MoA Interpretation of the Iterative Conjugate Gradient Method with Psi Reduction - A Tutorial to teach the Mathematically literate in Linear and Tensor Algebra: Part I

Lenore M. Mullin and Paul Sebexen

April 5, 2019
1 Motivation and Introduction

1.0.1 Intentions of Tutorial

It is often difficult to learn new mathematics semantically and syntactically, even when there are similarities in the words and meaning when discussed aloud. The goal of this document is to facilitate learning through explanations and definitions relating our common mathematical knowledge and highlighting what is new. It is meant to be a working document that will evolve based on feedback from target audiences, those mathematically literate in linear and tensor algebra, those that want to learn MoA, Psi Calculus [1], and its uses [2], those that want and need the ability to prove a design, either in hardware or software through the ONF, Operational Normal Form, and those wanting to exploit all resources optimally, especially when Tensor Algebra, i.e. algorithms foundational to their application, are needed: Knowledge Representation and Modeling, Scientific Computing, Signal Processing, etc.

1.0.2 Matrix Multiply and Solvers

We aim to solve for $\vec{x}$, a vector, in

$$A\vec{x} = \vec{b}$$

(1)

where $A$ is symmetric, real, positive-definite, and all variables are conformable for the operation, i.e. $\vec{x}$, $\vec{b}$, and the number of columns in $A$ have the same number of components.

The matrix-vector product, like other inner products from linear algebra, can be expressed in MoA as $\newcommand\mb[1]{\langle #1 \rangle} \mb{+} \cdot \times$, denoting point-wise scalar extension of multiplication over the inner dimension, followed by a single application of additive reduction.

We are motivated by a few anomalies in Linear and Tensor Algebra. From a CMU CS professor’s notes on painless-conjugate-gradient [3]: “... A is an $n$ by $n$ matrix, and $x$ and $b$ are vectors, that is, $n$ by 1 matrices. The inner product of two vectors is written $x^T y$ and represents the scalar sum $\sum_{i=1}^{n} x_i y_i$ ... In general, expressions that reduce to 1 by 1 matrices are treated as scalars.”

Although we can build symbolic software systems that apply the same equalities as one might do by hand, that implementation is often slow. If we are to have the mathematical prowess of tensor algebra and the computational/communication prowess needed for IoT and the complexity of real time systems with knowledge and reasoning one must realize that it is hard to prove properties of programs if grammars have anomalies. Ideally, if everything was a linear, or multilinear operation, reasoning was possible. We often forget that what can be done in our minds, on paper, etc. mathematically, must now be transformed to an accurate answer done in the most computationally efficient way, given a certain set of resources. What we ignore costs computational resources to analyze and deal with the anomaly. Linear, or multilinear, operations, have the advantage that everything is mathematical without exception. In the above case, at one moment it says a vector is an $n$ by 1, then in the summation it is represented as a vector. They ARE different entities. In a tensor world, a scalar is a 0-dimensional array, a vector is 1-d, and a 1 by 1, is a 2-d array. MoA and Psi Calculus give us this formality while providing the full tensor algebra we are familiar with.

We study the CG because it is the most prominent iterative method for solving sparse systems of linear equations. For dense systems, factor and solve with back substitution is often best. Re complexity, the time spent factoring a dense matrix is roughly equivalent to the time spent solving the system iteratively. Finally, we study the CG because, if it is fundamental to other solvers. We aim to parameterize the variants, while we assure flexible use, performance, and accuracy.

2 An Iterative Solver: The Conjugate Gradient (CG)

Rewriting the objective with this new notation, we now aim to solve

$$A_+ \cdot \times \vec{x} \equiv \vec{b}$$

(2)

where $n$ is a positive integer indicating the size of the problem, and the shapes are denoted as

$$\rho A \equiv \langle n \ n \rangle$$

(3)
and

\[ \rho \vec{x} \equiv \rho \vec{b} \equiv < n > \]  (4)

2.1 Base Case

The pseudocode gives us two lines setting the initial condition of the working arrays:

\[ r_0 := b - Ax_0 \]  (5)
\[ p_0 := r^T_0 \]  (6)

Figure 1: Pseudo Code: Base Case

The base case populated prior to the first iteration can be expressed by first initializing the guess for \( \vec{x} \) as a zero vector of shape \(< n >\) (could be set to a better guess if available). Then, we rewrite the pseudocode.

