# Structured matrix

*Wenlei Gao*[*]


## ABSTRACT

The documentation for fast MSSA and relationship between multi-dimensional convolution and structured matrix, such as Circulant, Toeplitz and Hankel matrix.


## INTRODUCTION

(Korobeynikov, 2009) proposed the computation and memory efficient method for MSSA.

## METHOD

We start from the definition of discrete fourier transform.

### Discrete Fourier transform (DFT)

For a vector $\mathbf{s}$ length of $N$, the discrete Fourier transform is defined as

$$\tilde{s}_k = \sum_{l=0}^{N-1} e^{-i2\pi kl/N} s_l \, , \tag{1}$$

where the index $k = 0, 1, 2, \cdots, N-1$. Suppose the vector $\mathbf{s}$ represent a time series length of $N$ with time sampling interval is $\Delta t$. So the frequency sampling interval $\Delta f$ can be computed as

$$\Delta f = 1/\Delta t/N = \frac{1}{N\Delta t} \tag{2}$$

The Fourier transform for this time series in continueous case can be expressed as

$$\tilde{s}(f) = \int e^{-i2\pi ft} s(t) dt \tag{3}$$

In discrete case and assume the frequency $f$ can be expressed as $f = k\Delta f$ and using the expression in equation 2

$$\begin{aligned}
\tilde{s}_{k\Delta f} &= \sum_{l=0}^{N-1} e^{-i2\pi\, k\Delta f\, l\Delta t} s_{l\Delta t} \\
\tilde{s}_{k\Delta f} &= \sum_{l=0}^{N-1} e^{-i2\pi\, k\frac{1}{N\Delta t}\, l\Delta t} s_{l\Delta t} \\
\tilde{s}_k &= \sum_{l=0}^{N-1} e^{-i2\pi\, kl/N} s_l \, .
\end{aligned} \tag{4}$$

The the last equation is exactly same as the equation 1. Through this derivation, we can understand the physical meaning of the Fourier coefficients obtained via discrete Fourier transform. Note that the first coefficient corresponding to 0 frequency and so as to the time series. The first element of $\mathbf{s}$ is the sample at 0 time.

The above operation can be expressed into the matrix-vector form. To simplify the definition of Fourier transform matrix, we define a variable $z = e^{-i\pi/N}$, so the Fourier transform matrix, which is a $N$-by-$N$ square matrix and the element $F_{kl} = z^{kl}$. It can be expanded as

$$\mathbf{F}_N = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & z & \cdots & z^{N-1} \\ 1 & z^2 & \cdots & z^{2(N-1)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & z^{N-1} & \cdots & z^{(N-1)(N-1)} \end{bmatrix} \tag{5}$$

As $\mathbf{F}_N$ is a orthogonal (but not orthonormal) matrix, we can be easily get the its inverse matrix

$$\mathbf{F}_N^{-1} = \frac{1}{N}\mathbf{F}_N^H = \frac{1}{N}\mathbf{F}_N^* \tag{6}$$

where the super-script $^H$ represent complex conjugate transpose and $*$ indicate conjugate transpose. As the matrix $\mathbf{F}_N$ is a symmetrical matrix, that's why we can get equation 6.

## Properties of Fourier transform

In this section, we review the properties of discrete Fourier transform (**DFT**), most of the properties are derived in continuous domain as it is much easier for the derivation, but the discrete version are verified by *Julia* code.

We define the Fourier transform as

$$X(\omega) = \int_{-\infty}^{\infty} x(t)e^{-i\omega t}dt \tag{7}$$

where $\omega$ is radian frequency and $t$ is time. The above operation can be simplified as

$$\boldsymbol{\mathcal{F}}[x(t)] \rightarrow X(\omega) \tag{8}$$

*1. time shift*

$$\boldsymbol{\mathcal{F}}[x(t-t_0)] = X(\omega)e^{i\omega t_0} \tag{9}$$

**Proof**:

$$\boldsymbol{\mathcal{F}}[x(t-t_0)] = \int_{-\infty}^{\infty} x(t-t_0)e^{-i\omega t}dt$$

we set $t' = t - t_0$ and exchange the variables, we can get

$$\boldsymbol{\mathcal{F}}[x(t-t_0)] = \int_{-\infty}^{\infty} x(t')e^{-i\omega(t'+t_0)}dt'$$

$$\boldsymbol{\mathcal{F}}[x(t-t_0)] = e^{i\omega t_0}\int_{-\infty}^{\infty} x(t')e^{-i\omega t'}dt' = X(\omega)e^{i\omega t_0}$$

