Chapter 2

An Improved Wavelet Based Preconditioner for Sparse Linear Problems
2.1 Introduction

In this chapter, we present the construction of purely algebraic Daubechies wavelet based preconditioners for Krylov subspace iterative methods to solve linear sparse system of equations. Effective preconditioners are designed with DWTPerMod algorithm by knowing size of the matrix and the order of Daubechies wavelet. A notable feature of this algorithm is that it enables wavelet level to be chosen automatically making it more robust than other wavelet based preconditioners and avoids user choosing a level of transform. We demonstrate the efficiency of these preconditioners by applying them to several sparse matrices [Table 1.1] for restarted GMRES.

DWT is a big boon especially in signal and image processing where in smooth data can be compressed sufficiently without loosing primary features of the data. Compressed data either thresholding or cutting is of much use if we are not using it in further process such as using it as preconditioner, which causes large fill-in in its routine applications. However, the novel scheme, DWT with permutation, explained by Chen[6], avoids these drawbacks especially creation of finger pattern matrices etc, and enables in selecting efficient preconditioners using wavelets. Stan-
standard DWT results in matrices with finger pattern. If DWT is followed by the permutation of the rows and columns of the matrix then it centres/brings the finger pattern about the leading diagonal. This strategy is termed as DWTPer. Elegant analysis presented by Chen enables in predicting the width of band of entries of matrix which have larger absolute values. Later, an approximate form of this can be formed and taken as preconditioner, which controls fill-in whenever schemes like decomposition is used. Similar criteria is adopted in transforming non-smooth parts, if any, which are horizontal/vertical bands and are shifted to the bottom and right-hand edges of the matrix after applying DWTPer. This procedure is termed as DWTPerMod[14,15] and takes care of other cases where non-smooth parts are located in the matrix. This scheme is more effective in incorporating the missing finer details such as fixing of precise bandwidth and automatic selection of optimal choice of transform level. Using DWTPerMod algorithm, Ford[14] has presented its salient features by applying it to standard dense matrices arising in various disciplines/fields of interest.

Chen[7] has exhaustively listed various wavelet based schemes obtained by matrix/operator splitting with Discrete Wavelet Transform (DWT) to
construct new and improved preconditioners for dense matrices. Chen[6] and Ford[14] have developed this category of preconditioners for dense matrices. In this class of schemes DWT, DWTPer (DWT with Permutation) and DWTPerMod(DWTPer with modification) are attractive and work efficiently compared with other classical(standard) methods. Where as Rathish Kumar and Mehra[34] confine to large sparse matrices having high condition numbers. They use iDWT(incomplete DWT i.e., implementation of DWT by zero padding) and find iDWTPer to be more robust preconditioners for sparse systems.

Motivated by the work of Chen[6] and Ford [14], we have successfully used DWTPerMod algorithm, for selecting the level of wavelet transform automatically by knowing the size of the sparse matrix, order of wavelet used and construct effective preconditioner for sparse unsymmetric linear systems.

2.1.1 Practical Computations

Suppose that a vector (signal) s from vector space $\mathbb{R}^n$ is given. One may construct it as an infinite sequence by extending the signal by periodic[24,40] and use this extended signal as a sequence of scaling co-
coefficients for some underlying function \( f_L \) in some fixed space \( V_L \) of \( L^2 \) (from (1.30)) as

\[
f_L(x) = \sum_k s_{L,k} \phi_{L,k}(x)
\]

From (1.16), \( V_L = V_r + W_{r+1} + ... + W_{L-1} \), this implies that

\[
f_L(x) = \sum_k s_{r,k} \phi_{r,k}(x) + \sum_{t=r}^{L-1} \sum_i d_{t,i} \psi_{t,i}(x)
\]

coefficients \( s, d \) are called smooth (filtered by lowpass) and detail/difference (filtered by highpass) parts of \( f \) respectively, where

\[
s_{j-1,k} = \sum_m h_{m-2k}s_{j,m} \quad \text{and} \quad d_{j-1,k} = \sum_m g_{m-2k}s_{j,m} \quad (2.1)
\]

\[
s_{j+1,n} = \sum_k h_{n-2k}s_{j,k} + g_{n-2k}d_{j,k} \quad (2.2)
\]

The process of obtaining (2.1) for various \( j \)'s is termed as Discrete Wavelet Transform and (2.2) is inverse of (2.1) (Mallat Algorithm[28]).

