Comput Econ (2008) 31:95–113
DOI 10.1007/s10614-007-9108-0

# Solving Linear Rational Expectations Models: A Horse Race

**Gary S. Anderson**

Received: 4 May 2006 / Accepted: 4 August 2007 / Published online: 14 September 2007
© Springer Science+Business Media, LLC 2007

**Abstract** This paper compares the generality, accuracy and computational speed of alternative approaches to solving linear rational expectations models, including the procedures of Sims (Solving linear rational expectations models, 1996), Anderson and Moore (Unpublished manuscript, 1983), Binder and Pesaran (Multivariate rational expectations models and macroeconometric modelling: A review and some new results, 1994), King and Watson (*International Economic review, 39*, 1015–1026, 1998), Klein (*Journal of Economic Dynamics and Control, 24*, 1405–1423, 1999), and Uhlig (A toolkit for analyzing nonlinear dynamic stochastic models easily, 1999). While all six procedures yield equivalent results for models with a unique stationary solution, the algorithm of Anderson and Moore (Unpublished manuscript, 1983) is the fastest and provides the highest accuracy; furthermore, the speed advantage increases with the size of the model.

**Keywords** Linear rational expectations · Blanchard–Kahn · Saddle point solution

## 1 Introduction and Summary

Since Blanchard and Kahn (1980) a number of alternative approaches for solving linear rational Expectations models have emerged. This paper describes, compares and contrasts the techniques of Gary Anderson (Unpublished manuscript), Gary Anderson and George Moore (Unpublished manuscript, 1985), Binder and Pesaran (1994), King and Watson (1998), Klein (1999), Sims (1996), and Uhlig (1999). All

G. S. Anderson (✉)
Board of Governors of the Federal Reserve System, Washington, DC, USA
e-mail: ganderson@frb.gov

these authors provide MATLAB code implementing their algorithm.[1] The paper compares the computational speed, generality and accuracy of these MATLAB implementations. The paper uses numerical examples to characterize practical differences in employing the alternative procedures.

Economists use the output of these procedures for simulating models, estimating models, computing impulse response functions, calculating asymptotic covariance, solving infinite horizon linear quadratic control problems and constructing terminal constraints for nonlinear models. All these applications benefit from the use of reliable, efficient and easy to use code.

A comparison of the algorithms reveals that:

- For models satisfying the Blanchard–Kahn conditions, the algorithms provide equivalent solutions.[2]
- The Anderson–Moore algorithm requires fewer floating point operations to achieve the same result. This computational advantage increases with the size of the model.
- While the Anderson–Moore, Sims and Binder–Pesaran approaches provide matrix output for accommodating arbitrary exogenous processes, the King–Watson and Uhlig implementations only provide solutions for VAR exogenous process.[3] Fortunately, there are straightforward formulae for augmenting the King–Watson, Uhlig and Klein approaches with the matrices characterizing the impact of arbitrary shocks.
- The Anderson–Moore suite of programs provides a simple modeling language for developing models. In addition, the Anderson–Moore implementation requires no special treatment for models with multiple lags and leads. To use each of the other algorithms, one must cast the model in a form with at most one lead or lag.[4] This can be a tedious and error prone task for models with more than a couple of equations.
- Using the Anderson–Moore algorithm to solve the quadratic matrix polynomial equation improves the performance of both Binder–Pesaran's and Uhlig's algorithms.

Section 2 states the problem and introduces notation. This paper divides the algorithms into three categories: eigensystem, QZ, and matrix polynomial methods. Section 3 describes the eigensystem methods. Section 4 describes applications of the QZ algorithm. Section 5 describes applications of the matrix polynomial approach. Section Appendix B compares the computational speed, generality and accuracy of the algorithms. Section 6 concludes the paper. The appendices provide usage notes for each of the algorithms as well as information about how to compare inputs and outputs from each of the algorithms.

[1] Although Broze, Gouriéroux, and Szafarz (1995) and Zadrozny (1998) describe algorithms, however, I was unable to locate MATLAB code implementing the algorithms.

[2] Blanchard and Kahn (1980) developed conditions for existence and uniqueness of linear rational expectations models. In their setup, the solution of the rational expectations model is unique if the number of unstable eigenvectors of the system is exactly equal to the number of forward-looking (control) variables.

[3] I modified Klein's MATLAB version to include this feature by translating the approach he used in his Gauss version.

[4] In addition to Anderson–Moore, Zadrozny's method also handles models with multiple leads and lags Zadrozny (1998).

## 2 Problem Statement and Notation

These algorithms compute solutions for models of the form

$$\sum_{i=-\tau}^{\theta} H_i x_{t+i} = \psi z_t, \quad t = 0, \ldots, \infty \tag{1}$$

with initial conditions, if any, given by constraints of the form

$$x_i = x_i^{data}, \quad i = -\tau, \ldots, -1 \tag{2}$$

where both $\tau$ and $\theta$ are non-negative, and $x_t$ is an L dimensional vector of endogenous variables with

$$\lim_{t \to \infty} \|x_t\| < \infty \tag{3}$$

and $z_t$ is a $k$ dimensional vector of exogenous variables.

Solutions of the homogeneous, $\psi = 0$, problem can be cast in the form

$$(x_t - x^*) = B_1 (x_{t-1} - x^*) \tag{4}$$

Given any algorithm that computes the $B_i$, one can easily compute other quantities useful for characterizing the impact of exogenous variables. For models with $\tau = \theta = 1$ the formulae are especially simple. A unique stable homogeneous solution of

$$H_{-1} x_{t-1} + H_0 x_t + H_1 x_{t+1} = 0 \tag{5}$$

produces a matrix $B$.

Now, let

$$\phi = (H_0 + H_1 B_1)^{-1} \tag{6}$$

$$F = -\phi H_1 B_1 \tag{7}$$

To solve Eq. 1, write

$$(x_t - x^*) = B_1 (x_{t-1} - x^*) + \sum_{s=0}^{\infty} F^s \phi \psi z_{t+s} \tag{8}$$

and when

$$z_{t+1} = \Upsilon z_t \tag{9}$$

$$vec(\vartheta) = (I - \Upsilon^T \otimes F)^{-1} vec(\phi\psi) \tag{10}$$

$$(x_t - x^*) = B_1(x_{t-1} - x^*) + \vartheta z_t \tag{11}$$

Consult Gary Anderson (Unpublished manuscript) for other useful formulae concerning rational expectations model solutions. The text which follows focuses on how the various algorithms compute the homogeneous solution.

## 3 Eigensystem Methods

### 3.1 The Anderson–Moore Algorithm

Gary Anderson (Unpublished manuscript), Anderson and Moore (1985) developed their algorithm in the mid 80's for solving rational expectations models that arise in large scale macro models.Appendix A.1 on page 11 provides a synopsis of the model concepts and algorithm inputs and outputs. The algorithm determines whether Eq. 1 has a unique solution, an infinity of solutions or no solutions at all.

The solution methodology entails

1. Manipulating Eq. 1 to compute a state space transition matrix, $A$, and auxiliary initial conditions identified in the computation of the transition matrix($Q_{aux}$).
2. Computing the eigenvalues and the left invariant space vectors—the invariant space associated with the explosive eigenvalues of $A^T$.
3. Combining the constraints provided by:
   (a) $Q_{aux}$ and
   (b) the invariant space vectors of $A^T$ ($Q_{unstab}$)
   to produce the matrix $Q = \begin{bmatrix} Q_{aux} \\ Q_{unstab} \end{bmatrix}$
4. using the initial conditions along with the matrix $Q$ to determine the existence and uniqueness of solutions satisfying

$$Q \begin{bmatrix} x_{t-1} \\ x_t \end{bmatrix} = 0 = \begin{bmatrix} Q_L & Q_R \end{bmatrix} \begin{bmatrix} x_{t-1} \\ x_t \end{bmatrix} \tag{12}$$

The invertibility of $Q_R$ guarantees the existence and uniqueness of solutions of the form $B_1 = Q_R^{-1} Q_L$ for 12. For details consult Anderson and Moore (1985).

