SOLVING A CLASS OF BOUNDARY VALUE PROBLEMS BY LSQR

by

Yu-Yang QIU*

College of Statistics and Mathematics, Zhejiang Gongshang University, Hangzhou, China

Original scientific paper
https://doi.org/10.2298/TSCI160715059Q

Boundary value problems arising in fluid mechanics and thermal science can be transformed uniformly to a set of linear equations, whose coefficient matrix is circulant. This paper adopts a matrix iteration LSQR to solve the inverse of coefficient matrix. The solution process is elucidated step by step, and the numerical results reveal the effectiveness and feasibility of the presented method.

Key words: *boundary value problems, circulant, iteration LSQR, inverse*

Introduction

Some boundary value problems arising in fluid mechanics and thermal science problems can be written as the following form [1-6]:

$$\begin{cases} Lu(x,y) = \mathrm{f}(x,y), (x,y) \in \Omega, \\ Bu(x,y) = \mathrm{g}(x,y), (x,y) \in \partial\Omega \end{cases} \tag{1}$$

where L is a linear differential operator, B – a boundary operator, f(x, y) and g(x, y) are two known functions, and $\Omega \in R^2$ is an open bounded domain with boundary $\partial\Omega$. By the method of fundamental solutions, problem (1) can be transformed uniformly to a set of linear equations $C_n z = b$ with unknown vector z, the circulant or block-circulant matrix C_n, and the given vector b [1]. So the key to the problem (1) is to solve the inverse of the circulant matrix, C_n. Generally, we should add the additional conditions such as positive definite on C_n to get its inverse, and fast Fourier transformation is necessary [7-11]. However, it may not maintain workable as n gets bigger.

In this paper, we adopt an iteration algorithm LSQR to solve the inverse of coefficient matrix [12]. Our ideas are based on the following observations: 1 – since the inverse of the circulant matrix C_n is circulant [7], it can be mapped to the independent parameter vector with its special structure, 2 – the inverse of the circulant matrix can be transformed to solve a linear equation, and it will be obtained by vector form iteration LSQR involved Kronecker product, and 3 – the LSQR only involves matrix-vector product, however, it has Kronecker product which will increase the computational complexity. So we intend to release Kronecker product and get the corresponding matrix form iteration. In this paper, the solution process is elucidated step by step and numerical results reveal the effectiveness and feasibility of the presented method.

* Corresponding author, e-mail: yuyangqiu77@163.com

Notation. In the rest of paper, $R^{m\times n}$ denotes the space of real $m\times n$ matrix. For any matrix $X = [x_1, x_2,\ldots, x_n] \in R^{m\times n}$, X^T stands for its transpose, and the long vector expanded by columns of X is denoted by $vec(X) = [x_1^T, x_2^T,\ldots,x_n^T]^T$. For any vector $v \in R^n$, $v(i)$ is its i^{th} component. The norm $\|\ \|_F$ is the Frobenius norm of matrix, while $\|\ \|_2$ is 2-norm of vector or matrix. The notation $\otimes$ is Kronecker product.

The inverse of circulant matrix and LSQR involved Kronecker product

The circulant matrix from the problem (1) has the following form:

$$C_n = \begin{pmatrix} c_0 & c_1 & \cdots & c_{n-2} & c_{n-1} \\ c_{n-1} & c_0 & \cdots & c_{n-3} & c_{n-2} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ c_2 & c_3 & \ddots & c_0 & c_1 \\ c_1 & c_2 & \cdots & c_{n-1} & c_0 \end{pmatrix}$$

where c_i, $i = 0, 1,\ldots, n-1$ are parameters. Suppose:

$$G = \begin{pmatrix} 0 & 1 & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \ddots & 0 & 1 \\ 1 & 0 & \cdots & 0 & 0 \end{pmatrix}$$

then

$$C_n = c_0 G^0 + c_1 G + \cdots + c_{n-1} G^{n-1} \tag{2}$$

where

$$G^0 = I_n$$

Note that C_n is a circulant matrix, its inverse matrix C_{inv} is also circulant [7]. So C_{inv} can be rewritten:

$$C_{\text{inv}} = h_0 G^0 + h_1 G + \cdots + h_{n-1} G^{n-1} \tag{3}$$

where $h_i = 0, 1,\ldots, n-1$ are the parameters as in eq. (2). Hence:

