Tri-diagonal Transition Matrix
John Kormylo

One of the problems with state vector models is that they require $o(n^2)$ operations per step whereas an ARMA model (or its state vector equivalent) only requires $o(n)$ operations. However, the ARMA model is notoriously subject to numerical sensitivity problems, whereas state vector models can be formed with much more robust characteristics (e.g. the balanced realization).

As a compromise, I propose taking a state vector model and reducing it to tri-diagonal form. This would also require only $o(n)$ operations/step but would hopefully be more robust.

One could always perform an eigenvalue decomposition of the transition matrix, construct 2×2 submatrices from complex conjugate pairs, then string these along the main diagonal together with any real eigenvalues. This is, however, a lot more work than is actually needed.

Closed Form Solution

For any invertable matrix T one can show that the impulse response for a state vector model satisfies

$$h(k) = \mathbf{h}'\Phi^k\boldsymbol{\gamma} = (\mathbf{h}'T^{-1})(T\Phi T^{-1})^k(T\boldsymbol{\gamma}) \tag{1}$$

for $k = 0, 1, 2, \ldots$ (proof by induction). We will choose

$$T = T(n-1)T(n-2)\ldots T(2)T(1) \tag{2}$$

so as to reduce Φ to tri-diagonal form, where

$$T(i) = \begin{bmatrix} I_{i\times i} & 0 \\ 0 & Q(i) \end{bmatrix} \quad \text{for} \quad i = 1, 2, \ldots, n-1 \tag{3}$$

and $Q(i)$ is of the form

$$Q(i) = I - 2\mathbf{u}(i)\mathbf{v}'(i) \tag{4}$$

for some vectors $\mathbf{u}(i)$ and $\mathbf{v}(i)$ such that $\mathbf{v}'(i)\mathbf{u}(i) = 1$ (which may or may not qualify as a Householder transform).

One can easily show that $Q^{-1}(i) = Q(i)$ and $T^{-1}(i) = T(i)$ and therefore

$$T^{-1} = T(1)T(2)\ldots T(n-2)T(n-1) \quad . \tag{5}$$

At each step i of the reduction, the reduced transition matrix is of the form

$$\Phi(i) = \begin{bmatrix} & & & 0 \\ & A(i) & & \vdots \\ & & & 0 \\ & & & \mathbf{c}'(i) \\ 0 & \ldots & 0 \quad \mathbf{b}(i) & D(i) \end{bmatrix} \tag{6}$$

where $A(i)$ is an $(i \times i)$ tri-diagonal matrix. The next step is given by

$$\begin{aligned}\Phi(i+1) &= T(i)\Phi(i)T(i) \\ &= \begin{bmatrix} & A(i) & & 0 \\ & & & \vdots \\ & & & 0 \\ & & & \mathbf{c}'(i)Q(i) \\ 0 \quad \ldots & 0 & Q(i)\mathbf{b}(i) & Q(i)D(i)Q(i) \end{bmatrix} \quad . \end{aligned} \tag{7}$$

To be tri-diagonal, vectors $\mathbf{u}(i)$ and $\mathbf{v}(i)$ must be chosen to satisfy (dropping the index i)

$$Q\mathbf{b} = \mathbf{b} - 2\mathbf{u}(\mathbf{v}'\mathbf{b}) = \begin{bmatrix} \alpha \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{8}$$

and

$$Q'\mathbf{c} = \mathbf{c} - 2\mathbf{v}(\mathbf{u}'\mathbf{c}) = \begin{bmatrix} \beta \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{9}$$

for some unknown α and β.

Multiplying both sides of (8) by $\mathbf{v}'$ gives us

$$-\mathbf{v}'\mathbf{b} = v_1\alpha \tag{10}$$

while multiplying both sides of (9) by $\mathbf{u}'$ gives us

$$-\mathbf{u}'\mathbf{c} = u_1\beta \quad . \tag{11}$$

Substituting these back into (8) and (9) we get

$$b_1 = \alpha(1 - 2u_1v_1) \tag{12}$$

$$c_1 = \beta(1 - 2u_1v_1) \tag{13}$$

$$b_k = -2\alpha v_1 u_k \quad \forall \quad k > 1 \tag{14}$$

$$c_k = -2\beta u_1 v_k \quad \forall \quad k > 1 \tag{15}$$

and therefore

$$\begin{aligned}\mathbf{c}'\mathbf{b} &= \alpha\beta\big(1 - 4u_1v_1 + 4u_1v_1(\mathbf{v}'\mathbf{u})\big) \\ &= \alpha\beta \end{aligned} \tag{16}$$

since $\mathbf{v}'\mathbf{u} = 1$. Multiplying b_1 times c_1 using (12) and (13) and substituting for $\alpha\beta$ using (16), we get the solution

$$\frac{b_1 c_1}{\mathbf{c}'\mathbf{b}} = (1 - 2u_1 v_1)^2 \tag{17}$$

assuming that $\mathbf{c}'\mathbf{b} \neq 0$.