*2. frequency modulation*

$$\mathcal{F}[x(t)e^{i\omega_0 t}] = X(\omega - \omega_0) \tag{10}$$

**Proof**:

$$\mathcal{F}[x(t)e^{i\omega_0 t}] = \int_{-\infty}^{\infty} x(t)e^{i\omega_0 t}e^{-i\omega t}dt = \int_{-\infty}^{\infty} x(t)e^{-i(\omega-\omega_0)t}dt = X(\omega - \omega_0)$$

Note: this is the property we forget very often.

*3. time reversal*

$$\mathcal{F}[x(-t)] = X(-\omega) \tag{11}$$

**Proof**:

$$\mathcal{F}[x(-t)] = \int_{-\infty}^{\infty} x(-t)e^{-i\omega t}dt$$

Set $t' = -t$ and change the variables we can get

$$\mathcal{F}[x(-t)] = \int_{\infty}^{-\infty} x(t')e^{-i\omega(-t')}d(-t') = \int_{-\infty}^{\infty} x(t')e^{-i(-\omega)t'}dt' = X(-\omega)$$

Based on this property, we can get two extensions, if the time function is and even function, that is

$$x(t) = x(-t) \to X(\omega) = X(-\omega)$$

if the time function is odd function, the Fourier transform of this function is also an odd function

$$x(t) = -x(-t) \to X(\omega) = -X(-\omega)$$

*4. time and frequency stretching*

$$\mathcal{F}[x(at)] = \frac{1}{a}X(\frac{\omega}{a}) \tag{12}$$

where $a$ is positive number

**Proof**:

$$\mathcal{F}[x(at)] = \int_{-\infty}^{\infty} x(at)e^{-i\omega t}dt$$

Set $t' = at$, By change the variable, we can get

$$\mathcal{F}[x(at)] = \int_{-\infty}^{\infty} x(t')e^{-i\omega\frac{t'}{a}}d\frac{t'}{a} = \frac{1}{a}\int_{-\infty}^{\infty} x(t')e^{-i\frac{\omega}{a}t'}dt' = \frac{1}{a}X(\frac{w}{a})$$

Stretching in one domain means squeezing in another domain and vis versa.

*5. Complex conjugate*

$$\boldsymbol{\mathcal{F}}[x^*(t)] = X^*(-\omega) \tag{13}$$

where the super-script $^*$ represent complex conjugate.

**Proof**:

$$\int_{-\infty}^{\infty} x^*(t)e^{-i\omega t}dt = \int_{-\infty}^{\infty} \left(x(t)e^{i\omega t}\right)^* dt = \int_{-\infty}^{\infty} \left(x(t)e^{-i(-\omega)t}\right)^* dt = \left(\int_{-\infty}^{\infty} x(t)e^{-i(-\omega)t}dt\right)^* = X^*(-\omega)$$

## Circulant matrix

An $9 \times 9$ circulant matrix is defined as

$$\mathbf{C} = \begin{bmatrix} c_1 & c_9 & c_8 & c_7 & c_6 & c_5 & c_4 & c_3 & c_2 \\ c_2 & c_1 & c_9 & c_8 & c_7 & c_6 & c_5 & c_4 & c_3 \\ c_3 & c_2 & c_1 & c_9 & c_8 & c_7 & c_6 & c_5 & c_4 \\ c_4 & c_3 & c_2 & c_1 & c_9 & c_8 & c_7 & c_6 & c_5 \\ c_5 & c_4 & c_3 & c_2 & c_1 & c_9 & c_8 & c_7 & c_6 \\ c_6 & c_5 & c_4 & c_3 & c_2 & c_1 & c_9 & c_8 & c_7 \\ c_7 & c_6 & c_5 & c_4 & c_3 & c_2 & c_1 & c_9 & c_8 \\ c_8 & c_7 & c_6 & c_5 & c_4 & c_3 & c_2 & c_1 & c_9 \\ c_9 & c_8 & c_7 & c_6 & c_5 & c_4 & c_3 & c_2 & c_1 \end{bmatrix} \tag{14}$$

