Time Series in Stata

Outline

TS structure
Stata tools
Date Formats
import fred
Calendars
TS operators

Univariate models
ARIMA model
ARCH model

Multivariate models
VAR model

Forecast

Bayesian VAR
Estimation
Minnesota priors

Local projection IRFs

Summing up

References

Introduction to time-series analysis using Stata

Gustavo Sánchez

StataCorp LLC

May 9, 2023

Outline
TS structure
Stata tools
Date Formats
import fred
Calendars
TS operators
Univariate models
ARIMA model
ARCH model
Multivariate models
VAR model
Forecast
Bayesian VAR
Estimation
Minnesota priors
Local projection IRFs
Summing up
References

Outline

- Stata tools
- Inputing and managing time series in Stata
  - Date formats in Stata
  - Importing data `import fred`
  - Specifying the time structure and business calendar
  - Time series operators
- Univariate time-series analysis
  - ARIMA models
  - ARCH models
- Multivariate time-series analysis
  - Vector autoregressive (VAR) model
    - Forecasting systems of equations
  - Bayesian VAR models
  - Local projection IRFs

From OLS to Time Series

- Many of the results in OLS and cross-sectional models assume that $\{y_i, x_i\}_{i=1}^{n}$ are independent draws from the same distribution, like rolling dice n times, i.e, they are independent and identically distributed, i.i.d.
- In time series we are working with draws that depend on each other, e.g. the exchange rate today is not independent from the exchange rate yesterday, for instance.

Stata Tools

- **Data management**
- Univariate time series
  - **ARIMA (arimasoc new in Stata 18)**, ARFIMA, **ARCH**
  - Prais, Newey, unobserved component model (UCM)
  - Markov-Switching and threshold regressions
  - Time-series smoothers and filters
  - Diagnostic tools
- Multivariate time series.
  - Dynamic factor (dfactor)
  - Multivariate GARCH
  - State space
  - **VAR, BVAR**, SVAR, VEC models
  - Forecasting, inference, and interpretation
  - Diagnostic tools, graphs and more.
  - **Local projections (new in Stata18)**
- **Forecasting models**

Managing the time series structure in Stata

Outline
TS structure
Stata tools
Date Formats
import fred
Calendars
TS operators
Univariate models
ARIMA model
ARCH model
Multivariate models
VAR model
Forecast
Bayesian VAR
Estimation
Minnesota priors
Local projection IRFs
Summing up
References

Some date formats in Stata

Format	Description	Coding	
%td	daily (same as %d)	0 = 01jan1960,	1 = 02jan1960
%tw	weekly	0 = 1960w1,	1 = 1960w2
%tm	monthly	0 = 1960m1,	1 = 1960m2
%tq	quarterly	0 = 1960q1,	1 = 1960q2
%th	halfyearly	0 = 1960h1,	1 = 1960h2
%ty	yearly	1960 = 1960,	1961 = 1961
%tc	Datetime		

Note: times before 1960 are allowed. For instance, -1 means 31dec1959 in %td format and 1959q4 in %tq format.

- use **generic** for unformatted time variable
- use **datetime business calendars** for business-day data or other irregular time formats

Time Series in Stata

Outline

TS structure
Stata tools
Date Formats
import fred
Calendars
TS operators

Univariate models
ARIMA model
ARCH model

Multivariate models
VAR model

Forecast

Bayesian VAR
Estimation
Minnesota priors

Local projection IRFs

Summing up

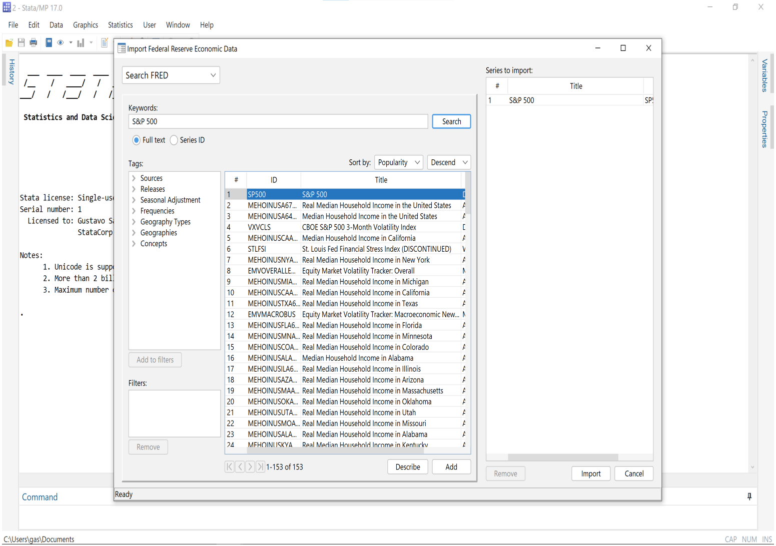
References

Example 1: Importing data from FRED

- We can use `import fred` to get data from the Federal Reserve Economic Data (FRED).

```
. import fred SP500 CBBTCUSD VIXCLS, daterange(2017-01-03 2023-04-30)
. rename SP500     sp500
. rename VIXCLS    vix
. rename CBBTCUSD  bitcoin

. describe

Contains data
 Observations:         2,309
    Variables:             5
--------------------------------------------------------------------------------
Variable      Storage   Display    Value
    name         type    format    label      Variable label
--------------------------------------------------------------------------------
datestr         str10   %-10s                 observation date
daten           int     %td                   numeric (daily) date
sp500           float   %9.0g                 S&P 500
bitcoin         float   %9.0g                 Coinbase Bitcoin
vix             float   %9.0g                 CBOE Volatility Index: VIX
--------------------------------------------------------------------------------
Sorted by: datestr
     Note: Dataset has changed since last saved.
```

`import fred`: Dialog box



Calendars and (non)missing data

- Specify the time series structure

```
. tsset daten,daily
Time variable: daten, 03jan2017 to 30apr2023
        Delta: 1 day
```

- Some processes imply that there is no data for certain periods. For example, Weekends and holidays in daily financial data.
- For some days there is no data, but there are no missing observations on the process

```
. list daten sp500 bitcoin vix  in 1/8,sep(4)
```

	daten	sp500	bitcoin	vix
1.	03jan2017	2257.83	1020.67	12.85
2.	04jan2017	2270.75	1130.3	11.85
3.	05jan2017	2269	1007	11.67
4.	06jan2017	2276.98	895.71	11.32
5.	07jan2017	.	909	.
6.	08jan2017	.	923.33	.
7.	09jan2017	2268.9	902.66	11.56
8.	10jan2017	2268.9	907	11.49

- Use **datetime business calendars**

Outline
TS structure
Stata tools
Date Formats
import fred
Calendars
TS operators
Univariate models
ARIMA model
ARCH model
Multivariate models
VAR model
Forecast
Bayesian VAR
Estimation
Minnesota priors
Local projection IRFs
Summing up
References

bcalstocks.stbcal

```
 version 18
dateformat dmy

range 03jan2017 28apr2023
centerdate 03jan2017

omit dayofweek (Sa Su)
omit date  1jan*
omit date  4jul*
omit dowinmonth +3 Mo of jan
omit dowinmonth +3 Mo of feb
omit dowinmonth -1 Mo of may
omit dowinmonth +1 Mo of sep
omit dowinmonth +4 Th of Nov
omit date 24dec2021
omit date 25dec*
omit date 14apr2017
omit date 30mar2018
omit date  5dec2018
omit date 19apr2019
omit date 10apr2020
omit date  3jul2020
omit date  2apr2021
omit date  5jul2021
omit date 15apr2022
omit date 20jun2022
omit date 26dec2022
omit date  2jan2023
omit date  7apr2023
```

