State Space Methods in RATS
Technical Paper No. 2010-2

Thomas A. Doan *

January 2010

*Estima, 1560 Sherman Ave #510, Evanston, IL 60201 USA, email:tomd@estima.com
Prepared for special issue of the *Journal of Statistical Software* on software packages for state space time series analysis.

1 Introduction

RATS is a general-purpose econometrics and programming package, with a specialty in time series analysis. The instruction in **RATS** for handling state space models is called **DLM** (for **D**ynamic **L**inear **M**odels). This was introduced with version 5.00 in 2001. Since then, there have been many improvements. With version 8, the instruction has the following features:

- All component matrices can be fixed matrices, or time-varying formulas, or can be computed using functions of arbitrary complexity.
- Multiple observables (y_t with dimension > 1) are permitted, with proper handling of situations where some components are missing, but some are present.
- Non-stationary roots in the transition matrix are treated with the "exact" (limit) methods of Koopman (1997) and Durbin and Koopman (2002). The transition matrix is analyzed automatically for stationary and non-stationary roots.
- The ergodic variance for the stationary (linear combinations of) states is computed using the efficient Schur decomposition method described in Doan (2010).
- The calculations of the Kalman filter and smoother can switch under user control to use the last calculated values of the Kalman gain and predictive variances to save time in large models with time-invariant component matrices.

With the **RATS** distribution, we include the worked examples from several textbooks devoted to space-space models, including Durbin and Koopman (2002), Commandeur and Koopman (2007) and West and Harrison (1997). These are also posted on our web site at `www.estima.com/textbookindex.shtml`.

Our state space structure takes a bit broader form than the one described in the introduction to this issue. Because the components are input to the `DLM` instruction using short alphabetical names based upon our own description of the state space model, we will use that from this point on in this article:[1]

$$\mathbf{X}_t = \mathbf{A}_t\mathbf{X}_{t-1} + \mathbf{Z}_t + \mathbf{F}_t\mathbf{w}_t \tag{1}$$

$$\mathbf{Y}_t = \mu_t + \mathbf{C}_t'\mathbf{X}_t + \mathbf{v}_t \tag{2}$$

The addition of the μ_t term to the measurement equation is only a minor matter of convenience, since the identical model can be produced by subtracting μ_t from both sides of the equation. However, the enhanced form of the state equation with the $\mathbf{Z}_t$ state shift can't so easily be accomodated in the simpler form, particularly when the state shift component is time-varying.

Given a state space model, you can choose to:

[1] Note that **RATS** uses a different timing on the components in the state equation.

- Kalman filter, computing the likelihood function assuming Gaussian errors; this also computes predictions for the states and the observables, the prediction errors and variances for the predictions and states.
- Kalman smooth, with calculations of smoothed states and their variances, disturbances and their variances.
- Simulate unconditionally, with random draws for the state and measurement shocks, producing simulated states and observables.
- Simulate conditional on observed data, producing simulated states and shocks.

A recent addition to **RATS** is the instruction `DSGE`, which takes a model with expectational terms and solves it symbolically for a backwards-looking state space representation. The combination of `DSGE` and `DLM` can be used to evaluate the likelihood (for Gibbs sampling) or directly estimate by maximum likelihood the deep parameters in a DSGE.

This paper is organized as follows. Section 2 introduces the model used in the example. Section 3 demonstrates Kalman smoothing, given values for the variances. Section 4 shows the various ways to estimate the hyperparameters (variances). Section 5 discusses several types of calculated or graphed diagnostics for the state space model. Section 6 shows how to forecast out-of-sample and Section 7 offers examples of the use of both unconditional and conditional simulations. In all cases, we are providing only the segment of code needed to demonstrate a technique. The full running examples are available on our web site at `www.estima.com/resources_articles.shtml`.

