Introduction to Time Series Using Stata

David Schenck
Senior Econometrician, Stata

PAA Workshop, Washington, D.C.
April 10, 2025

Workshop Introduction

- In this course, you will learn about some of Stata's capabilities for time-series analysis.
- You will also gain more proficiency with using Stata.

Workshop Schedule

1. Introduction
   - Time Series basics
   - Stata basics
   - Stata's time-series environment
2. ARMA models
3. Trends, seasons, and cycles
4. Multivariate time series models
   - Autoregressive Distributed Lag
   - Vector Autoregression

Table of Contents

Table of Contents

Main features of time-series data I

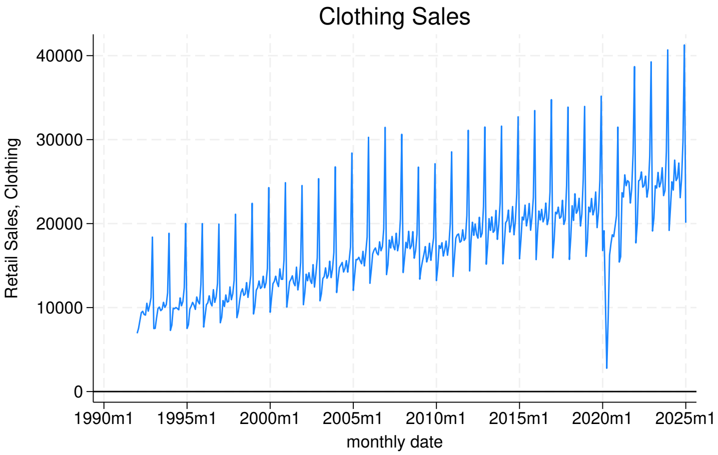
- A time series is a collection of observations made sequentially in time
- Time-series data has a natural ordering, unlike cross-section data
- This talk focuses on equally spaced discrete time data

Main features of time-series data II

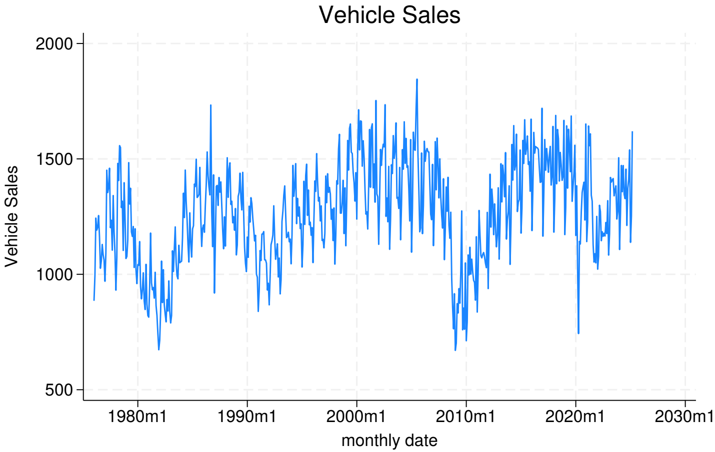
Plotting a time sries is a useful early step in analysis

- Trend: upward or downward patterns as time progresses
- Cycles: repetition of behavior in a regular patterns
- Seasonality: periodic behavior at a set period (monthly, quarterly, etc)
- Heteroskedasticity: changing variance
- Dependence: positive (successive observations are similiar) or negative (the reverse)
- Outliers, breaks, structural change . . .

