11

Vector Autoregressive Models for Multivariate Time Series

11.1 Introduction

The *vector autoregression* (VAR) *model* is one of the most successful, flexible, and easy to use models for the analysis of multivariate time series. It is a natural extension of the univariate autoregressive model to dynamic multivariate time series. The VAR model has proven to be especially useful for describing the dynamic behavior of economic and financial time series and for forecasting. It often provides superior forecasts to those from univariate time series models and elaborate theory-based simultaneous equations models. Forecasts from VAR models are quite flexible because they can be made conditional on the potential future paths of specified variables in the model.

In addition to data description and forecasting, the VAR model is also used for structural inference and policy analysis. In structural analysis, certain assumptions about the causal structure of the data under investigation are imposed, and the resulting causal impacts of unexpected shocks or innovations to specified variables on the variables in the model are summarized. These causal impacts are usually summarized with impulse response functions and forecast error variance decompositions.

This chapter focuses on the analysis of covariance stationary multivariate time series using VAR models. The following chapter describes the analysis of nonstationary multivariate time series using VAR models that incorporate cointegration relationships.

This chapter is organized as follows. Section 11.2 describes specification, estimation and inference in VAR models and introduces the `S+FinMetrics` function `VAR`. Section 11.3 covers forecasting from VAR model. The discussion covers traditional forecasting algorithms as well as simulation-based forecasting algorithms that can impose certain types of conditioning information. Section 11.4 summarizes the types of structural analysis typically performed using VAR models. These analyses include Granger-causality tests, the computation of impulse response functions, and forecast error variance decompositions. Section 11.5 gives an extended example of VAR modeling. The chapter concludes with a brief discussion of Bayesian VAR models.

This chapter provides a relatively non-technical survey of VAR models. VAR models in economics were made popular by Sims (1980). The definitive technical reference for VAR models is Lütkepohl (1991), and updated surveys of VAR techniques are given in Watson (1994) and Lütkepohl (1999) and Waggoner and Zha (1999). Applications of VAR models to financial data are given in Hamilton (1994), Campbell, Lo and MacKinlay (1997), Cuthbertson (1996), Mills (1999) and Tsay (2001).

11.2 The Stationary Vector Autoregression Model

Let $\mathbf{Y}_t = (y_{1t}, y_{2t}, \ldots, y_{nt})'$ denote an $(n \times 1)$ vector of time series variables. The basic p-lag *vector autoregressive* (VAR(p)) model has the form

$$\mathbf{Y}_t = \mathbf{c} + \mathbf{\Pi}_1 \mathbf{Y}_{t-1} + \mathbf{\Pi}_2 \mathbf{Y}_{t-2} + \cdots + \mathbf{\Pi}_p \mathbf{Y}_{t-p} + \boldsymbol{\varepsilon}_t, \; t = 1, \ldots, T \quad (11.1)$$

where $\mathbf{\Pi}_i$ are $(n \times n)$ coefficient matrices and $\boldsymbol{\varepsilon}_t$ is an $(n \times 1)$ unobservable zero mean white noise vector process (serially uncorrelated or independent) with time invariant covariance matrix $\mathbf{\Sigma}$. For example, a bivariate VAR(2) model equation by equation has the form

$$\begin{pmatrix} y_{1t} \\ y_{2t} \end{pmatrix} = \begin{pmatrix} c_1 \\ c_2 \end{pmatrix} + \begin{pmatrix} \pi_{11}^1 & \pi_{12}^1 \\ \pi_{21}^1 & \pi_{22}^1 \end{pmatrix} \begin{pmatrix} y_{1t-1} \\ y_{2t-1} \end{pmatrix} \quad (11.2)$$

$$+ \begin{pmatrix} \pi_{11}^2 & \pi_{12}^2 \\ \pi_{21}^2 & \pi_{22}^2 \end{pmatrix} \begin{pmatrix} y_{1t-2} \\ y_{2t-2} \end{pmatrix} + \begin{pmatrix} \varepsilon_{1t} \\ \varepsilon_{2t} \end{pmatrix} \quad (11.3)$$

or

$$\begin{aligned} y_{1t} &= c_1 + \pi_{11}^1 y_{1t-1} + \pi_{12}^1 y_{2t-1} + \pi_{11}^2 y_{1t-2} + \pi_{12}^2 y_{2t-2} + \varepsilon_{1t} \\ y_{2t} &= c_2 + \pi_{21}^1 y_{1t-1} + \pi_{22}^1 y_{2t-1} + \pi_{21}^2 y_{1t-1} + \pi_{22}^2 y_{2t-1} + \varepsilon_{2t} \end{aligned}$$

where $cov(\varepsilon_{1t}, \varepsilon_{2s}) = \sigma_{12}$ for $t = s$; 0 otherwise. Notice that each equation has the same regressors – lagged values of y_{1t} and y_{2t}. Hence, the VAR(p) model is just a *seemingly unrelated regression* (SUR) model with lagged variables and deterministic terms as common regressors.

In lag operator notation, the VAR(p) is written as

$$\mathbf{\Pi}(L)\mathbf{Y}_t = \mathbf{c} + \boldsymbol{\varepsilon}_t$$

where $\mathbf{\Pi}(L) = \mathbf{I}_n - \mathbf{\Pi}_1 L - ... - \mathbf{\Pi}_p L^p$. The VAR($p$) is stable if the roots of

$$\det\left(\mathbf{I}_n - \mathbf{\Pi}_1 z - \cdots - \mathbf{\Pi}_p z^p\right) = 0$$

lie outside the complex unit circle (have modulus greater than one), or, equivalently, if the eigenvalues of the companion matrix

$$\mathbf{F} = \begin{pmatrix} \mathbf{\Pi}_1 & \mathbf{\Pi}_2 & \cdots & \mathbf{\Pi}_n \\ \mathbf{I}_n & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \ddots & \mathbf{0} & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{I}_n & \mathbf{0} \end{pmatrix}$$

have modulus less than one. Assuming that the process has been initialized in the infinite past, then a stable VAR(p) process is stationary and ergodic with time invariant means, variances, and autocovariances.

If $\mathbf{Y}_t$ in (11.1) is covariance stationary, then the unconditional mean is given by

$$\boldsymbol{\mu} = (\mathbf{I}_n - \mathbf{\Pi}_1 - \cdots - \mathbf{\Pi}_p)^{-1}\mathbf{c}$$

The *mean-adjusted* form of the VAR(p) is then

$$\mathbf{Y}_t - \boldsymbol{\mu} = \mathbf{\Pi}_1(\mathbf{Y}_{t-1} - \boldsymbol{\mu}) + \mathbf{\Pi}_2(\mathbf{Y}_{t-2} - \boldsymbol{\mu}) + \cdots + \mathbf{\Pi}_p(\mathbf{Y}_{t-p} - \boldsymbol{\mu}) + \boldsymbol{\varepsilon}_t$$

The basic VAR(p) model may be too restrictive to represent sufficiently the main characteristics of the data. In particular, other deterministic terms such as a linear time trend or seasonal dummy variables may be required to represent the data properly. Additionally, stochastic exogenous variables may be required as well. The general form of the VAR(p) model with deterministic terms and exogenous variables is given by

$$\mathbf{Y}_t = \mathbf{\Pi}_1\mathbf{Y}_{t-1} + \mathbf{\Pi}_2\mathbf{Y}_{t-2} + \cdots + \mathbf{\Pi}_p\mathbf{Y}_{t-p} + \mathbf{\Phi}\mathbf{D}_t + \mathbf{G}\mathbf{X}_t + \boldsymbol{\varepsilon}_t \qquad (11.4)$$

where $\mathbf{D}_t$ represents an $(l \times 1)$ matrix of deterministic components, $\mathbf{X}_t$ represents an $(m \times 1)$ matrix of exogenous variables, and Φ and $\mathbf{G}$ are parameter matrices.

Example 64 *Simulating a stationary VAR(1) model using* `S-PLUS`

A stationary VAR model may be easily simulated in `S-PLUS` using the `S+FinMetrics` function `simulate.VAR`. The commands to simulate $T = 250$ observations from a bivariate VAR(1) model

$$\begin{aligned} y_{1t} &= -0.7 + 0.7y_{1t-1} + 0.2y_{2t-1} + \varepsilon_{1t} \\ y_{2t} &= 1.3 + 0.2y_{1t-1} + 0.7y_{2t-1} + \varepsilon_{2t} \end{aligned}$$

with

$$\mathbf{\Pi}_1 = \begin{pmatrix} 0.7 & 0.2 \\ 0.2 & 0.7 \end{pmatrix}, \ \mathbf{c} = \begin{pmatrix} -0.7 \\ 1.3 \end{pmatrix}, \ \boldsymbol{\mu} = \begin{pmatrix} 1 \\ 5 \end{pmatrix}, \ \mathbf{\Sigma} = \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix}$$

and normally distributed errors are

```
> pi1 = matrix(c(0.7,0.2,0.2,0.7),2,2)
> mu.vec = c(1,5)
> c.vec = as.vector((diag(2)-pi1)%*%mu.vec)
> cov.mat = matrix(c(1,0.5,0.5,1),2,2)
> var1.mod = list(const=c.vec,ar=pi1,Sigma=cov.mat)
> set.seed(301)
> y.var = simulate.VAR(var1.mod,n=250,
+ y0=t(as.matrix(mu.vec)))
> dimnames(y.var) = list(NULL,c("y1","y2"))
```

The simulated data are shown in Figure 11.1. The VAR is stationary since the eigenvalues of $\mathbf{\Pi}_1$ are less than one:

```
> eigen(pi1,only.values=T)
$values:
[1] 0.9 0.5

$vectors:
NULL
```

Notice that the intercept values are quite different from the mean values of y_1 and y_2:

```
> c.vec
[1] -0.7  1.3
> colMeans(y.var)
     y1    y2
 0.8037 4.751
```

11.2.1 Estimation

Consider the basic VAR(p) model (11.1). Assume that the VAR(p) model is covariance stationary, and there are no restrictions on the parameters of the model. In SUR notation, each equation in the VAR(p) may be written as

$$\mathbf{y}_i = \mathbf{Z}\boldsymbol{\pi}_i + \mathbf{e}_i, \ i = 1, \ldots, n$$

where $\mathbf{y}_i$ is a $(T \times 1)$ vector of observations on the i^{th} equation, $\mathbf{Z}$ is a $(T \times k)$ matrix with t^{th} row given by $\mathbf{Z}_t' = (1, \mathbf{Y}_{t-1}', \ldots, \mathbf{Y}_{t-p}')$, $k = np + 1$, $\boldsymbol{\pi}_i$ is a $(k \times 1)$ vector of parameters and $\mathbf{e}_i$ is a $(T \times 1)$ error with covariance matrix $\sigma_i^2 \mathbf{I}_T$. Since the VAR(p) is in the form of a SUR model

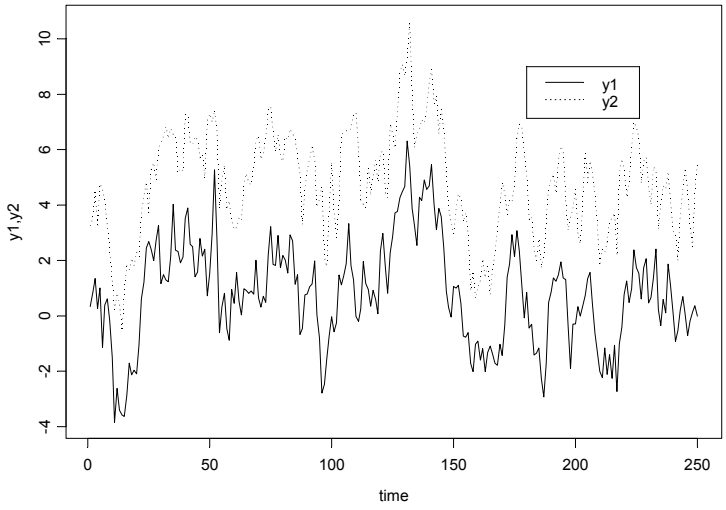

FIGURE 11.1. Simulated stationary VAR(1) model.

where each equation has the same explanatory variables, each equation may be estimated separately by ordinary least squares without losing efficiency relative to generalized least squares. Let $\hat{\mathbf{\Pi}} = [\hat{\boldsymbol{\pi}}_1, \ldots, \hat{\boldsymbol{\pi}}_n]$ denote the $(k \times n)$ matrix of least squares coefficients for the n equations.

Let $vec(\hat{\mathbf{\Pi}})$ denote the operator that stacks the columns of the $(n \times k)$ matrix $\hat{\mathbf{\Pi}}$ into a long $(nk \times 1)$ vector. That is,

$$vec(\hat{\mathbf{\Pi}}) = \begin{pmatrix} \hat{\boldsymbol{\pi}}_1 \\ \vdots \\ \hat{\boldsymbol{\pi}}_n \end{pmatrix}$$

Under standard assumptions regarding the behavior of stationary and ergodic VAR models (see Hamilton (1994) or Lütkepohl (1991)) $vec(\hat{\mathbf{\Pi}})$ is consistent and asymptotically normally distributed with asymptotic covariance matrix

$$\widehat{avar}(vec(\hat{\mathbf{\Pi}})) = \hat{\mathbf{\Sigma}} \otimes (\mathbf{Z}'\mathbf{Z})^{-1}$$

where

$$\hat{\mathbf{\Sigma}} = \frac{1}{T-k} \sum_{t=1}^{T} \hat{\boldsymbol{\varepsilon}}_t \hat{\boldsymbol{\varepsilon}}_t'$$

and $\hat{\boldsymbol{\varepsilon}}_t = \mathbf{Y}_t - \hat{\mathbf{\Pi}}'\mathbf{Z}_t$ is the multivariate least squares residual from (11.1) at time t.

