dim(\mathcal{Z})$. While Saad et al. (2000) suggest using the harmonic Ritz projection, we found Ritz projection slightly more efficient in our numerical experiments and we therefore include this in the full algorithm described in Appendix C. In another difference with Saad et al. (2000) we assemble the (small) generalized eigenvalue problem matrices in the harmonic Ritz projection using the matrix-matrix products (see Algorithm 2 in Appendix B.1.1) instead of the optimized algorithm of Saad et al. (2000) (Section 5.1). This is because we expect that the extra computational cost in our application is negligible and we therefore have opted for the simplicity.

There is no general recommendation for the choice of the number of the deflation vectors, $k$, and the dimension of the recycling subspace, $\dim_p$. Higher $k$ may result in an increase of the overall number of matrix-vector products (one has to apply the system matrix to $k$ deflation vectors before starting deflated (P)CG for each system) and high $\dim_p$ may cause numerical instabilities in solving the eigenvalue problems determining the new deflation vectors. On the other hand, the small values of $k$ and $\dim_p$ may not bring enough speed up. We test this numerically in Section 4.

### 3.3. Impact of the eigenvalue multiplicity.

One limiting factor to the efficiency of this approach, and more generally to performance of any two-level preconditioner whenever the deflation space is estimated with help of standard iterative techniques, such as Arnoldi or Lanczos iterations, comes from a higher multiplicity of eigenvalues, i.e., presence of multiple eigenvectors with the same corresponding eigenvalue. This can arise either due to some symmetries in the scanning strategies, in the case of the map-making systems of equations, and similarities in the noise covariances of the different single frequency maps, in the case of the component separation problem as studied here, see Appendix A for a worked example. Admittedly, such symmetries and/or similarities are not typically expected in the cases of real data analysis, however they can arise in the cases of simulated data in particular if simplifying assumptions are adopted in order to speed and/or simplify the simulation process.

To shed a light on this problem we consider an SPD matrix $A$ and assume that $\lambda$ is an eigenvalue with multiplicity higher than one. This means that there exists a subspace $V$ with $\dim(V) > 1$ such that

$$Av = \lambda v, \qquad \forall v \in V.$$

Let $w$ be an arbitrary vector and $w_V$ its projection onto $V$, i.e.

$$w = w_V + w', \quad w_V \in V,\ w' \cap V = \emptyset.$$

This decomposition is unique for any normal matrix $A$ (in particular, an SPD matrix is normal). Then

$$A^\ell w = A^\ell w_V + A^\ell w' = \lambda^\ell w_V + A^\ell w', \qquad A^\ell w' \cap V = \emptyset.$$

Therefore, the $j$th Krylov subspace satisfies

$$\begin{aligned}\mathcal{K}_j(A, w) &= \mathrm{span}\{w, Aw, A^2w, \ldots, A^{j-1}w\} \\ &= \mathrm{span}\{w_V\} \cup \mathrm{span}\{w', Aw', A^2w', \ldots, A^{j-1}w'\}\end{aligned}$$

and

$$\mathcal{K}_j(A, w) \cap V = \mathrm{span}\{w_V\}.$$

Consequently, whenever we use Lanczos (or Arnoldi) method, which is based on Krylov subspace approximation (see Appendix B.1.2 for more details), for approximating the eigenpairs, we cannot recover neither multiplicity of the eigenvalues (without using some additional techniques) nor the full subspace associated to an eigenvalue with higher multiplicity which will always be approximated by a single vector. If the corresponding eigenvalue happens to be small, this approximation may not be sufficient hampering the performance of the preconditioner constructed with help of the combination of Lanczos/Arnoldi method.

This issue can be overcome by using a more advanced eigenvalue solver, which can detect and handle the higher multiplicity of eigenvalues. An efficient implementation of such a solver is for instance provided by the ARPACK library (Lehoucq et al. 1998). In this case one may need to resort to a precomputation of the preconditioner with help of such advanced routines instead of the constructing it on the fly as proposed here. If the presence of the eigenvalue multiplicity and the corresponding eigenvectors is known ahead of time, one can accommodate these vectors in the on-the-fly procedure proposed here. This is indeed the case we have encountered in one of the test cases discussed later in this work.

We point out that the multiplicity of the eigenvalues is in principle advantageous for the standard (P)CG. In exact arithmetic, the effect of the whole subspace could be eliminated at the cost of a single iteration. This fact is often used in the analysis of preconditioning methods where proper preconditioners shift some, possibly many, eigenvalues to the same value.

Last but not least, we emphasize that an eigenvalue multiplicity does not imply any indistinguishability or degeneracy of the corresponding eigenmodes, unless the corresponding, common eigenvalue happens to be (numerically) zero. If this is not the case, the different eigenmodes, as present in the right hand side of the linear system, will be merely weighted by the inverse system matrix in the same way, even though their contributions to the right hand side may be very different start with.

### 3.4. Choice of the initial guess

The simplest and the standard way to solve the sequence is to run the PCG method with the initial guess set to zero. However, some evidence exists showing that at least in the map-making case this may not always be the best choice (Papež et al. 2018) in particular in the high signal-to-noise cases. In the cases of the sequences all the systems involve the same initial data set with the same signal and noise content so even in the low signal-to-noise data one may expect that adapting the initial guess following previous results may bring some interesting speed-up. Consequently, we explore here two alternative choices and show via numerical experiments that they are indeed much more efficient.

#### 3.4.1. Previous solution as the initial guess

A natural idea is to run PCG for the (new) problem corresponding to $\beta_{i+1}$ starting with the computed approximation $\mathbb{s}_{\beta_i}^{(final)}$,

$$\mathbb{s}_{\beta_{i+1}}^{(0)} := \mathbb{s}_{\beta_i}^{(final)}. \tag{3.1}$$

This can be in particular efficient if the parameters $\beta_i$ and $\beta_{i+1}$ do not significantly differ and one can expect that so do $\mathbb{s}_{\beta_i}$ and $\mathbb{s}_{\beta_{i+1}}$.

#### 3.4.2. Adapted previous solution as the new initial guess

Eq. (3.1) can be further adapted capitalizing on the particular structure of the mixing matrix. To start with we rewrite Eq. (2.17) as,

$$\widetilde{M}_\beta^\top (\widetilde{A}\widetilde{M}_\beta\, \mathbb{s}_\beta - \widetilde{P}^\top \widetilde{N}^{-1}\widetilde{d}) = 0. \tag{3.2}$$

If the matrix $\widetilde{M}_\beta$ were square (and non-singular), then

$$\widetilde{M}_\beta\, \mathbb{s}_\beta = \widetilde{A}^{-1}\widetilde{P}^\top \widetilde{N}^{-1}\widetilde{d}$$

would be the vector independent of $\beta$. The solution $\mathbb{s}_\beta$ could then be interpreted as the coefficients w.r.t. the basis given by the columns of $\widetilde{M}_\beta$. Therefore we would have,

$$\mathbb{s}_{\bar\beta} = (\widetilde{M}_{\bar\beta})^{-1}\widetilde{M}_\beta\, \mathbb{s}_\beta$$

for arbitrary $\bar\beta$ (for which $\widetilde{M}_{\bar\beta}$ is non-singular).

In our case, matrix $\widetilde{M}_\beta$ is rectangular of size $2\,n_{\mathrm{freq}}\,n_{\mathrm{pix}} \times 6\,n_{\mathrm{pix}}$ and has full column rank. When the number of frequencies $n_{\mathrm{freq}}$ is not significantly higher than 3, we can then generalize the above idea and use as the initial guess for the new system the vector,

$$\mathbb{s}_{\beta_{i+1}}^{(0)} := (\widetilde{M}_{\beta_{i+1}})^\dagger \widetilde{M}_{\beta_i}\, \mathbb{s}_{\beta_i}^{(final)}, \tag{3.3}$$

where $M^\dagger$ is the (Moore–Penrose) pseudoinverse[2] of $M$,

$$M^\dagger \equiv (M^\top M)^{-1} M^\top.$$

Clearly, such vector satisfies a natural requirement: for $\beta_{i+1} = \beta_i$ there holds,

$$(\widetilde{M}_{\beta_{i+1}})^\dagger \widetilde{M}_{\beta_i} = I \quad \text{and therefore} \quad (\widetilde{M}_{\beta_{i+1}})^\dagger \widetilde{M}_{\beta_i}\, \mathbb{s}_{\beta_i} = \mathbb{s}_{\beta_i}.$$

Finally, we note that the computation of the vector in (3.3) is very cheap due to the Kronecker structure (2.22) of the matrices $\widetilde{M}_\beta$. Writing $\widetilde{M}_\beta = K_\beta \otimes I$, we obtain,

$$(\widetilde{M}_{\beta_{i+1}})^\dagger \widetilde{M}_{\beta_i} = \left((K_{\beta_{i+1}}^\top K_{\beta_{i+1}})^{-1} K_{\beta_{i+1}}^\top K_{\beta_i}\right) \otimes I,$$

in other words, only the matrices of size $2\,n_{\mathrm{freq}} \times 6$ need to be handled and the cost of the proposed adaptation is nearly negligible.

[2] We recall our assumption that $M$ is of full column rank.

## 4. Numerical experiments

### 4.1. Simulated data

For our numerical tests we use a simulated data set composed of time-ordered, multi-frequency observations with a correlated, '1/f', noise. The characteristics of this data set are so follows.

