Computational efficiency of ultrasonic guided wave imaging algorithms

James S. Hall, *Student Member, IEEE,*
and Jennifer E. Michaels, *Senior Member, IEEE*

***Abstract*—Guided wave imaging techniques employed for structural health monitoring (SHM) can be computationally demanding, especially for adaptive techniques such as minimum variance distortionless response (MVDR) imaging, which requires a matrix inversion for each pixel calculation. Instantaneous windowing has been shown in previous work to improve guided wave imaging performance. The use of instantaneous windowing has the additional benefit of significantly reducing the computational requirements of image generation. This paper derives a formulation for MVDR imaging using instantaneous windowing and shows that the matrix inversion associated with MVDR imaging can be optimized, reducing the computational complexity to that of conventional delay-and-sum imaging algorithms. Additionally, a vectorized approach is presented for implementing guided wave imaging algorithms, including delay-and-sum imaging, in matrix-based software packages. The improvements in computational efficiency are quantified by measuring computation time for different array sizes, windowing assumptions, and imaging methods.**

***Index Terms*—minimum variance, MVDR, instantaneous windowing.**

I. Introduction

Guided wave imaging techniques are used in structural health monitoring (SHM) applications to perform damage detection and localization in large, plate-like structures. These techniques have been applied to distributed sparse arrays, where sensors are permanently attached throughout the structure to provide a cost-effective alternative to traditional bulk wave inspection. While most laboratory experiments using distributed sparse arrays are sufficiently small so as to sidestep the issue of computational complexity, the implementation costs and time sensitivity of large, realistic systems can be impacted by the computational demands of guided wave imaging algorithms.

Conventional delay-and-sum imaging algorithms, also known as elliptical imaging algorithms [1], [2], establish pixel values based on the weighted sum of signals within a specific time window. Recent work by the authors [3] has indicated that because guided wave SHM systems are able to reduce noise levels by signal averaging, imaging performance is maximized when the time window is reduced to an instantaneous point in time. In addition to improving imaging performance, however, instantaneous windowing also reduces the computational demands for imaging.

Minimum variance distortionless response (MVDR) [4], [5] has recently been incorporated into conventional delay-and-sum imaging to adaptively reduce the presence of imaging artifacts [3], [6], [7]. While this adaptive technique, referred to as MVDR imaging, improves performance significantly, the performance improvements come at the cost of higher computational demands. For large arrays, the increase in computational requirements and associated increase in implementation cost may outweigh the benefits in imaging performance.

This work was supported by the NASA Graduate Student Research Program, Grant No. NNX08AY93H, and the Air Force Office of Scientific Research, Grant No. FA9550-08-1-0241.

Both authors are with the School of Electrical and Computer Engineering at the Georgia Institute of Technology, 777 Atlantic Dr., NW, Atlanta, GA 30332-0250 USA (phone: 404-894-2994; fax: 404-894-4641; e-mail: jennifer.michaels@ece.gatech.edu).

©2011 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

When instantaneous windowing is used, however, the computational demands of MVDR imaging can be reduced to the same order of magnitude as conventional delay-and-sum imaging. The technique to enable this reduction is presented in this paper and the improvement in computation time is shown as a function of the number of transducers used.

Another significant impact on the computation time of guided wave imaging algorithms is related to the software package used for imaging. Many numerical analysis software packages are tailored for operation with matrices, such as MATLAB® by The Mathworks [8], [9], GNU Octave [10], and SciLab [11]. As such, computation time is reduced when the imaging algorithm is implemented as a series of matrix operations. This paper presents a vectorization method to tailor the delay-and-sum and MVDR imaging algorithms to matrix-oriented software programs and demonstrates the impact that vectorization has on computation time.

This paper is organized as follows. First, the delay-and-sum imaging algorithm is presented and is formulated for the case of instantaneous windowing. MVDR imaging is then similarly presented and reformulated. Next, both imaging algorithms are presented in vectorized format for efficient implementation in matrix-based software packages. Finally, results are presented and a summary is provided.

II. Guided Wave Imaging

Both conventional delay-and-sum and MVDR imaging, as implemented here for SHM, generate an image based on differenced signals. Differenced signals are obtained by subtracting damage-free baseline signals from signals recorded at the time of test. The differencing operation isolates energy that is scattered from defects in the structure. The differenced signals may be handled as raw (RF) differenced signals, complex analytic signals, or the envelope of the analytic signals. The relative computational demands illustrated in this paper are independent of the data format chosen and are applicable to any of the above cases.