Business Calendar for the stocks data

```
. generate bdate=bofd("bcalstocks",daten)
(718 missing values generated)

. format bdate %tbbcalstocks

. tsset bdate

Time variable: bdate, 03jan2017 to 28apr2023
        Delta: 1 day

. drop if bdate==.
(718 observations deleted)

. list daten bdate sp500 bitcoin vix  in 1/8,sep(4)
```

	daten	bdate	sp500	bitcoin	vix
1.	03jan2017	03jan2017	2257.83	1020.67	12.85
2.	04jan2017	04jan2017	2270.75	1130.3	11.85
3.	05jan2017	05jan2017	2269	1007	11.67
4.	06jan2017	06jan2017	2276.98	895.71	11.32
5.	09jan2017	09jan2017	2268.9	902.66	11.56
6.	10jan2017	10jan2017	2268.9	907	11.49
7.	11jan2017	11jan2017	2275.32	795.77	11.26
8.	12jan2017	12jan2017	2270.44	812.25	11.54

Preparing your data format with date functions

- Use the `date()` function to convert your string dates to a numerical date
- Use one of the following functions to convert your daily date to your desired frequency

`wofd(`%td_daily_date_exp`)`	returns %tw date
`mofd(`%td_daily_date_exp`)`	returns %tm date
`qofd(`%td_daily_date_exp`)`	returns %tq date
`hofd(`%td_daily_date_exp`)`	returns %th date
`yofd(`%td_daily_date_exp`)`	returns %ty date

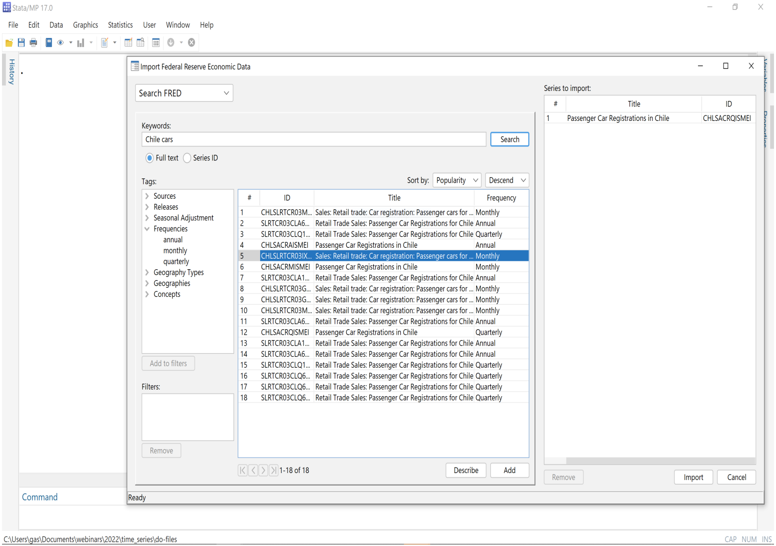
- Then use `tsset` to specify your time variable

Example 2: Passenger car registrations in Chile

- Get data from the Federal Reserve Economic Data (FRED).

```
 . import fred CHLSACRQISMEI, daterange(1990-01-01 2010-12-31) ///
>          aggregate(quarterly) clear
Summary
------------------------------------------------------------------------------
Series ID                   Nobs    Date range                 Frequency
------------------------------------------------------------------------------
CHLSACRQISMEI               68      1994-01-01 to 2010-10-01   Quarterly
------------------------------------------------------------------------------
# of series imported: 1
   highest frequency: Quarterly
    lowest frequency: Quarterly
.
. rename CHLSACRQISMEI  cars_cl
. describe
Contains data
 Observations:            68
    Variables:             3
------------------------------------------------------------------------------
Variable      Storage   Display    Value
    name         type    format    label      Variable label
------------------------------------------------------------------------------
datestr        str10    %-10s                 observation date
daten          int      %td                   numeric (daily) date
cars_cl        float    %9.0g                 Sales: Retail Trade: Car
                                                Registration: Passenger Cars
                                                for Chile
------------------------------------------------------------------------------
Sorted by: datestr
     Note: Dataset has changed since last saved.
```

`import fred`: Dialog box



Outline
TS structure
Stata tools
Date Formats
import fred
Calendars
TS operators
Univariate models
ARIMA model
ARCH model
Multivariate models
VAR model
Forecast
Bayesian VAR
Estimation
Minnesota priors
Local projection IRFs
Summing up
References

Example 2: Continue (Date functions)

```
. generate quarter = qofd(daten)
. format quarter %tq

.
. /* Specify time-series structure */
. tsset quarter
Time variable: quarter, 1994q1 to 2010q4
        Delta: 1 quarter

.
. list daten quarter cars_cl if tin(1994q1,1994q4)
```

	daten	quarter	cars_cl
1.	01jan1994	1994q1	39.41069
2.	01apr1994	1994q2	37.96296
3.	01jul1994	1994q3	37.82406
4.	01oct1994	1994q4	38.48565

Time-series operators

- Once you specify the time identifier variable with `tsset`, you can then use time-series operators on the variables
  - `L` is the lag operator; `L`k`.x`$=x_{t-k}$
  - `D` is the first-difference operator; `D.x` $= \Delta x_t = x_t - x_{t-1}$
    - `D`k`.x` $= \Delta^k x_t = \Delta \cdots (\Delta(\Delta x_t)) \cdots)$
  - `F` is lead operator; `F`k`.x` $= x_{t+k}$
  - `S` is the seasonal-difference operator; `S`k`.x` $= x_t - x_{t-k}$
  - `L(`*numlist*`).(`*varlist*`)`

```
. list quarter cars_cl L.cars_cl L2.cars_cl F.cars_cl D.cars_cl ///
>      if tin(1994q1,1994q4), noobs
```

quarter	cars_cl	L. cars_cl	L2. cars_cl	F. cars_cl	D. cars_cl
1994q1	39.41069	.	.	37.96296	.
1994q2	37.96296	39.41069	.	37.82406	-1.447731
1994q3	37.82406	37.96296	39.41069	38.48565	-.1388969
1994q4	38.48565	37.82406	37.96296	46.53998	.6615829

Time-series operators

- Once you specify the time identifier variable with `tsset`, you can then use time-series operators on the variables
  - `L` is the lag operator; `L`k`.x`$=x_{t-k}$
  - `D` is the first-difference operator; `D.x` $= \Delta x_t = x_t - x_{t-1}$
    - `D`k`.x` $= \Delta^k x_t = \Delta \cdots (\Delta(\Delta x_t)) \cdots)$
  - `F` is lead operator; `F`k`.x` $= x_{t+k}$
  - `S` is the seasonal-difference operator; `S`k`.x` $= x_t - x_{t-k}$
  - `L(`*numlist*`).(`*varlist*`)`

```
. list quarter cars_cl L.cars_cl L2.cars_cl F.cars_cl D.cars_cl ///
>       if tin(1994q1,1994q4), noobs
```

quarter	cars_cl	L. cars_cl	L2. cars_cl	F. cars_cl	D. cars_cl
1994q1	39.41069	.	.	37.96296	.
1994q2	37.96296	39.41069	.	37.82406	-1.447731
1994q3	37.82406	37.96296	39.41069	38.48565	-.1388969
1994q4	38.48565	37.82406	37.96296	46.53998	.6615829

Example 2: Continue (Time-series graphs)

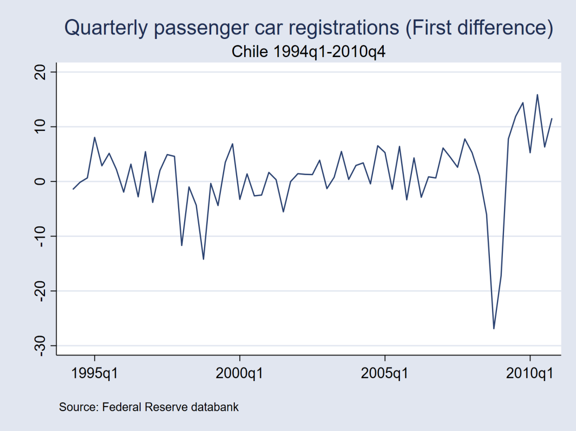
- You can also use the `tsline` command, which recognizes the time structure previously specified with `tsset`:

```
 . tsline cars_cl,                                              ///
 >    title("Quarterly total passenger car registrations")      ///
 >    subtitle("Chile 1994q1-2010q4") ytitle(" ") xtitle(" ")   ///
 >     note(Source: Federal Reserve databank)
```

