Efficient time-domain simulation of nonlinear, state-space, transmission-line models of the cochlea (L)

Shuokai Pan[a] and Stephen J. Elliott
Institute of Sound and Vibration Research, University of Southampton, Highfield Campus, Southampton SO17 1BJ, United Kingdom

Paul D. Teal
School of Engineering and Computer Science, Victoria University of Wellington, P.O. Box 600, Wellington 6140, New Zealand

Ben Lineton
Institute of Sound and Vibration Research, University of Southampton, Highfield Campus, Southampton SO17 1BJ, United Kingdom

(Received 18 February 2015; revised 1 May 2015; accepted 8 May 2015)

Nonlinear models of the cochlea are best implemented in the time domain, but their computational demands usually limit the duration of the simulations that can reasonably be performed. This letter presents a modified state space method and its application to an example nonlinear one-dimensional transmission-line cochlear model. The sparsity pattern of the individual matrices for this alternative formulation allows the use of significantly faster numerical algorithms. Combined with a more efficient implementation of the saturating nonlinearity, the computational speed of this modified state space method is more than 40 times faster than that of the original formulation.
 [http://dx.doi.org/10.1121/1.4921550]

[BLM] Pages: 3559–3562

I. INTRODUCTION

Active nonlinearity is an essential feature of cochlear mechanics that is generally best modelled in the time domain. A range of time domain numerical methods have been proposed and applied to a number of models over the years (Diependaal *et al.*, 1987; Elliott *et al.*, 2007). One widely adopted two-stage strategy (Diependaal *et al.*, 1987) first solves the model boundary value problem using the finite difference approximation, in order to compute the pressure difference, and then uses numerical integration techniques to solve the remaining initial value problem. More recently, a state space matrix formulation has been proposed in Elliott *et al.* (2007), which has also been applied in several studies such as Liu and Neely (2010), Sisto *et al.* (2010), Rapson *et al.* (2012), and Ayat *et al.* (2014). It expresses the model states as a single set of coupled first-order ordinary differential equations (ODEs), the time domain solution of which can then be obtained using well-established ODE solvers. An extra benefit of this approach is that it allows rigorous inspection of the stability of linearized models, which, for instance, facilitates the study of spontaneous otoacoustic emissions (Ku *et al.*, 2009). The relation between the state space formulation and Diependaal's method has previously been discussed in Rapson *et al.* (2012).

Despite the advantages of time domain solutions, they are usually very computationally challenging and memory demanding, because a sufficient spatial and time domain resolution is required to ensure simulation accuracy. Taking the standard state space method as an example, it is typically thousands of times slower than real time when implemented in the time domain on a desktop computer. A hybrid direct-iterative solver was developed in Bertaccini and Sisto (2011) to accelerate the simulation of a nonlinear feed-forward model within the standard state space framework. This letter proposes a modified state space (MSS) method for the time domain simulation of nonlinear one-dimensional (1D) transmission-line cochlear models. Numerical efficiency of this MSS is compared with both the standard state space (SS) and Diependaal's method, and it is found to be more than 40 times more efficient than the SS and marginally faster than Diependaal's method.

II. THE EXAMPLE COCHLEAR MODEL

The example model adopted here for comparison of algorithm complexity is that reported in Ku *et al.* (2009), which is a nonlinear adaptation to the linear active model originally proposed by Neely and Kim (1986) for a cat cochlea, but with parameters re-tuned to match human cochlear physiology. The micromechanics of this model is represented by an array of 500 discrete elements, with the first one modelling the middle ear, the last one the helicotrema and the remaining 498 elements modelling the cochlear partition, as shown in Fig. 4 of Elliott *et al.* (2007). The cochlear partition elements are modelled as two degree-of-freedom (DOF) lumped-parameter oscillators [Fig. 1 of Elliott *et al.* (2007)], coupled solely by the cochlear fluid. The saturating active mechanism is simulated by placing a compressive nonlinearity, a first-order Boltzmann function, before the active impedance in the micromechanical feedback loop, so that the input to the Boltzmann function is the difference between the displacements of the two masses of each oscillator. The dimensionless factor, γ, which regulates the strength of the active feedback

[a]Electronic mail: sp2g12@soton.ac.uk

force, is then determined as the absolute value of the ratio of the output and input of the Boltzmann function, as shown in Eq. (8) of Ku *et al.* (2009). A two-port network model of the ear canal and middle ear, based on the model of Kringlebotn (1988) and programmed by Ku (2008), is also included. A thorough description of the model and its parameter values can be found in Ku (2008).