11.2.2 Inference on Coefficients

The i^{th} element of $vec(\hat{\Pi})$, $\hat{\pi}_i$, is asymptotically normally distributed with standard error given by the square root of i^{th} diagonal element of $\hat{\mathbf{\Sigma}} \otimes (\mathbf{Z}'\mathbf{Z})^{-1}$. Hence, asymptotically valid t-tests on individual coefficients may be constructed in the usual way. More general linear hypotheses of the form $\mathbf{R} \cdot vec(\mathbf{\Pi}) = \mathbf{r}$ involving coefficients across different equations of the VAR may be tested using the Wald statistic

$$Wald = (\mathbf{R} \cdot vec(\hat{\mathbf{\Pi}}) - \mathbf{r})' \left\{ \mathbf{R} \left[\widehat{avar}(vec(\hat{\mathbf{\Pi}})) \right] \mathbf{R}' \right\}^{-1} (\mathbf{R} \cdot vec(\hat{\mathbf{\Pi}}) - \mathbf{r}) \quad (11.5)$$

Under the null, (11.5) has a limiting $\chi^2(q)$ distribution where $q = rank(\mathbf{R})$ gives the number of linear restrictions.

11.2.3 Lag Length Selection

The lag length for the VAR(p) model may be determined using model selection criteria. The general approach is to fit VAR(p) models with orders $p = 0, ..., p_{max}$ and choose the value of p which minimizes some model selection criteria. Model selection criteria for VAR(p) models have the form

$$IC(p) = \ln |\tilde{\mathbf{\Sigma}}(p)| + c_T \cdot \varphi(n, p)$$

where $\tilde{\mathbf{\Sigma}}(p) = T^{-1} \sum_{t=1}^{T} \hat{\boldsymbol{\varepsilon}}_t \hat{\boldsymbol{\varepsilon}}_t'$ is the residual covariance matrix *without a degrees of freedom correction* from a VAR(p) model, c_T is a sequence indexed by the sample size T, and $\varphi(n, p)$ is a penalty function which penalizes large VAR(p) models. The three most common information criteria are the Akaike (AIC), Schwarz-Bayesian (BIC) and Hannan-Quinn (HQ):

$$\begin{aligned} AIC(p) &= \ln |\tilde{\mathbf{\Sigma}}(p)| + \frac{2}{T} pn^2 \\ BIC(p) &= \ln |\tilde{\mathbf{\Sigma}}(p)| + \frac{\ln T}{T} pn^2 \\ HQ(p) &= \ln |\tilde{\mathbf{\Sigma}}(p)| + \frac{2 \ln \ln T}{T} pn^2 \end{aligned}$$

The AIC criterion asymptotically overestimates the order with positive probability, whereas the BIC and HQ criteria estimate the order consistently under fairly general conditions if the true order p is less than or equal to p_{max}. For more information on the use of model selection criteria in VAR models see Lütkepohl (1991) chapter four.

11.2.4 Estimating VAR Models Using the S+FinMetrics Function VAR

The `S+FinMetrics` function `VAR` is designed to fit and analyze VAR models as described in the previous section. `VAR` produces an object of class "`VAR`"

for which there are `print`, `summary, plot` and `predict` methods as well as extractor functions `coefficients`, `residuals`, `fitted` and `vcov`. The calling syntax of `VAR` is a bit complicated because it is designed to handle multivariate data in matrices, data frames as well as "`timeSeries`" objects. The use of `VAR` is illustrated with the following example.

Example 65 *Bivariate VAR model for exchange rates*

This example considers a bivariate VAR model for $\mathbf{Y}_t = (\Delta s_t, fp_t)'$, where s_t is the logarithm of the monthly spot exchange rate between the US and Canada, $fp_t = f_t - s_t = i_t^{US} - i_t^{CA}$ is the forward premium or interest rate differential, and f_t is the natural logarithm of the 30-day forward exchange rate. The data over the 20 year period March 1976 through June 1996 is in the `S+FinMetrics` "`timeSeries`" `lexrates.dat`. The data for the VAR model are computed as

```
> dspot = diff(lexrates.dat[,"USCNS"])
> fp = lexrates.dat[,"USCNF"]-lexrates.dat[,"USCNS"]
> uscn.ts = seriesMerge(dspot,fp)
> colIds(uscn.ts) = c("dspot","fp")
> uscn.ts@title = "US/CN Exchange Rate Data"
> par(mfrow=c(2,1))
> plot(uscn.ts[,"dspot"],main="1st difference of US/CA spot
+ exchange rate")
> plot(uscn.ts[,"fp"],main="US/CN interest rate
+ differential")
```

Figure 11.2 illustrates the monthly return Δs_t and the forward premium fp_t over the period March 1976 through June 1996. Both series appear to be $I(0)$ (which can be confirmed using the `S+FinMetrics` functions `unitroot` or `stationaryTest`) with Δs_t much more volatile than fp_t. fp_t also appears to be heteroskedastic.

Specifying and Estimating the VAR(p) Model

To estimate a VAR(1) model for $\mathbf{Y}_t$ use

```
> var1.fit = VAR(cbind(dspot,fp)~ar(1),data=uscn.ts)
```

Note that the VAR model is specified using an `S-PLUS` formula, with the multivariate response on the left hand side of the `~` operator and the built-in AR term specifying the lag length of the model on the right hand side. The optional `data` argument accepts a data frame or "`timeSeries`" object with variable names matching those used in specifying the formula. If the data are in a "`timeSeries`" object or in an unattached data frame ("`timeSeries`" objects cannot be attached) then the `data` argument must be used. If the data are in a matrix then the `data` argument may be omitted. For example,

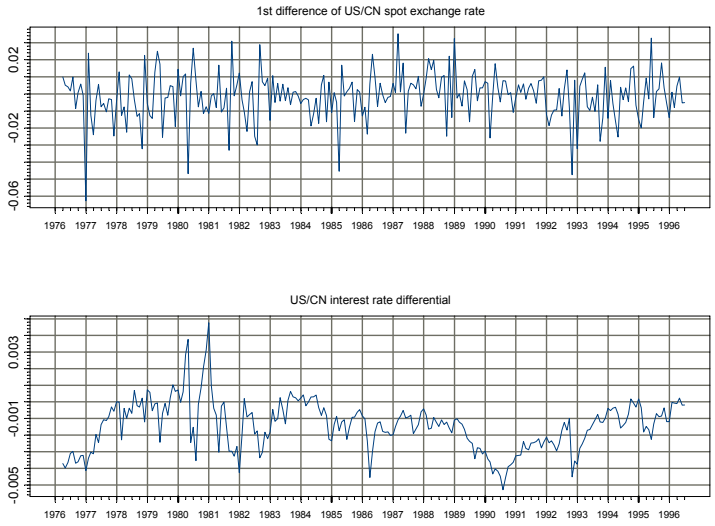

FIGURE 11.2. US/CN forward premium and spot rate.

```
> uscn.mat = as.matrix(seriesData(uscn.ts))
> var2.fit = VAR(uscn.mat~ar(1))
```

If the data are in a "`timeSeries`" object then the `start` and `end` options may be used to specify the estimation sample. For example, to estimate the VAR(1) over the sub-period January 1980 through January 1990

```
> var3.fit = VAR(cbind(dspot,fp)~ar(1), data=uscn.ts,
+ start="Jan 1980", end="Jan 1990", in.format="%m %Y")
```

may be used. The use of `in.format=\%m %Y"` sets the format for the date strings specified in the `start` and `end` options to match the input format of the dates in the positions slot of `uscn.ts`.

The VAR model may be estimated with the lag length p determined using a specified information criterion. For example, to estimate the VAR for the exchange rate data with p set by minimizing the BIC with a maximum lag $p_{\max} = 4$ use

```
> var4.fit = VAR(uscn.ts,max.ar=4, criterion="BIC")
> var4.fit$info
    ar(1) ar(2) ar(3) ar(4)
BIC -4028 -4013 -3994 -3973
```

When a formula is not specified and only a data frame, "`timeSeries`" or matrix is supplied that contains the variables for the VAR model, `VAR` fits

all VAR(p) models with lag lengths p less than or equal to the value given to `max.ar`, and the lag length is determined as the one which minimizes the information criterion specified by the `criterion` option. The default criterion is `BIC` but other valid choices are `logL`, `AIC` and `HQ`. In the computation of the information criteria, a common sample based on `max.ar` is used. Once the lag length is determined, the VAR is re-estimated using the appropriate sample. In the above example, the BIC values were computed using the sample based on `max.ar=4` and $p = 1$ minimizes BIC. The VAR(1) model was automatically re-estimated using the sample size appropriate for $p = 1$.

Print and Summary Methods

The function `VAR` produces an object of class "`VAR`" with the following components.

```
> class(var1.fit)
[1] "VAR"
> names(var1.fit)
 [1] "R"         "coef"      "fitted"    "residuals"
 [5] "Sigma"     "df.resid"  "rank"      "call"
 [9] "ar.order"  "n.na"      "terms"     "Y0"
```

To see the estimated coefficients of the model use the `print` method:

```
> var1.fit

Call:
VAR(formula = cbind(dspot, fp) ~ar(1), data = uscn.ts)

Coefficients:
              dspot      fp
(Intercept) -0.0036 -0.0003
 dspot.lag1 -0.1254  0.0079
    fp.lag1 -1.4833  0.7938

Std. Errors of Residuals:
  dspot     fp
 0.0137 0.0009

Information Criteria:
  logL   AIC   BIC    HQ
  2058 -4104 -4083 -4096

                   total residual
Degree of freedom:   243      240
Time period: from Apr 1976 to Jun 1996
```

The first column under the label "`Coefficients:`" gives the estimated coefficients for the Δs_t equation, and the second column gives the estimated coefficients for the fp_t equation:

$$\begin{aligned} \Delta s_t &= -0.0036 - 0.1254 \cdot \Delta s_{t-1} - 1.4833 \cdot fp_{t-1} \\ fp_t &= -0.0003 + 0.0079 \cdot \Delta s_{t-1} + 0.7938 \cdot fp_{t-1} \end{aligned}$$

Since `uscn.ts` is a "`timeSeries`" object, the estimation time period is also displayed.

The `summary` method gives more detailed information about the fitted VAR:

```
> summary(var1.fit)

Call:
VAR(formula = cbind(dspot, fp) ~ar(1), data = uscn.ts)

Coefficients:
               dspot       fp
(Intercept)  -0.0036  -0.0003
  (std.err)   0.0012   0.0001
   (t.stat)  -2.9234  -3.2885

 dspot.lag1  -0.1254   0.0079
  (std.err)   0.0637   0.0042
   (t.stat)  -1.9700   1.8867

    fp.lag1  -1.4833   0.7938
  (std.err)   0.5980   0.0395
   (t.stat)  -2.4805  20.1049

Regression Diagnostics:
                dspot     fp
     R-squared 0.0365 0.6275
Adj. R-squared 0.0285 0.6244
  Resid. Scale 0.0137 0.0009

Information Criteria:
  logL   AIC   BIC    HQ
  2058 -4104 -4083 -4096

                   total residual
Degree of freedom:   243      240
Time period: from Apr 1976 to Jun 1996
```

In addition to the coefficient standard errors and t-statistics, `summary` also displays R^2 measures for each equation (which are valid because each equa-

tion is estimated by least squares). The summary output shows that the coefficients on Δs_{t-1} and fp_{t-1} in both equations are statistically significant at the 10% level and that the fit for the fp_t equation is much better than the fit for the Δs_t equation.

As an aside, note that the `S+FinMetrics` function `OLS` may also be used to estimate each equation in a VAR model. For example, one way to compute the equation for Δs_t using `OLS` is

```
> dspot.fit = OLS(dspot~ar(1)+tslag(fp),data=uscn.ts)
> dspot.fit

Call:
OLS(formula = dspot ~ar(1) + tslag(fp), data = uscn.ts)

Coefficients:
 (Intercept) tslag(fp)    lag1
 -0.0036     -1.4833   -0.1254

Degrees of freedom: 243 total; 240 residual
Time period: from Apr 1976 to Jun 1996
Residual standard error: 0.01373
```

Graphical Diagnostics

The `plot` method for "`VAR`" objects may be used to graphically evaluate the fitted VAR. By default, the `plot` method produces a menu of plot options:

```
> plot(var1.fit)

Make a plot selection (or 0 to exit):

1: plot: All
2: plot: Response and Fitted Values
3: plot: Residuals
4: plot: Normal QQplot of Residuals
5: plot: ACF of Residuals
6: plot: PACF of Residuals
7: plot: ACF of Squared Residuals
8: plot: PACF of Squared Residuals
Selection:
```

Alternatively, `plot.VAR` may be called directly. The function `plot.VAR` has arguments

```
> args(plot.VAR)
function(x, ask = T, which.plots = NULL, hgrid = F, vgrid
= F, ...)
```

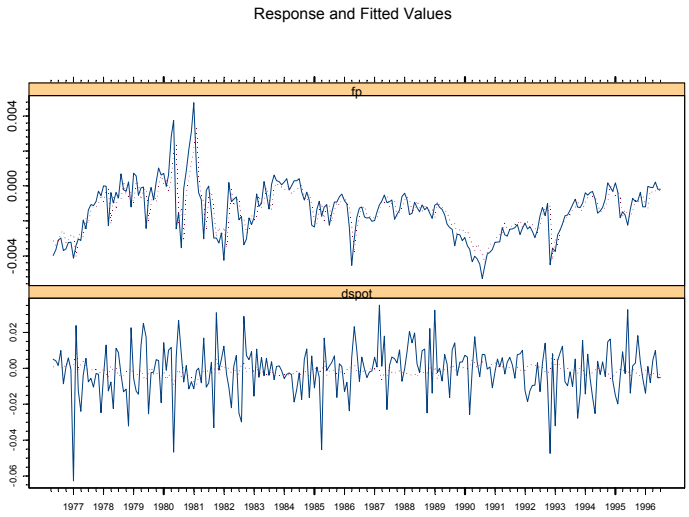

FIGURE 11.3. Response and fitted values from VAR(1) model for US/CN exchange rate data.