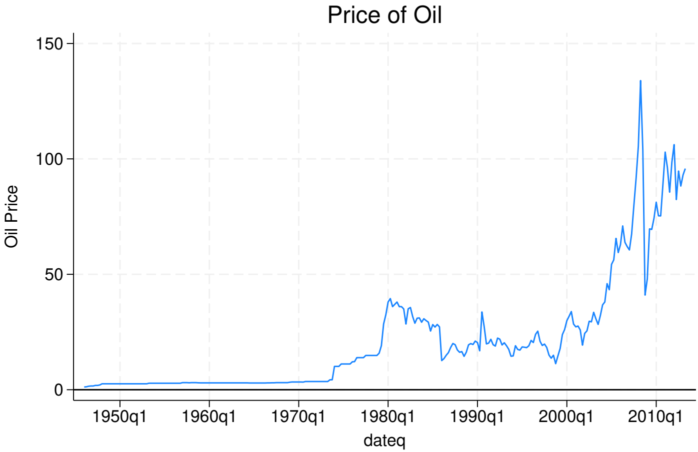
Some Example Time Series I



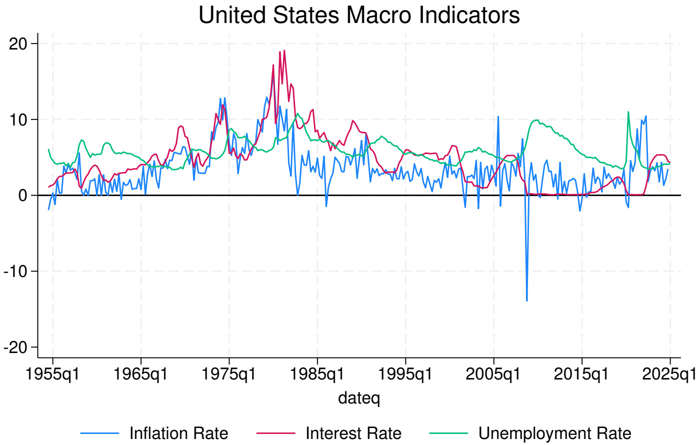
Some Example Time Series II



Some Example Time Series III



Some Example Time Series IV



Some Example Time Series V

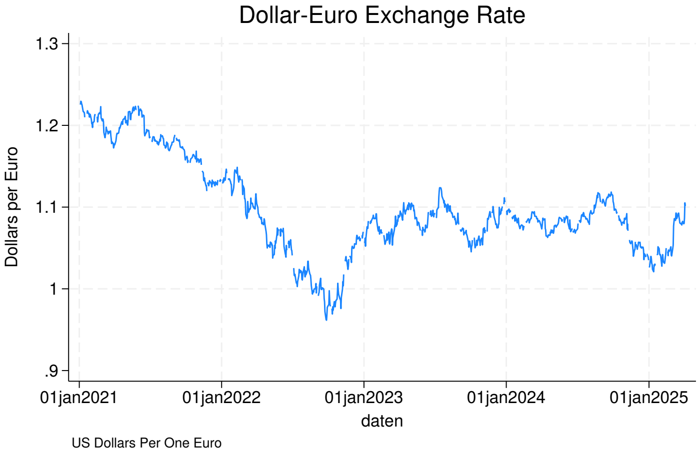


Table of Contents

Working with Stata

- Type a little, get a little
  - Type a little more, get a little more
- Simple reproducibility
- Easy and complete extensibility
- Easy sharing
- Web awareness

Typography

- As we go, you'll see Stata commands
- Commands to be typed as written are in `type font`
- e.g. `regress y x`
  - `x` and `y` are variables in your dataset
- Items in *italics* are to be substituted
- e.g. `regress` *yvar xvar*
  - *xvar* and *yvar* should be substituted with variables in your dataset

Getting around

- Open Stata from the desktop, Start menu, or Launchpad (Mac)
- Within Stata, type `pwd` to view the current working directory
- Type `dir` or `ls` to list files in the current directory
- Type `dir *.dta` to list all `.dta` files
- To change the directory, type `cd` *newdir*
    - where *newdir* is a directory of your choice

Log files

- Log files record every command you type and the output it produces
- Useful for documenting your work.
- If your work takes some time to run, saving a log to view later saves the cost of re-running the work.
- I use log files in conjunction with do-files.

Do-files

- do-files are just text files that contain Stata commands
- Usually end in `.do`
- Useful for reproducibility
- Combine do-files and log-files for reproducible, documented analysis

Table of Contents

Importing data

- Import Excel or text data:

```
. import excel usmacro.xls, firstrow
. import delimited usmacro.csv
. import delimited usmacro.txt
```

- Import Stata datasets:

```
. use usmacro
. use http://data.macrohistory.net/JST/JSTdatasetR5.dta
```

- Import datasets used in Stata manuals:

```
. webuse usmacro
```

- Import data from Federal Reserve Economic Database:

```
. import fred UNRATE
```

(`import fred` requires a FRED API key)

Dates and times: getting help

- [U] **12.3 – dates and times**
- [U] **25 – working with dates and times**
- [D] **datetime – date and time values and variables**

