## 5 Conclusions

This paper introduces the velocity-dependent friction with the Stribeck effect into the moving load model for the vibration of a car disc brake. By solving its corresponding eigenvalue problem, a bounded region of instability is obtained for the rotating speed of the disc versus the friction coefficient at the disc/pads interface, which is compatible with observed squeal phenomenon of a car disc brake.


### Acknowledgments

This research is supported by the Engineering and Physical Sciences Research Council of UK (grant number GR/L91061), TMD Friction UK and TRW Automotive.


### References


[1] Yang, S., and Gibson, R. F., 1997, ''Brake Vibration and Noise: Review, Comments and Proposals,'' Int. J. of Materials and Product Tech., **12**, pp. 496–513.
[2] Mottershead, J. E., 1998, ''Vibration and Friction-Induced Instability in Discs,'' Shock Vib. Dig., **30**, pp. 14–31.
[3] Nishiwaki, M., 1993, ''Generalised Theory of Brake Noise,'' Proc. Inst. Mech. Eng., Part D (J. Automob. Eng.), **207**, pp. 195–202.
[4] Chan, S. N., Mottershead, J. E., and Cartmell, M. P., 1994, ''Parametric Resonances at Subcritical Speeds in Discs with Rotating Frictional Loads,'' Proc. Inst. Mech. Eng., Part C: J. Mech. Eng. Sci., **208**, pp. 417–425.
[5] Mottershead, J. E., Ouyang, H., Cartmell, M. P., and Friswell, M. I., 1997, ''Parametric Resonances in an Annular Disc, with a Rotating System of Distributed Mass and Elasticity; and the Effects of Friction and Damping,'' Proc. R. Soc. London, Ser. A, **453**, pp. 1–19.
[6] Ouyang, H., Mottershead, J. E., Cartmell, M. P., and Friswell, M. I., 1998, ''Friction-Induced Parametric Resonances in Discs: Effect of a Negative Friction-Velocity Relationship,'' J. Sound Vib., **209**, pp. 251–264.
[7] Ouyang, H., Mottershead, J. E., Brookfield, D. J., James, S., and Cartmell, M. P., 2000, ''A Methodology for the Determination of Dynamic Instabilities in a Car Disc Brake,'' Int. J. Veh. Des., **23**, pp. 241–262.
[8] Ibrahim, R. A., 1994, ''Friction-Induced Vibration, Chatter, Squeal, and Chaos: Part I- Mechanics of Friction; Part II - Dynamics and Modeling,'' Appl. Mech. Rev., **47**, pp. 209–253.
[9] Popp, K., and Stelter, P., 1990, ''Stick-Slip Vibrations and Chaos,'' Proc. R. Soc. London, Ser. A, **332**, pp. 89–105.


# Fast Transient Analysis for Locally Nonlinear Structures by Recursive Block Convolution


**Joshua H. Gordis**
Mem. ASME, Department of Mechanical Engineering
Naval Postgraduate School,
Monterey, CA 93943-5146
e-mail: jgordis@nps.navy.mil

**Beny Neta**
Department of Mathematics, Naval Postgraduate School,
Monterey, CA 93943-5146



*The transient analysis of large structural systems with localized nonlinearities is a computationally demanding process, inhibiting dynamic redesign and optimization. A previously developed integral equation formulation for transient structural synthesis has demonstrated the ability to solve large locally nonlinear transient problems in a fraction of the time required by traditional direct integration methods, with equivalent or better accuracy. A recursive block-by-block convolution algorithm is developed for the solution of the governing integral equation that further reduces the solution time required. A computing time comparison of single-block versus multiple-block solutions is provided.*


[DOI: 10.1115/1.1389083]

Contributed by the Technical Committee on Vibration and Sound for publication in the JOURNAL OF VIBRATION AND ACOUSTICS. Manuscript received May 2000; revised March 2001. Associate Editor: A. F. Vakakis.


