

**Electronic Supplementary Information for  
New scaling relations to compute atom-in-material polarizabilities and  
dispersion coefficients: part 2. Linear-scaling computational algorithms and  
parallelization**

Thomas A. Manz<sup>1\*</sup> and Taoyi Chen<sup>1</sup>

<sup>1</sup>Department of Chemical & Materials Engineering, New Mexico State University, Las Cruces,  
New Mexico, 88003-8001.

\*Corresponding author email: [tmanz@nmsu.edu](mailto:tmanz@nmsu.edu)

**Contents**

**S1. Proof that  $\alpha^{\text{force-field}}$ ,  $\alpha^{\text{non-dir}}(\mathbf{u})$ ,  $\alpha^{\text{low\_freq}}$ , and  $\alpha^{\text{screened}}(\mathbf{u})$  are  $\geq 0$**

**S2. Derivations and proofs related to the failproof conjugate residual (FCR) algorithm**

**S3. Proof the wp lookup table method yields error proportional to the increment squared**

**S4. Linear-scaling algorithm for setting up the lists of interacting atom pairs**

**S1. Proof that  $\alpha^{\text{force-field}}$ ,  $\alpha^{\text{non-dir}}(\mathbf{u})$ ,  $\alpha^{\text{low\_freq}}$ , and  $\alpha^{\text{screened}}(\mathbf{u})$  are  $\geq 0$**

Incremental non-directional screening takes the form

$$\alpha_{j+1}^A(\mathbf{u}) = \alpha_j^A(\mathbf{u}) - \sum_{b \in \text{small\_list}} \left[ \left( \frac{2\alpha_A^{\text{unscreened}}(\mathbf{u})}{\alpha_A^{\text{unscreened}}(\mathbf{u}) + \alpha_B^{\text{unscreened}}(\mathbf{u})} \right) (\alpha_j^A(\mathbf{u}) (\Delta_j \tau_{\text{non-dir},j}^{\text{Ab}}(\mathbf{u})) \alpha_j^b(\mathbf{u})) \right] \quad (\text{S1})$$

where  $\Delta_j$  is the screening increment. Eqn (S1) can be factored to give

$$\alpha_{j+1}^A(\mathbf{u}) = \alpha_j^A(\mathbf{u}) \left( 1 - \sum_{b \in \text{small\_list}} \left( \frac{2\alpha_A^{\text{unscreened}}(\mathbf{u}) (\Delta_j \tau_{\text{non-dir},j}^{\text{Ab}}(\mathbf{u})) \alpha_j^b(\mathbf{u})}{\alpha_A^{\text{unscreened}}(\mathbf{u}) + \alpha_B^{\text{unscreened}}(\mathbf{u})} \right) \right) \quad (\text{S2})$$

Since the MCLF method corresponds to the limit  $\Delta_j \rightarrow 0$ , we have

$$\lim_{\Delta_j \rightarrow 0} \left( 1 - \sum_{b \in \text{small\_list}} \left( \frac{2\alpha_A^{\text{unscreened}}(\mathbf{u}) (\Delta_j \tau_{\text{non-dir},j}^{\text{Ab}}(\mathbf{u})) \alpha_j^b(\mathbf{u})}{\alpha_A^{\text{unscreened}}(\mathbf{u}) + \alpha_B^{\text{unscreened}}(\mathbf{u})} \right) \right) > 0 \quad (\text{S3})$$

Combining eqn (S2) and (S3) gives

$$\lim_{\Delta_j \rightarrow 0} \left( \frac{\alpha_{j+1}^A(\mathbf{u})}{\alpha_j^A(\mathbf{u})} \right) > 0 \quad (\text{S4})$$

Hence,  $\alpha_{j+1}^A(\mathbf{u})$  and  $\alpha_j^A(\mathbf{u})$  have the same sign. Since  $\alpha_{j=1}^A(\mathbf{u}) = \alpha_A^{\text{unscreened}}(\mathbf{u}) \geq 0$ , it follows that  $\alpha_j^A(\mathbf{u}) \geq 0$  for all values of  $j$ . Since  $\alpha_A^{\text{non-dir}}(\mathbf{u})$  corresponds to the final value of  $j$  in the limit  $\Delta_j \rightarrow 0$ , it necessarily follows that

$$\alpha_A^{\text{non-dir}}(\mathbf{u}) \geq 0 \quad (\text{S5})$$

Since  $\alpha_A^{\text{force-field}} = \alpha_A^{\text{non-dir}}(\mathbf{u} = \text{Nimfreqs})$ , then  $\alpha_A^{\text{force-field}} \geq 0$ .

Incremental directional screening to compute  $\alpha_A^{\text{screened}}(\mathbf{u})$  takes the form

$$\mathbf{M1} = (\Delta_j \mathbf{f}_{\text{MBS}}^{\text{Ab}} \mathbf{f}_{\text{cutoff}}^{\text{Ab}} \phi_j^{\text{Ab}}(\mathbf{u})) (\vec{\alpha}_j^A(\mathbf{u}) \cdot (\vec{\eta}^{\text{Ab}}(\mathbf{u})) \cdot \vec{\alpha}_j^b(\mathbf{u})) \quad (\text{S6})$$

$$\mathbf{M2} = \Delta_j \vec{\alpha}_j^A(\mathbf{u}) \cdot \vec{\mathbf{h}}^{\text{AB}}(\mathbf{u}) \cdot \vec{\alpha}_j^B(\mathbf{u}) \quad (\text{S7})$$

$$\begin{aligned} \vec{\alpha}_{j+1}^A(\mathbf{u}) = \vec{\alpha}_j^A(\mathbf{u}) - \sum_{b \in \text{small\_list}} \left[ \left( \frac{\alpha_A^{\text{non-dir}}(\mathbf{u})}{\alpha_A^{\text{non-dir}}(\mathbf{u}) + \alpha_B^{\text{non-dir}}(\mathbf{u})} \right) (\mathbf{M1} + \mathbf{M1}^T) \right] \\ - \sum_{B \in \text{large\_list}} \left[ \left( \frac{\alpha_A^{\text{non-dir}}(\mathbf{u})}{\alpha_A^{\text{non-dir}}(\mathbf{u}) + \alpha_B^{\text{non-dir}}(\mathbf{u})} \right) (\mathbf{M2} + \mathbf{M2}^T) \right] \end{aligned} \quad (\text{S8})$$

where  $\mathbf{M1}$  and  $\mathbf{M2}$  are square matrices with 3 rows and

$$\phi_j^{\text{Ab}}(\mathbf{u}) = - \left( \frac{1}{(d_{\text{Ab}})^3} \operatorname{erfc} \left( \frac{d_{\text{Ab}}}{\sigma_{\text{AB},j}(\mathbf{u})} \right) + \left( \frac{2}{\sqrt{\pi}} \right) \left( \frac{1}{(d_{\text{Ab}})^2 \sigma_{\text{AB},j}(\mathbf{u})} + \frac{2}{3(\sigma_{\text{AB},j}(\mathbf{u}))^3} \right) \exp \left( - \left( \frac{d_{\text{Ab}}}{\sigma_{\text{AB},j}(\mathbf{u})} \right)^2 \right) \right) \quad (\text{S9})$$

$$\alpha_j^{\wedge}(\mathbf{u}) = \text{trace}(\bar{\bar{\alpha}}_j^{\wedge}(\mathbf{u}))/3 \quad (\text{S10})$$

Eqn (S8) can be factored to give

$$\bar{\bar{\alpha}}_{j+1}^{\wedge}(\mathbf{u}) = \frac{1}{2}\bar{\bar{\alpha}}_j^{\wedge}(\mathbf{u})(\mathbf{I} - \Delta_j \mathbf{M3}) + \frac{1}{2}(\mathbf{I} - \Delta_j \mathbf{M3})\bar{\bar{\alpha}}_j^{\wedge}(\mathbf{u}) = \bar{\bar{\alpha}}_j^{\wedge}(\mathbf{u}) - \frac{1}{2}\Delta_j \underbrace{\left( \bar{\bar{\alpha}}_j^{\wedge}(\mathbf{u})\mathbf{M3} + \mathbf{M3}\bar{\bar{\alpha}}_j^{\wedge}(\mathbf{u}) \right)}_{\text{symmetric}} \quad (\text{S11})$$

where M3 is some finite matrix. Since M3 remains finite, then in the limit  $\Delta_j \rightarrow 0$  the eigenvalues of  $(\mathbf{I} - \Delta_j \mathbf{M3})$  must approach 1. Since the determinant of a matrix is the product of its eigenvalues,

$$\det(\mathbf{I} - \Delta_j \mathbf{M3}) > 0 \quad (\text{S12})$$

for sufficiently small finite  $\Delta_j$ . Since the determinant of the product of two matrices is the product of their determinants, then from eqn (S12) it follows that

$$\det(\bar{\bar{\alpha}}_j^{\wedge}(\mathbf{u})(\mathbf{I} - \Delta_j \mathbf{M3})) = \det((\mathbf{I} - \Delta_j \mathbf{M3})\bar{\bar{\alpha}}_j^{\wedge}(\mathbf{u})) = \det(\mathbf{I} - \Delta_j \mathbf{M3})\det(\bar{\bar{\alpha}}_j^{\wedge}(\mathbf{u})) \quad (\text{S13})$$

has the same sign as  $\det(\bar{\bar{\alpha}}_j^{\wedge}(\mathbf{u}))$ . Hence, for sufficiently small finite  $\Delta_j$ , the eigenvalues of

$\bar{\bar{\alpha}}_j^{\wedge}(\mathbf{u})(\mathbf{I} - \Delta_j \mathbf{M3})$  and  $(\mathbf{I} - \Delta_j \mathbf{M3})\bar{\bar{\alpha}}_j^{\wedge}(\mathbf{u})$  must have the same signs as the eigenvalues of

$\bar{\bar{\alpha}}_j^{\wedge}(\mathbf{u})$ . Since  $\det(\mathbf{I} - \Delta_j \mathbf{M3})$  is not close to zero for sufficiently small finite  $\Delta_j$  (eqn (S12)),

then none of the eigenvalues of  $\bar{\bar{\alpha}}_j^{\wedge}(\mathbf{u})(\mathbf{I} - \Delta_j \mathbf{M3})$  and  $(\mathbf{I} - \Delta_j \mathbf{M3})\bar{\bar{\alpha}}_j^{\wedge}(\mathbf{u})$  can be zero if none of the eigenvalues of  $\bar{\bar{\alpha}}_j^{\wedge}(\mathbf{u})$  are zero. By definition, a real-valued matrix M is positive definite if and only if

$$\mathbf{z}^T \mathbf{M} \mathbf{z} > 0 \quad (\text{S14})$$

for every non-zero vector z. A real-valued matrix is positive definite if and only if its symmetric part has all positive eigenvalues.<sup>1</sup> Taken together, the above results show that when  $\bar{\bar{\alpha}}_j^{\wedge}(\mathbf{u})$  is positive definite, then  $\bar{\bar{\alpha}}_j^{\wedge}(\mathbf{u})(\mathbf{I} - \Delta_j \mathbf{M3})$  and  $(\mathbf{I} - \Delta_j \mathbf{M3})\bar{\bar{\alpha}}_j^{\wedge}(\mathbf{u})$  are also positive definite for sufficiently small finite  $\Delta_j$ . The sum of two positive definite matrices  $\mathbf{M} = \mathbf{M}' + \mathbf{M}''$  is always positive definite:

$$\mathbf{z}^T (\mathbf{M}' + \mathbf{M}'') \mathbf{z} = \underbrace{\mathbf{z}^T \mathbf{M}' \mathbf{z}}_{>0} + \underbrace{\mathbf{z}^T \mathbf{M}'' \mathbf{z}}_{>0} > 0 \quad (\text{S15})$$

Since  $\bar{\bar{\alpha}}_{j+1}^{\wedge}(\mathbf{u})$  is one-half the sum of  $\bar{\bar{\alpha}}_j^{\wedge}(\mathbf{u})(\mathbf{I} - \Delta_j \mathbf{M3})$  and  $(\mathbf{I} - \Delta_j \mathbf{M3})\bar{\bar{\alpha}}_j^{\wedge}(\mathbf{u})$ , it follows that

$\bar{\bar{\alpha}}_{j+1}^{\wedge}(\mathbf{u})$  is positive definite if  $\bar{\bar{\alpha}}_j^{\wedge}(\mathbf{u})(\mathbf{I} - \Delta_j \mathbf{M3})$  and  $(\mathbf{I} - \Delta_j \mathbf{M3})\bar{\bar{\alpha}}_j^{\wedge}(\mathbf{u})$  are positive definite.

Therefore,  $\bar{\bar{\alpha}}_{j+1}^{\wedge}(\mathbf{u})$  is positive definite when  $\bar{\bar{\alpha}}_j^{\wedge}(\mathbf{u})$  is positive definite for sufficiently small finite  $\Delta_j$ .