DWT transforms the vector \( s \in \mathbb{R}^n \) to

\[
w = [s_r^T, d_r^T, d_{r+1}^T, ..., d_{L-1}^T]^T \quad (2.3)
\]

The goal of wavelet transform is to make the transformed vector to be nearly sparse. This can be achieved by increasing the number of vanishing moments(\( m \)) of wavelet function i.e. \( \int \psi(x)x^t dx = 0 \) for \( t = 0, 1, ..., m - 1 \). For a given even \( D \in N \), there exists compactly supported Daubechies wavelet of order \( D \) (Daub-D) having \( D/2 \) number
of vanishing moments with finite filter coefficients and related by \( g_n = (-1)^{n}h_{D-1-n} \), with \( D \) being the length of \( \{h_k\} \) i.e., \( h_0, h_1, \ldots, h_{D-1} \) [11].

### 2.2 Wavelet Based Preconditioners

After applying wavelet transform to a signal, its local features (singularities etc if any) are scattered in (2.3), i.e., standard wavelet transform is not centred one. To bring (2.3) to centred one, Chen [6] has applied permutation matrices to (2.3) and hence the name DWTPer. For the brief description of DWT and DWTPer, we have their matrix representations in the following forms.

Assume \( n = 2^L \), for some positive integer \( L \) and \( r \) an integer such that \( 2^r < D \) and \( 2^{r+1} \geq D \), where \( D \) is the order of Daubechies wavelet, \( r = 0 \), for \( D = 2 \) (Haar wavelets) and \( r = 1 \) for \( D = 4 \).

In matrix form of (2.3) can be expressed as

\[
w = P_kW_kP_{k-1}W_{k-1}\ldots P_2W_2P_1W_1
\]  

\[
w = Ws
\]  

45
with $\tilde{P}_i$ is a permutation matrix of size $\frac{n}{2^i-1}$, $\tilde{P}_i = I(1,3,5,\ldots,\frac{n}{2^i-1} = 1,2,4,\ldots,\frac{n}{2^i-1})$, $J_i$ is identity matrix of size $n-\frac{n}{2^i-1}$ and $W_i = \begin{pmatrix} \tilde{W}_i \\ J_i \end{pmatrix}$

with $\tilde{W}_i$ is defined by the following matrix.

$$
\tilde{W}_i = \begin{bmatrix}
    h_0 & h_1 & h_2 & \cdots & h_{D-1} \\
    g_0 & g_1 & g_2 & \cdots & g_{D-1} \\
    h_0 & h_1 & h_2 & \cdots & h_{D-1} \\
    g_0 & g_1 & g_2 & \cdots & g_{D-1} \\
    \vdots & \vdots & \vdots & \ddots & \vdots \\
    h_0 & h_1 & h_2 & \cdots & h_{D-1} \\
    g_0 & g_1 & g_2 & \cdots & g_{D-1} \\
    h_2 & \cdots & h_{D-1} & \vdots & \vdots \\
    g_2 & \cdots & g_{D-1} & g_0 & g_1
\end{bmatrix}_{n \times n}
$$

Let $W$ and $\hat{W}$ denote DWT and DWTPer matrices respectively for Daubechies orthogonal wavelet. Chen[6] defines one level DWTPer ma-
matrix for Daubechies wavelet of order $D$ and it is given by

$$
\begin{bmatrix}
  h_0 & \varphi & h_1 & \varphi & h_2 & \varphi & \ldots & h_{D-1} \\
  \varphi & I & \varphi & \varphi & \varphi & \varphi & \ldots & \varphi \\
  g_0 & \varphi & g_1 & \varphi & g_2 & \varphi & \ldots & g_{D-1} \\
  \varphi & \varphi & I & \varphi & \varphi & \ldots & \varphi & \ldots & \varphi \\
  \ldots & \varphi & I & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\
  \vdots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\
  h_2 & \varphi & \ldots & h_{D-1} & h_0 & \varphi & h_1 & \varphi \\
  \varphi & \varphi & \varphi & I & \varphi & \varphi \\
  g_2 & \varphi & \ldots & g_{D-1} & g_0 & \varphi & g_1 & \varphi \\
  \varphi & \varphi & \ldots & \varphi & \varphi & \varphi & \varphi & \varphi & \varphi & 1
\end{bmatrix}_{n \times n}
$$

Here $I$ is an identity matrix of size $2^{i-1} - 1$ and $\varphi$'s block zero matrices. For $i = 1$, both $I$ and $\varphi$ are of size 0, i.e. $\hat{W}_1 = \hat{W}_1 = W_1$.

