A time-varying parameter VAR à la Primiceri (2005)

Benedikt Kolb*

21st October 2015

Abstract

This note outlines the use of a time-varying parameter structural VAR as in Primiceri (2005), corrected as in Del Negro and Primiceri (2015). Following my own appropriation of the time-varying parameter vector autoregression (TVP-VAR) techniques, but contrary to usual outlines, I start by "walking through the code" and give the theoretical foundations of the procedures only after that. While the procedure might be outlined with more expertise in Primiceri (2005) and Del Negro and Primiceri (2015), this note tries to focus on practical implementation issues and aims at economists dealing with TVP-VAR for the first time.

1 Motivation

To me, working with (good) code is the best way to learn a new empirical technique. So when I started a project with a time-varying parameter VAR, I first looked at different codes related to that subject.[1] While a thorough understanding of the underlying econometric theory is important, this understanding is often facilitated by knowing which regressions are actually run on what data and which assumptions the algorithm requires. I understand that presenting theory first and application later seems more appropriate, however, this note is addressed rather on researchers who want to start with an application. This note follows closely the code that comes with it[2], explaining the underlying theory only after detailing the steps used in the application. For anyone who prefers the standard method (theory, then application), I recommend the beautiful and publicly available[3] outline in Blake and Mumtaz (2012), which treats time-varying parameters, but stops short of stochastic volatility. The papers by Primiceri (2005) and Del Negro and Primiceri (2015) – P05 and DNP in the following – plus the respective appendices provide details on the estimation

*For questions, write me under benedikt@bkolb.eu

1. These were codes made available by Marco Del Negro and Giorgio Primiceri, by Haroon Mumtaz, and by Dimitris Korobilis and Gary Koopman. My codes also rely to these to a large extent; the codes (in the order mentioned here) can be found at
   http://restud.oxfordjournals.org/content/suppl/2015/06/22/rdv024.DC1/Supplementary.zip
   http://www.bankofengland.co.uk/education/Documents/ccbs/technical_handbooks/Coding/code.zip
   https://sites.google.com/site/dimitriskorobilis/matlab/code-for-vars/matlab-bvar-code/TVP_VAR_CK.zip
2. Both available under www.bkolb.eu/codes/TVPVAR.zip
3. See http://www.bankofengland.co.uk/education/Pages/ccbs/technical_handbooks/techbook4.aspx

with stochastic volatility, although the outline is more dense there. If you deem even these notes here for too long and want to look directly at the code, Appendix 7.2 will connect my code and outline below.
This note closely follows P05, however I treat the elements of the covariance matrix of innovations by themselves, not in a joint covariance matrix (V) as Primiceri does. While his outline is the more general one, I believe that separating its elements makes the outline somewhat simpler (albeit longer) for beginners. Moreover, the independence assumptions in P05 which make my distinction of the V elements possible will hold in most applications of this model.
The outline of the reminder of this note is as follows: Section 2 outlines the model we want to estimate, i.e. the one in P05. As I stick to the notation there, one could also refer to the paper directly. The following sections are more specific to the code: Section 3 explains how to initialise the TVP-VAR using a training sample as in P05 and how to choose the priors. Section 4 will walk you through the Gibbs sampler which constitutes the core of the estimation. As stated above, I focus on the application and mention the theory behind it only after that. Section 5 explains in more detail the Carter-Kohn algorithm (Carter and Kohn 1994) used heavily in the Gibbs sampler. The Appendix provides additional points of interest, which are however not central to the estimation itself.

2 The model we want to estimate

Following P05, we want to estimate a three-variable VAR containing quarterly series for inflation, unemployment and short-term nominal interest rates. We are particularly interested in the time variation of three sets of coefficients: The VAR coefficients B_t, the stochastic volatility of the orthogonalised VAR residuals Σ_t, as well as the variation in the contemporaneous relationships between these orthogonal residuals (i.e., the time-varying Cholesky matrices guaranteeing identification of the structural VAR), A_t. We assume the VAR to take the following form (as Primiceri, I write the model generically for n_l lags, although I will only consider two in the application):

$$y_t = c_t + B_{1,t}y_{t-1} + \cdots + B_{nl,t}y_{t-nl} + u_t, \qquad \text{where } u_t \sim N(0, \Omega_t) \tag{1}$$