Example 2: Continue (Time-series graphs)

- Let's also plot the first difference of the series:

```
. tsline D.cars_cl,
>    title("Quarterly passenger car registrations (First difference)")
>    subtitle("Chile 1994q1-2010q4") ytitle(" ") xtitle(" ")
>     note(Source: Federal Reserve databank)
```



Time Series in Stata

Outline

TS structure
Stata tools
Date Formats
import fred
Calendars
TS operators

Univariate models
ARIMA model
ARCH model

Multivariate models
VAR model

Forecast

Bayesian VAR
Estimation
Minnesota priors

Local projection IRFs

Summing up

References

ARIMA models

ARMA models

- In general, ARIMA models assume that we are working with weakly stationary time series.
- These models can be understood as combinations of white noise ε_t processes (which, in short, are stationary processes with zero mean, constant variance, and covariances that don't depend on t).
- These are the models we are considering here:

$$\begin{aligned} \text{AR(1):} \quad & y_t = \phi y_{t-1} + \varepsilon_t \\ \text{MA(1):} \quad & y_t = \varepsilon_t + \theta \varepsilon_{t-1} \end{aligned}$$

$$\begin{aligned} \text{AR(p):} \quad & y_t = \phi_1 y_{t-1} + \phi_2 y_{t-1} + \cdots + \phi_p y_{t-p} + \varepsilon_t \\ \text{MA(q):} \quad & y_t = \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-1} + \cdots + \theta_p \varepsilon_{t-q} \end{aligned}$$

$$\text{ARMA(p,q):} \quad y_t = \phi_1 y_{t-1} + \cdots + \phi_p y_{t-p} + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \cdots + \theta_p \varepsilon_{t-q}$$

ARMA models

- In general, ARIMA models assume that we are working with weakly stationary time series.
- These models can be understood as combinations of white noise ε_t processes (which, in short, are stationary processes with zero mean, constant variance, and covariances that don't depend on t).
- These are the models we are considering here:

$$\begin{aligned} \text{AR(1):} \quad & y_t = \phi y_{t-1} + \varepsilon_t \\ \text{MA(1):} \quad & y_t = \varepsilon_t + \theta \varepsilon_{t-1} \end{aligned}$$

$$\begin{aligned} \text{AR(p):} \quad & y_t = \phi_1 y_{t-1} + \phi_2 y_{t-1} + \cdots + \phi_p y_{t-p} + \varepsilon_t \\ \text{MA(q):} \quad & y_t = \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-1} + \cdots + \theta_p \varepsilon_{t-q} \end{aligned}$$

$$\text{ARMA(p,q):} \quad y_t = \phi_1 y_{t-1} + \cdots + \phi_p y_{t-p} + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \cdots + \theta_p \varepsilon_{t-q}$$

ARMA models

- In general, ARIMA models assume that we are working with weakly stationary time series.
- These models can be understood as combinations of white noise ε_t processes (which, in short, are stationary processes with zero mean, constant variance, and covariances that don't depend on t).
- These are the models we are considering here:

$$\text{AR(1):} \quad y_t = \phi y_{t-1} + \varepsilon_t$$
$$\text{MA(1):} \quad y_t = \varepsilon_t + \theta \varepsilon_{t-1}$$

$$\text{AR(p):} \quad y_t = \phi_1 y_{t-1} + \phi_2 y_{t-1} + \cdots + \phi_p y_{t-p} + \varepsilon_t$$
$$\text{MA(q):} \quad y_t = \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-1} + \cdots + \theta_p \varepsilon_{t-q}$$

$$\text{ARMA(p,q):} \quad y_t = \phi_1 y_{t-1} + \cdots + \phi_p y_{t-p} + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \cdots + \theta_p \varepsilon_{t-q}$$

Example 2: ARIMA model for Passenger car registrations

- Let's use the `corrgram` command to look at the correlogram for D.cars_cl:

```
. corrgram D.cars_cl,lags(20)

                                          -1       0       1 -1       0       1
 LAG       AC       PAC      Q     Prob>Q  [Autocorrelation]  [Partial autocor]
-------------------------------------------------------------------------------
1        0.3851   0.4003   10.388  0.0013
2        0.1556   0.0117   12.111  0.0023
3       -0.0876  -0.1742   12.665  0.0054
4       -0.2105  -0.1701   15.916  0.0031
5       -0.1665  -0.0143   17.983  0.0030
6       -0.1755  -0.1617   20.317  0.0024
7       -0.1792  -0.2584   22.792  0.0019
8       -0.0524  -0.0521   23.007  0.0034
9        0.1178   0.3449   24.113  0.0041
10       0.1141   0.1567    25.17  0.0050
11       0.1257   0.0633   26.475  0.0055
12       0.0403  -0.0801   26.611  0.0088
13      -0.0303  -0.1055    26.69  0.0137
14      -0.0219  -0.0203   26.732  0.0209
15      -0.0351   0.0194   26.841  0.0301
16      -0.1297  -0.3395   28.367  0.0286
17       0.0119   0.1622    28.38  0.0407
18       0.0189   0.0500   28.414  0.0560
19      -0.0014  -0.2319   28.414  0.0758
20       0.0002  -0.3739   28.414  0.0999
```

Example 2: Continue (Model selection)

- Let's now use the new (in Stata 18) command -arimasoc- to select the AR MA lag combination that best fit the model D.cars_cl:

```
. arimasoc D.cars_cl
Fitting models (9): ......... done
Lag-order selection criteria
Sample: 1994q2 thru 2010q4                          Number of obs = 67
```

Model	LL	df	AIC	BIC	HQIC
ARMA(0,0)	-221.561	2	447.1219	451.5313	448.8667
ARMA(0,1)	-217.0645	3	440.129	446.7431	442.7463
ARMA(0,2)	-215.5533	4	439.1067	447.9255	442.5963
ARMA(1,0)	-216.0282	3	438.0565	444.6705	440.6737
ARMA(1,1)	-216.0261	4	440.0521	448.8709	443.5417
ARMA(1,2)	-215.466	5	440.932	451.9554	445.294
ARMA(2,0)	-216.0243	4	440.0486	448.8674	443.5382
ARMA(2,1)	-214.7625	4	437.525	446.3437	441.0146
ARMA(2,2)	-214.6133	6	441.2265	454.4547	446.4609

```
Selected (max) LL:   ARMA(2,2)
Selected (min) AIC:  ARMA(2,1)
Selected (min) BIC:  ARMA(1,0)
Selected (min) HQIC: ARMA(1,0)
```

Example 2: Continue (Final model)

- We now fit the selected model (AR1):

```
. arima D.cars_cl,ar(1) nolog
ARIMA regression
Sample: 1994q2 thru 2010q4                     Number of obs     =         67
                                               Wald chi2(1)      =      17.27
Log likelihood = -216.0282                     Prob > chi2       =     0.0000
```

D.cars_cl	Coefficient	OPG std. err.	z	P>|z|	[95% conf.	interval]
cars_cl						
_cons	1.247205	1.410933	0.88	0.377	-1.518172	4.012582
ARMA						
ar L1.	.3948356	.0950154	4.16	0.000	.2086088	.5810624
/sigma	6.074494	.3603246	16.86	0.000	5.368271	6.780717

```
Note: The test of the variance against zero is one sided, and the two-sided
      confidence interval is truncated at zero.
```

Outline
TS structure
Stata tools
Date Formats
import fred
Calendars
TS operators
Univariate models
ARIMA model
ARCH model
Multivariate models
VAR model
Forecast
Bayesian VAR
Estimation
Minnesota priors
Local projection IRFs
Summing up
References

Autoregressive coditional heteroskedasticity

Conditional Heteroskedasticity

- So far we have not modelled variances explicitly. In the stationary case we assumed they were constant.
- Evidently, time series experience different degrees of volatility in different periods.
- It is important not only to model the mean but the variance of a time series.
- Autoregressive conditional heteroskedasticity models capture this idea.