2 The example

The model that we'll use for the examples is the local level model, applied to the Nile flow data, annual from 1871 to 1970. The model is

$$y_t = \alpha_t + \varepsilon_t$$
$$\alpha_t = \alpha_{t-1} + \eta_t$$

where α_t is the unobservable local level. The model has time-invariant components $\mathbf{A} = \mathbf{C} = \mathbf{F} = 1$, $\mathbf{Z} = \mu = 0$. These are the default values for all but C. The measurement error variance σ_ε^2 is input using the `SV` option, while the state shock variance σ_η^2 comes in through the `SW` option.

As with other Unobserved Components (UC) models, the state has non-stationary dynamics. To handle the initial conditions, we can use the option `PRESAMPLE=DIFFUSE`, which indicates that the initial condition for the state is fully diffuse. This is implemented using the "exact" method of Koopman (1997) and Durbin and Koopman (2002). The same outcome will be obtained using the more flexible `PRESAMPLE=ERGODIC`, which analyzes the transition matrix and determines its roots.

3 Kalman Smoothing

For now, we'll take the component variances as given, and discuss estimation in Section 4. We'll peg them at $\sigma_\varepsilon^2 = 15099$ and $\sigma_\eta^2 = 1469.1$ which are the maximum likelihood values. The instruction for Kalman smoothing with the Nile data is:

```
dlm(a=1.0,c=1.0,sv=15099.0,sw=1469.1,presample=diffuse,y=nile,$
  type=smooth) / xstates vstates
```

TYPE=SMOOTH chooses Kalman smoothing. The default analysis is Kalman filtering—the extra calculations for Kalman smoothing aren't done unless requested. The XSTATES parameter gets the smoothed state estimates and VSTATES gets the smoothed state variances. Since the state vector is (in almost all cases) bigger than a single element, XSTATES is a time series of vectors and VSTATES is a time series of (symmetric) matrices. Code for generating 90% confidence intervals and graphing them is given next:

```
set a     = %scalar(xstates)
set p     = %scalar(vstates)
set lower = a+sqrt(p)*%invnormal(.05)
set upper = a+sqrt(p)*%invnormal(.95)
graph(footer="Figure 1. Smoothed state and 90% confidence intervals") 4
# nile
# a
# lower / 3
# upper / 3
```

SET is the main **RATS** instruction for creating and transforming time series. The %SCALAR function selects the first element out of a vector or matrix, so the series A will be the time series of estimated states, and P the time series of estimated variances. GRAPH is the time series graphing instruction; the / 3 on the last two lines forces the upper and lower bounds to use the same color or pattern. The graph produced by this is Figure 1.

4 Estimation of Hyperparameters

The DLM instruction will always, as a side effect, compute the log likelihood of the model given the input variances. This can be maximized, with a wide range of choices for optimization, allowing for both derivative-based hill-climbing techniques, and slower but more flexible search methods. It also has the ability to (easily) incorporate equality or inequality constraints.

One way to estimate the two variances in the local level model is:

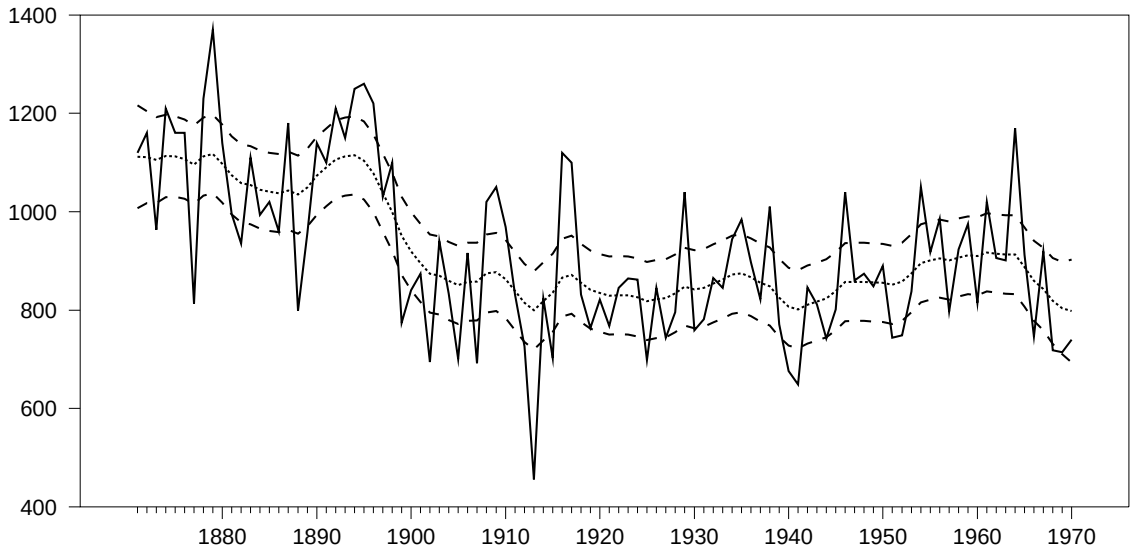

Figure 1: Smoothed state and 90% confidence intervals

```
nonlin psi
compute psi=0.0
dlm(a=1.0,c=1.0,sv=1.0,sw=exp(psi),y=nile,$
  presample=diffuse,method=bfgs,var=concentrate)
```

The NONLIN instruction declares the set of free parameters to be estimated—here, that's $\psi = \log\left(\sigma_\eta^2/\sigma_\varepsilon^2\right)$. The measurement error variance is concentrated out, which can sometimes be helpful in improving the behavior of difficult estimation problems. The estimation method being used here is the hill-climbing method BFGS. The output is shown in Table 1. Note that, while there are 100 data points, the likelihood is calculated using only the final 99 of them. This is done automatically here because of the diffuse initial conditions—the predictive variance for observation 1 is infinite, and so it's dropped from the calculation of the likelihood. DLM has an additional option CONDITION which can control the number of data points which are included in the filtering calculations, but omitted from the likelihood used for estimation. This is generally not needed, since DLM handles the diffuse states automatically, but is useful when the number of non-stationary states isn't known *a priori*, if, for instance, autoregressive parameters are being estimated.

Both variances can also be estimated directly with:

```
nonlin sigsqeps sigsqeta
stats(noprint) nile
compute sigsqeps=.5*%variance,sigsqeta=.1*sigsqeps
*
dlm(a=1.0,c=1.0,sv=sigsqeps,sw=sigsqeta,y=nile,$
  method=bfgs,presample=diffuse) 1871:1 1970:1
```

Direct estimation of the variances requires a bit more care with guess values. This

```
DLM - Estimation by BFGS
Convergence in     6 Iterations. Final criterion was  0.0000001 <=  0.0000100
Annual Data From 1871:01 To 1970:01
Usable Observations    100
Rank of Observables     99
Log Likelihood           -632.54563
Concentrated Variance    15098.51951564

    Variable                     Coeff      Std Error      T-Stat      Signif
********************************************************************************
1.  PSI                      -2.329895195  1.012133212     -2.30196  0.02133715
```

Table 1: Estimation with Concentrated Variance

uses scalings of the series sample variance, which should get the order of magnitude correct. The output is in Table 2.

```
DLM - Estimation by BFGS
Convergence in     9 Iterations. Final criterion was  0.0000000 <=  0.0000100
Annual Data From 1871:01 To 1970:01
Usable Observations    100
Rank of Observables     99
Log Likelihood           -632.54563

    Variable                     Coeff      Std Error      T-Stat      Signif
********************************************************************************
1.  SIGSQEPS                 15098.510028  3126.130999      4.82978  0.00000137
2.  SIGSQETA                  1469.172357  1266.235944      1.16027  0.24593993
```

Table 2: Estimation with Both Variances

While not important here, the NONLIN instruction can also handle various constraints on the parameters, either equality or inequality. With no change to the setup, we could estimate this with σ_η^2 pegged to zero (which here gives a model with a fixed mean) using

```
nonlin sigsqeps sigsqeta=0.0
dlm(a=1.0,c=1.0,sv=sigsqeps,sw=sigsqeta,y=nile,$
  method=bfgs,presample=diffuse) 1871:1 1970:1
```