2.1.1 Pseudocode to MoA: Base Case

In the pseudocode we see there are \( k \) iterations, implying that a matrix with \( k \) rows will hold all recurrences. But, it turns out we only need 2 rows in each matrix used in the recurrence, i.e. the previous time step, and the present time step. We keep the same loop bounds, i.e. \( 0 \leq i < n \), but we change the loop body from \( i \) and \( i + 1 \) to \( i - i \) and \( i + 1 - i \), i.e. 0 and 1. We then create the following matrices \( X, P \) and \( R \), each a 2 by \( n \). Notice how Expression (6) looks now. The Transpose has been eliminated because the transpose of a vector in MoA is the same vector.

\[ < 0 > \psi X \equiv < 0 ... 0 > \]  (7)
\[ < 0 > \psi P \equiv < 0 > \psi R \equiv \vec{b} - A_+ \cdot \vec{x} < 0 > \psi X \]  (8)

Figure 2: MoA: Base Case

2.2 Subsequent Iterations

2.2.1 Pseudocode to MoA: Subsequent Iterations

The iterative solver can be expressed in MoA for a temporal index, \( < i > \), where

\[ 0 \leq i < n \]  (18)

effectively replaces the function of \( k \) in the pseudocode and allows the procedure to be written using the array equality operator, “\( \equiv \)”, without requiring the assignment operator, “\( \colon= \)”. This can be thought of as unrolling the temporal dimension. This subtlety, is why we are making the change in variable name.

2.2.2 The Loop Body

We rewrite the loop body in MoA syntax by:

1. Replacing bold vectors with iteration subscripts, with a matrix and an index of that iteration. Vectors that are really vectors are denoted by the same named variable, e.g. \( x \), but as vector symbols: \( \vec{x} \rightarrow \vec{x} \)
2. Replacing subscript indexing with the \( \psi \) binary function: \( r_k \rightarrow < i > \psi R \)
3. Using transpose unary prefix operator \( \psi \): \( r_k^T \rightarrow \psi \psi R \)
4. Using inner product binary operator \( \cdot \vec{x} \), as discussed earlier
The pseudocode next gives us

\[ k := 0 \] (9)

\[
\text{repeat} \\
\alpha_k := \frac{r_k^T r_k}{p_k^T A p_k} \\
x_{k+1} := x_k + \alpha_k p_k \\
r_{k+1} := r_k - \alpha_k A p_k \\
\text{if } r_{k+1} \text{ is sufficiently small, then exit loop} \\
\beta_k := \frac{r_k^T r_{k+1}}{r_k^T r_k} \\
p_{k+1} := r_{k+1} + \beta_k p_k \\
k := k + 1
\] (10-16)

end repeat

Result is \( x_{k+1} \). (17)

---

The resulting algorithm is written as

\[ < i > \psi \Vec{\alpha} \equiv \frac{(\bigotimes < i > \psi R)_+ \bullet_x (< i > \psi R)}{(\bigotimes < i > \psi P)_+ \bullet_x A_+ \bullet_x (< i > \psi P)} \times (< i > \psi P) \] (19)

\[ < i + 1 > \psi X \equiv (< i > \psi X) + (< i > \psi \Vec{\alpha}) \times (< i > \psi P) \] (20)

\[ < i + 1 > \psi R \equiv (< i > \psi R) - (< i > \psi \Vec{\alpha}) \times A_+ \bullet_x (< i > \psi P) \] (21)

\[ < i > \psi \Vec{\beta} \equiv \frac{(\bigotimes < i + 1 > \psi R)_+ \bullet_x (< i + 1 > \psi R)}{(\bigotimes < i > \psi R)_+ \bullet_x (< i > \psi R)} \times (< i > \psi R) \] (22)

\[ < i + 1 > \psi P \equiv (< i + 1 > \psi R) + (< i > \psi \Vec{\beta}) \times (< i > \psi P) \] (23)

Result is \( < i + 1 > \psi X \). (24)

---

2.2.3 Resulting Algorithm

The vectors \( \Vec{\alpha} \) and \( \Vec{\beta} \) can be substituted in for each usage. The expanded algorithm is now