Therefore, DWTPer for a vector $s \in \mathbb{R}^n$ is defined by $\hat{w} = \hat{W}s$, with $\hat{W} = \hat{W}_k\hat{W}_{k-1}\ldots\hat{W}_2\hat{W}_1$ for $k$ levels. Since Daubechies wavelets is orthogonal, this implies that $W^{-1} = W^T$. These wavelets were defined on the real line, i.e. on a one-dimensional domain. To create wavelets on higher dimensional domains, one of the approaches is to perform the wavelet transform independently for each dimension. For two dimensional cases,
let $A$ be a $n \times n$ matrix then its wavelet transform is

$$\tilde{A} = WAW^T \quad (2.6)$$

Here $\tilde{A}$ contains four types of coefficients/subbands $[30,40,47]$:
- LL-lowpass in both the horizontal and vertical directions (approximation/average coefficients),
- LH-lowpass in the vertical, highpass in the horizontal direction,
- HL and HH (detail coefficients).

When iterated on the approximation coefficients, the result is multiresolution decomposition as shown in the Figure 2.1. This LL subband contains lowpass information, which is essentially a low resolution and represents a coarser version of the original matrix very much.

![1.jpg](image-url)

**Figure 1.** Two-dimensional wavelet transform: iteration on the LL subbands (average coefficients).

If $A$ is smooth, the transformed matrix $\tilde{A}$ will have a large part of small coefficients, corresponding to detail coefficients (after thresholding). Singularity features within $A$ give rise to additional large entries in $\tilde{A}$. For
a matrix that is smooth apart from the diagonal, the large entries form a
finger pattern, as shown in left-side of Figure 2.2. This sparsity pattern is
not convenient for preconditioning purposes, because of large amount of
fill-in that occurs under LU factorisation. One way of avoiding the finger
pattern is to permute the rows and columns of $\tilde{A}$ so as to bring the detail
coefficients into a diagonal band. The sparsity pattern of this DWTPer
transform is shown in the centre of Figure 2.2. For a matrix $A$ of order
$n \times n$, the DWTPer would give $\tilde{A} = \tilde{W}AW^T$. To relate $\hat{A}$ to $\tilde{A}$ from a
standard DWT or relate $\hat{W}$ to $W$, Chen[6] proved that $\hat{A} = R\tilde{A}R^T$,
where $R = P_1^T P_2^T ... P_k^T$ and each $P_i$ is a permutation matrix as defined
in (2.5).
2.3 Banded wavelet based preconditioner

To solve $Ax = b$, using wavelet based preconditioner following algorithm is considered [6,34].

**Algorithm 2.1[6]:**

1) Apply DWTPer to $Ax = b$ to obtain $\hat{A}u = z$
2) Select a suitable band form $\hat{M}$ of $\hat{A}$ (Theorem 2.1)
3) Use $\hat{M}$ as a preconditioner to solve $\hat{A}u = z$ iteratively
4) Apply Inverse Wavelet Transform on $u$ to get required solution.

The strategy is that we take in preconditioning step is split the given matrix $A = P + Q$, where $P$ is a band($\alpha$, $\beta$) for some $\alpha, \beta \in N$. First apply DWTPer with $k$ levels to give

$$\hat{A}u = (\hat{P} + \hat{Q})u = z$$  \hspace{1cm} (2.7)

Here $\hat{P}$ is at most of band ($\lambda_1, \lambda_2$), $\lambda_1, \lambda_2$ are first given by Chen[6] and further tightened by Ford[14], which are given in the following Theorem. Select a matrix $\hat{M}$ as preconditioner of band($\alpha_1, \beta_1$) part of the matrix $\hat{A}$ such that $\alpha_1 \ll \lambda_1$ and $\beta_1 \ll \lambda_2$
Theorem 2.1 [14] : When an order $D$, level $k$ DWTPer is applied to a band $(\alpha, \beta)$ matrix $A$ with lower bandwidth $\alpha$ and upper bandwidth $\beta$, the resulting $\hat{A}$ which is at most a band $(\lambda_1, \lambda_2)$ with $\lambda_1 - \alpha = \lambda_2 - \beta = (D - 1)(2^k - 1) + 2^{k-1}$.

By using Algorithm 2.1, Rathish Kumar and Mehra[34] developed wavelet based preconditioners for Krylov subspace iterative methods for ill conditioned sparse matrices and shown that these preconditioners are more effective compared with that of classical preconditioners by applying them to several test matrices. Performance of Algorithm 2.1 depends on the level of wavelet transform used, which must be decided in advance by user. The following impulsive results due to [14] overcome this limitation. In the following section, it is briefly summarized.