In a more complex model, where there is some chance that a component variance might be zero, the NONLIN instruction can be used to set an inequality constraint:

```
nonlin sigsqeps sigsqeta>=0.0
dlm(a=1.0,c=1.0,sv=sigsqeps,sw=sigsqeta,y=nile,$
  method=bfgs,presample=diffuse) 1871:1 1970:1
```

This uses a penalty function variation on BFGS. Since it's quite a bit slower than standard BFGS, we generally don't recommend using it unless the simpler unconstrained estimates fail to provide values in range. The equality constraints from the previous

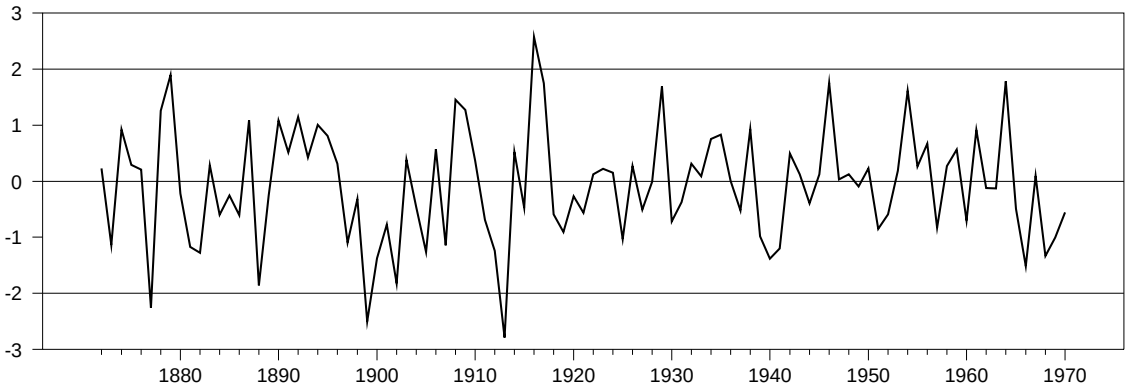

Figure 2: Standardized prediction errors

case, on the other hand, are done by taking the constrained parameter out of the parameter set and using standard BFGS, so it actually runs faster than unconstrained BFGS.

5 Diagnostics

The most straightforward diagnostics come from the standardized residuals. These can be computed with the help of the VHAT and SVHAT options. VHAT is used to fetch the measurement errors and SVHAT the predictive variance. Again, these will be in the form of a VECTOR (for VHAT) and a SYMMETRIC matrix (for SVHAT) to allow for the possibility of multiple observables. The following generates standardized predictive errors (into the series EHAT), graphs them (Figure 2) and does a standard set of diagnostics on the recursive residuals (output in Table 3):

```
dlm(a=1.0,c=1.0,sv=sigsqeps,sw=sigsqeta,y=nile,$
  method=bfgs,presample=diffuse,$
  vhat=vhat,svhat=svhat) 1871:1 1970:1
set ehat = %scalar(vhat)/sqrt(%scalar(svhat))
graph(footer="Standardized residual",vgrid=||-2.0,2.0||)
# ehat
@STAMPDiags(ncorrs=9) ehat
```

The VGRID=||-2.0,2.0|| option on the GRAPH puts the horizontal lines at ± 2. Note that, because of the diffuse prior, the first standardized error is omitted. This is handled automatically in the code because the SVHAT for 1871:1 is a missing value.

The diagnostics in Table 3 include a Ljung-Box Q test for serial correlation, a Jarque-Bera normality test and a Goldfeld-Quandt style test for heteroscedasticity. The STAMPDiags procedure also produces the graph of autocorrelations seen in Figure 3.