ARCH MODELS

- The general framework can be written as:

$$
\begin{aligned}
y_t &= x_t\beta + ARMA(p,q) + \varepsilon_t \\
Var\left(\sigma_t^2\right) &= \gamma_0 + A(\sigma,\varepsilon) + B(\sigma,\varepsilon)^2
\end{aligned}
$$

- The original ARCH(m) model was written as:

$$
\begin{aligned}
y_t &= x_t\beta + \varepsilon_t \\
\sigma_t^2 &= \gamma_0 + \gamma_1\varepsilon_{t-1}^2 + \gamma_2\varepsilon_{t-2}^2 + \ldots + \gamma_m\varepsilon_{t-1}^2
\end{aligned}
$$

- The original GARCH(m,k) model is given by:

$$
\begin{aligned}
y_t &= x_t\beta + \varepsilon_t \\
\sigma_t^2 &= \gamma_0 + \gamma_1\varepsilon_{t-1}^2 + \gamma_2\varepsilon_{t-2}^2 + \ldots + \gamma_m\varepsilon_{t-1}^2 + \delta_1\sigma_{t-1}^2 + \delta_2\sigma_{t-2}^2 + \ldots + \delta_k\sigma_{t-k}^2
\end{aligned}
$$

ARCH MODELS

- The general framework can be written as:

$$\begin{aligned} y_t &= x_t\beta + ARMA(p,q) + \varepsilon_t \\ Var\left(\sigma_t^2\right) &= \gamma_0 + A(\sigma,\varepsilon) + B(\sigma,\varepsilon)^2 \end{aligned}$$

- The original ARCH(m) model was written as:

$$\begin{aligned} y_t &= x_t\beta + \varepsilon_t \\ \sigma_t^2 &= \gamma_0 + \gamma_1\varepsilon_{t-1}^2 + \gamma_2\varepsilon_{t-2}^2 + \ldots + \gamma_m\varepsilon_{t-1}^2 \end{aligned}$$

- The original GARCH(m,k) model is given by:

$$\begin{aligned} y_t &= x_t\beta + \varepsilon_t \\ \sigma_t^2 &= \gamma_0 + \gamma_1\varepsilon_{t-1}^2 + \gamma_2\varepsilon_{t-2}^2 + \ldots + \gamma_m\varepsilon_{t-1}^2 + \delta_1\sigma_{t-1}^2 + \delta_2\sigma_{t-2}^2 + \ldots + \delta_k\sigma_{t-k}^2 \end{aligned}$$

ARCH MODELS

- The general framework can be written as:

$$\begin{aligned} y_t &= x_t\beta + ARMA(p,q) + \varepsilon_t \\ Var\left(\sigma_t^2\right) &= \gamma_0 + A(\sigma,\varepsilon) + B(\sigma,\varepsilon)^2 \end{aligned}$$

- The original ARCH(m) model was written as:

$$\begin{aligned} y_t &= x_t\beta + \varepsilon_t \\ \sigma_t^2 &= \gamma_0 + \gamma_1\varepsilon_{t-1}^2 + \gamma_2\varepsilon_{t-2}^2 + \ldots + \gamma_m\varepsilon_{t-1}^2 \end{aligned}$$

- The original GARCH(m,k) model is given by:

$$\begin{aligned} y_t &= x_t\beta + \varepsilon_t \\ \sigma_t^2 &= \gamma_0 + \gamma_1\varepsilon_{t-1}^2 + \gamma_2\varepsilon_{t-2}^2 + \ldots + \gamma_m\varepsilon_{t-1}^2 + \delta_1\sigma_{t-1}^2 + \delta_2\sigma_{t-2}^2 + \ldots + \delta_k\sigma_{t-k}^2 \end{aligned}$$

Example 3: ARCH model for Mexican Pesos to U.S. Dollar

- Get data from the Federal Reserve Economic Data (FRED).

```
. import fred EXMXUS, daterange(2000-01-01 .) clear
Summary
```

Series ID	Nobs	Date range	Frequency
EXMXUS	280	2000-01-01 to 2023-04-01	Monthly

```
# of series imported: 1
   highest frequency: Monthly
    lowest frequency: Monthly
.
. generate month=mofd(daten)
. label var month "numeric (monthly) date"
. rename EXMXUS exchrate_mx
.
. tsset month,monthly
Time variable: month, 2000m1 to 2023m4
        Delta: 1 month
.
. list if tin(2023m1,2023m4),abbreviate(12)
```

	datestr	daten	exchrate_mx	month
277.	2023-01-01	01jan2023	18.9705	2023m1
278.	2023-02-01	01feb2023	18.6368	2023m2
279.	2023-03-01	01mar2023	18.3892	2023m3
280.	2023-04-01	01apr2023	18.0948	2023m4

Time Series in Stata

Outline

TS structure
Stata tools
Date Formats
import fred
Calendars
TS operators

Univariate models
ARIMA model
ARCH model

Multivariate models
VAR model

Forecast

Bayesian VAR
Estimation
Minnesota priors

Local projection IRFs

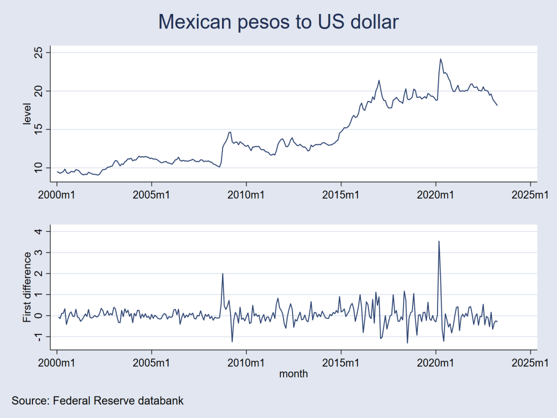
Summing up

References

Example 3: Continue (Time-series graph)

- You can combine two `tsline` graphs to plot the levels and first difference of the exchange rate series:

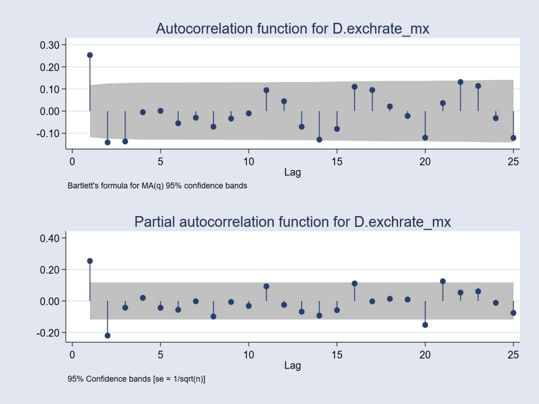
```
 . tsline exchrate_mx, name(level,replace) ytitle(level) xtitle("")
. tsline D.exchrate_mx, name(difference,replace)                ///
>                       ytitle(First difference) xtitle("month")
. graph combine level difference,rows(2)                        ///
>      title("Mexican pesos to US dollar")                      ///
>      note(Source: Federal Reserve databank)
```



Example 3: (Continue)

- Use `ac` and `pac` to look at the autocorrelation and partial autocorrelation functions for dexchrate_mx:

```
 . ac D.exchrate_mx,lags(25) ytitle(" ")  name(ac_dexchrate_mx,replace)   //
>    title("Autocorrelation function for D.exchrate_mx")
. pac D.exchrate_mx,lags(25) ytitle(" ") name(pac_dexchrate_mx,replace)   //
>     title("Partial autocorrelation function for D.exchrate_mx")
.
. graph combine ac_dexchrate_mx pac_dexchrate_mx,rows(2)
```