## 1 Introduction

A general formulation for the transient analysis of locally nonlinear structures was presented in Gordis and Radwick [1], referred to as transient structural synthesis. The synthesis provides for the direct calculation of transient response resulting from structural modifications (linear or nonlinear), substructure coupling, and the application of base excitation through linear or nonlinear elements. The formulation is independent of model size, in that only those physical coordinates directly subjected to forces of synthesis (e.g., reactions due to modification) need be retained. This physical coordinate formulation is governed by the following nonlinear Volterra integral equation,

$$\mathbf{x}^*(t)=\mathbf{x}(t)-\int_0^t \mathbf{H}(t-\tau)\mathbf{f}^*(t,\tau,\mathbf{x}^*(\tau),\dot{\mathbf{x}}^*(\tau))d\tau, \tag{1}$$

where $\mathbf{x}^*(t)$ is the vector of synthesized transient responses, the $\mathbf{x}(t)$ vector contains both the initial displacement and response due to externally applied excitations, $\mathbf{H}(t)$ is an impulse response function (IRF) matrix assembled from individual impulse response functions $h(t)$, and $\mathbf{f}^*(t)$ is a vector of synthesized reactions acting on all retained physical DOF.

## 2 Solution of Governing Equation (1)

As was shown in Gordis and Radwick [1], the solution of Eq. (1) is found by replacing the integral with a suitable quadrature,

$$\mathbf{x}^*(i\Delta t)=\mathbf{x}(i\Delta t)-(\Delta t)^{\alpha}\sum_{j=0}^{i-\beta} w_j\mathbf{H}((i-j)\Delta t)\mathbf{f}^*(j\Delta t), \tag{2}$$

where we have abbreviated the general nonlinear force as $\mathbf{f}^*$, $\alpha$ and $\beta$ are real scalar constants depending on the quadrature rule chosen, and the $w_j$ are the quadrature weights. A simple iteration was shown in Gordis and Neta [2] to converge rapidly to the solution, due to the contractive nature of the integral operator.

For the purpose at hand, consider the simplest of quadrature rules, the rectangular rule, $\alpha=1$, $\beta=1$, and $w_j=1$. For $i=0,1,2$, Eq. (2) becomes

$$\mathbf{x}^*(0\Delta t)=\mathbf{x}(0\Delta t) \tag{3}$$

$$\mathbf{x}^*(1\Delta t)=\mathbf{x}(1\Delta t)-\Delta t[\mathbf{H}(1\Delta t)\mathbf{f}^*(0\Delta t)+\mathbf{H}(0\Delta t)\mathbf{f}^*(1\Delta t)] \tag{4}$$

$$\mathbf{x}^*(2\Delta t)=\mathbf{x}(2\Delta t)-\ \ldots\ \ \ldots\Delta t[\mathbf{H}(2\Delta t)\mathbf{f}^*(0\Delta t)+\mathbf{H}(1\Delta t)\mathbf{f}^*(1\Delta t)+\mathbf{H}(0\Delta t)\mathbf{f}^*(2\Delta t)] \tag{5}$$

and we note that $\mathbf{H}(t=0)=0$, yielding the correct series for the rectangular rule. It is important to recognize that the bracketed terms in Eqs. (3), (4), and (5) are equivalent to those produced by the discrete convolution. To state this generally, a discrete convolution produces a result identical to that produced by the rectangular rule applied to a convolution-type integral.

## 3 Recursive Block-by-Block Convolution Algorithm

While the iterative solution defined by Eq. (2) was shown to be very fast in comparison with a standard direct transient analysis [2], further significant reduction in solution time can be obtained by use of the following recursive, block-by-block convolution algorithm. This algorithm reduces compute times by dividing the total time interval into sub-intervals (''blocks''). To facilitate this, a block-by-block convolution algorithm is developed, which does not require data overlap or zero-padding. The $j$th iteration for the response of the $k$th block is

$$\mathbf{x}_k^j=\mathbf{x}_k-\sum_{m=1}^{k-1}(\mathbf{H}_{km}\cdot\mathbf{f}_m)-\mathbf{H}_{kk}\cdot\mathbf{f}_k^{j-1}, \tag{6}$$

where $\mathbf{H}_{km}$ is the $k$th-block IRF filter matrix (to be defined below), and the converged forces of prior blocks $1,2,3\ldots,k-1$ are used.