Unlike the other approaches outlined here, the Anderson–Moore methodology does not explicitly distinguish between predetermined and non-predetermined variables. The algorithm assumes that history fixes the values of all variables dated prior to time t and that these initial conditions, the saddle point property terminal conditions, and the model equations determine all subsequent variable values.

### 3.2 King & Watson's Canonical Variables/System Reduction Method

King and Watson (1998) describe an alternative method for solving rational expectations models. Appendix A.2 provides a synopsis of the model concepts and algorithm inputs and outputs. The algorithm consists of two parts: system reduction for speed and canonical variables solution for solving the saddle point problem. Although their paper describes how to accommodate arbitrary exogenous shocks, the MATLAB function does not return the relevant matrices.

Unlike Anderson–Moore's procedure, King–Watson's procedure distinguishes between predetermined and non-predetermined variables. For comparison sake, divide the Anderson–Moore state variable into these two categories. Let $k_t$ correspond to the predetermined variables and $\Lambda_t$ correspond to the non-predetermined variables.

$$x_t = \begin{bmatrix} \Lambda_t \\ k_t \end{bmatrix} \tag{13}$$

we have

$$H_{-1}x_{t-1} + H_0 x_t + H_1 x_{t+1} \tag{14}$$

King and Watson treat models of the form

$$H_{-1} = \begin{bmatrix} 0 & P, \end{bmatrix}, \quad H_1 = \begin{bmatrix} N & 0 \end{bmatrix} \tag{15}$$

Similar to the Anderson–Moore procedure, the King–Watson procedure uses left eigenvectors to help characterize stable trajectories. However, King–Watson adapt a procedure developed by Gantmacher (1959) to compute "infinite eigenvalues" and an associated set of linear constraints to help them characterize unique stable trajectories. The constraints arising from the "infinite eigenvalues" play a role analagous to that played by the auxiliary initial conditions in the Anderson–Moore algorithm.

## 4 Applications of the QZ Algorithm

Several authors exploit the properties of the Generalized Schur Form(See Golub and van Loan (1989)). Here, as with all the algorithms except the Anderson–Moore algorithm, one must cast the model in a form with one lag and no leads or one lead and no lags.

### 4.1 Sims' QZ Method

Sims procedure also distinguishes between predetermined and non-predetermined. In addition, the approach also includes a new set of variables characterizing "expectational errors." For comparison sake, divide the Anderson–Moore state variable into these three categories. Let $k_t$ correspond to the predetermined variables and $\Lambda_t$ correspond to the non-predetermined variables and $\eta_t$ correspond to "expectational errors."

Sims (1996) QZ-Method solves linear rational expectations models of the form:

$$x_t = \begin{bmatrix} \Lambda_t \\ k_t \\ \eta_t \end{bmatrix} \tag{16}$$

$$H_{-1}x_{t-1} + H_0 x_t + H_1 x_{t+1} = \psi z_t \tag{17}$$

with

$$E_t \eta_{t+1} = 0 \tag{18}$$

$$H_{-1} = \begin{bmatrix} -\Gamma_{NN1} & -\Gamma_{NP1} & 0 \\ -\Gamma_{PN1} & -\Gamma_{PP1} & 0 \\ 0 & 0 & 0 \end{bmatrix}, H_0 = \begin{bmatrix} \Gamma_{NN0} & \Gamma_{NP0} & 0 \\ \Gamma_{PN0} & \Gamma_{PP0} & 0 \\ 0 & 0 & 0 \end{bmatrix},$$

$$H_1 = \begin{bmatrix} -\Pi & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & I \end{bmatrix}, \tag{19}$$

Appendix A.3 summarizes the Sims' QZ method model concepts and algorithm inputs and outputs.

The $\Pi$ matrix, consisting entirely of zeroes and ones, identifies the equations with "expectational" errors.

In Sims procedure, the QZ algorithm computes large eigenvalues and "infinite eigenvalue" which, along with the corresponding linear spaces characterizing unique stable trajectories.

### 4.2 Klein's Approach

Appendix A.4 summarizes the model concepts and algorithm inputs and outputs for Klein (1999). Although the paper characterizes the algorithm using the Complex Generalized Schur Decomposition, Klein's Matlab code shows how to use the more computationally efficient Real Generalized Schur Decomposition to solve the saddle point problem. Klein's procedure also distinguishes between predetermined and non-predetermined variables. The procedure solves models of the same form as King–Watson:

$$x_t = \begin{bmatrix} \Lambda_t \\ k_t \end{bmatrix} \tag{20}$$

$$H_{-1}x_{t-1} + H_0 x_t + H_1 x_{t+1} \tag{21}$$

with

$$H_{-1} = \begin{bmatrix} 0 & P \end{bmatrix}, \quad H_1 = \begin{bmatrix} N & 0 \end{bmatrix} \tag{22}$$

## 5 Applications of the Matrix Polynomial Approach

The next two methods rely on determining a matrix $C$ satisfying

$$H_1 C^2 + H_0 C + H_{-1} = 0. \tag{23}$$

Generally there are many solutions, but, when the homogeneous linear system has a unique saddle-path solution, only one of these solutions has all roots inside the unit circle. The Anderson–Moore algorithm constructs this unique stable matrix $C_{AM} = B$ that satisfies the quadratic matrix equation.

$$H_1 C_{AM}^2 + H_0 C_{AM} + H_{-1} = 0 \tag{24}$$

### 5.1 Binder & Pesaran's Method

Binder and Pesaran (1994) describe an iterative method for solving Eq. 23. Their procedure solve models of the form:

$$H_{-1} x_{t-1} + x_t + H_1 x_{t+1} = \psi z_t \tag{25}$$

According to Binder and Pesaran (1994), under certain conditions, the unique stable solution, if it exists, is given by:

$$x_t = C x_{t-1} + \sum_{i=0}^{\infty} F^i E(w_{t+1}) \tag{26}$$

where

$$F = (I - H_+ C)^{-1} H_- \tag{27}$$

and $C$ satisfies the quadratic Eq. 23.

Their Matlab program iterates

$$C_{k+1} = (I - H_+ C_k)^{-1} H_- \tag{28}$$

until $\|C_{k+1} - C_k\| < \epsilon$

### 5.2 Uhlig's Technique

Uhlig (1999) uses generalized eigenvalue calculations to obtain a solution for the matrix polynomial equation. Uhlig's procedure distinguishes between state variables and non-state variables. The procedure solves models of form similar to the King–Watson form:

$$x_t = \begin{bmatrix} x_{n,t} \\ x_{s,t} \end{bmatrix} \tag{29}$$

$$H_{-1}x_{t-1} + H_0 x_t + H_1 x_{t+1} = \psi z_t \tag{30}$$

with

$$H_{-1} = \begin{bmatrix} 0 & H_a \\ 0 & H_h \end{bmatrix}, \quad H_0 = \begin{bmatrix} H_c & H_b \\ H_k & H_g \end{bmatrix}, \quad H_1 = \begin{bmatrix} 0 & 0 \\ H_j & H_f \end{bmatrix}, \tag{31}$$

The specification excludes models with lagged values of the non state variables. Uhlig's technique preprocesses the input matrices to reduce the dimension of the quadratic matrix polynomial To solve the matrix polynomial problem, Uhlig employs a generalized eigenvalue computation Uhlig (1999). As is the case for Sims and Klein, the eigenvectors associated with large eigenvalues and "infinite eigenvalues" characterize the space of stable solutions.

### 5.3 Computational Speed

Nearly all the algorithms successfully computed solutions for all the examples. Each of the algorithms, except Binder–Pesaran's, successfully computed solutions for all of Uhlig's examples. Uhlig's algorithm failed to provide a solution for the given parametrization of one of King's examples. It is possible, however, Binder–Pesaran's and Uhlig's routines would solve alternative parametrization of the models.

Appendix Table 1 presents the MATLAB-reported floating point operations (flops) counts for each of the algorithms applied to the example models.