#### 4.1.1. Pointing matrix

We adopt a simple scanning strategy used in Papež et al. (2018). The entire time-ordered data set is composed of $O(10^8)$ measurements per frequency and divided into four, consecutive subsets. The pointing is assumed to be the same for each frequency. The underlying sky is pixelized using Healpix pixelization scheme (Górski et al. 2005) with the resolution parameter, $n_{\rm side}$ set to 1024. The scan consists of repetitive scanning of a rectangular patch made of 256 pixel rows and columns. The scanning is either horizontal, i.e., along the pixel rows, for the first and third subset, or vertical, i.e., along the pixel columns for the second and fourth subset. During a single left-to-right, or bottom-up sweep, each sky pixel is sampled only once and the direction of the polarizer, $\varphi_t$, is fixed for each of the four subsets and equal, with respect to the sky, to 0, $\pi/4$, $\pi/2$, and, $3\pi/4$.

The sky signal contribution to every measurement is modeled as,

$$d_c(t) \;=\; Q_c^\star(p(t))\cos 2\varphi_t \;+\; U_c^\star(p(t))\,\sin 2\varphi_t, \tag{4.1}$$

where $p(t)$ denotes the pixel observed at time $t$, we do not include the total intensity, and $Q_c$ and $U_c$ stand for $Q$ and $U$ Stokes parameters of the combined, CMB + foregrounds, sky signal observed at frequency $\nu_c$.

#### 4.1.2. Sky maps

We assume 6 frequency channels roughly corresponding to those accessible for a ground based experiment. These are,

$$\nu_c \;\in\; \Big\{30, 40, 90, 150, 220, 270\Big\}\ \text{GHz}. \tag{4.2}$$

The sky signal is composed of three components, CMB, dust, and synchrotron. The CMB signal is simulated using the current best-fit CMB model (Planck Collaboration et al. 2016b), while we use the so-called COMMANDER templates (Planck Collaboration et al. 2016a) to model the dust and synchrotron signals, which we scale to our reference frequency, $\nu_{\rm ref}$ = 150GHz, using Planck's fiducial laws.

For the scaling laws we take a black-body for the CMB component ($T_{\rm CMB}$ = 2.7525K), a power-law for the synchrotron, and a modified black-body for the dust, hence,

$$\mathcal{S}^{\rm sync}(\nu,\beta_s^\star) \;=\; \nu^{\beta_s^\star} \tag{4.3}$$

$$\mathcal{S}^{\rm dust}(\nu,\beta_d^\star,T_d^\star) \;=\; \left(\frac{h\nu}{kT_d^\star}\right)^{\beta_d^\star} B(\nu,T_d^\star), \tag{4.4}$$

where the star distinguishes the true values of the parameters, which are,

$$\beta^\star \;\equiv\; [\beta_s^\star,\beta_d^\star,T_d^\star] \;=\; [-3.1, 1.59, 19.6\,\text{K}], \tag{4.5}$$

and $B(\nu,T)$ denotes a black-body at temperature, $T$. The simulated maps are expressed in the thermodynamic units and given by,

$$Q_p^\star(\nu) = Q_p^{\rm cmb,\star} + \Gamma_{\rm RJ}(\nu)\Big[\frac{\mathcal{S}^{\rm dust}(\nu,\beta_d^\star,T_d^\star)}{\mathcal{S}^{\rm dust}(\nu_{\rm ref},\beta_d^\star,T_d^\star)}\,Q_p^{\rm dust,\star}(\nu_{\rm ref}) + \frac{\mathcal{S}^{\rm sync}(\nu,\beta_s^\star)}{\mathcal{S}^{\rm sync}(\nu_{\rm ref})}\,Q_p^{\rm sync,\star}(\nu_{\rm ref},\beta_s^\star)\Big],$$

for each frequency $\nu = \nu_c$ and each observed sky pixel, $p$. An analogous formula holds for the Stokes $U$ parameter. Here, $\Gamma_{\rm RJ}(\nu)$ stands for a conversion factor from the Rayleigh-Jeans to thermodynamic units. This expression is consistent with Eq. (2.22) upon a suitable definition of the coefficients $\alpha$.

In our numerical experiments, we fix the dust temperature, $T_d$, to its true value and assume that only the spectral indices, $\beta = [\beta_s,\beta_d]$, are determined from the data. We assume that these are estimated via a maximization of the spectral likelihood, Eq. (2.16), using a truncated Newton maximization procedure. We use this approach to generate a single sequence of $\{\beta_i\}$, which, as explained in Sect. 3, we adopt consistently in all our runs. The sequence is made of 26 values and is shown in Fig. 1. In appendix D we show for completeness results of the similar test but performed for a sequence of $\beta$ derived via sampling of the spectral likelihood. The main conclusions derived in both these examples are consistent.

#### 4.1.3. Noise

We assume a correlated noise in the time domain with a spectrum given by

$$P(f) \;=\; \sigma^2_{rms}\,(1+\frac{f_{\rm knee}}{f}), \tag{4.6}$$

where $f$ is the time-domain frequency. The values of $f_{\rm knee}$ adopted here are different for different frequency channels and taken to be such that there are strong noise correlations within a single sweep. They span the range from 0.5 up to 3Hz from the lowest to the highest frequency channel, respectively. The noise is apodized at very low frequencies so the noise power is finite. $\sigma^2_{\rm rms}$ is taken to be around $30\mu$K per sample reflecting the fact that each measurements effectively corresponds to the combined measurement of many modern detectors operating at the same frequency. This leads to sky maps with the noise, $\sigma^{Q/U}_{\rm rms} \sim 2{-}3\mu$K per pixel.

### 4.2. Multiplicity of the eigenvalues due to the particular scanning strategy

We denote two pointing matrices for two horizontal scans as $P_0$ and $P_{\pi/2}$ and two pointing matrices for two vertical scans as $P_{\pi/4}$ and $P_{3\pi/4}$, where the subscript stands for the polarizer angle in the sky coordinates. Those are related as follows,

$$P_{\pi/2} \;=\; P_0\,\mathcal{R}_2(\pi/4),$$
$$P_{3\pi/4} \;=\; P_{\pi/4}\,\mathcal{R}_2(\pi/4),$$

where $\mathcal{R}$ is a $12\,n_{\rm pix}$-by-$12\,n_{\rm pix}$ block diagonal matrix with each diagonal block equal to a 2-by-2, spin-2 rotation matrix for each pixel of the 6 frequency maps. While this is clearly a result of the simplifying assumption made in the simulations, this example may be of interest also in more realistic circumstances where certain relations of this sort may happen to be fulfilled approximately following from some common, albeit inexact, symmetries of typically adopted scans. We will therefore explore here briefly consequences of this fact.

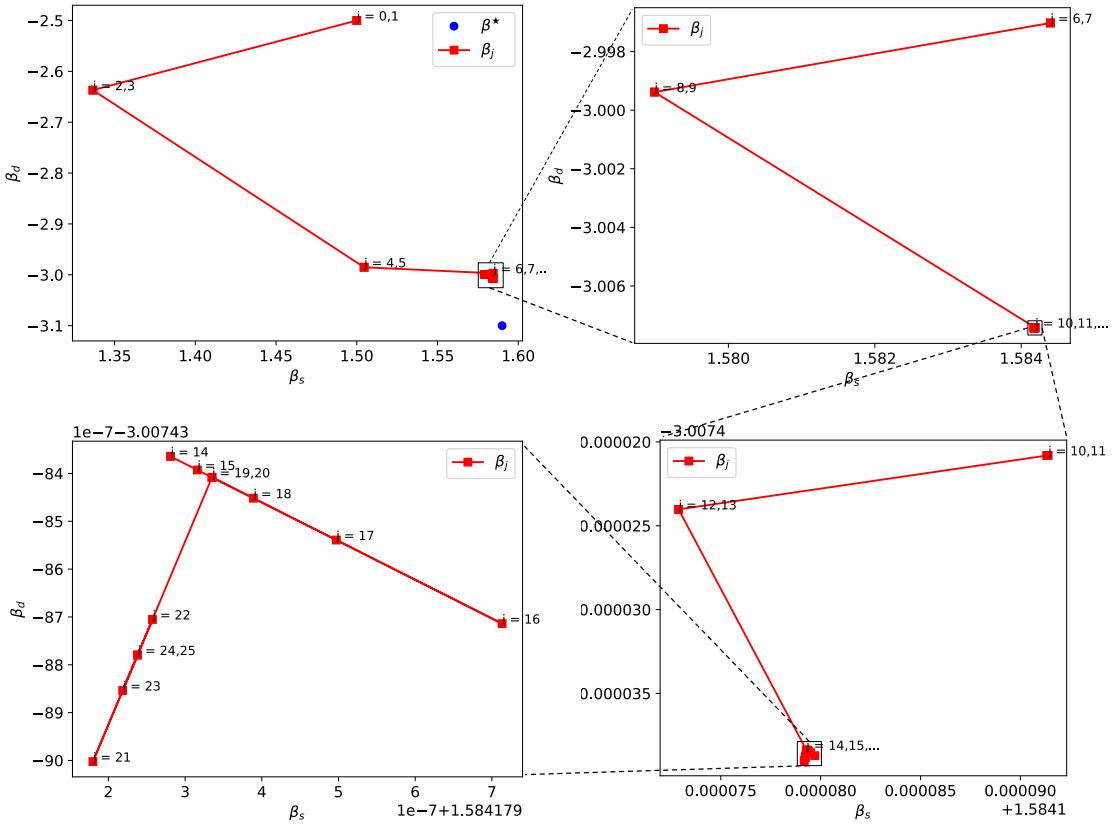


**Fig. 1.** The sequence of the spectral parameters parameters $\beta_i = [\beta_{i,s}, \beta_{i,d}]$ used in our experiments and derived from the maximization of the spectral likelihood, Eq. (2.16). The panels in the clock-wise order show consecutive zooms on the sequence which converged on the likelihood peak values of $\beta = [-3.006, 1.584]$ slightly off the true values marked by the blue solid point in the top left panel and given by Eq. (4.5) (with $T_d$ fixed in our test cases). The sequence was generated as described at the end of Sect. 4.1.2.