Durbin and Koopman (2002) recommend also computing auxiliary residuals, which

```
          Statistic Sig. Level
Q(9-1)         8.84     0.3557
Normality      0.12     0.9441
H(33)          0.61     0.1650
```

Table 3: State Space Model Diagnostics

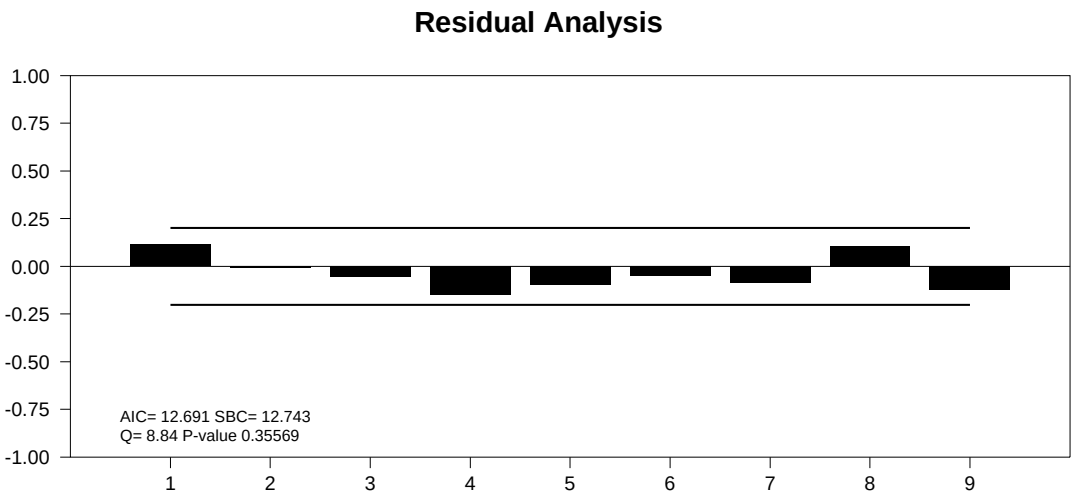

Figure 3: Autocorrelations of prediction errors

are the Kalman smoothed estimates for the measurement errors and state disturbances. Large values for these can help identify outliers (in the measurement errors) or structural shifts (in the state disturbances). These can be obtained using the VHAT and WHAT options when Kalman smoothing. The results returned from those are standardized to mean zero, unit variance.

```
dlm(a=1.0,c=1.0,sv=sigsqeps,sw=sigsqeta,y=nile,$
  type=smooth,presample=diffuse,$
  vhat=vhat,what=what)
*
set outlier = %scalar(vhat)
diff(standardize) outlier
set break   = %scalar(what)
diff(standardize) break
```

The following graphs both of these. This uses SPGRAPH instructions to create a graph page with two panes. The result is Figure 4.