To simplify comparisons, a minimally dispersive propagating environment is assumed in which backpropagation corresponds to a simple time-shift. This assumption is reasonable for narrow-band signals in frequency ranges with little dispersion [2], [3].

A. Delay-and-Sum Imaging

This section first introduces the delay-and-sum imaging algorithm in the context of how each pixel is generated, and then simplifies the algorithm under instantaneous windowing assumptions for comparison with MVDR imaging in the following section.

1) General Approach: As shown in Hall and Michaels [3], the pixel value for delay and sum imaging is described as:

$$P_{DS} = \vec{w}_{DS}^{H} \mathbf{R} \vec{w}_{DS}, \tag{1}$$

where $\mathbf{R}$ is a correlation matrix of the backpropagated, differenced signals, the $\vec{w}_{DS}$ vector is a unit-norm vector chosen to maximize the pixel value if damage is present, and the superscript "H" indicates the Hermetian transpose of a matrix or vector. The correlation matrix is calculated as:

$$\mathbf{R} = \sum_{t=t_1}^{t_2} \overrightarrow{r}\,(t)\, \overrightarrow{r}^{\,H}\,(t), \tag{2}$$

where the elements of the $\vec{r}(t)$ vector correspond to the backpropagated, differenced signals:

$$\overrightarrow{r}\,(t) = \left[\begin{array}{ccc} r_1\left(t + \frac{d_1}{c_g}\right) & \cdots & r_M\left(t + \frac{d_M}{c_g}\right) \end{array} \right]^T. \tag{3}$$

In the above equation, c_g is the propagation velocity, $r_m(t)$ is the differenced signal for the m^{th} transmitter-receiver pair, d_m is the

propagation distance from transmitter to pixel location to receiver for the m^{th} transmitter-receiver pair, and M is the number of pairs.

To maximize the pixel value at a damage location, the $\vec{w}_{DS}$ vector must reflect the relationship between signals. If damage is present, then the transmitted signal propagates from the transmitter to the pixel location and interacts with the defect. A scattered signal then propagates from the pixel location to the receiver. Therefore, the relationship between signals is based on the expected scattering characteristics of the damage and the product of the propagation distances:

$$\vec{w}_{DS} \sim \left[\begin{array}{ccc} \frac{\psi_1}{\sqrt{d_1^{\times}}} & \cdots & \frac{\psi_M}{\sqrt{d_M^{\times}}} \end{array} \right]^T, \tag{4}$$

where ψ_m quantifies the scattering behavior of potential damage and $d_m^{\times}$ is the product of the propagation distances from transmitter to pixel location and from pixel location to receiver for the m^{th} transmitter-receiver pair. The square root of the distance product accounts for the geometric spreading of the signal.

2) Instantaneous Windowing: When instantaneous windowing is used, the summation interval in (2) is reduced to a single point in time, τ. In other words,

$$\mathbf{R} = \overrightarrow{r}\left(\tau\right)\overrightarrow{r}^{H}\left(\tau\right), \tag{5}$$

where τ is a time reference that corresponds to the maximum amplitude of the transmitted signal. When the correlation matrix is reduced to the form of (5), the pixel value calculation is similarly reduced to:

$$P_{DS} = \left|\overrightarrow{r}^{H}\left(\tau\right)\overrightarrow{w}_{DS}\right|^2. \tag{6}$$

From (6), the computational complexity for calculating a single pixel value is $O(M)$, where M is the number of transmitter-receiver pairs used for imaging.

B. MVDR Imaging

MVDR imaging is briefly presented in this section. Refer to previous work by the authors [3] for a more thorough treatment of MVDR imaging, including a derivation of the algorithm and discussion of associated implementation issues and expected performance improvements. As in the previous section, after introducing MVDR imaging, the algorithm is reformulated for instantaneous windowing.