The directional screening process starts with

$$\vec{\vec{\alpha}}_{j=1}^A(\mathbf{u}) = \begin{bmatrix} \alpha_A^{\text{non-dir}}(\mathbf{u}) & 0 & 0 \\ 0 & \alpha_A^{\text{non-dir}}(\mathbf{u}) & 0 \\ 0 & 0 & \alpha_A^{\text{non-dir}}(\mathbf{u}) \end{bmatrix} \quad (\text{S16})$$

and ends with  $\alpha_A^{\text{screened}}(\mathbf{u})$  as one-third the trace of the tensor on the left-side of eqn (S8) after the last screening increment finishes. Therefore,

$$\det(\vec{\vec{\alpha}}_{j=1}^A(\mathbf{u})) = (\alpha_A^{\text{non-dir}}(\mathbf{u}))^3 \geq 0 \quad (\text{S17})$$

and the eigenvalues of  $\vec{\vec{\alpha}}_{j=1}^A(\mathbf{u})$  are  $\alpha_A^{\text{non-dir}}(\mathbf{u}) \geq 0$ . There are two cases. Case 1: If

$\alpha_A^{\text{non-dir}}(\mathbf{u}) = 0$ , then all eigenvalues of  $\vec{\vec{\alpha}}_{j=1}^A(\mathbf{u})$  are zero, and by eqn (S11) the limit  $\Delta_j \rightarrow 0$  gives  $\vec{\vec{\alpha}}_{j+1}^A(\mathbf{u}) = 0$  and hence  $\alpha_A^{\text{screened}}(\mathbf{u}) = 0$ . Case 2: If  $\alpha_A^{\text{non-dir}}(\mathbf{u}) > 0$ , then all eigenvalues of  $\vec{\vec{\alpha}}_{j=1}^A(\mathbf{u})$  are  $> 0$  (i.e.,  $\vec{\vec{\alpha}}_{j=1}^A(\mathbf{u})$  is positive definite), and by the analysis of the prior paragraph,  $\vec{\vec{\alpha}}_{j+1}^A(\mathbf{u})$  is also positive definite for sufficiently small finite  $\Delta_j$ . Since the trace is the sum of eigenvalues, all positive eigenvalues yields positive trace. Thus,

$$\alpha_A^{\text{screened}}(\mathbf{u}) \geq 0 \quad (\text{S18})$$

in all cases. Since  $\alpha_A^{\text{low-freq}} = \alpha_A^{\text{screened}}(\mathbf{u} = \text{Nimfreqs})$ , then  $\alpha_A^{\text{low-freq}} \geq 0$ .

## S2. Derivations and proofs related to the failproof conjugate residual (FCR) algorithm

### S2.1 Problem definition and matrix conditioning

FCR solves any linear equation system with Hermitian coefficients matrix

$$\mathcal{A}\mathbf{x} = \mathcal{B} \quad (\text{S19})$$

for an exact solution (within the convergence tolerance) if one exists or for a conditioned least-squares solution if no exact solution exists. The matrices  $\mathcal{A}$  and  $\mathcal{B}$  may contain a single column or more than one column. Matrix  $\mathcal{A}$  is non-singular if and only if its determinant is nonzero (i.e., all of its eigenvalues are non-zero). In this case, matrix  $\mathcal{A}$  is invertible and the equation  $\mathcal{A}\mathbf{x} = \mathcal{B}$  has the unique solution  $\mathbf{x} = \mathcal{A}^{-1}\mathcal{B}$ . If matrix  $\mathcal{A}$  is singular, then  $\mathcal{A}\mathbf{x} = \mathcal{B}$  has either no solution or an infinite number of solutions. If the linear equation system is consistent (i.e., has at least one solution), this FCR algorithm returns one of its solutions. When matrix  $\mathcal{B}$  has more than one column, the FCR algorithm is applied separately to each column. When  $\mathcal{B} = \mathbf{I}$  (identity matrix), the method solves for  $\mathbf{x} = \mathcal{A}^{-1}$ , the inverse of matrix  $\mathcal{A}$ .

The goal of conditioning is to rotate and scale matrix  $\mathcal{A}$  to improve convergence speed:

$$\mathbf{M} = \mathbf{C}\mathcal{A}\mathbf{C}^H \quad (\text{S20})$$

$$\mathbf{M}\mathbf{y} = \mathbf{C}\mathcal{A}\mathbf{C}^H\mathbf{y} = \mathbf{C}(\mathcal{B} - \mathcal{A}\mathbf{x}_0) = \mathbf{W} \quad (\text{S21})$$

where

$$\mathbf{x} = \mathbf{C}^H\mathbf{y} + \mathbf{x}_0 \quad (\text{S22})$$

Here,  $x_0$  is any initial guess for  $x$ . This shift always makes  $y_0=0$  as the initial guess for  $y$ . The conditioning matrix  $C$  must be non-singular. Since  $A$  is Hermitian, the matrix  $M$  will automatically be Hermitian for any conditioning matrix  $C$ :

$$M^H = (CAC^H)^H = (C^H)^H A^H C^H = CAC^H = M \quad (S23)$$

As shown in Section 2.2 below, the algorithm's convergence speed is largely affected by matrix  $M$ 's eigenspectrum. As with other conjugate gradient methods, convergence is most rapid if the eigenvalues are clustered together.<sup>2</sup> Thus, conditioning's primary goal is to make the eigenspectrum more clustered together.<sup>2</sup>

In this article, the dot product of two vectors  $v$  and  $w$  is defined as

$$\langle v | w \rangle = v^H w = (v^*)^T w = \sum_i v_i^* w_i \quad (S24)$$

A Hermitian matrix  $M = M^H$  can be freely moved between sides of the dot product. For example,

$$\langle MMq | p \rangle = \langle Mq | Mp \rangle = \langle q | MMp \rangle = q^H M^2 p \quad (S25)$$

As commonly known, the Hermitian conjugate of a matrix product follows

$$(\Upsilon\Lambda)^H = \Lambda^H \Upsilon^H \quad (S26)$$

If the linear equation system (i.e., eqn (S19)) is inconsistent (i.e., has no exact solution), this FCR algorithm returns a statement that the linear equation system has no exact solution along with a value  $y = y^{FCR}$  that minimizes the least-squares problem

$$\text{Minimize } F = \langle z | z \rangle = |W - My|^2 \quad (S27)$$

This represents a best possible choice for  $y$  irrespective of whether an exact solution to eqn (S19) exists. There are some applications where this least-squares fit has utility even if an exact solution to eqn (S19) does not exist. (Note: The case where  $My=W$  is consistent arises as the special case of eqn (S27) where the least-squares error is simply zero.)

As commonly defined, the kernel of  $M$  is the set of all  $y$  values that solve  $My=0$ ,

$$\text{kernel}(M) \Rightarrow My = 0 \quad (S28)$$

When  $M$  is non-singular,  $y=0$  is the only vector in the kernel. When  $M$  is singular, the kernel includes  $y=0$  along with an infinite number of non-zero vectors. Manifestly, any vector from the kernel of  $M$  can be added to  $y^{FCR}$  without changing the conditioned residual value. Therefore, the set of all equally good (aka 'best') solutions to eqn (S27) is

$$y^{\text{all}} = y^{\text{FCR}} + \text{span}[\text{kernel}(M)] \quad (S29)$$

If the  $\text{kernel}(M)$  is known, this allows calculating the entire set of  $\{y^{\text{all}}\}$  that give the same minimum value of  $F$ . If  $My=W$  is consistent, this is the set of  $y$  values yielding  $F=0$  (i.e., satisfying eqn (S21)). The number of vectors in  $\{y^{\text{all}}\}$  is one if  $M$  is non-singular; otherwise,  $\{y^{\text{all}}\}$  contains an infinite number of vectors.

## S2.2 Eigenvalue decomposition analysis using Krylov subspaces

### S2.2.1 Mathematical formalism

The method optimizes the conditioned residuals

$$z^{(i)} = W - My^{(i)} = C\Gamma^{(i)} = C(B - Ax^{(i)}) \quad (S30)$$

where enclosed subscripts refer to the iteration number. A Krylov subspace is defined as

$$K_n(M, W) = \text{span}[W, MW, M^2W, M^3W, \dots, M^{n-1}W] \quad (S31)$$

In each iteration,  $y^{(i)}$  is chosen from the Krylov subspace

$$y^{(i)} \in K_{2i}(M, W) \quad (S32)$$

to minimize the norm

$$\Gamma^{(i)} = \langle z^{(i)} | z^{(i)} \rangle \quad (S33)$$

Since  $z^{(0)} = W$ , from eqn (S30) – (S32) it follows that

$$z^{(i)} \in K_{2i+1}(M, W) \quad (S34)$$

Convergence analysis can be obtained by expanding  $W$  in terms of the eigenvectors  $\{\vec{V}_j\}$  of  $M$ :

$$W = \sum_j c_j \vec{V}_j \quad (S35)$$

The building blocks of the Krylov subspace are thus

$$M^k W = \sum_j c_j (\lambda_j)^k \vec{V}_j \quad (S36)$$

where  $\{\lambda_j\}$  are the corresponding eigenvalues of  $M$ . These building blocks can generate at most  $\Xi$  linearly independent vectors, where  $W$  is a linear combination of eigenvectors of  $M$  having exactly  $\Xi$  distinct eigenvalues  $\lambda_j$ .

FCR expands  $y^{(i)}$  as

$$y^{(i)} = a_0 W + a_1 MW + \dots + a_{2i-1} M^{2i-1} W = \sum_j c_j \text{Poly}^{(i)}(\lambda_j) \vec{V}_j \quad (S37)$$

$$My^{(i)} = a_0 MW + a_1 M^2 W + \dots + a_{2i-1} M^{2i} W = \sum_j c_j \lambda_j \text{Poly}^{(i)}(\lambda_j) \vec{V}_j \quad (S38)$$

$$z^{(i)} = W - My^{(i)} = \sum_j c_j \vec{V}_j - \sum_j c_j \lambda_j \text{Poly}^{(i)}(\lambda_j) \vec{V}_j = \sum_j c_j \vec{V}_j (1 - \lambda_j \text{Poly}^{(i)}(\lambda_j)) \quad (S39)$$

where  $\{a_k\}$  are some optimized coefficients and

$$\text{Poly}^{(i)}(\lambda) = \sum_{k=0}^{2i-1} a_k (\lambda)^k \quad (S40)$$

is the associated polynomial. Examining eqn (S39), an exact solution is reached if

$$\text{Poly}^{(i)}(\lambda_j) \Rightarrow 1 / \lambda_j \quad (S41)$$

The FCR convergence properties are thus dictated by the difficulty of representing  $\{1/\lambda_j\}$  via a polynomial  $\text{Poly}^{(i)}(\lambda)$ . When the eigenvalues are close to one, this is trivial because

$$\frac{1}{1 + \underbrace{(\lambda_j - 1)}_{\varepsilon_j}} = 1 - \varepsilon_j + \varepsilon_j^2 - \varepsilon_j^3 + \varepsilon_j^4 \dots \quad (\text{S42})$$

When the eigenvalues are extremely spread out in values, then it takes a higher-order polynomial (and hence a larger number of FCR iterations) to make  $\text{Poly}^{(i)}(\lambda) \approx 1/\lambda$  for all of the eigenvalues contributing to  $W$  (eqn (S35)). Thus, the primary goal of conditioning (eqn (S20)) is to make the eigenvalues less spread out in values.

Combining eqn (S31) and (S35) means the Krylov subspaces can be expanded in terms of eigenvalues as

$$K_n(M, W) = \text{span} \left[ \sum_j c_j \vec{V}_j \sum_{k=0}^{n-1} b_k (\lambda_j)^k \right] \quad (\text{S43})$$

where span refers to the space generated by all possible  $\{b_k\}$  values.

### S2.2.2 Non-singular coefficients matrix

The matrix  $M$  is non-singular if and only if all of its eigenvalues are non-zero:  $\lambda_j \neq 0$  for every  $j$ . Every non-singular matrix  $M$  is invertible. Therefore, the linear equation system  $My=W$  is consistent and has the unique solution:  $y = M^{-1}W$ .

For non-singular matrices in exact arithmetic, FCR converges to the exact solution in at most  $\text{ceiling}(\Xi/2)$  iterations. This follows immediately from eqn (S37) and (S41). Specifically, the exact solution occurs when  $\text{Poly}^{(i)}(\lambda_j) = 1/\lambda_j$  for  $j = 1, 2, \dots, \Xi$  distinct eigenvalues. A polynomial containing  $\Xi$  coefficients (e.g.,  $a_0, a_1, \dots, a_{(\Xi-1)}$ ) can obviously be fit to pass through exactly  $\Xi$  points. By eqn (S37), the polynomial contains  $2i$  coefficients, where  $i$  is the number of FCR iterations. Therefore, in exact arithmetic FCR converges to the exact solution in at most  $\text{ceiling}(\Xi/2)$  iterations. Because the polynomial adds two new coefficients in each FCR iteration, we say each FCR iteration searches 2 independent directions. Note that  $\Xi \leq N_{\text{rows}}$ , where  $N_{\text{rows}}$  is the number of rows in matrix  $M$ . For example, if matrix  $M$  contains  $N_{\text{rows}} = 1$  million, but matrix  $M$  has only 12 distinct eigenvalues, then FCR will converge to the exact result in at most 6 iterations in exact arithmetic.

### S2.2.3 Singular coefficients matrix

Matrix  $M$  is singular if and only if one of its distinct eigenvalues equals zero. Singular matrices are not invertible. Let  $\lambda^\circ = 0$  be the particular eigenvalue that is zero. Let  $c^\circ$  be the coefficient in eqn (S35) associated with this particular eigenvalue. If  $c^\circ \neq 0$ , then let  $V^\circ$  be the eigenvector (associated with zero eigenvalue  $\lambda^\circ$ ) that makes non-zero contribution to  $W$  in eqn (S35).

The eigenvectors of a Hermitian matrix can always be chosen to form an orthonormal basis that spans the whole  $N$ -rows-dimensional space. Consequently, for singular matrices the eigenvector(s) spanning  $\text{kernel}(M)$  are orthogonal to all eigenvectors not in  $\text{kernel}(M)$ .