Working with dates I

- Stata has a suite of commands to handle dates and times. We start with daily dates as a baseline.
- Look at `exchange.dta`:

```
. use exchange, clear

. describe

Contains data from exchange.dta
 Observations:         6,850
    Variables:             2                  8 Apr 2025 10:56
-------------------------------------------------------------------------------
Variable      Storage   Display    Value
    name         type    format    label      Variable label
-------------------------------------------------------------------------------
datestr         str10   %-10s                 observation date
eurorate        float   %9.0g                 Dollars per Euro
-------------------------------------------------------------------------------
Sorted by: datestr
```

- It contains data on the dollar-Euro exchange rate
  - `datestr` is a string variable
  - `eurorate` has the daily price of dollars in terms of Euros

Working with dates II

```
. list in 1/5
```

	datestr	eurorate
1.	1999-01-04	1.1812
2.	1999-01-05	1.176
3.	1999-01-06	1.1636
4.	1999-01-07	1.1672
5.	1999-01-08	1.1554

- Our first goal is to declare to Stata that this is time-series data.
- To do that, we will need to create a numeric date variable.
- The `date()` function converts string dates into numeric dates

```
. generate daten = date(datestr, "YMD")

. list in 1/5
```

	datestr	eurorate	daten
1.	1999-01-04	1.1812	14248
2.	1999-01-05	1.176	14249
3.	1999-01-06	1.1636	14250
4.	1999-01-07	1.1672	14251
5.	1999-01-08	1.1554	14252

Working with dates III

- Now we declare to Stata that `daten` contains daily dates:

```
. tsset daten, daily

Time variable: daten, 04jan1999 to 04apr2025, but with gaps
        Delta: 1 day

. list in 1/5

     +----------------------------------+
     |    datestr   eurorate      daten |
     |----------------------------------|
  1. | 1999-01-04     1.1812  04jan1999 |
  2. | 1999-01-05      1.176  05jan1999 |
  3. | 1999-01-06     1.1636  06jan1999 |
  4. | 1999-01-07     1.1672  07jan1999 |
  5. | 1999-01-08     1.1554  08jan1999 |
     +----------------------------------+
```

- By the way, now you can run the previous exchange rate figure:

```
. tsline eurorate if year(daten) >= 2021, cmissing(n)
```

- `tsline` is a useful way to make simple graphs.
- Use `twoway` for more complicated graphs.

More on dates

- Notice the "but with gaps" note
  - Sometimes gaps are intentional
  - If not, see **datetime business calendars**
- If your data is annual, `tsset year` is sufficient
- Stata has formats for weekly, monthly, and quarterly dates
- Also support for times (hours, minutes, seconds, milliseconds . . .)
- Stata has many commands for extracting pieces of a date (month, quarter, year, . . .)

Example: monthly dates I

- Data:

```
. use monthly, clear

. describe

Contains data from monthly.dta
 Observations:           591
    Variables:             3                  8 Apr 2025 10:56
-------------------------------------------------------------------------------
Variable      Storage   Display    Value
    name         type    format    label      Variable label
-------------------------------------------------------------------------------
datestr         str7    %9s
vehicle         float   %9.0g                 Vehicle Sales
clothing        float   %9.0g                 Retail Sales, Clothing
-------------------------------------------------------------------------------
Sorted by:

. list datestr in 1/5

     +---------+
     | datestr |
     |---------|
  1. |  1976M1 |
  2. |  1976M2 |
  3. |  1976M3 |
  4. |  1976M4 |
  5. |  1976M5 |
     +---------+
```

Example: monthly dates II

- Notice we have a string date
- The `monthly()` function converts a string month variable into a numeric month date

```
. generate datem = monthly(datestr, "YM")

. lab var datem "monthly date"

. tsset datem, monthly

Time variable: datem, 1976m1 to 2025m3
        Delta: 1 month

. order datestr datem
```

- Looking at the results:

Example: monthly dates III

```
. describe

Contains data from monthly.dta
 Observations:           591
    Variables:             4                  8 Apr 2025 10:56
───────────────────────────────────────────────────────────────────────────────
Variable      Storage   Display    Value
    name         type    format    label      Variable label
───────────────────────────────────────────────────────────────────────────────
datestr         str7    %9s
datem           float   %tm                   monthly date
vehicle         float   %9.0g                 Vehicle Sales
clothing        float   %9.0g                 Retail Sales, Clothing
───────────────────────────────────────────────────────────────────────────────
Sorted by: datem
     Note: Dataset has changed since last saved.

. list datestr datem in 1/5

     ┌──────────────────┐
     │ datestr    datem │
     ├──────────────────┤
  1. │  1976M1   1976m1 │
  2. │  1976M2   1976m2 │
  3. │  1976M3   1976m3 │
  4. │  1976M4   1976m4 │
  5. │  1976M5   1976m5 │
     └──────────────────┘
```

After generating the date

- The `tsset` *timevar* command declares your data to be time-series
  - `tsset date`
  - `tsset datem, monthly`
- Stata will recognize lead, lag, and seasonal difference operators
- You can extract subcomponents (year, month, quarter, ...)
- Most time-series commands require the data to be `tsset` prior to use
- Example with quarterly data
- Also see: Ortiz, "Mastering Stata's datetime concepts and functions" US Stata Conference 2023.