In the case at hand we can represent the total pointing matrix as

$$\widetilde{P} = \begin{bmatrix} P_0 \\ P_{\pi/4} \\ P_{\pi/2} \\ P_{3\pi/4} \end{bmatrix} = \begin{bmatrix} P_{-\pi/4} \\ P_0 \\ P_{\pi/4} \\ P_{\pi/2} \end{bmatrix} \mathcal{R}_{\pi/4} = \begin{bmatrix} P_{3\pi/4} \\ P_0 \\ P_{\pi/4} \\ P_{\pi/2} \end{bmatrix} \mathcal{R}_{\pi/4} = \widetilde{P}' \, \mathcal{R}_{\pi/4},$$

given that all four scan subsets observe exactly the same sky.

In the case when moreover the noise covariance for each of the four scan subsets is exactly the same, then

$$\widetilde{P}^\top \widetilde{N}^{-1} \widetilde{P} = \widetilde{P}'^\top \widetilde{N}^{-1} \widetilde{P}' = \mathcal{R}_{\pi/4}^\top \widetilde{P}^\top \widetilde{N}^{-1} \widetilde{P} \mathcal{R}_{\pi/4}.$$

We note that this holds if the noise properties vary from one frequency channel to another as it is indeed the case in our simulations.

If now $v$ is an eigenvector of the matrix $\widetilde{A} = \widetilde{P}^\top \widetilde{N}^{-1} \widetilde{P}$ with a corresponding eigenvalue, $\lambda_v$, then

$$\widetilde{A} v = \widetilde{P}^\top \widetilde{N}^{-1} \widetilde{P} v = \mathcal{R}_{\pi/4}^\top \widetilde{P}^\top \widetilde{N}^{-1} \widetilde{P} \mathcal{R}_{\pi/4} v = \lambda_v v, \qquad (4.7)$$

and therefore,

$$\widetilde{A} \mathcal{R}_{\pi/4} v = \widetilde{P}^\top \widetilde{N}^{-1} \widetilde{P} \mathcal{R}_{\pi/4} v = \lambda_v \mathcal{R}_{\pi/4} v,$$

meaning that also $\mathcal{R}_{\pi/4} v$ is an eigenvector of the matrix $A$ with the same eigenvalue, $\lambda_v$. As this reasoning applies as much to the matrix $A$ as the matrix $B = \widetilde{P}^\top \mathrm{diag}\, \widetilde{N}^{-1} \widetilde{P}$, we have

$$\begin{aligned} \widetilde{P}^\top \widetilde{N}^{-1} \widetilde{P} v &= \lambda'_v \widetilde{P}^\top \mathrm{diag}\, \widetilde{N}^{-1} \widetilde{P} v \\ \widetilde{P}^\top \widetilde{N}^{-1} \widetilde{P} (\mathcal{R}_{\pi/4} v) &= \lambda'_v \widetilde{P}^\top \mathrm{diag}\, \widetilde{N}^{-1} \widetilde{P} (\mathcal{R}_{\pi/4} v). \end{aligned}$$

Whether this implies that the component separation system matrix preconditioned with the block-diagonal preconditioner, Eq. (2.20), which is given by,

$$\mathbb{B}^{-1} \mathbb{A} = (\widetilde{M}_\beta^\top \widetilde{B} \widetilde{M}_\beta)^{-1} (\widetilde{M}_\beta^\top \widetilde{A} \widetilde{M}_\beta),$$

has two eigenvectors corresponding to the same eigenvalue related by the rotation operator acting in the component space, will depend whether the subspace spanned by $v$ and $\mathcal{R}_{\pi/4} v$ is contained in the subspace spanned by the columns of the mixing matrix, $\widetilde{M}_\beta$. Otherwise, it may have a single (in the case of the intersecting subspaces) or no eigenvectors (in the case of the disjoint subspaces) with the corresponding eigenvalue. Which situation is actually realized in practice will be case-dependent and will in general depend on the value of $\beta$.

We found that the former case is common enough that in our numerical experiments we have opted to account explicitly on the possibility of having duplicity of the eigenvectors due to the scanning. Consequently, while we use the subspace recycling to approximate only one eigenvector we compute the other one by applying the rotation operator. We then use both vectors to construct the deflation operator. Given that $\mathcal{R}_{\pi/4} = -\mathcal{R}_{-\pi/4}$, there is no ambiguity due to the fact that *a priori* we do not know which of the vectors we estimate via the subspace recycling and this approach leads to the same deflation space, whichever rotation is applied. This technique has led to an important speed-up in the cases studied hereafter.

### 4.3. Results

We compare the convergence using the relative norm of the $j$th residual

$$\frac{\|\widetilde{M}_\beta^\top \widetilde{P}^\top \widetilde{N}\widetilde{d} - \widetilde{M}_\beta^\top \widetilde{A}\widetilde{M}_\beta \mathbb{s}_\beta^{(j)}\|}{\|\widetilde{M}_\beta^\top \widetilde{P}^\top \widetilde{N}\widetilde{d}\|}.$$

The iterations for each system are stopped once this value drops below tolerance TOL $= 10^{-8}$, but we always perform at least one iteration.

We first show that the systems corresponding to different $\beta$s from the sequence are indeed "close to each other" in a sense that they display a similar behavior during the solution process. We illustrate this by showing the convergence for PCG with zero initial guess in Figure 2. We find that all the convergence curves are indeed very similar even for the initial elements of the sequence, where the values of $\beta$ parameters keep on changing quite significantly.

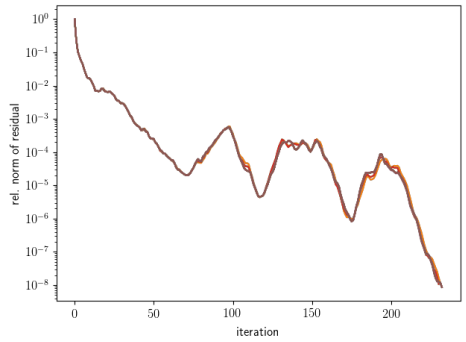


**Fig. 2.** Convergence of the PCG with zero initial guess and the block-Jacobi preconditioner for all 26 systems in the sequence shown in Fig. 1.

We can therefore focus on the "true" system corresponding to $\beta = \beta^\star$ in order to investigate improvement due to the deflation of the eigenvectors corresponding to the smallest eigenvalues of the system matrix. This is depicted in Figure 3. Here, the eigenvectors are computed using the ARPACK eigensolver (Lehoucq et al. 1998) and, as expected, we find a significant improvement thanks to the deflation.

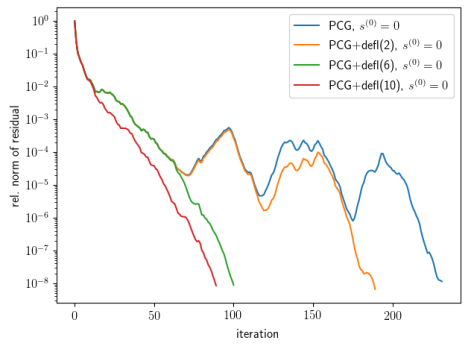


**Fig. 3.** Convergence of the PCG with the deflation applied to 2, 6, and 10 slowest eigenvectors for the true system with $\beta = \beta^\star$.

Then, we illustrate the effect of the deflation space built by recycling. For that, we consider first six systems and we start the iterations always with zero initial guess, $\mathbb{s}^{(0)} = 0$. The result for $k = 10$ eigenvectors approximated using the dimension of the subspace, $\dim_p = 100$, is given in Figure 4.

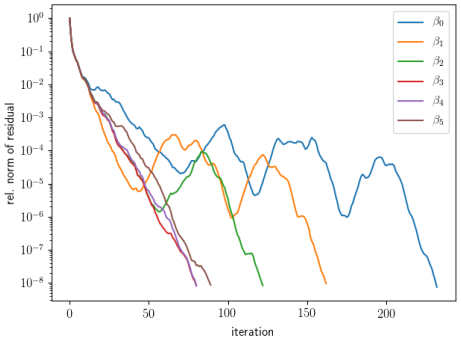


**Fig. 4.** Convergence of deflated PCG with the deflation subspace build by recycling. Here we consider the first six systems from the sequence and we start the iterations always with zero initial guess. $k = 10$ eigenvectors are approximated using $\dim_p = 100$.

In Figure 5 we compare the convergence for the full sequence of the 26 systems using the techniques for setting the initial guess proposed in Eq. (3.1) and Eq. (3.3) and using the subspace recycling. We recall that standard PCG with zero initial vector takes more than 6000 iterations; cf. Figure 2. Despite the fact that the subspace recycling variant requires additional $25 \times 10$ matrix-vector products[3] as compared to the PCG with the block-Jacobi preconditioner for any choice of the initial guess, it still provides an interesting speed up.