For singular matrices, there are obviously two separate cases. *Case 1:* When  $c^\circ = 0$ , then the zero-eigenvalue of  $M$  does not contribute anything to the vector  $W$ . In this case, eigenvector(s)  $\{V^\circ\}$  associated with zero eigenvalue do not appear in the Krylov subspace sequence is built-up as  $W, MW, M^2W, \dots$ . Since the conjugate residual method is based on this Krylov subspace sequence, the eigenstate(s)  $\{(\lambda^\circ, V^\circ)\}$  are completely irrelevant to the problem in exact arithmetic, because these zero eigenstates simply do not appear in the equation to be solved nor in its solution. In this case, the linear equation system  $My=W$  is consistent and its solution in exact arithmetic proceeds analogously to the non-singular case.

*Case 2:* When  $c^\circ \neq 0$ , then a zero eigenstate makes non-zero contribution to  $W$  in eqn (S35). Examining eqn (S39), an exact solution would be reached if

$$1 - \lambda_j \text{Poly}^{(i)}(\lambda_j) = 0 \quad (\text{S44})$$

for every eigenstate contributing to  $W$ . For  $\lambda^\circ = 0$ , eqn (S44) simplifies to

$$1 = 0$$

which is manifestly inconsistent. Therefore, the linear equation system  $My=W$  is manifestly inconsistent in this case. Because  $My$  multiplies each eigenvector contributing to  $y$  by its eigenvalue, then  $My$  does not contain any contributions from zero eigenstates. Therefore, whenever  $W$  contains a non-zero contribution from a zero eigenstate, no exact solution exists.

For singular coefficients matrix (whether the linear equation system is consistent or inconsistent), each iteration removes conditioned residual along two  $z$ -search directions that are orthogonal to each other and all prior  $z$ -search directions. Therefore, convergence is reached in at most  $\text{ceiling}(\Xi/2)$  iterations, which is the same result as for the non-singular case given in Section S2.2.2 above.

#### S2.2.4 Deriving the convergence criteria

Regardless of whether the linear equation system is non-singular consistent, singular consistent, or singular inconsistent, the least-squares solution corresponds to removing all contributions to  $W$  that are not in  $\text{kernel}(M)$ , with the final residual being that portion of  $W$  that lies within  $\text{kernel}(M)$ :

$$z^{\min} = W \cap \text{kernel}(M) \quad (\text{S45})$$

$$\Gamma^{\min} = \langle z^{\min} | z^{\min} \rangle = \min |W - My|^2 \quad (\text{S46})$$

Examining eqn (S39), this FCR method always proceeds normally along all  $z$ -search directions that are not in  $\text{kernel}(M)$ . Because these  $z$ -search directions are a linear combination of eigenstates that are orthogonal to  $\text{kernel}(M)$ , projection of the conditioned residual onto  $\text{kernel}(M)$  is not effected during this search.

Manifestly,  $\langle \mathbf{Mz}^{(i)} | \mathbf{Mz}^{(i)} \rangle = 0$  if only if  $\mathbf{z}^{(i)} \in \text{kernel}(\mathbf{M})$ , in which case  $\mathbf{z}^{(i)} = \mathbf{z}^{\min}$  and the algorithm is converged. Therefore, a needed condition to construct a fail-proof conjugate residual method is that one of the z-search directions (say  $\mathbf{Mp}^{(i)}$ ) is chosen such that

$$\langle \mathbf{Mp}^{(i)} | \mathbf{z}^{(i-1)} \rangle = \langle \mathbf{Mz}^{(i-1)} | \mathbf{Mz}^{(i-1)} \rangle \geq 0 \quad (\text{S47})$$

so that z-search always reduces the conditioned residual norm in each and every iteration until convergence. If  $\langle \mathbf{Mp}^{(i)} | \mathbf{Mp}^{(i)} \rangle = 0$  then  $\mathbf{p}^{(i)} \in \text{kernel}(\mathbf{M})$  and hence

$$\langle \mathbf{Mp}^{(i)} | \mathbf{z}^{(i-1)} \rangle = \langle \mathbf{Mz}^{(i-1)} | \mathbf{Mz}^{(i-1)} \rangle = 0. \text{ Therefore, if } \langle \mathbf{Mz}^{(i-1)} | \mathbf{Mz}^{(i-1)} \rangle > 0 \text{ then } \langle \mathbf{Mp}^{(i)} | \mathbf{Mp}^{(i)} \rangle > 0.$$

Therefore, the FCR algorithm has two convergence criteria. If  $\mathbf{z}^{(i)} \Rightarrow 0$ , the FCR algorithm converged to an exact solution (within the convergence tolerance). If  $\mathbf{z}^{(i)} \not\approx 0$  but  $\langle \mathbf{Mz}^{(i)} | \mathbf{Mz}^{(i)} \rangle \Rightarrow 0$ , then the linear equation system is inconsistent and the FCR algorithm converged to the least-squares solution (within the convergence tolerance).

### S2.3 Defining conjugate search directions that span the Krylov subspaces

Each iteration involves two search directions,  $\mathbf{p}^{(i)}$  and  $\mathbf{q}^{(i)}$ :

$$\mathbf{y}^{(i)} = \mathbf{y}^{(i-1)} + \gamma^{(i)}\mathbf{p}^{(i)} + \tau^{(i)}\mathbf{q}^{(i)} \quad (\text{S48})$$

$$\mathbf{z}^{(i)} = \mathbf{z}^{(i-1)} - \gamma^{(i)}\mathbf{Mp}^{(i)} - \tau^{(i)}\mathbf{Mq}^{(i)} \quad (\text{S49})$$

subject to the constraints

$$\mathbf{p}^{(i)}, \mathbf{q}^{(i)} \in \mathbf{K}_{2i}(\mathbf{M}, \mathbf{W}) \quad (\text{S50})$$

$$\mathbf{K}_{2i}(\mathbf{M}, \mathbf{W}) = \text{span}[\mathbf{p}^{(1)}, \mathbf{q}^{(1)}, \dots, \mathbf{p}^{(i)}, \mathbf{q}^{(i)}] \quad (\text{S51})$$

In exact arithmetic, all z-search directions are chosen to be orthogonal, thus ensuring a conjugate method:

$$\langle \mathbf{Mq}^{(i)} | \mathbf{Mq}^{(j \neq i)} \rangle = \langle \mathbf{Mp}^{(i)} | \mathbf{Mq}^{(j)} \rangle = \langle \mathbf{Mp}^{(i)} | \mathbf{Mp}^{(j \neq i)} \rangle = 0 \quad (\text{S52})$$

The conditioned residual component along z-search directions  $\mathbf{Mp}^{(i)}$  and  $\mathbf{Mq}^{(i)}$  are removed in iteration  $i$ . Since all other z-search directions are orthogonal to  $\mathbf{Mp}^{(i)}$  and  $\mathbf{Mq}^{(i)}$ , no conditioned residual (except for round-off errors) is subsequently contaminated along these directions. Hence,

$$\langle \mathbf{Mq}^{(i)} | \mathbf{z}^{(j \geq i)} \rangle = \langle \mathbf{Mp}^{(i)} | \mathbf{z}^{(j \geq i)} \rangle = 0 \quad (\text{S53})$$

Let  $\mathbf{S}^{(i)}$  be the new z-search space for iteration  $i$ . Then,

$$\mathbf{S}^{(i)} = \text{span}[\mathbf{Mp}^{(i)}, \mathbf{Mq}^{(i)}] \quad (\text{S54})$$

Following eqn (S53) and (S54), the conditioned residual is perpendicular to the current and prior search subspaces:

$$\mathbf{z}^{(j \geq i)} \perp \text{span}[\mathbf{S}^{(1)}, \mathbf{S}^{(2)}, \dots, \mathbf{S}^{(i)}] \quad (\text{S55})$$

Because two conjugate search directions are chosen from a Krylov subspace that increases by two directions per iteration (and are orthogonal to all prior z-search directions), it follows that

$$\text{span}\left[S^{(1)}, S^{(2)}, \dots, S^{(i+k)}\right] = \text{span}\left[MW, M^2W, \dots, M^{2i+2k}W\right] \quad (\text{S56})$$

$$M^{2k>0}p^{(i)}, M^{1+2k>0}p^{(i)}, M^{2k>0}q^{(i)}, M^{1+2k>0}q^{(i)} \in \text{span}\left[S^{(1)}, S^{(2)}, \dots, S^{(i+k)}\right] \quad (\text{S57})$$

Similarly,

$$Mz^{(i)}, M^2z^{(i)} \in \text{span}\left[S^{(1)}, S^{(2)}, \dots, S^{(i+1)}\right] \quad (\text{S58})$$

Thus,

$$\left\langle z^{(j \geq i+1)} \middle| Mz^{(i)} \right\rangle = \left\langle z^{(j \geq i+1)} \middle| M^2z^{(i)} \right\rangle = 0 \quad (\text{S59})$$

Combining eqn (S55) and (S57) yields:

$$\left\langle z^{(j \geq i+k)} \middle| M^{2k>0}p^{(i)} \right\rangle = \left\langle z^{(j \geq i+k)} \middle| M^{1+2k>0}p^{(i)} \right\rangle = \left\langle z^{(j \geq i+k)} \middle| M^{2k>0}q^{(i)} \right\rangle = \left\langle z^{(j \geq i+k)} \middle| M^{1+2k>0}q^{(i)} \right\rangle = 0 \quad (\text{S60})$$

Combining eqn (S49), (S52), and (S53) yields

$$\gamma^{(i)} = \frac{\left\langle Mp^{(i)} \middle| z^{(i-1)} \right\rangle}{\left\langle Mp^{(i)} \middle| Mp^{(i)} \right\rangle} \quad (\text{S61})$$

$$\tau^{(i)} = \frac{\left\langle Mq^{(i)} \middle| z^{(i-1)} \right\rangle}{\left\langle Mq^{(i)} \middle| Mq^{(i)} \right\rangle} \quad (\text{S62})$$

As stated above in eqn (S51) – (S52),  $p^{(i)}, q^{(i)}$  are  $M^2$ -conjugate to all prior search directions that span the Krylov subspaces of orders  $\leq 2i-2$ . Since

$$M^{j \geq 0}(p, q)^{(i)} \in K_{2i+j} \quad (\text{S63})$$

it follows that for  $j \geq 0$ :

$$\left\langle (p, q)^{(k > i + (j/2))} \middle| M^{2+j}(p, q)^{(i)} \right\rangle = 0 \quad (\text{S64})$$

where  $(p, q)$  means the equation holds for both  $p$  and  $q$  (i.e.,  $M^{j \geq 0}(p, q)^{(i)}$  means it holds for  $M^{j \geq 0}p^{(i)}$  and  $M^{j \geq 0}q^{(i)}$ ). These equations are for exact arithmetic.

These requirements will be fulfilled by choosing

$$p^{(i)} = Mz^{(i-1)} - \sum_{j=1}^{i-1} \beta_{i-j}^{(i)} p^{(j)} - \sum_{j=1}^{i-1} \xi_{i-j}^{(i)} q^{(j)} \quad (\text{S65})$$

First, we derive the conjugacy conditions. Contracting eqn (S65) with  $M^2p^{(k < i)}$  gives

$$\left\langle M^2p^{(k)} \middle| p^{(i)} \right\rangle = \left\langle M^2p^{(k)} \middle| Mz^{(i-1)} \right\rangle - \sum_{j=1}^{i-1} \beta_{i-j}^{(i)} \left\langle M^2p^{(k)} \middle| p^{(j)} \right\rangle - \sum_{j=1}^{i-1} \xi_{i-j}^{(i)} \underbrace{\left\langle M^2p^{(k)} \middle| q^{(j)} \right\rangle}_0 \Rightarrow 0 \quad (\text{S66})$$

which must equal zero to satisfy eqn (S52). By eqn (S60),

$$\left\langle M^2p^{(k < i-1)} \middle| Mz^{(i-1)} \right\rangle = \left\langle M^2q^{(k < i-1)} \middle| Mz^{(i-1)} \right\rangle = 0 \quad (\text{S67})$$

which gives

$$\beta_{n \geq 2}^{(i)} = 0 \quad (\text{S68})$$

$$\beta^{(i)} = \beta_1^{(i)} = \frac{\langle \mathbf{M}^2 \mathbf{p}^{(i-1)} | \mathbf{Mz}^{(i-1)} \rangle}{\langle \mathbf{Mp}^{(i-1)} | \mathbf{Mp}^{(i-1)} \rangle} \quad (\text{S69})$$

Since  $\beta_1^{(i)}$  is the only non-zero component, for convenience the subscript is dropped. Contracting eqn (S65) with  $\mathbf{M}^2 \mathbf{q}^{(k-i)}$  gives

$$\langle \mathbf{M}^2 \mathbf{q}^{(k)} | \mathbf{p}^{(i)} \rangle = \langle \mathbf{M}^2 \mathbf{q}^{(k)} | \mathbf{Mz}^{(i-1)} \rangle - \sum_{j=1}^{i-1} \beta_{i-j}^{(i)} \underbrace{\langle \mathbf{M}^2 \mathbf{q}^{(k)} | \mathbf{p}^{(j)} \rangle}_0 - \sum_{j=1}^{i-1} \xi_{i-j}^{(i)} \langle \mathbf{M}^2 \mathbf{q}^{(k)} | \mathbf{q}^{(j)} \rangle \Rightarrow 0 \quad (\text{S70})$$

Making use of eqn (S67) gives

$$\xi_{n \geq 2}^{(i)} = 0 \quad (\text{S71})$$

$$\xi^{(i)} = \xi_1^{(i)} = \frac{\langle \mathbf{M}^2 \mathbf{q}^{(i-1)} | \mathbf{Mz}^{(i-1)} \rangle}{\langle \mathbf{Mq}^{(i-1)} | \mathbf{Mq}^{(i-1)} \rangle} \quad (\text{S72})$$