III. MODIFIED STATE SPACE FORMULATION

The original state space formulation proposed by Elliott *et al.* (2007) used the finite difference approximation for the 1D wave propagation inside the cochlea, Eq. (1), and state space formulation for the element micromechanics, Eqs. (2) and (3), so that the final fluid-coupled state space equation can be derived as Eq. (4),

$$\mathbf{F}\mathbf{p}(t) - \ddot{\mathbf{w}}(t) = \mathbf{q}(t), \tag{1}$$

$$\dot{\mathbf{x}}(t) = \mathbf{A}_E\mathbf{x}(t) + \mathbf{B}_E\mathbf{p}(t), \tag{2}$$

$$\dot{\mathbf{w}}(t) = \mathbf{C}_E\mathbf{x}(t), \tag{3}$$

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{q}(t), \tag{4}$$

where $\mathbf{F}$ is the second-order finite difference matrix, representing the coupling between the cochlear partition acceleration vector $\ddot{\mathbf{w}}(t)$ and fluid pressure difference vector $\mathbf{p}(t)$; $\mathbf{q}(t)$ is the stapes acceleration vector, the only nonzero entry of which is the first one; $\mathbf{x}(t)$ is the complete state vector; $\mathbf{A} = [\mathbf{I} - \mathbf{B}_E\mathbf{F}^{-1}\mathbf{C}_E]^{-1}\mathbf{A}_E$ is the system matrix; and $\mathbf{B} = [\mathbf{I} - \mathbf{B}_E\mathbf{F}^{-1}\mathbf{C}_E]^{-1}\mathbf{B}_E\mathbf{F}^{-1}$ is the input matrix. Complete derivations and definitions of these matrices can be found in Elliott *et al.* (2007). An alternative method for deriving the final state space equation, with much better computational efficiency, is to first combine Eqs. (2) and (3) to eliminate variable $\dot{\mathbf{x}}(t)$, and then combine the result with Eq. (1) to eliminate dependence on $\ddot{\mathbf{w}}(t)$,

