

# ITERATIVE SOLVERS AND MINIMIZATION ALGORITHMS

Ross Bannister, DARC

August 2003/March 2004

## CONTENTS

1. The Conjugate Gradient Method for Hermitian Problems . . . . .	2
Iterative Solvers . . . . .	2
The Conjugate Gradient Method . . . . .	2
'Gradients', 'Residuals' and 'Search Directions' . . . . .	3
The Search Directions . . . . .	4
Minimizing Quadratic Surfaces when $\mathbf{A}$ is Unknown . . . . .	6
In summary . . . . .	7
2. Preconditioning . . . . .	7
Left Preconditioning . . . . .	7
Right preconditioning . . . . .	8
Choosing the operator . . . . .	8
3. The Conjugate Gradient Method for Non-Hermitian Problems . . . . .	8
4. The Generalized Conjugate Residual Method . . . . .	9
Similarities with the Analysis Used for the Conjugate Gradient Method . . . . .	9
Conditions Imposed . . . . .	10
Imposing These Conditions . . . . .	10
Line Minimization . . . . .	11
5. Example of the Generalized Conjugate residual Method With and Without Preconditioning . . . . .	12
The GCR Algorithm With no Preconditioning . . . . .	12
The GCR Algorithm With Right Preconditioning . . . . .	13
The ADI Preconditioner . . . . .	14
5. Further Information . . . . .	14

## 1. THE CONJUGATE GRADIENT METHOD FOR HERMITIAN PROBLEMS

### Iterative solvers

We wish to find the solution of the linear system,

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \quad (1.1)$$

where the matrix  $\mathbf{A}$  is Hermitian,

$$\mathbf{A} = \mathbf{A}^\dagger. \quad (1.2)$$

$\mathbf{b}$  is a known vector (akin to a source term) and  $\mathbf{x}$  is the as yet unknown vector, which is the solution of Eq. (1.1). If the number of elements in the vector,  $n$ , is small, then Eq. (1.1) can be solved by matrix inversion,

$$\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}, \quad (1.3)$$

assuming that  $\mathbf{A}$  is non-singular (in the following, we also assume that  $\mathbf{A}$  is positive definite). Matrix inversion is not possible when  $n$  is large. A more efficient method is the conjugate gradient (CG) method, which works by minimizing a function associated with Eq. (1.1). Consider the following quadratic function,

$$J = \frac{1}{2}\mathbf{x}^\dagger\mathbf{A}\mathbf{x} - \mathbf{b}^\dagger\mathbf{x}, \quad (1.4)$$

where the matrix and vectors are the same as in Eq. (1.1). The quadratic exists in the  $n$ -dimensional space that is the representation of  $\mathbf{x}$ .

Solving the variational problem of finding the  $\mathbf{x}$  in the  $n$ -dimensional space that minimizes  $J$  is the same as the solution to Eq. (1.1). How does this work? At the minimum, the gradient vanishes. The gradient, which is the vector  $\nabla_{\mathbf{x}}J$ , is, for Hermitian  $\mathbf{A}$ ,

$$\nabla_{\mathbf{x}}J = \mathbf{A}\mathbf{x} - \mathbf{b}. \quad (1.5)$$

By setting Eq. (1.5) to zero Eq. (1.1) is recovered - Q.E.D.

The CG method is a numerical method used to estimate the position of the minimum of  $J$  accurately and efficiently, thus solving Eq. (1.1). It can also be used to minimize functions in their own right, which have no associated linear equation akin in Eq. (1.1), such as the cost function which arises in variational data assimilation.

### The conjugate gradient method

The CG method is a method of searching systematically for the minimum of  $J$  in an iterative

fashion. It is not the only method available. In fact it is one in a family of related methods called "iterative solvers" or "minimization methods". The simplest method in this family is called the steepest descent algorithm. The CG method is more sophisticated, and more efficient than the steepest descent method.

### **'Gradients', 'residuals' and 'search directions'**

The algorithm is iterative in nature and is now developed. Throughout it is assumed that all variables, vectors and operators are real-valued. At the  $i$ th iteration, we are at position  $\mathbf{x}_i$ . The value of  $J$  is,

$$J = \frac{1}{2} \mathbf{x}_i^\dagger \mathbf{A} \mathbf{x}_i - \mathbf{b}^\dagger \mathbf{x}_i, \quad (1.6)$$

and the gradient is,

$$\nabla_{\mathbf{x}_i} J = \mathbf{A} \mathbf{x}_i - \mathbf{b} \equiv -\mathbf{r}_i. \quad (1.7)$$

It is customary to relate the gradient vector to the residual,  $\mathbf{r}_i$ . These are defined here to be negatives of each other, as defined on the right hand side of Eq. (1.7). The residual is the difference between  $\mathbf{b}$  and  $\mathbf{A} \mathbf{x}_i$ , which is a measure of how far we are away from the solution. The solution is where the residual vanishes,  $\mathbf{b} - \mathbf{A} \mathbf{x} = 0$ , as Eq. (1.1). We would like  $\mathbf{r}_i$  to be as small in value as possible.