Hence,

$$\mathbf{p}^{(i)} = \mathbf{Mz}^{(i-1)} - \beta^{(i)} \mathbf{p}^{(i-1)} - \xi^{(i)} \mathbf{q}^{(i-1)} \quad (\text{S73})$$

This choice for  $\mathbf{p}^{(i)}$  satisfies the requirement in eqn (S47) above:

$$\langle \mathbf{p}^{(i)} | \mathbf{Mz}^{(i-1)} \rangle = \langle \mathbf{Mz}^{(i-1)} | \mathbf{Mz}^{(i-1)} \rangle - \beta^{(i)} \langle \mathbf{p}^{(i-1)} | \mathbf{Mz}^{(i-1)} \rangle - \xi^{(i)} \langle \mathbf{q}^{(i-1)} | \mathbf{Mz}^{(i-1)} \rangle \quad (\text{S74})$$

which rearranges to give

$$\langle \mathbf{Mp}^{(i)} | \mathbf{z}^{(i-1)} \rangle = \langle \mathbf{Mz}^{(i-1)} | \mathbf{Mz}^{(i-1)} \rangle - \beta^{(i)} \underbrace{\langle \mathbf{Mp}^{(i-1)} | \mathbf{z}^{(i-1)} \rangle}_0 - \xi^{(i)} \underbrace{\langle \mathbf{Mq}^{(i-1)} | \mathbf{z}^{(i-1)} \rangle}_0 = \langle \mathbf{Mz}^{(i-1)} | \mathbf{Mz}^{(i-1)} \rangle \quad (\text{S75})$$

Next, a method is derived to compute  $\mathbf{q}^{(i)}$ . Since

$$\mathbf{K}_{2i}(\mathbf{M}, \mathbf{W}) = \text{span}[\mathbf{W}, \mathbf{MW}, \mathbf{M}^2 \mathbf{p}^{(i)}, \mathbf{M}^2 \mathbf{q}^{(i)}, \dots, \mathbf{M}^2 \mathbf{p}^{(i-1)}, \mathbf{M}^2 \mathbf{q}^{(i-1)}] \quad (\text{S76})$$

it follows that  $\mathbf{M}^2 \mathbf{q}^{(i-1)}$  and  $\mathbf{M}^2 \mathbf{p}^{(i-1)}$  are linearly independent of  $\mathbf{K}_{2i-2}(\mathbf{M}, \mathbf{W})$  and each other (unless the FCR algorithm has converged). Together with eqn (S56)–(S57) this means that  $\mathbf{M}^3 \mathbf{p}^{(i-1)}$  and  $\mathbf{M}^3 \mathbf{q}^{(i-1)}$  have non-zero projections onto  $\mathbf{S}^{(i)}$ . Since  $\mathbf{Mp}^{(i)}$  and  $\mathbf{Mq}^{(i)}$  form a basis for  $\mathbf{S}^{(i)}$  (eqn (S54)), we can construct the vector  $\mathbf{Mq}^{(i)}$  by first constructing a linear combination of  $\mathbf{M}^3 \mathbf{p}^{(i-1)}$  and  $\mathbf{M}^3 \mathbf{q}^{(i-1)}$  that is orthogonal to  $\mathbf{Mp}^{(i)}$ , and then subtracting all projections onto the prior search spaces  $\mathbf{S}^{(j \leq i)}$  (which are spanned by  $\mathbf{Mp}^{(j \leq i)}$  and  $\mathbf{Mq}^{(j \leq i)}$ ). By eqn (S64),

$$\langle \mathbf{M}^3(\mathbf{p}, \mathbf{q})^{(i-1)} | \mathbf{M}(\mathbf{p}, \mathbf{q})^{(h < i-2)} \rangle = \langle (\mathbf{p}, \mathbf{q})^{(i-1)} | \mathbf{M}^4(\mathbf{p}, \mathbf{q})^{(h < i-2)} \rangle = 0 \quad (\text{S77})$$

Therefore, projections onto  $\mathbf{S}^{(h < i-2)}$  are automatically zero in exact arithmetic. The following ordered sequence of assignment operations gives  $\mathbf{q}^{(i)}$  with these required properties. Assigning

$$\kappa_1^{(i)} = -\frac{\langle M^2 p^{(i)} | M^2 q^{(i-1)} \rangle}{\sqrt{\langle M^2 p^{(i)} | M^2 q^{(i-1)} \rangle^2 + \langle M^2 p^{(i)} | M^2 p^{(i-1)} \rangle^2}} \quad (\text{S78})$$

$$\kappa_2^{(i)} = \frac{\langle M^2 p^{(i)} | M^2 p^{(i-1)} \rangle}{\sqrt{\langle M^2 p^{(i)} | M^2 q^{(i-1)} \rangle^2 + \langle M^2 p^{(i)} | M^2 p^{(i-1)} \rangle^2}} \quad (\text{S79})$$

$$q^{(i)} \leftarrow \kappa_1^{(i)} M^2 p^{(i-1)} + \kappa_2^{(i)} M^2 q^{(i-1)} \quad (\text{S80})$$

yields a provisional  $q^{(i)}$  that is  $M^2$ -conjugate to  $p^{(i)}$ . Assigning

$$\chi_2^{(i)} = \frac{\langle M^2 p^{(i-2)} | q^{(i)} \rangle}{\langle M p^{(i-2)} | M p^{(i-2)} \rangle} \quad (\text{S81})$$

$$\sigma_2^{(i)} = \frac{\langle M^2 q^{(i-2)} | q^{(i)} \rangle}{\langle M q^{(i-2)} | M q^{(i-2)} \rangle} \quad (\text{S82})$$

$$q^{(i)} \leftarrow q^{(i)} - \chi_2^{(i)} p^{(i-2)} - \sigma_2^{(i)} q^{(i-2)} \quad (\text{S83})$$

yields a provisional  $q^{(i)}$  that is  $M^2$ -conjugate to  $p^{(i-2)}$  and  $p^{(i-2)}$  in exact arithmetic. Assigning

$$\chi_1^{(i)} = \frac{\langle M^2 p^{(i-1)} | q^{(i)} \rangle}{\langle M p^{(i-1)} | M p^{(i-1)} \rangle} \quad (\text{S84})$$

$$\sigma_1^{(i)} = \frac{\langle M^2 q^{(i-1)} | q^{(i)} \rangle}{\langle M q^{(i-1)} | M q^{(i-1)} \rangle} \quad (\text{S85})$$

$$q^{(i)} \leftarrow q^{(i)} - \chi_1^{(i)} p^{(i-1)} - \sigma_1^{(i)} q^{(i-1)} \quad (\text{S86})$$

yields a provisional  $q^{(i)}$  that is  $M^2$ -conjugate to  $p^{(i-1)}$  and  $p^{(i-1)}$ . Assigning

$$\chi_0^{(i)} = \frac{\langle M^2 p^{(i)} | q^{(i)} \rangle}{\langle M p^{(i)} | M p^{(i)} \rangle} \quad (\text{S87})$$

$$q^{(i)} \leftarrow q^{(i)} - \chi_0^{(i)} p^{(i)} \quad (\text{S88})$$

yields non-provisional  $q^{(i)}$  that is  $M^2$ -conjugate to  $p^{(i)}$ .

The algorithm is initialized with

$$p^{(i \leq 0)} = q^{(i \leq 0)} = 0 \quad (\text{S89})$$

$$p^{(1)} = Mz^{(0)} = MW \quad (\text{S90})$$

$$q^{(1)} = z^{(0)} - \frac{\langle MW | M^2 W \rangle}{\langle M p^{(1)} | M p^{(1)} \rangle} p^{(1)} \quad (\text{S91})$$

This choice satisfies the conjugacy condition:

$$\begin{aligned}
\langle \mathbf{M}\mathbf{p}^{(i)} | \mathbf{M}\mathbf{q}^{(i)} \rangle &= \left\langle \mathbf{M}\mathbf{p}^{(i)} \left| \mathbf{M}\mathbf{z}^{(0)} - \frac{\langle \mathbf{M}\mathbf{W} | \mathbf{M}^2\mathbf{W} \rangle}{\langle \mathbf{M}\mathbf{p}^{(i)} | \mathbf{M}\mathbf{p}^{(i)} \rangle} \mathbf{M}\mathbf{p}^{(i)} \right. \right\rangle \\
&= \left\langle \mathbf{M}^2\mathbf{W} \left| \mathbf{M}\mathbf{W} - \frac{\langle \mathbf{M}\mathbf{W} | \mathbf{M}^2\mathbf{W} \rangle}{\langle \mathbf{M}^2\mathbf{W} | \mathbf{M}^2\mathbf{W} \rangle} \mathbf{M}^2\mathbf{W} \right. \right\rangle \\
&= \langle \mathbf{M}^2\mathbf{W} | \mathbf{M}\mathbf{W} \rangle - \langle \mathbf{M}\mathbf{W} | \mathbf{M}^2\mathbf{W} \rangle \\
&= 0
\end{aligned} \tag{S92}$$

#### S2.4 Vector lengths in exact arithmetic

The above relationships prove the z-search directions are orthogonal and members of the appropriate Krylov subspaces, but additional analysis is required to prove the search directions have non-zero lengths (at least until convergence). If  $\mathbf{p}^{(i)} = 0$ ,  $\mathbf{M}\mathbf{p}^{(i)} = 0$ ,  $\mathbf{q}^{(i)} = 0$ , or  $\mathbf{M}\mathbf{q}^{(i)} = 0$  occur on any iteration except the last one, then the correct Krylov subspaces may not be spanned by these vectors. Note that a vector  $\mathfrak{g}$  equals zero if and only if its squared length ( $\langle \mathfrak{g} | \mathfrak{g} \rangle$ ) is also zero. Also, the squared length of any vector is non-negative:

$$\langle \mathfrak{g} | \mathfrak{g} \rangle \geq 0 \tag{S93}$$

Here is a proof that  $\mathbf{p}^{(i)} \neq 0$  until convergence is reached. From eqn (S53),

$$\langle \mathbf{M}\mathbf{z}^{(i-1)} | \mathbf{p}^{(i-1)} \rangle = \langle \mathbf{M}\mathbf{z}^{(i-1)} | \mathbf{q}^{(i-1)} \rangle = 0 \tag{S94}$$

Thus, taking the dot product of eqn (S65) with itself yields

$$\langle \mathbf{p}^{(i)} | \mathbf{p}^{(i)} \rangle = \langle \mathbf{M}\mathbf{z}^{(i-1)} | \mathbf{M}\mathbf{z}^{(i-1)} \rangle + \langle \beta^{(i)} \mathbf{p}^{(i-1)} + \xi^{(i)} \mathbf{q}^{(i-1)} | \beta^{(i)} \mathbf{p}^{(i-1)} + \xi^{(i)} \mathbf{q}^{(i-1)} \rangle > 0 \tag{S95}$$

From eqn (S93), the two terms on the right side of eqn (S95) must be non-negative. As mentioned in Section S2.2.4 above, if  $\langle \mathbf{M}\mathbf{z}^{(i-1)} | \mathbf{M}\mathbf{z}^{(i-1)} \rangle \Rightarrow 0$  the FCR algorithm would have already exited in iteration (i-1) due to convergence. Thus, during the FCR algorithm,

$$\langle \mathbf{p}^{(i)} | \mathbf{p}^{(i)} \rangle > 0 \tag{S96}$$

It was already proved in Section S2.2.4 above that if  $\langle \mathbf{M}\mathbf{z}^{(i-1)} | \mathbf{M}\mathbf{z}^{(i-1)} \rangle > 0$ , then

$\langle \mathbf{M}\mathbf{p}^{(i)} | \mathbf{M}\mathbf{p}^{(i)} \rangle > 0$ . Therefore,  $\mathbf{M}\mathbf{p}^{(i)} \neq 0$  during the FCR algorithm.

The condition under which  $\mathbf{q}^{(i)} = 0$  could occur is now derived. Examining eqn (S78) – (S88),  $\mathbf{q}^{(i)}$  is a linear combination of  $\mathbf{M}^2\mathbf{p}^{(i-1)}$ ,  $\mathbf{M}^2\mathbf{q}^{(i-1)}$ ,  $\mathbf{p}^{(i-1)}$ ,  $\mathbf{q}^{(i-1)}$ ,  $\mathbf{p}^{(i-2)}$ , and  $\mathbf{q}^{(i-2)}$  that fulfills the conjugacy relations in eqn (S52). (Note that in exact arithmetic,  $\chi_0^{(i)} = 0$  because eqn (S78) – (S80) already enforce  $\langle \mathbf{M}\mathbf{p}^{(i)} | \mathbf{M}\mathbf{q}^{(i)} \rangle = 0$ ) The condition  $\mathbf{q}^{(i)} = 0$  would imply that  $\text{span}[\mathbf{M}^2\mathbf{p}^{(i-1)}, \mathbf{M}^2\mathbf{q}^{(i-1)}] \subset \text{span}[\mathbf{p}^{j \leq i}, \mathbf{q}^{j < i}]$ . This could only occur if

$\text{span}\left[\mathbf{p}^{j \leq i}, \mathbf{q}^{j < i}, \mathbf{M}^2 \mathbf{p}^{(i-1)}, \mathbf{M}^2 \mathbf{q}^{(i-1)}\right]$  contains fewer than  $2i$  linearly independent vectors. Combining eqn (S51) and (S76) gives

$$\text{span}\left[\mathbf{p}^{j \leq i}, \mathbf{q}^{j < i}, \mathbf{M}^2 \mathbf{p}^{(i-1)}, \mathbf{M}^2 \mathbf{q}^{(i-1)}\right] = \mathbf{K}_{2i}(\mathbf{M}, \mathbf{W}) \quad (\text{S97})$$

Therefore  $\mathbf{K}_{2i}(\mathbf{M}, \mathbf{W})$  contains  $\leq (2i - 1)$  linearly independent terms if  $\mathbf{q}^{(i)} = 0$ . From eqn (S43), this can only mean there are  $\Xi \leq 2i - 1$  distinct eigenvalues having non-zero contributions to  $\mathbf{W}$ . As derived in Sections S2.2.2 and S2.2.3 above, FCR converges in at most  $\text{ceiling}(\Xi / 2)$  iterations. Thus, FCR converges in at most  $\text{ceiling}((2i-1)/2) = i$  iterations when  $\mathbf{q}^{(i)} = 0$ . Therefore,  $\mathbf{q}^{(i)} = 0$  can occur only on the last iteration during which FCR converges.