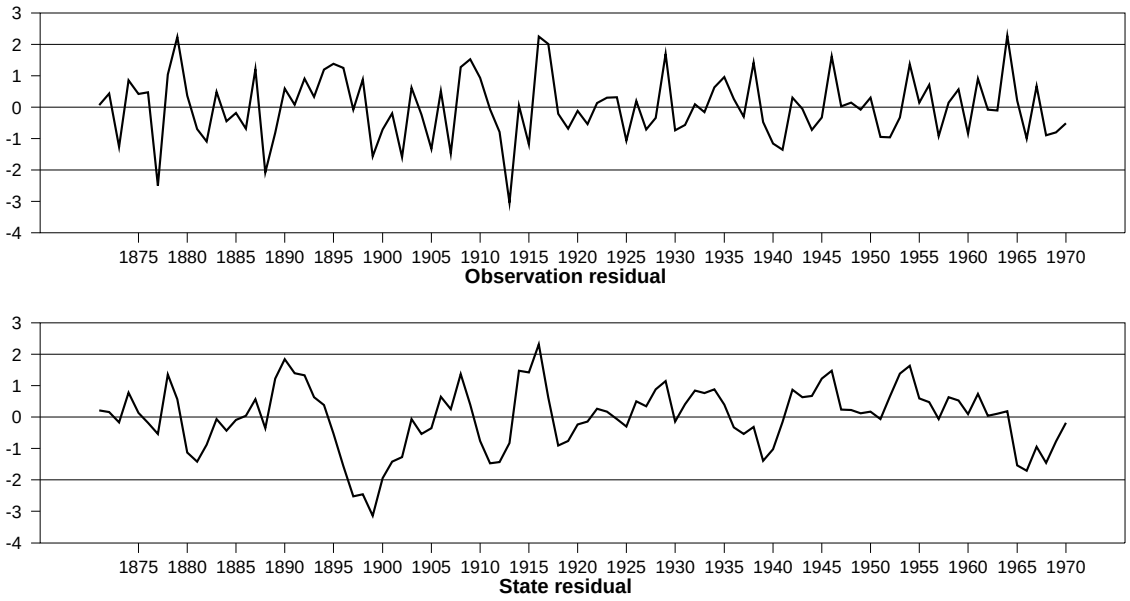

Figure 4: Diagnostic plots for auxiliary residuals

```
spgraph(vfields=2,$
  footer="Diagnostic plots for auxiliary residuals")
graph(vgrid=||-2.0,2.0||,hlabel="Observation residual")
# outlier
graph(vgrid=||-2.0,2.0||,hlabel="State residual")
# break
spgraph(done)
```

6 Forecasts

Out-of-sample forecasts *can* be generated by simply running a Kalman filter past the end of the data set. When the Y value is missing, `DLM` does the Kalman "update" step but not the "correction". This is how embedded missing values are handled. For out-of-sample forecasts, however, it's generally more straightforward to Kalman filter through the observed data, then run a separate filter into the forecast range.

This next code segment uses the `X0` and `SX0` options to feed in the final estimated mean and variance for the states (from Kalman filtering over the sample) into the Kalman filter for the forecast range. The `YHAT` and `SVHAT` options are used to get the prediction and the predictive error variance for the dependent variable. You can also get the predicted value of the state and its predictive variance using the standard state parameters.

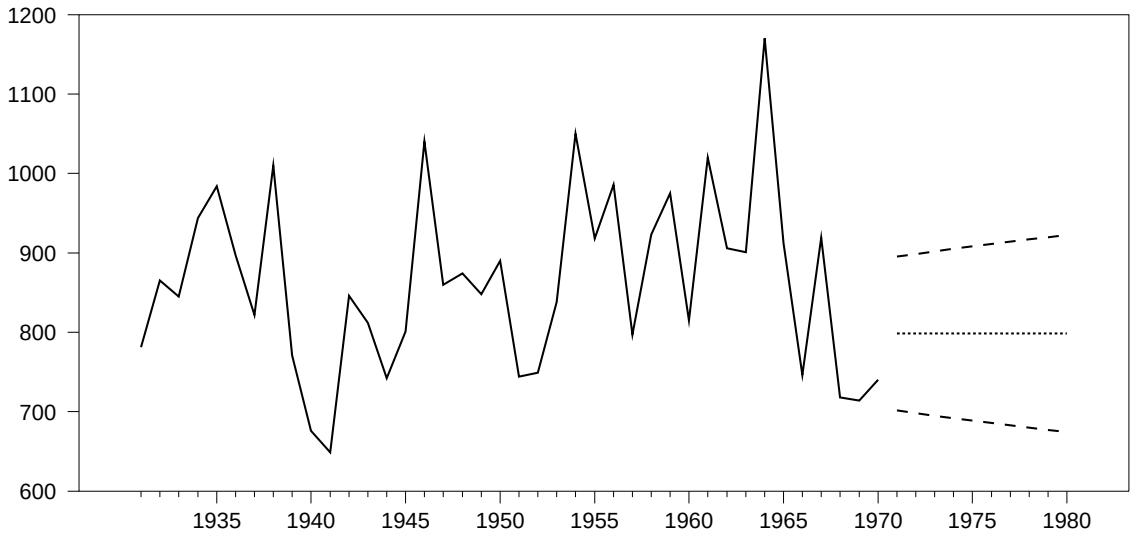

Figure 5: Out-of-sample Forecasts with 50% CI

```
dlm(a=1.0,c=1.0,sv=15099.0,sw=1469.1,presample=diffuse,y=nile,$
  type=filter) / xstates vstates
dlm(a=1.0,c=1.0,sv=15099.0,sw=1469.1,$
  x0=xstates(1970:1),sx0=vstates(1970:1),$
  yhat=yhat,svhat=svhat) 1971:1 1980:1
set forecast 1971:1 1980:1 = %scalar(yhat)
set stderr   1971:1 1980:1 = sqrt(%scalar(svhat))
```