In step  $i$ , we move from  $\mathbf{x}_i$  to  $\mathbf{x}_{i+1}$  along the search direction  $\mathbf{p}_i$  to lower the value of  $J$ . The direction  $\mathbf{p}_i$ , and the distance to move along it has yet to be determined. Mathematically this step is,

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \beta_i \mathbf{p}_i, \quad (1.8)$$

where  $\beta_i$  is a scalar pertaining to the  $i$ th iteration. Acting on each side with  $\mathbf{A}$  and subtracting  $\mathbf{b}$  yields,

$$\mathbf{A} \mathbf{x}_{i+1} - \mathbf{b} = \mathbf{A} \mathbf{x}_i - \mathbf{b} + \beta_i \mathbf{A} \mathbf{p}_i, \quad (1.9)$$

$$-\mathbf{r}_{i+1} = -\mathbf{r}_i + \beta_i \mathbf{A} \mathbf{p}_i. \quad (1.10)$$

First, given  $\mathbf{p}_i$  (see below), how do we choose  $\beta_i$ ? Two approaches are presented here, which later are shown to be equivalent in the CG method.

#### **(a) Orthogonal residuals**

We first choose  $\beta_i$  such as to make the new residual (ie the negative of the local gradient) orthogonal to the current residual. Mathematically this is,

$$\mathbf{r}_{i+1}^\dagger \mathbf{r}_i = 0, \quad (1.11)$$

which when imposed on Eq. (1.10) yields,

$$(\mathbf{r}_i - \beta_i \mathbf{A} \mathbf{p}_i)^\dagger \mathbf{r}_i = 0, \quad (1.12)$$

$$\beta_i = \frac{\mathbf{r}_i^\dagger \mathbf{r}_i}{\mathbf{r}_i^\dagger \mathbf{A} \mathbf{p}_i}. \quad (1.13)$$

Thus as long as we can act on  $\mathbf{p}_i$  with  $\mathbf{A}$ ,  $\beta_i$  is relatively easily found.

### **(b) Line minimization**

Another approach is to choose  $\beta_i$  to minimize  $J$  along the search direction. By varying  $\beta_i$ , the constrained minimum position is where  $dJ/d\beta_i = 0$ . This can be written in an alternative form using the chain rule,

$$\frac{dJ}{d\beta_i} = \sum_j \left( \frac{\partial J}{\partial (x)_j} \right) \bigg|_{x_{i+1}} \left( \frac{d(x)_j}{d\beta_i} \right) \bigg|_{x_{i+1}} = 0, \quad (1.14)$$

$$= \sum_j \frac{\partial J}{\partial (x_{i+1})_j} \frac{d(x_{i+1})_j}{d\beta_i} = 0, \quad (1.15)$$

where  $(x_{i+1})_j$  is the  $j$ th component of the position at the start of the  $(i+1)$ th iteration. The last equation can be written in the more compact form,

$$\frac{dJ}{d\beta_i} = \frac{\partial J}{\partial \mathbf{x}_{i+1}} \frac{d\mathbf{x}_{i+1}}{d\beta_i} = 0, \quad (1.16)$$

which is implicitly the inner product between a row and a column vector. The first vector is the gradient of  $J$  at  $\mathbf{x}_{i+1}$ , which, by definition, is minus the residual,  $-\mathbf{r}_{i+1}^\dagger$ , and the second vector can be found trivially from Eq. (1.8). Putting this together yields,

$$\frac{dJ}{d\beta_i} = -\mathbf{r}_{i+1}^\dagger \mathbf{p}_i = 0, \quad (1.17)$$

which says that the new residual is orthogonal to the search direction. Substituting  $\mathbf{r}_{i+1}$  from Eq. (1.10) gives,

$$-\mathbf{r}_i^\dagger \mathbf{p}_i + \beta_i \mathbf{p}_i^\dagger \mathbf{A} \mathbf{p}_i = 0, \quad (1.18)$$

$$\beta_i = \frac{\mathbf{r}_i^\dagger \mathbf{p}_i}{\mathbf{p}_i^\dagger \mathbf{A} \mathbf{p}_i}.$$

This equation shall be returned to later.

### **The search directions**

What is the 'search direction'  $\mathbf{p}_i$ ? Further information is needed to find this. In the CG algorithm, the search direction takes the following form,

$$\mathbf{p}_{i+1} = \mathbf{r}_{i+1} - \sum_{k=0}^i \alpha_{ki} \mathbf{p}_k. \quad (1.19)$$

This equation says that search direction (actually found during iteration  $i$  in advance of the next iteration) is the residual modified by a linear combination of previous search directions. The coefficients,  $\alpha_{ki}$ , in the linear combination are chosen to satisfy the following conjugacy condition,

$$\mathbf{p}_i^\dagger \mathbf{A} \mathbf{p}_j = 0 \text{ for } i \neq j \text{ and } \neq 0 \text{ for } i = j. \quad (1.20)$$

(The form of Eqs. (1.19) and (1.20) are then akin to the Gram-Schmidt procedure.)