To create all seven plots without using the menu, set `ask=F`. To create the Residuals plot without using the menu, set `which.plot=2`. The optional arguments `hgrid` and `vgrid` control printing of horizontal and vertical grid lines on the plots.

Figures 11.3 and 11.4 give the Response and Fitted Values and Residuals plots for the VAR(1) fit to the exchange rate data. The equation for fp_t fits much better than the equation for Δs_t. The residuals for both equations look fairly random, but the residuals for the fp_t equation appear to be heteroskedastic. The qq-plot (not shown) indicates that the residuals for the Δs_t equation are highly non-normal.

Extractor Functions

The residuals and fitted values for each equation of the VAR may be extracted using the generic extractor functions `residuals` and `fitted`:

```
> var1.resid = resid(var1.fit)
> var1.fitted = fitted(var.fit)
> var1.resid[1:3,]
 Positions     dspot          fp
 Apr 1976  0.0044324 -0.00084150
 May 1976  0.0024350 -0.00026493
 Jun 1976  0.0004157  0.00002435
```

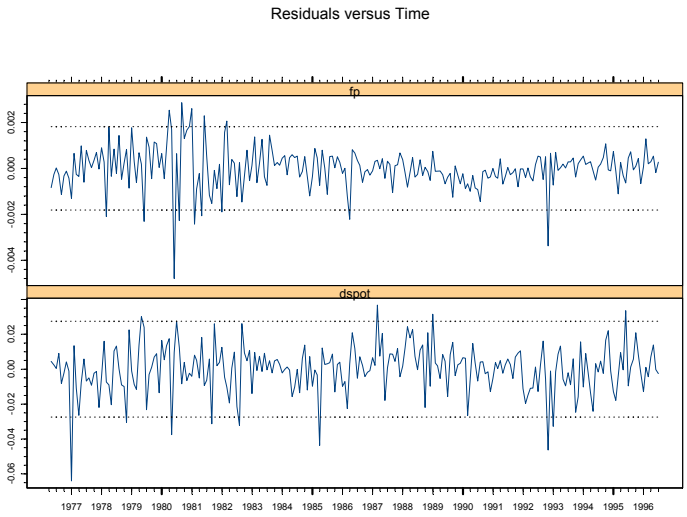

FIGURE 11.4. Residuals from VAR(1) model fit to US/CN exchange rate data.

Notice that since the data are in a "`timeSeries`" object, the extracted residuals and fitted values are also "`timeSeries`" objects.

The coefficients of the VAR model may be extracted using the generic `coef` function:

```
> coef(var1.fit)
                    dspot            fp
(Intercept) -0.003595149 -0.0002670108
 dspot.lag1 -0.125397056  0.0079292865
    fp.lag1 -1.483324622  0.7937959055
```

Notice that `coef` produces the (3×2) matrix $\hat{\mathbf{\Pi}}$ whose columns give the estimated coefficients for each equation in the VAR(1).

To test stability of the VAR, extract the matrix $\mathbf{\Pi}_1$ and compute its eigenvalues

```
> PI1 = t(coef(var1.fit)[2:3,])
> abs(eigen(PI1,only.values=T)$values)
[1] 0.7808 0.1124
```

Since the modulus of the two eigenvalues of $\mathbf{\Pi}_1$ are less than 1, the VAR(1) is stable.

Testing Linear Hypotheses

Now, consider testing the hypothesis that $\mathbf{\Pi}_1 = \mathbf{0}$ (i.e., $\mathbf{Y}_{t-1}$ does not help to explain $\mathbf{Y}_t$) using the Wald statistic (11.5). In terms of the columns of $vec(\mathbf{\Pi})$ the restrictions are $\boldsymbol{\pi}_1 = (c_1, 0, 0)'$ and $\boldsymbol{\pi}_2 = (c_2, 0, 0)$ and may be expressed as $\mathbf{R}vec(\mathbf{\Pi}) = \mathbf{r}$ with

$$\mathbf{R} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}, \mathbf{r} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

The Wald statistic is easily constructed as follows

```
> R = matrix(c(0,1,0,0,0,0,
+ 0,0,1,0,0,0,
+ 0,0,0,0,1,0,
+ 0,0,0,0,0,1),
+ 4,6,byrow=T)
> vecPi = as.vector(var1.fit$coef)
> avar = R%*%vcov(var1.fit)%*%t(R)
> wald = t(R%*%vecPi)%*%solve(avar)%*%(R%*%vecPi)
> wald
      [,1]
[1,] 417.1
> 1-pchisq(wald,4)
[1] 0
```

Since the p-value for the Wald statistic based on the $\chi^2(4)$ distribution is essentially zero, the hypothesis that $\mathbf{\Pi}_1 = \mathbf{0}$ should be rejected at any reasonable significance level.

11.3 Forecasting

Forecasting is one of the main objectives of multivariate time series analysis. Forecasting from a VAR model is similar to forecasting from a univariate AR model and the following gives a brief description.

11.3.1 Traditional Forecasting Algorithm

Consider first the problem of forecasting future values of $\mathbf{Y}_t$ when the parameters $\mathbf{\Pi}$ of the VAR(p) process are assumed to be known and there are no deterministic terms or exogenous variables. The best linear predictor, in terms of minimum mean squared error (MSE), of $\mathbf{Y}_{t+1}$ or 1-step forecast based on information available at time T is

$$\mathbf{Y}_{T+1|T} = \mathbf{c} + \mathbf{\Pi}_1 \mathbf{Y}_T + \cdots + \mathbf{\Pi}_p \mathbf{Y}_{T-p+1}$$

Forecasts for longer horizons h (h-step forecasts) may be obtained using the *chain-rule of forecasting* as

$$\mathbf{Y}_{T+h|T} = \mathbf{c} + \mathbf{\Pi}_1 \mathbf{Y}_{T+h-1|T} + \cdots + \mathbf{\Pi}_p \mathbf{Y}_{T+h-p|T}$$

where $\mathbf{Y}_{T+j|T} = \mathbf{Y}_{T+j}$ for $j \leq 0$. The h-step forecast errors may be expressed as

$$\mathbf{Y}_{T+h} - \mathbf{Y}_{T+h|T} = \sum_{s=0}^{h-1} \mathbf{\Psi}_s \boldsymbol{\varepsilon}_{T+h-s}$$

where the matrices $\mathbf{\Psi}_s$ are determined by recursive substitution

$$\mathbf{\Psi}_s = \sum_{j=1}^{p-1} \mathbf{\Psi}_{s-j} \mathbf{\Pi}_j \tag{11.6}$$

with $\mathbf{\Psi}_0 = \mathbf{I}_n$ and $\mathbf{\Pi}_j = 0$ for $j > p$.[1] The forecasts are unbiased since all of the forecast errors have expectation zero and the MSE matrix for $\mathbf{Y}_{t+h|T}$ is

$$\begin{aligned} \mathbf{\Sigma}(h) &= MSE\left(\mathbf{Y}_{T+h} - \mathbf{Y}_{T+h|T}\right) \\ &= \sum_{s=0}^{h-1} \mathbf{\Psi}_s \mathbf{\Sigma} \mathbf{\Psi}_s' \end{aligned} \tag{11.7}$$

Now consider forecasting $\mathbf{Y}_{T+h}$ when the parameters of the VAR(p) process are estimated using multivariate least squares. The best linear predictor of $\mathbf{Y}_{T+h}$ is now

$$\hat{\mathbf{Y}}_{T+h|T} = \hat{\mathbf{\Pi}}_1 \hat{\mathbf{Y}}_{T+h-1|T} + \cdots + \hat{\mathbf{\Pi}}_p \hat{\mathbf{Y}}_{T+h-p|T} \tag{11.8}$$

where $\hat{\mathbf{\Pi}}_j$ are the estimated parameter matrices. The h-step forecast error is now

$$\mathbf{Y}_{T+h} - \hat{\mathbf{Y}}_{T+h|T} = \sum_{s=0}^{h-1} \mathbf{\Psi}_s \boldsymbol{\varepsilon}_{T+h-s} + \left(\mathbf{Y}_{T+h} - \hat{\mathbf{Y}}_{T+h|T}\right) \tag{11.9}$$

and the term $\left(\mathbf{Y}_{T+h} - \hat{\mathbf{Y}}_{T+h|T}\right)$ captures the part of the forecast error due to estimating the parameters of the VAR. The MSE matrix of the h-step forecast is then

$$\hat{\mathbf{\Sigma}}(h) = \mathbf{\Sigma}(h) + MSE\left(\mathbf{Y}_{T+h} - \hat{\mathbf{Y}}_{T+h|T}\right)$$

[1] The `S+FinMetrics` fucntion `VAR.ar2ma` computes the Ψ_s matrices given the $\mathbf{\Pi}_j$ matrices using (11.6).

In practice, the second term $MSE\left(\mathbf{Y}_{T+h} - \hat{\mathbf{Y}}_{T+h|T}\right)$ is often ignored and $\hat{\mathbf{\Sigma}}(h)$ is computed using (11.7) as

$$\hat{\mathbf{\Sigma}}(h) = \sum_{s=0}^{h-1} \hat{\mathbf{\Psi}}_s \hat{\mathbf{\Sigma}} \hat{\mathbf{\Psi}}_s' \tag{11.10}$$

with $\hat{\mathbf{\Psi}}_s = \sum_{j=1}^{s} \hat{\mathbf{\Psi}}_{s-j} \hat{\mathbf{\Pi}}_j$. Lütkepohl (1991, chapter 3) gives an approximation to $MSE\left(\mathbf{Y}_{T+h} - \hat{\mathbf{Y}}_{T+h|T}\right)$ which may be interpreted as a finite sample correction to (11.10).

Asymptotic $(1-\alpha)\cdot 100\%$ confidence intervals for the individual elements of $\hat{\mathbf{Y}}_{T+h|T}$ are then computed as

$$\left[\hat{y}_{k,T+h|T} - c_{1-\alpha/2}\hat{\sigma}_k(h),\ \hat{y}_{k,T+h|T} + c_{1-\alpha/2}\hat{\sigma}_k(h)\right]$$

where $c_{1-\alpha/2}$ is the $(1-\alpha/2)$ quantile of the standard normal distribution and $\hat{\sigma}_k(h)$ denotes the square root of the diagonal element of $\hat{\mathbf{\Sigma}}(h)$.

Example 66 *Forecasting exchange rates from a bivariate VAR*

Consider computing h-step forecasts, $h = 1, \ldots, 12$, along with estimated forecast standard errors from the bivariate VAR(1) model for exchange rates. Forecasts and forecast standard errors from the fitted VAR may be computed using the generic `S-PLUS` `predict` method

```
> uscn.pred = predict(var1.fit,n.predict=12)
```

The `predict` function recognizes `var1.fit` as a "`VAR`" object, and calls the appropriate method function `predict.VAR`. Alternatively, `predict.VAR` can be applied directly on an object inheriting from class "`VAR`". See the online help for explanations of the arguments to `predict.VAR`.

The output of `predict.VAR` is an object of class "`forecast`" for which there are `print`, `summary` and `plot` methods. To see just the forecasts, the `print` method will suffice:

```
> uscn.pred

Predicted Values:

                dspot      fp
 1-step-ahead -0.0027 -0.0005
 2-step-ahead -0.0026 -0.0006
 3-step-ahead -0.0023 -0.0008
 4-step-ahead -0.0021 -0.0009
 5-step-ahead -0.0020 -0.0010
 6-step-ahead -0.0018 -0.0011
 7-step-ahead -0.0017 -0.0011
```

```
 8-step-ahead -0.0017 -0.0012
 9-step-ahead -0.0016 -0.0012
10-step-ahead -0.0016 -0.0013
11-step-ahead -0.0015 -0.0013
12-step-ahead -0.0015 -0.0013
```

The forecasts and their standard errors can be shown using `summary`:

```
> summary(uscn.pred)

Predicted Values with Standard Errors:

               dspot      fp
1-step-ahead -0.0027 -0.0005
   (std.err)  0.0137  0.0009
2-step-ahead -0.0026 -0.0006
   (std.err)  0.0139  0.0012

...

12-step-ahead -0.0015 -0.0013
    (std.err)  0.0140  0.0015
```

Lütkepohl's finite sample correction to the forecast standard errors computed from asymptotic theory may be obtained by using the optional argument `fs.correction=T` in the call to `predict.VAR`.

