Modeling Financial Time Series using S+FinMetrics™

Eric Zivot, Associate Professor of Economics
and
Gary Waterman Distinguished Scholar
University of Washington

Dec 5th, 2002

Agenda

- Introduction
  - Typical Applications in Finance
  - Overview of S-PLUS / S+FinMetrics™
- Time Series Objects in S-PLUS
- Applications of S+FinMetrics™
  - Rolling Analysis of Time Series
  - Vector Autoregressive Models (VAR)
- Conclusions
- Literature/References

Agenda

- **Introduction**
  - **Typical Applications in Finance**
  - **Overview of S-PLUS / S+FinMetrics™**
- Time Series Objects in S-PLUS
- Applications of S+FinMetrics™
  - Rolling Analysis of Time Series
  - Vector Autoregressive Models (VAR)
- Conclusions
- Literature/References

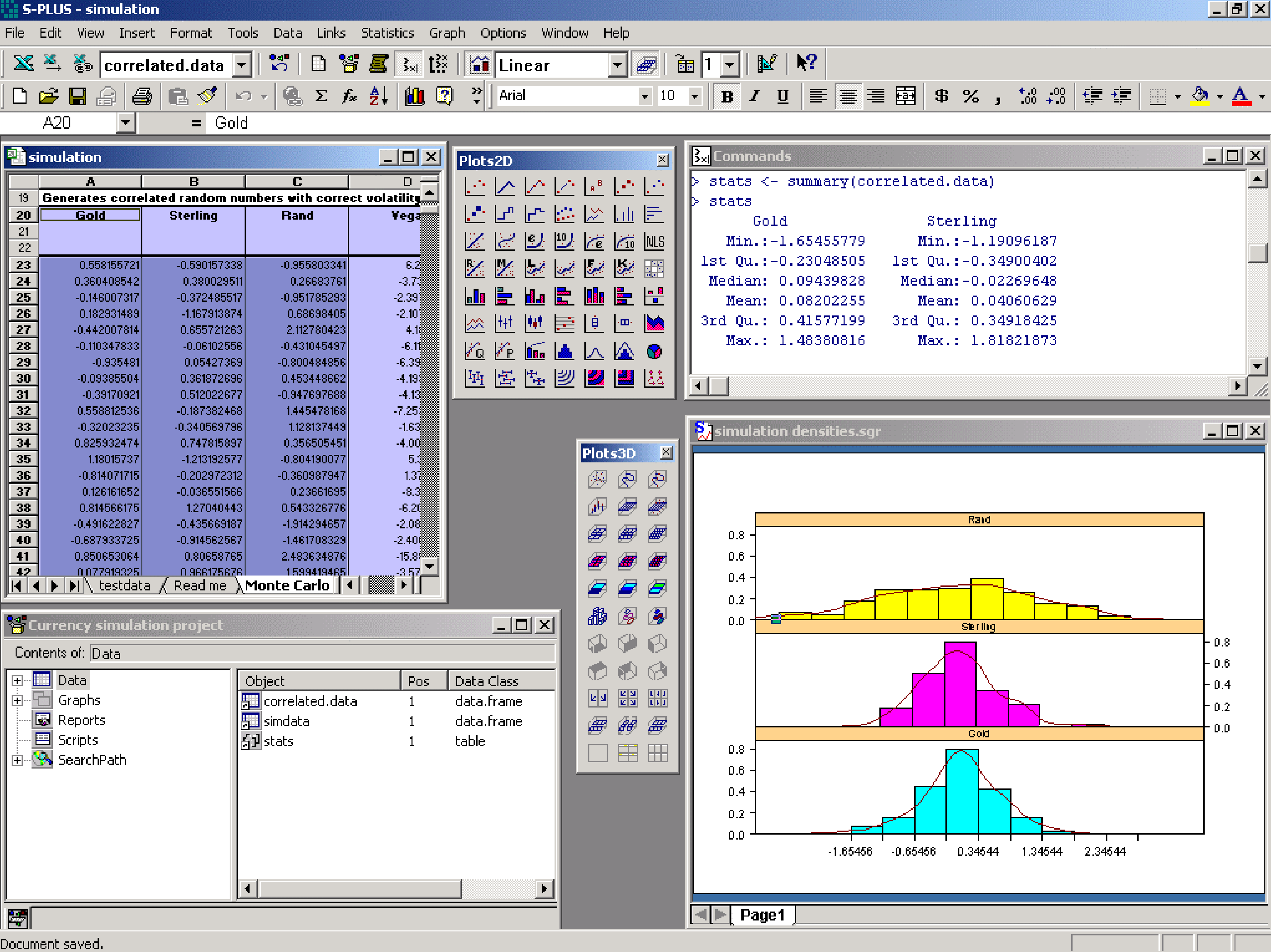
Typical Applications in Finance

- Modeling risk and derivative valuations
- Predicting financial performance
- Building program trading applications
- Predicting multivariate macroeconomic and financial time series
- Back testing regression and other generic analytic models
- Predicting daily volatilities and correlations for asset returns

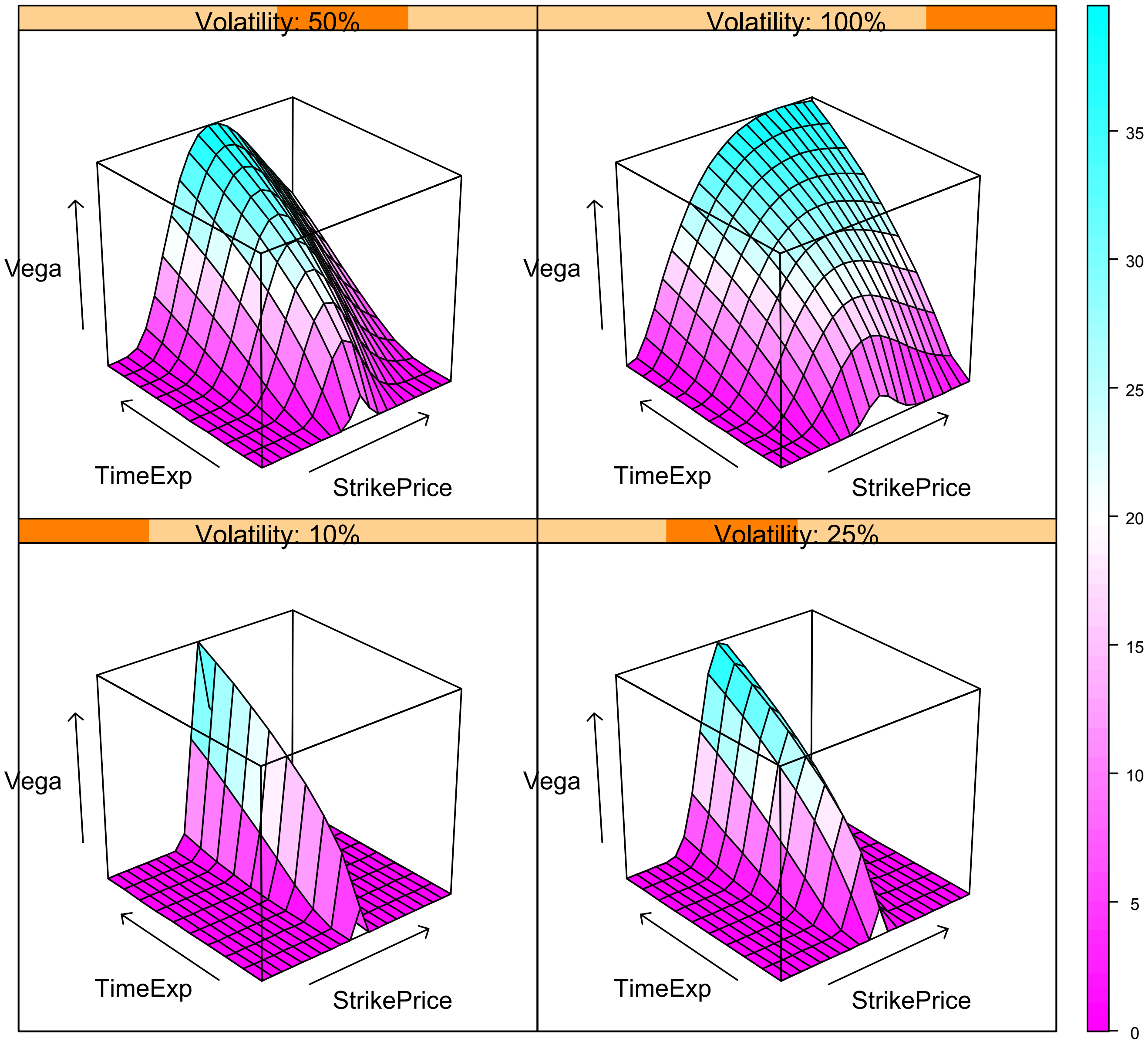
What is S-PLUS?