$$vec C_{\text{inv}} = C_T h \tag{4}$$

where

$$C_T = [vec(G^0), vec(G^1),\cdots, vec(G^{n-1})]$$

and

$$h = (h_0, h_1,\cdots, h_{n-1})^T$$

Obviously, the inverse circulant matrix C_{inv} can be represented by the unknown vector h with (4). Therefore:

$$C_n C_{\text{inv}} = I_n$$

is equivalent to:

$$Ah = b \tag{5}$$

with

$$A = (I_n \otimes C_n)C_T \quad \text{and} \quad b = vec(I_n)$$

That is, we can solve eq. (5) to get the vector h and reconstruct the inverse matrix C_{inv} by eq. (3). So the key to C_{inv} is the solution of eq. (5). We introduce the iteration LSQR to solve eq. (5), since it has good numerical performance [7].

Iteration LSQR

1 – Initialization.

Set $h = h_{int}$, $r = b - Ah_{int}$.

$$\beta_0 = \| r \|_2, \qquad u = \frac{r}{\beta_0}, \qquad v = A^T u, \qquad \alpha_1 = \| v \|_2, \qquad v = \frac{v}{\alpha_1},$$

$$\bar{\varphi}_1 = \beta_0, \qquad \bar{\alpha}_1 = \alpha_1, \qquad w = v$$

2 – Iteration. For $i = 1, 2,\ldots$ until convergence.

2.1 – Compute u.

$$u = Av - \alpha_i u, \qquad \beta_i = \| u \|_2, \qquad u = \frac{u}{\beta_i}$$

2.2 – Compute Givens rotation $G_i = \begin{pmatrix} c & s \\ -s & c \end{pmatrix}$.

$$\text{if} \quad |\bar{\alpha}_i| \geq |\beta_i|, \qquad t = \frac{\beta_i}{\bar{\alpha}_i}, \qquad \eta_i = \sqrt{1+t^2}, \qquad c = \frac{1}{\eta_i}, \qquad s = tc, \qquad \gamma_i = \eta_i \alpha_i,$$

$$\text{else} \quad t = \frac{\bar{\alpha}_i}{\beta_i}, \qquad \eta_i = \sqrt{1+t^2}, \qquad s = \frac{1}{\eta_i}, \qquad c = ts, \qquad \gamma_i = \eta_i \beta_i$$

2.3 – Compute v.

$$v = A^T u - \beta_i v, \qquad \alpha_{i+1} = \| v \|_2, \qquad v = \frac{v}{\alpha_{i+1}}$$

2.4 – Update $\bar{\alpha}_{i+1}$ and φ_i.

$$\xi_i = s\alpha_{i+1}, \quad \bar{\alpha}_{i+1} = c\alpha_{i+1}, \quad \bar{\varphi}_{i+1} = -s\bar{\varphi}_i, \quad \varphi_i = c\bar{\varphi}_i$$

2.5 – Update h and w.

$$h = h + \frac{\varphi_i}{\gamma_i} w, \qquad w = v - \frac{\xi_i}{\gamma_i} w$$

2.6 – Check convergence $|\alpha_{k+1} c_k| / \|A\|_F \leq \tau$ with the given number τ.

Applying iteration LSQR on eq. (5), we get the corresponding algorithm LSQR_v (see [7] for details).

Iteration LSQR without Kronecker product

The two basic operations in iteration LSQR_v are Av and $A^T u$, where $v \in R^n$, $u \in R^{n^2}$. They will be very big when n increases, since the matrix A has Kronecker product, which will increase the computational complexity. So we intend to release it and get the corresponding matrix form iteration.

For the vector v there exists a circulant matrix $V \in R^{n\times n}$ such that:

$$vec(V) = C^T v$$

where

$$V = \sum_{i=1}^{n} v(i) G^{i-1}$$

so

$$Av = (I \otimes C_n) C_T v = (I \otimes C_n) vec(V) = vec(C_n V)$$

Now we want to correspond the vector $A^T u$ to the matrix $P_C(A^T u)$ in circulant matrix space. For this end, we should find the matrix U such that $U = vec(U)$, and represent $g_C(U)$ which is the co-ordinate of U. It is not difficult to verify that:

$$P_C[(C_T)^T u] = \sum_{i=0}^{n-1} trace(U^T G^j) G^j$$

and

$$g_C(U)_j = \frac{trace(U^T G^j)}{n}, \qquad j = 0,1,\cdots, \qquad n-1$$

Set

$$\bar{U} = (C_n)^T U$$

then

$$P_C[A^T vec(U)] = P_C[C_T{}^T vec(\bar{U})] = \sum_{i=0}^{n-1} trace(\bar{U}^T G^j) G^j$$

So algorithm LSQR_v can be rewritten as its matrix form iteration LSQR_M.