1) General Approach: Similar to (1), the pixel values for MVDR imaging are calculated as:

$$P_{MV} = \vec{w}_{MV}^{H}\mathbf{R}\vec{w}_{MV}. \tag{7}$$

However, the $\vec{w}_{MV}$ vector in MVDR imaging is chosen to minimize the pixel value subject to a constraint that preserves pixel values at damage locations. The $\vec{w}_{MV}$ vector is chosen to satisfy the following constrained optimization problem:

$$\vec{w}_{MV} = \arg\min_{\vec{w}} \overrightarrow{w}^{T}\mathbf{R}\overrightarrow{w} \text{ such that } \overrightarrow{w}^{H}\overrightarrow{e} = 1. \tag{8}$$

Here $\vec{e}$ is a unit-norm vector referred to as the steering vector, or look direction. For MVDR imaging, the $\vec{e}$ vector reflects the anticipated relationship between signals if damage is present at the pixel location. Therefore, $\vec{e}$ is identical to $\vec{w}_{DS}$ for conventional delay-and-sum imaging (see (4)). The solution to (8) is found using Lagrange multipliers:

$$\vec{w}_{MV} = \frac{\mathbf{R}^{-1}\overrightarrow{e}}{\overrightarrow{e}^{H}\mathbf{R}^{-1}\overrightarrow{e}}. \tag{9}$$

The above equation indicates that a matrix inversion is required to calculate each pixel value. The matrix inversion operation is responsible for the bulk of the computational complexity of MVDR imaging. Assuming that Gauss-Jordan elimination is used for the matrix inversion, the computational complexity of obtaining each pixel value is $O(M^3)$ [12].

2) Instantaneous Windowing: When instantaneous windowing is used, the correlation matrix, $\mathbf{R}$, takes on a known structure. The eigendecomposition, or spectral decomposition, for $\mathbf{R}$ is simply:

$$\mathbf{R} = \overrightarrow{r}\left(\tau\right)\overrightarrow{r}^{H}\left(\tau\right) = \lambda\overrightarrow{v}\overrightarrow{v}^{H} \tag{10}$$

where $\lambda = \|\vec{r}\left(\tau\right)\|^2$ and $\vec{v} = \vec{r}\left(\tau\right)/\|\vec{r}\left(\tau\right)\|$.

Clearly, the correlation matrix is not full rank when instantaneous windowing is used. In Hall and Michaels [3], diagonal loading is applied to the correlation matrix prior to inversion. The diagonal loading is chosen to be some factor, f, of the largest eigenvalue, which is λ for the instantaneous case considered here. The eigendecomposition of a diagonally loaded instantaneous correlation matrix is:

$$\begin{aligned}\hat{\mathbf{R}} &= \vec{r}\left(\tau\right)\vec{r}^{H}\left(\tau\right) + f\lambda\mathbf{I} \\ &= \left(1+f\right)\lambda\vec{v}\vec{v}^{H} + f\lambda\mathbf{N}\mathbf{N}^{H},\end{aligned} \tag{11}$$

where $\mathbf{I}$ is an $M \times M$ identity matrix and $\mathbf{N}$ is a set of $M-1$ orthonormal vectors spanning the null space of $\vec{v}$. Using the structure of $\hat{\mathbf{R}}$ in (11), the inverse of $\hat{\mathbf{R}}$ is:

$$\hat{\mathbf{R}}^{-1} = \frac{1}{\left(1+f\right)\lambda}\vec{v}\vec{v}^{H} + \frac{1}{f\lambda}\mathbf{N}\mathbf{N}^{H}. \tag{12}$$

Substituting (12) into (9), the optimal weights that satisfy (8) can be expressed as:

$$\overrightarrow{w}_{MV} = \frac{\frac{\vec{v}^{H}\vec{e}}{\left(1+f\right)\lambda}\vec{v} + \frac{1}{f\lambda}\mathbf{N}\mathbf{N}^{H}\vec{e}}{\frac{\left|\vec{v}^{H}\vec{e}\right|^2}{\left(1+f\right)\lambda} + \frac{n^2}{f\lambda}}, \tag{13}$$

where

$$n^2 = \vec{e}^{H}\mathbf{N}\mathbf{N}^{H}\vec{e}. \tag{14}$$

Since $\mathbf{N}$ is a set of orthonormal vectors, $\mathbf{N}^{H}\mathbf{N} = \mathbf{I}$. Therefore, (14) is equivalently:

$$n^2 = \vec{e}^{H}\mathbf{N}\mathbf{N}^{H}\mathbf{N}\mathbf{N}^{H}\vec{e} = \left\|\mathbf{N}\mathbf{N}^{H}\vec{e}\right\|^2. \tag{15}$$

From (15), n^2 is the squared-norm of the projection of $\vec{e}$ onto the null-space of $\vec{v}$. Therefore, n^2 can be calculated directly from $\vec{e}$ and $\vec{v}$ as:

$$n^2 = \left\|\left(\mathbf{I} - \overrightarrow{v}\overrightarrow{v}^{H}\right)\overrightarrow{e}\right\|^2 = \left\|\overrightarrow{e} - \left(\overrightarrow{v}^{H}\overrightarrow{e}\right)\overrightarrow{v}\right\|^2. \tag{16}$$