A circulant matrix is fully specified by its first column $\mathbf{c} = \begin{bmatrix} c_1 & c_2 & c_3 & c_4 & c_5 & c_6 & c_7 & c_8 & c_9 \end{bmatrix}^T$, we know that the product between circulant matrix and a vector $\mathbf{v}$ length of $N$ is equivalent to the circular convolution between vector $\mathbf{c}$ and $\mathbf{v}$, and the circular convolution can be computed efficiently via Fast Fourier transform (FFT). So we have

$$\mathbf{Cv} = \mathbf{F}_N^{-1}\left(\mathbf{F}_N\mathbf{c} \circ \mathbf{F}_N\mathbf{v}\right) = \mathbf{F}_N^{-1}\left(diag(\mathbf{F}_N\mathbf{c}) \cdot \mathbf{F}_N\mathbf{v}\right) \ , \tag{15}$$

where the symbol $\circ$ and *diag* represents Hardmard (element-wise) multiplication and building diagonal matrix from a vector.

## Toeplitz matrix

Toeplitz matrix is associated with linear convolution, suppose we have two vectors $\mathbf{s} = \begin{bmatrix} s_1 & s_2 & s_3 & s_4 & s_5 \end{bmatrix}^T$ and $\mathbf{f} = \begin{bmatrix} f_1 & f_2 & f_3 & f_4 & f_5 \end{bmatrix}^T$, the linear convolution of these two vectors is vector $\mathbf{r}$ length of 9. We can represent the convolution into matrix-vector form as

$$\mathbf{r} = \mathbf{h} * \mathbf{f} = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \\ r_7 \\ r_8 \\ r_9 \end{bmatrix} = \begin{bmatrix} s_1 & & & & \\ s_2 & s_1 & & & \\ s_3 & s_2 & s_1 & & \\ s_4 & s_3 & s_2 & s_1 & \\ s_5 & s_4 & s_3 & s_2 & s_1 \\ & s_5 & s_4 & s_3 & s_2 \\ & & s_5 & s_4 & s_3 \\ & & & s_5 & s_4 \\ & & & & s_5 \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \end{bmatrix} \tag{16}$$

The efficient linear convolution usually padding 0 to vectors make their length equal to 9, after padding zeros, we get new vector $\hat{\mathbf{s}} = \begin{bmatrix} s_1 & s_2 & s_3 & s_4 & s_5 & 0 & 0 & 0 & 0 \end{bmatrix}$ and $\hat{\mathbf{f}} = \begin{bmatrix} f_1 & f_2 & f_3 & f_4 & f_5 & 0 & 0 & 0 & 0 \end{bmatrix}$, the circular convolution between $\hat{\mathbf{s}}$ and $\hat{\mathbf{f}}$ can be expressed as

$$\hat{\mathbf{r}} = \hat{\mathbf{s}} \circledast \hat{\mathbf{f}} = \begin{bmatrix} s_1 & 0 & 0 & 0 & 0 & s_5 & s_4 & s_3 & s_2 \\ s_2 & s_1 & 0 & 0 & 0 & 0 & s_5 & s_4 & s_3 \\ s_3 & s_2 & s_1 & 0 & 0 & 0 & 0 & s_5 & s_4 \\ s_4 & s_3 & s_2 & s_1 & 0 & 0 & 0 & 0 & s_5 \\ s_5 & s_4 & s_3 & s_2 & s_1 & 0 & 0 & 0 & 0 \\ 0 & s_5 & s_4 & s_3 & s_2 & s_1 & 0 & 0 & 0 \\ 0 & 0 & s_5 & s_4 & s_3 & s_2 & s_1 & 0 & 0 \\ 0 & 0 & 0 & s_5 & s_4 & s_3 & s_2 & s_1 & 0 \\ 0 & 0 & 0 & 0 & s_5 & s_4 & s_3 & s_2 & s_1 \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{17}$$

When can see that $\hat{\mathbf{r}} = \mathbf{r}$, so linear convolution can be efficiently computed by padding zeros to vectors then compute circular convolution in frequency domain.