Iteration LSQR_M

1 – Initialization.

Set

$$H = H_0, \qquad R = I_n - C_n H_0, \qquad \beta_0 = \| R \|_F, \qquad U = \frac{R}{\beta_0}, \qquad \bar{U} = C_n{}^T U,$$

$$V_t = P_C[C_T{}^T vec(\bar{U})], \qquad \alpha_1 = \| g_C(V_t) \|_2, \qquad V = \frac{V_t}{\alpha_1}, \qquad \bar{\varphi}_1 = \beta_0,$$

$$\bar{\alpha}_1 = \alpha_1, \qquad W = V, \qquad N_\tau = 0$$

2 – Iteration. For $i = 1, 2, \cdots$ until convergence.

2.1 – Compute U.

$$U = C_n V - \alpha_i U, \qquad \beta_i = \| U \|_F, \qquad U = \frac{U}{\beta_i}$$

2.2 – Compute Givens rotation $G_i = \begin{pmatrix} c & s \\ -s & c \end{pmatrix}$.

$$\text{if} \quad |\bar{\alpha}_i| \geq |\beta_i|, \qquad t = \frac{\beta_i}{\bar{\alpha}_i}, \qquad \eta_i = \sqrt{1+t^2}, \qquad c = \frac{1}{\eta_i}, \qquad s = tc, \qquad \gamma_i = \eta_i \alpha_i,$$

$$\text{else} \quad t = \frac{\bar{\alpha}_i}{\beta_i}, \qquad \eta_i = \sqrt{1+t^2}, \qquad s = \frac{1}{\eta_i}, \qquad c = ts, \qquad \gamma_i = \eta_i \beta_i$$

2.3 – Compute:

$$\bar{U} = C_n{}^T U, \qquad V_t = P_C[C_T{}^T vec(\bar{U})], \qquad \bar{V} = V_t - \beta_i V, \qquad \alpha_{i+1} = \| g_C(V_t) \|_2,$$

$$V = \frac{\bar{V}}{\alpha_{i+1}}, \qquad N_\tau = N_\tau + \alpha_i{}^2 + \beta_i{}^2$$

2.4 – Update $\bar{\alpha}_{i+1}$ and φ_i.

$$\xi_i = s\alpha_{i+1}, \qquad \bar{\alpha}_{i+1} = c\alpha_{i+1}, \qquad \bar{\varphi}_{i+1} = -s\bar{\varphi}_i, \qquad \varphi_i = c\bar{\varphi}_i$$

2.5 – Update H and W.

$$H = H + \left(\frac{\varphi_i}{\gamma_i}\right) W, \qquad W = V - \left(\frac{\xi_i}{\gamma_i}\right) W$$

2.6 – Check convergence $|\alpha_{k+1} c_k| \leq (N_\tau)^{1/2}$ with the given number τ.

In iteration LSQR_M, the Kronecker product has been released, and the iteration only involve the matrix-matrix and matrix-vector product.

Numerical example

In this section, we test our iteration LSQR_M. The matrix size, n, varies from $n = 20$ to $n = 500$, the circulant matrix C_n is constructed:

$$c = rand(n,1), \qquad C_n = \sum_{i=1}^{n} c(i) G^{i-1}$$

where G is set by:

$$e = ones(n,1), \qquad Me = e\, e^T$$

$$G = diag[diag(Me,1),1] + diag[diag(Me,n-1),1-n]$$

Two errors are denoted by $\varepsilon_{\text{left}} = \| C_{\text{inv}} C_n - I_n \|_F$ and $\varepsilon_{\text{right}} = \| C_n C_{\text{inv}} - I_n \|_F$, respectively. For the given stopping criteria τ, the iteration numbers and the CPU time seem to depend on the matrix size n. As n increases, the central processing unit (CPU) time grows quickly, but $\varepsilon_{\text{left}}$ and $\varepsilon_{\text{right}}$ change a little. In tab. 1 we list the CPU time, $\varepsilon_{\text{left}}$, $\varepsilon_{\text{right}}$, and iteration numbers for different values of n with $\tau = 10^{-11}$, respectively.