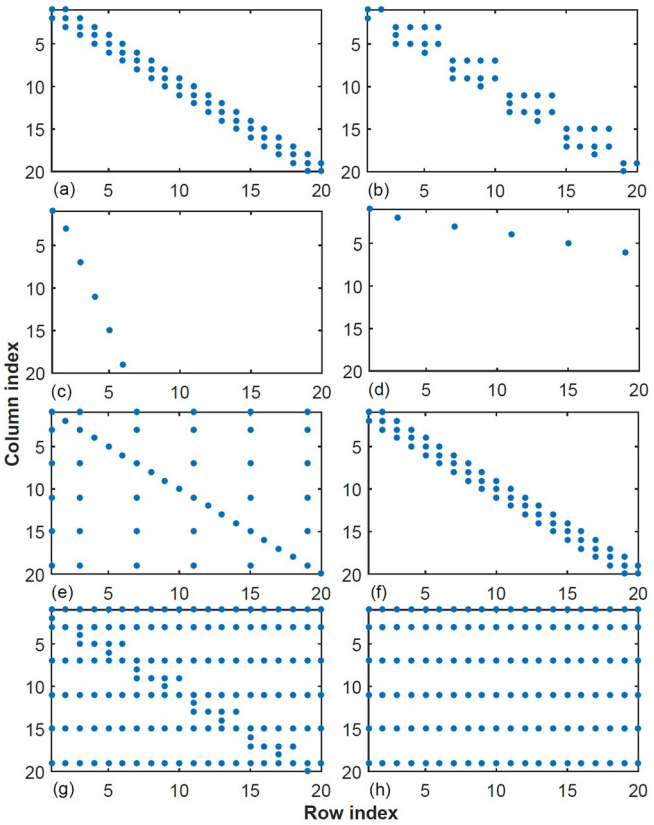

FIG. 1. (Color online) Sparsity patterns, sizes, nnz, and matrix densities of state space matrices. Only the first 20×20 elements of each matrix are displayed for better visualization of their internal patterns. (a) $\mathbf{F}$, 500×500, nnz $= 1498$, $\delta \approx 0.599\%$; (b) $\mathbf{A}_E$, 1996×1996, nnz $= 4985$, $\delta \approx 0.125\%$; (c) $\mathbf{B}_E$, 1996×500, nnz $= 500$, $\delta \approx 0.0501\%$; (d) $\mathbf{C}_E$, 500×1996, nnz $= 500$, $\delta \approx 0.0501\%$; (e) $\mathbf{I} - \mathbf{B}_E\mathbf{F}^{-1}\mathbf{C}_E$, 1996×1996, nnz $= 251496$, $\delta \approx 6.31\%$; (f) $\mathbf{F} - \mathbf{C}_E\mathbf{B}_E$, 500×500, nnz $= 1498$, $\delta \approx 0.599\%$; (g) $\mathbf{A}$ or $\mathbf{A}'$, 1996×1996, nnz $= 1\,000\,490$, $\delta \approx 25.1\%$; and (h) $\mathbf{B}$ or $\mathbf{B}'$, 1996×500, and nnz $= 250\,000$, $\delta \approx 25.05\%$.

$$\ddot{\mathbf{w}}(t) = \mathbf{C}_E\dot{\mathbf{x}}(t) = \mathbf{C}_E[\mathbf{A}_E\mathbf{x}(t) + \mathbf{B}_E\mathbf{p}(t)], \tag{5}$$

$$\ddot{\mathbf{w}}(t) = \mathbf{F}\mathbf{p}(t) - \mathbf{q}(t). \tag{6}$$

Equating the right hand side of Eqs. (5) and (6) leads to the pressure difference vector as

$$\mathbf{p}(t) = (\mathbf{F} - \mathbf{C}_E\mathbf{B}_E)^{-1}[\mathbf{C}_E\mathbf{A}_E\mathbf{x}(t) + \mathbf{q}(t)]. \tag{7}$$

Substituting Eq. (7) back into Eq. (2), the modified state space equation is obtained as

$$\dot{\mathbf{x}}(t) = \mathbf{A}'\mathbf{x}(t) + \mathbf{B}'\mathbf{q}(t), \tag{8}$$

where the new system matrix becomes $\mathbf{A}' = \mathbf{A}_E + \mathbf{B}_E(\mathbf{F} - \mathbf{C}_E\mathbf{B}_E)^{-1}\mathbf{C}_E\mathbf{A}_E$; and new input matrix is $\mathbf{B}' = \mathbf{B}_E(\mathbf{F} - \mathbf{C}_E\mathbf{B}_E)^{-1}$. This new set of state space matrices $\mathbf{A}'$, $\mathbf{B}'$ can be shown to be analytically identical to the original ones $\mathbf{A}$, $\mathbf{B}$, using the Woodbury matrix identity for the inverse of a sum of matrices (Henderson and Searle, 1981). Applying this identity to the original system matrix $\mathbf{A}$, we have

$$\begin{aligned}\mathbf{A} &= [\mathbf{I} - \mathbf{B}_E\mathbf{F}^{-1}\mathbf{C}_E]^{-1}\mathbf{A}_E \\ &= [\mathbf{I} + \mathbf{B}_E(\mathbf{F} - \mathbf{C}_E\mathbf{B}_E)^{-1}\mathbf{C}_E]\mathbf{A}_E = \mathbf{A}'.\end{aligned} \tag{9}$$