Imposing Eq. (1.20) on Eq. (1.19) yields,

$$\begin{aligned} \left( \mathbf{r}_i - \sum_{k=0}^{i-1} \alpha_{k,i-1} \mathbf{p}_k \right)^\dagger \mathbf{A} \mathbf{p}_j &= 0 \text{ for } i \neq j, \\ \mathbf{r}_i^\dagger \mathbf{A} \mathbf{p}_j - \sum_{k=0}^{i-1} \alpha_{k,i-1} \mathbf{p}_k^\dagger \mathbf{A} \mathbf{p}_j &= 0 \text{ for } i \neq j. \end{aligned} \quad (1.21)$$

Note that by choosing  $j = i$  in Eq. (1.21) leads to no contribution from the summation as the index  $k$  never equals  $i$ . Equation (1.21) is then more suitably written as,

$$\mathbf{r}_i^\dagger \mathbf{A} \mathbf{p}_j - \alpha_{j,i-1} \mathbf{p}_j^\dagger \mathbf{A} \mathbf{p}_j = 0 \text{ for } j < i. \quad (1.22)$$

Equation (1.22) is considered under two circumstances.

**(a)  $j = i - 1$**

$$\mathbf{r}_i^\dagger \mathbf{A} \mathbf{p}_{i-1} - \alpha_{i-1,i-1} \mathbf{p}_{i-1}^\dagger \mathbf{A} \mathbf{p}_{i-1} = 0,$$

leading to an expression for  $\alpha_{ii}$ ,

$$\alpha_{ii} = \frac{\mathbf{r}_{i+1}^\dagger \mathbf{A} \mathbf{p}_i}{\mathbf{p}_i^\dagger \mathbf{A} \mathbf{p}_i}. \quad (1.23)$$

**(b)  $j < i - 1$**

In order to analyse the result in this case, we impose an extension of Eq. (1.11) that all residuals be orthogonal. First use Eq. (1.10) to rewrite the first term of Eq. (1.22),

$$\mathbf{r}_i^\dagger \left( \frac{\mathbf{r}_j - \mathbf{r}_{j+1}}{\beta_j} \right) - \alpha_{j,i-1} \mathbf{p}_j^\dagger \mathbf{A} \mathbf{p}_j = 0, \quad (1.24)$$

where the first term is zero under the conditions of  $j < i - 1$ . This shows that, under these conditions,

$$\begin{aligned} \alpha_{j,i-1} &= 0 \text{ for } j < i - 1, \\ \text{ie } \alpha_{ji} &= 0 \text{ for } j < i. \end{aligned} \quad (1.25)$$

Putting this information into the assumed form, Eq. (1.19) gives,

$$\mathbf{p}_{i+1} = \mathbf{r}_{i+1} - \alpha_{ii} \mathbf{p}_i, \quad (1.26)$$

$$= \mathbf{r}_{i+1} - \frac{\mathbf{r}_{i+1}^\dagger \mathbf{A} \mathbf{p}_i}{\mathbf{p}_i^\dagger \mathbf{A} \mathbf{p}_i} \mathbf{p}_i, \quad (1.27)$$

which gives the 'new' search direction in terms of the new residual and the previous search direction only. It is due to the zero elements of Eq. (1.25) that makes the CG method a highly efficient approach. If these elements were not zero, all previous search directions would have to be stored and used in the full Eq. (1.19).

### Minimizing quadratic surfaces when $\mathbf{A}$ is unknown

Most of the above theory has been developed with the problem, Eq. (1.1), in mind. The CG algorithm can also be applied when minimizing a function for the specific purpose of minimization, such as in variational data assimilation. In such applications, the value,  $J$ , at each point can be calculated, together with the local gradient,  $-\mathbf{r}$ , but the operator  $\mathbf{A}$  (actually called the Hessian) is unknown. This means that the parameters  $\beta_i$  (as in Eq. (1.13)) and  $\alpha_{ii}$  (as in Eq. (1.23)) cannot be computed directly.

$\alpha_{ii}$  can be rewritten in a form that does not explicitly involve  $\mathbf{A}$ ,

$$\begin{aligned} \text{Eq. (1.23)} \quad \Rightarrow \quad \alpha_{ii} &= \frac{\mathbf{r}_{i+1}^\dagger \mathbf{A} \mathbf{p}_i}{\mathbf{p}_i^\dagger \mathbf{A} \mathbf{p}_i}, \\ \text{use Eq. (1.10)} \quad \Rightarrow \quad &= \frac{1}{\beta_i} \frac{\mathbf{r}_{i+1}^\dagger (\mathbf{r}_i - \mathbf{r}_{i+1})}{\mathbf{p}_i^\dagger \mathbf{A} \mathbf{p}_i}, \\ \text{orthogonal residuals} \quad \Rightarrow \quad &= -\frac{1}{\beta_i} \frac{\mathbf{r}_{i+1}^\dagger \mathbf{r}_{i+1}}{\mathbf{p}_i^\dagger \mathbf{A} \mathbf{p}_i}, \\ \text{use Eq. (1.13)} \quad \Rightarrow \quad &= -\frac{\mathbf{r}_i^\dagger \mathbf{A} \mathbf{p}_i \mathbf{r}_{i+1}^\dagger \mathbf{r}_{i+1}}{\mathbf{r}_i^\dagger \mathbf{r}_i \mathbf{p}_i^\dagger \mathbf{A} \mathbf{p}_i}, \\ \text{use Eq. (1.26)} \quad \Rightarrow \quad &= -\frac{(\mathbf{p}_i + \alpha_{i-1,i-1} \mathbf{p}_{i-1})^\dagger \mathbf{A} \mathbf{p}_i \mathbf{r}_{i+1}^\dagger \mathbf{r}_{i+1}}{\mathbf{r}_i^\dagger \mathbf{r}_i \mathbf{p}_i^\dagger \mathbf{A} \mathbf{p}_i}, \\ \mathbf{A}\text{-orthogonal search directions} \quad \Rightarrow \quad &= -\frac{\mathbf{p}_i^\dagger \mathbf{A} \mathbf{p}_i \mathbf{r}_{i+1}^\dagger \mathbf{r}_{i+1}}{\mathbf{r}_i^\dagger \mathbf{r}_i \mathbf{p}_i^\dagger \mathbf{A} \mathbf{p}_i}, \\ &= -\frac{\mathbf{r}_{i+1}^\dagger \mathbf{r}_{i+1}}{\mathbf{r}_i^\dagger \mathbf{r}_i}. \end{aligned} \quad (1.28)$$