In order to make use of Eq. (6), the block IRF filter matrix $\mathbf{H}_{km}$ needs to be developed, which is central to the algorithm. We first recognize that the convolution of two vectors $\mathbf{h}$ and $\mathbf{f}$ can be written as a matrix-vector product,

$$\mathbf{x}=\mathbf{h}^*\mathbf{f}=\Phi(\mathbf{h})\cdot\mathbf{f}=\begin{bmatrix} h_1 & 0 & \cdots & & \cdots & 0 \\ h_2 & h_1 & 0 & \cdots & \cdots & \vdots \\ \vdots & h_2 & h_1 & 0 & \cdots & \\ \vdots & \vdots & & \ddots & \ddots & \vdots \\ h_{n-1} & h_{n-2} & \cdots & \ddots & h_1 & 0 \\ h_n & h_{n-1} & h_{n-2} & \cdots & h_2 & h_1 \end{bmatrix} \times \begin{Bmatrix} f_1 \\ f_2 \\ \vdots \\ \\ \vdots \\ f_{n-1} \\ f_n \end{Bmatrix}, \tag{7}$$

where $\Phi(\mathbf{h})$ is a filter matrix [3] constructed from the elements of the vector, $\mathbf{h}$. We define a delay matrix $\mathbf{D}$ where the dimension of $\mathbf{D}$ is consistent with the length of the vector on which it operates. The matrix $\mathbf{D}$ produces a delay in time by one sample. For example, consider the 3 by 1 vector $\mathbf{h}$,

$$\mathbf{D}\cdot\mathbf{h}=\begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}\begin{Bmatrix} h_1 \\ h_2 \\ h_3 \end{Bmatrix}=\begin{Bmatrix} 0 \\ h_1 \\ h_2 \end{Bmatrix}, \tag{8}$$

where the product $\mathbf{Dh}$ is equivalent to the vector $\mathbf{h}$ shifted forward in time (delayed) by one sample. We can introduce delays of arbitrary samples as $\mathbf{D}^j$. The product $\mathbf{D}^j\mathbf{h}$ produces a vector equivalent to the vector $\mathbf{h}$ but delayed by $j$ samples.

The filter matrix $\Phi$ is equal to the summation of powers of the delay matrix multiplied by the filter weights, $h_i$. Alternatively, the columns of the filter matrix $\Phi$ are each products of powers of the delay matrix $\mathbf{D}$ and the vector $\mathbf{h}$, i.e., the $j$th column of $\mathbf{h}$ is given by $\mathbf{D}^j\mathbf{h}$. The filter matrix of a vector $\mathbf{h}$ of length $n$ is therefore,

$$\Phi(\mathbf{h})=\sum_{j=0}^{n-1} h_j\times\mathbf{D}^k=[\mathbf{D}^0\mathbf{h}\ \ \mathbf{D}^1\mathbf{h}\ \cdots\ \mathbf{D}^{n-2}\mathbf{h}\ \ \mathbf{D}^{n-1}\mathbf{h}] \tag{9}$$

We now construct the block-by-block (BBB) convolution of two vectors, $\mathbf{h}$ and $\mathbf{f}$, i.e., $\mathbf{h}^*\mathbf{f}$. We subdivide the entire time record of duration $T$ seconds, consisting of $N$ sample points ($\Delta t=T/N$) into a number of equally sized blocks, or subintervals, i.e., each subinterval contains the same number of sample points. We will subdivide the entire record into "$K$" blocks, where each block consists of $J=N/K$ samples, and the duration of each block is $J\Delta t$ seconds. It is important to emphasize that there is a delay of $J$ samples between blocks. For the purpose of developing the BBB algorithm, we will need to extract those rows of a vector corresponding to a particular block. To this end, we define the row extraction matrix $\mathbf{r}$:

$$\mathbf{r}=\begin{bmatrix} 0 & \cdots & 0 & 1 & 0 & \cdots & & \\ \vdots & & & 0 & 1 & 0 & & \\ & & & & 0 & 1 & 0 & \\ \vdots & & & & & 0 & 1 & 0 \\ 0 & \cdots & & & & \cdots & 0 & 1 \end{bmatrix} \tag{10}$$