2.2.4 Substituting for \( \Vec{\alpha} \) and \( \Vec{\beta} \)

\[ < i + 1 > \psi X \equiv (< i > \psi X) + \left( \frac{(\bigotimes < i > \psi R)_+ \bullet_x (< i > \psi R)}{(\bigotimes < i > \psi P)_+ \bullet_x A_+ \bullet_x (< i > \psi P)} \right) \times (< i > \psi P) \] (25)

\[ < i + 1 > \psi R \equiv (< i > \psi R) - \left( \frac{(\bigotimes < i > \psi P)_+ \bullet_x A_+ \bullet_x (< i > \psi P)}{(\bigotimes < i > \psi R)_+ \bullet_x (< i > \psi R)} \right) \times (< i > \psi P) \] (26)

\[ < i + 1 > \psi P \equiv (< i + 1 > \psi R) + \left( \frac{(\bigotimes < i + 1 > \psi R)_+ \bullet_x (< i + 1 > \psi R)}{(\bigotimes < i > \psi R)_+ \bullet_x (< i > \psi R)} \right) \times (< i > \psi P) \] (27)

Result is \( < i + 1 > \psi X \). (28)
2.2.5 Transpose

Note that transpose is an unnecessary operation on vectors in MoA (the typical issue of row- and column-vectors encountered in linear algebra is ignored). The reason why, is based on the definition of inner product, which states that,

The shape of the result of the inner product, is the shape of the left argument, exclusive of its last component on the right concatenated to the shape of the right argument, exclusive of its first component. Also, the last component of the left argument, must be the same, as the first component of the right.

So for example, with two vectors, the left argument \( \vec{w} \), and \( \vec{v} \), the right argument, both have shape \(<5>\), i.e., \( \rho \vec{w} = <5> \) and \( \rho \vec{v} = <5> \). Substituting in what was said above:

1. The last component of the left argument, \((-1) \Delta (\rho \vec{w}) = (-1) \Delta <5> = <5>\), and the first component of the right argument \(1 \Delta (\rho \vec{v}) = 1 \Delta <5> = <5>\), are the same:

2. Shape of the result: Everything but the last component of the left argument, which is the empty vector, \(\Theta\), i.e. \((-1) \nabla (\rho \vec{w}) = (-1) \nabla <5> = \Theta\), concatenated to everything but the first argument of the right argument, also \(\Theta\), i.e. \(1 \nabla (\rho \vec{v}) = 1 \nabla <5> = \Theta\). Thus, \(\Theta + \Theta\). By definition of concatenation, the concatenation of two empty vectors is \(\Theta\), which is the shape of a scalar in MoA, that is, what we know is the result of the inner product of two vectors, or row vectors.

The inner product operator behaves as expected on a pair of vector arguments, so each occurrence of transpose can be eliminated. Also note that, within a single iteration only two index values are used on the LHS of \(\psi\): \(<i + 1>\) and \(<i>\). Without loss of generality, we can subtract \(i\) from each index to reduce the size of working memory.

\[
\begin{align*}
<i - i + 1> \psi_X &\equiv (<i - i> \psi_X) + \left( \frac{(<i - i> \psi R)_+ \bullet \times (<i - i> \psi P)}{(i - i + 1) \psi P} \right) \times (<i - i> \psi P) \\
(<i - i + 1> \psi R) &\equiv (<i - i> \psi R) - \left( \frac{(<i - i> \psi R)_+ \bullet \times (<i - i> \psi R)}{(i - i + 1) \psi P} \right) \times A_+ \bullet \times (<i - i> \psi P) \\
(<i - i + 1> \psi P) &\equiv (<i - i + 1> \psi R) + \left( \frac{(<i - i + 1> \psi R)_+ \bullet \times (<i - i + 1> \psi R)}{(i - i + 1) \psi R} \right) \times (<i - i> \psi P)
\end{align*}
\]

Result is \(<i - i + 1> \psi_X\).

Figure 5: MoA: Subsequent Iterations - Transpose Removed

2.2.6 Simplifying
Observing that \(< i > - < i > ≤ < 0 >\) for all valid indices \(< i >\), s.t. \(0 ≤ i < n\), we can simplify the expressions to

\[
<1 > \psi X ≡ (<0 > \psi X) + \left( \frac{(<0 > \psi R) + \bullet_x (<0 > \psi R)}{(<0 > \psi P) + \bullet_x (<0 > \psi P)} \right) × ( <0 > \psi P)
\]

(33)

\[
<1 > \psi R ≡ (<0 > \psi R) - \left( \frac{(<0 > \psi R) + \bullet_x (<0 > \psi R)}{(<0 > \psi P) + \bullet_x (<0 > \psi P)} \right) × ( <0 > \psi P)
\]

(34)

\[
<1 > \psi P ≡ (<1 > \psi R) + \left( \frac{(<1 > \psi R) + \bullet_x (<1 > \psi R)}{(<0 > \psi R) + \bullet_x (<0 > \psi R)} \right) × ( <0 > \psi P)
\]

(35)

Result is \(<1 > \psi X\).