Table of Contents

Stationarity

- A time series is called stationary if its unconditional mean μ and unconditional variance σ^2 are constant over time
- In such a case it makes sense to take averages, e.g.

$$\widehat{\mu} = \sum_{t=1}^{T} y_t$$

- Similarly we can compute the variance and the autocovariances

$$\widehat{\gamma}_h = \frac{1}{T} \sum_{t=1}^{T-|h|} (x_t - \widehat{\mu})(x_{t+h} - \widehat{\mu})$$

- These reduced-form statistics underlie model parameters and do not have meaning if the data is nonstationary
- Transformations to induce stationarity are discussed later

ARMA model basics

- ARMA stands for AutoRegressive Moving Average
- AR models explain y_t in terms of its own lags:

$$y_t = \phi_1 y_{t-1} + \cdots + \phi_p y_{t-p} + e_t$$

- MA models explain y_t in terms of lags of the shocks:

$$y_t = e_t + \theta_1 e_{t-1} + \cdots + \theta_q e_{t-q}$$

- ARMA models combine both.

$$y_t = \phi_1 y_{t-1} + \cdots + \phi_p y_{t-p} + e_t + \theta_1 e_{t-1} + \cdots + \theta_q e_{t-q}$$

The `arima` command

- Syntax: `arima` *depvar* [*indepvars*] `[if] [in] [, options]`
  - First variable is the dependent variable
  - Additional variables are covariates
- Important options:
  - `ar(`*numlist*`)`
  - `ma(`*numlist*`)`
  - `arima(`#*p*,#*d*,#*q*`)`

Fitting ARMA models with arima I

```
. use quarterly_clean.dta

. describe

Contains data from quarterly_clean.dta
 Observations:           317
    Variables:             8                  8 Apr 2025 09:14
-------------------------------------------------------------------------------
Variable      Storage   Display    Value
    name         type    format    label      Variable label
-------------------------------------------------------------------------------
datestr         str6    %9s
dateq           float   %tq
oilprice        float   %9.0g                 Oil Price
cpi             float   %9.0g                 Consumer Price Index
unrate          float   %9.0g                 Unemployment Rate
fedfunds        float   %9.0g                 Interest Rate
ln_cpi          float   %9.0g                 Log of price level
inflation       float   %9.0g                 Inflation Rate
-------------------------------------------------------------------------------
Sorted by: dateq
```

Fitting ARMA models with arima II

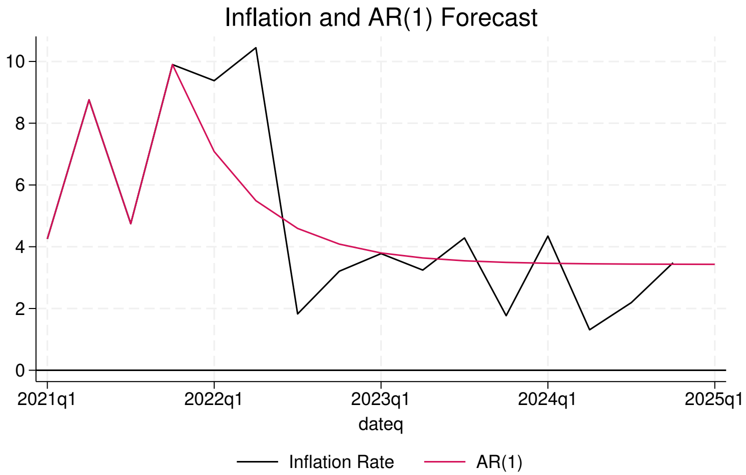
```
. arima inflation if tin(, 2021q4), ar(1) nolog

ARIMA regression

Sample: 1947q2 thru 2021q4                      Number of obs     =        299
                                                Wald chi2(1)      =     306.98
Log likelihood = -733.1644                      Prob > chi2       =     0.0000

------------------------------------------------------------------------------
             |                 OPG
   inflation | Coefficient  std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
inflation    |
       _cons |   3.426653   .3799913     9.02   0.000     2.681884    4.171423
-------------+----------------------------------------------------------------
ARMA         |
          ar |
         L1. |   .5647946   .0322357    17.52   0.000     .5016139    .6279753
-------------+----------------------------------------------------------------
      /sigma |   2.808235   .0600581    46.76   0.000     2.690523    2.925947
------------------------------------------------------------------------------
Note: The test of the variance against zero is one sided, and the two-sided
      confidence interval is truncated at zero.
```