For simplicity, let us define the vector of regressors $x_t \equiv (1, y_{t-1}, \ldots, y_{t-nl})$ and coefficients $B_t \equiv (c_t, B_{1,t}, \ldots, B_{nl,t})$. Moreover, use a triangular reduction of the VAR residual covariance matrix Ω_t, to move from reduced-form residuals u_t to structural shocks ε_t. The triangular reduction is established as $A_t\Omega_t A_t' = \Sigma_t\Sigma_t'$, where A_t and Σ_t are lower triangular and diagonal matrices,

respectively:

$$A_t = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ \alpha_{21,t} & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ \alpha_{n_l 1,t} & \cdots & \alpha_{n_l n_l-1,t} & 1 \end{bmatrix}$$

$$\Sigma_t = \begin{bmatrix} \sigma_{1,t} & 0 & \cdots & 0 \\ 0 & \sigma_{2,t} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \sigma_{nl,t} \end{bmatrix}$$

Using these transformation, we can rewrite the VAR as

$$\underset{(n_y \times 1)}{y_t} = \underset{(n_y \times n_r)}{x_t} \cdot \underset{(n_r \times 1)}{B_t} + \underset{(n_y \times n_A)}{A_t^{-1}} \underset{(n_y \times n_y)}{\Sigma_t} \underset{(n_y \times 1)}{\varepsilon_t}, \text{ where } \varepsilon_t \sim N(0, I_{ny}) \tag{2}$$

and where n_r is the number of regressors (here seven, one constant and two lags for three variables) times the number of variables n_y, so $n_B = 21$ in our case. We need to make some assumptions on the distributions of B_t, A_t and Σ_t now. For simplicity and to save on hyperparameters[4], we assume B_t and A_t to follow a random walk as in P05. We would like to do the same for Σ_t, but as its elements enter the model multiplicatively, we have to square and take logs (see P05, p. 846) and define[5]

$$H_t \equiv \begin{bmatrix} \log(\sigma_{1,t}^2) & 0 & \cdots & 0 \\ 0 & \log(\sigma_{2,t}^2) & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \log(\sigma_{n_l,t}^2) \end{bmatrix}$$

Moreover, the covariance between inflation and unemployment,$A_{2,t} \equiv \alpha_{21,t}$, is assumed to be independent of the covariance of both inflation and unemployment with the short-term interest rate, i.e. $A_{2,t} \perp A_{3,t} \equiv (\alpha_{31,t}, \alpha_{32,t})$, as the latter are in the realm of monetary policy, while the former is not. This gives the following set of equations:[6]

$$B_t = B_{t-1} + \upsilon_t, \qquad \text{where } \upsilon_t \sim N(0, Q) \tag{3}$$

$$A_{2,t} = A_{2,t-1} + \zeta_{1,t}, \qquad \text{where } \zeta_{1,t} \sim N(0, S_2) \tag{4}$$

$$A_{3,t} = A_{3,t-1} + \zeta_{3,t}, \qquad \text{where } \zeta_{3,t} \sim N(0, S_3) \tag{5}$$

$$H_t = H_{t-1} + \eta_t, \qquad \text{where } \eta_t \sim N(0, W) \tag{6}$$

4. Hyperparameters are parameters needed for estimation, but in which the researcher has no economic interest. Common examples are covariances of coefficients to be estimated or their autoregressive parameters (which are trivially fixed to unity for random walks).

5. As H_t follows a random walk, the stochastic volatilities σ_t follow a geometric random walk.

6. Of course (6) could be rewritten as $\log \sigma_t = \log \sigma_{t-1} + \eta_t$.

Note that here I have already incorporated the assumptions of block exogeneity between B_t, $A_{2,t}$, $A_{3,t}$ and H_t. Explicitly, we could write this – as Primiceri does – in terms of an overall covariance matrix V for the coefficients:

$$V = \text{Var}\left(\begin{bmatrix} \varepsilon_t \\ \upsilon_t \\ \zeta_{2,t} \\ \zeta_{3,t} \\ \eta_t \end{bmatrix}\right) = \begin{bmatrix} I_n & 0 & 0 & 0 & 0 \\ 0 & Q & 0 & 0 & 0 \\ 0 & 0 & S_2 & 0 & 0 \\ 0 & 0 & 0 & S_3 & 0 \\ 0 & 0 & 0 & 0 & W \end{bmatrix} \tag{7}$$

Keep in mind that the assumptions of random walk and block exogeneity are not essential to the analysis, but make it simpler. See P05, p. 824f for (good) arguments why these assumptions are sensible. Finally, note that the model we want to estimate is hence in state-space form given by an observation equation, (2), and several transition equations, (3) to (6).