As noted above, the performance of the PCG with the deflation space built by recycling is affected by the number of the deflation vectors, $k$, and the dimension of the recycling subspace, $\dim_p$. There is no general recommendation for their choice. Higher $k$ may result to increase of the overall number of matrix-vector products (one has to apply the system matrix to $k$ deflation vectors before starting deflated PCG for each system) and high $\dim_p$ may cause numerical instabilities in solving the eigenvalue problems determining the new deflation vectors. On the other hand, the small values of $k$ and $\dim_p$ may not bring enough speed up. We compare the convergence of PCG with some choices of $k$ and $\dim_p$ in Figure 6 and in Table 1. One can see that the deflation has a small effect for small $\dim_p$, i.e., when the eigenvectors are not approximated accurately.

| $\dim_p$ | $k$ | #iterations | #MatVecs | |
|---|---|---|---|---|
| | | | deflation | total |
| 20 | 6 | 933 | 54 | 987 |
| 50 | 6 | 783 | 54 | 837 |
| 100 | 6 | 708 | 54 | 762 |
| 20 | 10 | 867 | 90 | 957 |
| 50 | 10 | 775 | 90 | 865 |
| 100 | 10 | 612 | 90 | 702 |

**Table 1.** The number of PCG iterations and matrix-vector products (MatVecs) for different choices of $k$ and $\dim_p$ for the first 10 systems in the sequence. The initial guess is as in Eq. (3.3).

## 5. Conclusions and further perspectives

We have presented a procedure for an effective solution of sequences of the linear systems arising in CMB parametric com-

[3] Applying the matrix to deflation vectors is necessary at the beginning of the deflated PCG to build the projection matrices, see Algorithm 1 in Appendix C

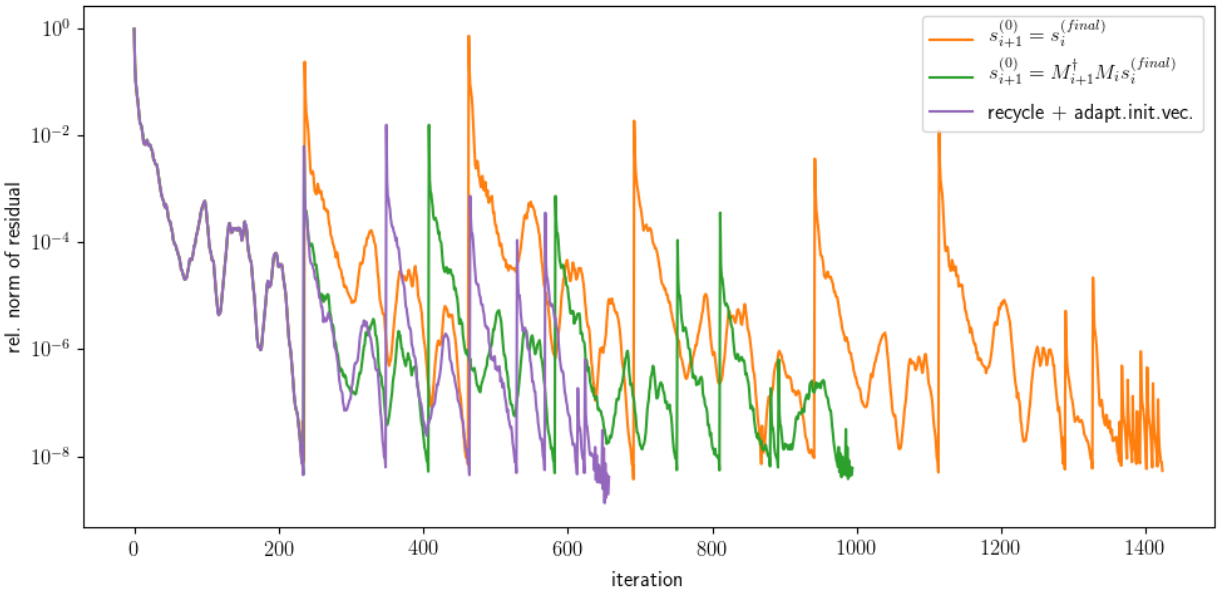


**Fig. 5.** Comparison of PCG with different choices of initial guess (as in Eq. (3.1) and Eq. (3.3)) and the PCG with the subspace recycling (together with the choice of the initial guess as in Eq. (3.3)). For the recycling we consider $k = 10$ eigenvectors approximated using $\dim_p = 100$.

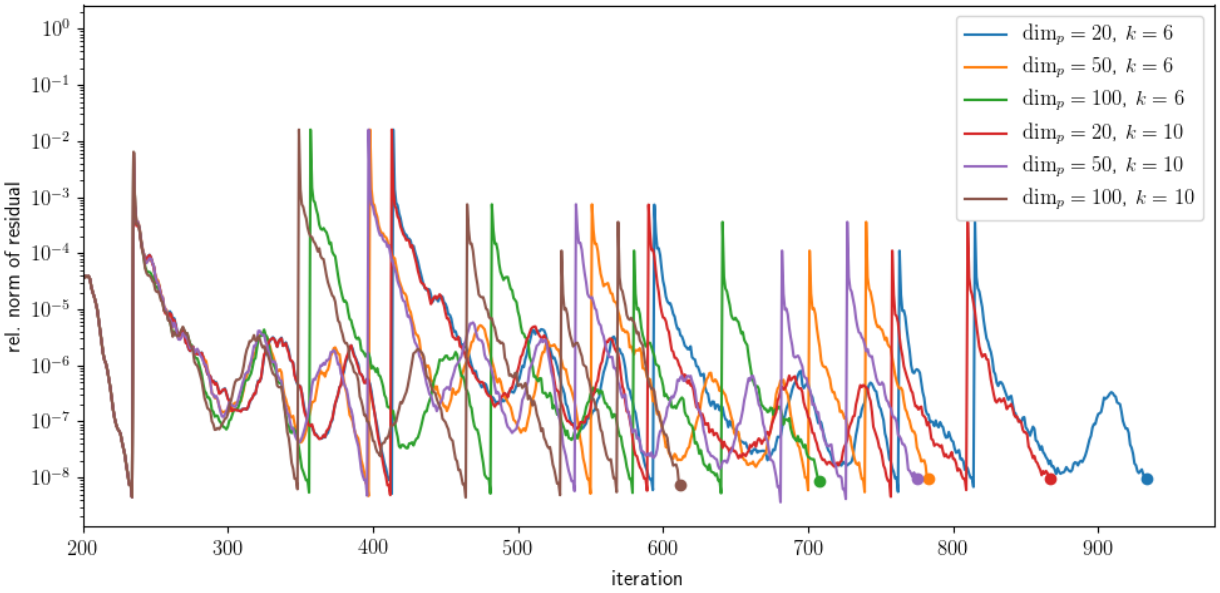


**Fig. 6.** Comparison of PCG with different choices of $k$ and $\dim_p$ for the first 10 systems in the sequence. The initial guess is as in Eq. (3.3). The iteration counts are given also in Table 1. Since the convergence for the first system is the same for arbitrary $\dim_p$ and $k$, the $x$-axis is depicted starting from the iteration 200.

ponent separation problem. Two main novelties are in the proposed choice of the initial vectors and the recycling technique used to determine the deflation space. These techniques can be also used in other problems in CMB data analysis which lead to a sequence or a set of linear systems. Motivated by our simulated data, we have also emphasized and elaborated on the role of the multiplicity of the eigenvalues, in particular in the context of their impact on performance of two-level preconditioners.

The overall speed-ups we have recovered, $\sim$ 5–7, are quite impressive. While a bulk of the improvement comes from reusing the solution of an earlier system as the initial guess of the follow-up one, a simple but not trivial observation owing to the fact that this is the same data set which is being used in all the solutions, other proposed amendments provide important additional performance boost on order of $\sim$ 2, particularly significant in the case of the sampling-based application. Further improvements and optimizations are clearly possible. For instance, the number of required matrix-vector products can be decreased by not using the two-level preconditioner for all the systems as the experiments showed that when two consecutive system solutions are very close to each other, the PCG with the diagonal preconditioner and a proper initial guess (e.g. as proposed in Section 3.4.2) can be already sufficient converging in few iterations.

We emphasize that in practice we will be only able to capitalize on this kind of approaches if they are implemented in a form of highly efficient, high performance massively parallel numerical algorithms and codes. We leave a full demonstration of this to future work, noting only here that many of the relevant techniques have been already studied in the past and recent literature showing that this indeed should be plausible (e.g., Cantalupo et al. 2010; Sudarsan et al. 2011; Szydlarski et al. 2014; Papež et al. 2018; Seljebotn et al. 2019).

The presented techniques, rendered in a form of efficient, high performance codes, should allow for the efficient maximization of the data likelihood or the posterior distributions in the component separation problems producing the reliable sky component estimates for at least some of the forthcoming data sets. However, in the cases of sampling algorithms, when many thousands of the linear systems may need to be solved, this still remains to be demonstrated and further improvements will likely be required. Those will however in general depend on specific details of the employed sampling technique and we leave them here to future work.

*Acknowledgements.* We thank Olivier Tissot for insightful discussions and Dominic Beck and Josquin Errard for their help with numerical experiments. The first two authors' work was supported by the NLAFET project as part of European Union's Horizon 2020 research and innovation program under grant 671633. This work was also supported in part by the French National Research Agency (ANR) contract ANR-17-C23-0002-01 (project B3DCMB). This research used resources of the National Energy Research Scientific Computing Center (NERSC), a DOE Office of Science User Facility supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.

## Appendix A: Eigenvalue multiplicity in component separation problem - a worked example.

In this section we discuss the eigenvector and eigenvalue of the preconditioned matrix defined in Eq. (2.19) in the context of eigenvalue multiplicity in some specific set-up, which in particular assumes that the pointing matrix is the same for each frequency and that the noise has the same covariance (up to a scaling factor) for each frequency, i.e., that

$$P_f = P, \qquad N_f = N, \qquad f = 1, \ldots, n_{\mathrm{freq.}}.$$

While such requirements are not very likely to be realized strictly in any actual data set, they can be fulfilled approximately leading to near multiplicities of the eigenvalues. Those if not accounted for may be as harmful to the action of the preconditioner as the exact ones. Moreover, this worked example demonstrates that the component separation problem is in general expected to be more affected by this kind of effects than for instance the standard map-making solver, therefore emphasizing that a due diligence is necessary in this former application.