The key point is that Eq. (1.28) does not involve  $\mathbf{A}$ .

As long as the  $\beta_i$  in Eq. (1.18) derived by approach (b) (line minimization) is the same as that in Eq. (1.13) derived by approach (a) (orthogonal residuals) then a consequence is that  $\beta_i$  itself does not actually need to be calculated at all. Instead, Eq. (1.8) can be by-passed by performing a line minimization along the direction  $\mathbf{p}_i$  to arrive at the new position  $\mathbf{x}_{i+1}$ .

It then remains to be shown that the two versions of  $\beta_i$  are identical. Starting from Eq. (1.18) using method (b),

$$\begin{aligned}
\text{Eq. (1.18)} \quad \Rightarrow \quad \beta_i &= \frac{\mathbf{r}_i^\dagger \mathbf{p}_i}{\mathbf{p}_i^\dagger \mathbf{A} \mathbf{p}_i}, \\
\text{use Eq. (1.26)} \quad \Rightarrow \quad &= \frac{\mathbf{r}_i^\dagger (\mathbf{r}_i - \alpha_{i-1, i-1} \mathbf{p}_{i-1})}{\mathbf{p}_i^\dagger \mathbf{A} (\mathbf{r}_i - \alpha_{i-1, i-1} \mathbf{p}_{i-1})}, \\
\text{use Eq. (1.17)} \quad \Rightarrow \quad &= \frac{\mathbf{r}_i^\dagger \mathbf{r}_i}{\mathbf{p}_i^\dagger \mathbf{A} (\mathbf{r}_i - \alpha_{i-1, i-1} \mathbf{p}_{i-1})}, \\
\mathbf{p}_i \text{ are } \mathbf{A}\text{-orthogonal} \quad \Rightarrow \quad &= \frac{\mathbf{r}_i^\dagger \mathbf{r}_i}{\mathbf{p}_i^\dagger \mathbf{A} \mathbf{r}_i}, \tag{1.29}
\end{aligned}$$

Equation (1.29) is identical to Eq. (1.13) found from method (a), showing that the two methods are equivalent.

### In summary

In summary, the following conditions are imposed in or follow from the CG method:

1. The residuals are orthogonal, ie  $\mathbf{r}_i^\dagger \mathbf{r}_j = 0$  if  $i \neq j$ .
2. The search directions are ' $\mathbf{A}$ -orthogonal', ie  $\mathbf{p}_i^\dagger \mathbf{A} \mathbf{p}_j = 0$  if  $i \neq j$ .
3. The 'next' residual is orthogonal to the search direction, ie  $\mathbf{r}_{i+1}^\dagger \mathbf{p}_i = 0$ .

## 2. PRECONDITIONING

Although the solution of Eq. (1.1) can be found in principle by minimizing  $J$  in Eq. (1.4) in a relatively efficient way using the CG method, further improvements can be made by preconditioning the problem first.

The number of iterations required for convergence is said to be related to the conditioning number of the operator  $\mathbf{A}$ . The conditioning number is the ratio of largest to smallest eigenvalue, which ideally should be of order unity. Preconditioning is all about transforming the problem to an equivalent matrix equation that has a unitary (or close to unitary) conditioning number. Preconditioning is important because, for problems of large rank, it can make the difference between a practical and impractical convergence rate. We discuss two simple methods of preconditioning.

## Left preconditioning

Let  $\mathbf{P}$  be an operator which is approximately equal to  $\mathbf{A}$ , but can be inverted easily. Acting from the left with  $\mathbf{P}^{-1}$  on Eq. (1.1) gives,

$$\mathbf{P}^{-1}\mathbf{A}\mathbf{x} = \mathbf{P}^{-1}\mathbf{b}. \quad (2.1)$$

The solution to Eq. (2.1) is the same as that of Eq. (1.1), but where, in the CG algorithm,  $\mathbf{A} \rightarrow \mathbf{P}^{-1}\mathbf{A}$  and  $\mathbf{b} \rightarrow \mathbf{P}\mathbf{b}$ . Since, by design,  $\mathbf{P} \approx \mathbf{A}$ , the conditioning number of  $\mathbf{P}^{-1}\mathbf{A}$  in Eq. (1.30) would be expected to be close to unity.