The forecasts can also be plotted together with the original data using the generic `plot` function as follows:

```
> plot(uscn.pred,uscn.ts,n.old=12)
```

where the `n.old` optional argument specifies the number of observations to plot from `uscn.ts`. If `n.old` is not specified, all the observations in `uscn.ts` will be plotted together with `uscn.pred`. Figure 11.5 shows the forecasts produced from the VAR(1) fit to the US/CN exchange rate data[2]. At the beginning of the forecast horizon the spot return is below its estimated mean value, and the forward premium is above its mean values. The spot return forecasts start off negative and grow slowly toward the mean, and the forward premium forecasts decline sharply toward the mean. The forecast standard errors for both sets of forecasts, however, are fairly large.

[2]Notice that the dates associated with the forecasts are not shown. This is the result of "`timeDate`" objects not having a well defined frequency from which to extrapolate dates.

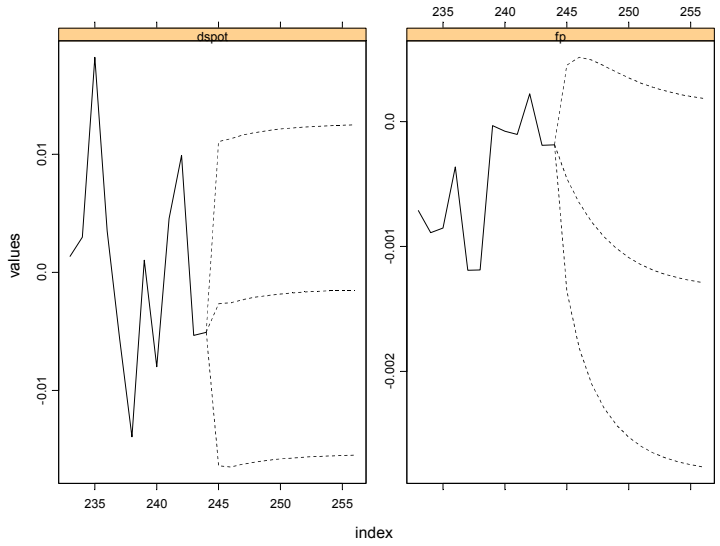

FIGURE 11.5. Predicted values from VAR(1) model fit to US/CN exchange rate data.

11.3.2 Simulation-Based Forecasting

The previous subsection showed how to generate multivariate forecasts from a fitted VAR model, using the chain-rule of forecasting (11.8). Since the multivariate forecast errors (11.9) are asymptotically normally distributed with covariance matrix (11.10), the forecasts of $\mathbf{Y}_{t+h}$ can be simulated by generating multivariate normal random variables with mean zero and covariance matrix (11.10). These simulation-based forecasts can be obtained by setting the optional argument `method` to `"mc"` in the call to `predict.VAR`.

When `method="mc"`, the multivariate normal random variables are actually generated as a vector of standard normal random variables scaled by the Cholesky factor of the covariance matrix (11.10). Instead of using standard normal random variables, one could also use the standardized residuals from the fitted VAR model. Simulation-based forecasts based on this approach are obtained by setting the optional argument method to `"bootstrap"` in the call to `predict.VAR`.

Example 67 *Simulation-based forecasts of exchange rate data from bivariate VAR*

The h-step forecasts ($h = 1, \ldots, 12$) for Δs_{t+h} and fp_{t+h} using the Monte Carlo simulation method are

```
> uscn.pred.MC = predict(var1.fit,n.predict=12,method="mc")
> summary(uscn.pred.MC)

Predicted Values with Standard Errors:

                dspot      fp
 1-step-ahead -0.0032 -0.0005
    (std.err)  0.0133  0.0009
 2-step-ahead -0.0026 -0.0006
    (std.err)  0.0133  0.0012
...
12-step-ahead -0.0013 -0.0013
    (std.err)  0.0139  0.0015
```

The Monte Carlo forecasts and forecast standard errors for fp_{t+h} are almost identical to those computed using the chain-rule of forecasting. The Monte Carlo forecasts for Δs_{t+h} are slightly different and the forecast standard errors are slightly larger than the corresponding values computed from the chain-rule.

The h-step forecasts computed from the bootstrap simulation method are

```
> uscn.pred.boot = predict(var1.fit,n.predict=12,
+ method="bootstrap")
> summary(uscn.pred.boot)

Predicted Values with Standard Errors:

                dspot      fp
 1-step-ahead -0.0020 -0.0005
    (std.err)  0.0138  0.0009
 2-step-ahead -0.0023 -0.0007
    (std.err)  0.0140  0.0012
...
12-step-ahead -0.0023 -0.0013
    (std.err)  0.0145  0.0015
```

As with the Monte Carlo forecasts, the bootstrap forecasts and forecast standard errors for fp_{t+h} are almost identical to those computed using the chain-rule of forecasting. The bootstrap forecasts for Δs_{t+h} are slightly different from the chain-rule and Monte Carlo forecasts. In particular, the bootstrap forecast standard errors are larger than corresponding values from the chain-rule and Monte Carlo methods.

The simulation-based forecasts described above are different from the traditional simulation-based approach taken in VAR literature, e.g., see

Runkle (1987). The traditional approach is implemented using the following procedure:

1. Obtain VAR coefficient estimates $\mathbf{\Pi}$ and residuals $\boldsymbol{\varepsilon}_t$.

2. Simulate the fitted VAR model by Monte Carlo simulation or by bootstrapping the fitted residuals $\hat{\boldsymbol{\varepsilon}}_t$.

3. Obtain new estimates of $\mathbf{\Pi}$ and forecasts of $\mathbf{Y}_{t+h}$ based on the simulated data.

The above procedure is repeated many times to obtain simulation-based forecasts as well as their confidence intervals. To illustrate this approach, generate 12-step ahead forecasts from the fitted VAR object `var1.fit` by Monte Carlo simulation using the `S+FinMetrics` function `simulate.VAR` as follows:

```
> set.seed(10)
> n.pred=12
> n.sim=100
> sim.pred = array(0,c(n.sim, n.pred, 2))
> y0 = seriesData(var1.fit$Y0)
> for (i in 1:n.sim) {
+    dat = simulate.VAR(var1.fit,n=243)
+    dat = rbind(y0,dat)
+    mod = VAR(dat~ar(1))
+    sim.pred[i,,] = predict(mod,n.pred)$values
+ }
```

The simulation-based forecasts are obtained by averaging the simulated forecasts:

```
> colMeans(sim.pred)
             [,1]       [,2]
 [1,] -0.0017917 -0.0012316
 [2,] -0.0017546 -0.0012508
 [3,] -0.0017035 -0.0012643
 [4,] -0.0016800 -0.0012741
 [5,] -0.0016587 -0.0012814
 [6,] -0.0016441 -0.0012866
 [7,] -0.0016332 -0.0012904
 [8,] -0.0016253 -0.0012932
 [9,] -0.0016195 -0.0012953
[10,] -0.0016153 -0.0012967
[11,] -0.0016122 -0.0012978
[12,] -0.0016099 -0.0012986
```

Comparing these forecasts with those in `uscn.pred` computed earlier, one can see that for the first few forecasts, these simulated forecasts are slightly different from the asymptotic forecasts. However, at larger steps, they approach the long run stable values of the asymptotic forecasts.

Conditional Forecasting

The forecasts algorithms considered up to now are unconditional multivariate forecasts. However, sometimes it is desirable to obtain forecasts of some variables in the system conditional on some knowledge of the future path of other variables in the system. For example, when forecasting multivariate macroeconomic variables using quarterly data from a VAR model, it may happen that some of the future values of certain variables in the VAR model are known, because data on these variables are released earlier than data on the other variables. By incorporating the knowledge of the future path of certain variables, in principle it should be possible to obtain more reliable forecasts of the other variables in the system. Another use of conditional forecasting is the generation of forecasts conditional on different "policy" scenarios. These scenario-based conditional forecasts allow one to answer the question: if something happens to some variables in the system in the future, how will it affect forecasts of other variables in the future?

`S+FinMetrics` provides a generic function `cpredict` for computing conditional forecasts, which has a method `cpredict.VAR` for "`VAR`" objects. The algorithms in `cpredict.VAR` are based on the conditional forecasting algorithms described in Waggoner and Zha (1999). Waggoner and Zha classify conditional information into "hard" conditions and "soft conditions". The hard conditions restrict the future values of certain variables at fixed values, while the soft conditions restrict the future values of certain variables in specified ranges. The arguments taken by `cpredict.VAR` are:

```
> args(cpredict.VAR)
function(object, n.predict = 1, newdata = NULL, olddata = NULL,
method = "mc", unbiased = T, variables.conditioned =
NULL, steps.conditioned = NULL, upper = NULL, lower =
NULL, middle = NULL, seed = 100, n.sim = 1000)
```

Like most `predict` methods in `S-PLUS`, the first argument must be a fitted model object, while the second argument, `n.predict`, specifies the number of steps to predict ahead. The arguments `newdata` and `olddata` can usually be safely ignored, unless exogenous variables were used in fitting the model.

With classical forecasts that ignore the uncertainty in coefficient estimates, hard conditional forecasts can be obtained in closed form as shown by Doan, Litterman and Sims (1984), and Waggoner and Zha (1999). To obtain hard conditional forecasts, the argument `middle` is used to specify fixed values of certain variables at certain steps. For example, to fix the

1-step ahead forecast of `dspot` in `var1.fit` at -0.005 and generate other predictions for 2-step ahead forecasts, use the following command:

```
> cpredict(var1.fit, n.predict=2, middle=-0.005,
+ variables="dspot", steps=1)

Predicted Values:

               dspot      fp
1-step-ahead -0.0050 -0.0005
2-step-ahead -0.0023 -0.0007
```

In the call to `cpredict`, the optional argument `variables` is used to specify the restricted variables, and `steps` to specify the restricted steps.

To specify a soft condition, the optional arguments `upper` and `lower` are used to specify the upper bound and lower bound, respectively, of a soft condition. Since closed form results are not available for soft conditional forecasts, either Monte Carlo simulation or bootstrap methods are used to obtain the actual forecasts. The simulations follow a similar procedure implemented in the function `predict.VAR`, except that a reject/accept method to sample from the distribution conditional on the soft conditions is used. For example, to restrict the range of the first 2-step ahead forecasts of `dspot` to be $(-0.004, -0.001)$ use:

```
> cpredict(var1.fit, n.predict=2, lower=c(-0.004, -0.004),
+ upper=c(-0.001, -0.001), variables="dspot",
+ steps=c(1,2))

Predicted Values:

               dspot      fp
1-step-ahead -0.0027 -0.0003
2-step-ahead -0.0029 -0.0005
```

11.4 Structural Analysis

The general VAR(p) model has many parameters, and they may be difficult to interpret due to complex interactions and feedback between the variables in the model. As a result, the dynamic properties of a VAR(p) are often summarized using various types of *structural analysis*. The three main types of structural analysis summaries are (1) *Granger causality tests*; (2) *impulse response functions*; and (3) *forecast error variance decompositions*. The following sections give brief descriptions of these summary measures.

11.4.1 Granger Causality

One of the main uses of VAR models is forecasting. The structure of the VAR model provides information about a variable's or a group of variables' forecasting ability for other variables. The following intuitive notion of a variable's forecasting ability is due to Granger (1969). If a variable, or group of variables, y_1 is found to be helpful for predicting another variable, or group of variables, y_2 then y_1 is said to *Granger-cause* y_2; otherwise it is said to *fail to Granger-cause* y_2. Formally, y_1 fails to Granger-cause y_2 if for all $s > 0$ the MSE of a forecast of $y_{2,t+s}$ based on $(y_{2,t}, y_{2,t-1}, \ldots)$ is the same as the MSE of a forecast of $y_{2,t+s}$ based on $(y_{2,t}, y_{2,t-1}, \ldots)$ and $(y_{1,t}, y_{1,t-1}, \ldots)$. Clearly, the notion of Granger causality does not imply true causality. It only implies forecasting ability.

Bivariate VAR Models

In a bivariate VAR(p) model for $\mathbf{Y}_t = (y_{1t}, y_{2t})'$, y_2 fails to Granger-cause y_1 if all of the p VAR coefficient matrices $\mathbf{\Pi}_1, \ldots, \mathbf{\Pi}_p$ are lower triangular. That is, the VAR(p) model has the form

$$\begin{pmatrix} y_{1t} \\ y_{2t} \end{pmatrix} = \begin{pmatrix} c_1 \\ c_2 \end{pmatrix} + \begin{pmatrix} \pi_{11}^1 & 0 \\ \pi_{21}^1 & \pi_{22}^1 \end{pmatrix} \begin{pmatrix} y_{1t-1} \\ y_{2t-1} \end{pmatrix} + \cdots + \begin{pmatrix} \pi_{11}^p & 0 \\ \pi_{21}^p & \pi_{22}^p \end{pmatrix} \begin{pmatrix} y_{1t-p} \\ y_{2t-p} \end{pmatrix} + \begin{pmatrix} \varepsilon_{1t} \\ \varepsilon_{2t} \end{pmatrix}$$

so that all of the coefficients on lagged values of y_2 are zero in the equation for y_1. Similarly, y_1 fails to Granger-cause y_2 if all of the coefficients on lagged values of y_1 are zero in the equation for y_2. The p linear coefficient restrictions implied by Granger non-causality may be tested using the Wald statistic (11.5). Notice that if y_2 fails to Granger-cause y_1 and y_1 fails to Granger-cause y_2, then the VAR coefficient matrices $\mathbf{\Pi}_1, \ldots, \mathbf{\Pi}_p$ are diagonal.