- Premier solution for exploratory data analysis and statistical modeling
- Point-and-click to import various data formats, select statistical functions and display results
- Over 4,200 data analysis functions, including the most comprehensive set of robust and modern methods
- Modify existing methods or develop new ones with the award-winning S programming language
- Powerful, interactive graphics

S-PLUS® 6


S-PLUS - simulation
File Edit View Insert Format Tools Data Links Statistics Graph Options Window Help
correlated.data
Linear
Arial 10
A20 = Gold
simulation
A B C D
19 Generates correlated random numbers with correct volatility
20 Gold Sterling Rand Vega
23 0.558155721 -0.590157338 -0.955803341
24 0.360408542 0.380029511 0.26683761
25 -0.146007317 -0.372485517 -0.951785293
26 0.182931489 -1.167913874 0.68698405
27 -0.442007814 0.655721263 2.112780423
28 -0.110347833 -0.06102556 -0.431045497
29 -0.935481 0.05427369 -0.800484856
30 -0.09385504 0.361872696 0.453448662
31 -0.39170921 0.512022677 -0.947697688
32 0.558812536 -0.187382468 1.445478168
33 -0.32023235 -0.340569796 1.128137449
34 0.825932474 0.747815897 0.356505451
35 1.18015737 -1.213192577 -0.804190077
36 -0.814071715 -0.202972312 -0.360987947
37 0.126161652 -0.036551566 0.23661695
38 0.814566175 1.27040443 0.543326776
39 -0.491622827 -0.435669187 -1.914294657
40 -0.687933725 -0.914562567 -1.461708329
41 0.850653064 0.80658765 2.483634876
testdata Read me Monte Carlo
Plots2D
Plots3D
Commands
> stats <- summary(correlated.data)
> stats
Gold Sterling
Min.:-1.65455779 Min.:-1.19096187
1st Qu.:-0.23048505 1st Qu.:-0.34900402
Median: 0.09439828 Median:-0.02269648
Mean: 0.08202255 Mean: 0.04060629
3rd Qu.: 0.41577199 3rd Qu.: 0.34918425
Max.: 1.48380816 Max.: 1.81821873
simulation densities.sgr
Rand
Sterling
Gold
0.0 0.2 0.4 0.6 0.8
-1.65456 -0.65456 0.34544 1.34544 2.34544
Page1
Currency simulation project
Contents of: Data
Data
Graphs
Reports
Scripts
SearchPath
Object Pos Data Class
correlated.data 1 data.frame
simdata 1 data.frame
stats 1 table
Document saved.

Calculation of Vega for various scenarios



What is S+FinMetrics™?

- S-PLUS module with 500+ functions for the econometric modeling and prediction of economic and financial time series
- Created for researchers and practitioners in the finance industry, academic researchers in economics and finance, and advanced MBA and graduate students in economics and finance
- Exposes the power of S-PLUS for modeling univariate and multivariate time series data

S-PLUS / S+FinMetrics™

Simple Descriptive Tools

Advanced Modeling Tools

- Smoothing & Filtering
- ACF & PACF
- Spectral Analysis
- Seasonal Adjustment
- ARIMA with Regressors
- FARIMA
- Dynamic Ordinary Least Squares
- Rolling Estimation
- Seemingly Unrelated Regression
- Autoregression and Vector AR
- GARCH – Univariate, Multivariate & FI
- VaR Calculations
- Unit Root, Cointegration & VECM
- Technical Analysis & F. I. Analytics
- Statistical Multi-Factor Models
- Extreme Value Theory
- State Space Modeling

Agenda

- Introduction
  - S-PLUS/S+FinMetrics™
- **Time Series Objects in S-PLUS**
- Applications of S+FinMetrics™
  - Rolling Analysis of Time Series
  - Vector Autoregressive Models (VAR)
- Conclusions
- Literature/References

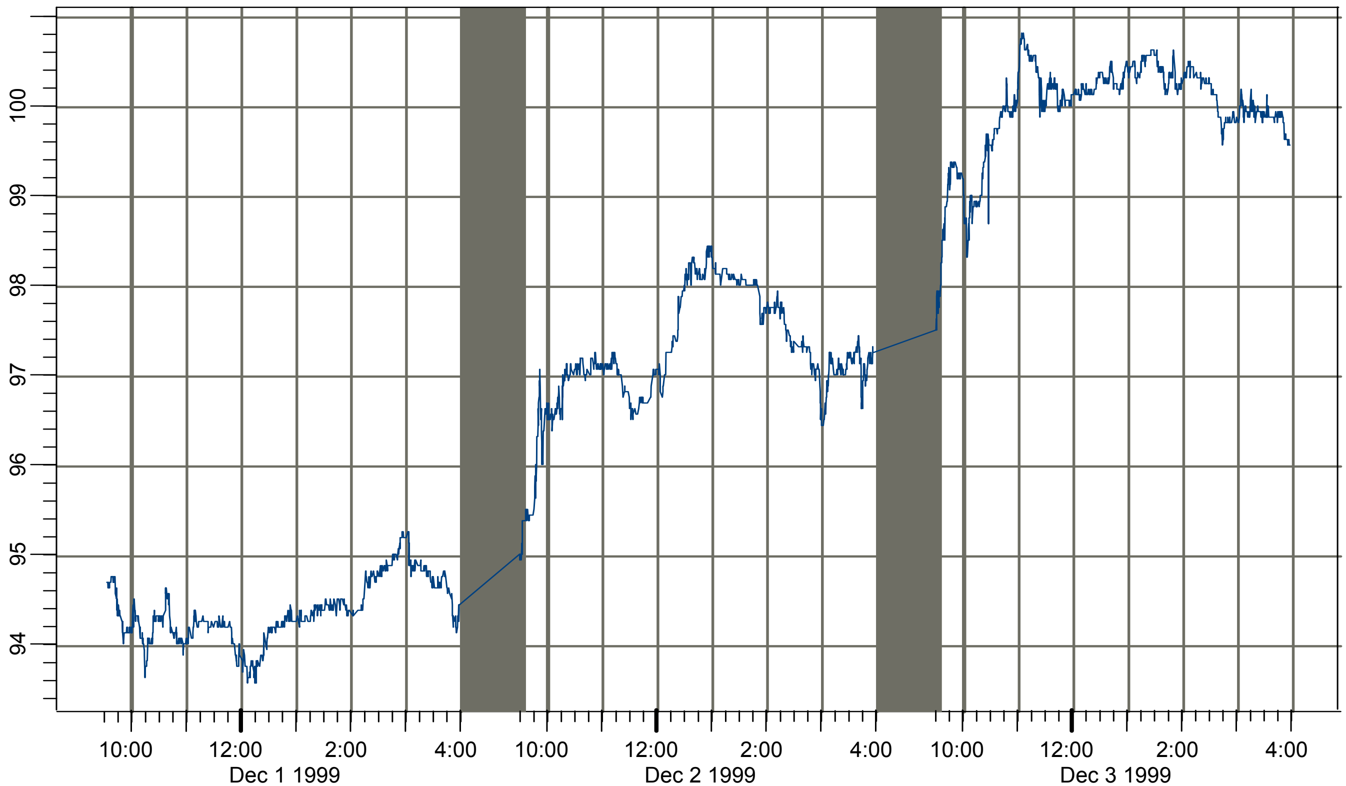
Time Series in S-PLUS

- S-PLUS 6.0 “timeSeries” Objects
  - Combines data with “timeDate” object
  - Flexible enough to describe essentially all types of financial time series data
    - Regularly spaced calendar data
    - Irregularly spaced tick-by-tick data
    - Allows time-zone specification
    - Easy event handling
      - Holidays, market closures, etc.
  - Powerful plotting functionality