Similarly, for the original input matrix $\mathbf{B}$, we have

$$\begin{aligned}\mathbf{B} &= [\mathbf{I} - \mathbf{B}_E\mathbf{F}^{-1}\mathbf{C}_E]^{-1}\mathbf{B}_E\mathbf{F}^{-1} \\ &= [\mathbf{I} - \mathbf{B}_E\mathbf{F}^{-1}\mathbf{C}_E]^{-1}\mathbf{B}_E\mathbf{F}^{-1}\mathbf{C}_E\mathbf{C}_E^{-1} \\ &= [\mathbf{I} - \mathbf{B}_E\mathbf{F}^{-1}\mathbf{C}_E]^{-1}[\mathbf{I} - (\mathbf{I} - \mathbf{B}_E\mathbf{F}^{-1}\mathbf{C}_E)]\mathbf{C}_E^{-1} \\ &= [\mathbf{I} - \mathbf{B}_E\mathbf{F}^{-1}\mathbf{C}_E]^{-1}\mathbf{C}_E^{-1} - \mathbf{C}_E^{-1} \\ &= [\mathbf{I} + \mathbf{B}_E(\mathbf{F} - \mathbf{C}_E\mathbf{B}_E)^{-1}\mathbf{C}_E]\mathbf{C}_E^{-1} - \mathbf{C}_E^{-1} \\ &= \mathbf{B}_E(\mathbf{F} - \mathbf{C}_E\mathbf{B}_E)^{-1} = \mathbf{B}'.\end{aligned} \tag{10}$$

IV. NUMERICAL ADAVANTAGE OF THE MODIFIED STATE SPACE METHOD

As shown in Eqs. (10), (21)–(23) of Elliott *et al.* (2007), $\mathbf{F}$ is a tri-diagonal matrix (i.e., having nontrivial elements only on the diagonal, first sub-diagonal and first super-diagonal), $\mathbf{A}_E$, $\mathbf{B}_E$, and $\mathbf{C}_E$ are block diagonal matrices (in which the diagonal elements are square matrices of any size and off-diagonal elements are zero). It is therefore expected that considerable memory and calculation savings can be achieved by employing sparse matrix storage and computation methods, especially when several hundred micro-elements are used. The sparsity patterns of the state space matrices are shown in Fig.

1, where the upper two rows show the individual matrices and the bottom two rows both those inverted and the final matrices in the state space formulation. Each square represents the structure of a matrix and each dot denotes a nonzero element at the corresponding location. Only the first 20×20 elements of each matrix are depicted for better visualization of their internal patterns. The size of each matrix for a 500-element model, the number of nonzero entries (nnz) and sparse matrix density, δ, defined as the ratio of nnz to the total number of entries, are also given in the figure caption. It can be seen that more than 99% of the matrices $\mathbf{A}_E$, $\mathbf{B}_E$, $\mathbf{C}_E$, $\mathbf{F}$, and $\mathbf{F}-\mathbf{C}_E\mathbf{B}_E$ are empty, whereas this number reduces to about 75% for the final set of state space matrices $\mathbf{A}$ and $\mathbf{B}$. In MATLAB, sparse algorithms (Gilbert *et al.*, 1992) can be employed when these matrices are defined to be sparse and the resultant sparse state space method is about twice as efficient as the original formulation. The modified state space formulation, however, can be arranged to be even faster with sparse algorithms, because the intermediate matrix $\mathbf{F}-\mathbf{C}_E\mathbf{B}_E$ is tri-diagonal and the calculation of the pressure difference vector $\mathbf{p}(t)$ in Eq. (7) is identical to solving a sparse tri-diagonal linear system of the form $Ax=b$. There exist very rapid algorithms for computing the solution of such linear system using either iterative or direct methods. Direct solvers based upon sparse matrix factorization are preferable in this case due to the dimension of the model and the nature of matrix $\mathbf{F}-\mathbf{C}_E\mathbf{B}_E$. In MATLAB, the matrix left divide operator, mldivde or backslash (\), encapsulates a host of algorithms for solving sparse linear systems and invokes the most suitable one according to the sparsity pattern of involved matrices (Davis, 2006). One solver is specifically designed for tri-diagonal systems, and direct use of this algorithm leads to a speed improvement of at least 20 times compared to the original formulation. But since the tri-diagonal matrix, $\mathbf{F}-\mathbf{C}_E\mathbf{B}_E$, is fixed during the entire simulation, the speed of solving this linear system can be further increased by performing a sparse LU decomposition beforehand. Essentially, this converts the tri-diagonal system into a combination of a lower and an upper triangular system which can be easily solved using forward and backward substitution. According to (Davis, 2004), this decomposition for the matrix $\mathbf{F}-\mathbf{C}_E\mathbf{B}_E$ takes the form of

$$\mathbf{F}-\mathbf{C}_E\mathbf{B}_E=\mathbf{R}\mathbf{P}^{-1}\mathbf{L}\mathbf{U}\mathbf{Q}^{-1}, \tag{11}$$