## Right preconditioning

Right preconditioning involves an explicit change of variables. Let,

$$\mathbf{y} = \mathbf{P}\mathbf{x}, \quad (2.2)$$

$$\text{i.e. } \mathbf{x} = \mathbf{P}^{-1}\mathbf{y}, \quad (2.3)$$

where  $\mathbf{P}$  is the same operator as discussed above. This change of variables makes Eq. (1.1),

$$\mathbf{A}\mathbf{P}^{-1}\mathbf{y} = \mathbf{b}, \quad (2.4)$$

and the CG method would be applied with  $\mathbf{A} \rightarrow \mathbf{A}\mathbf{P}^{-1}$ . The solution,  $\mathbf{y}$ , to Eq. (2.4) needs to be transformed using Eq. (2.3). An example of the use of right preconditioning is given later in the context of the General Conjugate Residual method for non-Hermitian operators (section 5).

## Choosing the operator $\mathbf{P}$

There are a number of strategies of choosing  $\mathbf{P}$ . If most of the weight of  $\mathbf{A}$  is along its diagonal then it may suffice to choose,

$$\mathbf{P} = \text{diag}(\mathbf{A}), \quad (2.5)$$

where the 'diag' operation sets to zero all off diagonal elements of its argument, and leaves diagonal elements unchanged.  $\mathbf{P}$  is trivial to invert. Slightly more sophisticated, but on a similar note, is to create a tridiagonal matrix,

$$\mathbf{P} = \text{tridiag}(\mathbf{A}), \quad (2.6)$$

which is possible to invert efficiently. Other, more complicated techniques also exist, e.g. the "Alternating Direction Implicit" method discussed later in the context of the "Generalised Conjugate Residual" method (section 5).

### 3. THE CONJUGATE GRADIENT METHOD FOR NON-HERMITIAN PROBLEMS

If the matrix  $\mathbf{A}$  is non-Hermitian, then the state of  $\mathbf{x}$  that minimizes the function Eq. (1.4) is not the solution of Eq. (1.1) (Eq. (1.4) has been formed by assuming Hermitian symmetry properties of  $\mathbf{A}$ ). The author does not know of any function that can be written in this way for non-Hermitian  $\mathbf{A}$ . However, a Hermitian problem that *is* equivalent to Eq. (1.1) in the case that  $\mathbf{A}$  is non-Hermitian can be written if the adjoint of  $\mathbf{A}$  is known. Act with  $\mathbf{A}^\dagger$  on each side of Eq. (1.1),

$$\mathbf{A}^\dagger \mathbf{A} \mathbf{x} = \mathbf{A}^\dagger \mathbf{b}. \quad (3.1)$$

The active operator in this problem is  $\mathbf{A}^\dagger \mathbf{A}$  and is Hermitian. Now the CG algorithm can be applied with the substitution,  $\mathbf{A} \rightarrow \mathbf{A}^\dagger \mathbf{A}$  and  $\mathbf{b} \rightarrow \mathbf{A}^\dagger \mathbf{b}$ . It is found that systems of equations akin to Eq. (3.1) tend to be very badly conditioned. Since the treatment of badly conditioned systems (section 2) becomes expensive, alternative algorithms are sought.

### 4. THE GENERALIZED CONJUGATE RESIDUAL METHOD

We wish to solve Eq. (1.1) in the case that  $\mathbf{A}$  is non-Hermitian. If  $\mathbf{A}$  is Hermitian, we could construct a function, Eq. (1.4), whose minimum is exactly the solution of Eq. (1.1). Since we cannot write a function directly for a non-Hermitian operator, we minimize the  $L_2$  norm,  $J^2$ , of the residual,

$$J^2 = \frac{1}{2} \|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2^2, \quad (4.1)$$

$$= \frac{1}{2} (\mathbf{b} - \mathbf{A}\mathbf{x})^\dagger (\mathbf{b} - \mathbf{A}\mathbf{x}), \quad (4.2)$$

and  $J$  is not the same function used in the CG method. The factor of 1/2 is present for convenience. In particular, with this factor, differentiating Eq. (4.2) gives the gradient which contains no numerical factors,

$$\nabla_{\mathbf{x}} J^2 = \mathbf{A}^\dagger \mathbf{A} \mathbf{x} - \mathbf{A}^\dagger \mathbf{b}. \quad (4.3)$$

It is assumed that the minimum of  $J^2$ , ie where  $\nabla_{\mathbf{x}} J^2 = 0$  in Eq. (4.3), is close to the solution of Eq. (1.1). It will become clear later how the GCR solver minimizes this function. It is assumed, for simplicity, that all variables, vectors and operators are real-valued.

#### Similarities with the analysis used for the conjugate gradient method

Under the generalized conjugate residual (GCR) method, ideas similar to the CG approach are used. In particular, Eq. (1.8) giving the position at the start of the 'next' iteration is,

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \beta \mathbf{p}_i. \quad (4.4)$$

Here, the residual shall be defined as the difference,

$$\mathbf{A}\mathbf{x}_i - \mathbf{b} = \mathbf{r}_i. \quad (4.5)$$

This is the negative of the definition used before for the CG method, Eq. (1.7), which could just as well have been used here. The different form of Eq. (4.5) is taken here for compatibility with the particular algorithm that the author is using (Met Office GCR(k) algorithm). Combining Eqs. (4.4) and (4.5), gives an expression for the 'next' residual in the iteration,

$$\mathbf{r}_{i+1} = \mathbf{r}_i + \beta_i \mathbf{A}\mathbf{p}_i. \quad (4.6)$$

Unlike for the CG method, this residual is not the gradient of the scalar to be minimized (or minus the gradient), as was found in Eq. (1.7). This is not the case here because the form of the scalar to be minimized has a different form to that used in section 1.