2.4 Border Block Preconditioner

After applying $k$ level of DWT to matrix $A$, detail coefficients in $\hat{A}$ are brought to make diagonal band by permutation matrices (thus obtained is $\hat{A}$). Now permute the rows and columns so that average coefficients in $\hat{A}$ are confined within bands of width $\left\lceil n \div 2^k \right\rceil$, at the bottom (horizontal) and to the right-hand (vertical) edges of $\hat{A}$. Then preconditioner is
constructed by setting to zero all entries that fall outside diagonal band and those two edge bands (right side of Figure 2.2). This modification is termed as DWTPerMod.

Once we determine the diagonal bandwidth (using Theorem 2.1), we can estimate the cost of applying the factorised block preconditioner by forward and backward substitution, based on widths of diagonal, horizontal and vertical bands. The cost of forward and backward substitution is proportional to the number $N_z(k)$ of nonzero entries in $LU$ factors of $P$.

Hence we choose $k$ such that $N_z(k)$ is minimum. A matrix of size $n$ with borders of width $r$ and a diagonal band with lower and upper bandwidths $p$ can be factored into $LU$ factors such that $N_z(k) \approx n \times (3p + 2r)$ [17].

We summarise our new preconditioning method as Algorithm 2.2.

**Algorithm 2.2 [DWTPerMod Preconditioner]:** Given a sparse matrix $A$ of size $n$ and a DWT of order $D$, compute DWTPerMod preconditioner as follows:

1. For $i = 1, 2, \ldots, \lfloor \log_2 n/(D-1) \rfloor$, compute
   
   $p(i)$ using Theorem 2.1,

   $r(i) = n \div 2^i$, $N_z(i) = n \times (3p(i) + 2r(i))$.

2. Choose $k$ such that $N_z(k) = \min_i \{N_z(i)\}$. 

52
3. Apply a level $k$ DWTPer to $A$ to obtain $\hat{A}$.

4. Permute the rows and columns of $\hat{A}$ so that the average coefficients lie in bands at the bottom and right hand edges.

5. Form a border block preconditioner $M$ by setting to zero entries in the matrix obtained in the step 4 outside of diagonal band of width $p_1(k) \ll p(k)$ and borders of width $r_1(k) \ll |r(k)|$.

For simplicity we took the upper and lower bandwidths equal ($\alpha = \beta$). However, this preconditioning strategy is equally applicable when bandwidths are different [14].

### 2.5 Numerical Experiments

To test the robustness of above explained wavelet based preconditioners, we have considered various problems given in Table 1.1. The right hand side of linear system was computed from the solution vector of all ones. This choice is suggested by Zlatev [51]. We have implemented the proposed algorithms using Matlab-7.5 and Mathematica-7. The initial guess is always $x_0 = 0$ and stopping criteria is relative residual is less than or equal to $10^{-6}$ (i.e., $\| b - Ax \|_2 \leq \| b \|_2 10^{-6}$) and the Krylov Subspace iterative Method adapted is GMRES (25). The symbol $\Omega$ stands for no
convergence in the tables and numbers in the table represent number of iterations required for convergence. Last column of Table 2.3 is obtained without using preconditioner for GMRES (25). Daubechies wavelets of order two, four and six are considered in our numerical experiments for construction of preconditioners for GMRES (25). Tables 2.3 to 2.5 are obtained for $\alpha = \beta = 5$. Table 2.6 is obtained for various values of $\alpha$ and $\beta$.

By applying Algorithm 2.2, we can easily fix the level of transform for a matrix of given size and prescribed Daubechies wavelet in developing wavelet based preconditioners for Krylov subspace iterative methods. To illustrate further salient features of DWTPeRMod based preconditioner the computed details are shown in Table 2.6 for Gre-512 matrix. It is of interest to note that DWT based preconditioner fail to yield convergence with Daubechies wavelet of order two, four and six, where as DWTPeRMod based preconditioner with Daub-4 and Daub-6 result into faster convergence of iterative schemes.
Table 2.3: Convergence details using Daub-2 based preconditioners