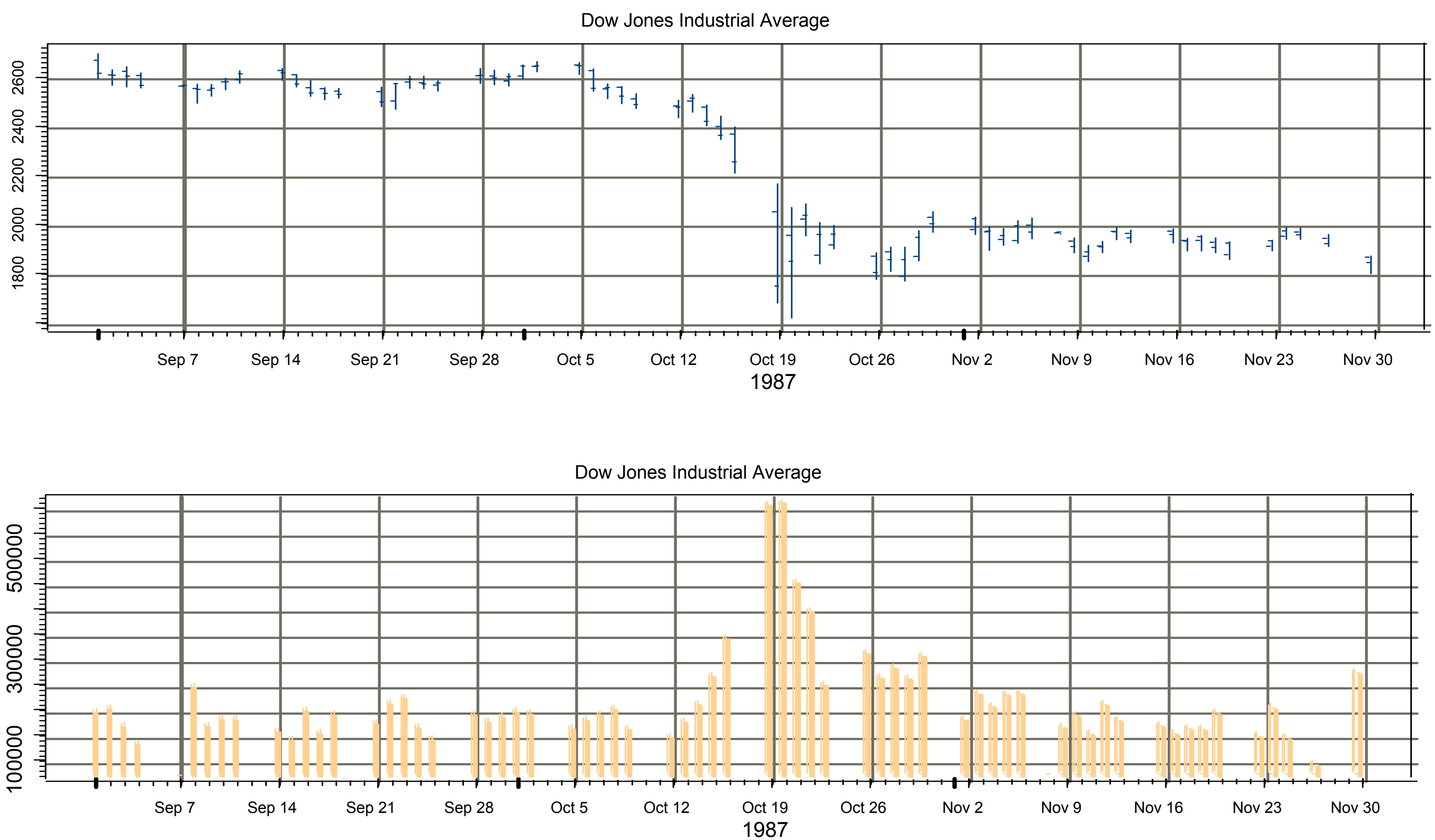
Time Series in S-PLUS

```
> class(hf3M.ts)
[1] "timeSeries‟

> slotNames(hf3M.ts)
[1] "data"                "positions"          "start.position"
[4] "end.position"        "future.positions"   "units"
[7] "title"               "documentation"      "attributes"
[10] "fiscal.year.start" "type"

> hf3M.ts[1:5,]
           Positions trade.day trade.time trade.price
 12/1/99 9:33:32 AM 1         34412      94.69
 12/1/99 9:33:34 AM 1         34414      94.69
 12/1/99 9:33:34 AM 1         34414      94.69
 12/1/99 9:33:34 AM 1         34414      94.69
 12/1/99 9:33:34 AM 1         34414      94.69
```