Let's consider another Toeplitz matrix built from a vector $\mathbf{c} = \begin{bmatrix} c_1 & c_2 & c_3 & c_4 & c_5 & c_6 & c_7 & c_8 & c_9 \end{bmatrix}$ with length $N = 9$, we first determine two integer: $L$ and $K$ make $L + K - 1 = N$. To make the matrix close to square form, we determine them by

$$L = \left\lfloor \frac{N}{2} \right\rfloor + 1, \; K = N + 1 - L \tag{18}$$

in this special case, $N = 9$, so $L = K = 5$, the Toeplitz matrix build from this vector can be expressed as

$$\mathbf{T} = \begin{bmatrix} c_5 & c_4 & c_3 & c_2 & c_1 \\ c_6 & c_5 & c_4 & c_3 & c_2 \\ c_7 & c_6 & c_5 & c_4 & c_3 \\ c_8 & c_7 & c_6 & c_5 & c_4 \\ c_9 & c_8 & c_7 & c_6 & c_5 \end{bmatrix} \tag{19}$$

The vector $\mathbf{c}$ can be uniquely determined from the last column and last row. There are two way to embedding this Teoplitz matrix into a circular matrix, the first to embedding this matrix at the lower-left part of the circular matrix created by vector $\mathbf{c}$, which shows as below as

$$\begin{array}{ccccc|cccc} c_1 & c_9 & c_8 & c_7 & c_6 & c_5 & c_4 & c_3 & c_2 \\ c_2 & c_1 & c_9 & c_8 & c_7 & c_6 & c_5 & c_4 & c_3 \\ c_3 & c_2 & c_1 & c_9 & c_8 & c_7 & c_6 & c_5 & c_4 \\ c_4 & c_3 & c_2 & c_1 & c_9 & c_8 & c_7 & c_6 & c_5 \\ \hline c_5 & c_4 & c_3 & c_2 & c_1 & c_9 & c_8 & c_7 & c_6 \\ c_6 & c_5 & c_4 & c_3 & c_2 & c_1 & c_9 & c_8 & c_7 \\ c_7 & c_6 & c_5 & c_4 & c_3 & c_2 & c_1 & c_9 & c_8 \\ c_8 & c_7 & c_6 & c_5 & c_4 & c_3 & c_2 & c_1 & c_9 \\ c_9 & c_8 & c_7 & c_6 & c_5 & c_4 & c_3 & c_2 & c_1 \end{array}$$

The above circulant matrix is called $\mathbf{C}_1$. We can create a new vector $\hat{\mathbf{c}} = \begin{bmatrix} c_5 & c_6 & c_7 & c_8 & c_9 & c_1 & c_2 & c_3 & c_4 \end{bmatrix}^T$, which is obtained by permutating the elements of the original vector, the Toeplitz matrix can be embedded at the upper-left part of the circulant matrix created from $\hat{\mathbf{c}}$ shows as bellow

$$\begin{array}{ccccc|cccc}
\cline{1-5}
\multicolumn{1}{|c}{c_5} & c_4 & c_3 & c_2 & c_1 & c_9 & c_8 & c_7 & c_6 \\
\multicolumn{1}{|c}{c_6} & c_5 & c_4 & c_3 & c_2 & c_1 & c_9 & c_8 & c_7 \\
\multicolumn{1}{|c}{c_7} & c_6 & c_5 & c_4 & c_3 & c_2 & c_1 & c_9 & c_8 \\
\multicolumn{1}{|c}{c_8} & c_7 & c_6 & c_5 & c_4 & c_3 & c_2 & c_1 & c_9 \\
\multicolumn{1}{|c}{c_9} & c_8 & c_7 & c_6 & c_5 & c_4 & c_3 & c_2 & c_1 \\
\cline{1-5}
c_1 & c_9 & c_8 & c_7 & \multicolumn{1}{c}{c_6} & c_5 & c_4 & c_3 & c_2 \\
c_2 & c_1 & c_9 & c_8 & \multicolumn{1}{c}{c_7} & c_6 & c_5 & c_4 & c_3 \\
c_3 & c_2 & c_1 & c_9 & \multicolumn{1}{c}{c_8} & c_7 & c_6 & c_5 & c_4 \\
c_4 & c_3 & c_2 & c_1 & \multicolumn{1}{c}{c_9} & c_8 & c_7 & c_6 & c_5
\end{array}$$

and call this new matrix as $\mathbf{C}_2$.