The condition  $\mathbf{M}\mathbf{q}^{(i)} = 0$  could only occur if  $\mathbf{q}^{(i)} \in \text{kernel}(\mathbf{M})$ . This can occur in the  $\mathbf{q}^{(i)} = 0$  case discussed in the prior paragraph (i.e., on the last iteration during which FCR converges). It may be possible for this condition to also arise during other iterations. The algorithm includes a division by zero check to handle cases where  $\langle \mathbf{M}\mathbf{q}^{(i)} | \mathbf{M}\mathbf{q}^{(i)} \rangle = 0$ . When  $\langle \mathbf{M}\mathbf{q}^{(i)} | \mathbf{M}\mathbf{q}^{(i)} \rangle = 0$  the required  $\mathbf{M}^2$ -conjugacy between a vector  $\mathfrak{G}$  and  $\mathbf{q}^{(i)}$  is automatically satisfied:  $\langle \mathbf{M}\mathfrak{G} | \mathbf{M}\mathbf{q}^{(i)} = 0 \rangle = 0$ . Therefore, the corresponding parameter  $\tau^{(i)}$  (eqn (S62)),  $\xi^{(i)}$  (eqn (S72)),  $\kappa_1^{(i)}$  (eqn (S78)),  $\kappa_2^{(i)}$  (eqn (S79)),  $\sigma_1^{(i)}$  (eqn (S85)), or  $\sigma_2^{(i)}$  (eqn (S82)) is set to zero when its corresponding denominator would be zero.

### S2.5 Robust convergence and round-off error resistance

The Orthodir conjugate gradient algorithm can work for both positive definite and indefinite Hermitian coefficients matrix  $\mathbf{M}$ .<sup>3</sup> However, it suffers from the accumulation of round-off errors. In the Orthodir algorithm, the  $y$ -search direction at each successive iteration is computed as

$$\mathbf{p}^{(i)} = \mathbf{M}\mathbf{p}^{(i-1)} - \zeta^{(i)} \mathbf{p}^{(i-1)} - \mathfrak{G}^{(i)} \mathbf{p}^{(i-2)} \quad (\text{S98})$$

where  $\zeta^{(i)}$  and  $\mathfrak{G}^{(i)}$  are chosen to fulfill some chosen conjugacy condition.<sup>3</sup> In exact arithmetic, eqn (S98) would enforce orthogonality between  $\mathbf{M}\mathbf{p}^{(i)}$ ,  $\mathbf{M}\mathbf{p}^{(i-1)}$ , and  $\mathbf{M}\mathbf{p}^{(i-2)}$ . Because the choice of direction  $\mathbf{p}^{(i)}$  does not explicitly depend on the residual's value (see eqn (S98)), a buildup of round-off errors over many iterations can cause the chosen  $z$ -search direction  $\mathbf{M}\mathbf{p}^{(i)}$  to become uncorrelated to the residual's value. When this occurs, the Orthodir algorithm is does not operate as intended and may fail to converge.

The FCR algorithm solves this problem by making the new search direction  $\mathbf{p}^{(i)}$  depend explicitly on the residual's current value (eqn (S73)). Thus, irrespective of any round-off errors that have occurred, this search direction will never become uncoupled from the residual's value. Round-off errors can potentially cause the  $\mathbf{q}^{(i)}$  direction to stray from its precise value.

FCR explicitly enforces orthogonality between  $\mathbf{M}_p^{(i)}$ ,  $\mathbf{M}_q^{(i)}$ ,  $\mathbf{M}_p^{(i-1)}$ , and  $\mathbf{M}_q^{(i-1)}$  in each iteration regardless of any round-off errors that have occurred. Eqn (S87)–(S88) ensure that

$$\langle \mathbf{M}_q^{(i)} | \mathbf{M}_p^{(i)} \rangle = 0 \quad (\text{S99})$$

even if round-off errors occurred during prior computations. With this result, eqn (S84)–(S86) and eqn (S65), (S69), and (S72) ensure that

$$\langle \mathbf{M}_q^{(i)} | \mathbf{M}_p^{(i-1)} \rangle = \langle \mathbf{M}_q^{(i)} | \mathbf{M}_q^{(i-1)} \rangle = 0 \quad (\text{S100})$$

$$\langle \mathbf{M}_p^{(i)} | \mathbf{M}_p^{(i-1)} \rangle = \langle \mathbf{M}_p^{(i)} | \mathbf{M}_q^{(i-1)} \rangle = 0 \quad (\text{S101})$$

even if round-off errors occurred during prior computations.  $\mathbf{M}^2$ -conjugacy between  $(p, q)^{(i)}$  and  $(p, q)^{(h < i-1)}$  may be compromised by round-off errors.

Irrespective of any round-off errors that occurred during prior computations, eqn (S99), (S61), and (S62) ensure that

$$\langle \mathbf{M}_q^{(i)} | \mathbf{z}^{(i)} \rangle = \langle \mathbf{M}_p^{(i)} | \mathbf{z}^{(i)} \rangle = 0 \quad (\text{S102})$$

In other words, during iteration  $i$  the conditioned residual components along  $\mathbf{z}$ -search directions  $\mathbf{M}_p^{(i)}$  and  $\mathbf{M}_q^{(i)}$  are completely removed regardless of any round-off errors that occurred during prior computations.

It will now be proved  $\Gamma^{(i)}$  necessarily decreases in each iteration until FCR converges, even when using fixed precision arithmetic. Rearranging eqn (S49) gives

$$\mathbf{z}^{(i-1)} = \mathbf{z}^{(i)} + \gamma^{(i)} \mathbf{M}_p^{(i)} + \tau^{(i)} \mathbf{M}_q^{(i)} \quad (\text{S103})$$

Taking the dot product of eqn (S103) with itself gives

$$\begin{aligned} \langle \mathbf{z}^{(i-1)} | \mathbf{z}^{(i-1)} \rangle &= \langle \mathbf{z}^{(i)} | \mathbf{z}^{(i)} \rangle + |\gamma^{(i)}|^2 \langle \mathbf{M}_p^{(i)} | \mathbf{M}_p^{(i)} \rangle + |\tau^{(i)}|^2 \langle \mathbf{M}_q^{(i)} | \mathbf{M}_q^{(i)} \rangle \\ &+ \underbrace{\gamma^{(i)} \langle \mathbf{z}^{(i)} | \mathbf{M}_p^{(i)} \rangle}_0 + \underbrace{\tau^{(i)} \langle \mathbf{z}^{(i)} | \mathbf{M}_q^{(i)} \rangle}_0 + \underbrace{(\gamma^{(i)*} \tau^{(i)}) \langle \mathbf{M}_p^{(i)} | \mathbf{M}_q^{(i)} \rangle}_0 \\ &+ \underbrace{(\gamma^{(i)*}) \langle \mathbf{M}_p^{(i)} | \mathbf{z}^{(i)} \rangle}_0 + \underbrace{(\tau^{(i)*}) \langle \mathbf{M}_q^{(i)} | \mathbf{z}^{(i)} \rangle}_0 + \underbrace{(\tau^{(i)*}) \gamma^{(i)} \langle \mathbf{M}_q^{(i)} | \mathbf{M}_p^{(i)} \rangle}_0 \end{aligned} \quad (\text{S104})$$

Hence,

$$\langle \mathbf{z}^{(i-1)} | \mathbf{z}^{(i-1)} \rangle - \langle \mathbf{z}^{(i)} | \mathbf{z}^{(i)} \rangle = |\gamma^{(i)}|^2 \langle \mathbf{M}_p^{(i)} | \mathbf{M}_p^{(i)} \rangle + |\tau^{(i)}|^2 \langle \mathbf{M}_q^{(i)} | \mathbf{M}_q^{(i)} \rangle \quad (\text{S105})$$

Substituting eqn (S33), (S61), and (S62) into (S105) gives

$$\Gamma^{(i)} = \Gamma^{(i-1)} - \frac{|\langle \mathbf{M}_p^{(i)} | \mathbf{z}^{(i-1)} \rangle|^2}{\langle \mathbf{M}_p^{(i)} | \mathbf{M}_p^{(i)} \rangle} - \frac{|\langle \mathbf{M}_q^{(i)} | \mathbf{z}^{(i-1)} \rangle|^2}{\langle \mathbf{M}_q^{(i)} | \mathbf{M}_q^{(i)} \rangle} \quad (\text{S106})$$

Expanding  $\mathbf{p}^{(i)}$  using eqn (S73) gives

$$\langle \mathbf{M}\mathbf{p}^{(i)} | \mathbf{z}^{(i-1)} \rangle = \langle \mathbf{M}\mathbf{z}^{(i-1)} | \mathbf{M}\mathbf{z}^{(i-1)} \rangle - \underbrace{(\beta^{(i)} *)}_{0} \langle \mathbf{M}\mathbf{p}^{(i-1)} | \mathbf{z}^{(i-1)} \rangle - \underbrace{(\xi^{(i)} *)}_{0} \langle \mathbf{M}\mathbf{q}^{(i-1)} | \mathbf{z}^{(i-1)} \rangle \quad (\text{S107})$$

where the quantities marked zero via eqn (S53). Substituting eqn (S107) into (S106) gives

$$\Gamma^{(i-1)} - \Gamma^{(i)} = \frac{\langle \mathbf{M}\mathbf{z}^{(i-1)} | \mathbf{M}\mathbf{z}^{(i-1)} \rangle^2}{\langle \mathbf{M}\mathbf{p}^{(i)} | \mathbf{M}\mathbf{p}^{(i)} \rangle} + \frac{|\langle \mathbf{M}\mathbf{q}^{(i)} | \mathbf{z}^{(i-1)} \rangle|^2}{\langle \mathbf{M}\mathbf{q}^{(i)} | \mathbf{M}\mathbf{q}^{(i)} \rangle} > 0 \quad (\text{S108})$$

As discussed in Section S2.2.4 above, FCR is not yet converged when  $\langle \mathbf{M}\mathbf{z}^{(i-1)} | \mathbf{M}\mathbf{z}^{(i-1)} \rangle > 0$ .

Inserting this into eqn (S108) gives  $\Gamma^{(i-1)} - \Gamma^{(i)} > 0$  when FCR is not yet converged. Hence, the FCR algorithm always decreases the conditioned residual norm in every iteration until convergence. Because the required conjugacy conditions (eqn (S99) – (S102)) hold even for fixed precision arithmetic, eqn (S108) remains valid for fixed precision arithmetic.

## S2.6 Computational algorithm

Convergence thresholds: The following three parameters control convergence thresholds.

(1) FCR will exit if the maximum absolute value of every conditioned residual component is less than CR\_convergence\_tol. (2) FCR will exit if SQRT( $\langle \mathbf{M}\mathbf{z} | \mathbf{M}\mathbf{z} \rangle$ ) is less than Mz\_length\_tolerance. The division\_tolerance = (Mz\_length\_tolerance<sup>2</sup>) sets the threshold for when  $\langle \mathbf{M}\mathbf{q} | \mathbf{M}\mathbf{q} \rangle$  and related denominators are considered to be effectively zero.

Mz\_length\_tolerance should be set to a value much smaller than CR\_convergence\_tol. (3) max\_CR\_steps specifies the maximum number of FCR iterations that can be performed.

Description of key variables to declare: (a) scalars: CR\_gamma, CR\_tau, CR\_beta, CR\_xi, CR\_z\_z, CR\_Mz\_Mz, CR\_Mp\_z, CR\_Mq\_z, CR\_Mp\_Mq, temp1, temp2, temp3, (b) vectors with a small number of rows and one column: CR\_kappa(2 rows), CR\_chi(2 rows), CR\_sigma(2 rows), CR\_Mp\_Mp(2 rows), CR\_Mq\_Mq(2 rows), (c) vectors with N rows and one column: W, y, z, Mz, Mp, Mq, (d) arrays with N rows and 3 columns: p, MMp, q, MMq. If both M and W are real-valued, all of these will be declared as real variables; otherwise, they will be declared as complex variables.