<table>
<thead>
<tr>
<th>Matrix Name</th>
<th>DWT</th>
<th>DWTPer</th>
<th>DWTPerMod</th>
<th>P = I</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bcsstk02</td>
<td>93</td>
<td>91</td>
<td>29</td>
<td>165</td>
</tr>
<tr>
<td>Gre-216a</td>
<td>$\Omega$</td>
<td>$\Omega$</td>
<td>267</td>
<td>$\Omega$</td>
</tr>
<tr>
<td>Qc324</td>
<td>$\Omega$</td>
<td>12</td>
<td>11</td>
<td>$\Omega$</td>
</tr>
<tr>
<td>Ck400</td>
<td>172</td>
<td>22</td>
<td>23</td>
<td>$\Omega$</td>
</tr>
<tr>
<td>Epb0</td>
<td>$\Omega$</td>
<td>1042</td>
<td>1247</td>
<td>$\Omega$</td>
</tr>
<tr>
<td>Dw8192</td>
<td>$\Omega$</td>
<td>17</td>
<td>16</td>
<td>$\Omega$</td>
</tr>
</tbody>
</table>

Table 2.4: Convergence details using Daub-4 based preconditioners

<table>
<thead>
<tr>
<th>Matrix Name</th>
<th>DWT</th>
<th>DWTPer</th>
<th>DWTPerMod</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bcsstk02</td>
<td>113</td>
<td>67</td>
<td>16</td>
</tr>
<tr>
<td>Gre-216a</td>
<td>$\Omega$</td>
<td>71</td>
<td>21</td>
</tr>
<tr>
<td>Qc324</td>
<td>$\Omega$</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>Ck400</td>
<td>443</td>
<td>64</td>
<td>51</td>
</tr>
<tr>
<td>Epb0</td>
<td>$\Omega$</td>
<td>684</td>
<td>214</td>
</tr>
<tr>
<td>Dw8192</td>
<td>$\Omega$</td>
<td>23</td>
<td>25</td>
</tr>
</tbody>
</table>
Table 2.5: Convergence details using Daub-6 based preconditioners

<table>
<thead>
<tr>
<th>Matrix Name</th>
<th>DWT</th>
<th>DWTPer</th>
<th>DWTPerMod</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bcsstk02</td>
<td>66</td>
<td>24</td>
<td>16</td>
</tr>
<tr>
<td>Gre-216a</td>
<td>Ω</td>
<td>765</td>
<td>65</td>
</tr>
<tr>
<td>Qc324</td>
<td>394</td>
<td>16</td>
<td>17</td>
</tr>
<tr>
<td>Ck400</td>
<td>Ω</td>
<td>69</td>
<td>119</td>
</tr>
<tr>
<td>Epb0</td>
<td>Ω</td>
<td>712</td>
<td>194</td>
</tr>
<tr>
<td>Dw8192</td>
<td>Ω</td>
<td>35</td>
<td>50</td>
</tr>
</tbody>
</table>

Table 2.6: Convergence details using various orders of wavelet based preconditioners for Gre-512 matrix.

<table>
<thead>
<tr>
<th>Wavelet of order</th>
<th>DWT</th>
<th>DWTPer</th>
<th>DWTPerMod</th>
</tr>
</thead>
<tbody>
<tr>
<td>Daub-2 with $\alpha = \beta (\leq 30)$</td>
<td>Ω</td>
<td>Ω</td>
<td>Ω</td>
</tr>
<tr>
<td>Daub-4 with $\alpha = \beta = 30$</td>
<td>Ω</td>
<td>Ω</td>
<td>43</td>
</tr>
<tr>
<td>Daub-6 with $\alpha = \beta = 20$</td>
<td>Ω</td>
<td>751</td>
<td>17</td>
</tr>
</tbody>
</table>

2.6 Conclusion and future work

DWTPerMod algorithm improves on other preconditioners providing (a) tighter bounds on the bandwidth for DWTPer band preconditioning, resulting into preconditioning to be done at lower cost; (b) it removes...
uncertainty about choosing an appropriate bandwidth, wavelet level and results into more robust scheme.

Convergence details are presented in Tables 2.3 to 2.6. It is remarkable to observe the rapid convergence of GMRES (25) with preconditioners designed using DWTPerMod compared with other schemes. Preconditioners developed here can also be used for other Krylov subspace iterative methods and no user intervention is required to choose an appropriate transform level for each example. This is a significant advancement towards the development of purely algebraic wavelet based preconditioning strategy for sparse matrices.

In the next chapter, we have presented non orthogonal/biorthogonal wavelet based preconditioners and compare their efficiency with the existing Daubechies orthogonal wavelet based preconditioners for various test sparse matrices.