General VAR Models

Testing for Granger non-causality in general n variable VAR(p) models follows the same logic used for bivariate models. For example, consider a VAR(p) model with $n = 3$ and $\mathbf{Y}_t = (y_{1t}, y_{2t}, y_{3t})'$. In this model, y_2 does not Granger-cause y_1 if all of the coefficients on lagged values of y_2 are zero in the equation for y_1. Similarly, y_3 does not Granger-cause y_1 if all of the coefficients on lagged values of y_3 are zero in the equation for y_1. These simple linear restrictions may be tested using the Wald statistic (11.5). The reader is encouraged to consult Lütkepohl (1991) or Hamilton (1994) for more details and examples.

Example 68 *Testing for Granger causality in bivariate VAR(2) model for exchange rates*

Consider testing for Granger causality in a bivariate VAR(2) model for $\mathbf{Y}_t = (\Delta s_t, fp_t)'$. Using the notation of (11.2), fp_t does not Granger cause Δs_t if $\pi_{12}^1 = 0$ and $\pi_{12}^2 = 0$. Similarly, Δs_t does not Granger cause fp_t if $\pi_{21}^1 = 0$ and $\pi_{21}^2 = 0$. These hypotheses are easily tested using the Wald statistic (11.5). The restriction matrix $\mathbf{R}$ for the hypothesis that fp_t does not Granger cause Δs_t is

$$\mathbf{R} = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

and the matrix for the hypothesis that Δs_t does not Granger cause fp_t is

$$\mathbf{R} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

The S-PLUS commands to compute and evaluate these Granger causality Wald statistics are

```
> var2.fit = VAR(cbind(dspot,fp)~ar(2),data=uscn.ts)
> # H0: fp does not Granger cause dspot
> R = matrix(c(0,0,1,0,0,0,0,0,0,0,
+ 0,0,0,0,1,0,0,0,0,0),
+ 2,10,byrow=T)
> vecPi = as.vector(coef(var2.fit))
> avar = R%*%vcov(var2.fit)%*%t(R)
> wald = t(R%*%vecPi)%*%solve(avar)%*%(R%*%vecPi)
> wald
         [,1]
[1,] 8.468844
> 1-pchisq(wald,2)
[1] 0.01448818

> R = matrix(c(0,0,0,0,0,0,1,0,0,0,
+ 0,0,0,0,0,0,0,0,1,0),
+ 2,10,byrow=T)
> vecPi = as.vector(coef(var2.fit))
> avar = R%*%vcov(var2.fit)%*%t(R)
> wald = t(R%*%vecPi)%*%solve(avar)%*%(R%*%vecPi)
> wald
      [,1]
[1,] 6.157
> 1-pchisq(wald,2)
[1] 0.04604
```

The p-values for the Wald tests indicate a fairly strong rejection of the null that fp_t does not Granger cause Δs_t but only a weak rejection of the null that Δs_t does not Granger cause fp_t. Hence, lagged values of fp_t appear

to be useful for forecasting future values of Δs_t and lagged values of Δs_t appear to be useful for forecasting future values of fp_t.

11.4.2 Impulse Response Functions

Any covariance stationary VAR(p) process has a Wold representation of the form

$$\mathbf{Y}_t = \boldsymbol{\mu} + \boldsymbol{\varepsilon}_t + \boldsymbol{\Psi}_1 \boldsymbol{\varepsilon}_{t-1} + \boldsymbol{\Psi}_2 \boldsymbol{\varepsilon}_{t-2} + \cdots \tag{11.11}$$

where the $(n \times n)$ moving average matrices $\boldsymbol{\Psi}_s$ are determined recursively using (11.6). It is tempting to interpret the (i,j)-th element, ψ_{ij}^s, of the matrix $\boldsymbol{\Psi}_s$ as the dynamic multiplier or impulse response

$$\frac{\partial y_{i,t+s}}{\partial \varepsilon_{j,t}} = \frac{\partial y_{i,t}}{\partial \varepsilon_{j,t-s}} = \psi_{ij}^s, \ i,j = 1,\ldots,n$$

However, this interpretation is only possible if $var(\boldsymbol{\varepsilon}_t) = \boldsymbol{\Sigma}$ is a diagonal matrix so that the elements of $\boldsymbol{\varepsilon}_t$ are uncorrelated. One way to make the errors uncorrelated is to follow Sims (1980) and estimate the *triangular structural* VAR(p) model

$$\begin{aligned}
y_{1t} &= c_1 + \boldsymbol{\gamma}_{11}'\mathbf{Y}_{t-1} + \cdots + \boldsymbol{\gamma}_{1p}'\mathbf{Y}_{t-p} + \eta_{1t} \\
y_{2t} &= c_1 + \beta_{21} y_{1t} + \boldsymbol{\gamma}_{21}'\mathbf{Y}_{t-1} + \cdots + \boldsymbol{\gamma}_{2p}'\mathbf{Y}_{t-p} + \eta_{2t} \\
y_{3t} &= c_1 + \beta_{31} y_{1t} + \beta_{32} y_{2t} + \boldsymbol{\gamma}_{31}'\mathbf{Y}_{t-1} + \cdots + \boldsymbol{\gamma}_{3p}'\mathbf{Y}_{t-p} + \eta_{3t} \\
&\vdots \\
y_{nt} &= c_1 + \beta_{n1} y_{1t} + \cdots + \beta_{n,n-1} y_{n-1,t} + \boldsymbol{\gamma}_{n1}'\mathbf{Y}_{t-1} + \cdots + \boldsymbol{\gamma}_{np}'\mathbf{Y}_{t-p} + \eta_{nt}
\end{aligned} \tag{11.12}$$

In matrix form, the triangular structural VAR(p) model is

$$\mathbf{B}\mathbf{Y}_t = \mathbf{c} + \boldsymbol{\Gamma}_1 \mathbf{Y}_{t-1} + \boldsymbol{\Gamma}_2 \mathbf{Y}_{t-2} + \cdots + \boldsymbol{\Gamma}_p \mathbf{Y}_{t-p} + \boldsymbol{\eta}_t \tag{11.13}$$

where

$$\mathbf{B} = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ -\beta_{21} & 1 & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots \\ -\beta_{n1} & -\beta_{n2} & \cdots & 1 \end{pmatrix} \tag{11.14}$$

is a lower triangular matrix with $1's$ along the diagonal. The algebra of least squares will ensure that the estimated covariance matrix of the error vector $\boldsymbol{\eta}_t$ is diagonal. The uncorrelated/orthogonal errors $\boldsymbol{\eta}_t$ are referred to as *structural* errors.

The triangular structural model (11.12) imposes the *recursive causal ordering*

$$y_1 \rightarrow y_2 \rightarrow \cdots \rightarrow y_n \tag{11.15}$$

The ordering (11.15) means that the contemporaneous values of the variables to the left of the arrow $\rightarrow$ affect the contemporaneous values of the variables to the right of the arrow but not vice-versa. These contemporaneous effects are captured by the coefficients β_{ij} in (11.12). For example, the ordering $y_1 \rightarrow y_2 \rightarrow y_3$ imposes the restrictions: y_{1t} affects y_{2t} and y_{3t} but y_{2t} and y_{3t} do not affect y_1; y_{2t} affects y_{3t} but y_{3t} does not affect y_{2t}. Similarly, the ordering $y_2 \rightarrow y_3 \rightarrow y_1$ imposes the restrictions: y_{2t} affects y_{3t} and y_{1t} but y_{3t} and y_{1t} do not affect y_2; y_{3t} affects y_{1t} but y_{1t} does not affect y_{3t}. For a VAR(p) with n variables there are $n!$ possible recursive causal orderings. Which ordering to use in practice depends on the context and whether prior theory can be used to justify a particular ordering. Results from alternative orderings can always be compared to determine the sensitivity of results to the imposed ordering.

Once a recursive ordering has been established, the Wold representation of $\mathbf{Y}_t$ based on the orthogonal errors $\boldsymbol{\eta}_t$ is given by

$$\mathbf{Y}_t = \boldsymbol{\mu} + \boldsymbol{\Theta}_0\boldsymbol{\eta}_t + \boldsymbol{\Theta}_1\boldsymbol{\eta}_{t-1} + \boldsymbol{\Theta}_2\boldsymbol{\eta}_{t-2} + \cdots \tag{11.16}$$

where $\boldsymbol{\Theta}_0 = \mathbf{B}^{-1}$ is a lower triangular matrix. The impulse responses to the orthogonal shocks η_{jt} are

$$\frac{\partial y_{i,t+s}}{\partial \eta_{j,t}} = \frac{\partial y_{i,t}}{\partial \eta_{j,t-s}} = \theta_{ij}^s, \; i,j = 1,\ldots,n; s > 0 \tag{11.17}$$

where θ_{ij}^s is the (i,j) th element of $\boldsymbol{\Theta}_s$. A plot of θ_{ij}^s against s is called the *orthogonal impulse response function* (IRF) of y_i with respect to η_j. With n variables there are n^2 possible impulse response functions.

In practice, the orthogonal IRF (11.17) based on the triangular VAR(p) (11.12) may be computed directly from the parameters of the non triangular VAR(p) (11.1) as follows. First, decompose the residual covariance matrix $\boldsymbol{\Sigma}$ as

$$\boldsymbol{\Sigma} = \mathbf{A}\mathbf{D}\mathbf{A}'$$

where $\mathbf{A}$ is an invertible lower triangular matrix with $1's$ along the diagonal and $\mathbf{D}$ is a diagonal matrix with positive diagonal elements. Next, define the structural errors as

$$\boldsymbol{\eta}_t = \mathbf{A}^{-1}\boldsymbol{\varepsilon}_t$$

These structural errors are orthogonal by construction since $var(\boldsymbol{\eta}_t) = \mathbf{A}^{-1}\boldsymbol{\Sigma}\mathbf{A}^{-1\prime} = \mathbf{A}^{-1}\mathbf{A}\mathbf{D}\mathbf{A}'\mathbf{A}^{-1\prime} = \mathbf{D}$. Finally, re-express the Wold representation (11.11) as

$$\begin{aligned} \mathbf{Y}_t &= \boldsymbol{\mu} + \mathbf{A}\mathbf{A}^{-1}\boldsymbol{\varepsilon}_t + \boldsymbol{\Psi}_1\mathbf{A}\mathbf{A}^{-1}\boldsymbol{\varepsilon}_{t-1} + \boldsymbol{\Psi}_2\mathbf{A}\mathbf{A}^{-1}\boldsymbol{\varepsilon}_{t-2} + \cdots \\ &= \boldsymbol{\mu} + \boldsymbol{\Theta}_0\boldsymbol{\eta}_t + \boldsymbol{\Theta}_1\boldsymbol{\eta}_{t-1} + \boldsymbol{\Theta}_2\boldsymbol{\eta}_{t-2} + \cdots \end{aligned}$$

where $\boldsymbol{\Theta}_j = \boldsymbol{\Psi}_j\mathbf{A}$. Notice that the structural B matrix in (11.13) is equal to $\mathbf{A}^{-1}$.

Computing the Orthogonal Impulse Response Function Using the `S+FinMetrics` Function `impRes`

The orthogonal impulse response function (11.17) from a triangular structural VAR model (11.13) may be computed using the `S+FinMetrics` function `impRes`. The function `impRes` has arguments

```
> args(impRes)
function(x, period = NULL, std.err = "none", plot = F,
unbiased = T, order = NULL, ...)
```

where `x` is an object of class "`VAR`" and `period` specifies the number of responses to compute. By default, no standard errors for the responses are computed. To compute asymptotic standard errors for the responses, specify `std.err="asymptotic"`. To create a panel plot of all the response functions, specify `plot=T`. The default recursive causal ordering is based on the ordering of the variables in $\mathbf{Y}_t$ when the VAR model is fit. The optional argument `order` may be used to specify a different recursive causal ordering for the computation of the impulse responses. The argument `order` accepts a character vector of variable names whose order defines the recursive causal ordering. The output of `impRes` is an object of class "`impDecomp`" for which there are `print`, `summary` and `plot` methods. The following example illustrates the use of `impRes`.

Example 69 *IRF from VAR(1) for exchange rates*

Consider again the VAR(1) model for $\mathbf{Y}_t = (\Delta s_t, fp_t)'$. For the impulse response analysis, the initial ordering of the variables imposes the assumption that structural shocks to fp_t have no contemporaneous effect on Δs_t but structural shocks to Δs_t do have a contemporaneous effect on fp_t. To compute the four impulse response functions

$$\frac{\partial \Delta s_{t+h}}{\partial \eta_{1t}}, \frac{\partial \Delta s_{t+h}}{\partial \eta_{2t}}, \frac{\partial fp_{t+h}}{\partial \eta_{1t}}, \frac{\partial fp_{t+h}}{\partial \eta_{2t}}$$

for $h = 1, \ldots, 12$ we use `S+FinMetrics` function `impRes`. The first twelve impulse responses from the VAR(1) model for exchange rates are computed using

```
> uscn.irf = impRes(var1.fit, period=12, std.err="asymptotic")
```

The `print` method shows the impulse response values without standard errors:

```
> uscn.irf

Impulse Response Function:
(with responses in rows, and innovations in columns)
```

```
, , lag.0
        dspot      fp
dspot  0.0136  0.0000
   fp  0.0000  0.0009

, , lag.1
        dspot      fp
dspot -0.0018 -0.0013
   fp  0.0001  0.0007

, , lag.2
        dspot      fp
dspot  0.0000 -0.0009
   fp  0.0001  0.0006

...

, , lag.11
        dspot      fp
dspot  0.0000 -0.0001
   fp  0.0000  0.0001
```

The `summary` method will display the responses with standard errors and t-statistics. The `plot` method will produce a four panel Trellis graphics plot of the impulse responses

```
> plot(uscn.irf)
```

A plot of the impulse responses can also be created in the initial call to `impRes` by using the optional argument `plot=T`.