Substituting (13) into (7) and collecting terms yields

$$P_{MV} = \frac{\lambda\left|\overrightarrow{v}^{H}\overrightarrow{e}\right|^2}{\left(\left|\overrightarrow{v}^{H}\overrightarrow{e}\right|^2 + n^2\frac{1+f}{f}\right)^2}, \tag{17}$$

since $\mathbf{N}^{H}\mathbf{R} = \mathbf{0}$. The above formulation, along with (16), indicates that instantaneous windowing allows each pixel value to be computed without the need to explicitly calculate the correlation matrix, the associated eigenvalues, or a regularized matrix inversion for each pixel. The computational complexity for a single MVDR pixel calculated as in (17) is $O(M)$, which represents a significant improvement over the more general finite window case of $O(M^3)$ and is comparable to the computational requirements of conventional delay-and-sum imaging.

C. Vectorization

The method employed to implement guided wave imaging in matrix-based software packages such as MATLAB® has a dramatic impact on computation time because of the software package's internal structure. It is well-known that matrix-based software packages perform more efficiently with vectorized data [8], [9]. This section constructs the guided wave imaging algorithms discussed in the previous sections in vectorized format to aid the reader in vectorization of this specific problem.

It is important to note that throughout this section, *all matrix operations are performed element-wise*. The nature of the problem does not lend itself to traditional row-column matrix multiplications; rather the matrix structure is employed here to better adapt the problem to the software tool. Also, although the convention in this section is to use two-dimensional (2-D) matrices to store pixel-specific data, the dimensionality of the problem can be further reduced to store the 2-D matrices in a 1-D array. The current format was chosen over the alternative for readability purposes and is not expected to have a noticeable impact on computational requirements.

Let $\mathbf{X}$ and $\mathbf{Y}$ be matrices of x- and y-coordinates, respectively, for each pixel location:

$$\mathbf{X} = \begin{bmatrix} x_0 & x_1 & \cdots \\ \vdots & \vdots & \vdots \\ x_0 & x_1 & \cdots \end{bmatrix}, \mathbf{Y} = \begin{bmatrix} y_0 & \cdots & y_0 \\ y_1 & \cdots & y_1 \\ \vdots & \cdots & \vdots \end{bmatrix}. \tag{18}$$

Then M separate 2-D matrices are generated that correspond to the distance from transmitter to pixel location, $\hat{\mathbf{D}}_m$, and pixel location to receiver, $\check{\mathbf{D}}_m$:

$$\hat{\mathbf{D}}_m = \sqrt{\left(\mathbf{X} - x_{t(m)}\right)^2 + \left(\mathbf{Y} - y_{t(m)}\right)^2} \tag{19}$$

$$\check{\mathbf{D}}_m = \sqrt{\left(\mathbf{X} - x_{r(m)}\right)^2 + \left(\mathbf{Y} - y_{r(m)}\right)^2}, \tag{20}$$

where $x_{t(m)}$, $y_{t(m)}$, $x_{r(m)}$, and $y_{r(m)}$ correspond to the x- and y-coordinates of the transmitter and receiver for the m^{th} transmitter-receiver pair.

The instantaneous windowing assumption allows the backpropagated signals to be obtained simply by selecting a single time sample from the differenced signals:

$$\mathbf{B}_m = r_m\left(\tau + \frac{\hat{\mathbf{D}}_m + \check{\mathbf{D}}_m}{c_g}\right). \tag{21}$$

Here a 2-D matrix, $\mathbf{B}_m$, is constructed in which each element of $\mathbf{B}_m$ corresponds to a specific time in $r_m(t)$ defined by the argument in (21). Note that if a sufficiently high sampling rate is used so that interpolation in the time domain is unnecessary, the argument in (21) can be multiplied by the sampling frequency and rounded to the nearest integer to obtain the desired sample index.

From (21), if the M 2-D matrices are stacked in a third dimension, then $\vec{r}(\tau)$ for each pixel value is stored along the third dimension. An identical structure is used for the $\vec{e}$ and $\vec{v}$ vectors.