The first column of each table identifies the example model. The second column provides the flops required by the Anderson–Moore algorithm to compute $B$ followed by the flops required to compute $B$, $\vartheta$, $\phi$, and $F$. Columns three through seven report the flops required by each algorithm divided by the flops required by the Anderson–Moore algorithm for a given example model.

Note that the Anderson–Moore algorithm typically required a fraction of the number of flops required by the other algorithms. For example, King–Watson's algorithm required more than three times the flops required by the Anderson–Moore algorithm for the first Uhlig example. In the first row, one observes that Uhlig's algorithm required only 92% of the number of flops required by the Anderson–Moore algorithm, but this is the only instance where an alternative to the Anderson–Moore algorithm required fewer flops.

In general, Anderson–Moore provides solutions with the least computational effort. There were only a few cases where some alternative had approximately the same number of floating point operations. The speed advantage was especially pronounced for larger models. King–Watson generally used twice to three times the number of floating point operations. Sims generally used thirty times the number of floating point operations—never fewer than Anderson–Moore, King–Watson or Uhlig. It had about the same performance as Klein. Klein generally used thirty times the number of floating

point operations. It never used fewer than Anderson–Moore, King–Watson or Uhlig. Binder–Pesaran was consistently the most computationally expensive algorithm. It generally used hundreds of times more floating point operations. In one case, it took as many as 100,000 times the number of floating point operations. Uhlig generally used about twice the flops of Anderson–Moore even for small models and many more flops for larger models.

### 5.4 Numerical Accuracy

Appendix Tables 2–4 presents the MATLAB relative errors. Although $\vartheta$ and $F$ are relatively simple linear transformations of $B$, each of the authors uses radically different methods to compute these quantities. I have employed a symbolic algebra version of the Anderson–Moore algorithm to compute solutions to arbitrarily high precision.[5] I then compare the matrices computed in MATLAB by each algorithm to the high precision solution.

Anderson–Moore always computed the correct solution and in almost every case produced the most accurate solution. Relative errors were on the order of $10^{-16}$. King–Watson always computed the correct solution, but produced solutions with relative errors generally three times the size of the Anderson–Moore algorithm. Sims always computed correct solutions but produced solutions with relative errors generally five times the size of the Anderson–Moore algorithm.[6] Sim's F calculation produced errors that were 20 times the size of the Anderson–Moore relative errors. Klein always computed the correct solution but produced solutions with relative errors generally five times the size of the Anderson–Moore algorithm.

Uhlig provides accurate solutions with relative errors about twice the size of the Anderson–Moore algorithm for each case for which it converges. It cannot provide a solution for King's example 3 for the particular parametrization I employed. For the $\vartheta$ computation, the results were similar. The algorithm was unable to compute $\vartheta$ for King example 3. Errors were generally 10 times the size of Anderson–Moore relative errors.

The Binder–Pesaran algorithm does not converge to the unique stable saddle path solution for some of the example models. In each case, the resulting matrix solves the quadratic matrix polynomial, but the particular solution has an eigenvalue greater than one in magnitude even though an alternative matrix solution exists with eigen-

[5] Using Mathematica I was able to compute exact(i.e. InfinitePrecision) solutions for nearly all of these relatively small example models. This was possible because of the low degree of the characteristic polynomials involved in the eigenvalue/eigenvector calculation. I was not able to compute exact solutions for the BP1, BP2 and Klein1 models. For these three examples, I used Mathematica to compute solutions with 300 digits of precision- much greater than the 16 digits of machine precision available for the Matlab computations.

[6] To compare $F$ for Sims note that

$$\psi = I_L \tag{32}$$

$$F = (\Theta_y.\Theta_f)\phi_{exact}^{-1} \tag{33}$$

values less than unity. Furthermore, for Uhlig's example 3, the algorithm diverges and produces a matrix with NaN's. Even when the algorithm converges to approximate the stable saddle path solution, the discrepancies between the computed solution and the exact solution are much larger than for the other algorithms. One could tighten the convergence criterion at the expense of increasing computational time, but the algorithm is already the slowest of the algorithms evaluated. Binder–Pesaran's algorithm does not converge for either of Sims' examples. The algorithm provides accurate answers for King & Watson's examples. The $\vartheta$ and $F$ results were similar to the $B$ results. The algorithm was unable to compute $H$ for Uhlig 3 in addition to Uhlig 7. It computed the wrong value for Uhlig 6. It was unable to compute values for either of Sims's examples.

## 6 Conclusions

A comparison of the algorithms reveals that:

- With the exception of Binder–Pesaran, the algorithms consistently provided numerically accurate solutions for a range of example models satisfying the Blanchard–Kahn conditions.
- The Anderson–Moore algorithm proved to be the most accurate.
- Using the Anderson–Moore algorithm to solve the quadratic matrix polynomial equation improves the performance of both Binder–Pesaran's and Uhlig's algorithms.
- The Anderson–Moore algorithm requires fewer floating point operations to achieve the same result. This computational advantage increases with the size of the model.
- The Anderson–Moore suite of programs provides a simple modeling language for developing models.
- The Anderson–Moore and Zadrozny's algorithm require no special treatment for models with multiple lags and leads. To use each of the other algorithms, one must cast the model in a form with at most one lead or lag. This can be tedious and error prone task for models with more than a couple of equations.

**Acknowledgements** I would like to thank Robert Tetlow, Andrew Levin and Brian Madigan and an anonymous referee for useful discussions and suggestions. I would like to thank Ed Yao for valuable help in obtaining and installing the MATLAB code. The views expressed in this document are my own and do not necessarily reflect the position of the Federal Reserve Board or the Federal Reserve System.

## Appendix A Model Concepts

The following sections present the inputs and outputs for each of the algorithms for the following simple example:

$$V_{t+1} = (1 + R)V_t - D_{t+1} \tag{34}$$

$$D = (1 - \delta)D_{t-1} \tag{35}$$

**Appendix A.1** Anderson–Moore

| Model variable | Description | Dimensions |
|---|---|---|
| | Inputs | |
| | $\sum_{i=-\tau}^{\theta} H_i x_{t+i} = \psi z_t$ (36) | |
| $x_{t-\tau}, \ldots, x_t, \ldots, x_{t+\theta}$ | Model variables | $L(\tau+\theta) \times 1$ |
| $z_t$ | Exogenous variables | $M \times 1$ |
| $\theta$ | Longest lead | $1 \times 1$ |
| $\tau$ | Longest lag | $1 \times 1$ |
| $H_i$ | Structural coefficients matrix | $(L \times L)(\tau+\theta+1)$ |
| $\psi$ | Exogenous shock coefficients matrix | $L \times M$ |
| $\Upsilon$ | Optional exogenous VAR coefficients matrix ($z_{t+1} = \Upsilon z_t$) | $M \times M$ |
| | Outputs | |
| | $x_t = B \begin{bmatrix} x_{t-\tau} \\ \vdots \\ x_{t-1} \end{bmatrix} + \begin{bmatrix} 0 \ldots 0 \ I \end{bmatrix} \sum_{s=0}^{\infty} \left( F^s \begin{bmatrix} 0 \\ \phi\psi z_{t+s} \end{bmatrix} \right)$ (37) | |
| $B$ | Reduced form coefficients matrix | $L \times L(\tau+\theta)$ |
| $\phi$ | Exogenous shock scaling matrix | $L \times L$ |
| $F$ | Exogenous shock transfer matrix | $L\theta \times L\theta$ |
| $\vartheta$ | Autoregressive shock transfer matrix when $z_{t+1} = \Upsilon z_t$ the infinite sum simplifies to give $x_t = B \begin{bmatrix} x_{t-\tau} \\ \vdots \\ x_{t-1} \end{bmatrix} + \vartheta z_t$ | $L \times M$ |

Anderson–Moore input:

AIM modeling language input parameter file input

$$H = \begin{bmatrix} 0 & 0 & -1.1 & 0 & 1 & 1. \\ 0 & -0.7 & 0 & 1 & 0 & 0 \end{bmatrix}, \quad \psi = \begin{bmatrix} 4. & 1. \\ 3. & -2. \end{bmatrix}, \quad \Upsilon \begin{bmatrix} 0.9 & 0.1 \\ 0.05 & 0.2 \end{bmatrix} \tag{38}$$

Produces output:

$$B = \begin{bmatrix} 0. & 1.225 \\ 0. & 0.7 \end{bmatrix} F = \begin{bmatrix} 0.909091 & 0.909091 \\ 0. & 0. \end{bmatrix} \phi = \begin{bmatrix} -0.909091 & 1.75 \\ 0. & 1. \end{bmatrix} \tag{39}$$

$$\phi\psi = \begin{bmatrix} 1.61364 & -4.40909 \\ 3. & -2. \end{bmatrix} \vartheta = \begin{bmatrix} 21.0857 & -3.15714 \\ 3. & -2. \end{bmatrix} \tag{40}$$

Usage notes for Anderson–Moore algorithm

1. "Align" model variables so that the data history (without applying model equations), completely determines all of $x_{t-1}$, but none of $x_t$.
2. Develop a "model file" containing the model equations written in the "AIM modeling language."
3. Apply the model pre-processor to create MATLAB programs for initializing the algorithm's input matrix,$(H)$. Create $\Psi$ and, optionally, $\Upsilon$ matrices.
4. Execute the MATLAB programs to generate $B$, $\phi$, $F$ and optionally $\vartheta$.

Users can obtain code for the algorithm and the preprocessor from the author http://www.bog.frb.fed.us/pubs/oss/oss4/aimindex.html July, 1999.

**Appendix A.2** King–Watson

| Model variable | Description | Dimensions |
|---|---|---|
| | Inputs | |
| | $x_t = \begin{bmatrix} \Lambda_t \\ k_t \end{bmatrix}$ (41) | |
| | $H_{-1}x_{t-1} + H_0 x_t + H_1 x_{t+1} = \psi z_t$ (42) | |
| with | | |
| | $H_{-1} = \begin{bmatrix} 0 \\ B_2 \end{bmatrix}, \quad H_0 = \begin{bmatrix} A_2 \\ B_1 \end{bmatrix}, \quad H_1 = \begin{bmatrix} A_1 & 0 \end{bmatrix}$ (43) | |
| $\theta$ | Longest lead | $1 \times 1$ |
| $y_t$ | Endogenous variables | $m \times 1$ |
| $\Lambda_t$ | Non-predetermined endogenous variables | $(m - p) \times 1$ |
| $k_t$ | Predetermined endogenous variables | $p \times 1$ |
| $x_t$ | Exogenous variables | $n_x \times 1$ |
| $\delta_t$ | Exogenous variables | $n_\delta \times 1$ |
| A | Structural coefficients matrix associated with lead endogenous variables, $y_{t+1}$ | $m \times m$ |
| B | Structural coefficients matrix associated with contemporaneous endogenous variables, $y_t$ | $m \times m$ |
| $C_i$ | Structural coefficients matrix associated with contemporaneous and lead exogenous variables, $x_t$ | $m \times n$ |
| Q | Structural coefficients matrix associated with contemporaneous exogenous variables, $x_t$ | $n_x \times n_\delta$ |
| $\rho$ | Vector autoregression matrix for exogenous variables | $n_\delta \times n_\delta$ |
| G | Matrix multiplying Exogenous Shock | $n_\delta \times 1$ |
| | Outputs | |
| | $y_t = \Pi s_t$ (44) | |
| | $s_t = M s_{t-1} + \bar{G}\epsilon_t$ (45) | |
| | $s_t = \begin{bmatrix} k_t \\ \delta_t \end{bmatrix}$ (46) | |
| $s_t$ | Exogenous variables and predetermined variables | $(n_x + p) \times 1$ |
| $\Pi$ | Matrix relating endogenous variables to exogenous and predetermined variables | $m \times (p + n_x)$ |
| $M$ | | $(p + n_x) \times (p + n_x)$ |
| $\bar{G}$ | Matrix multiplying exogenous shock | $(n_x + p) \times l$ |

King–Watson input:

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & -1. \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -1.1 & 0 \\ 0 & -0.7 & 0 & 1 \end{bmatrix}, \quad C = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 4. & 1. \\ 3. & -2. \end{bmatrix}, \tag{47}$$

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \rho = \begin{bmatrix} 0.9 & 0.1 \\ 0.05 & 0.2 \end{bmatrix}, \quad G = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{48}$$

Produces output:

$$\Pi = \begin{bmatrix} 1. & 0. & 0. & 0. \\ 0. & 1. & 0. & 0. \\ 0. & 1.225 & -21.0857 & 3.15714 \\ 0. & 0.7 & -3. & 2. \end{bmatrix}, \quad M = \begin{bmatrix} 0. & 1.225 & -21.0857 & 3.15714 \\ 0. & 0.7 & -3. & 2. \\ 0. & 0. & 0.9 & 0.1 \\ 0. & 0. & 0.05 & 0.2 \end{bmatrix} \tag{49}$$

**Appendix A.3** Sims

| Model variable | Description | Dimensions |
|---|---|---|
| | Inputs | |
| | $\Gamma_0 y_t = \Gamma_1 y_{t-1} + C + \psi z_t + \Pi \eta_t$ (50) | |
| $y_t$ | State variables | $L \times 1$ |
| $z_t$ | Exogenous variables | $M_1 \times 1$ |
| $\eta_t$ | Expectational error | $M_2 \times 1$ |
| $\Gamma_0$ | Structural coefficients matrix | $L \times L$ |
| $\Gamma_1$ | Structural coefficients matrix | $L \times L$ |
| $C$ | Constants | $L \times 1$ |
| $\psi$ | Structural exogenous variables coefficients matrix | $L \times M_1$ |
| $\Pi$ | Structural expectational errors coefficients matrix | $L \times M_2$ |
| | Outputs | |
| | $y_t = \Theta_1 y_{t-1} + \Theta_c + \Theta_0 z_t + \Theta_y \sum_{s=1}^{\infty} \Theta_f^{s-1} \Theta_z E_t z_{t+s}$ (51) | |
| $\Theta_1$ | | $L \times L$ |
| $\Theta_c$ | | $L \times 1$ |
| $\Theta_0$ | | $L \times M_1$ |
| $\Theta_y$ | | $L \times M_2$ |
| $\Theta_f$ | | $M_2 \times M_2$ |
| $\Theta_z$ | | $M_2 \times M_1$ |

Sims input:

$$\Gamma_0 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -1.1 & 0 & 1 & 1. \\ 0 & 1 & 0 & 0 \end{bmatrix}, \quad \Gamma_1 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0.7 & 0 & 0 \end{bmatrix}, \quad C = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \tag{52}$$

$$\psi = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 4. & 1. \\ 3. & -2. \end{bmatrix}, \Pi = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{53}$$

Produces output:

$$\Theta_c \begin{bmatrix} 0. \\ 0. \\ 0. \\ 0. \end{bmatrix}, \quad \Theta_0 = \begin{bmatrix} 1.61364 & -4.40909 \\ 3. & -2. \\ 3.675 & -2.45 \\ 2.1 & -1.4 \end{bmatrix}, \quad \Theta_f = \begin{bmatrix} 0.909091 & 1.23405 \\ 0. & 0. \end{bmatrix}, \tag{54}$$

$$\Theta_y = \begin{bmatrix} 1.28794 & 1.74832 \\ -2.2204510^{-16} & -1.11022\,10^{-16} \\ 1.41673 & 0.702491 \\ -1.11022\,10^{-16} & 1.22066 \end{bmatrix}, \quad \Theta_z = \begin{bmatrix} -0.0796719 & -2.29972 \\ 2.4577 & -1.63846 \end{bmatrix} \tag{55}$$

$$\Theta_y \Theta_z = \begin{bmatrix} 4.19421 & -5.82645 \\ -2.55168\,10^{-16} & 6.92546\,10^{-16} \\ 1.61364 & -4.40909 \\ 3. & -2. \end{bmatrix}. \tag{56}$$