Suppose we need to compute the product between Toeplitz matrix $\mathbf{T}$ and a vector $\mathbf{v}$

$$\mathbf{r} = \mathbf{T}\mathbf{v} \tag{20}$$

where the length of $\mathbf{v}$ is $K$. There are two way to do it efficiently via FFT, and these two ways corresponding to the two embedding methods. The first step is to padding vector $\mathbf{v}$ with $L-1$ 0s get a new vector $\hat{\mathbf{v}}$. Following above example, $K = L = 5$ and $\mathbf{v} = \begin{bmatrix} v_1 & v_2 & v_3 & v_4 & v_5 \end{bmatrix}^T$, and the padded vector $\hat{\mathbf{v}} = \begin{bmatrix} v_1 & v_2 & v_3 & v_4 & v_5 & 0 & 0 & 0 & 0 \end{bmatrix}^T$, The product between $\mathbf{C}_1$ and $\hat{\mathbf{v}}$ are

$$\hat{\mathbf{r}}_1 = \mathbf{C}_1\hat{\mathbf{v}} = \begin{bmatrix}
c_1 & c_9 & c_8 & c_7 & c_6 & c_5 & c_4 & c_3 & c_2 \\
c_2 & c_1 & c_9 & c_8 & c_7 & c_6 & c_5 & c_4 & c_3 \\
c_3 & c_2 & c_1 & c_9 & c_8 & c_7 & c_6 & c_5 & c_4 \\
c_4 & c_3 & c_2 & c_1 & c_9 & c_8 & c_7 & c_6 & c_5 \\
c_5 & c_4 & c_3 & c_2 & c_1 & c_9 & c_8 & c_7 & c_6 \\
c_6 & c_5 & c_4 & c_3 & c_2 & c_1 & c_9 & c_8 & c_7 \\
c_7 & c_6 & c_5 & c_4 & c_3 & c_2 & c_1 & c_9 & c_8 \\
c_8 & c_7 & c_6 & c_5 & c_4 & c_3 & c_2 & c_1 & c_9 \\
c_9 & c_8 & c_7 & c_6 & c_5 & c_4 & c_3 & c_2 & c_1
\end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{21}$$

where the last $L$ elements of $\hat{\mathbf{r}}_1$ equal to $\mathbf{r}$. On the other hand, we can take advantage of the circulate matrix $\mathbf{C}_2$, we have

$$\hat{\mathbf{r}}_2 = \mathbf{C}_2\hat{\mathbf{v}} = \begin{bmatrix}
c_5 & c_4 & c_3 & c_2 & c_1 & c_9 & c_8 & c_7 & c_6 \\
c_6 & c_5 & c_4 & c_3 & c_2 & c_1 & c_9 & c_8 & c_7 \\
c_7 & c_6 & c_5 & c_4 & c_3 & c_2 & c_1 & c_9 & c_8 \\
c_8 & c_7 & c_6 & c_5 & c_4 & c_3 & c_2 & c_1 & c_9 \\
c_9 & c_8 & c_7 & c_6 & c_5 & c_4 & c_3 & c_2 & c_1 \\
c_1 & c_9 & c_8 & c_7 & c_6 & c_5 & c_4 & c_3 & c_2 \\
c_2 & c_1 & c_9 & c_8 & c_7 & c_6 & c_5 & c_4 & c_3 \\
c_3 & c_2 & c_1 & c_9 & c_8 & c_7 & c_6 & c_5 & c_4 \\
c_4 & c_3 & c_2 & c_1 & c_9 & c_8 & c_7 & c_6 & c_5
\end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{22}$$

So the first $L$ elements of $\hat{\mathbf{r}}_2$ equal to $\mathbf{r}$. We summarize the efficient computation of the product between Toeplitz matrix and a vector in Algorithm 1 and Algorithm 2

Another method is summarized in Algorithm 2 as

**Algorithm 1:** Teoplitz-Times-Vector

**Function** $\mathbf{r} = ttv(\mathbf{c}, \mathbf{v})$
- length of $\mathbf{c}$: $N = length(\mathbf{c})$
- compute $L = \lfloor \frac{N}{2} \rfloor + 1$, $K = N + 1 - L$
- padding $L - 1$ 0s to the end of $\mathbf{v}$ to get a new vector $\hat{\mathbf{v}}$ with length of $N$
- $\mathbf{r}_1 = \mathbf{F}_N^{-1}(\mathbf{F}_N \mathbf{c} \circ \mathbf{F}_N \hat{\mathbf{v}})$
- take the last $L$ elements from $\mathbf{r}_1$