The 3-D matrix of $\vec{e}$ vectors for MVDR imaging (or $\vec{w}_{DS}$ vector for conventional delay-and-sum imaging) is determined as in (4). To begin, the unnormalized vectors are calculated:

$$\mathbf{F}_m = \frac{\mathbf{\Psi}_m}{\sqrt{\hat{\mathbf{D}}_m \check{\mathbf{D}}_m}}, \tag{22}$$

where $\mathbf{\Psi}_m$ is a matrix of pixel-specific scattering coefficients for the m^{th} transmitter-receiver pair. The pixel specific norm is then obtained,

$$\|\mathbf{F}\| = \sqrt{\sum_m |\mathbf{F}_m|^2}, \tag{23}$$

and finally the pixel-specific vectors are normalized,

$$\mathbf{E}_m = \frac{\mathbf{F}_m}{\|\mathbf{F}\|}. \tag{24}$$

The conventional delay-and-sum image is generated as

$$\mathbf{P}_{DS} = \left|\sum_m \mathbf{B}_m^* \mathbf{E}_m\right|^2, \tag{25}$$

where * is the element-wise complex conjugate operation.

To calculate the pixel values for an MVDR image, a 2-D matrix of λ values is calculated:

$$\mathbf{\Lambda} = \sum_m |\mathbf{B}_m|^2. \tag{26}$$

The $\mathbf{\Lambda}$ matrix is then used to obtain the eigenvectors:

$$\mathbf{V}_m = \frac{\mathbf{B}_m}{\sqrt{\mathbf{\Lambda}}}. \tag{27}$$

For convenience, two intermediate matrices are calculated:

$$\mathbf{P} = \sum_m \mathbf{V}_m^* \mathbf{E}_m \tag{28}$$

$$\mathbf{S} = \sum_m |\mathbf{E}_m - \mathbf{P}\mathbf{V}_m|^2 \tag{29}$$

which correspond to the $\vec{v}^H\vec{e}$ and n^2 terms, respectively for the single-pixel case in (17). Finally, the MVDR pixel value is calculated using element-wise matrix operations:

$$\mathbf{P}_{MV} = \frac{\mathbf{\Lambda}\,|\mathbf{P}|^2}{\left||\mathbf{P}|^2 + \left(\frac{1+f}{f}\right)\mathbf{S}\right|^2}. \tag{30}$$

Equations (25) and (30) reflect the vectorized conventional delay-and-sum and MVDR imaging algorithms, respectively, with instantaneous windowing. Note that complete vectorization of the MVDR imaging algorithm is not possible without the instantaneous windowing optimization presented in Section II-B2. Vectorization is expected to reduce the computation time required for image generation when using matrix-oriented numerical analysis software.

III. Computational Results

Simulation data were used to verify the computational complexity of the proposed methods above. Guided wave images were generated using 2 to 24 transducers for five separate cases: (1) MVDR imaging with traditional matrix inversion computed with for-loops, (2) MVDR imaging optimized for instantaneous windowing computed with for-loops, (3) MVDR imaging optimized for instantaneous windowing and computed with vectorized data, (4) conventional delay-and-sum imaging computed with for-loops, and (5) conventional delay-and-sum imaging computed with vectorized data. For comparison purposes, all images were created using instantaneous windowing, meaning that the correlation matrix (if calculated) is constructed as in (5). Mathworks' MATLAB® was used to generate the images using a Hewlett-Packard laptop with an Intel® Core™2 Duo CPU operating at 2.26 GHz with 4 GB of RAM and running Windows® Vista Home Premium. Each image was composed of 7744 pixels, corresponding to a 610 x 610 mm plate imaged with pixels spaced 7 mm apart. Images were each generated 20 separate times and the average computation time was recorded.

Figure 1 depicts computation time as a function of the number of transducers. Several features of Figure 1 are worth noting. First, the optimization for MVDR imaging presented in Section II-B2 significantly reduces the computational requirements of MVDR imaging (broken lines) to the point that it can be performed in a comparable amount of time as conventional delay-and-sum imaging (solid lines). As mentioned earlier, matrix inversion requires $O\left(M^3\right)$ operations, which is compounded by the fact that if N transducers are used for imaging, the number of *pairs* of transducers, $M = N(N-1)/2$. As a result, the computation time for MVDR imaging without optimization grows much faster than any of the other cases as the number of transducers is increased. In contrast, the computation time for MVDR imaging optimized for instantaneous windowing is a constant multiple of the computation time required for conventional delay-and-sum imaging. Another important observation is that, as expected,