**Appendix A.4** Klein

| Model variable | Description | Dimensions |
|---|---|---|

Inputs

$$A\mathcal{E}x_{t+1} = Bx_t + Cz_{t+1} + Dz_t \tag{57}$$

$$z_{t+1} = \phi z_t + \epsilon_t x_t = \begin{bmatrix} k_t \\ u_t \end{bmatrix} \tag{58}$$

| | | |
|---|---|---|
| $x_t$ | Endogenous variables | $L \times 1$ |
| $z_t$ | Exogenous variables | $M \times 1$ |
| $n_k$ | The number of state variables | $M \times 1$ |
| $k_t$ | State variables | $n_k \times 1$ |
| $u_t$ | The non-state variables | $(L - n_k) \times 1$ |
| $A$ | Structural coefficients matrix for future variables | $L \times L$ |
| $B$ | Structural coefficient matrix for contemporaneous variables | $L \times L$ |

Outputs

$$u_t = Fk_t \tag{59}$$

$$k_t = Pk_{t-1} \tag{60}$$

| | | |
|---|---|---|
| $F$ | Decision rule | $(L - n_k) \times n_k$ |
| $P$ | Law of motion | $n_k \times n_k$ |

Klein input:

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & -1 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -1.1 & 0 \\ 0 & -0.7 & 0 & 1 \end{bmatrix}, \quad C = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 4 & 1 \\ 3 & -2 \end{bmatrix}, \tag{61}$$

Produces output:

$$P = \begin{bmatrix} 0.9 & 0.1 \\ 0.05 & 0.2 \end{bmatrix}, \quad F = \begin{bmatrix} -21.0857 & 3.15714 \\ -3 & 2 \end{bmatrix}. \tag{62}$$

**Appendix A.5** Binder-Pesaran

| Model variable | Description | Dimensions |
|---|---|---|

Inputs

$$\hat{C}x_t = \hat{A}x_{t-1} + \hat{B}\mathcal{E}x_{t+1} + D_1 w_t + D_2 w_{t+1} \tag{63}$$

$$w_t = \Gamma w_{t-1} \tag{64}$$

| | | |
|---|---|---|
| $x_t$ | State variables | $L \times 1$ |
| $w_t$ | Exogenous variables | $M \times 1$ |
| $\hat{A}$ | Structural coefficients matrix | $L \times L$ |
| $\hat{B}$ | Exogenous shock coefficients matrix | $L \times L$ |
| $\hat{C}$ | Exogenous shock coefficients matrix | $L \times M$ |
| $D_1$ | Exogenous shock coefficients matrix | $L \times M$ |
| $D_2$ | Exogenous shock coefficients matrix | $L \times M$ |
| $\Gamma$ | Exogenous shock coefficients matrix | $L \times M$ |

Outputs

$$x_t = Cx_{t-1} + Hw_t \tag{65}$$

$$x_t = Cx_{t-1} + \sum_{i=0}^{\infty} F^i E(w_{t+1}) \tag{66}$$

**Appendix A.5** continued

| Model variable | Description | Dimensions |
|---|---|---|
| $C$ | Reduced form comprised of convergence constraints | $L \times L(\tau + \theta)$ |
| $H$ | Exogenous shock scaling matrix | $L \times M$ |

Binder–Pesaran input:

$$A = \begin{bmatrix} 0 & 0 \\ 0 & 0.7 \end{bmatrix}, \quad B = \begin{bmatrix} -1 & -1 \\ 0 & 0 \end{bmatrix}, \quad C = \begin{bmatrix} -1.1 & 0 \\ 0 & 1 \end{bmatrix}, \quad D_1 = \begin{bmatrix} 4 & 1 \\ 3 & -2 \end{bmatrix} \tag{67}$$

$$D_2 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad \Gamma = \begin{bmatrix} 0.9 & 0.1 \\ 0.05 & 0.2 \end{bmatrix} \tag{68}$$

Produces output:

$$C = \begin{bmatrix} 0 & 1.225 \\ 0 & 0.7 \end{bmatrix}, \quad H = \begin{bmatrix} 21.0857 & -3.15714 \\ 3 & -2 \end{bmatrix}. \tag{69}$$

**Appendix A.6** Using

| Model variable | Description | Dimensions |
|---|---|---|
| | Inputs | |
| | $Ax_t + Bx_{t-1} + Cy_t + Dz_t = 0$ (70) | |
| | $\mathcal{E}(Fx_{t+1} + Gx_t + Hx_{t-1} + Jy_{t+1} + Ky_t + Lz_{t+1} + Mz_t)$ (71) | |
| | $z_{t+1} = Nz_t + \epsilon_{t+1}$ (72) | |
| | $\mathcal{E}\epsilon_{t+1} = 0$ (73) | |
| $x_t$ | State variables | $m \times 1$ |
| $y_t$ | Endogenous "jump" variables | $n \times 1$ |
| $z_t$ | Exogenous variables | $k \times 1$ |
| $A, B$ | Structural coefficients matrix | $l \times m$ |
| $C$ | Structural coefficients matrix | $l \times n$ |
| $D$ | Structural coefficients matrix | $l \times k$ |
| $F, G, H$ | Structural coefficients matrix | $(m + n - l) \times m$ |
| $J, K$ | Structural coefficients matrix | $(m + n - l) \times n$ |
| $L, M$ | Structural coefficients matrix | $m + n - l \times k$ |
| $N$ | Structural coefficients matrix | $k \times k$ |
| | Outputs | |
| | $x_t = Px_{t-1} + Qz_t$ (74) | |
| | $y_t = Rx_{t-1} + Sz_t$ (75) | |
| $P$ | | $m \times m$ |
| $Q$ | | $m \times k$ |
| $R$ | | $n \times m$ |
| $S$ | | $n \times k$ |

For Uhlig cannot find $C$ with the appropriate rank condition without augmenting the system with a "dummy variable" and equation like $W_t = D_t + V_t$.

Uhlig input:

$$A = \begin{bmatrix} 1 & 0 \\ 1 & -1 \end{bmatrix}, \quad B = \begin{bmatrix} -0.7 & 0 \\ 0 & 0 \end{bmatrix}, \quad C = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad D = \begin{bmatrix} -3 & 2 \\ 0 & 0 \end{bmatrix} \tag{76}$$

$$F = \begin{bmatrix} 1. & 0 \end{bmatrix}, \quad G = \begin{bmatrix} 0 & 0 \end{bmatrix}, \quad H = \begin{bmatrix} 0 & 0 \end{bmatrix}, \quad J = \begin{bmatrix} 1 \end{bmatrix}, \quad K = \begin{bmatrix} -1.1 \end{bmatrix} \tag{77}$$

**Appendix A.6** continued

| Model variable | Description | Dimensions |
|---|---|---|

$$L = \begin{bmatrix} 0 & 0 \end{bmatrix}, \quad M = \begin{bmatrix} -4 & -1 \end{bmatrix}, \quad N = \begin{bmatrix} 0.9 & 0.1 \\ 0.05 & 0.2 \end{bmatrix} \tag{78}$$

Produces output:

$$P = \begin{bmatrix} 0.7 & 0 \\ 1.925 & 6.16298\ 10^{-33} \end{bmatrix}, \quad Q = \begin{bmatrix} 3 & -2 \\ 24.0857 & -5.15714 \end{bmatrix} \tag{79}$$

$$R = \begin{bmatrix} 1.225 & 6.16298\ 10^{-33} \end{bmatrix}, \quad S = \begin{bmatrix} 21.0857 & -3.15714 \end{bmatrix} \tag{80}$$