(36)

Figure 6: MoA: Subsequent Iterations - Simplified Indexing

2.3 Psi Reduction

2.3.1 MoA Definition of Inner Product: n-d

To paraphrase the definition of inner product, given that \(\xi_l\) denotes the left argument, and \(\xi_r\) the right, providing \(1 ≤ \delta \xi_l\) and \(1 ≤ \delta \xi_r\), i.e. the dimensionality of both arguments must be greater or equal to 1.

\[
q ≡ (\nabla \rho \xi_l)[0] ≡ (\nabla \rho \xi_r)[0]
\]

(37)

q is a scalar, so we must index the one element vector produced by -1 take of the left shape and 1 take of the right shape. Thus, the shape of the result is defined by

\[
\rho \xi_l + \bullet_x \xi_r ≡ (\nabla \rho \xi_l) + (\nabla \rho \xi_r)
\]

(38)

and for \(0 ≤ i < (\nabla \rho \xi_l)[0]\) and \(0 ≤ j < (\nabla \rho \xi_r)[0]\)

\[
(<i j>) \psi \xi_l(\bullet_{op} \bullet_{op}) \xi_r ≡ \psi \xi_l(\bullet_{op} \bullet_{op}) \xi_r
\]

(39)

2.3.2 Applying the Definition of Inner Product

Before we begin, we observe that many of the pieces of the derivation are the same. Consequently, we pick pieces, reduce to normal form, then put the pieces together to show an ONF for the CG. Applying this definition, we can start to reduce the terms containing relevant inner product applications

\[
q ≡ (\nabla \rho R)[0] ≡ (\nabla < 2 n >)[0] ≡ < n > [0] ≡ n
\]

(40)

\[
(<0 > \psi R) + \bullet_x (<0 > \psi R) ≡ (\Theta, \Theta) \psi ( <0 > \psi R) + \bullet_x (<0 > \psi R)
\]

(41)

\[
≡ \bar{red}(\Theta \psi <0 > \psi R) × (\psi \Omega <0 > \Theta) ( <0 > \psi R)
\]

(42)

Here we say, take each scalar, from the left argument \(\psi\), and concatenate to each vector of the right argument, \(\Theta\).

\(\Theta\) indexing any array is that array.

\[
≡ \bar{red}(<0 > \psi R) × (\psi \Omega <0 > \Theta) ( <0 > \psi R)
\]

(43)

The result of applying Omega is the following.

\[
≡ \bar{red}(<0 > \psi R) × \begin{bmatrix}
0 \\
\vdots \\
0 \\
(\psi <0 > \psi R)
\end{bmatrix}
\]

(44)

Note that the shape \(< n 1 >\), of the above array, selects every element of the right argument \(<0 > \psi R\), so we have

\[
≡ \bar{red}(<0 > \psi R) × <0 > \psi R
\]

(45)
From the definition of reduction, we know

\[ +\text{red} \vec{v} \equiv \sum_{i=0}^{-1+\tau\vec{v}} \vec{v}[i] \] (46)

We can further simplify expression (33).

\[ (<0>\psi R)_+ \bullet_\times (<0>\psi R) \equiv \sum_{j=0}^{-1+\tau((<0>\psi R) \times (<0>\psi R))} ((<0>\psi R) \times (<0>\psi R))[j] \] (47)

and since \(-1 + \tau((<0>\psi R) \times (<0>\psi R)) \equiv -1 + \tau(<0>\psi R) \equiv n - 1\)

\[ \equiv \sum_{j=0}^{n-1} (<0>\psi R)[j] \times (<0>\psi R)[j] \] (48)

In MoA, \text{rav} flattens any array, in various orderings, e.g. row major.