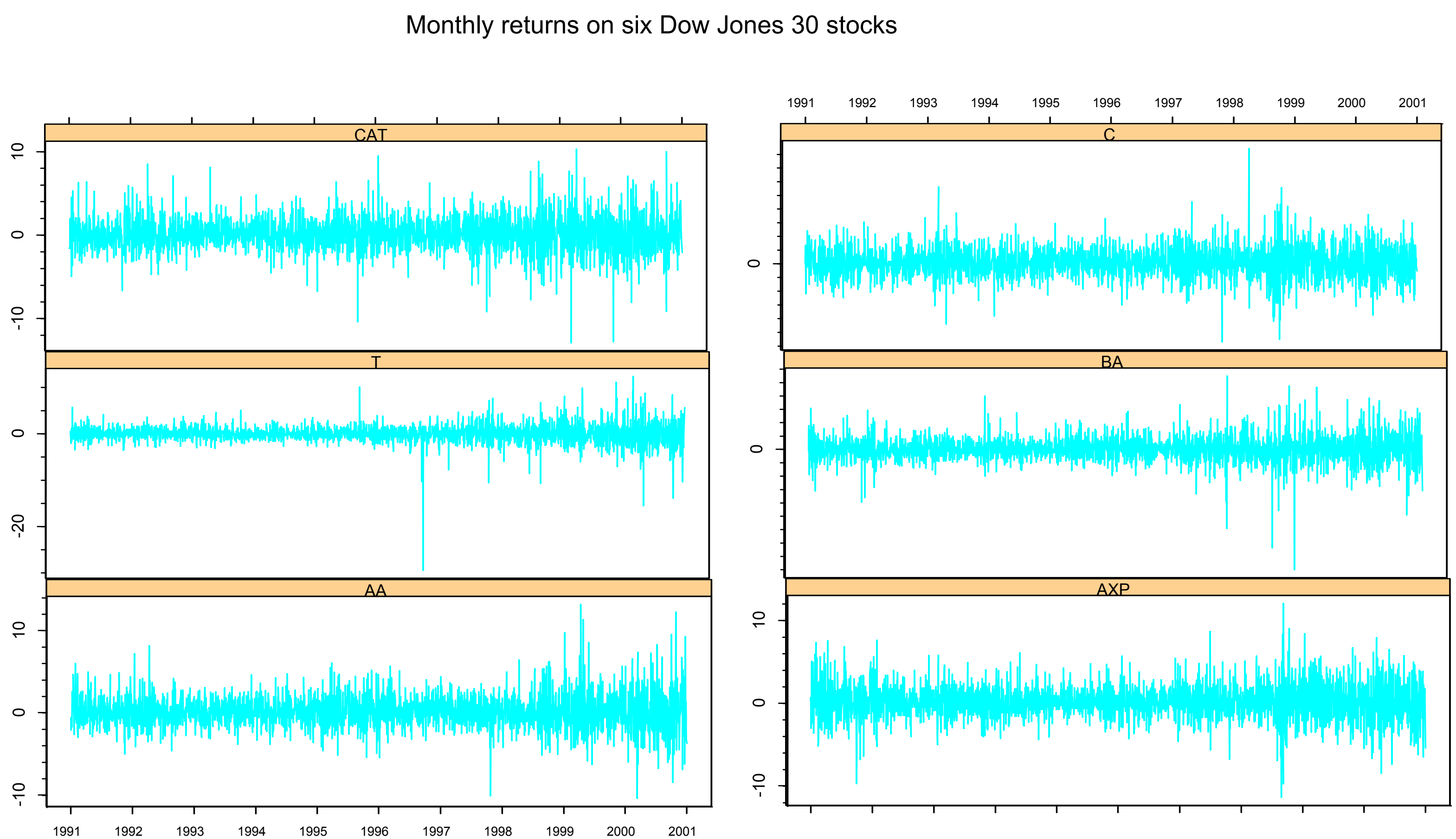
Time Series in S-PLUS



Time Series in S-PLUS



Time Series in S-PLUS



Agenda

- Introduction
  - Typical Applications in Finance
  - Overview of S-PLUS / S+FinMetrics™
- Time Series Objects in S-PLUS
- **Applications of S+FinMetrics™**
  - **Rolling Analysis of Time Series**
  - Vector Autoregressive Models (VAR)
- Conclusions
- Literature/References

Rolling Analysis of Time Series

- A rolling analysis of a time series model is often used to assess the model’s stability over time.
- Rolling analysis is a “poor man’s” time varying parameter model.
- Rolling analysis is commonly used to “backtest” a statistical model on historical data to evaluate stability and predictive accuracy

Example: Rolling Correlations

- Compute returns with `getReturns`

```
> ret.ts =
  getReturns(singleIndex.dat,type="continuous")

> ret.ts
 Positions          MSFT        SP500
 Feb 1990    0.0653798   0.0085027
 Mar 1990    0.1147079   0.0239655
…
 Jan 2001    0.3420147   0.0340502
```

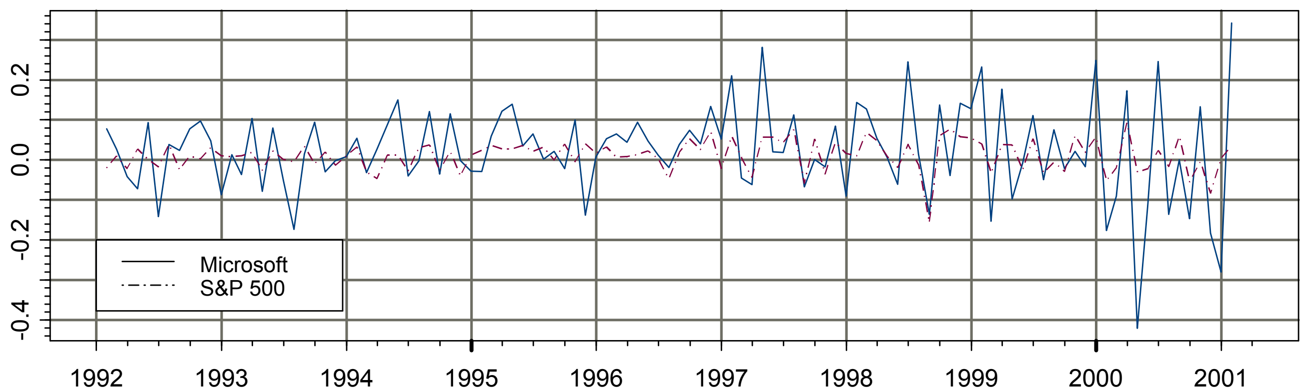
Rolling Correlations

- Compute 24-month rolling correlations using `aggregateSeries`

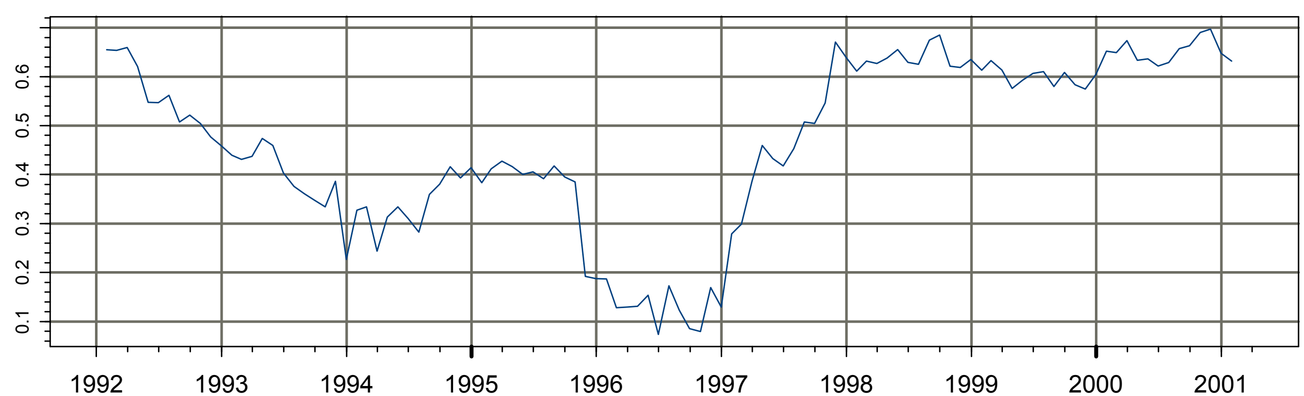
```
> cor.coef = function(x) cor(x)[1,2]
> roll.cor =
  aggregateSeries(ret.ts,moving=24,together=T,
+ adj=1,FUN=cor.coef)

> roll.cor
 Positions           1
 Jan 1992   0.65490645
 Feb 1992   0.65351145
…
 Jan 2001   0.63164385
```

Rolling Correlations

Returns on Microsoft and S&P 500 index



24-month rolling correlations



Exponentially Weighted Moving Average (EWMA)

- Definitions
  - EWMA estimates of variance, covariance and correlation

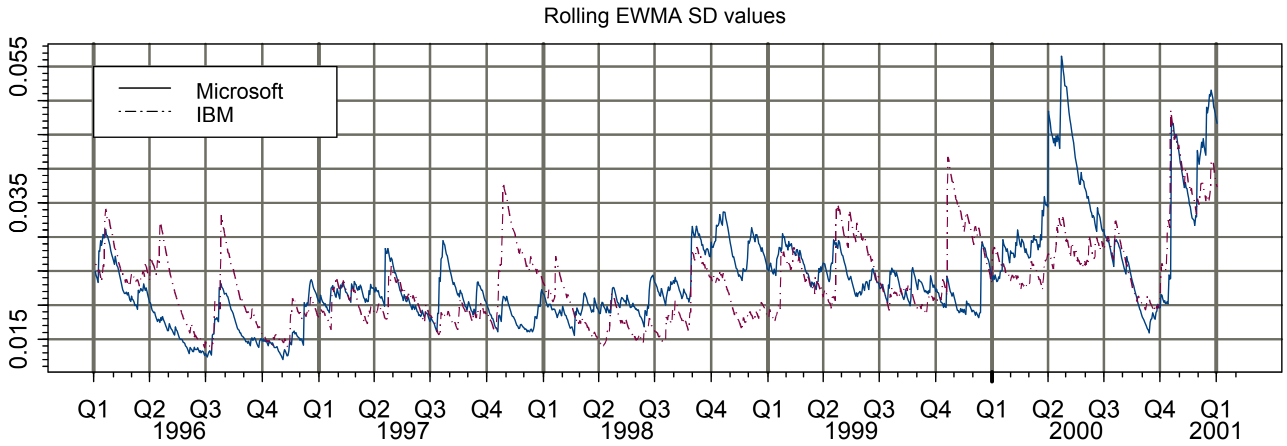
$$\hat{\sigma}_t^2(\lambda) = (1-\lambda)r_t^2 + \lambda\hat{\sigma}_{t-1}^2(\lambda)$$

$$\hat{\sigma}_{12,t}^2(\lambda) = (1-\lambda)r_{1t}r_{2t} + \lambda\hat{\sigma}_{12,t-1}^2(\lambda)$$