## Appendix B Computational Speed

**Table 1** Matlab flop counts

| Model | AIM $(L, \tau, \theta)$ | KW $(m, p)$ | Sims $(L, M_2)$ | Klein $(L, n_k)$ | BP $L$ | Uhlig $(m, n)$ |
|---|---|---|---|---|---|---|
| Computes | $(B, \vartheta, \phi, F)$ | $(B, \vartheta)$ | $(B, \phi, F)$ | $(B)$ | $(B, \vartheta, \phi, F)$ | $(B, \vartheta)$ |
| Uhlig 1 | (2514, 4098) | 3.38385 | 23.9387 | 28.8007 | 88.7832 | 0.922832 |
| | (4,1,1) | (8,4) | (8,4) | (8,3) | 4 | (3,1) |
| Uhlig 2 | (6878, 9817) | 3.81346 | 46.9301 | 29.1669 | 15347.3 | 1.88979 |
| | (6,1,1) | (12,6) | (12,6) | (12,5) | 6 | (5,1) |
| Uhlig 3 | (6798, 9773) | 3.292 | 52.2499 | 35.7318 | 6468.78 | 1.80494 |
| | (6,1,1) | (12,6) | (12,6) | (12,5) | 6 | (5,1) |
| Uhlig 4 | (112555, 276912) | 3.62782 | 40.5874 | 22.6611 | 205753. | 16.7633 |
| | (14,1,1) | (28,14) | (28,14) | (28,13) | 14 | (10,4) |
| Uhlig 5 | (6850, 9789) | 3.58292 | 45.675 | 29.5441 | 16250.5 | 1.89752 |
| | (6,1,1) | (12,6) | (12,6) | (12,5) | 6 | (5,1) |
| Uhlig 6 | (6850, 9789) | 3.58292 | 45.675 | 29.5441 | 16250.5 | 1.89752 |
| | (6,1,1) | (12,6) | (12,6) | (12,5) | 6 | (5,1) |
| Uhlig 7 | (7809, 26235) | 2.53182 | 47.5947 | 46.3196 | 190.905 | 3.42169 |
| | (6,1,1) | (12,6) | (12,6) | (12,4) | 6 | (4,2) |
| Uhlig 8 | (82320, 108000) | 2.8447 | 48.1792 | 28.9946 | 131.785 | 2.60423 |
| | (13,1,1) | (26,13) | (26,13) | (26,11) | 13 | (11,2) |
| King 1 | (2288, 2566) | 2.36713 | 9.64904 | 20.7072 | 165.623 | 1.76311 |
| | (3,1,1) | (6,3) | (6,3) | (6,1) | 3 | (2,1) |
| King 2 | (1028, 2306) | 2.81809 | 21.1284 | 77.1858 | 368.545 | *NA* |
| | (3,1,1) | (6,3) | (6,3) | (6,−1) | 3 | (2,1) |
| King 3 | (40967, 146710) | 5.2211 | 39.8819 | 30.493 | 185.299 | 14.2569 |
| | (9,1,1) | (18,9) | (18,9) | (18,5) | 9 | (3,6) |
| Sims 1 | (3501, 5576) | 4.70923 | 59.9526 | 59.2871 | *NA* | 5.72694 |
| | (5,1,1) | (10,5) | (10,5) | (10,3) | 5 | (4,1) |
| Sims 2 | (16662, 39249) | 4.20292 | 56.8375 | 48.4135 | *NA* | 9.2849 |
| | (8,1,1) | (16,8) | (16,8) | (16,5) | 8 | (6,2) |
| Klein 1 | (3610, 15147) | 2.13213 | 10.2756 | 16.7828 | 35555.2 | 2.30526 |
| | (3,1,1) | (6,3) | (6,3) | (6,2) | 3 | (1,2) |
| BP 1 | (533, 1586) | 3.1257 | 13.8987 | 59.7749 | 613.814 | 2.88743 |
| | (2,1,1) | (4,2) | (4,2) | (4,0) | 2 | (1,1) |

**Table 1** continued

| Model | AIM $(L, \tau, \theta)$ | KW $(m, p)$ | Sims $(L, M_2)$ | Klein $(L, n_k)$ | BP $L$ | Uhlig $(m, n)$ |
|---|---|---|---|---|---|---|
| Computes | $(B, \vartheta, \phi, F)$ | $(B, \vartheta)$ | $(B, \phi, F)$ | $(B)$ | $(B, \vartheta, \phi, F)$ | $(B, \vartheta)$ |
| BP 2 | (3510, 5659)<br>(5,1,1) | 3.5265<br>(10,5) | 77.2456<br>(10,5) | 67.7846<br>(10,1) | 800.98<br>5 | 21.4259<br>(4,1) |

For $L, \tau, \theta, B, \vartheta, \phi, F$ seeAppendix A.1. For $m, p$, seeAppendix A.2. For $L, M_2$, seeAppendix A.3. For $L, n_k$, seeAppendix A.4. For $L$, seeAppendix A.5. For $m, n$, seeAppendix A.6. KW, Sims, Klein, BP, Uhlig column not normalized by dividing by comparable AIM value. For $m, n$, seeAppendix A.6
Column two presents the flop counts for computing the matrix $B$ followed by the additional flops required to compute $\vartheta, \phi$, and $F$. Columns 3–7 provide the flop counts as a multiple of the values in column one
The small integers in each column refer to the problem dimensions for each of the example models

## Appendix C Accuracy

**Table 2** Matlab relative errors in $B \frac{\|B_i - B_{exact}\|}{\|B_{exact}\|}$

| Model | AIM $(L, \tau, \theta)$ | KW $(m, p)$ | Sims $(L, M_2)$ | Klein $(L, n_k)$ | BP $L$ | Uhlig $(m, n)$ |
|---|---|---|---|---|---|---|
| Computes | $(B, \vartheta, \phi, F)$ | $(B, \vartheta)$ | $(B, \phi, F)$ | $(B)$ | $(B, \vartheta, \phi, F)$ | $(B, \vartheta)$ |
| Uhlig 0 | $1 := 1.69446\,10^{-16}$<br>(4,1,1) | 8.<br>(8,4) | 40.<br>(8,4) | 11.<br>(8,3) | 134848.<br>4 | 49.<br>(3,1) |
| Uhlig 1 | $1 := 3.82375\,10^{-15}$<br>(6,1,1) | 0.518098<br>(12,6) | 2.63371<br>(12,6) | 0.617504<br>(12,5) | 10217.1<br>6 | 3.32793<br>(5,1) |
| Uhlig 2 | $1 := 4.88785\,10^{-15}$<br>(6,1,1) | 1.<br>(12,6) | 2.00589<br>(12,6) | 6.43964<br>(12,5) | 9520.85<br>6 | 2.82951<br>(5,1) |
| Uhlig 3 | $1 := 1.15015\,10^{-15}$<br>(14,1,1) | 2.65517<br>(28,14) | 1.78046<br>(28,14) | 1.45676<br>(28,13) | $8.57994\,10^{14}$<br>14 | 20.9358<br>(10,4) |
| Uhlig 4 | $1 := 1.18357\,10^{-15}$<br>(6,1,1) | 2.2561<br>(12,6) | 1.00348<br>(12,6) | 14.7909<br>(12,5) | 10673.4<br>6 | 34.6115<br>(5,1) |
| Uhlig 5 | $1 := 1.18357\,10^{-15}$<br>(6,1,1) | 2.2561<br>(12,6) | 1.00348<br>(12,6) | 14.7909<br>(12,5) | 10673.4<br>6 | 34.6115<br>(5,1) |
| Uhlig 6 | $1 := 1.60458\,10^{-15}$<br>(6,1,1) | 7.32281<br>(12,6) | 38.3459<br>(12,6) | 13.4887<br>(12,4) | $7.63524\,10^{13}$<br>6 | 203.291<br>(4,2) |
| Uhlig 7 | $1 := 2.33332\,10^{-14}$<br>(13,1,1) | 7.59657<br>(26,13) | 4.15248<br>(26,13) | 0.335751<br>(26,11) | *NA*<br>13 | 2.63499<br>(11,2) |
| King 2 | $1 := 0.$<br>(3,1,1) | $3.65169\,10^{-16}$<br>(6,3) | $1.61642\,10^{-16}$<br>(6,3) | 0.<br>(6,1) | 0.<br>3 | 0<br>(2,1) |
| King 3 | $1 := 0.$<br>(3,1,1) | 0.<br>(6,3) | $2.22045\,10^{-16}$<br>(6,3) | $2.49183\,10^{-15}$<br>(6,1) | 0.<br>3 | *NA*<br>(2,1) |
| King 4 | $1 := 2.25\,10^{-15}$<br>(9,1,1) | $3.99199\,10^{-15}$<br>(18,9) | $1.48492\,10^{-15}$<br>(18,9) | $2.0552\,10^{-15}$<br>(18,5) | $6.38338\,10^{-16}$<br>9 | $2.63969\,10^{-9}$<br>(3,6) |
| Sims 0 | $1 := 0.$<br>(5,1,1) | $8.84516\,10^{-17}$<br>(10,5) | $3.57788\,10^{-16}$<br>(10,5) | $2.34864\,10^{-16}$<br>(10,3) | *NA*<br>5 | $5.55112\,10^{-17}$<br>(4,1) |
| Sims 1 | $1 := 1.18603\,10^{-15}$<br>(8,1,1) | $7.58081\,10^{-16}$<br>(16,8) | $9.08685\,10^{-16}$<br>(16,8) | $1.28298\,10^{-15}$<br>(16,6) | *NA*<br>8 | $8.11729\,10^{-16}$<br>(6,2) |
| Klein 1 | $1 := 1.29398\,10^{-14}$<br>(3,1,1) | $3.2259\,10^{-14}$<br>(6,3) | $8.72119\,10^{-14}$<br>(6,3) | $1.76957\,10^{-13}$<br>(6,1) | $4.54742\,10^{-10}$<br>3 | $4.61522\,10^{-13}$<br>(1,2) |
| BP 1 | $1 := 1.54607\,10^{-15}$<br>(2,1,1) | $2.82325\,10^{-15}$<br>(4,2) | $5.10874\,10^{-15}$<br>(4,2) | $1.23685\,10^{-14}$<br>(4,0) | $5.95443\,10^{-10}$<br>2 | $1.06208\,10^{-14}$<br>(1,1) |
| BP 2 | $1 := 5.00765\,10^{-16}$<br>(5,1,1) | $5.15101\,10^{-15}$<br>(10,5) | $5.52053\,10^{-15}$<br>(10,5) | $5.21764\,10^{-15}$<br>(10,1) | $7.14801\,10^{-16}$<br>5 | $3.98148\,10^{-14}$<br>(4,1) |