by PCT, which flattens contiguous row(s) of R based on ordering

\[ \equiv \sum_{j=0}^{n-1} (\text{rav} R)[(0 \times \pi(1 \triangle \rho R)) + j] \times (\text{rav} R[(0 \times \pi(1 \triangle \rho R)) + j]) \] (49)

\[ \equiv \sum_{j=0}^{n-1} (\text{rav} R)[j]^2 \] (50)

2.3.3 Another Common Expression

The only difference in the following is \(<1>\psi R\). Similarly, we can simplify expression (35).

\[ (<1>\psi R)_+ \bullet_\times (<1>\psi R) \equiv \sum_{j=0}^{n-1} (<1>\psi R)[j] \times (<1>\psi R)[j] \] (51)

\[ \equiv \sum_{j=0}^{n-1} (\text{rav} R)[(1 \times \pi(1 \triangle \rho R)) + j] \times (\text{rav} R[(1 \times \pi(1 \triangle \rho R)) + j]) \] (52)

\[ \equiv \sum_{j=0}^{n-1} (\text{rav} R)[n + j]^2 \] (53)
2.3.4 Final Common Section

Now, we reduce the remaining inner product term in expressions (33) and (34). ∀ i,j s.t. 0 ≤ i < (ρA)[0] and 0 ≤ j < (ρA)[1]

\[
\langle 0 > ψP \rangle + \bullet_\times A_+ \bullet_\times (< 0 > ψP) \equiv \langle 0 > ψP \rangle + \bullet_\times (A_+ \bullet_\times (< 0 > ψP)) \\
\equiv \langle 0 > ψP \rangle + \bullet_\times ((\Theta, \Theta)ψA_+ \bullet_\times (< 0 > ψP)) \\
\equiv \langle 0 > ψP \rangle + \bullet_\times (∧red(ΘψA)_+Ω<1,0> < 0 > ψP) \\
\equiv \langle 0 > ψP \rangle + \bullet_\times \sum_{j=0}^{n-1} (< j > ψA) \times (0 > ψP)[j] \\
\equiv \langle 0 > ψP \rangle + \bullet_\times \sum_{j=0}^{n-1} (rav A)[i \times π(1 Δ ρA)] + j \times
\]

\[
(rav P)[0 \times π(1 Δ ρP)] + j \\
\equiv \langle 0 > ψP \rangle + \bullet_\times \sum_{j=0}^{n-1} (rav A)[i \times n] + j \times (rav P)[j] \\
\equiv \sum_{i=0}^{n-1} (rav P)[i] \times \sum_{j=0}^{n-1} (rav A)[i \times n] + j \times (rav P)[j] \\
\equiv \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} (rav P)[i] \times (rav A)[j + i \times n] \times (rav P)[j] \\
\]

2.3.5 Putting it All Together

Applying these reductions to the iterative algorithm step, we obtain the new expressions for all 0 ≤ k < n and ∀ i,j s.t. 0 ≤ i < (ρA)[0] and 0 ≤ j < (ρA)[1].

\[
(rav X)[n + k] \equiv (rav X)[k] + (rav P)[k] \times \frac{\sum_{j=0}^{n-1} (rav R)[j] \times (rav A)[j + i \times n] \times (rav P)[j]}{\sum_{i=0}^{n-1} \sum_{j=0}^{n-1} (rav P)[i] \times (rav A)[j + i \times n] \times (rav P)[j]} \\
\]

\[
(rav R)[n + k] \equiv (rav R)[k] - \frac{\sum_{i=0}^{n-1} (rav P)[i] \times (rav A)[j + i \times n] \times (rav P)[j]}{\sum_{i=0}^{n-1} \sum_{j=0}^{n-1} (rav P)[i] \times (rav A)[j + i \times n] \times (rav P)[j]} \times
\]

\[
\sum_{j=0}^{n-1} (rav A)[(k \times n) + j] \times (rav P)[j] \\
\]

\[
(rav P)[n + k] \equiv (rav R)[n + k] + (rav P)[k] \times \frac{\sum_{j=0}^{n-1} (rav R)[n + j] \times (rav R)[j]}{\sum_{j=0}^{n-1} (rav R)[j]} \\
\]

Result is (rav X)[n + k].