Now there is one additional complication: Remember that we defined $H_t = \log(\Sigma_t \Sigma_t')$ to get a linear system of equations. However, this also transforms the structural VAR error ε_t into $\log(\varepsilon_t^2)$, which no longer is normally distributed, but follows a $\log \chi^2(1)$ distribution (see DNP, p. 2). There is a method developed by Kim, Shephard, and Chib (1998), however, to approximate each elements of H_t by a mixture of normal distributions, getting us back to Gaussian errors. However, the mixture procedure entails a cost: We have to keep track of the indicator variables s_t specifying which of the normal approximations in the mixture to use for each element of H_t. These indicator are thus added to our coefficients in the Gibbs sampler (more on the mixture of normals in Appendix 7.3 below).[7]

To conclude, we are interested in the following estimates: The coefficients B_t, a $(T \times n_B \times n_r)$ matrix, A_t, a $(T \times n_A \times n_r)$ matrix and H_t, a $(T \times n_y \times n_r)$ matrix, as well as the hyperparameters Q, an $(n_B \times n_B \times n_r)$ matrix, S_2, a $(1 \times 1 \times n_r)$ matrix, S_3, a $(2 \times 2 \times n_r)$ matrix, and W, a $(n_y \times n_y \times n_r)$ matrix.[8] We use the sample from P05 for this, giving quarterly data on inflation and unemployment rate as well as on short-term nominal interest rate, from 1953Q1 to 2001Q3.

I provide code for different specifications, which will yield however the same results (differing only due to the stochastic draws in the Gibbs sampler): My routine `run_TVPVAR1.m` replicates the code made available by Marco Del Negro and Giorgio Primiceri, but in different use of matrix algebra and notation (originally, it was based on code by Haroon Mumtaz and Dimitris Korobilis). If you set a seed to control Matlab's random number generation process (simply comment in the lines on top of the file), you can compare that the results are *exactly* identical to Del Negro and Primiceri's routine `main.m`, down to having the same matrices for any number of iterations.[9]

Second, the routine `run_TVPVAR2.m` is my preferred one; it is similar to `run_TVPVAR1.m`,

7. The error in P05 corrected by DNP lay in the implementation of this mixture of normals into the Gibbs sampler. Note that even the corrected version of the Gibbs sampler yields only "approximate posteriors" (DNP, p. 4), because the mixture of normals distribution is only an approximation to the $\log\chi^2(1)$ distribution. However, DNP show that results are virtually indistiguishable from a fully correct solution, which relies on (computationally costly) sampling from a Metropolis-Hastings algorithm.

8. I will stick to this differentiation of "coefficients" and "hyperparameters" in all below. Also note that DNP use a matrix $V = \text{diag}(Q, S_2, S_3, W)$ as in (7) in their code for all hyperparameters.

9. Just download their codes from http://restud.oxfordjournals.org/content/suppl/2015/06/22/rdv024.DC1/Supplementary.zip and copy the line setting the seed to the top of their `main.m` file.

but changes the order of the Gibbs sampler so that coefficient covariances (hyperparameters) are estimated directly after the respective coefficients (so Q after B_t, S_1 and S_{23} after A_t, W after H_t). I think this is more straightforward than using a big matrix V, but it of course hard-wires the block exogeneity assumptions into the code. Finally, the routine `run_TVPVAR3.m` also differentiates between the estimation of $A_{2,t} = \alpha_{21,t}$ and $A_{3,t} = (\alpha_{31,t}, \alpha_{32,t})$. While this clarifies further which coefficients are assumed to be independent, it also introduces some redundancies in writing out matrices. Overall, I recommend using `run_TVPVAR2.m` for studying with this note.

3 Initialising the Gibbs sampler

Before starting the actual Gibbs sampler, we will have to do two things: Specify the priors (i.e., their distribution and parameters) and initialise (i.e. take first draws for) all coefficients/hyperparameters except one (here, B_t). Then we can start the Gibbs sampler with drawing B_t given A_t, H_t, Q, S_2, S_3 and W, and follow its algorithm explained in Section 4. Priors are outlined for the different blocks of coefficients; given priors, initialisation is easy to achieve.