Initiation: The FCR algorithm is initialized as follows:

1. Initialize  $y=0$ ,  $z = W$ ,  $p(:, :) = 0$ ,  $q(:, :) = 0$ ,  $\text{MMp}(:, :) = 0$ ,  $\text{MMq}(:, :) = 0$
2. Compute  $\text{CR}_z_z = \langle z | z \rangle$  and the maximum absolute value of any z-component. If the maximum absolute value of any z-component is less than CR\_convergence\_tol, print the following message “The linear equation system is consistent. A nearly exact solution has been returned.” and exit.
3. Compute  $\text{Mz} = \mathbf{M} \times z$
4. Compute  $\text{CR}_Mz_Mz = \langle \text{Mz} | \text{Mz} \rangle$ . If  $\text{SQRT}(\text{CR}_Mz_Mz) < \text{Mz\_length\_tolerance}$ , print the following message “The linear equation system is inconsistent. A least squares solution has been returned.” and exit.
5. Initialize  $p(:, 1) = \text{Mz}(:, :)$
6. Compute  $\text{Mp} = \mathbf{M} \times p(:, 1)$

7. Compute  $CR\_Mp\_Mp(1) = \langle Mp \mid Mp \rangle$  and  $temp1 = \langle p(:,1) \mid Mp \rangle$ .
8. Initialize  $q(:,1) = z(:) - (temp1/CR\_Mp\_Mp(1)) \times p(:,1)$
9. Initialize  $Mq = Mz - (temp1/CR\_Mp\_Mp(1)) \times Mp$
10. Compute  $MMp(:,1) = M \times Mp$  and  $MMq(:,1) = M \times Mq$
11. Initialize  $CR\_Mq\_Mq(:) = 1$ ,  $CR\_Mp\_Mp(2) = 1$ ,  $CR\_kappa(:) = 0$

Iterations: Each FCR iteration follows the following sequence of steps:

1. Set  $CR\_Mq\_Mq(2) = CR\_Mq\_Mq(1)$ . Then compute  $CR\_Mq\_Mq(1) = \langle Mq \mid Mq \rangle$ ,  $CR\_Mp\_z = \langle Mp \mid z \rangle$ ,  $CR\_Mq\_z = \langle Mq \mid z \rangle$ , and  $CR\_Mz\_Mz = \langle Mz \mid Mz \rangle$ . Note that  $CR\_Mz\_Mz$  does not need to be used by the method, but is optionally printed along with  $CR\_Mp\_z$  as a consistency check to show round-off errors are suppressed since  $CR\_Mz\_Mz = CR\_Mp\_z$  in exact arithmetic.
2. Compute  $CR\_gamma = CR\_Mp\_z/CR\_Mp\_Mp(1)$ . IF  $(CR\_Mq\_Mq(1) > division\_tolerance)$  THEN  $CR\_tau = CR\_Mq\_z/CR\_Mq\_Mq(1)$ , ELSE  $CR\_tau = 0$ .
3. Compute  $y(:) = y(:) + CR\_gamma \times p(:,1) + CR\_tau \times q(:,1)$ ,  $z(:) = z(:) - CR\_gamma \times Mp(:) - CR\_tau \times Mq(:)$ , and  $Mz(:) = Mz(:) - CR\_gamma \times MMp(:,1) - CR\_tau \times MMq(:,1)$ .
4. Compute the conditioned residual norm  $CR\_z\_z = \langle z \mid z \rangle$  and the maximum absolute value of any z-component. If desired, these can be printed. If the maximum absolute value of z-components is less than  $CR\_convergence\_tol$ , print the following message "The linear equation system is consistent. A nearly exact solution has been returned." and exit.
5. Compute  $CR\_Mz\_Mz = \langle Mz \mid Mz \rangle$ . If  $SQRT(CR\_Mz\_Mz) < Mz\_length\_tolerance$ , print the following message "The linear equation system is inconsistent. A least squares solution has been returned." and exit.
6. *Optional printing of conjugacy tests:* If desired, the following quantities can be recomputed and printed:  $CR\_Mp\_z = \langle Mp \mid z \rangle$ ,  $CR\_Mq\_z = \langle Mq \mid z \rangle$ , and  $CR\_Mp\_Mq = \langle Mp \mid Mq \rangle$ . When the conjugacy conditions are satisfied, all three of these quantities should be zero. If desired,  $CR\_Mz\_Mz$  and  $CR\_Mp\_z$  can be printed; these should equal each other.
7. Shift the p and q vector related quantities to store them as prior iteration:  $p(:,3) = p(:,2)$ ,  $q(:,3) = q(:,2)$ ,  $MMp(:,3) = MMp(:,2)$ ,  $MMq(:,3) = MMq(:,2)$ ,  $p(:,2) = p(:,1)$ ,  $q(:,2) = q(:,1)$ ,  $MMp(:,2) = MMp(:,1)$ ,  $MMq(:,2) = MMq(:,1)$
8. Compute  $temp1 = \langle MMp(:,2) \mid Mz \rangle$ ,  $temp2 = \langle MMq(:,2) \mid Mz \rangle$ . Then compute  $CR\_beta = temp1/CR\_Mp\_Mp(1)$ . IF  $(CR\_Mq\_Mq(1) > division\_tolerance)$  THEN  $CR\_xi = temp2/CR\_Mq\_Mq(1)$ , ELSE  $CR\_xi = 0$ .
9. Compute  $p(:,1) = Mz(:) - CR\_beta \times p(:,2) - CR\_xi \times q(:,2)$
10. Compute  $Mp = M \times p(:,1)$
11. Compute  $MMp(:,1) = M \times Mp$

12. Use the following sequence to compute CR\_kappa. First, compute temp1 = - < MMp(:,1) | MMq(:,2) > and temp2 = < MMp(:,1) | MMp(:,2) >. Then compute temp3 = sqrt((temp1)×CONJG(temp1) + (temp2)×CONJG(temp2)). (Note: The function CONJG returns the complex conjugate.) IF ( temp3 > division\_tolerance) THEN CR\_kappa(1) = temp1/temp3 and CR\_kappa(2) = temp2/temp3, ELSE CR\_kappa(:) = 0. Then compute q(:,1) = CR\_kappa(1)×MMp(:,2) + CR\_kappa(2)×MMq(:,2).
13. Compute temp1 = < MMp(:,3) | q(:,1) > and temp2 = < MMq(:,3) | q(:,1) >. Then compute CR\_chi(2)= temp1/CR\_Mp\_Mp(2). IF (CR\_Mq\_Mq(2) > division\_tolerance) THEN CR\_sigma(2)= temp2/CR\_Mq\_Mq(2), ELSE CR\_sigma(2) = 0. Then compute q(:,1) = q(:,1) – CR\_chi(2)×p(:,3) – CR\_sigma(2)×q(:,3).
14. Compute temp1 = < MMp(:,2) | q(:,1) > and temp2 = < MMq(:,2) | q(:,1) >. Then compute CR\_chi(1)= temp1/CR\_Mp\_Mp(1). IF (CR\_Mq\_Mq(1) > division\_tolerance) THEN CR\_sigma(1)= temp2/CR\_Mq\_Mq(1), ELSE CR\_sigma(1) = 0. Then compute q(:,1) = q(:,1) – CR\_chi(1)×p(:,2) – CR\_sigma(1)×q(:,2).
15. Set CR\_Mp\_Mp(2) = CR\_Mp\_Mp(1). Then compute temp1 = < MMp(:,1) | q(:,1) > and CR\_Mp\_Mp(1) = < Mp | Mp >.
16. Compute q(:,1) = q(:,1) – temp1/CR\_Mp\_Mp(1)×p(:,1)
17. Compute Mq = M×q(:,1)
18. Compute MMq(:,1) = M×Mq
19. IF the current FCR iteration number equals max\_CR\_steps, print the following message “Maximum number of FCR cycles reached. The current estimate of the solution has been returned.” and exit; ELSE return to step 1 above to start the next FCR cycle.

### S3. Proof the wp lookup table method yields error proportional to the interval squared

As explained in the main text, a wp lookup table is constructed on a logarithmic scale:

$$x_1 = \ln(\min(\text{wp\_values})) - 10^{-2} \quad (\text{S109})$$

$$x_{\text{Num\_lookup}} = \ln(\max(\text{wp\_values})) + 10^{-2} \quad (\text{S110})$$

$$\text{interval} = x_i - x_{i-1} = \frac{x_{\text{Num\_lookup}} - x_1}{\text{Num\_lookup} - 1} \quad (\text{S111})$$

$$\text{wp\_table}_i = e^{x_i} \quad (\text{S112})$$

Each atom A’s polarizability  $\alpha_A$  contributes to two adjacent array values j and (j+1), where

$$j = \text{floor}\left(\frac{\ln(\text{wp\_values}_A) - x_1}{\text{interval}} + 1\right) \quad (\text{S113})$$

$$c_{j+1} = \frac{\text{wp\_values}_A - \text{wp\_table}_j}{\text{wp\_table}_{j+1} - \text{wp\_table}_j} \quad (\text{S114})$$

$$c_j = 1 - c_{j+1} \quad (\text{S115})$$

$$\text{alpha\_table}_j \leftarrow \text{alpha\_table}_j + c_j \alpha_A \quad (\text{S116})$$

$$\alpha\_table_{j+1} \leftarrow \alpha\_table_{j+1} + c_{j+1} \alpha_A \quad (S117)$$

$$C_6^{\text{total}} = \sum_{i=1}^{\text{Num\_lookup}} \sum_{k=1}^{\text{Natoms}} \left[ \frac{3}{2} (\alpha\_table_i) (\alpha_k) \frac{\text{wp\_table}_i \text{wp\_values}_k}{\text{wp\_table}_i + \text{wp\_values}_k} \right] \quad (S118)$$

Observe that

$$0 \leq c_j, c_{j+1} \leq 1 \quad (S119)$$

Under practical applications,

$$0.01 \leq \text{wp\_values}_A \leq 20 \quad (S120)$$

in atomic units. Because Num\_lookup=10<sup>5</sup> is much larger than ((ln(max(wp\_values)) + 10<sup>-2</sup>) – (ln(min(wp\_values)) – 10<sup>-2</sup>)), it follows that

$$0 < \text{interval} \ll 1 \quad (S121)$$

Defining

$$q = \text{wp\_values}_A / \text{wp\_values}_B \quad (S122)$$

gives

$$C_{6,AB} = \frac{3}{2} \alpha_A \alpha_B \frac{\text{wp\_values}_A \text{wp\_values}_B}{\text{wp\_values}_A + \text{wp\_values}_B} = \frac{3}{2} \alpha_A \alpha_B \frac{\text{wp\_values}_A}{q+1} \quad (S123)$$

For k = B in the inner loop of eqn (S118), this lookup table method corresponds to the following

$$C_{6,AB} \Rightarrow \frac{3}{2} \alpha_A \alpha_B \left[ c_j \frac{\text{wp\_table}_j \text{wp\_values}_B}{\text{wp\_table}_j + \text{wp\_values}_B} + c_{j+1} \frac{\text{wp\_table}_{j+1} \text{wp\_values}_B}{\text{wp\_table}_{j+1} + \text{wp\_values}_B} \right] \quad (S124)$$

Rearranging eqn (S114) gives

$$c_{j+1} = \frac{\text{wp\_values}_A - \text{wp\_table}_j}{\text{wp\_table}_j (e^{\text{interval}} - 1)} \quad (S125)$$

which further rearranges to give

$$\frac{\text{wp\_values}_A}{\text{wp\_table}_j} = 1 + c_{j+1} (e^{\text{interval}} - 1) \quad (S126)$$

$$\frac{\text{wp\_values}_A}{\text{wp\_table}_{j+1}} = \frac{\text{wp\_values}_A}{\text{wp\_table}_j e^{\text{interval}}} = \left[ 1 + c_{j+1} (e^{\text{interval}} - 1) \right] e^{-\text{interval}} = c_{j+1} + (1 - c_{j+1}) e^{-\text{interval}} \quad (S127)$$

Expanding

$$\frac{\text{wp\_table}_j \text{wp\_values}_B}{\text{wp\_table}_j + \text{wp\_values}_B} = \frac{\text{wp\_values}_A}{\frac{\text{wp\_values}_A}{\text{wp\_values}_B} + \frac{\text{wp\_values}_A}{\text{wp\_table}_j}} = \frac{\text{wp\_values}_A}{q + (1 + c_{j+1} (e^{\text{interval}} - 1))} \quad (S128)$$

$$\frac{\text{wp\_table}_{j+1} \text{wp\_values}_B}{\text{wp\_table}_{j+1} + \text{wp\_values}_B} = \frac{\text{wp\_values}_A}{\frac{\text{wp\_values}_A}{\text{wp\_values}_B} + \frac{\text{wp\_values}_A}{\text{wp\_table}_{j+1}}} = \frac{\text{wp\_values}_A}{q + (c_{j+1} + (1 - c_{j+1}) e^{-\text{interval}})} \quad (S129)$$

$$\frac{wp\_values_A wp\_values_B}{wp\_values_A + wp\_values_B} = \frac{wp\_values_A}{\left(\frac{wp\_values_A}{wp\_values_B}\right) + 1} = \frac{wp\_values_A}{q + 1} \tag{S130}$$

Hence, the relative error in computed  $C_{6,AB}$  is

$$\phi = \frac{\frac{3}{2} \alpha_A \alpha_B \left[ c_j \frac{wp\_table_j wp\_values_B}{wp\_table_j + wp\_values_B} + c_{j+1} \frac{wp\_table_{j+1} wp\_values_B}{wp\_table_{j+1} + wp\_values_B} \right]}{\frac{3}{2} \alpha_A \alpha_B \frac{wp\_values_A wp\_values_B}{wp\_values_A + wp\_values_B}} - 1 \tag{S131}$$

which expands as

$$\phi = (1 - c_{j+1}) \frac{q + 1}{q + (1 + c_{j+1}(e^{interval} - 1))} + c_{j+1} \frac{q + 1}{q + (c_{j+1} + (1 - c_{j+1})e^{-interval})} - 1 \tag{S132}$$

Using Mathcad’s symbolic engine, this was expanded to a Taylor series in powers of interval:

$$\phi = \frac{-(interval^2)c_{j+1}(1 - c_{j+1})q}{(1 + q)^2} + Order(interval^3) + \dots \tag{S133}$$

This formula lends itself to extremely easy magnitude analysis. Over the interval  $0 \leq c_{j+1} \leq 1$ , locating the maximum value of function  $c_{j+1}(1 - c_{j+1})$  using the standard technique of setting the function’s derivative to zero yields

$$0 \leq c_{j+1}(1 - c_{j+1}) \leq 1/4 \tag{S134}$$

For  $q \geq 0$ , a similar analysis for the function  $q/(1+q)^2$  yields

$$0 \leq \frac{q}{(1 + q)^2} \leq \frac{1}{4} \tag{S135}$$

Inserting eqn (S134) and (S135) into (S133) yields

$$|\phi| \leq \frac{interval^2}{16} \tag{S136}$$

where the third-order and higher terms can be neglected, because  $0 < interval \ll 1$ .