Figure 7: MoA: Subsequent Iterations - Psi Reduced, The ONF
2.4 Complete Iterative Solver, The MoA ONF

For all $0 \leq k < n$, set the initial values

$$(\text{rav } X)[k] \equiv 0 \quad \text{(or better guess if available)} \quad (66)$$

$$(\text{rav } P)[k] \equiv (\text{rav } R)[k] \equiv (\text{rav } b)[k] - \sum_{j=0}^{n-1} (\text{rav } A)[(k \times n) + j] \times (\text{rav } X)[j] \quad (67)$$

Then repeat the following steps:

Update the guess

$$(\text{rav } X)[n + k] \equiv (\text{rav } X)[k] + (\text{rav } P)[k] \times \left(\sum_{j=0}^{n-1} (\text{rav } R)[j]^2\right) / \left(\sum_{i=0}^{n-1} \sum_{j=0}^{n-1} (\text{rav } P)[i] \times (\text{rav } A)[j + i \times n] \times (\text{rav } P)[j]\right) \quad (68)$$

Calculate the new residual

$$(\text{rav } R)[n+k] \equiv (\text{rav } R)[k] - \left(\sum_{j=0}^{n-1} (\text{rav } R)[j]^2\right) / \left(\sum_{i=0}^{n-1} \sum_{j=0}^{n-1} (\text{rav } P)[i] \times (\text{rav } A)[j + i \times n] \times (\text{rav } P)[j]\right) \times \sum_{j=0}^{n-1} (\text{rav } A)[j + k \times n] \times (\text{rav } P)[j]$$

Check if the new residual meets the convergence criterion or if we have completed all $n$ iterations of the direct solution; stop iterating if so. If not, calculate the new value of $P$ and loop back to update the guess again.

$$(\text{rav } P)[n + k] \equiv (\text{rav } R)[n + k] + (\text{rav } P)[k] \times \left(\sum_{j=0}^{n-1} (\text{rav } R)[n + j]^2\right) / \left(\sum_{j=0}^{n-1} (\text{rav } R)[j]^2\right) \quad (69)$$

When complete, the solution is $(\text{rav } X)[n + k]$. Note that this definition can be readily translated to a C-like language by replacing each raveled array with a pointer to an address in memory and using the square brackets as the C array dereference operator.

Figure 8: Complete Solver: MoA ONF

2.4.1 Example

We consider a linear system $A \bullet x \equiv \bar{b}$ given by

$$A = \begin{bmatrix} 4 & 1 \\ 1 & 3 \end{bmatrix}, \quad \bar{b} = < x_0, x_1 > = < 1, 2 > \quad (70)$$

This system differs from the linear algebra formulation by removing details about row and column vectors that are unnecessary for finding a solution. While we could start with any guess without loss of generality, we will consider

$$x \equiv < x_0, x_1 > = < 2, 1 > \quad (71)$$
The remaining initial values are then

\[(\text{rav } P)[k] \equiv (\text{rav } R)[k] \equiv <1 2 > [k] - \sum_{j=0}^{1} <4 1 1 3 > [j + k \times 2] \times <2 1 > [j] \quad (72)\]

\[\equiv <1 2 > [k] - ( <4 1 1 3 > [0 + k \times 2] \times <2 1 > [0]) + (<4 1 1 3 > [1 + k \times 2] \times <2 1 > [1]) \quad (73)\]

\[\equiv <1 2 > [k] - ( <4 1 1 3 > [k \times 2] \times 2) + (<4 1 1 3 > [1 + k \times 2]) \quad (74)\]

Since \(0 \leq k < n\), we can expand easily for all values \(k\)

\[(\text{rav } P)[0] \equiv (\text{rav } R)[0] \equiv <1 2 > [0] - (<4 1 1 3 > [0 \times 2] \times 2) + (<4 1 1 3 > [1 + 0 \times 2]) \equiv 1 - (4 \times 2) + 1 \equiv -8 \quad (75)\]

(Remember, by default, without any operation hierarchy, MoA operations are evaluated from right to left)

\[(\text{rav } P)[1] \equiv (\text{rav } R)[1] \equiv <1 2 > [1] - (<4 1 1 3 > [1 \times 2] \times 2) + (<4 1 1 3 > [1 + 1 \times 2]) \equiv 2 - (1 \times 2) + 3 \equiv -3 \quad (76)\]

So our initial condition is

\[0\psi P \equiv 0\psi R \equiv < -8 -3 > \quad (77)\]