3.1 Priors

The distributions B_t, A_t and s_t specified as normals, H_t has a $\log \chi^2(1)$ distribution, and Q, S_2, S_3 and W are independently inverse Wishart distributed (see Appendix 7.1 on page 9 for some details on the inverse Wishart distribution). Thus we need to specify mean and variance for the normal distributions of B_t, A_t and s_t, as well as scale parameter and degrees of freedom for Q, S_2, S_3 and W. The priors are calibrated using an OLS estimation on a training sample, i.e. an initial subset of the data. P05 uses the first ten years of his sample (1953Q1 to 1962Q4) to do so. In the following, the prior setup is explained in more detail. I will denote any draw from the prior with a subscript pr, as in $B_{pr,t}$ – note that this is just hypothetical for now, actual draws will only be taken during initialisation (and for B_t and s_t only during the Gibbs sampling below).

Priors for B_t and Q. Denoting training sample variables by a zero subscript, an OLS regression of $y_{0,t}$ on $x_{0,t}$ for these 40 quarters yields a (time invariant) estimate, $\hat{B}_0$. Moreover, we obtain a measure of $\text{cov}(\hat{B}_0)$.[10] These two we use as mean and covariance for our priors: $B_{pr} \sim N(\hat{B}_0, \text{cov}(\hat{B}_0))$. Next, the degrees of freedom of Q are $T - 1$, as one degree of freedom is consumed in calculating the errors of B_t.[11] The scale parameter is equal to the squared differences in B_t over time, plus a constant from the training sample, $T_Q \cdot \text{cov}(\hat{B}_0) \cdot k_Q^2$. Here, T_Q is the size of the training sample (40) and k_Q is a constant (somewhat arbitrarily) set to 0.01 by Primiceri. Thus, $Q_{pr} \sim IW(T_Q \cdot \text{cov}(\hat{B}_0) \cdot k_Q^2, T_Q)$.

Priors for A_t, S_2 and S_3. To estimate the degree of covariation between residuals, we regress the residuals of the above OLS estimation, $\hat{u}_{0,t} = y_{0,t} - \hat{B}_0 x_{0,t}$ on each other to obtain a measure

10. The standard formula is $\text{cov}(\hat{B}_0) = (\hat{u}'_{0,t}\hat{u}_{0,t}/T_0) \otimes (x'_{0,t}x_{0,t})^{-1}$

11. In the code, T becomes readjusted after the training sample, so we have to re-add $TQ = 40$ when drawing from the prior.

of the respective $\hat{\alpha}_{i,t}$: Call $\hat{u}_{i,t}$ the residual of equation i in y_t. Then

$$\begin{aligned}\hat{u}_{2,t} &= \hat{u}_{1,t}\hat{\alpha}_{21}\\ \hat{u}_{3,t} &= \hat{u}_{1,t}\hat{\alpha}_{31}\\ \hat{u}_{3,t} &= \hat{u}_{2,t}\hat{\alpha}_{32}\end{aligned}$$

By these regressions, we obtain a measure of $\hat{A}_0 = [\hat{\alpha}_{21}, \hat{\alpha}_{31}, \hat{\alpha}_{32}]$. Then it is easy to also obtain a measure of $e_{A,t} = \hat{u}_t - \hat{x}_{u,t}\hat{A}_0$, where $\hat{x}_{u,t}$ is a collection of the respective regressors ($\hat{u}_{1,t}$ for $\hat{\alpha}_{21}$ etc.).[12] From this in turn we obtain $\text{cov}(\hat{A}_0)$.[13] Thus, $A_{pr} \sim N(\hat{A}_0, \text{cov}(\hat{A}_0))$. The degrees of freedom of S_2 and S_3 are set to 2 and 3, respectively, i.e. one plus their dimension (remember, S_2 is a scalar and S_3 a (2×2) matrix). Their scales are set to $T_{Si}\cdot\text{cov}(\hat{A}_0)\cdot k_S^2$, $i = 2, 3$, where $T_{S2} = 2$ and $T_{S3} = 3$, and where k_S is set to 0.1 as in P05. Thus, $S_{2,pr} \sim IW(T_{S2}\cdot\text{cov}(\hat{A}_0)\cdot k_S^2, T_{S2})$ and $S_{3,pr} \sim IW(T_{S3}\cdot\text{cov}(\hat{A}_0)\cdot k_S^2, T_{S3})$.