One can avoid the problem of having the left hand side of (17) being negative by swapping rows and columns. If $\mathbf{c}'\mathbf{b}$ is positive, search for the largest $b_k c_k$ and swap rows and columns with $b1$ and $c1$, respectively. Similarly, if $\mathbf{c}'\mathbf{b}$ is negative, search for the most negative $b_k c_k$. Also remember to swap the corresponding entries in $\mathbf{h}$ and $\boldsymbol{\gamma}$.

(Actually, instead of swapping rows and columns, one should move the k row up and k column left to the $(i+1)$ position, then push the intervening rows down and columns right one spot, for reasons which will not become apparent until later.)

As a solution to (17) we use

$$1 - 2u_1 v_1 = -\sqrt{\frac{b_1 c_1}{\mathbf{c}'\mathbf{b}}} \tag{18}$$

and therefore

$$u_1 v_1 = \frac{1}{2}\left(1 + \sqrt{\frac{b_1 c_1}{\mathbf{c}'\mathbf{b}}}\right) \tag{19}$$

so that $u_1 v_1 \neq 0$ even when $b_1 c_1 = \mathbf{c}'\mathbf{b}$. We can now solve for α and β using (12) and (13). Without loss of generality, we assume $u_1 = 1$ and solve for v_1 from (19), then solve for the remaing u_k and v_k using (14) and (15).

Special Cases

If $\mathbf{c}'\mathbf{b} = 0$, no transformation of the form of $T(i)$ from (3) will help. Any tranformation of the form of $T(j)$ for $j < i$ can and will be reversed when restoring the earlier rows and columns to tri-diagonal form. Only transformations involving the first row and column will not be undone while tri-diagonalizing the entire matrix again.

For example, one could move the last row and column to the first location and push the rest of the rows down and columns right one space. Every row and column completed pushes the problem closer to the bottom. If it is pushed to the last or next to last row and column, then the matrix is already tri-diagonalized.

If $\mathbf{c}'\mathbf{b} = 0$ occurs in the very first row and column, there is an alternative which always works. Specifically, one can use the transformation $T(0)\Phi T(0)$ where

$$T(0) = I - 2\mathbf{u}\mathbf{v}' \tag{20}$$

using

$$\mathbf{u} = \begin{bmatrix} 1 \\ \alpha\mathbf{c} \end{bmatrix} \quad \text{and} \quad \mathbf{v} = \begin{bmatrix} 1 \\ \beta\mathbf{b} \end{bmatrix} \tag{21}$$

so that $\mathbf{v}'\mathbf{u} = 1$ for any α and β.

Performing this transformation we see that

$$T(0)\begin{bmatrix} a & \mathbf{c}' \\ \mathbf{b} & D \end{bmatrix}T(0) = \begin{bmatrix} a & \mathbf{c}' \\ \mathbf{b} & D \end{bmatrix} - 2\mathbf{u}[\, a+\beta\mathbf{b}'\mathbf{b} \quad \mathbf{c}'+\beta\mathbf{b}'D\,]$$
$$- 2\begin{bmatrix} a+\alpha\mathbf{c}'\mathbf{c} \\ \mathbf{b}+\alpha D\mathbf{c} \end{bmatrix}\mathbf{v}' + 4\mathbf{u}(a+\beta\mathbf{b}'\mathbf{b}+\alpha\mathbf{c}'\mathbf{c}+\alpha\beta\mathbf{b}'D\mathbf{c})\mathbf{v}'$$
$$= \begin{bmatrix} a(0) & \mathbf{c}'(0) \\ \mathbf{b}(0) & D(0) \end{bmatrix} \tag{22}$$

where

$$a(0) = a + 2\alpha\mathbf{c}'\mathbf{c} + 2\beta\mathbf{b}'\mathbf{b} + 4\alpha\beta\mathbf{b}'D\mathbf{c} \tag{23}$$
$$\mathbf{b}(0) = -\mathbf{b} - 2\alpha D\mathbf{c} + 2\alpha(a+\beta\mathbf{b}'\mathbf{b}+2\alpha\mathbf{c}'\mathbf{c}+2\alpha\beta\mathbf{b}'D\mathbf{c})\mathbf{c} \tag{24}$$
$$\mathbf{c}'(0) = -\mathbf{c}' - 2\beta\mathbf{b}'D + 2\beta(a+\alpha\mathbf{c}'\mathbf{c}+2\beta\mathbf{b}'\mathbf{b}+2\alpha\beta\mathbf{b}'D\mathbf{c})\mathbf{b}' \tag{25}$$

and

$$D(0) = D - 2\alpha\mathbf{c}\mathbf{c}' - 2\beta\mathbf{b}\mathbf{b}') - 2\alpha\beta(\mathbf{c}\mathbf{b}'D + D\mathbf{c}\mathbf{b}')$$
$$+ 4\alpha\beta(a+\beta\mathbf{b}'\mathbf{b}+\alpha\mathbf{c}'\mathbf{c}+\alpha\beta\mathbf{b}'D\mathbf{c})\mathbf{c}\mathbf{b}' \quad . \tag{26}$$