Figure 11.6 shows the impulse response functions along with asymptotic standard errors. The top row shows the responses of Δs_t to the structural shocks, and the bottom row shows the responses of fp_t to the structural shocks. In response to the first structural shock, η_{1t}, Δs_t initially increases but then drops quickly to zero after 2 months. Similarly, fp_t initially increases, reaches its peak response in 2 months and then gradually drops off to zero after about a year. In response to the second shock, η_{2t}, by assumption Δs_t has no initial response. At one month, a sharp drop occurs in Δs_t followed by a gradual return to zero after about a year. In contrast, fp_t initially increases and then gradually drops to zero after about a year.

The orthogonal impulse responses in Figure 11.6 are based on the recursive causal ordering $\Delta s_t \rightarrow fp_t$. It must always be kept in mind that this ordering identifies the orthogonal structural shocks η_{1t} and η_{2t}. If the ordering is reversed, then a different set of structural shocks will be identified, and these may give very different impulse response functions. To compute

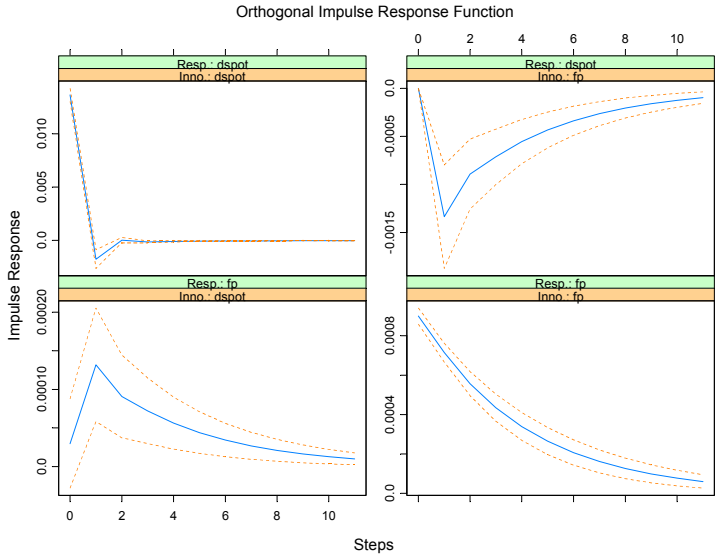

FIGURE 11.6. Impulse response function from VAR(1) model fit to US/CN exchange rate data with Δs_t ordered first.

the orthogonal impulse responses using the alternative ordering $fp_t \rightarrow \Delta s_t$ specify `order=c("fp","dspot")` in the call to `impRes`:

```
> uscn.irf2 = impRes(var1.fit,period=12,std.err="asymptotic",
+ order=c("fp","dspot"),plot=T)
```

These impulse responses are presented in Figure 11.7 and are almost identical to those computed using the ordering $\Delta s_t \rightarrow fp_t$. The reason for this response is that the reduced form VAR residuals $\hat{\varepsilon}_{1t}$ and $\hat{\varepsilon}_{2t}$ are almost uncorrelated. To see this, the residual correlation matrix may be computed using

```
> sd.vals = sqrt(diag(var1.fit$Sigma))
> cor.mat = var1.fit$Sigma/outer(sd.vals,sd.vals)
> cor.mat
         dspot       fp
dspot 1.000000 0.033048
   fp 0.033048 1.000000
```

Because of the near orthogonality in the reduced form VAR errors, the error in the Δs_t equation may be interpreted as an orthogonal shock to the exchange rate and the error in the fp_t equation may be interpreted as an orthogonal shock to the forward premium.

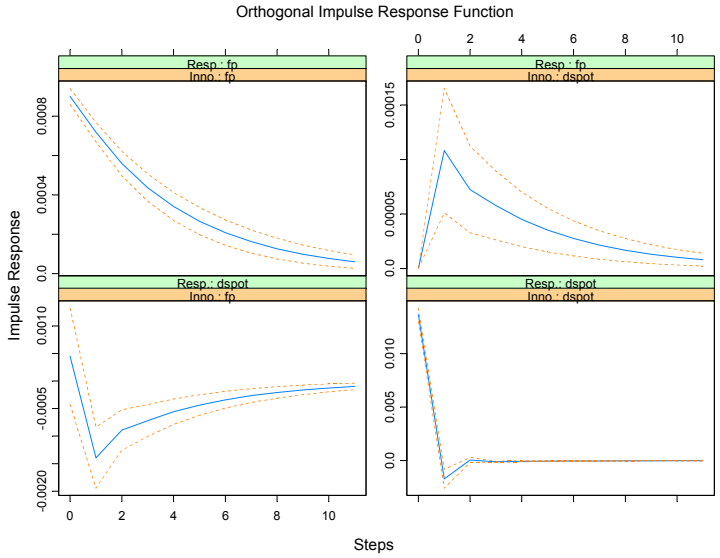

FIGURE 11.7. Impulse response function from VAR(1) model fit to US/CN exchange rate with fp_t ordered first.

11.4.3 Forecast Error Variance Decompositions

The *forecast error variance decomposition* (FEVD) answers the question: what portion of the variance of the forecast error in predicting $y_{i,T+h}$ is due to the structural shock η_j? Using the orthogonal shocks $\boldsymbol{\eta}_t$ the h-step ahead forecast error vector, with known VAR coefficients, may be expressed as

$$\mathbf{Y}_{T+h} - \mathbf{Y}_{T+h|T} = \sum_{s=0}^{h-1} \boldsymbol{\Theta}_s \boldsymbol{\eta}_{T+h-s}$$

For a particular variable $y_{i,T+h}$, this forecast error has the form

$$y_{i,T+h} - y_{i,T+h|T} = \sum_{s=0}^{h-1} \theta_{i1}^s \eta_{1,T+h-s} + \cdots + \sum_{s=0}^{h-1} \theta_{in}^s \eta_{n,T+h-s}$$

Since the structural errors are orthogonal, the variance of the h-step forecast error is

$$var(y_{i,T+h} - y_{i,T+h|T}) = \sigma_{\eta_1}^2 \sum_{s=0}^{h-1} (\theta_{i1}^s)^2 + \cdots + \sigma_{\eta_n}^2 \sum_{s=0}^{h-1} (\theta_{in}^s)^2$$

where $\sigma^2_{\eta j} = var(\eta_{jt})$. The portion of $var(y_{i,T+h} - y_{i,T+h|T})$ due to shock η_j is then

$$FEVD_{i,j}(h) = \frac{\sigma^2_{\eta_j} \sum_{s=0}^{h-1} \left(\theta^s_{ij}\right)^2}{\sigma^2_{\eta_1} \sum_{s=0}^{h-1} \left(\theta^s_{i1}\right)^2 + \cdots + \sigma^2_{\eta_n} \sum_{s=0}^{h-1} \left(\theta^s_{in}\right)^2}, \; i,j = 1,\ldots,n \tag{11.18}$$

In a VAR with n variables there will be n^2 $FEVD_{i,j}(h)$ values. It must be kept in mind that the FEVD in (11.18) depends on the recursive causal ordering used to identify the structural shocks $\boldsymbol{\eta}_t$ and is not unique. Different causal orderings will produce different FEVD values.

Computing the FEVD Using the `S+FinMetrics` Function `fevDec`

Once a VAR model has been fit, the `S+FinMetrics` function `fevDec` may be used to compute the orthogonal FEVD. The function `fevDec` has arguments

```
> args(fevDec)
function(x, period = NULL, std.err = "none", plot = F,
unbiased = F, order = NULL, ...)
```

where `x` is an object of class "`VAR`" and `period` specifies the number of responses to compute. By default, no standard errors for the responses are computed and no plot is created. To compute asymptotic standard errors for the responses, specify `std.err="asymptotic"` and to plot the decompositions, specify `plot=T`. The default recursive causal ordering is based on the ordering of the variables in $\mathbf{Y}_t$ when the VAR model is fit. The optional argument `order` may be used to specify a different recursive causal ordering for the computation of the FEVD. The argument `order` accepts a text string vector of variable names whose order defines the recursive causal ordering. The output of `fevDec` is an object of class "`impDecomp`" for which there are `print`, `summary` and `plot` methods. The use of `fevDec` is illustrated with the following example.

Example 70 *FEVD from VAR(1) for exchange rates*

The orthogonal FEVD of the forecast errors from the VAR(1) model fit to the US/CN exchange rate data using the recursive causal ordering $\Delta s_t \rightarrow fp_t$ is computed using

```
> uscn.fevd = fevDec(var1.fit,period=12,
+ std.err="asymptotic")
> uscn.fevd

Forecast Error Variance Decomposition:
(with responses in rows, and innovations in columns)
```

```
, , 1-step-ahead
       dspot     fp
dspot 1.0000 0.0000
   fp 0.0011 0.9989

, , 2-step-ahead
       dspot     fp
dspot 0.9907 0.0093
   fp 0.0136 0.9864

...

, , 12-step-ahead
       dspot     fp
dspot 0.9800 0.0200
   fp 0.0184 0.9816
```

The `summary` method adds standard errors to the above output if they are computed in the call to `fevDec`. The `plot` method produces a four panel Trellis graphics plot of the decompositions:

```
> plot(uscn.fevd)
```

The FEVDs in Figure 11.8 show that most of the variance of the forecast errors for Δs_{t+s} at all horizons s is due to the orthogonal Δs_t innovations. Similarly, most of the variance of the forecast errors for fp_{t+s} is due to the orthogonal fp_t innovations.

The FEVDs using the alternative recursive causal ordering $fp_t \rightarrow \Delta s_t$ are computed using

```
> uscn.fevd2 = fevDec(var1.fit,period=12,
+ std.err="asymptotic",order=c("fp","dspot"),plot=T)
```

and are illustrated in Figure 11.9. Since the residual covariance matrix is almost diagonal (see analysis of IRF above), the FEVDs computed using the alternative ordering are almost identical to those computed with the initial ordering.

11.5 An Extended Example

In this example the causal relations and dynamic interactions among monthly real stock returns, real interest rates, real industrial production growth and the inflation rate is investigated using a VAR model. The analysis is similar to that of Lee (1992). The variables are in the `S+FinMetrics` "`timeSeries`" object `varex.ts`

```
> colIds(varex.ts)
```

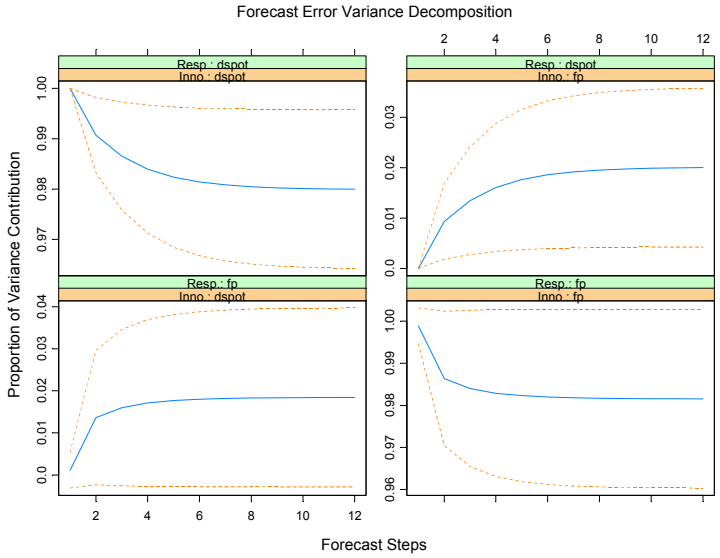

FIGURE 11.8. Orthogonal FEVDs computed from VAR(1) model fit to US/CN exchange rate data using the recursive causal ordering with Δs_t first.

```
[1] "MARKET.REAL" "RF.REAL" "INF" "IPG"
```

Details about the data are in the documentation slot of `varex.ts`

```
> varex.ts@documentation
```

To be comparable to the results in Lee (1992), the analysis is conducted over the postwar period January 1947 through December 1987

```
> smpl = (positions(varex.ts) >= timeDate("1/1/1947") &
+ positions(varex.ts) < timeDate("1/1/1988"))
```

The data over this period is displayed in Figure 11.10. All variables appear to be $I(0)$, but the real T-bill rate and the inflation rate appear to be highly persistent.

To begin the analysis, autocorrelations and cross correlations at leads and lags are computed using

```
> varex.acf = acf(varex.ts[smpl,])
```

and are illustrated in Figure 11.11. The real return on the market shows a significant positive first lag autocorrelation, and inflation appears to lead the real market return with a negative sign. The real T-bill rate is highly positively autocorrelated, and inflation appears to lead the real T-bill rate strongly with a negative sign. Inflation is also highly positively autocorrelated and, interestingly, the real T-bill rate appears to lead inflation with

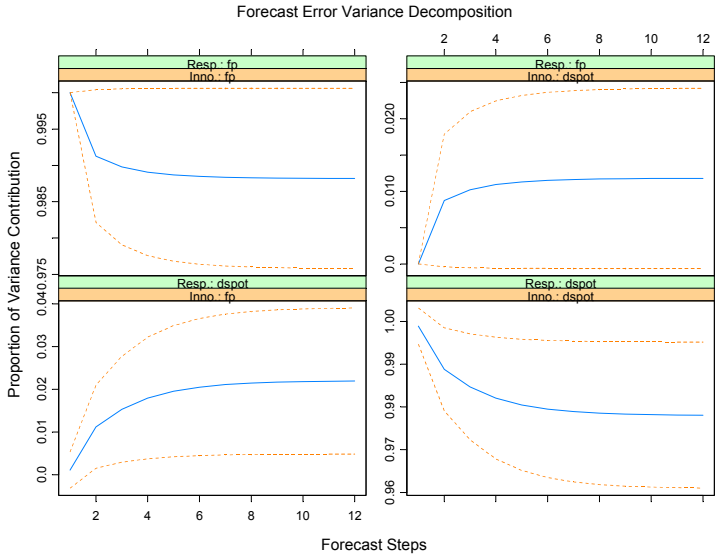

FIGURE 11.9. Orthogonal FEVDs from VAR(1) model fit to US/CN exchange rate data using recursive causal ordering with fp_t first.

a positive sign. Finally, industrial production growth is slightly positively autocorrelated, and the real market return appears to lead industrial production growth with a positive sign.