The following organizes a graph of the forecasts with their 50% confidence interval. Only forty years of actual data are included to give the forecast range enough space. This produces Figure 5.

```
set lower 1971:1 1980:1 = forecast+%invnormal(.25)*stderr
set upper 1971:1 1980:1 = forecast+%invnormal(.75)*stderr
graph(footer="Out-of-sample Forecasts with 50% CI") 4
# nile     1931:1 1970:1
# forecast / 2
# lower    / 3
# upper    / 3
```

7 Simulations

There are two choices for random simulations of a model: `TYPE=SIMULATE` chooses unconditional simulation, where shocks for the states and measurements are drawn independently, and `TYPE=CSIMULATE`, where they are drawn subject to the requirement that the observed data are produced. `TYPE=SIMULATE` would generally be used in out-of-sample operations, while `TYPE=CSIMULATE` is especially useful for Gibbs

sampling, since it draws a set of shocks and states conditional on the data and other parameters.

The following is an example of unconditional simulation. First, Kalman filtering is used through the observed range of the data to get the end-of-period estimates of the mean and variance of the state. Then 10000 realizations for the process over the next fifty periods are generated. The maximum flow for each realization is recorded. The percentiles are computed once the simulations are done. This could be used, for instance, to estimate the level for 50-year or 100-year floods.

```
dlm(a=1.0,c=1.0,sv=15099.0,sw=1469.1,presample=diffuse,y=nile,$
  type=filter) / xstates vstates
compute ndraws=10000
set maxflow 1 ndraws = 0.0
do reps=1,ndraws
  dlm(a=1.0,c=1.0,sv=15099.0,sw=1469.1,$
    x0=xstates(1970:1),sx0=vstates(1970:1),$
    type=simulate,yhat=yhat) 1971:1 2020:1 xstates
  set simflow 1971:1 2020:1 = %scalar(yhat)
  ext(noprint) simflow
  compute maxflow(reps)=%maximum
end do reps
stats(fractiles,nomoments) maxflow
```

```
Statistics on Series MAXFLOW
Observations       10000
Minimum        646.279562      Maximum 2277.525740
01-%ile        797.704496      99-%ile 1664.309419
05-%ile        884.530943      95-%ile 1498.885021
10-%ile        933.496769      90-%ile 1409.103502
25-%ile       1024.542961      75-%ile 1274.081468
Median        1135.600473
```

Table 4: Percentiles from Maximum Simulated Flows

The following is an example of the use of conditional simulation for Gibbs sampling. The two hyperparameters are modeled as the precision h of the measurement error and the relative variance ψ of the state shock to the measurement error. The two hyperparameters are given very loose priors, with h being inverse gamma with 1 degree of freedom and ψ being gamma with 1 degree of freedom.

```
compute nuh=1.0
compute s2h=100.0^2
compute hdraw=nuh/s2h
*
compute nupsi=1.0
compute s2psi=.1
compute psidraw=s2psi/nupsi
```