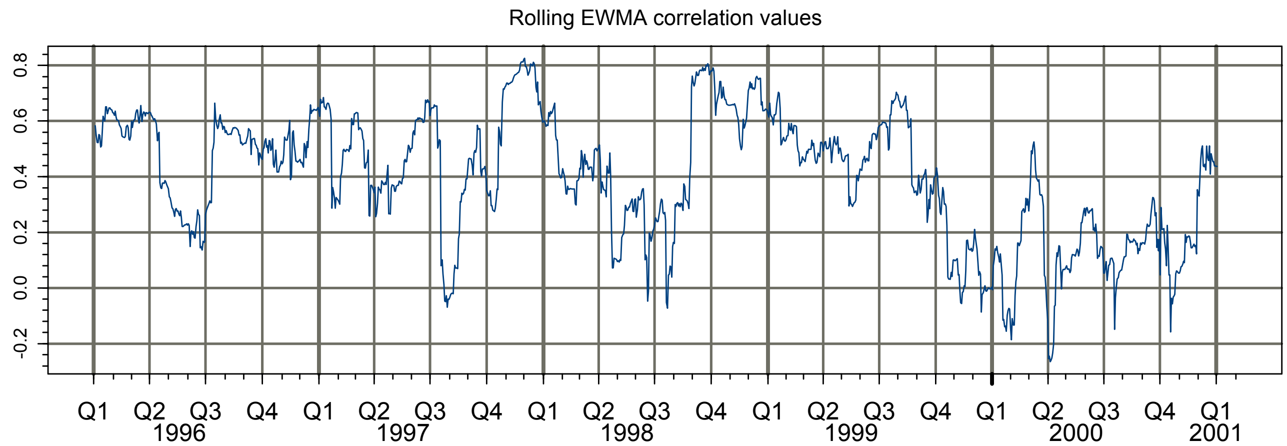
$$\hat{\rho}_{12,t}(\lambda) = \hat{\sigma}_{12,t}^2(\lambda) / \left(\hat{\sigma}_{1t}^2(\lambda)\cdot\hat{\sigma}_{2t}^2(\lambda)\right)^{1/2}$$

  - may be computed using `EWMA`

EWMA Correlations



$$\lambda = 0.95$$



General Roll Function

- S-PLUS function `aggregateSeries` is limited to simple functions
- S+FinMetrics function `roll` allows rolling analysis of general modeling functions
  - Rolling regression
  - Rolling glm
  - Rolling factor analysis
  - Rolling cointegration
  - Etc.

Rolling Regression

- For the linear regression model, rolling analysis may be used to assess the stability of the model's parameters and to provide a simple "poor man's" time varying parameter model
- Rolling predictions may be used to backtest a model's predictive performance over rolling windows of historical data, and to compare the predictive accuracy of rival models

Example: Predicting Stock Returns

- Goal: Predict annual real return on S&P 500 index using Dividend/Price as predictor

$$r_t(n) = \alpha(n) + \beta(n)x_t + \varepsilon_t(n),\ t = n,\ldots,T$$

$$r_t(n) = \text{ log real return}$$

$$x_t = \text{ lagged log dividend/price or log earnings/price}$$

$$n = \text{window width}$$

Predicting Stock Returns

- Compute 50-year rolling regression estimates, incremented by 1 year, using `rollOLS`

```
> roll.dp.fit = rollOLS(ln.r~tslag(ln.dpratio),data=stock.ts,
+ width=50,incr=1)
Rolling Window #1: Total Rows = 50
Rolling Window #2: Total Rows = 51
…
Rolling Window #70: Total Rows = 119

> class(roll.dp.fit)
[1] "rollOLS"

> names(roll.dp.fit)
 [1] "width"     "incr"      "nwin"      "contrasts" "rdf"
 [6] "coef"      "stddev"    "sigma"     "terms"     "call"
[11] "positions" "position"
```

Print Method

```
> roll.dp.fit
Call:
rollOLS(formula = ln.r ~
   tslag(ln.dpratio), data =
   stock.ts, width = 50, incr
   = 1)

Rolling Windows:
number width increment
 70    50          1
Time period: from 1882 to
   2000
```

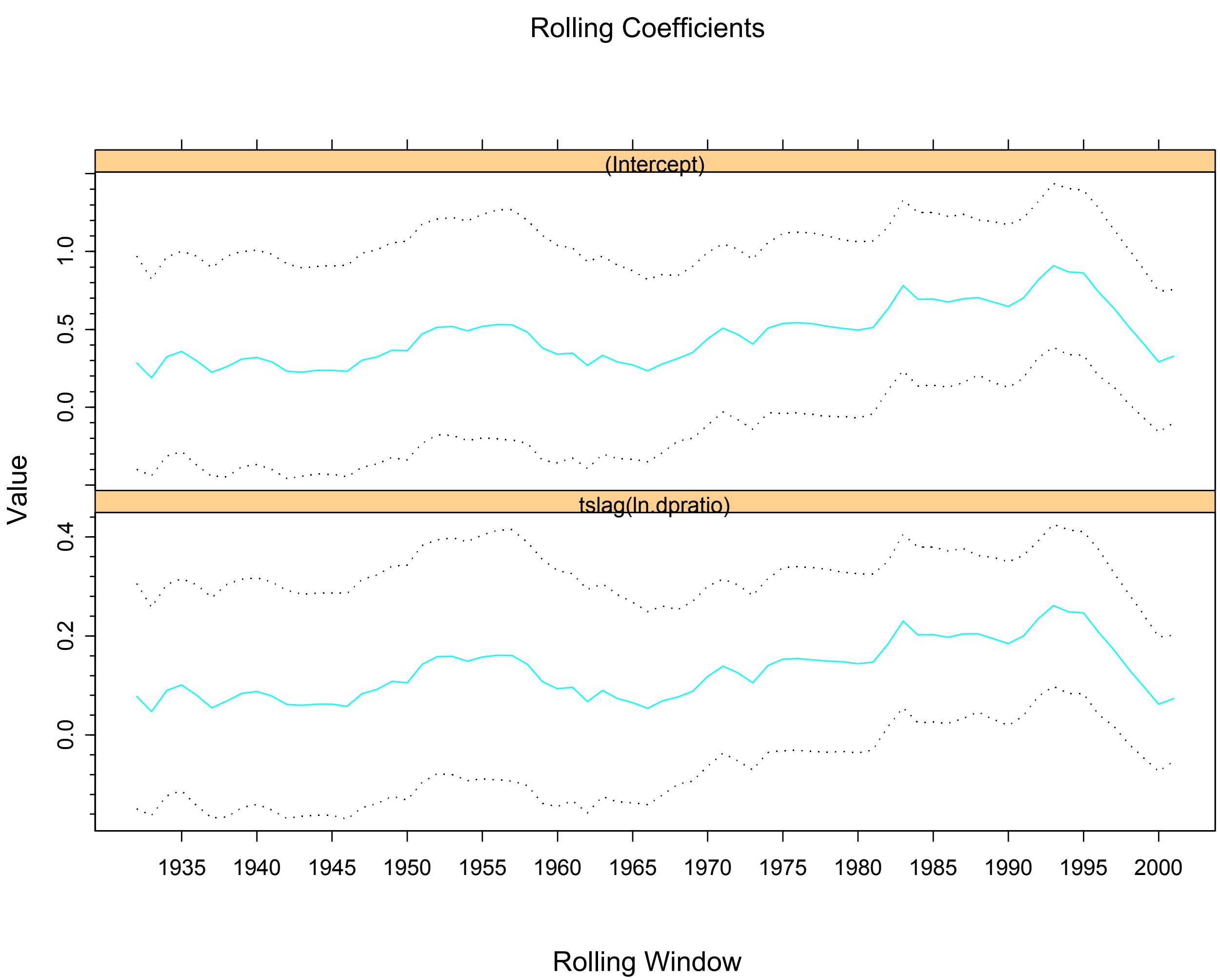
```
Coefficients:
(Intercept) tslag(ln.dpratio)
    mean  0.4588      0.1285
std. dev. 0.1841      0.0566

Coefficient Standard
   Deviations:
(Intercept) tslag(ln.dpratio)
    mean  0.3034      0.0996
std. dev. 0.0404      0.0164

Residual Scale Estimate:
 mean   std. dev.
 0.1912 0.0210
```