First, let $(\lambda_i, \mathsf{v}_i) = (\lambda_i, \lceil v_{i,q}, v_{i,u} \rfloor)$ be an eigenpair of the map-making matrix, i.e., there holds

$$P^\top N^{-1} P\, \mathsf{v}_i = \lambda_i P^\top \operatorname{diag}(N^{-1})\, P\, \mathsf{v}_i. \tag{A.1}$$

We note that

$$\widetilde{M}_\beta \lceil \mathsf{v}_i, 0, 0 \rfloor = \widetilde{M}_\beta \begin{bmatrix} v_{i,q} \\ v_{i,u} \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \begin{bmatrix} \alpha_{1,1}\, v_{i,q} \\ \alpha_{1,1}\, v_{i,u} \\ \vdots \\ \alpha_{n_{\mathrm{freq}},1}\, v_{i,q} \\ \alpha_{n_{\mathrm{freq}},1}\, v_{i,u} \end{bmatrix} = \begin{bmatrix} \alpha_{1,1}\, \mathsf{v}_i \\ \vdots \\ \alpha_{n_{\mathrm{freq}},1}\, \mathsf{v}_i \end{bmatrix} \tag{A.2}$$

due to the assumed form of the mixing Eq. (2.21). Consequently, using Eq. (A.1),

$$\begin{aligned} \widetilde{A}\widetilde{M}_\beta \lceil \mathsf{v}_i, 0, 0 \rfloor &= \begin{bmatrix} \alpha_{1,1}\, P^\top N^{-1} P\, \mathsf{v}_i \\ \vdots \\ \alpha_{n_{\mathrm{freq}},1}\, P^\top N^{-1} P\, \mathsf{v}_i \end{bmatrix} \\ &= \begin{bmatrix} \alpha_{1,1}\, \lambda_i P^\top \operatorname{diag}(N^{-1}) P\, \mathsf{v}_i \\ \vdots \\ \alpha_{n_{\mathrm{freq}},1} \lambda_i P^\top \operatorname{diag}(N^{-1}) P\, \mathsf{v}_i \end{bmatrix} = \lambda_i \widetilde{B}\widetilde{M}_\beta \lceil \mathsf{v}_i, 0, 0 \rfloor . \end{aligned}$$

Since the matrix $\widetilde{M}_\beta^\top \widetilde{A}\widetilde{M}_\beta$ is assumed to be non-singular (equivalently, since $\widetilde{M}_\beta$ is of full column rank), we can multiply the above equation from the left by $\widetilde{M}_\beta^\top$ showing that $(\lambda_i, \lceil \mathsf{v}_i, 0, 0 \rfloor)$ is the eigenpair of the matrix $(\widetilde{M}_\beta^\top \widetilde{B}\widetilde{M}_\beta)^{-1} \widetilde{M}_\beta^\top \widetilde{A}\widetilde{M}_\beta$. We can proceed analogously for the vectors $\lceil 0, \mathsf{v}_i, 0 \rfloor$ and $\lceil 0, 0, \mathsf{v}_i \rfloor$, with replacing in (A.2) $\alpha_{f,1}$ by $\alpha_{f,2}$ and $\alpha_{f,3}$, respectively.

There are $2\, n_{\mathrm{pix}}$ eigenpairs for $(P^\top \operatorname{diag}(N^{-1})P)^{-1}(P^\top N^{-1} P)$. As we have seen above, each of them generates three eigenpairs (with the same eigenvalue) of $(\widetilde{M}_\beta^t \widetilde{B}\widetilde{M}_\beta)^{-1} \widetilde{M}_\beta^\top \widetilde{A}\widetilde{M}_\beta$. This gives together $6\, n_{\mathrm{pix}}$ eigenpairs, in other words, we have described the full spectrum of the preconditioned system matrix in (2.19).

Finally, we remark that all the eigenpairs of the preconditioned matrix are, in the simplified setting, independent of the parameters $\beta$. In such case, we suggest to use a specialized eigensolver (as, e.g., ARPACK Lehoucq et al. (1998)) to compute the eigenpairs from (A.1), build the triplets of eigenvectors $\lceil \mathsf{v}_i, 0, 0 \rfloor$, $\lceil 0, \mathsf{v}_i, 0 \rfloor$, $\lceil 0, 0, \mathsf{v}_i \rfloor$, and then use the deflated PCG with those vectors.

Figure A.1 is as Figure 5 but for the simplified setting. Here two triplets of the eigenvectors are constructed following the above procedure.

## Appendix B: Ingredients of the proposed procedure

We present in this section in more detail two ingredients of the proposed procedure. Namely, the ways how the eigenpairs are estimated using the computed basis of Krylov subspace and the ways how the deflation of the approximate eigenvectors can be combined with another preconditioner. For the ease of presentation, we simplify the notation in this section.

### *Appendix B.1: Approximating the eigenvalues using Krylov subspace methods*

Arnoldi and Lanczos algorithms for approximating the eigenvalues of a general nonsingular or, respectively, a hermitian matrix, are based on Rayleigh-Ritz approximation that will be presented first. Then, we recall the Arnoldi and Lanczos algorithms and, finally, we briefly comment on their restarted variants.

The methods discussed below do not represent an exhaustive overview of methods for approximating several eigenvalues and the associated eigenvectors. Among the omitted methods, there is, e.g., the Jacobi–Davidson method (Sleijpen & Van der Vorst 2000), which proved to be particularly efficient for approximating an inner part of the spectrum. For a survey on the methods and a list of references see, e.g., Sorensen (2002).

#### Appendix B.1.1: Ritz values and harmonic Ritz values approximations

For a subspace $\mathcal{S} \subset \mathbb{C}^n$, we call $y \in \mathcal{S}$ a *Ritz vector* of $A$ with *Ritz value* $\theta$ if

$$Ay - \theta y \perp \mathcal{S}.$$

Using a (computed) basis $V_j$ of $\mathcal{S}$ and setting $y = V_j w$, the above relation is equivalent to solving

$$V_j^\top A V_j w = \theta V_j^\top V_j w. \tag{B.1}$$

Ritz values are known to approximate well the extreme eigenvalues of $A$. If an approximation to the interior eigenvalues is required, computing the harmonic[4] Ritz values can be preferable. Following Parks et al. (2006), we define harmonic Ritz values as the Ritz values of $A^{-1}$ with respect to the space $A\mathcal{S}$,

$$\widetilde{y} \in A\mathcal{S}, \quad A^{-1}\widetilde{y} - \widetilde{\mu}\widetilde{y} \perp A\mathcal{S}.$$

We call $\widetilde{\theta} \equiv 1/\widetilde{\mu}$ a *harmonic Ritz value* and $\widetilde{y}$ a *harmonic Ritz vector*. If $V_j$ is a basis of $\mathcal{S}$ and $\widetilde{y} = V_j \widetilde{w}$, the above relation can be represented as

$$V_j^\top A^\top V_j \widetilde{w} = \widetilde{\mu} V_j^\top A^\top A V_j \widetilde{w} \iff V_j^\top A^\top A V_j \widetilde{w} = \widetilde{\theta} V_j^\top A^\top V_j \widetilde{w}. \tag{B.2}$$

For the properties of the harmonic Ritz values approximations and the relationship with the iteration polynomial in MINRES method, see Paige et al. (1995).

*Remark 1.* There are various ways in the literature how the harmonic (Rayleigh–)Ritz procedure is presented and defined; often it is introduced to approximate eigenvalues close to a target $\tau \in \mathbb{C}$. For example, Wu (2017) prescribes the procedure by

$$\widetilde{y} \in \mathcal{S}, \quad A\widetilde{y} - \widetilde{\theta}\widetilde{y} \perp (A - \tau I)\mathcal{S}, \tag{B.3}$$

[4] The term *harmonic Ritz values* was introduced in Paige et al. (1995), where the references to previous works using this approximation can be found.

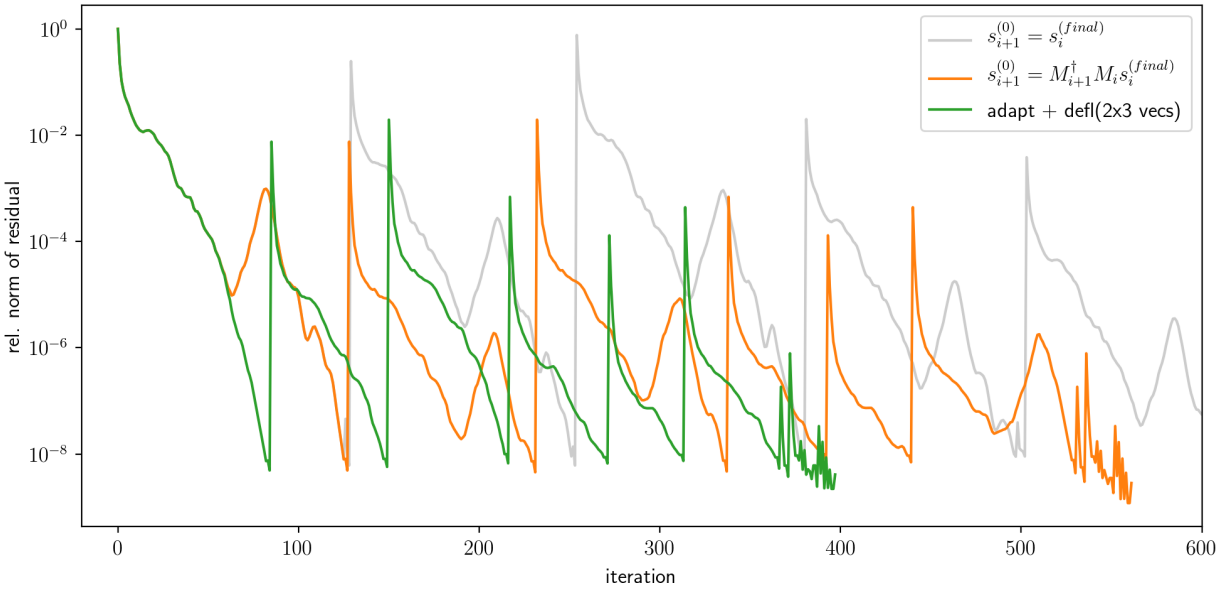


**Fig. A.1.** An analogy of Figure 5 for the simplified setting of Appendix A. Comparison of PCG with different choices of initial guess (as in Eq. (3.1) and Eq. (3.3)) and the deflated PCG with $2 \times 3$ vectors.