The remaining equation used is the one that updates the search direction for the 'next' iteration. This is Eq. (1.19) for the CG algorithm, given as Eq. (4.7) here for the GCR, with a minor modification (minus sign becomes a plus) for compatibility again with the particular algorithm used,

$$\mathbf{p}_{i+1} = \mathbf{r}_{i+1} + \sum_{k=0}^i \alpha_{ki} \mathbf{p}_k. \quad (4.7)$$

## Conditions imposed

The difference with the CG now follows. For the GCR, the following conditions are imposed:

1. The residuals,  $\mathbf{r}_i$ , at the end of the  $i$ th step (start of the  $i+1$ th step) are ' $\mathbf{A}$ -orthogonal' to the search directions for the  $i$ th step,  $\mathbf{p}_i$ ,

$$\mathbf{r}_{i+1}^\dagger \mathbf{A}\mathbf{p}_i = 0. \quad (4.8)$$

2. The search directions are ' $\mathbf{A}^\dagger \mathbf{A}$ -orthogonal',

$$(\mathbf{A}\mathbf{p}_i)^\dagger (\mathbf{A}\mathbf{p}_j) = 0 \text{ for } i \neq j. \quad (4.9)$$

These are alternative to the conditions of Eqs. (1.11) and (1.20) used for the CG method.

## Imposing these conditions

Imposing condition 1 (Eq. (4.8)) on Eq. (4.6) yields an expression for  $\beta_i$ ,

$$(\mathbf{r}_i + \beta_i \mathbf{A}\mathbf{p}_i)^\dagger \mathbf{A}\mathbf{p}_i = 0, \quad (4.10)$$

$$\beta_i = -\frac{\mathbf{r}_i^\dagger \mathbf{A}\mathbf{p}_i}{(\mathbf{A}\mathbf{p}_i)^\dagger (\mathbf{A}\mathbf{p}_i)}, \quad (4.11)$$

which is then used in Eq. (4.4) to give the position at the start of the next iteration.

The search direction has not yet been determined and is found using Eq. (4.7) with condition 2 (Eq. (4.9)) imposed. Act with  $\mathbf{A}$  on Eq. (4.7),

$$\mathbf{A}\mathbf{p}_{i+1} = \mathbf{A}\mathbf{r}_{i+1} + \sum_{k=0}^i \alpha_{ki} \mathbf{A}\mathbf{p}_k, \quad (4.12)$$

and perform an inner product with  $\mathbf{A}\mathbf{p}_j$ ,

$$(\mathbf{A}\mathbf{p}_j)^\dagger \mathbf{A}\mathbf{p}_{i+1} = (\mathbf{A}\mathbf{p}_j)^\dagger \mathbf{A}\mathbf{r}_{i+1} + \sum_{k=0}^i \alpha_{ki} (\mathbf{A}\mathbf{p}_j)^\dagger \mathbf{A}\mathbf{p}_k. \quad (4.13)$$

We will now consider two scenarios for the choice of  $j$ .

**(a)**  $j = i + 1$

The summation over  $k$  in Eq. (4.13) never reaches  $i + 1$ . Hence Eq. (4.8) becomes,

$$(\mathbf{A}\mathbf{p}_{i+1})^\dagger \mathbf{A}\mathbf{p}_{i+1} = (\mathbf{A}\mathbf{p}_{i+1})^\dagger \mathbf{A}\mathbf{r}_{i+1}. \quad (4.14)$$

**(b)**  $j \leq i$

This scenario yields information on the constants  $\alpha_{kj}$ . Due to the condition  $j \leq i$ , and Eq. (4.9), the left hand side of Eq. (4.13) is zero. This gives rise to the following,

$$\begin{aligned} 0 &= (\mathbf{A}\mathbf{p}_j)^\dagger \mathbf{A}\mathbf{r}_{i+1} + \alpha_{ji} (\mathbf{A}\mathbf{p}_j)^\dagger \mathbf{A}\mathbf{p}_j, \\ \alpha_{ji} &= -\frac{(\mathbf{A}\mathbf{p}_j)^\dagger \mathbf{A}\mathbf{r}_{i+1}}{(\mathbf{A}\mathbf{p}_j)^\dagger \mathbf{A}\mathbf{p}_j}. \end{aligned} \quad (4.15)$$

The constants  $\alpha_{ij}$ , which can now be calculated, are plugged back into Eq. (4.12) for the search direction for the 'next' iteration. Unlike for CG, these coefficients are not found to be zero for all previous search directions, and so all previous search directions have to be stored.

## Line minimization

Why do the conditions of Eqs. (4.8) and (4.9) lead to a minimization of  $J^2$  of Eq. (4.1)? An analysis, along the lines of that performed in section 1 for the CG algorithm is made here to show this.