Example 3: Continue (Model fit)

- We now fit the selected model:

```
. arch D.exchrate_mx,arch(1) garch(1) ar(1/2) nolog vsquish
ARCH family regression -- AR disturbances
Sample: 2000m2 thru 2023m4                      Number of obs     =        279
                                                Wald chi2(2)      =      18.90
Log likelihood = -128.7685                      Prob > chi2       =     0.0001

------------------------------------------------------------------------------
D.           |                 OPG
 exchrate_mx | Coefficient  std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
exchrate_mx  |
       _cons |   .0421098   .0306477     1.37   0.169    -.0179585    .1021782
-------------+----------------------------------------------------------------
ARMA         |
          ar |
         L1. |   .3574877   .0855923     4.18   0.000     .1897299    .5252456
         L2. |  -.1697136    .076158    -2.23   0.026    -.3189805   -.0204468
-------------+----------------------------------------------------------------
ARCH         |
        arch |
         L1. |   .1005962   .0233218     4.31   0.000     .0548863     .146306
       garch |
         L1. |    .911545   .0113217    80.51   0.000     .8893548    .9337352
       _cons |   .0021427    .000621     3.45   0.001     .0009255    .0033598
------------------------------------------------------------------------------
```

Time Series in Stata

Outline

TS structure

Stata tools

Date Formats

import fred

Calendars

TS operators

Univariate models

ARIMA model

ARCH model

Multivariate models

VAR model

Forecast

Bayesian VAR

Estimation

Minnesota priors

Local projection IRFs

Summing up

References

Vector autoregressive (VAR) model

Vector Autoregressive Models VAR

- VARs are extensions of AR(p) models for vector valued dependent variables with no structural form.
- A VAR model can be written as:

$$\mathbf{y_t} = \mathbf{v} + \mathbf{A_1 y_{t-1}} + \ldots + \mathbf{A_p y_{t-p}} + \mathbf{C_0 x_t} + \mathbf{C_1 x_{t-1}} + \ldots + \mathbf{C_s x_{t-s}} + \mathbf{u_t}$$

Where:

$\mathbf{y}_t = (y_{1t}, ..., y_{Kt})$ is a Kx1 random vector
$\mathbf{A_1}$ through $\mathbf{A_p}$ are KxK matrices of parameters.
$\mathbf{x_t}$ is an Mx1 vector of exogenous variables
$\mathbf{C_0}$ through $\mathbf{C_s}$ are KxM matrices of parameters.
$\mathbf{v}$ is a Kx1 vector of parameters
$\mathbf{u_t}$ is a vector assumed to be white noise:

$$\begin{aligned} E(\mathbf{u_t}) &= \mathbf{0} \\ E(\mathbf{u_t u_t}') &= \mathbf{\Sigma} \\ E(\mathbf{u_t u_s}') &= \mathbf{0} \;;\quad t \neq s \end{aligned}$$

- The number of coefficients is quadratic to the number of dependent variables and proportional the number of lags.

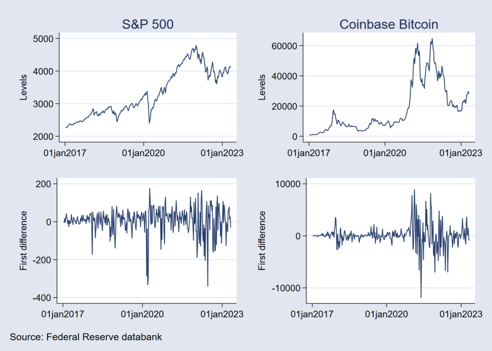
Example 4: Vector autoregressive (VAR) model for stock data

- We use `import fred` to get data from the Federal Reserve Economic Data (FRED).

```
 . import fred SP500 CBBTCUSD VIXCLS, daterange(2017-01-02 2023-04-28) ///
>    aggregate(weekly,avg) clear
. rename SP500 sp500
. rename CBBTCUSD bitcoin
. rename VIXCLS vix

 . describe
Contains data
 Observations:          330
    Variables:            5
-----------------------------------------------------------------------------
Variable      Storage   Display    Value
    name         type    format    label      Variable label
-----------------------------------------------------------------------------
datestr         str10   %-10s                 observation date
daten           int     %td                   numeric (daily) date
sp500           float   %9.0g                 S&P 500
bitcoin         float   %9.0g                 Coinbase Bitcoin
vix             float   %9.0g                 CBOE Volatility Index: VIX
-----------------------------------------------------------------------------
Sorted by: datestr
     Note: Dataset has changed since last saved.

.
. tsset daten,delta(7 days)
Time variable: daten, 06jan2017 to 28apr2023
        Delta: 7 days
```

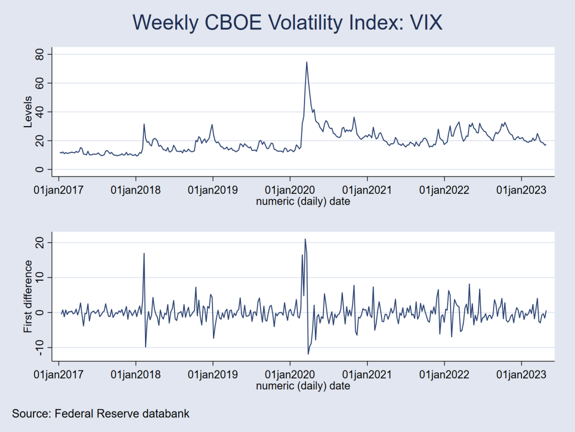
Endogenous variables for VAR model

```
 . tsline sp500,name(sp_levels,replace) ytitle("Levels")          ///
>      title("S&P 500") ylabels(#3,angle(0)) xlabels(#3) xtitle("")
. tsline D.sp500,name(sp_diff,replace) ytitle("First difference")   ///
>      ylabels(#3,angle(0)) xlabels(#3) xtitle("")
. tsline bitcoin,name(bitcoin_levels,replace) ytitle("Levels")      ///
>      title("Coinbase Bitcoin") ylabels(#3,angle(0)) xlabels(#3) xtitle("")
. tsline D.bitcoin,name(bitcoin_diff,replace) ytitle("First difference") ///
>      ylabels(#3,angle(0)) xlabels(#3) xtitle("")
.
. graph combine sp_levels bitcoin_levels sp_diff bitcoin_diff,rows(2)  ///
>      note(Source: Federal Reserve databank)
```



Exogenous variable for VAR model

```
. tsline vix,name(vix_levels,replace) ytitle("Levels")
. tsline D.vix,name(vix_diff,replace) ytitle("First difference")
.
. graph combine vix_levels vix_diff,             ///
>       title("Weekly CBOE Volatility Index: VIX")       ///
>       note(Source: Federal Reserve databank) rows(2)
```