For $L, \tau, \theta, B, \vartheta, \phi, F$ seeAppendix A.1. For $m, p$, seeAppendix A.2. For $L, M_2$, seeAppendix A.3. For $L, n_k$, seeAppendix A.4. For $L$, seeAppendix A.5. For $m, n$, seeAppendix A.6. KW, Sims, Klein, BP, Uhlig column not normalized by dividing by comparable AIM value

**Table 3** Matlab relative errors in $\vartheta$ $\frac{\|\vartheta_i - \vartheta_{exact}\|}{\|\vartheta_{exact}\|}$

| Model | AIM $(L, \tau, \theta)$ | KW $(m, p)$ | Sims $(L, M_2)$ | Klein $(L, n_k)$ | BP $L$ | Uhlig $(m, n)$ |
|---|---|---|---|---|---|---|
| Computes | $(B, \vartheta, \phi, F)$ | $(B, \vartheta)$ | $(B, \phi, F)$ | $(B)$ | $(B, \vartheta, \phi, F)$ | $(B, \vartheta)$ |
| Uhlig 0 | $1 := 9.66331\,10^{-16}$ | 0.532995 | *NA* | 1.34518 | 159728. | 9.54822 |
| | (4,1,1) | (8,4) | (8,4) | (8,3) | 4 | (3,1) |
| Uhlig 1 | $1 := 2.25938\,10^{-15}$ | 3.78248 | *NA* | 2.32241 | $5.30459\,10^{6}$ | 2.33909 |
| | (6,1,1) | (12,6) | (12,6) | (12,5) | 6 | (5,1) |
| Uhlig 2 | $1 := 5.05614\,10^{-15}$ | 0.43572 | *NA* | 2.62987 | $1.51637\,10^{6}$ | 1. |
| | (6,1,1) | (12,6) | (12,6) | (12,5) | 6 | (5,1) |
| Uhlig 3 | $1 := 8.93\,10^{-16}$ | 2.08127 | *NA* | 1.39137 | *NA* | 14.3934 |
| | (14,1,1) | (28,14) | (28,14) | (28,13) | 14 | (10,4) |
| Uhlig 4 | $1 := 0.0114423$ | 1. | *NA* | 1. | 0.999999 | 1. |
| | (6,1,1) | (12,6) | (12,6) | (12,5) | 6 | (5,1) |
| Uhlig 5 | $1 := 1.37388\,10^{-15}$ | 3.79327 | *NA* | 9.97923 | $8.03937\,10^{6}$ | 12.2245 |
| | (6,1,1) | (12,6) | (12,6) | (12,5) | 6 | (5,1) |
| Uhlig 6 | $1 := 6.97247\,10^{-14}$ | 1.00929 | *NA* | 1.83538 | $1.12899\,10^{13}$ | 27.5959 |
| | (6,1,1) | (12,6) | (12,6) | (12,4) | 6 | (4,2) |
| Uhlig 7 | $1 := 7.47708\,10^{-14}$ | 1.67717 | *NA* | 0.131889 | *NA* | 0.665332 |
| | (13,1,1) | (26,13) | (26,13) | (26,11) | 13 | (11,2) |
| King 2 | $1 := 4.996\,10^{-16}$ | $4.996\,10^{-16}$ | *NA* | $4.996\,10^{-16}$ | $4.996\,10^{-16}$ | $4.996\,10^{-16}$ |
| | (3,1,1) | (6,3) | (6,3) | (6,1) | 3 | (2,1) |
| King 3 | $1 := 2.58297\,10^{-15}$ | $5.02999\,10^{-15}$ | *NA* | $6.42343\,10^{-15}$ | $2.58297\,10^{-15}$ | *NA* |
| | (3,1,1) | (6,3) | (6,3) | (6,1) | 3 | (2,1) |
| King 4 | $1 := 3.45688\,10^{-16}$ | $1.12445\,10^{-15}$ | *NA* | $8.40112\,10^{-16}$ | $2.65811\,10^{-16}$ | $2.47355\,10^{-9}$ |
| | (9,1,1) | (18,9) | (18,9) | (18,5) | 9 | (3,6) |
| Sims 0 | $1 := 7.66638\,10^{-16}$ | $7.85337\,10^{-16}$ | *NA* | $9.53623\,10^{-16}$ | *NA* | $7.66638\,10^{-16}$ |
| | (5,1,1) | (10,5) | (10,5) | (10,3) | 5 | (4,1) |
| Sims 1 | $1 := 9.73294\,10^{-16}$ | $1.57671\,10^{-15}$ | *NA* | $4.23589\,10^{-15}$ | *NA* | $9.73294\,10^{-16}$ |
| | (8,1,1) | (16,8) | (16,8) | (16,6) | 8 | (6,2) |
| Klein 1 | $1 := 2.33779\,10^{-15}$ | $1.9107\,10^{-14}$ | *NA* | $1.07269\,10^{-13}$ | $2.3967\,10^{-9}$ | $2.66823\,10^{-13}$ |
| | (3,1,1) | (6,3) | (6,3) | (6,1) | 3 | (1,2) |
| BP 1 | $1 := 1.4802\,10^{-15}$ | $8.91039\,10^{-15}$ | *NA* | $1.20169\,10^{-14}$ | $5.55737\,10^{-9}$ | $1.16858\,10^{-14}$ |
| | (2,1,1) | (4,2) | (4,2) | (4,0) | 2 | (1,1) |
| BP 2 | $1 := 6.1809\,10^{-16}$ | $1.25347\,10^{-15}$ | *NA* | $1.3268\,10^{-15}$ | $6.1809\,10^{-16}$ | $8.9026\,10^{-15}$ |
| | (5,1,1) | (10,5) | (10,5) | (10,1) | 5 | (4,1) |