Summary Method

```
> summary(roll.dp.fit)

Call:
rollOLS(formula = ln.r ~
    tslag(ln.dpratio), data = stock.ts,
    width = 50,
    incr = 1)

Rolling Windows:
 number width increment
     70    50         1
Time period: from 1882 to 2000

Coefficient: (Intercept)
      Value Std. Error t value Pr(>|t|)
1931 0.2846     0.3425  0.8310   0.4101
1932 0.1897     0.3159  0.6006   0.5509
…
2000 0.3282     0.2139   1.535 0.131471
```

```
Coefficient: tslag(ln.dpratio)
       Value Std. Error t value Pr(>|t|)
1931 0.07812     0.1137  0.6869   0.4955
1932 0.04724     0.1053  0.4488   0.6556
…
1999 0.06204    0.06784  0.9145 0.365032
2000 0.07388    0.06346  1.1642 0.250097
```

Plot Method


Rolling Coefficients
(Intercept)
tslag(ln.dpratio)
Value
0.0
0.5
1.0
0.0
0.2
0.4
1935
1940
1945
1950
1955
1960
1965
1970
1975
1980
1985
1990
1995
2000
Rolling Window

Extractor Functions

- Generic extractor function `coefficients`

```
> coefficients(roll.dp.fit)
 Positions  (Intercept) tslag(ln.dpratio)
 1931       0.2846      0.07812
 1932       0.1897      0.04724
…
 2000       0.3282      0.07388
```

- Other common generic extractor functions are
  - `fitted` extracted fitted values
  - `residuals` extract residuals from fit
  - `vcov` extract coefficient covariance matrix
  - `IC` extract information criteria

Predicting and Backtesting

- Compute *h*-step predictions using `predict` method

```
> roll.dp.pred = predict(roll.dp.fit,n.steps=1:2)
> class(roll.dp.pred)
[1] "listof"
> names(roll.dp.pred)
[1] "1-Step-Ahead Forecasts" "2-Step-Ahead Forecasts"

> roll.dp.pred[1]
$"1-Step-Ahead Forecasts":
 Positions          1
 1932        0.102492
 1933        0.066315
…
 2000        0.015930
```

Prediction and Backtesting

- Forecast Evaluation Statistics

$$ME = \frac{1}{T-n-h+1}\sum_{t=n}^{T-h}\hat{\varepsilon}_{t+h|t}$$

$$MSE = \frac{1}{T-n-h+1}\sum_{t=n}^{T-h}\hat{\varepsilon}_{t+h|t}^{2}$$

$$RMSE = \sqrt{MSE(h)}$$

$$MAE = \frac{1}{T-n-h+1}\sum_{t=n}^{T-h}\left|\hat{\varepsilon}_{t+h|t}\right|$$

Prediction and Backtesting

- Functions to compute forecast errors and forecast evaluation statistics

```
> make.ehat = function(x,y){
+       ans = y - x
+       ans[!is.na(ans),]
+ }

> make.errorStats = function(x){
+       me = mean(x)
+       mse = as.numeric(var(x))
+       rmse = sqrt(mse)
+       mae = mean(abs(x))
+       ans = list(ME=me,MSE=mse,RMSE=rmse,MAE=mae)
+       ans
+ }
```

Prediction and Backtesting

- use `lapply` to apply functions to list variable

```
> ehat.dp.list = lapply(roll.dp.pred,make.ehat,
+ stock.ts[,"ln.r"])
> errorStats.dp.list =
  lapply(ehat.dp.list,make.errorStats)
> unlist(errorStats.dp.list)
 1-Step-Ahead Forecasts.ME 1-Step-Ahead Forecasts.MSE
                   0.03767                     0.0315
 1-Step-Ahead Forecasts.RMSE 1-Step-Ahead Forecasts.MAE
                   0.1775                      0.149
 2-Step-Ahead Forecasts.ME 2-Step-Ahead Forecasts.MSE
                   0.04424                    0.03223
 2-Step-Ahead Forecasts.RMSE 2-Step-Ahead Forecasts.MAE
                   0.1795                     0.1521
```

Agenda

- Introduction
  - Typical Applications in Finance
  - Overview of S-PLUS / S+FinMetrics™
- Time Series Objects in S-PLUS
- **Applications of S+FinMetrics™**
  - Rolling Analysis of Time Series
  - **Vector Autoregressive Models (VAR)**
- Conclusions
- Literature/References

VAR Models

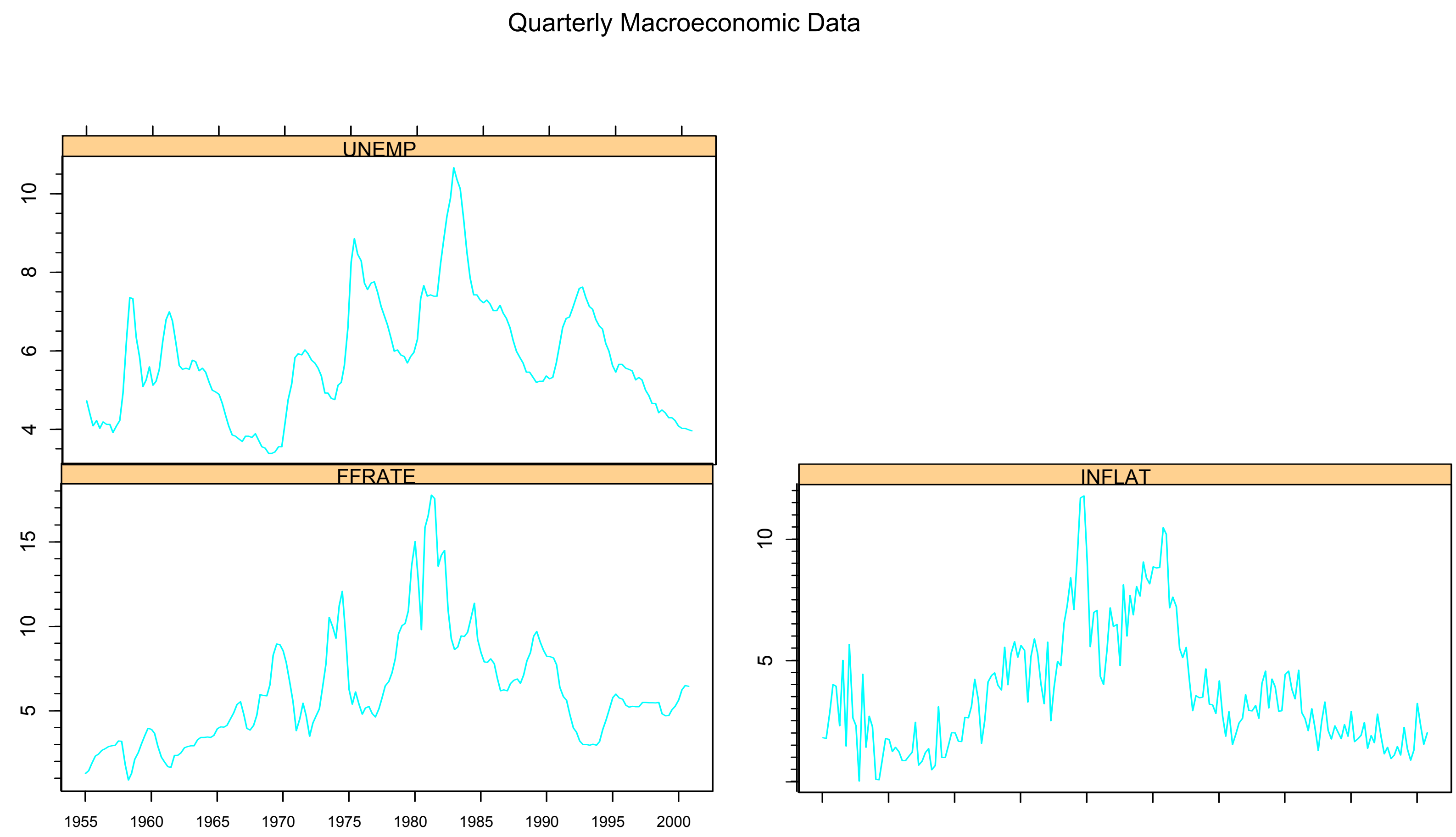
- Goal: Model and predict multivariate time series