Because  $C_6^{total}$  is the sum of *positive*  $C_{6,AB}$  values, the unsigned relative error (URE) of  $C_6^{total}$  is necessarily less than or equal to that of the largest URE of  $C_{6,AB}$ . Therefore, the URE in  $C_6^{total}$  is bounded by

$$\frac{|C_6^{total,calc} - C_6^{total,exact}|}{C_6^{total,exact}} \leq \frac{interval^2}{16} \leq \frac{\sim 4}{Num\_lookup^2} \tag{S137}$$

It follows from eqns (S120) and (S121) that

$$\ln\left(\frac{\max(wp\_values)}{\min(wp\_values)}\right) \leq \underbrace{\left(x_{Num\_lookup} - x_1\right)}_{(interval)(Num\_lookup-1)} < \sim 8 \tag{S138}$$

Substituting eqn (S138) into the center of eqn (S137) yields the right-most inequality of eqn (S137). For  $\text{Num\_lookup} = 10^5$ , a  $\text{URE} \leq \sim 4 \times 10^{-10}$  is expected.

## S4. Linear-scaling algorithm for setting up the lists of interacting atom pairs

### S4.1 Overview

Figure S1 is a flow diagram of the linear-scaling algorithm to set up the lists of interacting atom pairs. Data is grouped to enable fast computation by avoiding all array searches. For example, information is ordered such that arrays do not have to be searched to identify which atoms belong in each spatial region. Also, each array allocation is performed once, rather than continuously appending arrays (which would be extremely slow). This is accomplished by first performing a 'dry run' code block that executes a sequence to count up the required array size, followed by array allocation, followed by a code block that writes data to the allocated array. The four key steps to construct these lists are:

1. Define basis vectors and unit cell parallelepiped: A parallelepiped of non-zero volume is constructed to enclose the system's unit cell. Three basis vectors correspond to this parallelepiped's non-collinear edges. For a periodic direction, the basis vector is the corresponding periodic lattice vector. For a non-periodic direction, the basis vector is chosen to be of a non-zero length that fully encloses all the nuclear positions. Periodic basis vectors can be non-perpendicular to each other (e.g., triclinic unit cells), but each non-periodic basis vector is chosen to be perpendicular to the other basis vectors.
2. Divide the unit cell parallelepiped into spatial regions: This unit cell parallelepiped is divided into a whole number of spatial regions along each basis vector. A periodic direction produces an infinite number of periodic images of each region, while a non-periodic direction has only the reference image. Atoms in the reference unit cell are classified by region, and a sorted list is prepared such that atoms of the same region are adjacent in this sorted list. Because each spatial region is defined such that its volume is less than that of a sphere of dipole interaction cutoff length radius, the number of atoms in each region is always below a threshold. The code ignores regions that do not contain any atoms. Because empty regions are skipped, having a few atoms in the center of an enormous unit cell would execute quickly.
3. Construct arrays listing interacting region pair images: Two spatial region images interact if the minimum distance between inter-region points is  $\leq$  the dipole interaction cutoff length. Because the spatial regions and their images are coordinate system indexed, a list of interacting region pair images is constructed without having to construct a double summation over all region pairs. Thus, even for an extremely large unit cell (e.g., containing billions of atoms) divided into many regions (e.g., millions), the list of interacting region pair images is constructed in time and memory scaling linearly with increasing unit cell size. Different regions can interact with different periodic images. For example, in an extremely large unit cell, a region near the center would interact only with nearby regions in the reference unit cell, while a region not too

far from the left edge would interact with some regions in the reference unit cell and some other region images in the left-translated unit cell. Thus, a first array is constructed listing pairs of regions having any interacting images, and a second array is prepared that lists which specific images of each particular region pair interact. Region pairs that do not interact are not included in these two arrays.

4. Construct two lists of interacting atom pairs: Because interacting atom pairs must be contained in interacting region pair images, the code identifies the interacting atom pairs by executing an outer loop over the interacting region pair images and inner loops over the atoms in these regions (along with tests for inclusion criteria). Using the list of atoms sorted by region makes this process cache access friendly. For each such atom pair, tests are performed to determine if it meets the small and large list inclusion criteria. If so, its information is added to the small and/or large lists. Because the number of interacting region pair images scales linearly with large  $N_{\text{atoms}}$  and the number of atoms in each region is below a threshold, these small and large lists are constructed in linear-scaling computational time and memory for large  $N_{\text{atoms}}$ .

The zip archive included in the Electronic Supplementary Information contains a folder named “construct\_lists\_of\_interacting\_atom\_pairs”. This folder contains example Fortran code that implements the method described here. Readers can examine this for finer coding details. This code is in the form of modules that should be called by a main program (not included). The main program must supply the required input data (e.g., number of atoms, coordinates for each atom, unit cell information, etc).

The following Sections S4.2, S4.3, S4.4, and S4.5 explain important mathematical equations and procedural details for the four key steps to construct the lists of interacting atom pairs.

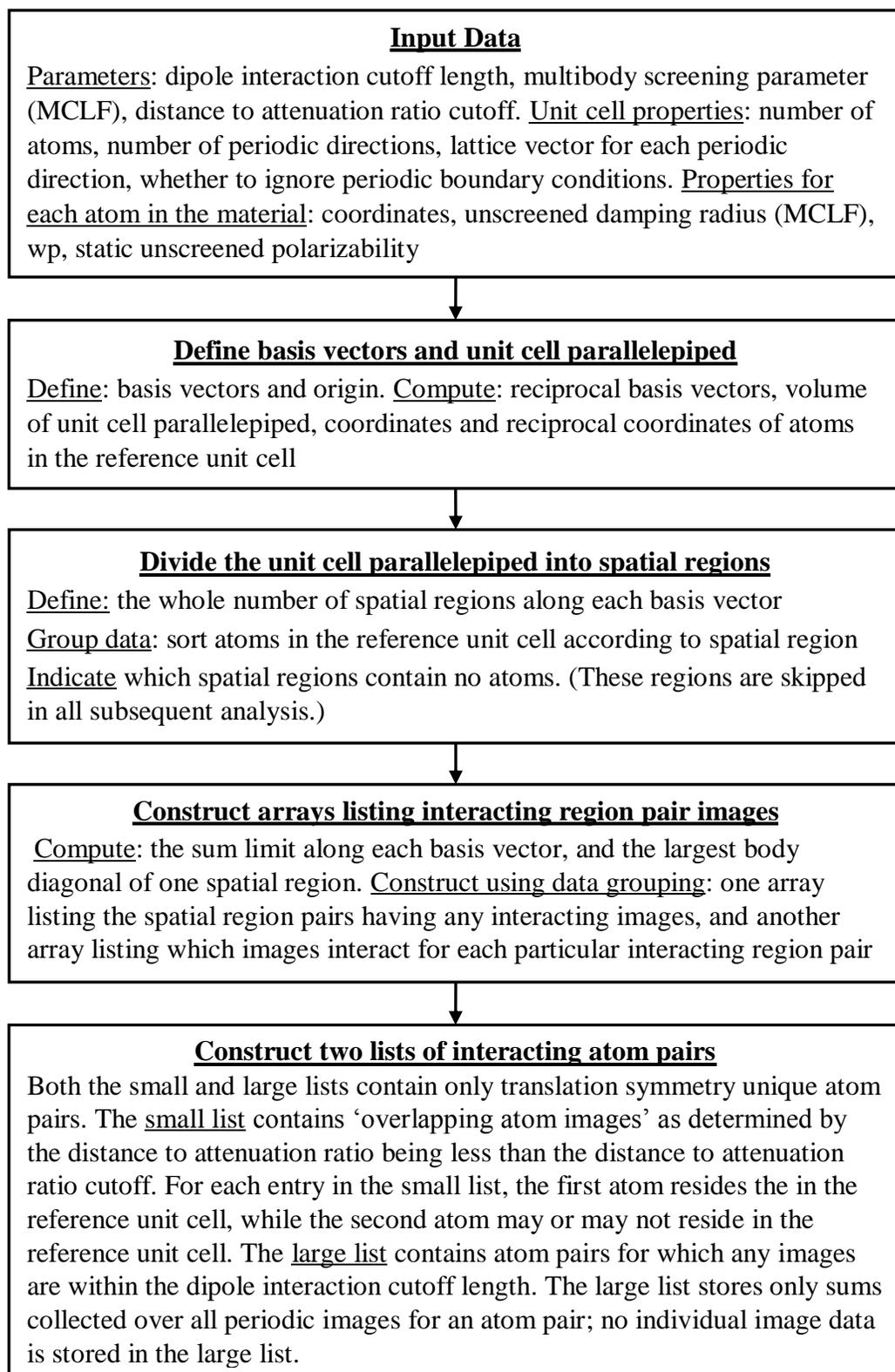


Figure S1: Flow diagram for setting up the lists of interacting atom pairs. The inputs followed by (MCLF) are needed only for the MCLF method and not the TS-SCS method.

## S4.2 Define basis vectors and unit cell parallelepiped

A parallelepiped of non-zero volume is constructed to enclose the system's unit cell. Three basis vectors ( $\vec{v}^{(A)}, \vec{v}^{(B)}, \vec{v}^{(C)}$ ) correspond to this parallelepiped's non-collinear edges. For a periodic direction, the basis vector is the corresponding periodic lattice vector. For a non-periodic direction, the basis vector is chosen to be of a non-zero length that fully encloses all the nuclear positions. Periodic basis vectors can be non-perpendicular to each other (e.g., triclinic unit cells), but each non-periodic basis vector is chosen to be perpendicular to the other basis vectors. Following is a brief description of how the basis vectors and origin are chosen for different numbers of periodic boundary conditions:

0. No periodic boundary conditions: The minimum and maximum values of atomic XYZ coordinates are computed in atomic units (bohrs). The three basis vectors are chosen as:  $\text{basis\_vector\_a} = \text{max\_X} - \text{min\_X} + 1$ ,  $\text{basis\_vector\_b} = \text{max\_Y} - \text{min\_Y} + 1$ , and  $\text{basis\_vector\_c} = \text{max\_Z} - \text{min\_Z} + 1$ . (This definition ensures each basis vector has a length  $\geq 1$  even for single atoms, linear molecules, and planar molecules.) The origin is set to  $\text{origin} = (\text{min\_X} - 1/2, \text{min\_Y} - 1/2, \text{min\_Z} - 1/2)$  to ensure the reciprocal coordinates are  $> 0$  even in the presence of rounding errors.
1. One periodic boundary condition: One basis vector is set equal to the periodic lattice vector. The two other basis vectors are defined to be perpendicular to this lattice vector and to each other. Two pre-basis vectors of unit lengths are constructed parallel to the two non-periodic directions. For each of the two non-periodic basis vectors, the following process is performed separately to determine its length: *A loop over atoms is performed. In each iteration of this loop, the dot product is constructed from the current atom's coordinates and the corresponding pre-basis vector. The minimum and maximum dot products across the set of all atoms is computed. The length of this basis vector (in atomic units) is then set to the  $\text{max\_dot\_product} - \text{min\_dot\_product} + 1$ .* Then the origin is defined such that the minimum  $\text{DOT\_PRODUCT}((\text{coords}(\text{atom1},:) - \text{origin}), \text{nonperiodic\_basis\_vector\_1}) = 1/2$  and minimum  $\text{DOT\_PRODUCT}((\text{coords}(\text{atom1},:) - \text{origin}), \text{nonperiodic\_basis\_vector\_2}) = 1/2$ , where the minimum is computed over the set of all atoms.
2. Two periodic boundary conditions: Two basis vectors are set equal to the periodic lattice vectors. The third basis vector is defined to be parallel to the cross product of the two periodic lattice vectors. A pre-basis vector of unit length is formed along this direction. Then, a loop over atoms is performed. In each iteration of this loop, the dot product is constructed from the current atom's coordinates and this pre-basis vector. The minimum and maximum dot products across the set of all atoms is computed. The length of this basis vector (in atomic units) is then set to the  $\text{max\_dot\_product} - \text{min\_dot\_product} + 1$ . Then the origin is defined such that the minimum  $\text{DOT\_PRODUCT}(((\text{coords}(\text{atom1},:) - \text{origin}), \text{nonperiodic\_basis\_vector})) = 1/2$ , where the minimum is computed over the set of all atoms.

3. Three periodic boundary conditions: The basis vectors are set equal to the periodic lattice vectors. The origin is set to (0,0,0).