Priors for H_t and W. The diagonal elements of the covariance matrix, H_t, are initialised as the squared residuals from the $\hat{A}_0$ regressions – put in diagonal form, and logged as well as multiplied by 0.5: $H_t = 0.5\cdot\log(\text{diag}(e'_{A,t}e_{A,t}/T_0))$. Finally, for W the degrees of freedom are set to $T_W = 4$, again one plus its dimension, and its scale is $T_W\cdot\text{cov}(\hat{A}_0)\cdot k_W^2$, where $k_W = 0.01$. This implies $W_{pr} \sim IW(T_W\cdot\text{cov}(H)_0\cdot k_W^2, T_W)$.
Note that we do not require priors for s_t, which are simply indicators. The parameters of the underlying normals (which are mixed to approximate the $\log\chi^2(1)$ distribution of H_t) are taken from Kim, Shephard, and Chib (1998).

3.2 Initialisation

All coefficients and hyperparameters except B_t and s_t are initialised by a draw from their respective distributions. The Gibbs sampler draws recursively, so we can leave one parameter without initialisation (B_t). s_t has a particular role; as it serves as an auxiliary coefficient only, it does not need to be initialised.

4 The Gibbs sampler

In the Gibbs sampler or Gibbs sampling algorithm (GSA), one draws from each "block" of parameters conditional on all the others.[14]
Here the GSA is given by[15]

12. The regressions are run in the companion form anyway.

13. The formula here is $\text{cov}(\hat{A}_0) = (\hat{x}'_{u,t}\hat{x}_{u,t})^{-1}\hat{x}_{u,t}\cdot(\hat{e}'_{A,t}\hat{e}_{A,t}/T_0 \otimes I_{T_0})\cdot\hat{x}_{u,t}(\hat{x}'_{u,t}\hat{x}_{u,t})^{-1}$.

14. This was the problem with the original outline in Primiceri (2005), where the draws of B_t, A_t, and Q, S_2, S_3 and W were not conditional on the mixture indicators s^T, see Del Negro and Primiceri (2015), p. 2f.

15. Note that Primiceri (2005) draws Q, W and S in a last step of the algorithm (Step 5.5: Sample $V = \text{diag}(I_{n_y}, Q, S, Q)$ from $p(V|y^T, B^T, A^T, \Sigma^T)$), p. 846f. As the draws of hyperparameters Q, S_2, S_3 and W only depend on the priors and respectively B_t, $A_{2,t}$, $A_{3,t}$ and H_t, the reordering does not affect results.

Gibbs Sampling Algorithm

1. Initialise A^T, Σ^T, s^T as well as Q, S_2, S_3 and W.
2. Sample B^T from $p(B^T|y^T, A^T, \Sigma^T, Q, S_2, S_3, W)$ and Q from $p(Q|y^T, B^T, A^T, \Sigma^T)$.
3. Sample A^T from $p(A^T|y^T, B^T, \Sigma^T, Q, S_2, S_3, W)$, as well as S_2 from $p(S_2|y^T, B^T, A^T, \Sigma^T)$ and S_3 from $p(S_3|y^T, B^T, A^T, \Sigma^T)$.
4. Sample s^T from $p(s^T|y^T, B^T, A^T, \Sigma^T, Q, S_2, S_3, W)$.[16]
5. Sample Σ^T from $p(\Sigma^T|y^T, B^T, A^T, Q, S_2, S_3, W)$ and W from $p(W|y^T, B^T, A^T, \Sigma^T)$.
6. Return to 2.

Drawing B^T, A^T and H^T requires the Carter-Kohn algorithm. This algorithm first uses the Kalman filter to get optimal estimates of the coefficients, and then uses a backward algorithm to compute stochastic realisations of the coefficients from their specified distribution (taking into account prior distribution and the realisation of other coefficients at that iteration). This gives us one draw of B_t, $t = 1, \dots, T$. The algorithm is detailed in Section 4.
Drawing the hyperparameters Q, S_2, S_3 and W is simpler: We mix the information from the realisation of the respective coefficients (i.e. the variation in B_t for Q, of $A_{2,t}$ for S_2, etc.) with the prior information from the OLS sample, and draw from an inverse Wishart. "Mixing the information" is accomplished by adding the prior scale parameter (e.g. $T_Q \cdot \text{cov}(\hat{B}_{OLS}) \cdot k_Q^2$ for Q) with the squared differences in the coefficients ($(\Delta B_t)' \cdot \Delta B_t$ for Q), and taking the whole sample size minus one (for the differencing of B_t) as degrees of freedom.
Drawing from s^T is the most complicated matter here.