Fitting ARMA models with arima III

- Estimated constant: 3.43%
- Autoregressive coefficient: 0.56
- Sigma: estimated standard deviation of the shock
- Now let's make a prediction and graph it

```
. predict pred_ar1 , dynamic(tq(2022q1))
(option xb assumed; predicted values)
. replace pred_ar1 = inflation if tin(, 2021q4)
(304 real changes made, 5 to missing)
. label variable pred_ar1 "AR(1)"

. tsline inflation pred_ar1 if tin(2021q1, ) ,     ///
>         title("Inflation and AR(1) Forecast")    /// title
>         lcolor(black stred)                      /// line colors
>         ylabel(0(2)10) yline(0, lstyle(solid))   //  y labels, lines
```

Fitting ARMA models with `arima` IV



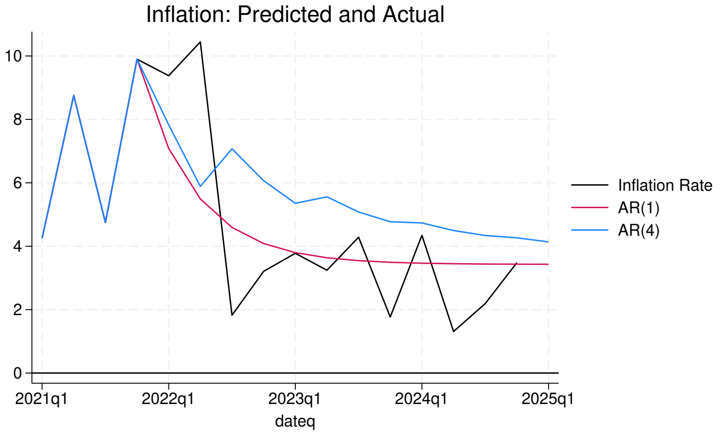
Sidebar on dynamic predictions

- Static prediction: one-step-ahead forecast using **true** past data
- Dynamic prediction: $k-$step-ahead forecast using **previous predictions** in intermediate periods
- Because dynamic predictions use past predictions as input, errors are allowed to compound
- But dynamic forecasts allow for multi-step forecasting
- Stata: static predictions are typically the default, and the `dynamic(`*time*`)` option begins dynamic forecasts at the specified time

Fitting a richer model I

```
. arima inflation if tin(, 2021m12), ar(1/4) nolog
  (output omitted)

. predict pred_ar4,  dynamic(tq(2022q1))
(option xb assumed; predicted values)

. replace pred_ar4 = inflation if tin(, 2021q4)
(304 real changes made, 5 to missing)

. label variable pred_ar4 "AR(4)"

. twoway  (tsline inflation pred*, lcolor(black stred stblue)) /// line plots
>         if tin(2021q1, ) ,                                   /// sample
>         title("Inflation: Predicted and Actual")             /// title
>         yline(0, lstyle(solid))                              /// line at 0
>         tlabel(2021q1(4)2025q1)                              /// axis labels
>         ylabel(0(2)10)
```

Fitting a richer model II



Choosing among ARMA models I

- So far I've estimated models with arbitrary lag structure
- Where do the maximum lags come from?
- This is sometimes referre to as ARMA model identification
- More complicated models capture more structure
- but are less efficient when a simpler model is available

Choosing among ARMA models II

- Sensible prior information (e.g., 1 year of lags)
- Informal assessment of autocorrelations (Box-Jenkins approach)
- Formal assessment via information criteria
- Information criteria reward model fit (measured by the log likelihood) and penalize model complexity (measured by the number of parameters)
- Common information criteria:

$$
\begin{aligned}
AIC &= -2 \ln L + 2k \\
BIC &= -2 \ln L + k \ln N \\
HQIC &= -2 \ln L + k \ln(\ln(N))
\end{aligned}
$$