We now evaluate the next approximate solution by solving for \(<1 > \psi X\) for all valid \(k\) and find

\[(\text{rav } X)[1 + k] \equiv (\text{rav } X)[k] + (\text{rav } P)[k] \times \left( \frac{1}{\sum_{j=0}^{1} (\text{rav } R)[j]^2} \right) / \left( \frac{1}{\sum_{j=0}^{1} \sum_{i=0}^{1} (\text{rav } P)[i] \times (\text{rav } A)[j + i \times 2] \times (\text{rav } P)[j]} \right) \quad (78)\]

Substituting in the calculated initial condition, we have

\[(\text{rav } X)[1 + k] \equiv < 2 1 > [k] + < -8 -3 > [k] \times \left( \sum_{j=0}^{1} < -8 -3 > [j]^2 \right) / \left( \sum_{j=0}^{1} \sum_{i=0}^{1} < -8 -3 > [i] \times <4 1 1 3 > [j + i \times 2] \times < -8 -3 > [j] \right) \quad (79)\]

which evaluates to

\[(\text{rav } X)[1 + k] \equiv < 2 1 > [k] + < -8 -3 > [k] \times \frac{73}{331} \approx < 0.2356 \ 0.3384 > \quad (80)\]

We then compute the next residual vector \(<1 > \psi R\)

\[(\text{rav } R)[2 + k] \equiv < -8 -3 > [k] - \left( \sum_{j=0}^{1} < -8 -3 > [j]^2 \right) / \left( \sum_{j=0}^{1} \sum_{i=0}^{n-1} < -8 -3 > [i] \times <4 1 1 3 > [j + i \times 2] \times < -8 -3 > [j] \right) \times \sum_{j=0}^{1} <4 1 1 3 > [j + k \times 2] \times < -8 -3 > [j] \quad (81)\]

\[(\text{rav } R)[2 + k] \equiv < -8 -3 > [k] - \frac{73}{331} \times \sum_{j=0}^{1} < 4 1 1 3 > [j + k \times 2] \times < -8 -3 > [j] \quad (82)\]
\[(\text{rav } R)[2 + 0] = \langle -8 - 3 \rangle [0] - \frac{73}{331} \times \sum_{j=0}^{1} < 4 \ 1 \ 1 \ 3 > [j + 0 \times 2] \times < -8 - 3 > [j]\] (83)

\[(\text{rav } R)[2] = \langle -8 - \frac{73}{331} \times \sum_{j=0}^{1} < 4 \ 1 \ 1 \ 3 > [j] \times < -8 - 3 > [j]\] (84)

\[\approx -8 - \frac{73}{331} \times (\langle 4 \ 1 \ 1 \ 3 \rangle [0] \times \langle -8 - 3 \rangle [0]) + (\langle 4 \ 1 \ 1 \ 3 \rangle [1] \times \langle -8 - 3 \rangle [1])\] (85)

\[\approx -8 - \frac{73}{331} \times (4 \times -8) + (1 \times -3)\] (86)

\[\approx -8 - \frac{73}{331} \times -35\] (87)

\[\approx -0.2810\] (88)

\[(\text{rav } R)[2 + 1] = \langle -8 - 3 \rangle [1] - \frac{73}{331} \times \sum_{j=0}^{1} < 4 \ 1 \ 1 \ 3 > [j + 1 \times 2] \times < -8 - 3 > [j]\] (89)

\[(\text{rav } R)[3] = \langle -3 - \frac{73}{331} \times \sum_{j=0}^{1} < 4 \ 1 \ 1 \ 3 > [j + 2] \times < -8 - 3 > [j]\] (90)

\[\approx -3 - \frac{73}{331} \times (\langle 4 \ 1 \ 1 \ 3 \rangle [2] \times \langle -8 - 3 \rangle [0]) + (\langle 4 \ 1 \ 1 \ 3 \rangle [3] \times \langle -8 - 3 \rangle [1])\] (91)

\[\approx -3 - \frac{73}{331} \times (1 \times -8) + (3 \times -3)\] (92)

\[\approx -3 - \frac{73}{331} \times -17\] (93)

\[\approx 0.7492\] (94)
The new residual vector is now \(< 1 > \psi_R \equiv< -0.2810 \ 0.7492 >\). Assuming this does not meet our convergence criterion, we will continue to calculate \(< 1 > \psi P\).