Why do we do this? The GSA represents a way of obtaining draws from an (unknown) joint distribution of parameters by iteratively drawing from conditional, known distributions – i.e. draw from the distribution of B_t, keeping all other coefficients/hyperparameters fixed, and then proceed to the next parameter of interest. Indeed, describing the joint distribution of all coefficients and hyperparameters would be very difficult. However, we can draw easily from normal and inverse Wishart distributions iteratively, and thereby get information on their posterior.
The GSA lends itself to the estimation of linear state-space models with non-Gaussian errors. Given our observation equation (2) and the transition equations (3) to (6), it is easy to verify that they are linear. Were the errors to the system Gaussian, could we just use a Kalman filter to obtain optimal estimates of the unobserved states. However, with stochastic volatility the errors are not Gaussian because of their interactions. We can at most assume they are mixtures of normals as in Carter and Kohn (1994).[17]

16. Note that the mistake in Primiceri (2005) which is corrected in Del Negro and Primiceri (2015) is that Step 4 came after Step 5 in the former paper (p. 847).

17. If the state-space model was additionally non-linear, we would have to resort to estimation techniques like the particle filter.

5 The Carter-Kohn algorithm

The centrepiece of the Gibbs sampler is the Carter-Kohn (CK) algorithm. It is used to obtain a draw from the parameter distribution $\tilde{\boldsymbol{\beta}}_T = [\beta_1, \ldots, \beta_T]$, using the Kalman filter and the CK recursion algorithm:

1. The Kalman filter gives an estimate of the mean and variance of the coefficients using all available information until then, $\beta_{t|t}$ and $\Omega_{t|t}$ for $t \in (1, \ldots, T)$.
2. Given these, we take a random draw of a (multivariate) normal with mean $\beta_{T|T}$ and covariance $\Omega_{T|T}$, call it $\tilde{\beta}_T$.
3. The CK algorithm allows us to obtain a history of coefficient estimates $\tilde{\boldsymbol{\beta}}_T$ and their covariance matrix

Why do we do this? The Kalman filter gives an efficient way of solving for the states of a state-space model.[18] The iterations yield $\beta_{t|t}$ and $\Omega_{t|t}$ for $t \in (1, \ldots, T)$.
CK use the fact that for a conditional distribution of the vector $\boldsymbol{\beta}_T = [\beta_1, \beta_2, \ldots, \beta_T]$ it holds that

$$\begin{aligned}
p(\boldsymbol{\beta}_T|y^T) &= p(\beta_T|y^T) \cdot p(\tilde{\beta}_{T-1}|\beta_T, y^T) \\
&= p(\beta_T|y^T) \cdot p(\beta_{T-1}|\beta_T, y^T) \cdot p(\beta_{T-2}|\beta_T, \beta_{T-1}, y^T) \cdot \cdots \cdot p(\beta_1|\beta^T, \beta^{T-1}, \ldots, \beta^2, y^T) \\
&= p(\beta_T|y^T) \cdot p(\beta_{T-1}|\beta_T, y^{T-1}) \cdot p(\beta_{T-2}|\beta_{T-1}, y^{T-2}) \cdot \cdots \cdot p(\beta_1|\beta^2, y^1) \\
&= p(\beta_T|y^T) \cdot \prod_{t=1}^{T-1} p(\beta_t|\beta_{t+1}, y^t)
\end{aligned}$$

where the penultimate step is due to the Markovian nature of β_T (as we specified it to be an $AR(1)$ process) - note the changes both in former β_{T-i} and data y^{T-i}! If we assume that the disturbances of both the observation equation (1) and the transition equations (3) to (6), e_t and v_t, are normally distributed, we have

$$\begin{aligned}
p(\beta_T|y^T) &\sim N(\beta_{T|T}, \Omega_{T|T}) \\
p(\beta_t|\beta_{t+1}, y^t) &\sim N(\beta_{t|t}|\beta_{t+1|t+1}, \Omega_{t|t}|\beta_{t+1|t+1})
\end{aligned}$$