Define the matrix of basis vectors as

$$\text{basis\_vectors} = \begin{pmatrix} \mathbf{v}_1^{(A)} & \mathbf{v}_2^{(A)} & \mathbf{v}_3^{(A)} \\ \mathbf{v}_1^{(B)} & \mathbf{v}_2^{(B)} & \mathbf{v}_3^{(B)} \\ \mathbf{v}_1^{(C)} & \mathbf{v}_2^{(C)} & \mathbf{v}_3^{(C)} \end{pmatrix} \quad (\text{S139})$$

Then, the unit cell volume is the absolute value of the determinant:

$$\text{unit\_cell\_volume} = \text{abs}(\det(\text{basis\_vectors})) \quad (\text{S140})$$

The reciprocal basis vectors are the transposed inverse:

$$\text{inverse\_vectors} = \left( (\text{basis\_vectors})^T \right)^{-1} \quad (\text{S141})$$

The reciprocal space coordinates for each atom1 are then given by

$$\text{inverse\_coords}(:, \text{atom1}) = \text{matmul}(\text{inverse\_vectors}, (\text{coords}(:, \text{atom1}) - \text{origin})) \quad (\text{S142})$$

where matmul is the matrix multiplication function. Then a modulo is applied to select the image of atom A that resides within the reference unit cell:

$$\text{inverse\_coords}(:, \text{atom1}) = \text{modulo}(\text{inverse\_coords}(:, \text{atom1}), 1.0) \quad (\text{S143})$$

This ensures the reciprocal space coordinates lie within the interval ( [0,1), [0,1), [0,1) ).

### S4.3 Divide the unit cell parallelepiped into spatial regions

First, a few geometric factors are computed from the unit cell parallelepiped:

$$\text{length\_a} = \sqrt{\text{DOT\_PRODUCT}(\vec{\mathbf{v}}^{(A)}, \vec{\mathbf{v}}^{(A)})} \quad (\text{S144})$$

$$\cos\_a\_b = \frac{\text{DOT\_PRODUCT}(\vec{\mathbf{v}}^{(A)}, \vec{\mathbf{v}}^{(B)})}{(\text{length\_a})(\text{length\_b})} \quad (\text{S145})$$

$$\sin\_a\_b = \sqrt{1 - (\cos\_a\_b)^2} \quad (\text{S146})$$

and so forth for the other basis vectors.

$$\text{height\_a} = \frac{\text{unit\_cell\_volume}}{\text{length\_b} \cdot \text{length\_c} \cdot \sin\_b\_c} \quad (\text{S147})$$

is the positive projection of  $\vec{\mathbf{v}}^{(A)}$  onto the direction perpendicular to the parallelogram formed by  $\vec{\mathbf{v}}^{(B)}$  and  $\vec{\mathbf{v}}^{(C)}$ . Note that  $(\text{length\_b} \cdot \text{length\_c} \cdot \sin\_b\_c)$  is the area of this parallelogram.

Analogously,

$$\text{height\_b} = \frac{\text{unit\_cell\_volume}}{\text{length\_a} \cdot \text{length\_c} \cdot \sin\_a\_c} \quad (\text{S148})$$

$$\text{height\_c} = \frac{\text{unit\_cell\_volume}}{\text{length\_a} \cdot \text{length\_b} \cdot \sin\_a\_b} \quad (\text{S149})$$

Then, the whole number of regions along each basis vector is calculated as:

$$\text{ref\_regions\_a} = \text{ceiling} \left( \left( \frac{3}{\text{dipole\_interaction\_cutoff\_length}} \right) \text{height\_a} \right) \quad (\text{S150})$$

$$\text{ref\_regions\_b} = \text{ceiling} \left( \left( \left( \frac{3}{\text{dipole\_interaction\_cutoff\_length}} \right) \text{height\_b} \right) \right) \quad (\text{S151})$$

$$\text{ref\_regions\_c} = \text{ceiling} \left( \left( \left( \frac{3}{\text{dipole\_interaction\_cutoff\_length}} \right) \text{height\_c} \right) \right) \quad (\text{S152})$$

The ceiling function rounds up to an integer. The factor  $(3/\text{dipole\_interaction\_cutoff\_length})$  sets the preferred length of a region to approximately one-third the dipole interaction cutoff length. This guarantees the size of each region is always substantially smaller than the dipole interaction cutoff length while also not creating an unnecessarily large number of regions.

During a loop performed over atoms, each atom is assigned to a spatial region as labeled by three whole number indices in the interval  $([1, \text{ref\_regions\_a}], [1, \text{ref\_regions\_b}], [1, \text{ref\_regions\_c}])$ . Along the first basis vector, the corresponding region index for atom1 is

$$\text{region\_a} = \text{NINT}(\text{ref\_regions\_a} \cdot \text{inverse\_coords}(1, \text{atom1}) + 0.5) \quad (\text{S153})$$

with analogous expressions for the second and third basis vectors, where NINT rounds to the closest integer (0.5 being rounded up). If any region does not contain any atoms, this region is marked as empty and skipped in all subsequent analysis.

Finally, data grouping is performed to prepare a list of atoms and their coordinates such that all atoms from the same region are grouped together in this list. The starting and ending index values for each region in this list is computed and stored. This makes it extremely easy to perform loops over all atoms in a particular region without having to do any array searches.

#### S4.4 Construct arrays listing interacting region pair images

When constructing a list of representative interacting region pair images, the first region can always be chosen within the reference unit cell. Since there are no periodic translations along a non-periodic direction, then for a non-periodic direction the second region in the interacting region pair image also always lies within the reference unit cell. Along a periodic direction, the second region in the interacting region pair image can be a periodically translated image of some region such that this image has some points located closer than  $\text{dipole\_interaction\_cutoff\_length}$  of some points in the first region. Therefore,  $\text{periodicsumlimitA}$ ,  $\text{periodicsumlimitB}$ , and  $\text{periodicsumlimitC}$  are defined such that only region images with integer translation indices

$$-\text{periodicsumlimitA} \leq L_1 \leq \text{periodicsumlimitA} \quad (\text{S154})$$

$$-\text{periodicsumlimitB} \leq L_2 \leq \text{periodicsumlimitB} \quad (\text{S155})$$

$$-\text{periodicsumlimitC} \leq L_3 \leq \text{periodicsumlimitC} \quad (\text{S156})$$

need to be tested when setting up the list of interacting region pair images. For a non-periodic direction, the corresponding periodic sum limit is obviously set to zero. If  $\text{ignore\_PBC} = \text{true}$ , then

periodic boundary conditions are ignored and all periodic sum limits are set to zero. For a periodic direction, the corresponding periodic sum limit is set to the corresponding one of the following:

$$\text{periodicsumlimitA} = \text{ceiling} \left( \frac{\text{dipole\_interaction\_cutoff\_length}}{\text{height\_a}} \right) \quad (\text{S157})$$

$$\text{periodicsumlimitB} = \text{ceiling} \left( \frac{\text{dipole\_interaction\_cutoff\_length}}{\text{height\_b}} \right) \quad (\text{S158})$$

$$\text{periodicsumlimitC} = \text{ceiling} \left( \frac{\text{dipole\_interaction\_cutoff\_length}}{\text{height\_c}} \right) \quad (\text{S159})$$

The longest body diagonal (aka largest\_body\_diag) of one spatial region is computed by finding the maximum real-space distance corresponding to a change of  $(\text{ref\_regions\_a})^{-1}$ ,  $\pm(\text{ref\_regions\_b})^{-1}$ ,  $\pm(\text{ref\_regions\_c})^{-1}$  in reciprocal space coordinates. The real-space distance along these diagonals is:

$$\text{distance} = \sqrt{\left( \frac{v_1^{(A)}}{\text{ref\_regions\_a}} \pm \frac{v_1^{(B)}}{\text{ref\_regions\_b}} \pm' \frac{v_1^{(C)}}{\text{ref\_regions\_c}} \right)^2 + \left( \frac{v_2^{(A)}}{\text{ref\_regions\_a}} \pm \frac{v_2^{(B)}}{\text{ref\_regions\_b}} \pm' \frac{v_2^{(C)}}{\text{ref\_regions\_c}} \right)^2 + \left( \frac{v_3^{(A)}}{\text{ref\_regions\_a}} \pm \frac{v_3^{(B)}}{\text{ref\_regions\_a}} \pm' \frac{v_3^{(C)}}{\text{ref\_regions\_c}} \right)^2} \quad (\text{S160})$$

All four  $(\pm, \pm')$  combinations are computed and the maximum real-space distance among these four combinations is chosen.

The condition for two region pair images to be included in the list of interacting region pair images is that the distance from the ‘lower left’ corner of the first region to the ‘lower left’ corner of the second region image should be  $\leq (\text{dipole\_interaction\_cutoff\_length} + \text{largest\_body\_diag})$ . Including the largest\_body\_diag ensures that all points in the first region are farther away than dipole\_interaction\_cutoff\_length from all points in the second region image. This list is assembled in linear-scaling computational time by using the following computational efficiencies. First, only translation indices satisfying eqn (S154) – (S156) need to be considered for the second region image. Second, according to eqn (S150) – (S152) the whole number of regions along each basis vector in the reference unit cell is chosen such that the region widths (along directions parallel to each basis vector) are as large as feasible without being greater than one-third of the dipole\_interaction\_cutoff\_length. This means a line segment of dipole\_interaction\_cutoff\_length placed parallel to a basis vector direction could completely cross at most four regions and partially cross at most five regions. Let each region in the reference unit cell be identified by whole number triplets in the interval  $( [1, \text{ref\_regions\_a}], [1, \text{ref\_regions\_b}], [1, \text{ref\_regions\_c}] )$ . Then let  $(\text{region\_A}, \text{region\_B}, \text{region\_C})$  be the region indices identifying a first region in the unit cell and  $(i, j, k)$  be the region indices identifying a second region in the reference unit cell. Then, the second region could potentially have one (or

more) images that interact with the first region if and only if all of the following conditions are satisfied. If  $\text{periodicsumlimitA} = 0$ , then

$$\max((\text{region\_A} - 5), 1) \leq i \leq \min((\text{region\_A} + 5), \text{ref\_regions\_a}) \quad (\text{S161})$$

else

$$i \in \{\text{modulo}(\text{region\_A} - B, \text{ref\_regions\_a})\} \text{ for } -5 \leq B \leq 5 \quad (\text{S162})$$

If  $\text{periodicsumlimitB} = 0$ , then

$$\max((\text{region\_B} - 5), 1) \leq j \leq \min((\text{region\_B} + 5), \text{ref\_regions\_b}) \quad (\text{S163})$$

else

$$j \in \{\text{modulo}(\text{region\_B} - B, \text{ref\_regions\_b})\} \text{ for } -5 \leq B \leq 5 \quad (\text{S164})$$

If  $\text{periodicsumlimitC} = 0$ , then

$$\max((\text{region\_C} - 5), 1) \leq k \leq \min((\text{region\_C} + 5), \text{ref\_regions\_c}) \quad (\text{S165})$$

else

$$k \in \{\text{modulo}(\text{region\_C} - B, \text{ref\_regions\_c})\} \text{ for } -5 \leq B \leq 5 \quad (\text{S166})$$

Loops over the above possibilities are performed and then a region pair image is added to the list if and only if the computed distance from the ‘lower left’ corner of the first region to the ‘lower left’ corner of the second region image is  $\leq (\text{dipole\_interaction\_cutoff\_length} + \text{largest\_body\_diag})$ . A first array is constructed listing pairs of regions having any interacting images, and a second array is prepared that lists which specific images of each particular region pair interact. The second array is constructed such that all images for a particular interacting region pair are grouped together. Region pairs that do not interact are not included in these arrays.

#### S4.5 Construct two lists of interacting atom pairs

To construct the ‘small’ and ‘large’ lists of interacting atom pairs, the outermost loop runs over atoms in the reference unit cell as taken from the array containing atoms grouped by region. For each such atom, the region to which it belongs and the atom’s coordinates are loaded from the array. A second loop, which is nested inside the first loop, runs over all second regions that have at least one image interacting with the first region (as taken from the arrays constructed in Section S4.4 above). The third loop, which is nested inside the second loop, runs over all atoms in this second region. The coordinates of the second atom are loaded here. The fourth loop, which is nested inside the third loop, runs over all potentially interacting images for the chosen region pair (as taken from the arrays constructed in Section S4.4 above). Inside this fourth loop, the distance between the two atoms is computed. If this distance is larger than  $\text{dipole\_interaction\_cutoff\_length}$ , then the program cycles to the next loop iteration. Otherwise, it checks whether the atom pair is translationally symmetry unique. If so, the atom pair is marked for inclusion in the ‘large’ list; otherwise, the program cycles to the next loop iteration. Then, the  $\text{distance\_to\_attenuation\_ratio}$  is computed for the atom pair. If  $\text{distance\_to\_attenuation\_ratio} \leq \text{distance\_to\_attenuation\_ratio\_cutoff}$ , then the pair is marked for inclusion in the ‘small’ list. Then the appropriate data is written to the ‘small’ and/or ‘large’ lists if the atom pair was marked

for inclusion in those lists, respectively. (The ‘large’ list accumulates sums over the interacting images for a particular pair of atoms in the reference unit cell. On the other hand, the ‘small’ list stores separate data for individual interacting atom pair images.) Then the program proceeds to the next loop iteration. This process is continued until all iterations are complete in the four nested loops.

**References:**

1. E. W. Weisstein, Positive Definite Matrix, MathWorld—A Wolfram Web Resource, <http://mathworld.wolfram.com/PositiveDefiniteMatrix.html>, (accessed February 2019).
2. O. Axelsson and G. Lindskog, On the rate of convergence of the preconditioned conjugate-gradient method, *Numer. Math.*, 1986, **48**, 499-523. [doi:10.1007/BF01389448](https://doi.org/10.1007/BF01389448)
3. S. F. Ashby, T. A. Manteuffel and P. E. Saylor, A taxonomy for conjugate-gradient methods, *Siam J. Numer. Anal.*, 1990, **27**, 1542-1568. [doi:10.1137/0727091](https://doi.org/10.1137/0727091)