\[
(rav \ P)[2 + k] \equiv< -8 \ -3 \ -0.2810 \ 0.7492 > [2 + k] + < -8 \ -3 > [k] \times \\
\left( \frac{\sum_{j=0}^{1} < -8 \ -3 \ -0.2810 \ 0.7492 > [2 + j]^2}{\sum_{j=0}^{1} < -8 \ -3 \ -0.2810 \ 0.7492 > [j]^2} \right)
\]

(95)

\[
(rav \ P)[2] \equiv< -8 \ -3 \ -0.2810 \ 0.7492 > [2] + < -8 \ -3 > [0] \times \\
\left( \frac{\sum_{j=0}^{1} < -8 \ -3 \ -0.2810 \ 0.7492 > [2 + j]^2}{\sum_{j=0}^{1} < -8 \ -3 \ -0.2810 \ 0.7492 > [j]^2} \right)
\]

(96)

\[
= -0.2810 + -8 \times \left( \sum_{j=0}^{1} < -0.2810 \ 0.7492 > [j]^2 \right) / \left( \sum_{j=0}^{1} < -8 \ -3 > [j]^2 \right)
\]

(97)

\[
= -0.2810 + -8 \times .6403/73
\]

(98)

\[
= -0.3512
\]

(99)

\[
(rav \ P)[3] \equiv< -8 \ -3 \ -0.2810 \ 0.7492 > [3] + < -8 \ -3 > [1] \times \\
\left( \frac{\sum_{j=0}^{1} < -8 \ -3 \ -0.2810 \ 0.7492 > [2 + j]^2}{\sum_{j=0}^{1} < -8 \ -3 \ -0.2810 \ 0.7492 > [j]^2} \right)
\]

(100)

\[
= 0.7492 + -3 \times \left( \sum_{j=0}^{1} < -0.2810 \ 0.7492 > [j]^2 \right) / \left( \sum_{j=0}^{1} < -8 \ -3 > [j]^2 \right)
\]

(101)

\[
= 0.7492 + -3 \times .6403/73
\]

(102)

\[
= 0.7229
\]

(103)

So, \(< 1 > \psi P \equiv< -0.3512 \ 0.7229 >\). Now we can iterate with the new values of \(X\), \(R\), and \(P\) until a satisfactory solution is obtained.

### 3 Translating to Pseudocode

Note that we have switched back to using an assignment operator “:=” to mimic execution behavior. Recall, that this definition can be readily translated to a C-like language by replacing each raveled array with a pointer to an address in memory, denoted by \(X\), \(R\), and \(P\), and using the square brackets as the C array dereference operator.

#### 3.1 The CG: ONF to Pseudocode

**3.1.1 Implementations from the ONF: Pseudocode**

We now have a design for hardware or software.
For $0 \leq k < n$:

\[
X[k] := 0 \quad \text{(or better guess if available)} \quad (104)
\]

\[
P[k] := R[k] := b[k] - \sum_{j=0}^{n-1} A[(k \times n) + j] \times X[j] \quad (105)
\]

While not converged:

\[
X[n + k] := X[k] + P[k] \times \left( \sum_{j=0}^{n-1} R[j]^2 \right) / \left( \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} P[i] \times A[j + i \times n] \times P[j] \right) \quad (106)
\]

\[
R[n + k] \equiv R[k] - \left( \sum_{j=0}^{n-1} R[j]^2 \right) / \left( \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} P[i] \times A[j + i \times n] \times P[j] \right) \times \sum_{j=0}^{n-1} A[j + k \times n] \times P[j]
\]

Exit now if converged.

\[
P[n + k] \equiv R[n + k] + P[k] \times \left( \sum_{j=0}^{n-1} R[n + j]^2 \right) / \left( \sum_{j=0}^{n-1} R[j]^2 \right) \quad (107)
\]

\[
X[k] := X[n + k] \quad (108)
\]

\[
R[k] := R[n + k] \quad (109)
\]

\[
P[k] := P[n + k] \quad (110)
\]

Repeat.

Return $X[n + k]$

Figure 9: The CG: ONF to Pseudocode

References

[1] L. M. R. Mullin. *A Mathematics of Arrays*. PhD thesis, Syracuse University, December 1988.

[2] L. R. Mullin and J. E. Raynolds. *Conformal Computing: Algebraically connecting the hardware/software boundary using a uniform approach to high-performance computation for software and hardware applications*. arXiv:0803.2386, 2008.

[3] Jonathan Richard Shewchuk. *An introduction to the conjugate gradient method without the agonizing pain*. Available online, August 1994.