$$Y_t = c + \Pi_1 Y_{t-1} + \Pi_2 Y_{t-2} + \cdots + Y_p \Pi_{t-p} + \varepsilon_t$$

$$\varepsilon_t \sim iid\ (0, \Sigma)$$

$$Y_t = (y_{1t}, \ldots, y_{nt})'$$

Example: Predicting Macroeconomic Data



Fitting the VAR

- Fit VAR model with specified lag length

```
> var.fit = VAR(cbind(FFRATE,INFLAT,UNEMP)~ar(4),
+ data=macro.ts)
```

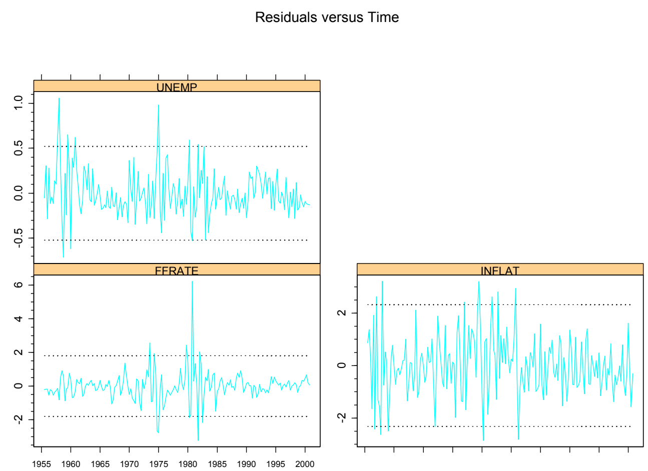
- Fit VAR model with automatic lag selection

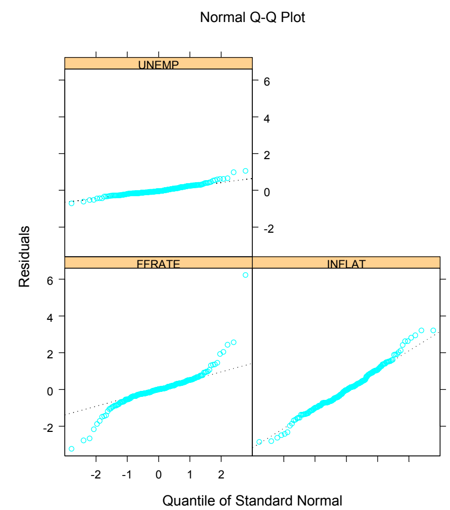
```
> var.fit =
+ VAR(macro.ts,max.ar=8,criterion="BIC")
> var.fit$info
        ar(1)    ar(2)   ar(3)    ar(4)
BIC 1194.074 1120.109 1137.44 1156.599
```

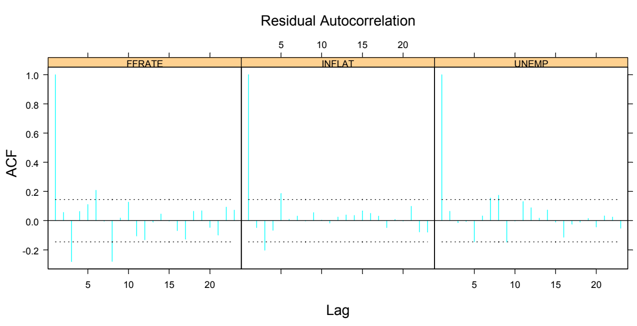
VAR Output

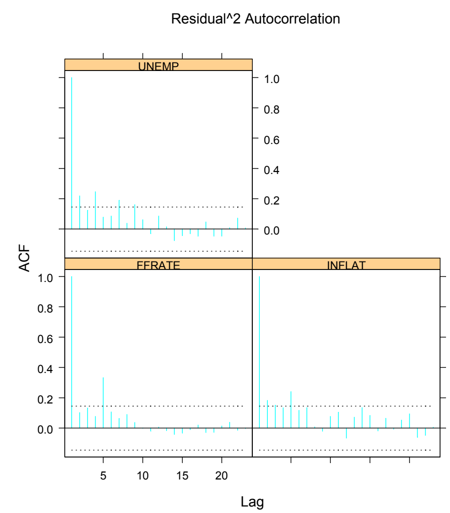
```
> summary(var.fit)

Call:
VAR(data = macro.ts, max.ar = 8, criterion =
    "BIC")

Coefficients:
               FFRATE   INFLAT    UNEMP
(Intercept)    0.3842   0.6790   0.2021
  (std.err)    0.2841   0.3659   0.0822
   (t.stat)    1.3523   1.8558   2.4575

FFRATE.lag1    0.9643   0.2767   0.0086
  (std.err)    0.0824   0.1061   0.0239
   (t.stat)   11.7007   2.6068   0.3603

INFLAT.lag1    0.0213   0.5172   0.0275
  (std.err)    0.0551   0.0710   0.0160
   (t.stat)    0.3855   7.2816   1.7216

 UNEMP.lag1   -1.0156  -0.1156   1.5056
  (std.err)    0.2378   0.3062   0.0688
   (t.stat)   -4.2711  -0.3776  21.8762

FFRATE.lag2   -0.0725  -0.2315   0.0245
  (std.err)    0.0834   0.1074   0.0241
   (t.stat)   -0.8696  -2.1563   1.0154

INFLAT.lag2    0.1319   0.3584  -0.0271
  (std.err)    0.0569   0.0732   0.0165
   (t.stat)    2.3204   4.8936  -1.6491

 UNEMP.lag2    0.9701   0.0308  -0.5749
  (std.err)    0.2270   0.2924   0.0657
   (t.stat)    4.2730   0.1052  -8.7485

Regression Diagnostics:
               FFRATE INFLAT  UNEMP
     R-squared 0.9261 0.7835 0.9708
Adj. R-squared 0.9235 0.7760 0.9698
  Resid. Scale 0.9004 1.1596 0.2606

Information Criteria:
      logL        AIC        BIC         HQ
 -510.1058  1062.2115  1129.4957  1089.4875

                   total  residual
Degree of freedom:   182       175
Time period: from 1955:III to 2000:IV
```