The VAR(p) model is fit with the lag length selected by minimizing the AIC and a maximum lag length of 6 months:

```
> varAIC.fit = VAR(varex.ts,max.ar=6,criterion="AIC",
+ start="Jan 1947",end="Dec 1987",
+ in.format="%m %Y")
```

The lag length selected by minimizing AIC is $p = 2$:

```
> varAIC.fit$info
     ar(1)  ar(2)  ar(3)  ar(4)  ar(5)  ar(6)
AIC -14832 -14863 -14853 -14861 -14855 -14862
> varAIC.fit$ar.order
[1] 2
```

The results of the VAR(2) fit are

```
> summary(varAIC.out)

Call:
VAR(data = varex.ts, start = "Jan 1947", end = "Dec 1987",
```

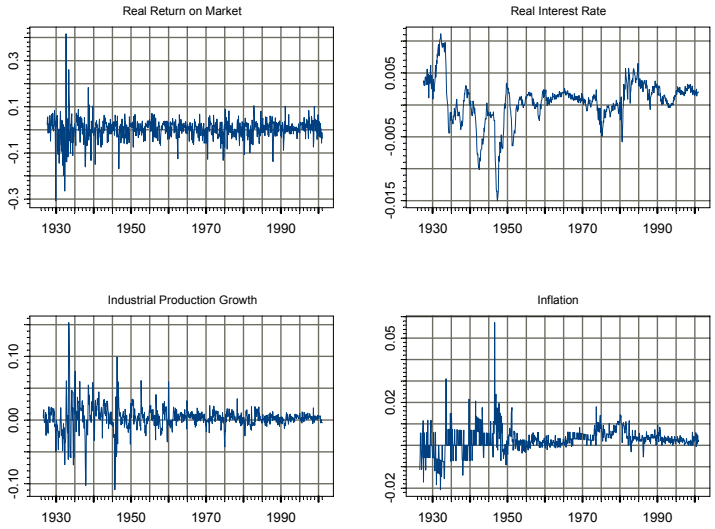

FIGURE 11.10. Monthly data on stock returns, interest rates, output growth and inflation.

```
max.ar = 6, criterion = "AIC", in.format = "%m %Y")

Coefficients:
                 MARKET.REAL  RF.REAL      INF      IPG
     (Intercept)    0.0074      0.0002   0.0010   0.0019
       (std.err)    0.0023      0.0001   0.0002   0.0007
        (t.stat)    3.1490      4.6400   4.6669   2.5819

MARKET.REAL.lag1    0.2450      0.0001   0.0072   0.0280
       (std.err)    0.0470      0.0011   0.0042   0.0146
        (t.stat)    5.2082      0.0483   1.7092   1.9148

    RF.REAL.lag1    0.8146      0.8790   0.5538   0.3772
       (std.err)    2.0648      0.0470   0.1854   0.6419
        (t.stat)    0.3945     18.6861   2.9867   0.5877

        INF.lag1   -1.5020     -0.0710   0.4616  -0.0722
       (std.err)    0.4932      0.0112   0.0443   0.1533
        (t.stat)   -3.0451     -6.3147  10.4227  -0.4710

                 MARKET.REAL  RF.REAL      INF      IPG
        IPG.lag1   -0.0003      0.0031  -0.0143   0.3454
```

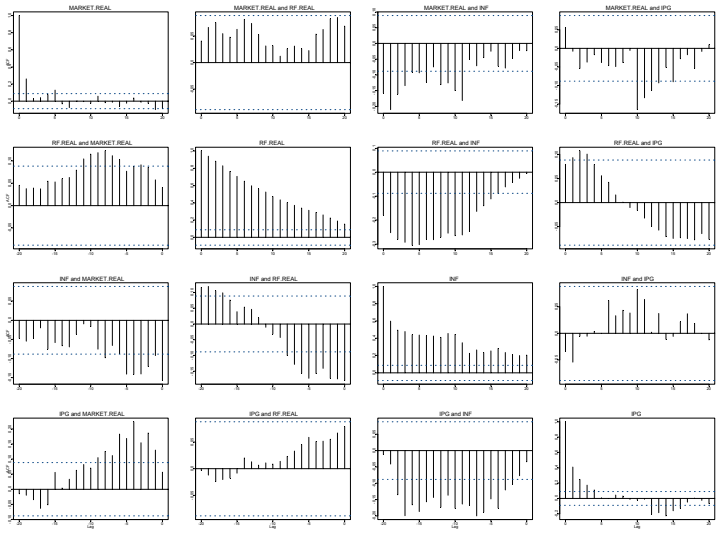

FIGURE 11.11. Autocorrelations and cross correlations at leads and lags of data in VAR model.

```
       (std.err)   0.1452      0.0033   0.0130   0.0452
        (t.stat)  -0.0018      0.9252  -1.0993   7.6501

MARKET.REAL.lag2  -0.0500      0.0022  -0.0066   0.0395
       (std.err)   0.0466      0.0011   0.0042   0.0145
        (t.stat)  -1.0727      2.0592  -1.5816   2.7276

    RF.REAL.lag2  -0.3481      0.0393  -0.5855  -0.3289
       (std.err)   1.9845      0.0452   0.1782   0.6169
        (t.stat)  -0.1754      0.8699  -3.2859  -0.5331

        INF.lag2  -0.0602      0.0079   0.2476  -0.0370
       (std.err)   0.5305      0.0121   0.0476   0.1649
        (t.stat)  -0.1135      0.6517   5.1964  -0.2245

          MARKET.REAL  RF.REAL      INF      IPG
 IPG.lag2  -0.1919      0.0028   0.0154   0.0941
(std.err)   0.1443      0.0033   0.0130   0.0449
 (t.stat)  -1.3297      0.8432   1.1868   2.0968

Regression Diagnostics:
               MARKET.REAL RF.REAL    INF    IPG
```

```
     R-squared 0.1031      0.9299  0.4109 0.2037
Adj. R-squared 0.0882      0.9287  0.4011 0.1905
  Resid. Scale 0.0334      0.0008  0.0030 0.0104

Information Criteria:
   logL    AIC    BIC     HQ
   7503 -14935 -14784 -14875

                   total residual
Degree of freedom:   489      480
Time period: from Mar 1947 to Nov 1987
```

The signs of the statistically significant coefficient estimates corroborate the informal analysis of the multivariate autocorrelations and cross lag autocorrelations. In particular, the real market return is positively related to its own lag but negatively related to the first lag of inflation. The real T-bill rate is positively related to its own lag, negatively related to the first lag of inflation, and positively related to the first lag of the real market return. Industrial production growth is positively related to its own lag and positively related to the first two lags of the real market return. Judging from the coefficients it appears that inflation Granger causes the real market return and the real T-bill rate, the real T-bill rate Granger causes inflation, and the real market return Granger causes the real T-bill rate and industrial production growth. These observations are confirmed with formal tests for Granger non-causality. For example, the Wald statistic for testing the null hypothesis that the real market return does not Granger-cause industrial production growth is

```
> bigR = matrix(0,2,36)
> bigR[1,29]=bigR[2,33]=1
> vecPi = as.vector(coef(varAIC.fit))
> avar = bigR%*%vcov(varAIC.fit)%*%t(bigR)
> wald = t(bigR%*%vecPi)%*%solve(avar)%*%(bigR%*%vecPi)
> as.numeric(wald)
[1] 13.82
> 1-pchisq(wald,2)
[1] 0.0009969
```

The 24-period IRF using the recursive causal ordering `MARKET.REAL` → `RF.REAL` → `IPG` → `INF` is computed using

```
> varAIC.irf = impRes(varAIC.fit,period=24,
+ order=c("MARKET.REAL","RF.REAL","IPG","INF"),
+ std.err="asymptotic",plot=T)
```

and is illustrated in Figure 11.12. The responses of `MARKET.REAL` to unexpected orthogonal shocks to the other variables are given in the first row of

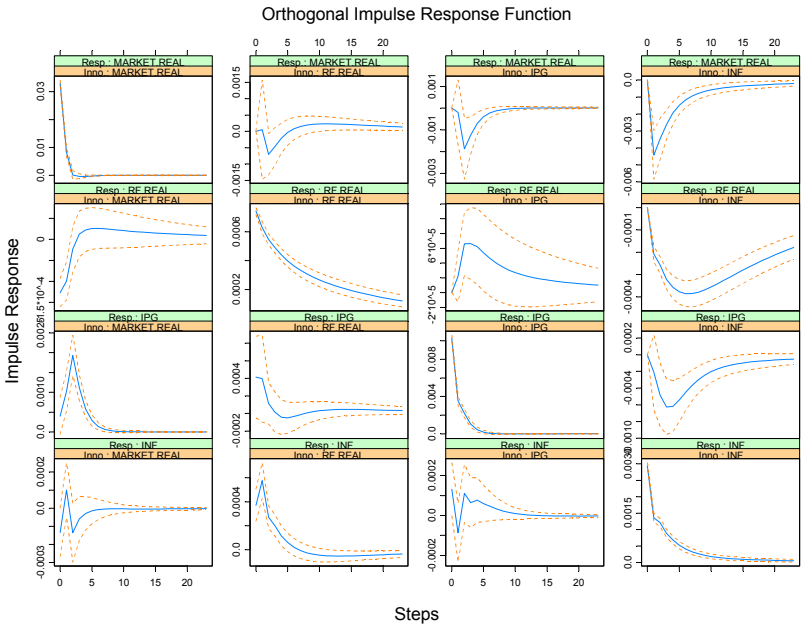

FIGURE 11.12. IRF using the recursive causal ordering `MARKET.REAL` → `RF.REAL` → `IPG` → `INF`.

the figure. Most notable is the strong negative response of `MARKET.REAL` to an unexpected increase in inflation. Notice that it takes about ten months for the effect of the shock to dissipate. The responses of `RF.REAL` to the orthogonal shocks is given in the second row of the figure. `RF.REAL` also reacts negatively to an inflation shock and the effect of the shock is felt for about two years. The responses of `IPG` to the orthogonal shocks is given in the third row of the figure. Industrial production growth responds positively to an unexpected shock to `MARKET.REAL` and negatively to shocks to `RF.REAL` and `INF`. These effects, however, are generally short term. Finally, the fourth row gives the responses of `INF` to the orthogonal shocks. Inflation responds positively to a shock to the real T-bill rate, but this effect is short-lived.

The 24 month FEVD computed using the recursive causal ordering as specified by `MARKET.REAL` → `RF.REAL` → `IPG` → `INF`,

```
> varAIC.fevd = fevDec(varAIC.out,period=24,
> order=c("MARKET.REAL","RF.REAL","IPG","INF"),
> std.err="asymptotic",plot=T)
```

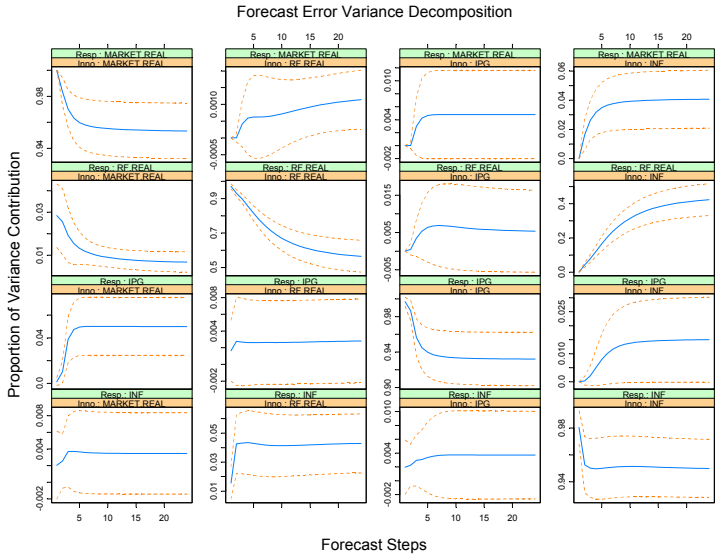

FIGURE 11.13. FEVDs using the recursive causal ordering `MARKET.REAL` → `RF.REAL` → `IPG` → `INF`.

is illustrated in Figure 11.13. The first row gives the variance decompositions for `MARKET.REAL` and shows that most of the variance of the forecast errors is due to own shocks. The second row gives the decompositions for `RF.REAL`. At short horizons, most of the variance is attributable to own shocks but at long horizons inflation shocks account for almost half the variance. The third row gives the variance decompositions for `IPG`. Most of the variance is due to own shocks and a small fraction is due to `MARKET.REAL` shocks. Finally, the fourth row shows that the forecast error variance of `INF` is due mostly to its own shocks.