The backward recursions draw $\beta_{CK,t}$ from a normal $N(\beta_{N,t}, P_{N,t})$ with

$$\begin{aligned}
\beta_{N,t} &= \beta_{t|t} + K(\beta_{CK,t} - F\beta_{t|t}) \\
P_{N,t} &= P_{t|t} - KFP_{t|t}
\end{aligned}$$

where $\beta_{t|t}$ and $P_{t|t}$ are the saved values from the Kalman filter and the initial value of $\beta_{N,T}$ and $P_{N,T}$ are the last saved values from the Kalman filter, $\beta_{T|T}$ and $P_{T|T}$. So in fact we have three values of "β" in every iteration t (from $T-1$ to 1):

18. For details, please refer to the notes in http://www.bkolb.eu/codes/kalman.zip.

1. the Kalman filtered value, $\beta_{t|t}$, saved from before and called in Carter-Kohn at iteration t (together with covariance matrix $P_{t|t}$),

2. the mean of the normal distribution, $\beta_{N,t} = \beta_{t|t}|\beta_{CK,t+1}$, from the above equation (plus the covariance matrix $P_{N,t}$)[19],

3. the draw from a Normal distribution $\beta_{CK,t} \sim N(\beta_{N,t}, P_{N,t})$, which is saved at iteration t.

6 Burn-in and thinning

The code uses a burn-in sample to reduce the dependence on potentially badly chosen initial values. Moreover, it uses "thinning" to reduce the serial correlation between the draws: For inference – here only plotting the path of stochastic volatilities for the three model equations – we only take every thin$^{\text{th}}$ draw (here, every 10^{th}).

7 Appendix

7.1 Random number generator for Inverse Wishart using normal draws

Since its 2006 version, Matlab allows to draw from an Inverse Wishart distribution using the command `iwishrnd`. However, there is a simple way to generate draws from an Inverse Wishart using normal draws: Say you want to draw $d \sim IW(sc, df)$ from an Inverse Wishart with an $(k \times k)$ scale matrix sc and degrees of freedom df. To do so using a normal draw, draw $z \sim N(k, df)$ and use

$$d = \left[\text{chol}(sc^{-1})' \cdot z \cdot z' \cdot \text{chol}(sc^{-1})\right]^{-1},$$

where

$$sc^{-1} = \text{chol}(sc^{-1})' \cdot \text{chol}(sc^{-1})$$

7.2 A walkthrough of my code (`run_TVPVAR2.m`)

This section outlines what my code does and connects it to the outline above.

Part A: Settings Make choices on estimation details like the number of lags, iterations and size of the burn-in period. Change the prior constants here if you like (they are set to the ones used in DNP). Also set the parameters to approximate the mixture of normals as in Kim, Shephard, and Chib (1998). Finally, choose plotting options like thinning of parameters (use only every i^{th} draw to reduce serial correlation) and horizons of the IRFs. For the prior choices, see See Subsection 3.1 on page 5.

Part B: Data Load the data (the same as in DNP) and define constants (here: one constant).

19. Again, the first iteration is different in that $\beta_{N,T}$ is initialised at $\beta_{T|T}$ (and $P_{N,T} = P_{T|T}$).

Part C: Priors Obtain prior from OLS regression on a training sample (default: 40 periods, as set in Part A). See Subsection 3.1 on page 5.

Part D: Initialisation Draw the first instances of A_t, H_t, Q, S_2, S_3 and W from the priors. By this, we can start the Gibbs sampler with a draw of B_t from $p(B_t|Y^T, A^T, H^T, Q, S_2, S_3, W)$. Also pre-allocate some matrices to collect the draws during estimation. See Subsection 3.2 on page 6.