The product of the matrix $\mathbf{r}$ with a vector $\mathbf{x}$ is the subvector of $\mathbf{x}$ consisting of the rows (samples) of the $K$th block, i.e., $\mathbf{r}\cdot\mathbf{x}=\mathbf{x}_k$ where $\mathbf{x}_K=[\mathbf{x}_{J(K-1)+1}\cdots x_{n-1}x_n]^T$. Using the delay matrix $\mathbf{D}$, we can define a matrix which extracts the rows of the $k$th block, where $k=1,2,\ldots,K$.

$$\mathbf{r}_k=\mathbf{r}\cdot\mathbf{D}^k=\begin{bmatrix} 0 & \cdots & 0 & 1 & 0 & \cdots & & & \cdots & 0 \\ \vdots & & & 0 & 1 & 0 & & & & \vdots \\ & & & & 0 & 1 & 0 & & & \\ \vdots & & & & & 0 & 1 & 0 & & \vdots \\ 0 & \cdots & & & & \cdots & 0 & 1 & 0 \cdots & 0 \end{bmatrix} \tag{11}$$

The matrix equation, Eq. (7), can be written in a block-partitioned form as follows. We can write the $k$th subvector of $\mathbf{x}$, i.e., $\mathbf{x}_k$, as

$$\mathbf{x}_k=\sum_{m=1}^{k}\mathbf{r}_{N-kJ}\Phi(\mathbf{h})\mathbf{r}_{N-mJ}^T\mathbf{r}_{N-mJ}\mathbf{f} \tag{12}$$

Therefore, the IRF block filter matrix of Eq. (6) is

$$\mathbf{H}_{km}=\mathbf{r}_{N-kJ}\Phi(\mathbf{h})\mathbf{r}_{N-mJ}^T \tag{13}$$

It is important to note that the block filter matrices $\mathbf{H}_{km}$ need never be formed, as the appropriate subvectors of $\mathbf{h}$ and $\mathbf{f}$ (as defined by Eq. (13)) can be convolved directly.

## 4 Performance Comparison—Standard and Block-by-Block Convolution

A traditional (single-block) convolution, for sufficiently long records of length $n$, is most efficiently computed using the FFT, yielding a total number of floating point operations (FLOPS) proportional to $n^*\log_2(n)$. The computing language MATLAB provides a built-in function for convolution that uses FIR filters for the calculation, and yields total FLOPS proportional to $n^2$. As we are here interested in comparing the performance of the BBB algorithm with the traditional single-block convolution, the use of the MATLAB function will provide much convenience with no loss in the ability to compare algorithms. The number of FLOPS for the BBB algorithm is given by:

$$\text{FLOPS}\propto K(2J^2-J)+\frac{1}{2}(K^2-K)(4J^2-4J+1)$$

which yields an optimum number of blocks greater than the total number of samples $N$, and is a noninteger number of blocks. What is useful about this solution is that is indicates that the FLOPS