Start with Eq. (4.4); given a search direction  $\mathbf{p}_i$ , how far in this direction from  $\mathbf{x}_i$  should one move to minimize  $J^2$  along this line? This gives us a  $\beta_i$  such that  $dJ^2/d\beta_i = 0$ . Start with Eq. (1.16), but for  $J^2$  instead of  $J$ ,

$$\frac{dJ^2}{d\beta_i} = \frac{\partial J^2}{\partial \mathbf{x}_{i+1}} \frac{d\mathbf{x}_{i+1}}{d\beta_i} = 0, \quad (4.16)$$

The gradient vector in Eq. (4.16), comes from Eq. (4.3) and the other derivative in Eq. (4.16) follows immediately from Eq. (4.4),

$$\begin{aligned}
\frac{dJ^2}{d\beta_i} &= (\mathbf{A}^\dagger \mathbf{A} \mathbf{x}_{i+1} - \mathbf{A}^\dagger \mathbf{b})^\dagger \mathbf{p}_i, \\
&= \{\mathbf{A}^\dagger (\mathbf{A} \mathbf{x}_{i+1} - \mathbf{b})\}^\dagger \mathbf{p}_i, \\
&= (\mathbf{A} \mathbf{x}_{i+1} - \mathbf{b})^\dagger \mathbf{A} \mathbf{p}_i, \\
&= \mathbf{r}_{i+1}^\dagger \mathbf{A} \mathbf{p}_i = 0,
\end{aligned} \tag{4.14}$$

which says that the new residual is  $\mathbf{A}$ -orthogonal to the search direction. Substituting  $\mathbf{r}_{i+1}$  from Eq. (4.6) gives,

$$\begin{aligned}
\mathbf{r}_i^\dagger \mathbf{A} \mathbf{p}_i + \beta_i (\mathbf{A} \mathbf{p}_i)^\dagger \mathbf{A} \mathbf{p}_i &= 0, \\
\beta_i &= -\frac{\mathbf{r}_i^\dagger \mathbf{A} \mathbf{p}_i}{(\mathbf{A} \mathbf{p}_i)^\dagger \mathbf{A} \mathbf{p}_i}.
\end{aligned} \tag{4.15}$$

This is the same result as Eq. (4.11), which was derived from the condition given as Eq. (4.8).

This says that the condition of Eq. (4.8) is equivalent to minimizing  $J^2$  along the search direction.

## 5. EXAMPLE OF THE GENERALIZED CONJUGATE RESIDUAL METHOD WITH AND WITHOUT PRECONDITIONING

The results shown in section 4 can be used to formulate an algorithm to solve Eq. (1.1). We first construct an algorithm that involves no preconditioning. This is then modified into an equivalent form that does involve preconditioning.

### The GCR algorithm with no preconditioning

Set up initial values

Guess the solution,  $\mathbf{x}_0$

The initial residual,  $\mathbf{r}_0 = \mathbf{A} \mathbf{x}_0 - \mathbf{b}$

The initial search direction,  $\mathbf{p}_0 = \mathbf{r}_0$

Set  $i=0$

While not converged

Calculate  $\beta_i = -\mathbf{r}_i^\dagger \mathbf{A} \mathbf{p}_i / [(\mathbf{A} \mathbf{p}_i)^\dagger \mathbf{A} \mathbf{p}_i]$

Update  $\mathbf{x}$ ,  $\mathbf{x}_{i+1} = \mathbf{x}_i + \beta_i \mathbf{p}_i$

Update residual,  $\mathbf{r}_{i+1} = \mathbf{r}_i + \beta_i \mathbf{A} \mathbf{p}_i$

Calculate  $\alpha_{ji}$  for  $0 \leq j \leq i$ ,  $\alpha_{ji} = -(\mathbf{A} \mathbf{p}_j)^\dagger \mathbf{A} \mathbf{r}_{i+1} / [(\mathbf{A} \mathbf{p}_j)^\dagger \mathbf{A} \mathbf{p}_j]$

Update search direction,  $\mathbf{p}_{i+1} = \mathbf{r}_{i+1} + \sum_k \alpha_{ki} \mathbf{p}_k$   
Increment  $i$   
End of while loop

### The GCR algorithm with right preconditioning

Using so-called "right preconditioning" (section 2), we apply the above algorithm to a different problem by making the following substitutions,

$$\begin{aligned}\mathbf{A} &\rightarrow \mathbf{A}\mathbf{P}^{-1}, \\ \mathbf{x}_i &\rightarrow \mathbf{P}\tilde{\mathbf{x}}_i, \\ \mathbf{p}_i &\rightarrow \tilde{\mathbf{p}}_i, \\ \mathbf{r}_i &\rightarrow \tilde{\mathbf{r}}_i, \\ \beta_i &\rightarrow \tilde{\beta}_i, \\ \alpha_{ji} &\rightarrow \tilde{\alpha}_{ji},\end{aligned}$$

where  $\mathbf{P}$  is a preconditioning operator (section 2). Making the above substitutions and acting with an additional  $\mathbf{P}^{-1}$  on some of the lines, gives the following new algorithm.