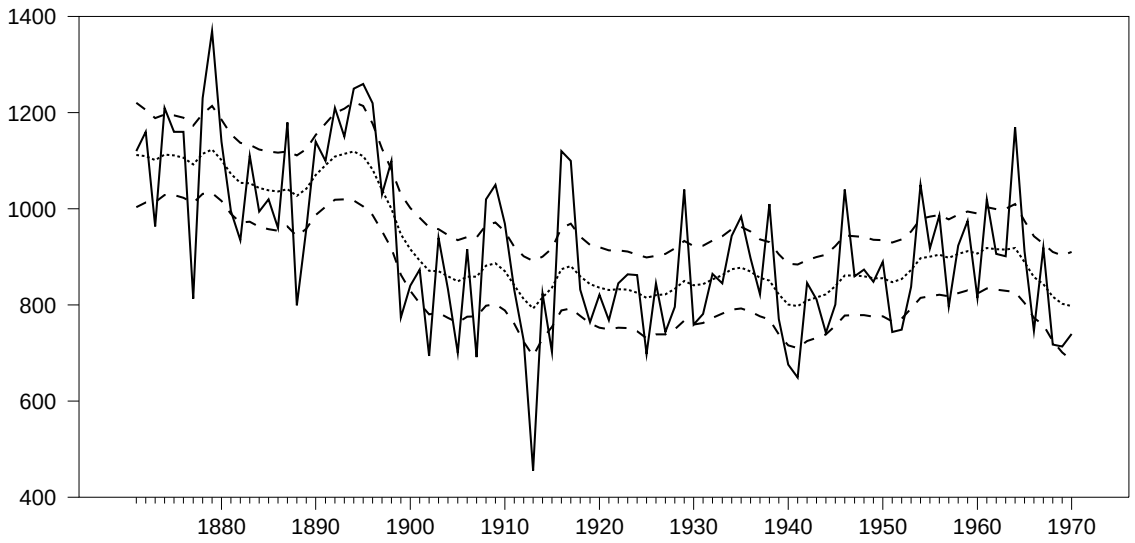

Figure 6: Local level and 90% CI from Gibbs sampling

In the example, the Gibbs sampler is run with 100 burn-in draws and 2000 keeper draws. DLM with TYPE=CSIMULATE does a draw from the joint distribution of the measurement error and state disturbances subject to the simulated data hitting the observed values for Y. The WHAT and VHAT options are used to get the simulated values of the disturbances, while the states parameter gets the simulated values of the states. This does the simulation conditional on the current draws for h and ψ.

```
dlm(a=1.0,c=1.0,sv=1.0/hdraw,sw=psidraw/hdraw,y=nile,$
   type=csimulate,presample=diffuse,$
   what=what,vhat=vhat) / xstates
```

The hyperparameters are then drawn conditional on the just-created draws for the disturbances. First ψ:

```
sstats / vhat(t)(1)^2>>sumvsq what(t)(1)^2>>sumwsq
compute psidraw=(hdraw*sumwsq+nupsi*s2psi)/%ranchisqr(%nobs+nupsi)
```

then h:

```
compute hdraw=$
   %ranchisqr(nuh+%nobs*2.0)/(nuh*s2h+sumvsq+sumwsq/psidraw)
```

Estimates of the local level and its 90% confidence interval are shown in Figure 6. This is similar to Figure 1, but allows for the fact that the hyperparameters are estimated, and not known.

8 Conclusion

This paper has used a simple example to give a taste of how **RATS** can be used to work with state space models. The DLM instruction has many options, allowing it to handle a wide range of tasks. Its internal calculations for filtering, smoothing and simulation have been highly optimized. When combined with the programming flexibility of the **RATS** package, many models which are quite cumbersome when done with matrix languages or less flexible packages can be done simply and quickly. We invite you to check our web site or e-mail us for more information.

References

COMMANDEUR, J. J. F., AND S. KOOPMAN (2007): *An Introduction to State Space Time Series Analysis*. Oxford: Oxford University Press.

DOAN, T. A. (2010): "Practical Issues with State-Space Models with Mixed Stationary and Non-Stationary Dynamics," *Estima Technical Paper*, (1).

DURBIN, J., AND S. KOOPMAN (2002): *Time Series Analysis by State Space Methods*. Oxford: Oxford University Press.

KOOPMAN, S. (1997): "Exact Initial Kalman Filtering and Smoothing for Nonstationary Time Series Models," *Journal of American Statistical Association*, 92(6), 1630–1638.

WEST, M., AND J. HARRISON (1997): *Bayesian Forecasting and Dynamic Models*. New York: Springer.