where $I$ is the identity matrix. With $\widetilde{y} = V_j\widetilde{w}$ this corresponds to the generalized eigenvalue problem

$$V_j^\top (A-\tau I)^\top (A-\tau I) V_j \widetilde{w} = (\widetilde{\theta}-\tau)(V_j^\top (A-\tau I)^\top V_j)\widetilde{w},$$

which becomes for $\tau = 0$ exactly the right-hand side equality in Eq. (B.2).

We note that harmonic Ritz approximation is often used in the Krylov subspace recycling methods also for approximating the smallest (in magnitude) eigenvalues and the associated eigenvectors.

Finally, we comment on the (harmonic) Ritz approximation in the case we want to compute the eigenvalues of the matrix $A$ preconditioned from the left by $M$. In a general case, assuming only that $A$, $M$ are nonsingular, the Ritz and harmonic Ritz approximation are applied as above just replacing in the formulas $A$ by $M^{-1}A$. When the matrix $A$ is hermitian and the preconditioner $M$ is SPD, there is also another option. First, we note that the matrix $M^{-1}A$ is not hermitian but it is self-adjoint with respect to the inner product induced by $M$, that is

$$(v, M^{-1}Aw)_M = (M^{-1}Av, w)_M, \qquad \forall v, w,$$

where $(v,w)_M \equiv v^\top M w$. This allows, in the definition of Ritz and harmonic Ritz approximation, to replace $A$ by $M^{-1}A$ and the standard inner product by the inner product induced by the matrix $M$, giving

$$y \in \mathcal{S}, \quad M^{-1}Ay - \theta y \perp_M \mathcal{S},$$

respectively,

$$\widetilde{y} \in M^{-1}A\mathcal{S}, \quad (M^{-1}A)^{-1}\widetilde{y} - \widetilde{\mu}\widetilde{y} \perp_M M^{-1}A\mathcal{S}.$$

The corresponding algebraic problems with $y = V_j w$, $\widetilde{y} = V_j\widetilde{w}$, are

$$V_j^\top A V_j w = \theta V_j^\top M V_j w,$$

respectively,

$$V_j^\top A^\top M^{-1} A V_j \widetilde{w} = (1/\widetilde{\mu}) V_j^\top A^\top V_j \widetilde{w}.$$

Note that the problems above involve hermitian matrices only.

### Appendix B.1.2: Arnoldi and Lanczos methods

Arnoldi and Lanczos algorithms for approximating the eigenvalues of a general nonsingular or, respectively, a hermitian matrix, are based on Ritz approximation with setting $\mathcal{S} = \mathcal{K}_j(A, v_1) = \mathrm{span}(v_1, Av_1, \ldots, A^{j-1}v_1)$, the $j$th Krylov subspace. The methods compute an orthogonal basis $V_j$ of $\mathcal{S}$ such that,

$$AV_j = V_j T_j + \beta v_{j+1} e_j^\top,$$

where $e_j$ is the last column vector of the identity matrix (of size $j$) and $V_j^\top V_j = I$, $V_j^\top v_{j+1} = 0$. Consequently, the eigenvalue problem (B.1) corresponding to the Ritz approximation reads

$$T_j w = \theta w.$$

The matrix $T_j$ is available during the iterations. The standard use of Arnoldi and Lanczos method for eigenvalue approximation consists of solving the above problem and setting the pairs $(\theta, V_j w)$ as the computed approximations.

One can naturally replace the Ritz approximation by the harmonic Ritz approximation. Then, the matrices in Eq. (B.2) become,

$$V_j^\top A^\top A V_j = T_j^\top T_j + \beta^2 e_j e_j^\top, \qquad V_j^\top A^\top V_j = T_j^\top.$$

*Remark 2.* The Lanczos algorithm is a variant of the Arnoldi algorithm for a hermitian $A$. The matrix $T_j = V_j^\top A V_j$, which is in Arnoldi method upper Hessenberg, is then also hermitian. Consequently, it is tridiagonal, which means that in each step of Lanczos method we orthogonalize the new vector only against two previous. This assures that the computational cost of each iteration is fixed and, if one is interested in approximating the eigenvalues only, storing three vectors $v_{j-1}$, $v_j$ and $v_{j+1}$ is sufficient instead of handling the full matrix $V_j$. The assumption on exact arithmetic is, however, crucial here. In finite precision computations, the global orthogonality is typically quickly lost, which can cause several stability issues.

As noted above, an orthonormal basis $V_j$ of $\mathcal{S}$ is advantageous for the Ritz approximation. For the harmonic Ritz approximation applied to an SPD matrix $A$, one can instead aim at constructing an $A$-orthonormal basis, which assures that the matrix $V_j^\top A^\top V_j = V_j^\top A V_j$ on the right-hand side of Eq. (B.2) is equal to the identity. An $A$-orthonormal basis of a Krylov subspace can be constructed within the iterations of conjugate gradient method by using the search direction vectors.

The Arnoldi method can be naturally applied also to the preconditioned matrix $M^{-1}A$ to compute an orthonormal basis $V_j$ of the associated Krylov subspace $\mathcal{K}_j(M^{-1}A, M^{-1}v_1)$, giving,

$$M^{-1}AV_j = V_jT_j + \beta v_{j+1}e_j^\top, \qquad V_j^\top V_j = I, \quad V_j^\top v_{j+1} = 0.$$

For a hermitian $A$ and an SPD preconditioner $M$, we can apply the Lanczos method following the comment made above – using the matrix $M^{-1}A$ and the inner product $(\cdot,\cdot)_M$ induced by $M$ instead of the standard euclidean $(\cdot,\cdot)$, giving,

$$M^{-1}AV_j = V_jT_j + \beta v_{j+1}e_j^\top, \qquad V_j^\top MV_j = I, \quad V_j^\top Mv_{j+1} = 0.$$

The computed basis $V_j$ is therefore $M$-orthonormal.

#### Appendix B.1.3: Restarted variants

The number of iterations necessary to converge is not a priori known in Arnoldi and Lanczos algorithms and, in general, it can be very high. High iteration counts require a large memory to store the basis vectors and, whenever a full-reorthogonalization is used, a high computational effort because of the growing cost of the reorthogonalization in each step. The idea behind implicitly restarted variants is to limit the dimension of the search space $\mathcal{S}$. This means that the iterations are stopped after a (fixed) number of steps, the dimension of the search space is reduced while maintaining its (Krylov) structure, and the Arnoldi/Lanczos iterations are resumed.

There are several restarted variants described in the literature (a detailed description is, however, beyond the scope of this paper): the implicitly restarted Arnoldi (IRA, Sorensen (1992)), the implicitly restarted Lanczos (IRL, Calvetti et al. (1994)), or the Krylov–Schur method (Stewart (2001/02); Wu & Simon (2000)).

The estimation of the spectrum of $A$ is possible within the GMRES, MINRES and CG iterations (applied to solve a system with $A$) because they are based on Arnoldi, respectively Lanczos algorithms. In contrast, a combination of restarted variants with solving a linear algebraic system is, to the best of our knowledge, not described in the literature.

### *Appendix B.2: Deflation and two-levels preconditioners*

In this section we first discuss a deflation preconditioner for Krylov subspace methods that can be regarded as eliminating the effect of several (given) vectors from the operator or, equivalently, augmenting by these vectors the space where we search for an approximation. Then we describe a combination of the deflation preconditioner with another preconditioner that is commonly used in practice.

The Krylov subspace methods (in particular CG, Hestenes & Stiefel. (1952), and GMRES, Saad & Schultz (1986)) are well-known for their minimization (optimal) properties over the consecutively built *Krylov subspace*

$$\mathcal{K}_j(A, v) = \mathrm{span}\{v, Av, A^2v, \ldots, A^{j-1}v\}.$$

A question then arises whether given some other subspace $\mathcal{U}$, we can modify the methods such that they have the same optimal properties over the union of $\mathcal{K}_j(A, v)$ and $\mathcal{U}$, which is often called an *augmented* Krylov subspace. The answer is positive and the implementation differs on the method — it is straightforward for GMRES and it requires more attention for CG. Hereafter, we denote by $I$ the identity matrix and by $Z$ the basis of $\mathcal{U}$.

The deflation in GMRES method is often (see, e.g., GCROT by Morgan (1995)) considered as a remedy to overcome the difficulties caused by restarts: for computational and memory restrictions, only a fixed number of GMRES iterations is typically performed giving an approximation that is then used as the initial vector for a new GMRES run. In GCROT, several vectors are saved and used to augment the Krylov subspace built after the restart. The GMRES method with the deflation was used for solving a sequence of linear algebraic systems, e.g., in Parks et al. (2006).