Scale factors α and β are chosen so that $\mathbf{c}'(0)\mathbf{b}(0) \neq 0$ where from (24) and (25) we have

$$\mathbf{c}'(0)\mathbf{b}(0) = 2\beta(\mathbf{b}'D\mathbf{b} - a\mathbf{b}'\mathbf{b}) + 2\alpha(\mathbf{c}'D\mathbf{c} - a\mathbf{c}'\mathbf{c})$$
$$- 4\beta^2(\mathbf{b}'\mathbf{b})^2 - 4\alpha^2(\mathbf{c}'\mathbf{c})^2$$
$$+ 4\alpha\beta(\mathbf{b}'D^2\mathbf{c} - 2a\mathbf{b}'D\mathbf{c} - \mathbf{b}'\mathbf{b}\mathbf{c}'\mathbf{c})$$
$$- 8\alpha\beta(\mathbf{b}'D\mathbf{c})(\beta\mathbf{b}'\mathbf{b}+\alpha\mathbf{c}'\mathbf{c}) - 8\alpha^2\beta^2(\mathbf{b}'D\mathbf{c})^2 \quad . \tag{27}$$

While any non-zero solution will do, for numerical purposes one would like a solution as far from zero as possible, such as a local extremum.

Solving for α and β independently unfortunately requires finding the roots of a fifth order polynomial, and is probably overkill anyway. Instead, assume $\alpha = \beta$ and set the derivative to zero, yeilding the cubic polynomial

$$0 = 16\alpha^3(\mathbf{b}'D\mathbf{c})^2$$
$$+ 12\alpha^2(\mathbf{b}'D\mathbf{c})(\mathbf{b}'\mathbf{b}+\mathbf{c}'\mathbf{c})$$
$$+ 4\alpha\big((\mathbf{b}'\mathbf{b})^2 + (\mathbf{c}'\mathbf{c})^2 + (\mathbf{b}'\mathbf{b})(\mathbf{c}'\mathbf{c}) + 2a\mathbf{b}'D\mathbf{c} - \mathbf{b}'D^2\mathbf{c}\big)$$
$$+ a(\mathbf{b}'vecb + \mathbf{c}'\mathbf{c}) - \mathbf{b}'D\mathbf{b} - \mathbf{c}'D\mathbf{c} \quad . \tag{28}$$

Alternatively, set $\alpha = 1$ and use

$$\beta = \frac{2\mathbf{b}'D^2\mathbf{c} + \mathbf{b}'D\mathbf{b} - a\mathbf{b}'\mathbf{b} - 4a\mathbf{b}'D\mathbf{c} - 2(\mathbf{b}'\mathbf{b})(\mathbf{c}'\mathbf{c}) - 4(\mathbf{b}'D\mathbf{c})(\mathbf{c}'\mathbf{c})}{4\mathbf{b}'\mathbf{b} + 8(\mathbf{b}'D\mathbf{c})(\mathbf{b}'\mathbf{b}) + 8(\mathbf{b}'D\mathbf{c})^2} \tag{29}$$

or set $\beta = 1$ and use

$$\alpha = \frac{2\mathbf{b}'D^2\mathbf{c} + \mathbf{c}'D\mathbf{c} - a\mathbf{c}'\mathbf{c} - 4a\mathbf{b}'D\mathbf{c} - 2(\mathbf{b}'\mathbf{b})(\mathbf{c}'\mathbf{c}) - 4(\mathbf{b}'D\mathbf{c})(\mathbf{b}'\mathbf{b})}{4\mathbf{c}'\mathbf{c} + 8(\mathbf{b}'D\mathbf{c})(\mathbf{c}'\mathbf{c}) + 8(\mathbf{b}'D\mathbf{c})^2} \quad . \tag{30}$$

At least one of the above should yeild a useful solution.

Numerical Robustness

While the above algorithm will give *a* tri-diagonal solution, it is probably not the best tri-diagonal solution. However, it can be used as the initial guess for non-linear optimization for the tri-diagonal parameters. The only problem is that the gradients for least-squares go to zero as the error goes to zero, which may result in convergence before the best solution is found. One can overcome this problem by using single precision (or a DSP) for the state vector model and double precision for the error and gradient calculations.