Set up initial values  
Guess the solution,  $\tilde{\mathbf{x}}_0$   
The initial residual,  $\tilde{\mathbf{r}}_0 = \mathbf{A}\mathbf{P}^{-1}\mathbf{P}\tilde{\mathbf{x}}_0 - \mathbf{b}$   
The initial search direction,  $\mathbf{P}^{-1}\tilde{\mathbf{p}}_0 = \mathbf{P}^{-1}\tilde{\mathbf{r}}_0$   
Set  $i=0$   
While not converged  
Calculate  $\tilde{\beta}_i = -\tilde{\mathbf{r}}_i^\dagger \mathbf{A}\mathbf{P}^{-1}\tilde{\mathbf{p}}_i / [(\mathbf{A}\mathbf{P}^{-1}\tilde{\mathbf{p}}_i)^\dagger \mathbf{A}\mathbf{P}^{-1}\tilde{\mathbf{p}}_i]$   
Update  $\tilde{\mathbf{x}}$ ,  $\mathbf{P}^{-1}\mathbf{P}\tilde{\mathbf{x}}_{i+1} = \mathbf{P}^{-1}\mathbf{P}\tilde{\mathbf{x}}_i + \tilde{\beta}_i \mathbf{P}^{-1}\tilde{\mathbf{p}}_i$   
Update residual,  $\tilde{\mathbf{r}}_{i+1} = \tilde{\mathbf{r}}_i + \tilde{\beta}_i \mathbf{A}\mathbf{P}^{-1}\tilde{\mathbf{p}}_i$   
Calculate  $\tilde{\alpha}_{ji}$  for  $0 \leq j \leq i$ ,  $\tilde{\alpha}_{ji} = -(\mathbf{A}\mathbf{P}^{-1}\tilde{\mathbf{p}}_j)^\dagger \mathbf{A}\mathbf{P}^{-1}\tilde{\mathbf{r}}_{i+1} / [(\mathbf{A}\mathbf{P}^{-1}\tilde{\mathbf{p}}_j)^\dagger \mathbf{A}\mathbf{P}^{-1}\tilde{\mathbf{p}}_j]$   
Update search direction,  $\mathbf{P}^{-1}\tilde{\mathbf{p}}_{i+1} = \mathbf{P}^{-1}\tilde{\mathbf{r}}_{i+1} + \sum_k \tilde{\alpha}_{ki} \mathbf{P}^{-1}\tilde{\mathbf{p}}_k$   
Increment  $i$   
End of while loop

Note the following:

- Apart from the initial guess of the solution, all occurrences of  $\bar{\mathbf{x}}_i$  are preceded by  $\mathbf{P}^{-1}\mathbf{P}$ . Since these operators cancel,  $\bar{\mathbf{x}}_i$  exists in the same space as  $\mathbf{x}_i$ .
- All occurrences of  $\bar{\mathbf{p}}_i$  are preceded by  $\mathbf{P}^{-1}$ . For convenience, write  $\mathbf{P}^{-1}\bar{\mathbf{p}}_i = \hat{\mathbf{p}}_i$ .

Making this change, and allowing  $\mathbf{P}^{-1}\mathbf{P}$  to cancel, gives the following algorithm.

Set up initial values

Guess the solution,  $\bar{\mathbf{x}}_0$

The initial residual,  $\bar{\mathbf{r}}_0 = \mathbf{A}\bar{\mathbf{x}}_0 - \mathbf{b}$

The initial search direction,  $\hat{\mathbf{p}}_0 = \mathbf{P}^{-1}\bar{\mathbf{r}}_0$

Set  $i=0$

While not converged

Set  $\mathbf{q}_i = \mathbf{A}\hat{\mathbf{p}}_i$

Calculate  $\bar{\beta}_i = -\bar{\mathbf{r}}_i^\dagger \mathbf{q}_i / [(\mathbf{A}\hat{\mathbf{p}}_i)^\dagger \mathbf{A}\hat{\mathbf{p}}_i]$

Update  $\bar{\mathbf{x}}$ ,  $\bar{\mathbf{x}}_{i+1} = \bar{\mathbf{x}}_i + \bar{\beta}_i \hat{\mathbf{p}}_i$

Update residual,  $\bar{\mathbf{r}}_{i+1} = \bar{\mathbf{r}}_i + \bar{\beta}_i \mathbf{q}_i$

Calculate  $\bar{\alpha}_{ji}$  for  $0 \leq j \leq i$ ,  $\bar{\alpha}_{ji} = -(\mathbf{A}\hat{\mathbf{p}}_j)^\dagger \mathbf{A}\mathbf{P}^{-1}\bar{\mathbf{r}}_{i+1} / [(\mathbf{A}\hat{\mathbf{p}}_j)^\dagger \mathbf{A}\hat{\mathbf{p}}_j]$

Update search direction,  $\hat{\mathbf{p}}_{i+1} = \mathbf{P}^{-1}\bar{\mathbf{r}}_{i+1} + \sum_k \bar{\alpha}_{ki} \hat{\mathbf{p}}_k$

Increment  $i$

End of while loop

This is the preconditioned algorithm. The preconditioner enters the problem in the computation of the initial search direction, in the calculation of  $\bar{\alpha}_{ji}$  and in the expression for the updated search direction. For efficiency, an extra variable,  $\mathbf{q}_i$ , has been introduced in the algorithm above. In this algorithm, it is now not strictly necessary to use overbars and hats on the variables (since their original counterparts - without overbars and hats - are not used, there is no need to distinguish between them).

## The ADI preconditioner

Section to be written.

## **6. FURTHER INFORMATION**

The document, "Iterative Solvers" (2000) by M. Zerroukat of the Met Office was very useful in writing these notes. It discusses other related methods and gives an application to the "New dynamics" formulation of the Unified Model. References are provided in this document.