Part E: Gibbs sampler The heart of the code. Given priors and initial draws for all coefficients and hyperparameters except B_t, we iteratively draw from the parameters, taking all others as given and starting with B_t. Note that the algorithm is the one corrected as in DNP. The routine calls the function `KalmanCarterKohn.m` for drawing the coefficients B_t, A_t and s_t. The function runs a Kalman filter on the coefficients and then takes draws using backward recursions as in Carter and Kohn (1994). First, the Kalman filter obtains an efficient estimate of unobserved states x_t of a state-space model of the form

$$x_t = Fx_t + G\epsilon_t$$
$$y_t = H_t x_{t-1} + J\xi_t$$

In the case of B_t, y_t is the VAR regressands Y^T, H_t are the VAR regressors X^T, and x_t is B_t, where $F = 1$ (or an appropriate identity matrix) due to the random walk assumption. Thus, the Kalman filter gives us the best possible fit of B_t for the VAR from $t = 1$ to $t = T$.[20]
Second, the Carter-Kohn backward recursion in some regards does the inverse of the Kalman filter: Starting with the last value B_T, it draws some random error from the prior distribution of B_t and generates a counter-factual, but not unlikely, incidence (draw) of B_{T-1}. It updates the respective forecast error matrix and continues to draw consistent counter-factuals for B_t from top (T) to the bottom ($t = 1$). This gives us a consistent draw for B_t, $t = 1, \ldots, T$, given the other coefficients and hyperparameters at that point, the priors, and our distributional assumptions (basically, Gaussian errors).
The Kalman-Carter/Kohn algorithm is also used for A_t and s_t, where regressands, regressors, initial values and measurement error matrix change accordingly.
Directly after using that algorithm for the coefficients, we can draw their respective covariance matrix from the inverse Wishart distribution. We save all these draws and start the next iteration, as outlined in the algorithm on page 7.

Part F: Plotting of results Here, I only plot the standard deviations of the different VAR equations, see Figure 1 in P05, and better (due to the correction) Figure 1 in the web appendix of DNP (comes with the supplementary data zip file).

20. "Best" only under the assumption of Gaussian errors, which we maintain.

7.3 Mixture-of-normals approximation

Kim, Shephard, and Chib (1998) – KSC in the following – focus on estimating a stochastic volatility model that can be brought into linear form as

$$\log(y_t^2) = h_t + \log(\varepsilon_t^2) \tag{8}$$

They find that even though a quasi-likelihood approach yields consistent and asymptotically normal estimators, still estimates will not be satisfying because in small samples, the normal distribution is a bad approximation for $\log(\varepsilon_t^2)$. So they propose instead to use several (specifically, seven) normally distributions that are conveniently parametrised to hit several moments of the $\log \chi^2(1)$ distribution that $\log(\varepsilon_t^2)$ actually follows. Additionally, they add an "offset constant" $\bar{c}$ to make the estimation robust to small values of y_t^2. Then the equation (8) takes the form

$$\log(y_t^2 + \bar{c}) = h_t + \sum_{i=1}^{7} q_i f_N(z_t | m_i - 1.2704, v_i^2),$$

where q_i, the means m_i and variances v_i^2 of the seven distributions are chosen to match moments of $\log(\varepsilon_t^2)$. In particular, we can also write the realisation of the normal draws, z_t, as dependent on and indicator variables s_t, which becomes our last coefficient to be estimated in the GSA:

$$\begin{aligned} z_t | s_t = i &\sim N(m_i - 1.2704, v_i^2) \\ p(s_t = i) &= q_i \end{aligned}$$

As the chosen values for q_i, m_i and v_i^2 are not specific to the model or application, P05 and we can freely use them for our approximation of a similar case of stochastic volatility estimation within our Gibbs sampler.

References

Blake, Andrew P, and Haroon Mumtaz. 2012. *Applied Bayesian econometrics for central bankers.* Technical Books 4. Centre for Central Banking Studies, Bank of England.

Carter, C. K., and R. Kohn. 1994. "On Gibbs Sampling for State Space Models" [in English]. *Biometrika* 81 (3): pages. ISSN: 00063444.

Del Negro, Marco, and Giorgio E. Primiceri. 2015. "Time Varying Structural Vector Autoregressions and Monetary Policy: A Corrigendum." *The Review of Economic Studies* 82 (4): 1342–1345.

Kim, Sangjoon, Neil Shephard, and Siddhartha Chib. 1998. "Stochastic Volatility: Likelihood Inference and Comparison with ARCH Models." *Review of Economic Studies* 65, no. 3 (July): 361–93.

Primiceri, Giorgio E. 2005. "Time Varying Structural Vector Autoregressions and Monetary Policy." *Review of Economic Studies* 72 (3): 821–852.