For $L, \tau, \theta, B, \vartheta, \phi, F$ seeAppendix A.1. For $m, p$, seeAppendix A.2. For $L, M_2$, seeAppendix A.3. For $L, n_k$, seeAppendix A.4. For $L$, seeAppendix A.5. For $m, n$, seeAppendix A.6. KW, Sims, Klein, BP, Uhlig column not normalized by dividing by comparable AIM value

**Table 4** Matlab relative errors in $F$ $\frac{\|F_i - F_{exact}\|}{\|F_{exact}\|}$

| Model | AIM $(L, \tau, \theta)$ $(L, \tau, \theta)$ | KW $(m, p)$ $(m, p)$ | Sims $(L, M_2)$ $(L, M_2)$ | Klein $(L, n_k)$ $(L, n_k)$ | BP $L$ $L$ | Uhlig $(m, n)$ $(m, n)$ |
|---|---|---|---|---|---|---|
| Computes | $(B, \vartheta, \phi, F)$ | $(B, \vartheta)$ | $(B, \phi, F)$ | $(B)$ | $(B, \vartheta, \phi, F)$ | $(B, \vartheta)$ |
| Uhlig 0 | $1 := 4.60411\,10^{-16}$ | NA | 38.383 | NA | 6280.03 | NA |
| | (4,1,1) | (8,4) | (8,4) | (8,3) | 4 | (3,1) |
| Uhlig 1 | $1 := 6.12622\,10^{-16}$ | NA | 4.25457 | NA | 3126.17 | NA |
| | (6,1,1) | (12,6) | (12,6) | (12,5) | 6 | (5,1) |
| Uhlig 2 | $1 := 6.13246\,10^{-16}$ | NA | 3.58093 | NA | 3918.23 | NA |
| | (6,1,1) | (12,6) | (12,6) | (12,5) | 6 | (5,1) |
| Uhlig 3 | $1 := 7.28843\,10^{-16}$ | NA | 12.9392 | NA | $1.40986\,10^{15}$ | NA |
| | (14,1,1) | (28,14) | (28,14) | (28,13) | 14 | (10,4) |
| Uhlig 4 | $1 := 6.03573\,10^{-16}$ | NA | 2.73637 | NA | 1028.17 | NA |
| | (6,1,1) | (12,6) | (12,6) | (12,5) | 6 | (5,1) |
| Uhlig 5 | $1 := 6.03573\,10^{-16}$ | NA | 2.73637 | NA | 1028.17 | NA |
| | (6,1,1) | (12,6) | (12,6) | (12,5) | 6 | (5,1) |

**Table 4** continued

| Model | AIM $(L, \tau, \theta)$ $(L, \tau, \theta)$ | KW $(m, p)$ $(m, p)$ | Sims $(L, M_2)$ $(L, M_2)$ | Klein $(L, n_k)$ $(L, n_k)$ | BP $L$ $L$ | Uhlig $(m, n)$ $(m, n)$ |
|---|---|---|---|---|---|---|
| Computes | $(B, \vartheta, \phi, F)$ | $(B, \vartheta)$ | $(B, \phi, F)$ | $(B)$ | $(B, \vartheta, \phi, F)$ | $(B, \vartheta)$ |
| Model | AIM $(L, \tau, \theta)$ | KW $(m, p)$ | Sims $(L, M_2)$ | Klein $(L, n_k)$ | BP $L$ | Uhlig $(m, n)$ |
| Uhlig 6 | $1 := 6.0499\,10^{-16}$ | NA | 308.153 | NA | $1.65292\,10^{13}$ | NA |
| | (6,1,1) | (12,6) | (12,6) | (12,4) | 6 | (4,2) |
| Uhlig 7 | $1 := 7.52423\,10^{-14}$ | NA | 1.72491 | NA | NA | NA |
| | (13,1,1) | (26,13) | (26,13) | (26,11) | 13 | (11,2) |
| King 2 | $1 := 7.49401\,10^{-16}$ | NA | $1.36349\,10^{-15}$ | NA | $7.49401\,10^{-16}$ | NA |
| | (3,1,1) | (6,3) | (6,3) | (6,1) | 3 | (2,1) |
| King 3 | $1 := 3.9968\,10^{-16}$ | NA | $6.53472\,10^{-15}$ | NA | $3.9968\,10^{-16}$ | NA |
| | (3,1,1) | (6,3) | (6,3) | (6,1) | 3 | (2,1) |
| King 4 | $1 := 2.13883\,10^{-15}$ | NA | $2.58686\,10^{-15}$ | NA | $7.29456\,10^{-16}$ | NA |
| | (9,1,1) | (18,9) | (18,9) | (18,5) | 9 | (3,6) |
| Sims 0 | $1 := 7.13715\,10^{-16}$ | NA | $1.1917\,10^{-15}$ | NA | NA | NA |
| | (5,1,1) | (10,5) | (10,5) | (10,3) | 5 | (4,1) |
| Sims 1 | $1 := 2.41651\,10^{-15}$ | NA | $2.92152\,10^{-15}$ | NA | NA | NA |
| | (8,1,1) | (16,8) | (16,8) | (16,6) | 8 | (6,2) |
| Klein 1 | $1 := 1.24387\,10^{-15}$ | NA | $1.66911\,10^{-13}$ | NA | $3.70873\,10^{-10}$ | NA |
| | (3,1,1) | (6,3) | (6,3) | (6,1) | 3 | (1,2) |
| BP 1 | $1 := 4.43937\,10^{-16}$ | NA | $7.51277\,10^{-15}$ | NA | $5.03025\,10^{-11}$ | NA |
| | (2,1,1) | (4,2) | (4,2) | (4,0) | 2 | (1,1) |
| BP 2 | $1 := 5.82645\,10^{-16}$ | NA | $1.71285\,10^{-15}$ | NA | $5.82645\,10^{-16}$ | NA |
| | (5,1,1) | (10,5) | (10,5) | (10,1) | 5 | (4,1) |

For $L, \tau, \theta, B, \vartheta, \phi, F$ seeAppendix A.1. For $m, p$, seeAppendix A.2. For $L, M_2$, seeAppendix A.3. For $L, n_k$, seeAppendix A.4. For $L$, seeAppendix A.5. For $m, n$, seeAppendix A.6. KW, Sims, Klein, BP, Uhlig column not normalized by dividing by comparable AIM value. For $m, n$, seeAppendix A.6

## References


Anderson, Gary, & Moore, George (1985). A linear algebraic procedure for solving linear perfect foresight models. *Economics Letters, 17*(3), 247–252.

Binder, Michael, & Pesaran, M. Hashem (1994). *Multivariate rational expectations models and macroeconometric modelling: A review and some new results*. Seminar Paper, May 1994.

Blanchard, Olivier Jean, & Kahn, C. (1980). The solution of linear difference models under rational expectations. *Econometrica, 48*.

Broze, Laurence, Gouriéroux, Christian, & Szafarz, Ariane (1995). Solutions of multivariate rational expectations models. *Econometric Theory, 11*, 229–257.

Gantmacher, F. R. (1959). *Theory of matrices*. Chelsea Publishing.

Golub, Gene H., & van Loan, Charles F. (1989). *Matrix computations*. Johns Hopkins.

King, Robert G., & Watson, Mark W. (1998). The solution of singular linear difference systems under rational expectations. *International Economic Review, 39*(4), 1015–1026.

Klein, Paul (1999). Using the generalized schur form to solve a multivariate linear rational expectations model. *Journal of Economic Dynamics and Control, 24*, 1405–1423.

Sims, Christopher A. (1996). *Solving linear rational expectations models*. Seminar paper.

Uhlig, Harald (1999). *A toolkit for analyzing nonlinear dynamic stochastic models easily*. User's Guide.

Zadrozny, Peter A. (1998). An eigenvalue method of undetermined coefficients for solving linear rational expectations models. *Journal of Economic Dynamics and Control, 22*, 1353–1373.