The IRFs and FEVDs computed above depend on the imposed recursive causal ordering. However, in this example, the ordering of the variables will have little effect on the IRFs and FEVDs because the errors in the reduced form VAR are nearly uncorrelated:

```
> sd.vals = sqrt(diag(varAIC.out$Sigma))
> cor.mat = varAIC.out$Sigma/outer(sd.vals,sd.vals)
> cor.mat
             MARKET.REAL  RF.REAL      INF     IPG
MARKET.REAL      1.00000 -0.16855 -0.04518 0.03916
    RF.REAL     -0.16855  1.00000  0.13046 0.03318
        INF     -0.04518  0.13046  1.00000 0.04732
        IPG      0.03916  0.03318  0.04732 1.00000
```

11.6 Bayesian Vector Autoregression

VAR models with many variables and long lags contain many parameters. Unrestricted estimation of these models reqires lots of data and often the estimated parameters are not very precise, the results are hard to interpret, and forecasts may appear more precise than they really are because standard error bands do not account for parameter uncertainty. The estimates and forecasts can be improved if one has prior information about the structure of the model or the possible values of the parameters or functions of the parameters. In a classical framework, it is difficult to incorporate non-sample information into the estimation. Nonsample information is easy to incorporate in a Bayesian framework. A Bayesian framework also naturally incorporates parameter uncertainty into common measures of precision. This section briefly describes the Bayesian VAR modeling tools in `S+FinMetrics` and illustrates these tools with an example. Details of underlying Bayesian methods are given in Sims and Zha (1998) and Zha (1998).

11.6.1 An Example of a Bayesian VAR Model

`S+FinMetrics` comes with a "`timeSeries`" object `policy.dat`, which contains six U.S. macroeconomic variables:

```
> colIds(policy.dat)
[1] "CP"  "M2"  "FFR" "GDP" "CPI" "U"
```

which represent IMF's index of world commodity prices, M2 money stock, federal funds rate, real GDP, consumer price index for urban consumers, and civilian unemployment rate. The data set contains monthly observations from January 1959 to March 1998. Tao Zha and his co-authors have analyzed this data set in a number of papers, for example see Zha (1998). To use the same time period as in Zha (1998), create a subset of the data:

```
> zpolicy.dat = policy.dat[1:264,]
> zpolicy.mat = as.matrix(seriesData(zpolicy.dat))
```

which contains monthly observations from January 1959 to December 1980.

Estimating a Bayesian VAR Model

To estimate a Bayesian vector autoregression model, use the `S+FinMetrics` function `BVAR`. For macroeconomic modeling, it is usually found that many trending macroeconomic variables have a unit root, and in some cases, they may also have a cointegrating relationship (as described in the next chapter). To incorporate these types of prior beliefs into the model, use the `unit.root.dummy` and `coint.dummy` optional arguments to the `BVAR`

function, which add some dummy observations to the beginning of the data to reflect these beliefs:

```
> zpolicy.bar13 = BVAR(zpolicy.mat~ar(13), unit.root=T,
+ coint=T)
> class(zpolicy.bar13)
[1] "BVAR"
```

The returned object is of class "BVAR", which inherits from "VAR", so many method functions for "VAR" objects work similarly for "BVAR" objects, such as the extractor functions, impulse response functions, and forecast error variance decomposition functions.

The Bayesian VAR models are controlled through a set of hyper parameters, which can be specified using the optional argument `control`, which is usually a list returned by the function `BVAR.control`. For example, the tightness of the belief in the unit root prior and cointegration prior is specified by `mu5` and `mu6`, respectively. To see what default values are used for these hyper parameters, use

```
> args(BVAR.control)
function(L0 = 0.9, L1 = 0.1, L2 = 1, L3 = 1, L4 = 0.05,
        mu5 = 5, mu6 = 5)
```

For the meanings of these hyper parameters, see the online help file for `BVAR.control`.

Adding Exogenous Variables to the Model

Other exogenous variables can be added to the estimation formula, just as for `OLS` and `VAR` functions. The `BVAR` function and related functions will automatically take that into consideration and return the coefficient estimates for those variables.

Unconditional Forecasts

To forecast from a fitted Bayesian VAR model, use the generic `predict` function, which automatically calls the method function `predict.BVAR` for an object inheriting from class "BVAR". For example, to compute 12-step ahead forecasts use

```
> zpolicy.bpred = predict(zpolicy.bar13,n.predict=12)
> class(zpolicy.bpred)
[1] "forecast"
> names(zpolicy.bpred)
[1] "values"  "std.err" "draws"
> zpolicy.bpred

Predicted Values:
```

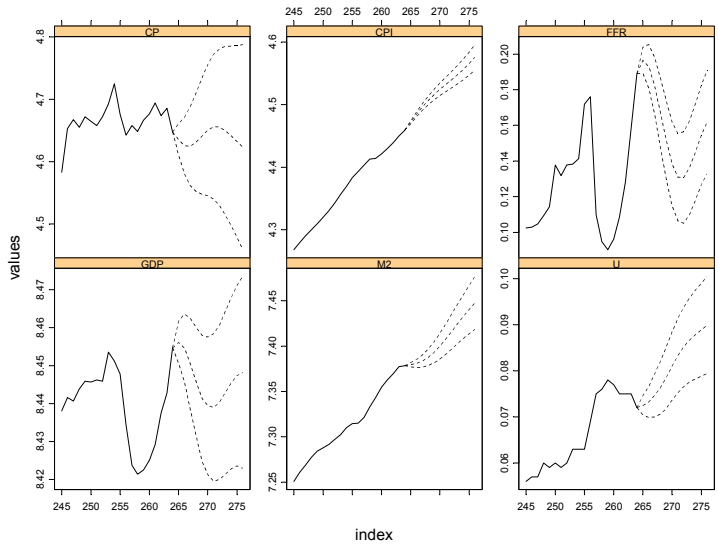

FIGURE 11.14. Forecasts from Bayesian VAR model.

```
                   CP     M2    FFR    GDP    CPI      U
 1-step-ahead 4.6354 7.3794 0.1964 8.4561 4.4714 0.0725
 2-step-ahead 4.6257 7.3808 0.1930 8.4546 4.4842 0.0732
 3-step-ahead 4.6247 7.3834 0.1823 8.4505 4.4960 0.0746
 4-step-ahead 4.6310 7.3876 0.1670 8.4458 4.5065 0.0763
 5-step-ahead 4.6409 7.3931 0.1515 8.4414 4.5160 0.0785
 6-step-ahead 4.6503 7.3998 0.1384 8.4394 4.5244 0.0810
 7-step-ahead 4.6561 7.4075 0.1309 8.4390 4.5321 0.0833
 8-step-ahead 4.6552 7.4159 0.1307 8.4403 4.5397 0.0852
 9-step-ahead 4.6496 7.4242 0.1362 8.4428 4.5475 0.0867
10-step-ahead 4.6415 7.4323 0.1451 8.4453 4.5561 0.0879
11-step-ahead 4.6321 7.4402 0.1546 8.4473 4.5655 0.0889
12-step-ahead 4.6232 7.4476 0.1618 8.4482 4.5753 0.0899
```

The forecasts can also be plotted along with the original data using

```
> plot(zpolicy.bpred, zpolicy.mat, n.old=20)
```

The resulting plot is shown in Figure 11.14. The Bayesian forecasts usually have wider error bands than classical forecasts, because they take into account the uncertainty in the coefficient estimates. To ignore the uncertainty in coefficient estimates, one can call the classical VAR `predict` method function, `predict.VAR`, directly instead of the generic `predict` function.

The forecasts from Bayesian VAR models are of class "`forecast`", and are computed using Monte Carlo integration. By default, 1000 simulation draws are used. To change the number of simulation draws and random seed, specify the `n.sim` and `seed` optional arguments, respectively. For forecasts from Bayesian VAR models, there is one more component in the returned object: `draws`, which contains all the simulated forecasts. This can be used to assess other statistical properties of the forecasts.

11.6.2 Conditional Forecasts

As mentioned earlier, conditional forecasts from classical VAR models ignore the uncertainty in estimated coefficients. In contrast, conditional forecasts from Bayesian VAR models take into account the uncertainty associated with estimated coefficients. To perform conditional forecasts from a fitted Bayesian VAR model, use the generic `cpredict` function. For example, if it is known that `FFR` in January 1981 is between 0.185 and 0.195, one can incorporate this (soft condition) information into the forecasts using:

```
> zpolicy.spred = cpredict(zpolicy.bar13, 12, steps=1,
+ variables="FFR", upper=0.195, lower=0.185)
> zpolicy.spred

Predicted Values:

                  CP     M2    FFR    GDP    CPI      U
 1-step-ahead 4.6374 7.3797 0.1910 8.4554 4.4714 0.0729
 2-step-ahead 4.6286 7.3816 0.1855 8.4540 4.4840 0.0736
 3-step-ahead 4.6279 7.3850 0.1743 8.4498 4.4954 0.0752
 4-step-ahead 4.6349 7.3899 0.1587 8.4452 4.5057 0.0768
 5-step-ahead 4.6447 7.3960 0.1443 8.4414 4.5149 0.0791
 6-step-ahead 4.6525 7.4033 0.1324 8.4406 4.5231 0.0814
 7-step-ahead 4.6549 7.4114 0.1270 8.4412 4.5307 0.0835
 8-step-ahead 4.6523 7.4201 0.1283 8.4428 4.5383 0.0851
 9-step-ahead 4.6453 7.4284 0.1349 8.4457 4.5461 0.0864
10-step-ahead 4.6389 7.4365 0.1432 8.4482 4.5547 0.0876
11-step-ahead 4.6317 7.4444 0.1516 8.4501 4.5641 0.0885
12-step-ahead 4.6264 7.4519 0.1572 8.4511 4.5741 0.0896
```

For conditional forecasts with soft conditions, a Monte Carlo integration with acceptance/rejection method is used. By default, 1000 simulation draws are used. However, it may occur that only a small number of draws satisfy the soft conditions if the intervals are very tight. To see how many draws satisfied the soft conditions and thus are used for inference, simply check the dimension of the `draws` component of the returned object (see the on-line help file for `forecast.object` for details):

```
> dim(zpolicy.spred$draws)
[1] 372  72
```

In this case, only 372 out of 1000 simulation draws satisfied the conditions. To continue simulating from the posterior moment distribution, use the same command as before, with seed set to the current value of `.Random.seed`:

```
> zpolicy.spred2 = cpredict(zpolicy.bar13, 12, steps=1,
+ variables="FFR", upper=0.195, lower=0.185, seed=.Random.seed)
> dim(zpolicy.spred2$draws)
[1] 389  72
```

Note that the draws in `zpolicy.spred2` can be combined with the draws in `zpolicy.spred` to obtain an updated and more accurate estimate of conditional forecasts.

To ignore the coefficient uncertainty for the conditional forecasts, call the classical method function `cpredict.VAR` directly on a fitted Bayesian VAR object. The technique introduced above can also be used for classical prediction with soft conditions.

11.7 References

[1] Campbell, J. A. Lo and C. MacKinlay (1997). *The Econometrics of Financial Markets*. Princeton University Press, New Jersey.

[2] Culbertson, K. (1996). *Quantitative Financial Economics: Stocks, Bonds and Foreign Exchange*. John Wiley and Sons, Chichester.

[3] Doan, T. A., Litterman, R. B., and Sims, C. A. (1984). "Forecasting and Conditional Projection Using Realistic Prior Distributions", *Econometric Reviews*, 3, 1-100.

[4] Granger, C.W.J. (1969). "Investigating Causal Relations by Econometric Models and Cross Spectral Methods," *Econometrica*, 37, 424-438.

[5] Hamilton, J.D. (1994). *Time Series Analysis*. Princeton University Press, Princeton.

[6] Lee, B.-S. (1992). "Causal Relations Among Stock Returns, Interest Rates, Real Activity, and Inflation," *Journal of Finance*, 47, 1591-1603.

[7] Lutkepohl, H. (1991). *Introduction to Multiple Time Series Analysis*. Springer-Verlag, Berlin.

[8] Lutkepohl, H. (1999). "Vector Autoregressions," unpublished manuscript, Institut für Statistik und Ökonometrie, Humboldt-Universität zu Berlin.

[9] Mills, T.C. (1999). *The Econometric Modeling of Financial Time Series, Second Edition*. Cambridge University Press, Cambridge.

[10] Runkle, D. E. (1987). "Vector Autoregressions and Reality," *Journal of Business and Economic Statistics*, 5 (4), 437-442.

[11] Sims, C.A. (1980). "Macroeconomics and Reality," *Econometrica*, 48, 1-48.

[12] Sims, C. A., and Zha, T. (1998). "Bayesian Methods for Dynamic Multivariate Models", *International Economic Review*, 39 (4), 949-968

[13] Stock, J.H. and M.W. Watson (2001). "Vector Autoregressions," *Journal of Economic Perspectives*, 15, 101-115.

[14] Tsay, R. (2001). *Analysis of Financial Time Series*. John Wiley & Sons. New York.

[15] Watson, M. (1994). "Vector Autoregressions and Cointegration," in *Handbook of Econometrics, Volume IV*. R.F. Engle and D. McFadden (eds.). Elsevier Science Ltd., Amsterdam.

[16] Waggoner, D. F., and Zha, T. (1999). "Conditional Forecasts in Dynamic Multivariate Models," *Review of Economics and Statistics*, 81 (4), 639-651.

[17] Zha, T. (1998). "Dynamic Multivariate Model for Use in Formulating Policy", *Economic Review*, Federal Reserve Bank of Atlanta, First Quarter, 1998.