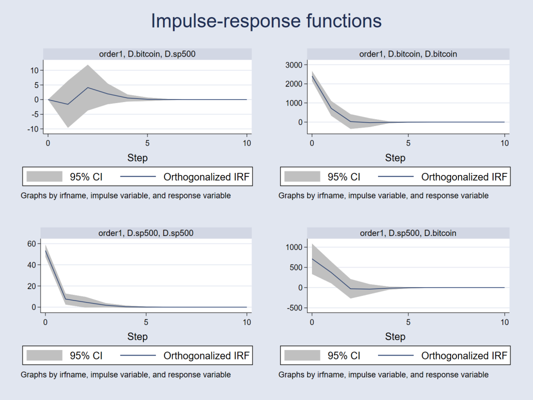


VAR model for S&P500 and Bitcoin with exogenous VIX variable

```
. var D.sp500 D.bitcoin if tin(06mar2020,28apr2023), exog(D.vix) vsquish
Vector autoregression
Sample: 06mar2020 thru 28apr2023                 Number of obs     =       165
Log likelihood =  -2409.346                      AIC               =  29.34965
FPE            =   1.91e+10                      HQIC              =  29.44134
Det(Sigma_ml)  =   1.65e+10                      SBIC              =  29.57553
Equation           Parms      RMSE     R-sq      chi2     P>chi2
----------------------------------------------------------------
D_sp500               6     54.3867   0.6144   262.9459   0.0000
D_bitcoin             6     2557.91   0.1530   29.79815   0.0000
----------------------------------------------------------------

------------------------------------------------------------------------------
             |
             | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
D_sp500      |
       sp500 |
         LD. |   .1520107   .0510943     2.98   0.003     .0518677    .2521536
        L2D. |   .0431388   .0512251     0.84   0.400    -.0572605    .1435382
     bitcoin |
         LD. |  -.0006655    .001696    -0.39   0.695    -.0039896    .0026586
        L2D. |   .0020031   .0016814     1.19   0.234    -.0012923    .0052985
         vix |
         D1. |  -17.52098   1.137303   -15.41   0.000    -19.75005   -15.29191
       _cons |   3.654321   4.168596     0.88   0.381    -4.515976    11.82462
-------------+----------------------------------------------------------------
D_bitcoin    |
       sp500 |
         LD. |   3.016874   2.403063     1.26   0.209    -1.693043    7.726791
        L2D. |  -2.044538   2.409215    -0.85   0.396    -6.766512    2.677436
     bitcoin |
         LD. |    .296882   .0797669     3.72   0.000     .1405417    .4532223
        L2D. |  -.0751003    .079078    -0.95   0.342    -.2300904    .0798897
         vix |
         D1. |  -148.4798   53.48954    -2.78   0.006    -253.3173    -43.6422
       _cons |   71.52681   196.0571     0.36   0.715     -312.738    455.7917
------------------------------------------------------------------------------

. estimates store eq_var
```

Impulse-response functions

```
 . irf create order1, step(10) set(myirf1,replace)
. irf graph oirf,impulse(D.bitcoin) response(D.sp500) name(i_bit__r_sp)
. irf graph oirf,impulse(D.bitcoin) response(D.bitcoin) name(i_bit__r_bit)
. irf graph oirf,impulse(D.sp500) response(D.sp500) name(i_sp__r_sp)
. irf graph oirf,impulse(D.sp500) response(D.bitcoin) name(i_sp__r_bit)
.
. graph combine i_bit__r_sp i_bit__r_bit i_sp__r_sp i_sp__r_bit, ///
>       title("Impulse-response functions")
```



Solving and forecasting systems of equations

Solving models for a collection of equations

- Components
  - Stochastic equations fit using estimation commands
  - Identities
  - Coefficient vectors
- Solving the model
  - Obtain static or dynamic forecasts
  - Alternative forecast scenarios
- `forecast` command

`forecast` subcommands

- Building the model
  - `create`
  - `estimates`
  - `identity`
  - `coefvector`
  - `exogenous`
- Solving the model
  - `solve`
  - `adjust`
- Utilities
  - `describe`
  - `list`
  - `clear`
  - `drop`
  - `query`

Example 5: System of equations VAR and ARIMA

- We now combine the results for the VAR model in example 4 with an ARIMA model for the exogenous variable vix.
- Let's fit an ARIMA model for vix and store the results:

```
. arima D.vix if tin(06mar2020,28apr2023), ar(1) nolog
ARIMA regression
Sample: 06mar2020 thru 28apr2023                Number of obs     =        165
                                                Wald chi2(1)      =      38.29
Log likelihood = -445.022                       Prob > chi2       =     0.0000

------------------------------------------------------------------------------
             |                 OPG
       D.vix | Coefficient  std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
vix          |
       _cons |  -.0778825    .436526    -0.18   0.858    -.9334576    .7776927
-------------+----------------------------------------------------------------
ARMA         |
          ar |
         L1. |   .2263361   .0365752     6.19   0.000       .15465    .2980221
-------------+----------------------------------------------------------------
      /sigma |   3.589633    .093999    38.19   0.000     3.405398    3.773868
------------------------------------------------------------------------------
Note: The test of the variance against zero is one sided, and the two-sided
      confidence interval is truncated at zero.

.
. estimates store eq_ar1
```

Example 5: Continue

- Use the suite of `forecast` commands to solve the model.

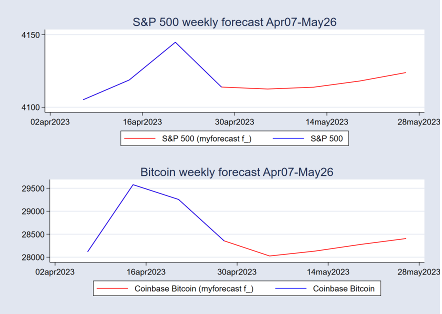
```
 . tsappend, add(4)
. forecast create myforecast
  Forecast model myforecast started.
. forecast estimates eq_var,names(dsp500 dbitcoin)
  Added estimation results from var.
  Forecast model myforecast now contains 2 endogenous variables.
. forecast estimates eq_ar1,names(dvix)
  Added estimation results from arima.
  Forecast model myforecast now contains 3 endogenous variables.
. forecast identity sp500=dsp500+L.sp500
  Forecast model myforecast now contains 4 endogenous variables.
. forecast identity bitcoin=dbitcoin+L.bitcoin
  Forecast model myforecast now contains 5 endogenous variables.
. forecast identity vix=dvix+L.vix
  Forecast model myforecast now contains 6 endogenous variables.
. forecast solve, begin(td(05may2023)) end(td(26may2023)) prefix(f_)
Computing dynamic forecasts for model myforecast.
-------------------------------------------------

Starting period:  05may2023
Ending period:    26may2023
Forecast prefix:  f_

05may2023:  .................
12may2023:  .................
19may2023:  .................
26may2023:  .................

Forecast 6 variables spanning 4 periods.
----------------------------------------
```

Example 5: Continue

- Let's look at the forecasts for the next 4 weeks.

```
. list daten sp500 f_sp500 bitcoin f_bitcoin vix f_vix ///
>       if tin(07apr2023,26may2023), noobs
```

daten	sp500	f_sp500	bitcoin	f_bitc~n	vix	f_vix
07apr2023	4105.13	4105.13	28114.17	28114.17	18.76	18.76
14apr2023	4118.77	4118.77	29576.95	29576.95	18.41	18.41
21apr2023	4144.8	4144.8	29255.55	29255.55	16.84	16.84
28apr2023	4113.9	4113.9	28355.44	28355.44	17.46	17.46
05may2023	.	4112.532	.	28025.55	.	17.54007
12may2023	.	4113.8	.	28132.04	.	17.49794
19may2023	.	4118.08	.	28276.94	.	17.42815
26may2023	.	4123.889	.	28405.1	.	17.3521



Bayesian VAR models with `bayes:var`

- Overparameterization in VAR models is particular problematic with small samples.
- Bayesian VAR allows shrinking the vector of regression coefficients by controlling the effective number of lags through the priors.
- The Minnesota family of priors represent a flexible specification that allows the expert's knowledge to be incorporated in the estimation.
- Bayes factors can be used to select the number of lags, and also the exogenous variables.

Time Series in Stata

Outline

TS structure
Stata tools
Date Formats
import fred
Calendars
TS operators

Univariate models
ARIMA model
ARCH model

Multivariate models
VAR model

Forecast

Bayesian VAR
Estimation
Minnesota priors

Local projection IRFs

Summing up

References

Bayesian VAR models with `bayes:var`

- The Bayesian approach to fit VAR models assigns prior distributions to all the regression parameters:
  - The likelihood is derived from the linear specification

$$\mathbf{y_t} = \mathbf{A_1 y_{t-1}} + \ldots + \mathbf{A_p y_{t-p}} + \mathbf{C x_t} + \mathbf{u_t} \quad ; \quad \mathbf{u_t} \sim N(0, \Sigma)$$

  - For the regression coefficients $\beta = vec(\mathbf{C}, \mathbf{A_1}, ..., \mathbf{A_p})$ the prior corresponds to a multivariate normal:

$$\beta | \boldsymbol{y} \sim N\left(\boldsymbol{\beta_0}, \boldsymbol{\Omega}\right)$$

  - For the regression covariance matrix $\boldsymbol{\Sigma}$ the prior distribution would be either inverse Wishart or Jeffreys.