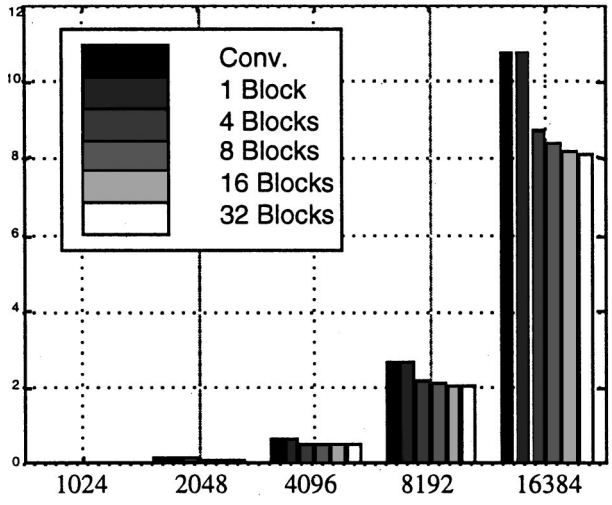


**Fig. 1 FLOPS($\times 10^{18}$) vs record length**

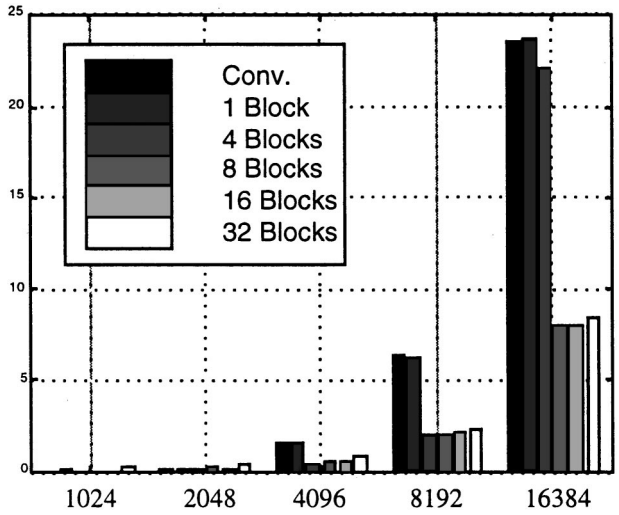


**Fig. 2 Compute time (sec) vs record length**

required by a BBB convolution decreases monotonically with increasing block number. This is shown in Fig. 1, which compares the FLOPS required by a standard convolution to the BBB convolution for varying total number of samples, $N$, and for different numbers of blocks. However, if we compare actual compute time (using MATLAB), we see that there is a point at which increasing the number of blocks results in increased compute times, as the computing ''overhead'' associated with increased block number outweighs the decrease in computing time due to the reduction in FLOPS required. This is shown in Fig. 2, and eight blocks yields the minimum time for this particular calculation.

To evaluate the BBB algorithm in the context of a transient analysis, the structural system of Gordis and Radwick [1], will be used. This is a simplified isolated square deck structure of approximately 51,500 DOF, with four nonlinear isolators at each corner, subjected to an impulsive base motion excitation. As reported in Gordis and Radwick [1], the total time for the direct transient analysis was 30 minutes 15 seconds, and the synthesis took 7 seconds. Here we compare multiple-block solution times with that for a single block. This is summarized in Table 1. The single-block solution is equivalent to a standard convolution. It is clear that a multiple-block solution provides a significant reduction in compute time.

**Table 1 Solution times vs. number of blocks**

| No. of Blocks | Time for Synthesis (sec) |
|---|---|
| 1 | 145.56 |
| 4 | 24.83 |
| 8 | 15.56 |

## 5 Conclusions

A new recursive block-by-block convolution algorithm has been developed for the solution of the governing nonlinear Volterra integral equation for locally nonlinear structural synthesis. The new algorithm is extremely fast, as compared with direct integration, and is also much faster than the previously reported algorithm [1]. The algorithm lends itself for use in nonlinear structural dynamic optimization. The algorithm can be used whenever the convolution of long time records is required.

## Acknowledgments

This work is dedicated to Joseph Robert Gordis, in honor of his second year. The authors gratefully acknowledge the support of the National Science Foundation, #9713481.

## Nomenclature

$\mathbf{D}$ = delay matrix
$\mathbf{f}$ = excitations
$\mathbf{h}$ = impulse response function
$\mathbf{H}$ = impulse response function matrix
$\Phi$ = filter matrix
$\mathbf{r}$ = row extraction matrix
$t, \tau$ = time
$w$ = quadrature weights
$\mathbf{x}$ = displacement response

## References

[1] Gordis, J. H., and Radwick, J. L., 1999, ''Efficient Transient Analysis for Large Locally Nonlinear Structures,'' Shock Vib. Dig., **6**, No. 1, pp. 1–9.
[2] Gordis, J. H., and Neta, B., 1999, ''Efficient Nonlinear Transient Dynamic Analysis for Structural Optimization Using an Exact Integral Equation Formulation,'' Report, Project #97-13481, National Science Foundation, Earthquake Hazard Mitigation Program.
[3] Strang, G., and Nguyen, T., 1996, *Wavelets and Filter Banks*, Wellesley-Cambridge Press, Wellesley, MA.