**Algorithm 2:** Teoplitz-Times-Vector

**Function** $\mathbf{r} = ttv(\mathbf{c}, \mathbf{v})$
- length of $\mathbf{c}$: $N = length(\mathbf{c})$
- compute $L = \lfloor \frac{N}{2} \rfloor + 1$, $K = N + 1 - L$
- padding $L - 1$ 0s to the end of $\mathbf{v}$ to get a new vector $\hat{\mathbf{v}}$ with length of $N$
- move the first $K - 1$ element of $\mathbf{c}$ to the end, get a new vector $\hat{\mathbf{c}}$
  $\mathbf{r}_2 = \mathbf{F}_N^{-1}(\mathbf{F}_N \hat{\mathbf{c}} \circ \mathbf{F}_N \hat{\mathbf{v}})$
- take the first $L$ elements from $\mathbf{r}_2$

## Hankel matrix

Hankel matrix of size $L \times K$ created from the same vector $\mathbf{c}$ is given as

$$\mathbf{H} = \begin{bmatrix} c_1 & c_2 & c_3 & c_4 & c_5 \\ c_2 & c_3 & c_4 & c_5 & c_6 \\ c_3 & c_4 & c_5 & c_6 & c_7 \\ c_4 & c_5 & c_6 & c_7 & c_8 \\ c_5 & c_6 & c_7 & c_8 & c_9 \end{bmatrix} \tag{23}$$

which can be determined from the first column and last row.

Note: The circulant matrix and Teoplitz matrix are constant on the diagonals. However, Hankel matrix are constant on the anti-diagonals.

One can convert a Hankel matrix to a Toeplitz matrix by reversing its columns. The columns reversing process can be represented by right multiplying a $K \times K$ matrix $\mathbf{R}$, whose elements are 1 on the main anti-diagonals and 0 elsewhere. Follow the toy example, we can get

$$\mathbf{T} = \begin{bmatrix} c_5 & c_4 & c_3 & c_2 & c_1 \\ c_6 & c_5 & c_4 & c_3 & c_2 \\ c_7 & c_6 & c_5 & c_4 & c_3 \\ c_8 & c_7 & c_6 & c_5 & c_4 \\ c_9 & c_8 & c_7 & c_6 & c_5 \end{bmatrix} = \mathbf{HR} = \begin{bmatrix} c_1 & c_2 & c_3 & c_4 & c_5 \\ c_2 & c_3 & c_4 & c_5 & c_6 \\ c_3 & c_4 & c_5 & c_6 & c_7 \\ c_4 & c_5 & c_6 & c_7 & c_8 \\ c_5 & c_6 & c_7 & c_8 & c_9 \end{bmatrix} \begin{bmatrix} & & & & 1 \\ & & & 1 & \\ & & 1 & & \\ & 1 & & & \\ 1 & & & & \end{bmatrix} \tag{24}$$

Before we discuss about the multiplication between Hankel matrix and a vector, we first look at the property of the column reversing matrix $\mathbf{R}$, we have

$$\mathbf{R} = \mathbf{R}^T = \mathbf{R}^{-1} \tag{25}$$

So the product between Hankel matrix $\mathbf{H}$ and a vector $\mathbf{v}$ is given as

$$\mathbf{r} = \mathbf{H}\mathbf{v} = \mathbf{H}\mathbf{R}\mathbf{R}^{-1}\mathbf{v} = \mathbf{H}\mathbf{R}\mathbf{R}\mathbf{v} = (\mathbf{H}\mathbf{R})(\mathbf{R}\mathbf{v}) = \mathbf{T}\hat{\mathbf{v}} \tag{26}$$

where $\mathbf{v}$ is the reversed version of the vector $\mathbf{v}$. From above derivations, we can see that the multiplication between a Hankel matrix and a vector is converted to the multiplication between Teoplitz matrix and a vector. So the algorithm for the multiplication between Hankel matrix and a vector is summarized in Algorithm 3 as

**Algorithm 3:** Hankel-Times-Vector

**Function** $\mathbf{r} = htv(\mathbf{c}, \mathbf{v})$
  length of $\mathbf{c}$: $N = length(\mathbf{c})$
  compute $L = \left\lfloor \frac{N}{2} \right\rfloor + 1$, $K = N + 1 - L$
  reverse the order of the elements of $\mathbf{v}$: $\hat{\mathbf{v}} = reverse(\mathbf{v})$
  call the function Toeplitz-times-vector: $\mathbf{r} = ttv(\mathbf{c}, \hat{\mathbf{v}})$
  **return** $\mathbf{r}$

## REFERENCES

Korobeynikov, A., 2009, Computation-and space-efficient implementation of ssa: arXiv preprint arXiv:0911.4498.