The augmentation of the Krylov subspace in CG is more delicate, since the original CG method can only be applied to an SPD matrix. The first such algorithm was proposed in Dostál (1988) and Nicolaides (1987). We note that it includes the construction of the conjugate projector

$$P_{c.proj.} = Z(Z^\top AZ)^{-1}Z^\top A$$

and, in each iteration, the computation of the preconditioned search direction $q_i = (I - P_{c.proj.})\, p_i$ and of the vector $Aq_i$. The latter can be avoided at the price of storing $Z$ *and* $AZ$ and performing additional multiplication with $AZ$. In both variants, the cost of a single iteration is higher than the cost of one standard CG iteration.

The combination of a preconditioner with a deflation is widely studied in the literature and therefore we present this only briefly; more details and extensive list of references can be found, e.g., in the review paper by Tang et al. (2009). The preconditioner stemming from the combination of a (typically relatively simple) traditional preconditioner with the deflation is called a *two-level preconditioner*[5]. While the traditional preconditioner aims at removing the effect of the largest (in magnitude) eigenvalues, the deflation (projection-type preconditioner) is intended to get rid of the effect of the smallest eigenvalues. Common choices for the traditional preconditioner are block Jacobi, (restricted) additive Schwarz method, and incomplete LU or Cholesky factorizations. Among many applications, two-level preconditioners proved to be efficient in the CMB data analysis; see, e.g., Grigori et al. (2012); Szydlarski et al. (2014).

We now present the combination of the traditional and projection-type (deflation) preconditioners following the discussion and notation of Tang et al. (2009). Hereafter, we assume that the system matrix $A$ and the traditional preconditioner $M$ are SPD. We note that some of the below mentioned preconditioners $\mathcal{P}_\square$ are not symmetric. However, their properties allow us to use them (with possible modification of the initial vector) as left preconditioners in PCG; see Tang et al. (2009) for details.

Let the deflation space spans the columns of the matrix $Z$. We denote

$$P \equiv I - AQ, \qquad Q \equiv Z(Z^\top AZ)^{-1}Z^\top. \tag{B.4}$$

Two-level preconditioners based on the deflation are given as

$$\mathcal{P}_{\mathrm{DEF1}} \equiv M^{-1}P, \qquad \mathcal{P}_{\mathrm{DEF2}} \equiv P^\top M^{-1}.$$

[5] As shown in Tang et al. (2009), one can see here an analogy with multilevel (multigrid) and domain decomposition methods.

Other preconditioners can be determined using the *additive* combination of two (SPD) preconditioners $C_1, C_2$ as

$$\mathcal{P}_{\text{add}} \equiv C_1 + C_2,$$

or, using the *multiplicative* combination of the preconditioners, as

$$\mathcal{P}_{\text{mult}} \equiv C_1 + C_2 - C_2 A C_1.$$

Three variants of two-level preconditioners are derived by choosing additive or multiplicative combination and setting $C_1 = M^{-1}$, $C_2 = Q$, or $C_1 = Q$, $C_2 = M^{-1}$. Other preconditioners can be derived using the multiplicative combination of three SPD matrices; see (Tang et al. 2009, Section 2.3.4).

The variants of two-level preconditioner mentioned above differ in the implementation cost and also in the numerical stability; see Tang et al. (2009). The variant $\mathcal{P}_{\text{DEF1}}$, which is often used in the procedures for solving the sequences of linear systems (see, e.g., Saad et al. (2000)), was found cheap but less robust, especially with respect to the accuracy of solving the coarse problem with the matrix $Z^\top AZ$ and with respect to the demanded accuracy. The conclusion drawn in Tang et al. (2009) is that "A-DEF2 seems to be the best and most robust method, considering the theory, numerical experiments and the computational cost". Therefore the preconditioner $\mathcal{P}_{\text{A-DEF2}}$,

$$\mathcal{P}_{\text{A-DEF2}} \equiv M^{-1} + Q - QAM^{-1} = P^\top M^{-1} + Q, \tag{B.5}$$

is of interest, in particular in the cases where the dimension of the deflation space (equal to the number of columns in $Z$) is high and/or the matrix $M^{-1}A$ is ill-conditioned,

As noted in Saad et al. (2000), the gradual lost of orthogonality of computed residuals with respect to the columns of $Z$ can cause stagnation, divergence or erratic behaviour of errors within the iterations; see also the comment in (Tang et al. 2009, Section 4.7). The suggested remedy in that case consists in the reorthogonalization of computed residuals as

$$r_j := W r_j, \qquad W \equiv I - Z(Z^\top Z)^{-1} Z^\top. \tag{B.6}$$

However, such instabilities were not observed in our experiments and the results depicted throughout the paper are for the standard (non-reorthogonalized) variant.

## Appendix C: Full algorithm

In this section we provide the pseudocodes for the deflated PCG (Algorithm 1), the subspace recycling (Algorithm 2), and for the full procedure (Algorithm 3) proposed in this paper and used in the numerical experiments in Section 4.

**Algorithm 1** deflated PCG (variant "def1")

**function** DEFLPCG($\mathbb{A}$, $\mathbb{B}$, $\mathbb{b}$, $\mathbb{s}^{(0)}$, $Z$, $\dim_p$, $j_{\max}$)
  $Q = Z(Z^\top \mathbb{A} Z)^{-1} Z^\top$ ▷ in practice we save $\mathbb{A}Z$ to use later
  $\mathbb{r}^{(0)} = (I - \mathbb{A}Q)(\mathbb{b} - \mathbb{A}\mathbb{s}^{(0)})$ ▷ with saved $\mathbb{A}Z$, $Q\mathbb{A}$ and $\mathbb{A}Q$ can be computed without applying $\mathbb{A}$
  $\mathbb{p}^{(0)} = \mathbb{r}^{(0)}$
  $\widetilde{\mathbb{r}}^{(0)} = \mathbb{B}^{-1}\mathbb{r}^{(0)}$
  **for** $j = 0, \ldots, j_{\max}$ **do**
    $\mathbb{w}^{(j)} = (I - \mathbb{A}Q)(\mathbb{A}\mathbb{p}^{(j)})$
    **if** $j \le \dim_p$ **then**
      **save** $(I - Q\mathbb{A})\mathbb{p}^{(j)}$ into $\widetilde{Z}$ ▷ in practice we save also $\mathbb{w}^{(j)}$ to avoid computation of $\mathbb{A}\widetilde{Z}$ later
    **end if**
    $\gamma^{(j)} = (\widetilde{\mathbb{r}}^{(j)}, \mathbb{r}^{(j)})/(\mathbb{p}^{(j)}, \mathbb{w}^{(j)})$
    $\mathbb{s}^{(j+1)} = \mathbb{s}^{(j)} + \gamma^{(j)}\mathbb{p}^{(j)}$
    $\mathbb{r}^{(j+1)} = \mathbb{r}^{(j)} - \gamma^{(j)}\mathbb{w}^{(j)}$
    **check** the stopping criterion
    $\widetilde{\mathbb{r}}^{(j+1)} = \mathbb{B}^{-1}\mathbb{r}^{(j+1)}$
    $\delta^{(j)} = (\widetilde{\mathbb{r}}^{(j+1)}, \mathbb{r}^{(j+1)})/(\widetilde{\mathbb{r}}^{(j)}, \mathbb{r}^{(j)})$
    $\mathbb{p}^{(j+1)} = \widetilde{\mathbb{r}}^{(j+1)} + \delta^{(j)}\mathbb{p}^{(j)}$
  **end for**
  $\mathbb{s}^{(final)} := Q\mathbb{b} + (I - Q\mathbb{A})\mathbb{s}^{(j)}$
  **return** $\mathbb{s}^{(final)}$; $\widetilde{Z}$
  ▷ to be efficient we return also $\mathbb{A}Z$ and the vectors $\{\mathbb{w}^{(j)}\}$
**end function**

**Algorithm 2** subspace recycling (variant "ritz")

**function** SUBSPREC($\mathbb{A}$, $\mathbb{B}$, $U$, $k$)
  $F = U^\top \mathbb{B} U$
  $G = U^\top \mathbb{A} U$ ▷ in practice we reuse $\mathbb{A}U$ saved during deflated PCG
  **solve** the generalized eigenvalue problem $GY = \text{diag}(\lambda_i)FY$
  **take** $k$ smallest $\lambda_i$ and the respective columns of $Y$, $Y_k$
  **return** $Z = UY_k$
**end function**

**Algorithm 3** full algorithm of the procedure

**Require:** $\beta_0$, $\mathbb{s}^{(0)}_{\beta_0}$

**Require:** $k$, $\dim_p$

set $Z := []$

**for** $i = 0, \ldots, i_{\max}$ **do**

**assembly** the system matrix $\mathbb{A}$, right-hand side $\mathbb{b}$, and the preconditioner $\mathbb{B}$ corresponding to $\beta_i$

**if** $i > 0$ **then** $\mathbb{s}^{(0)}_{\beta_i} = (\widetilde{M}_{\beta_i})^{\dagger}\mathbb{s}^{(0)}_{\beta_i}$

DEFLPCG($\mathbb{A}$, $\mathbb{B}$, $\mathbb{b}$, $\mathbb{s}^{(0)}_{\beta_i}$, $Z$, $\dim_p$, $j_{\max}$) $\longrightarrow$ ($\mathbb{s}^{(final)}_{\beta_i}$, $\widetilde{Z}$)

**check** the stopping criterion for $\beta_i$, **exit** if $i = i_{\max}$

$\mathbb{s}^{(0)}_{\beta_{i+1}} = \widetilde{M}_{\beta_i}\mathbb{s}^{(final)}_{\beta_i}$

(determine $\beta_{i+1}$) ▷ considered here as a black-box

SUBSPREC($\mathbb{A}$, $\mathbb{B}$, $U = [Z, \widetilde{Z}]$, $k$) $\longrightarrow Z$

**end for**

## Appendix D: Results for additional sequence of mixing parameters from a Monte Carlo sampling

In this section we provide results for a sequence of spectral parameters generated by a Monte Carlo sampling algorithm. In realistic circumstances such sequences contain possibly up to many thousands of samples, however here for computational efficiency we restrict ourselves to a sub-sequence made of merely 30 elements and use them in order to demonstrate performance of the proposed approach on a sequence with realistic properties but sufficiently different than those of the sequences encountered in the likelihood maximization process. We emphasize that it is not our purpose here to validate the method on the actual application yet and more work is needed to achieve this goal, see Sect. 5. The sequence is depicted in Figure D.1. It was produced using a publicly available software FGBUSTER[6] and indeed shows qualitatively a very different behavior than that of our standard case displayed in Fig. 1.