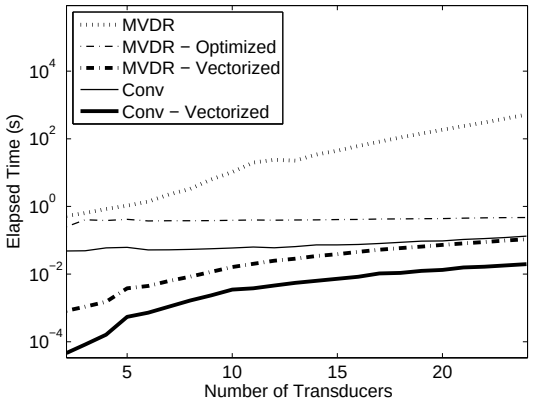

Fig. 1. Computation time *vs*. number of transducers for five separate cases: (1) MVDR imaging with traditional matrix inversion computed with for-loops, (2) MVDR imaging optimized for instantaneous windowing computed with for-loops, (3) MVDR imaging optimized for instantaneous windowing and computed with vectorized data, (4) conventional delay-and-sum imaging computed with for-loops, and (5) conventional delay-and-sum imaging computed with vectorized data.

vectorizing the imaging algorithms further reduces computation time. The initial offset in computation time between images generated with for-loops *vs*. vectorized data is attributed to overhead costs from setup of the for-loop operation. As the number of transducers is increased, the difference in computation time is expected to continue to decrease because the overhead associated with for-loop operation becomes small compared to image computational requirements.

IV. Summary

This paper demonstrates that (1) when instantaneous windowing is used, the computational demands for MVDR imaging are comparable to those of conventional delay-and-sum imaging, and (2) vectorizing the imaging algorithm can have a significant impact on computation time. Both conventional delay-and-sum imaging and MVDR imaging algorithms are presented and reformulated for the case when instantaneous windowing is used. Both imaging algorithms are then vectorized for efficient implementation in matrix-based software packages, such as MATLAB® by The Mathworks. Computational improvements are demonstrated by showing the computation time of each algorithm as a function of the number of transducers used for imaging.

The primary contributions of this paper include identifying and quantifying the computational improvements that are obtained by (1) using instantaneous windowing, and (2) tailoring the imaging problem for matrix-based software packages. The reduction in computational demands obtained from instantaneous windowing optimization enables the use of MVDR imaging, with its associated benefits in imaging performance, at computational costs comparable to conventional delay-and-sum imaging.

References

[1] C. H. Wang, J. T. Rose, and F.-K. Chang, "A synthetic time-reversal imaging method for structural health monitoring," *Smart Mater. Struct.*, vol. 13, pp. 415–423, 2004.

[2] J. E. Michaels, "Detection, localization and characterization of damage in plates with an *in situ* array of spatially distributed sensors," *Smart Mater. Struct.*, vol. 17, no. 035035, 2008.

[3] J. S. Hall and J. E. Michaels, "Minimum variance ultrasonic imaging applied to an in situ sparse guided wave array," *IEEE Trans. Ultrason., Ferroelectr., Freq. Control*, vol. 57, no. 10, pp. 2311–2323, 2010.

[4] O. L. Frost, "An algorithm for linearly constrained adaptive array processing," *Proc. IEEE*, vol. 60, no. 8, pp. 926–935, 1972.

[5] J. Capon, "High-resolution frequency-wavenumber spectrum analysis," *Proc. IEEE*, vol. 57, no. 8, pp. 1408–1418, 1969.

[6] J. E. Michaels, J. S. Hall, G. Hickman, and J. Krolik, "Sparse array imaging of change-detected signals by minimum variance processing," in *Review of Progress in QNDE*, vol. 28, 2009, pp. 642–649.

[7] J. E. Michaels, J. S. Hall, and T. E. Michaels, "Adaptive imaging of damage from changes in guided wave signals recorded from spatially distributed arrays," *Proc. SPIE*, vol. 7295, no. 729515, pp. 1–11, 2009.

[8] Mathworks, "Techniques for improving performance: vectorizing loops," *MATLAB User Guide*, 2010.

[9] ——, "Code vectorization guide," *MATLAB Product Support*, 2010.

[10] J. W. Eaton, D. Bateman, and S. Hauberg, *GNU Octave Manual Version 3*. Bristol: Network Theory Ltd., 2008.

[11] L. E. van Dijk and C. L. Spiel, *SciLab Bag of Tricks: The SciLab-2.5 Infrequently Asked Questions*, 2000.

[12] G. Hammerlin and K.-H. Hoffmann, *Numerical Mathematics*, ser. Undergraduate Texts in Mathematics: Readings in Mathematics. New York: Springer, 1991.