$$(\mathbf{F}-\mathbf{C}_E\mathbf{B}_E)^{-1}=\mathbf{Q}\mathbf{U}^{-1}\mathbf{L}^{-1}\mathbf{P}\mathbf{R}^{-1}, \tag{12}$$

where $\mathbf{L}$ is a lower triangular matrix, $\mathbf{U}$ is an upper triangular matrix, $\mathbf{P}$ and $\mathbf{Q}$ are row and column permutation matrices used to reduce the fill-in problem during the actual factorization process, and $\mathbf{R}$ is a diagonal row scaling matrix which can lead to a sparser and more stable factorization. Thus, in addition to using sparse matrix multiplication, the modified state space equation is implemented as the following:

$$\dot{\mathbf{x}}(t)=\mathbf{A}_E\mathbf{x}(t)+\bar{\mathbf{B}}\{\mathbf{U}^{-1}[\mathbf{L}^{-1}(\bar{\mathbf{C}}\mathbf{A}_E\mathbf{x}(t))]\}+\bar{\mathbf{q}}(t), \tag{13}$$

where $\bar{\mathbf{B}}=\mathbf{B}_E\mathbf{Q}$, $\bar{\mathbf{C}}=\mathbf{P}\mathbf{R}^{-1}\mathbf{C}_E$, and $\bar{\mathbf{q}}(t)=\mathbf{B}_E(\mathbf{F}-\mathbf{C}_E\mathbf{B}_E)^{-1}\mathbf{q}(t)$, all of which can be computed before calling the ODE solver. Such computational advantages are not possible with the original state space method, as its intermediate matrix, $\mathbf{I}-\mathbf{B}_E\mathbf{F}^{-1}\mathbf{C}_E$ contains significantly more nonzero entries (more than 167 times) and is of larger size (about 16 times) than those of $\mathbf{F}-\mathbf{C}_E\mathbf{B}_E$. The reduction of sparsity mainly results from $\mathbf{F}^{-1}$, which is a completely full matrix.

V. INCLUSION OF NONLINEARITY IN THE STATE SPACE EQUATIONS

Only the active feedback force needs to be compressed with stimulus level, so the matrix $\mathbf{A}_E$ can be decomposed into a time-invariant passive part, $\mathbf{A}_{E_{\text{pas}}}$ and a time-varying active part, $\mathbf{A}_{E_{\text{act}}}(\gamma)$, which consists of all the components that are functions of γ. One way of implementing the level-dependent nonlinearity in the time domain is to update $\mathbf{A}_{E_{\text{act}}}(\gamma)$ every time γ changes. This involves several modifications to a 1996×1996 matrix for every time step when using the ODE solver and can be extremely time-consuming. By observing that each individual block matrix inside $\mathbf{A}_{E_{\text{act}}}(\gamma)$ only contains nonzero entries on its first row, which also have a common factor γ, an equivalent implementation is to scale all the components of the state vector for each cochlear partition element by a factor $\gamma(n,t)$, which is given by

$$\gamma(n,t)=\left|\frac{f[x_d(n,t)]}{x_d(n,t)}\right|, \tag{14}$$

where n is the element index; $x_d(n,t)$ is the relative displacement between the BM and TM; and $f(x)$ is the Boltzmann function. The active part of the state space equation, $\mathbf{A}_{E_{\text{act}}}(\gamma)\mathbf{x}(t)$, can now be written as $\mathbf{A}_{E_{\text{full}}}\mathbf{x}_{\text{scaled}}(t)$, where $\mathbf{A}_{E_{\text{full}}}=\mathbf{A}_{E_{\text{act}}}(\gamma=1)$, is a constant matrix, $\mathbf{x}_{\text{scaled}}(t)=\boldsymbol{\gamma}(t)\odot\mathbf{x}(t)$, $\boldsymbol{\gamma}(t)$ is a column vector including all of the scaling factors, $\gamma(n,t)$, arranged in the same order as are the elements of the complete state vector $\mathbf{x}(t)$ and the symbol $\odot$ denotes element-by-element multiplication of the two column vectors $\boldsymbol{\gamma}(t)$ and $\mathbf{x}(t)$. Therefore, the final fluid-coupled state space equation for the nonlinear cochlear model is realized as the following:

$$\begin{aligned}\dot{\mathbf{x}}(t)=&\mathbf{A}_{E_{\text{pas}}}\mathbf{x}(t)+\mathbf{A}_{E_{\text{full}}}\mathbf{x}_{\text{scaled}}(t)+\bar{\mathbf{q}}(t)\\&+\bar{\mathbf{B}}\{\mathbf{U}^{-1}[\mathbf{L}^{-1}(\bar{\mathbf{C}}(\mathbf{A}_{E_{\text{pas}}}\mathbf{x}(t)+\mathbf{A}_{E_{\text{full}}}\mathbf{x}_{\text{scaled}}(t)))]\}.\end{aligned} \tag{15}$$

This method is considerably more effective than the first one, as none of the state space matrices is changed during the numerical integration. The original SS method for the nonlinear model is implemented in a similar way for comparison of numerical efficiency.

VI. RESULTS

This section presents a comparison of the computational efficiency of four time domain numerical algorithms: the SS, sparse state space (SSS), modified state space (MSS), and Diependaal's two-stage method after extending it to allow two DOF micromechanics following the Appendix of Diependaal *et al.* (1987). All of these were programmed and simulated in MATLAB R2015a using a desktop computer with a 3.40 GHz, quad-core Intel Core i5-3570 processor and 4 GB DDR3 RAM. The final component of all four algorithms is numerical integration. We implemented an explicit adaptive solver, which is a modified version of the MATLAB ode45 function,

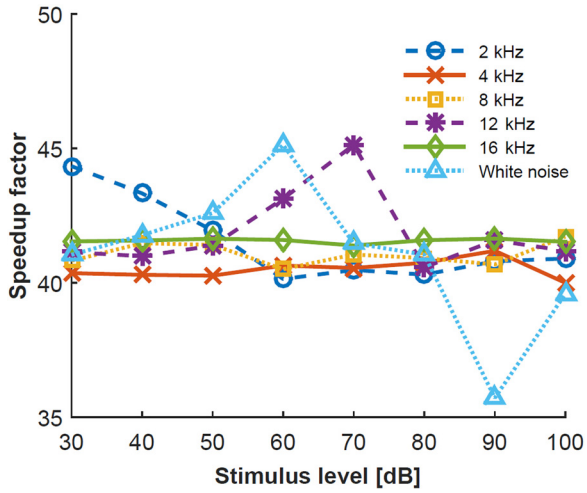

FIG. 2. (Color online) Dependence of speedup factor for the modified state space method on stimulus SPL for white noise and sinusoidal stimuli with frequencies of 2, 4, 8, 12, and 16 kHz.

removing unnecessary functionalities such as event function, mass matrix and non-negative solutions. The relative error tolerance was set as 10^{-5} in all experiments, whereas the absolute error tolerance was determined individually for each stimulus to be one order of magnitude lower than the corresponding scale of the model displacement response. This reflects the flexibility of the adaptive solver, which can solve the model with the required accuracy and without unnecessary effort. The effect of the stimulus spectrum and level on program runtime was investigated by using white noise and sinusoids having frequencies of 2, 4, 8, 12, and 16 kHz, each of which was sampled at 100 kHz, and had a duration of 30 ms with a 10 ms half-Hanning window onset ramp and levels varying from 30 to 100 dB sound pressure level (SPL) in step of 10 dB.

For each stimulus, the simulation time gradually rises with increasing input level and frequency for sinusoidal stimuli. But the speedup factor, defined as the ratio of the SS simulation time to that of any other method, is roughly constant across most stimulus types and levels, as shown in Fig. 2 for the MSS. Similarly, the speedup factor is found to be relatively independent of the number of microelements in the model (not shown). Table I shows the average runtime across different signal levels and frequencies of sinusoidal excitations for each type of stimulus and algorithm. The overall average runtime of all stimuli are taken as the final metric for comparison of the computational efficiency of each method and the overall speedup factor is shown in the last row of Table I. It can be seen that the sparse algorithms alone are able reduce the runtime by almost a half, but the MSS yields a speed improvement of more than a factor of 40 and is also about 40% quicker than the two-stage method.