Table 1. The LSQR_M for the inverse of circulant matrix

n	$\varepsilon_{\text{left}}$	$\varepsilon_{\text{right}}$	CPU time	Iteration numbers
20	$4.06514\cdot10^{-11}$	$4.06409\cdot10^{-11}$	0.01	14
50	$1.55306\cdot10^{-11}$	$1.55305\cdot10^{-11}$	0.24	37
100	$8.34321\cdot10^{-11}$	$8.34312\cdot10^{-11}$	1.01	68
200	$5.64657\cdot10^{-10}$	$5.64656\cdot10^{-10}$	10.32	133
300	$2.98367\cdot10^{-10}$	$2.98368\cdot10^{-10}$	40.44	177
400	$1.00367\cdot10^{-10}$	$1.00366\cdot10^{-10}$	127.34	189
500	$4.79357\cdot10^{-09}$	$4.79368\cdot10^{-09}$	200.46	215

Conclusion

This paper reports the iteration LSQR for the inverse of a circulant matrix, which is usually the key to a class of boundary value problems. Compared with the existing methods, our iteration only involves matrix-matrix product and it is suitable for all non-singular circulant matrix.

Acknowledgment

The research was supported in part by the National Natural Science Foundation of China, Natural Science Foundation of Zhejiang Province and Foundation for Young Talents of ZJGSU under Grant No. Y6110639, 11201422, 11571312, 1020XJ1314019.

References

[1] Liu, X. Y., *et al.*, Circulant Matrix and Conformal Mapping for Solving Partial Differential Equations, *Computers & Mathematics with Applications*, *68* (2014), 3, pp. 67-76
[2] He, J.-H., Effect on Temperature on Surface Tension of a Bubble and Hierarchical Ruptured Bubbles for Nanofiber Fabrication, *Thermal Science*, *16* (2012), 1, pp. 327-330
[3] He, J.-H., Variational Iteration Method – a Kind of Nonlinear Analytical Technique: Some Examples, *International Journal of Non-Linear Mechanics*, *34* (1999), 4, pp. 699-708
[4] Salimi, S., *et al.*, An Analytical Solution to the Thermal Problems with Varying Boundary Conditions around a Moving Source, *Applied Mathematical Modelling*, *40* (2016), 13, pp. 6690-6707
[5] Abassy, T. A., *et al.,* Toward a Modified Variational Iteration Method, *Journal of Computational and Applied Mathematics, 207* (2007), 1, pp. 137-147
[6] Lu, J. F., Ma, L., Analytical Approach to a Generalized Hirota-Satsuma Coupled Korteweg-de Vries Equations by Modified Variation Iteration Method, *Thermal Science, 20* (2016), 3, pp. 885-888
[7] Golub, G. H., Van Loan, C. F., Matrix Computations, in: *Special Linear Systems (Chapter 4)*, Johns Hopkins University Press, Baltimore, Md., USA, 3rd ed., 1996, pp. 133-205
[8] Chan, R. H., Jin, X. Q., *An Introduction to Iterative Toeplitz Solvers*, Society for Industrial and Applied Mathematics, Philadelphia, Penn., USA, 2007
[9] Chan, T. F., Hansen, P., A Look-Ahead Levinson Algorithm for Indefinite Toeplitz Systems, *SIAM J. Matrices Anal. Appl., 13* (1992), 2, pp.490-506
[10] Carmona, A., The Inverses of some Circulant Matrices, *Applied Mathematics and Computation, 270* (2015), 1, pp. 785-793
[11] Kailath, T., Sayed, A. H., Displacement Structure: Theory and Applications, *SIAM Reviews, 37* (1995), 3, pp. 297-386
[12] Paige, C. C., Saunders, A., LSQR: An Algorithm for Sparse Linear Equations and Sparse Least Squares, *ACM Transactions on Mathematical Software*, *8* (1982), 1, pp. 43-71

Paper submitted: July 15, 2016
Paper revised: August 27, 2016
Paper accepted: September 4, 2016