- Caution: use the same sample when comparing models

Choosing among ARMA models III

- Stata: `arimasoc`
- Loops through AR and MA lags to find the preferred model
- Let's see it work

```
. arimasoc inflation
  (output omitted)
```

- The BIC selects the AR(3) model
- Estimate:

```
arima inflation if tin(, 2021q4), ar(1/3) nolog
  (output omitted)
```

- Predict:

```
. predict pred_ar3, dynamic(tq(2022q1))
(option xb assumed; predicted values)

. replace pred_ar3 = inflation if tin(, 2021q4)
(304 real changes made, 5 to missing)

. label variable pred_ar3 "AR(3)"
```

Choosing among ARMA models IV

- New: predict forecast confidence intervals

```
. predict rmse_ar3, mse dynamic(tm(2022q1))
. replace rmse_ar3 = sqrt(rmse_ar3)
(317 real changes made)
. replace rmse_ar3 = 0 if tin(, 2021q4)
(304 real changes made)
. gen lo = pred_ar3 - 2*rmse_ar3
(5 missing values generated)
. gen hi = pred_ar3 + 2*rmse_ar3
(5 missing values generated)
```

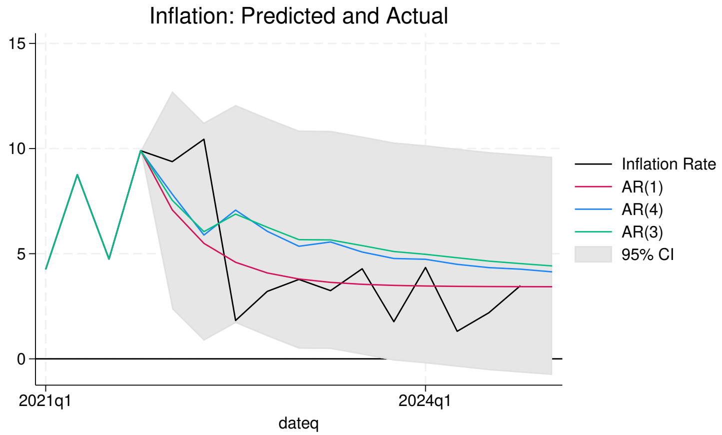
Choosing among ARMA models V

- New: graph point and interval forecasts

```
. local colors black stred stblue stgreen

. twoway  (rarea lo hi dateq, color(gs14))                /// CI plot
>         (tsline inflation pred*, lcolor(`colors´))      /// line plots
>         if tin(2021q1, )                                /// sample
>         , legend(order(2 3 4 5 1))                      /// legend order
>         legend(label(1 "95% CI"))                       /// legend label
>         title("Inflation: Predicted and Actual")        /// title
>         yline(0, lstyle(solid))                         /// line at 0
>         tlabel(2021q1(12)2025q1)                        //  axis labels
```

(type a lot, get a lot . . .)

Choosing among ARMA models VI



Sidebar on forecasting

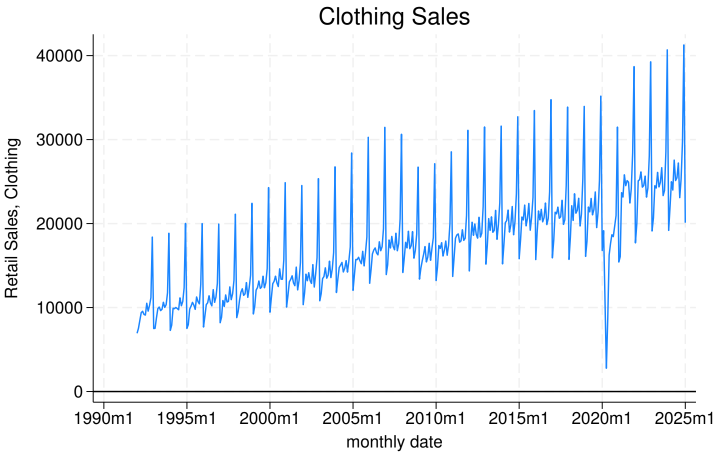
- For stationary models, the long-run (estimated) mean determines where the variable will end up
- The ARMA structure determines the path it takes to get there
- Stationarity – a constant long-run average – gives the model an anchor point
- What about confidence intervals?
  - Uncertainty due to disturbances (use `predict, rmse`)
  - Uncertainty due to estimation (need to use `forecast`)