Minnesota priors

- `bayes:var` has four prior family alternatives:
  - Minnesota prior with fixed covariance $\mathbf{\Sigma}$

    ```
    bayes, minnfixedcovprior...  :  var...
    ```

  - Conjugate Minnesota prior (The default)

    ```
    bayes, minnconjprior...  :  var...
    ```

  - Minnesota prior for β and inverse-Wishart prior for $\mathbf{\Sigma}$

    ```
    bayes, minnwishprior...  :  var...
    ```

  - Minnesota prior for β and Jeffreys prior for $\mathbf{\Sigma}$

    ```
    bayes, minnjeffprior...  :  var...
    ```

Conjugate prior with self-tightness equal to 1

```
 . matrix b0 = J(1,2,0)
. bayes,minnconjprior(mean(b0) selftight(1)) rseed(123) dryrun  ///
>    saving("$simul_dir\bvar_sim1",replace):                     ///
>    var D.sp500 D.bitcoin if tin(06mar2020,28apr2023), exog(D.vix)
Model summary
------------------------------------------------------------------------------
Likelihood:
  D_sp500 D_bitcoin ~ mvnormal(2,xb_D_sp500,xb_D_bitcoin,{Sigma,m})
Priors:
      {D_sp500:L( 2D).sp500}                                             (1)
    {D_sp500:L( 2D).bitcoin}                                             (1)
             {D_sp500:D.vix}                                             (1)
             {D_sp500:_cons}                                             (1)
    {D_bitcoin:L( 2D).sp500}                                             (2)
  {D_bitcoin:L( 2D).bitcoin}                                             (2)
           {D_bitcoin:D.vix}                                             (2)
           {D_bitcoin:_cons} ~ varconjugate(2,2,2,b0,{Sigma,m},_Phi0)    (2)
                   {Sigma,m} ~ iwishart(2,4,_Sigma0)
------------------------------------------------------------------------------
(1) Parameters are elements of the linear form xb_D_sp500.
(2) Parameters are elements of the linear form xb_D_bitcoin.
```

Conjugate prior / self-tightness equal to 1

```
 . matrix b0 = J(1,2,0)
. bayes, minnconjprior(mean(b0) selftight(1)) rseed(123) noheader  ///
>    nomodelsummary saving("$simul_dir\bvar_sim1",replace):        ///
>    var D.sp500 D.bitcoin if tin(06mar2020,28apr2023), exog(D.vix)

Burn-in ...
Simulation ...
```

	Mean	Std. dev.	MCSE	Median	Equal-tailed [95% cred.	interval]
D_sp500						
sp500						
LD.	.1505364	.0520951	.000521	.1513012	.0489193	.251928
L2D.	.0425588	.0508635	.000509	.0426899	-.0597908	.1421858
bitcoin						
LD.	-.0006188	.0017148	.000017	-.0006272	-.0039855	.0028131
L2D.	.0019468	.0016868	.000017	.0019415	-.0013687	.005238
vix						
D1.	-17.52243	1.162557	.011626	-17.51656	-19.8175	-15.25721
_cons	3.616037	4.180327	.042272	3.675909	-4.67629	11.93496
D_bitcoin						
sp500						
LD.	3.033556	2.37853	.023785	3.047389	-1.584201	7.774025
L2D.	-1.990006	2.375686	.022654	-1.994632	-6.59605	2.678747
bitcoin						
LD.	.2956686	.0804882	.000805	.2954231	.1401255	.4543287
L2D.	-.074123	.0786151	.000786	-.0747763	-.2278444	.0795755
vix						
D1.	-148.9566	54.49821	.544982	-149.034	-255.7674	-44.48812
_cons	69.01419	197.8763	1.97876	66.60244	-314.4022	462.218
Sigma_1_1	2918.457	316.7398	3.16393	2896.933	2366.812	3600.339
Sigma_2_1	37795.3	10907.73	109.077	37321.52	17190.46	60091.8
Sigma_2_2	6371910	704906.5	7049.06	6328732	5129745	7880294

```
file C:\Users\gas\Documents\webinars\2023\time_series\simul\bvar_sim1.dta saved
> .
```

Forecast and event probabilities for May 2023

```
 . bayesfcast compute bvar_, step(4) mcmcsaving("$simul_dir\fcast_mcmc",replace)
file C:\Users\gas\Documents\webinars\2023\time_series\simul\fcast_mcmc.dta save
> d.
. /* Use mcmc simulations for the predicted outcome variables */
.  use "$simul_dir\fcast_mcmc",clear
. /* Rename to identify daily predictions */
.   foreach var in D_sp500 D_bitcoin {
  2.      rename `var´_23135 `var´_May5
  3.      rename `var´_23142 `var´_May12
  4.      rename `var´_23149 `var´_May19
  5.   }
. /* Separate events for t+1 to t+3 (May5, May12, May19) */
. foreach var in D_sp500 D_bitcoin {
  2.     foreach date in May5 May12 May19 {
  3.        generate `var´_`date´_pos=cond(`var´_`date´>=0,1,0)
  4.     }
  5. }
. /* Combined events for one variable */
. foreach var in D_sp500 D_bitcoin {
  2.  gen `var´_pos=cond(`var´_May5>=0 & `var´_May12>=0 & `var´_May5>=0,1,0)
  3. }
. /* Combined events for both variables */
. foreach date in May5 May12 May19 {
  2.  gen sp_bc_`date´_pos=cond(D_sp500_`date´>=0 & D_bitcoin_`date´>=0,1,0)
  3. }
. generate sp_bc_pos=cond(D_sp500_May5>=0 & D_bitcoin_May5>=0   & ///
>                        D_sp500_May12>=0 & D_bitcoin_May12>=0 & ///
>                        D_sp500_May19>=0 & D_bitcoin_May19>=0,1,0)
```

Reporting event probabilities for May 2023

```
. collect preview
```

	May5	May12	May19	May5-May19
Result				
Change in SP500 > 0	0.506	0.503	0.507	0.281
Change in Bitcoin > 0	0.452	0.517	0.523	0.355
Change in SP500 & Bitcoin > 0	0.448	0.404	0.386	0.126

Local projection (LP) impulse-response functions

- Local projections Jordà (2005) estimate the IRF directly by running multistep regressions of response variables on impulse variables.

- The estimation is simplified for multiple endogenous variables and horizons.

- Confidence intervals have better coverage than the ones based on VAR's asymptotic distribution and the delta method (see references in the manual for simulations evidence).

- The sample size is reduced with local projection estimation because it conditions on H-1 trailing values to obtain the IRFs (H-1 corresponds to the maximum desired impulse-response step).

Outline
TS structure
Stata tools
Date Formats
import fred
Calendars
TS operators
Univariate models
ARIMA model
ARCH model
Multivariate models
VAR model
Forecast
Bayesian VAR
Estimation
Minnesota priors
Local projection IRFs
Summing up
References

Local projection (LP) impulse-response functions

- LP conditions on lags of models variables, specified in a series of regressions for each dependent variable and each horizon h=1,2,...,H.

$$y_{i,t+h-1} = \theta_{ijh} y_{j,t-1} + \mathbf{z_t}\delta + u_{t+h-1}$$

Where:

y_i is the response variable

y_j is the impulse variable

θ_{ijh} is the impulse-response coefficient

z_t additional control variables with associated δ (nuisance) parameters

u_{t+h-1} is the error term.