TABLE I. Comparison of average run-time in seconds, used by four algorithms to solve the nonlinear 1D cochlear model with 500 discrete elements in response to white noise and sinusoidal stimuli with frequencies of 2, 4, 8, 12, and 16 kHz, each of which was sampled at 100 kHz, had a duration of 30 ms with a 10 ms half-Hanning window onset ramp and levels varying from 30 to 100 dB SPL in step of 10 dB. SS: original dense state space; SSS: sparse state space; MSS: modified state space; two-stage: Diependaal's method.

	SS	SSS	MSS	Two-stage
Sinusoids	777.02	427.50	18.80	25.52
White noise	630.54	361.85	15.52	22.27
Average runtime	703.78	394.68	17.16	23.90
Speedup factor	1	1.78	41.01	29.45

VII. CONCLUSIONS

The main contribution of this paper is the description of the modified state space method, which has been applied here to the nonlinear and active cochlear model developed in Ku *et al.* (2009). Although analytically identical to the original state space equation, the sparsity pattern of the constituting matrix of this alternative formulation offers the opportunity for considerably more efficient numerical algorithms, producing a speedup factor of more than 40. Its computational efficiency is on a similar scale to that of the Diependaal's method. The approach presented here can be readily applied to various other 1D cochlear models, such as the nonlinear and active ones based on an array of one DOF lumped-parameter oscillators (Sisto *et al.*, 2010).

ACKNOWLEDGMENT

The Ph.D. work of S.P. is supported by Cirrus Logic.

Ayat, M., Teal, P. D., and McGuinness, M. (**2014**). "An integrated electro-mechanical model for the cochlear microphonic," Biocybern. Biomed. Eng. **34**, 206–219.
Bertaccini, D., and Sisto, R. (**2011**). "Fast numerical solution of nonlinear nonlocal cochlear models," J. Comput. Phys. **230**, 2575–2587.
Davis, T. A. (**2004**). "A column pre-ordering strategy for the unsymmetric-pattern multifrontal method," ACM Trans. Math. Software **30**, 165–195.
Davis, T. A. (**2006**). *Direct Methods for Sparse Linear Systems* (Society for Industrial and Applied Mathematics, Philadelphia, PA), Chap. 8, pp. 135–144).
Diependaal, R. J., Duifhuis, H., Hoogstraten, H., and Viergever, M. A. (**1987**). "Numerical methods for solving one-dimensional cochlear models in the time domain," J. Acoust. Am. Soc. **82**, 1655–1666.
Elliott, S. J., Ku, E. M., and Lineton, B. (**2007**). "A state space model for cochlear mechanics," J. Acoust. Am. Soc. **122**, 2759–2771.
Gilbert, J. R., Moler, C., and Schreiber, R. (**1992**). "Sparse matrices in MATLAB: Design and implementation," SIAM J. Matrix Anal. Appl. **13**, 333–356.
Henderson, H. V., and Searle, S. R. (**1981**). "On deriving the inverse of a sum of matrices," SIAM Rev. **23**, 53–60.
Kringlebotn, M. (**1988**). "Network model for the human middle ear," Scand. Audiol. **17**, 75–85.
Ku, E. M. (**2008**). "Modelling the human cochlea," Ph.D. thesis, University of Southampton, Southampton, UK.
Ku, E. M., Elliott, S. J., and Lineton, B. (**2009**). "Limit cycle oscillations in a nonlinear state space model of the human cochlea," J. Acoust. Am. Soc. **126**, 739–750.
Liu, Y.-W., and Neely, S. T. (**2010**). "Distortion product emissions from a cochlear model with nonlinear mechanoelectrical transduction in outer hair cells," J. Acoust. Am. Soc. **127**, 2420–2432.
Neely, S. T., and Kim, D. (**1986**). "A model for active elements in cochlear biomechanics," J. Acoust. Soc. Am. **79**, 1472–1480.
Rapson, M. J., Tapson, J. C., and Karpul, D. (**2012**). "Unification and extension of monolithic state space and iterative cochlear models," J. Acoust. Soc. Am. **131**, 3935–3952.
Sisto, R., Moleti, A., Paternoster, N., Botti, T., and Bertaccini, D. (**2010**). "Different models of the active cochlea, and how to implement them in the state-space formalism," J. Acoust. Soc. Am. **128**, 1191–1202.