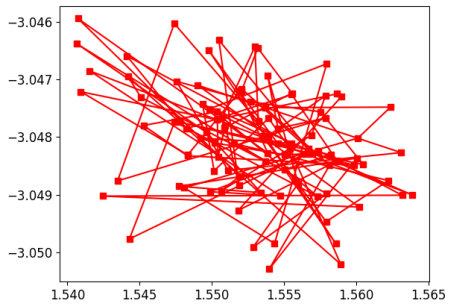


**Fig. D.1.** Plot of a sequence of the spectral parameters $\beta_i = [\beta_{i,s}, \beta_{i,d}]$ drawn via a Monte Carlo sampling technique and used as an alternative test case in numerical experiments described in Appendix D.

In Figure D.2 and in Table D.1, we give a comparison of the results obtained in this case applying the various techniques discussed and proposed in this work.

[6] FGBUSTER: `https://github.com/fgbuster`

| | #MatVecs | | |
|---|---|---|---|
| | iteration | deflation | total |
| $\mathbb{s}^{(0)}_{\beta_{i+1}}$ as in (3.1) | 4010 | 0 | 4010 |
| $\mathbb{s}^{(0)}_{\beta_{i+1}}$ as in (3.3) | 1768 | 0 | 1768 |
| recycle + $\mathbb{s}^{(0)}_{\beta_{i+1}}$ as in (3.3) | 1228 | 290 | 1518 |

**Table D.1.** The number of matrix-vector products (MatVecs) for different techniques as in Figure D.2.

## References


Brandt, W. N., Lawrence, C. R., Readhead, A. C. S., Pakianathan, J. N., & Fiola, T. M. 1994, ApJ, 424, 1

Calvetti, D., Reichel, L., & Sorensen, D. C. 1994, Electron. Trans. Numer. Anal., 2, 1

Cantalupo, C. M., Borrill, J. D., Jaffe, A. H., Kisner, T. S., & Stompor, R. 2010, ApJS, 187, 212

Dostál, Z. 1988, International Journal of Computer Mathematics, 23, 315

Eriksen, H. K., Jewell, J. B., Dickinson, C., et al. 2008, ApJ, 676, 10

Gaul, A., Gutknecht, M. H., Liesen, J., & Nabben, R. 2013, SIAM J. Matrix Anal. Appl., 34, 495

Górski, K. M., Hivon, E., Banday, A. J., et al. 2005, The Astrophysical Journal, 622, 759

Grigori, L., Stompor, R., & Szydlarski, M. 2012, in SC 2012: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis, 1–10

Hestenes, M. R. & Stiefel., E. 1952, Journal of research of the National Bureau of Standards., 49, 409

Jolivet, P. & Tournier, P.-H. 2016, in Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, SC '16 (Piscataway, NJ, USA: IEEE Press), 17:1–17:14

Kilmer, M. E. & de Sturler, E. 2006, SIAM J. Sci. Comput., 27, 2140

Lehoucq, R. B., Sorensen, D. C., & Yang, C. 1998, Software, Environments, and Tools, Vol. 6, ARPACK users' guide (Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA), xvi+142

Morgan, R. B. 1995, SIAM J. Matrix Anal. Appl., 16, 1154

Natoli, P., de Gasperis, G., Gheller, C., & Vittorio, N. 2001, A&A, 372, 346

Nicolaides, R. 1987, SIAM Journal on Numerical Analysis, 24, 355

O'Connell, M., Kilmer, M. E., de Sturler, E., & Gugercin, S. 2017, SIAM Journal on Scientific Computing, 39, B272

Paige, C. C., Parlett, B. N., & van der Vorst, H. A. 1995, Numer. Linear Algebra Appl., 2, 115

Papež, J., Grigori, L., & Stompor, R. 2018, A&A, 620, A59

Parks, M. L., de Sturler, E., Mackey, G., Johnson, D. D., & Maiti, S. 2006, SIAM J. Sci. Comput., 28, 1651

Planck Collaboration, Adam, R., Ade, P. A. R., et al. 2016a, Astronomy & Astrophysics, 594, A10

Planck Collaboration, Ade, P. A. R., Aghanim, N., et al. 2016b, Astronomy & Astrophysics, 594, A13

Puglisi, G., Poletti, D., Fabbian, G., et al. 2018, A&A, 618, A62

Saad, Y. & Schultz, M. H. 1986, SIAM Journal on Scientific and Statistical Computing, 7, 856

Saad, Y., Yeung, M., Erhel, J., & Guyomarc'h, F. 2000, SIAM J. Sci. Comput., 21, 1909

Seljebotn, D. S., Bærland, T., Eriksen, H. K., Mardal, K. A., & Wehus, I. K. 2019, A&A, 627, A98

Sleijpen, G. L. G. & Van der Vorst, H. A. 2000, SIAM Rev., 42, 267

Sorensen, D. C. 1992, SIAM J. Matrix Anal. Appl., 13, 357

Sorensen, D. C. 2002, Acta Numerica, 11, 519–584

Stewart, G. W. 2001/02, SIAM J. Matrix Anal. Appl., 23, 601

Stompor, R., Leach, S., Stivoli, F., & Baccigalupi, C. 2009, MNRAS, 392, 216

Sudarsan, R., Borrill, J., Cantalupo, C., et al. 2011, in Proceedings of the International Conference on Supercomputing, ICS '11 (New York, NY, USA: Association for Computing Machinery), 305–316

Szydlarski, M., Grigori, L., & Stompor, R. 2014, A&A, 572, A39

Tang, J. M., Nabben, R., Vuik, C., & Erlangga, Y. A. 2009, J. Sci. Comput., 39, 340

Wu, G. 2017, SIAM J. Matrix Anal. Appl., 38, 118

Wu, K. & Simon, H. 2000, SIAM J. Matrix Anal. Appl., 22, 602

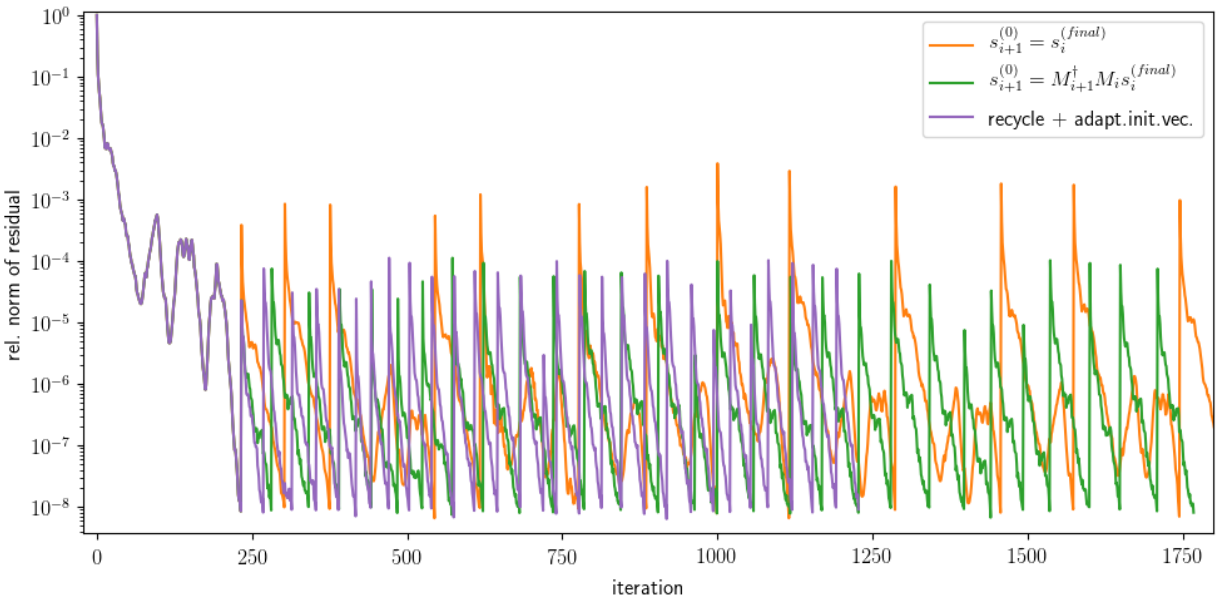


**Fig. D.2.** Comparison of PCG with different choices of initial guess (as in (3.1) and (3.3)) and the PCG with the subspace recycling (together with the choice of the initial guess as in (3.3)). For the recycling we consider $k = 10$ eigenvectors approximated using $\dim_p = 100$. The convergence for the whole sequence when using the initial guess (3.1) (the yellow line) requires 4010 iterations.