Table of Contents

Introduction to trends, seasons, and cycles

- So far we have looked at stationary data
- Many series show trends, cycles, and seasons
  - Trend: upward or downward patterns as time progresses
  - Cycles: repetition of behavior in a regular patterns
  - Seasonality: periodic behavior at a set period (monthly, quarterly, etc)
- This section: models for all of these phenomena using `ucm`

A nonstationary series



Some background on time series components

- An unobserved-components model relies on decomposing a time series into several additive (or multiplicative) components
- Classical decomposition:

$$y_t = \mu_t + c_t + s_t + u_t$$

- where
    - μ_t is the trend
    - c_t is a stochastic cycle
    - s_t is a seasonal cycle
    - u_t is an idiosyncratic component

Models of the trend I

- Linear time trends:

$$y_t = \alpha + \beta t + e_t$$

  Interpretation: y_t fluctuates around a deterministic trend

- Random walk:

$$y_t = y_{t-1} + e_t$$

  Interpretation: the change in y_t is stationary, $\Delta y_t = e_t$

- Random walk with drift:

$$y_t = \alpha + y_{t-1} + e_t$$

  Interpretation: the change in y_t is stationary around a constant, $\Delta y_t = \alpha + e_t$

Models of the trend II

- Random walk with drift, and the drift term moves slowly

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

- Some of these models can be handled with `regress` and `arima`
  - Deterministic trend: `regress y time L(1/4).y`
  - Deterministic trend: `arima y t, ar(1/4) ma(1/2)`
  - Random walk with drift: `arima y , arima(4,1,2)`
- Example with two trends

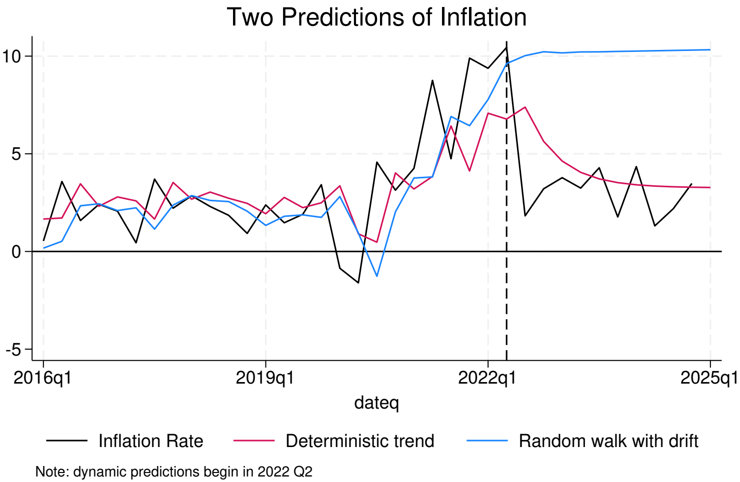
Trend specification and prediction I

- Using arima to make two forecasts:

```
. use quarterly_clean

.
. quietly arima inflation dateq if tin(,2022q2), arima(1,0,0)  // det. trend

. predict dtrend, dynamic(tq(2022q3))
(option xb assumed; predicted values)

. label variable dtrend "Deterministic trend"

.
. quietly arima inflation if tin(,2022q2), arima(1,1,0)  // r.w. with drift

. predict rwdrift, dynamic(tq(2022q3)) y
(6 missing values generated)

. label variable rwdrift "Random walk with drift"
```

- Graphing the predictions:

Trend specification and prediction II



Trend specification and prediction III

- We will not go into them today, but there are tests for a deterministic trend versus a unit root
- Dickey-Fuller tests:

$$
\begin{aligned}
y_t &= \alpha y_{t-1} + e_t \\
\implies \Delta y_t &= (\alpha - 1) y_{t-1} + e_t
\end{aligned}
$$

- $H_0 : (\alpha - 1) = 0$ (unit root)
- $H_A : (\alpha - 1) < 0$ (stationary)
- Can allow lags and time trends
- Nonstandard distribution; Stata provides correct critical values
- Stata: `dfuller`, `dfgls`, and `pperron`

The `ucm` command

- The `ucm` command allows for many different trend specifications
- Syntax: `ucm` *depvar* `[`*indepvars*`]` `[if]` `[in]` `[,` *options*`]`
- Important options:
  - `model(`*model_spec*`)`
  - `seasonal(#)`
  - `cycle()`