VAR `Plot` Method



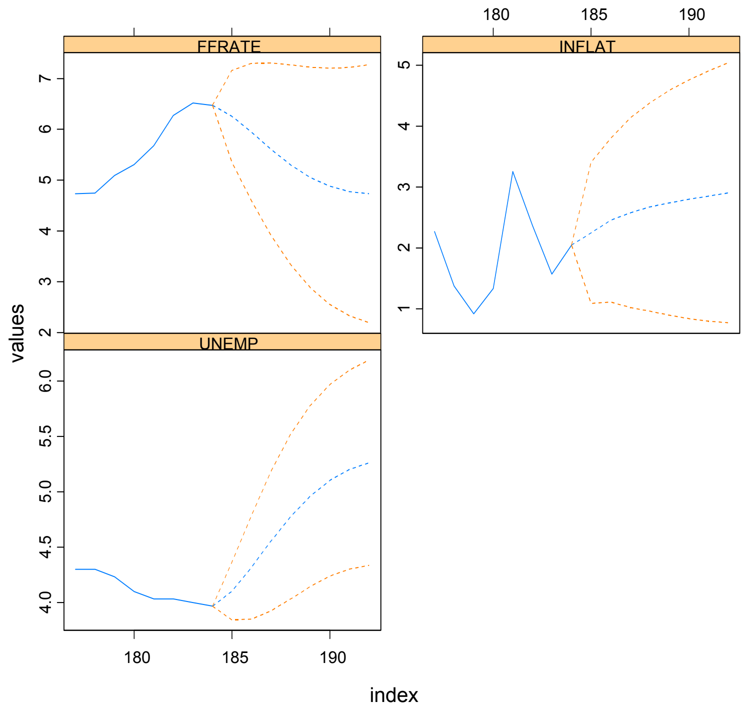






VAR `Predict` Method

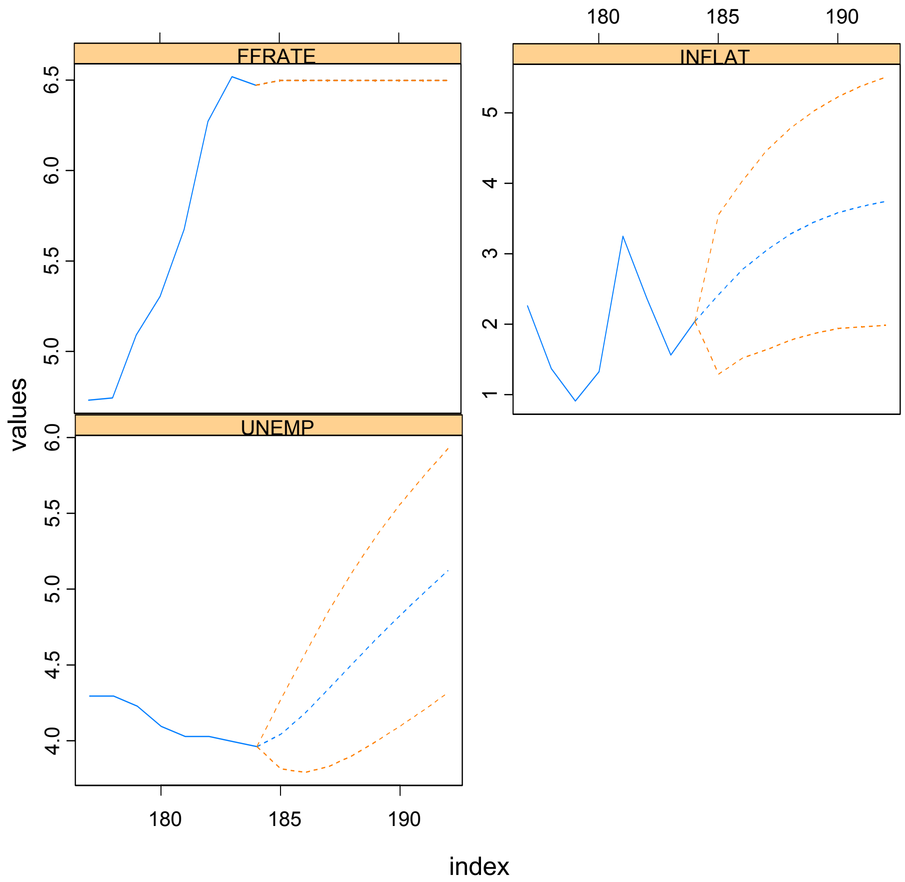
```
> var.pred =
   predict(var.fit,n.predict=8)
> var.pred

Predicted Values:

              FFRATE INFLAT  UNEMP
1-step-ahead 6.2560 2.2470 4.1039
2-step-ahead 5.9454 2.4546 4.3191
3-step-ahead 5.6067 2.5770 4.5566
4-step-ahead 5.3003 2.6721 4.7776
5-step-ahead 5.0534 2.7405 4.9623
6-step-ahead 4.8785 2.7980 5.1029
7-step-ahead 4.7743 2.8504 5.2006
8-step-ahead 4.7324 2.9029 5.2616
```



VAR Conditional Predictions

- Restrict future path of FFRATE to 6.5%

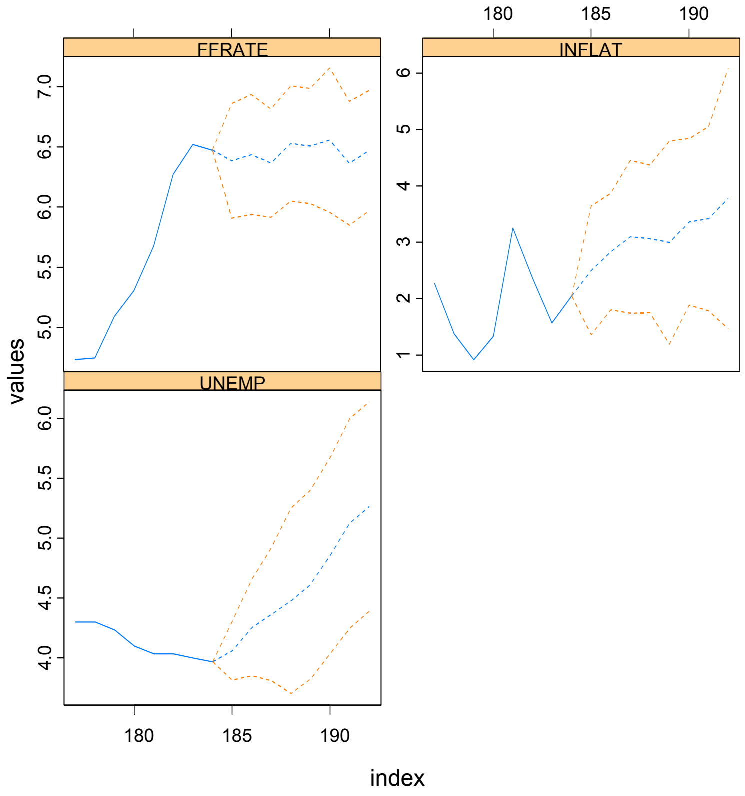
```
> var.cpred
+ cpredict(var.fit,
+ n.predict=8,
+ middle=rep(6.5,8),
+ variables="FFRATE",
+ steps=1:8)
```



VAR Conditional Predictions

- Restrict future path of FFRATE to (5.5%,7.5%)

```
> var.cpred =
+ cpredict(var.fit,
+ n.predict=8,
+ lower=rep(5.5,8),
+ upper=rep(7.5,8),
+ variables="FFRATE",
+ steps=1:8)
```



Conclusions

- S+FinMetrics™ fully exploits S-PLUS's powerful time series objects to deliver full set of tools for modeling financial time series
- Make cumbersome, complex problems easy to solve
  - Object orientation gives compact code
  - Rich collection of econometric functions speeds analysis projects

Contact Information/ References

- Eric Zivot (http://faculty.washington.edu/ezivot)
  - Zivot and Wang (2002). *Modeling Financial Time Series with S-PLUS*, Springer-Verlag

    Available from Amazon.com, Springer

- Insightful Corporation (www.insightful.com)
  - Sales, Consulting, Training
  - Contact: mpalmieri@insightful.com, cwedekind@insightful.com