Local projection estimation IRF

```
. lpirf D.sp500 D.bitcoin if tin(06mar2020,28apr2023),exog(D.vix)
Local-projection impulse-responses
Sample: 06mar2020 thru 10mar2023                 Number of obs        = 158
                                                 Number of impulses   =   3
                                                 Number of responses  =   2
                                                 Number of controls   =   2

------------------------------------------------------------------------------
             |        IRF
             | coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
D.sp500      |
       sp500 |
         FD. |   .1524063   .0524176     2.91   0.004     .0496696     .255143
        F2D. |    .007786    .080536     0.10   0.923    -.1500617    .1656337
        F3D. |  -.0002672   .0791268    -0.00   0.997    -.1553528    .1548184
        F4D. |  -.1120585    .075293    -1.49   0.137      -.25963     .035513
        F5D. |  -.0488908   .0744123    -0.66   0.511    -.1947361    .0969546
        F6D. |  -.1525613   .0752067    -2.03   0.043    -.2999637    -.005159
        F7D. |   .0343996   .0744402     0.46   0.644    -.1115006    .1802998
        F8D. |  -.0367886   .0740862    -0.50   0.619     -.181995    .1084178
             |
     bitcoin |
         FD. |   3.182076   2.447082     1.30   0.193    -1.614118    7.978269
        F2D. |  -.6191867   2.595158    -0.24   0.811    -5.705603    4.467229
        F3D. |  -3.627633   2.618679    -1.39   0.166    -8.760149    1.504883
        F4D. |  -1.634605   2.612712    -0.63   0.532    -6.755426    3.486217
        F5D. |  -1.117906   2.659048    -0.42   0.674    -6.329544    4.093733
        F6D. |  -1.712254   2.673699    -0.64   0.522    -6.952608    3.528101
        F7D. |    .173183   2.669603     0.06   0.948    -5.059142    5.405508
        F8D. |   1.092022   2.667588     0.41   0.682    -4.136355    6.320398
-------------+----------------------------------------------------------------
D.bitcoin    |
       sp500 |
         FD. |  -.0006708   .0017484    -0.38   0.701    -.0040975     .002756
        F2D. |   .0018445   .0026863     0.69   0.492    -.0034204    .0071095
        F3D. |   .0005155   .0026393     0.20   0.845    -.0046573    .0056884
        F4D. |   .0041666   .0025114     1.66   0.097    -.0007556    .0090888
        F5D. |   .0018821    .002482     0.76   0.448    -.0029825    .0067468
        F6D. |   .0003632   .0025085     0.14   0.885    -.0045534    .0052798
        F7D. |   .0004155   .0024829     0.17   0.867    -.0044509     .005282
        F8D. |   -.002118   .0024711    -0.86   0.391    -.0069613    .0027254
             |
     bitcoin |
         FD. |   .2981045   .0816219     3.65   0.000     .1381284    .4580805
        F2D. |   .0108941    .086561     0.13   0.900    -.1587622    .1805505
        F3D. |   .0181803   .0873455     0.21   0.835    -.1530137    .1893744
        F4D. |    .183811   .0871465     2.11   0.035     .0130071    .3546149
        F5D. |   .1152146    .088692     1.30   0.194    -.0586185    .2890478
        F6D. |    .037528   .0891807     0.42   0.674     -.137263    .2123189
        F7D. |   .0151451    .089044     0.17   0.865    -.1593781    .1896682
        F8D. |   .0627068   .0889769     0.70   0.481    -.1116846    .2370982
------------------------------------------------------------------------------
```

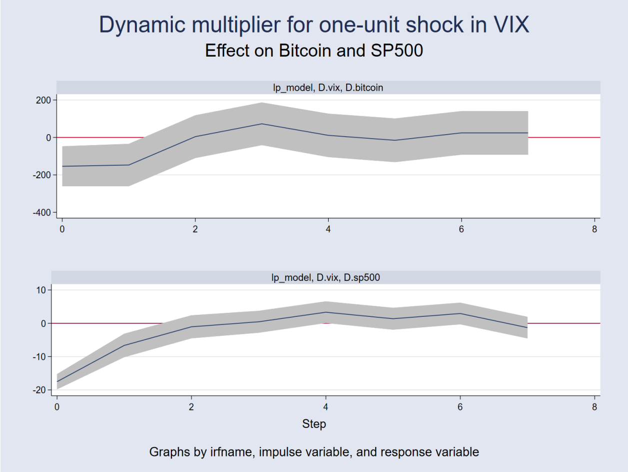
Local projection estimation IRF (Continue)

	IRF coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
D.vix						
sp500						
D1.	-17.51955	1.166784	-15.02	0.000	-19.8064	-15.23269
FD.	-6.647162	1.792682	-3.71	0.000	-10.16075	-3.13357
F2D.	-1.057563	1.761312	-0.60	0.548	-4.509671	2.394546
F3D.	.4510605	1.675974	0.27	0.788	-2.833789	3.73591
F4D.	3.322848	1.656371	2.01	0.045	.0764204	6.569275
F5D.	1.36846	1.674053	0.82	0.414	-1.912624	4.649545
F6D.	2.93284	1.656993	1.77	0.077	-.3148069	6.180488
F7D.	-1.298686	1.649114	-0.79	0.431	-4.530889	1.933518
bitcoin						
D1.	-153.9763	54.47053	-2.83	0.005	-260.7366	-47.216
FD.	-147.3984	57.7666	-2.55	0.011	-260.6188	-34.17791
F2D.	4.384475	58.29016	0.08	0.940	-109.8621	118.6311
F3D.	73.04049	58.15734	1.26	0.209	-40.94581	187.0268
F4D.	11.12316	59.18876	0.19	0.851	-104.8847	127.131
F5D.	-15.22615	59.51489	-0.26	0.798	-131.8732	101.4209
F6D.	24.45852	59.42369	0.41	0.681	-92.00978	140.9268
F7D.	24.44518	59.37886	0.41	0.681	-91.93524	140.8256

```
Note: IRF coefficients for exogenous variables are dynamic multipliers.
Impulses:  D.sp500 D.bitcoin D.vix
Responses: D.sp500 D.bitcoin
Controls:  L2D.bitcoin L2D.sp500
```

Local projection plot

- Let's look at the effect of one-unit shock in VIX on Bitcoin and the SP500:



Outline
TS structure
Stata tools
Date Formats
import fred
Calendars
TS operators
Univariate models
ARIMA model
ARCH model
Multivariate models
VAR model
Forecast
Bayesian VAR
Estimation
Minnesota priors
Local projection IRFs
Summing up
References

Summing up

- Stata tools
- Inputing and managing time series in Stata
- Univariate time-series analysis
  - ARIMA models
  - ARCH models
- Multivariate time-series analysis
  - Vector autoregressive (VAR) model
    - Forecasting systems of equations
  - Bayesian VAR models
  - Local projection IRFs

Time Series in Stata

Outline

TS structure

Stata tools

Date Formats

import fred

Calendars

TS operators

Univariate models

ARIMA model

ARCH model

Multivariate models

VAR model

Forecast

Bayesian VAR

Estimation

Minnesota priors

Local projection IRFs

Summing up

References

Proceedings for presentations on time series using Stata

- Technical tips on time series with Stata:
  https://www.stata.com/meeting/mexico11/materials/gsanchez.pdf
- Cointegrating VAR and probability forecasting:
  https://www.stata.com/meeting/spain12/abstracts/materials/Sanchez.pdf
- Forecasting tools in Stata
  https://www.stata.com/meeting/mexico13/abstracts/materials/mex13_sanchez_forecast.pdf
- Markov-switching regression models in Stata
  https://www.stata.com/meeting/spain15/abstracts/materials/spain15_sanchez.pdf
- Bayesian VAR models in Stata:
  https://www.stata.com/meeting/mexico21/slides/Mexico21_Sanchez.pdf

Reference

- Jordà, Ò . 2005. Estimation and inference of impulse responses by local projections. American Economic Review 95: 161-182. https://doi.org/10.1257/0002828053828518