The `ucm` command: models for trend

- 11 built-in models for trend
- Each has a keyword
  - `dconstant` for fluctuations around a constant
  - `dtrend` for fluctuations around a time trend
  - `rwalk` for a random walk
  - `rwdrift` for a random walk with drift
  - And more

Models for Seasons

- Seasons are patterns with deterministic length (e.g. monthly)
- Models for seasons:
  - Monthly dummies (etc)
  - `ucm` with seasonality
- Let's see it work

A UCM for Clothing Sales I

- Data:

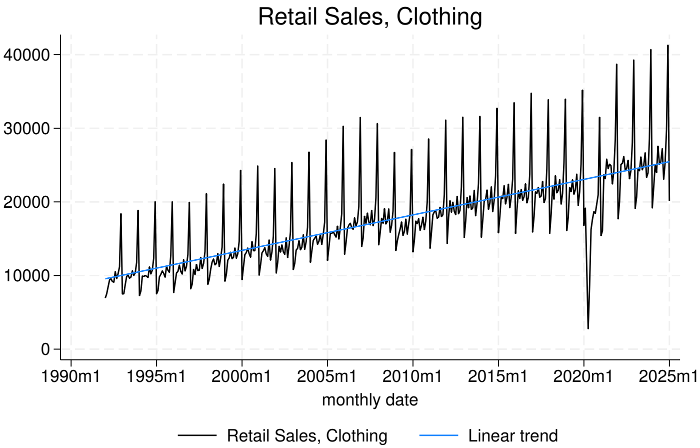
```
. use monthly_clean.dta, clear

. tsset

Time variable: datem, 1976m1 to 2025m3
        Delta: 1 month

. tsappend, add(9)
```

- Using regress to get a basic time trend:

```
. regress clothing datem

      Source |       SS           df       MS      Number of obs   =       397
-------------+----------------------------------   F(1, 395)       =    482.16
       Model |  8.4109e+09         1  8.4109e+09   Prob > F        =    0.0000
    Residual |  6.8905e+09       395  17444303.9   R-squared       =    0.5497
-------------+----------------------------------   Adj R-squared   =    0.5485
       Total |  1.5301e+10       396    38639870   Root MSE        =    4176.6

------------------------------------------------------------------------------
    clothing | Coefficient  Std. err.      t    P>|t|     [95% conf. interval]
-------------+----------------------------------------------------------------
       datem |    40.1631   1.829082    21.96   0.000     36.56714    43.75905
       _cons |   -5863.38   1084.968    -5.40   0.000    -7996.413   -3730.347
------------------------------------------------------------------------------

. predict pred_dtrend
(option xb assumed; fitted values)
```

A UCM for Clothing Sales II

- Graph:



A UCM for Clothing Sales III

- Deterministic trend model with 12-month season in ucm:

```
. ucm clothing if tin(, 2021m12), seasonal(12)  model(dtrend) nolog

Unobserved-components model
Components: deterministic trend, seasonal(12)

Sample: 1992m1 thru 2021m12                          Number of obs = 360
Log likelihood = -3138.364
```

clothing	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
var(seasonal)	9094.264	7455.884	1.22	0.111	0	23707.53
var(clothing)	3210170	250005.9	12.84	0.000	2720168	3700173

```
Note: Model is not stationary.
Note: Tests of variances against zero are one sided, and the two-sided
      confidence intervals are truncated at zero.

. predict pred_ucm, dynamic(tm(2022m1))

. lab var pred_ucm  "Prediction"
```

A UCM for Clothing Sales IV

- Predictions:

Models for Cycles

- Cycles are patterns with stochastic length (e.g. every 5-8 years)
- Models for cycles:
  - Filtering methods (`tsfilter`)
  - `ucm` with stochastic cycles

A Cyclical Model for Unemployment I

- Data:

```
. use quarterly_clean, clear
```

- Estimation:

```
. ucm unrate if tin(,2019q4), cycle(1, frequency(0.03)) nolog

Unobserved-components model
Components: random walk, order 1 cycle

Sample: 1948q1 thru 2019q4                        Number of obs =     288
                                                  Wald chi2(2)  = 2832.60
Log likelihood = -125.55825                       Prob > chi2   =  0.0000
```

unrate	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
frequency	.2715521	.0473095	5.74	0.000	.1788271	.364277
damping	.9325094	.0184627	50.51	0.000	.8963233	.9686956
var(level)	.0523546	.0191692	2.73	0.003	.0147836	.0899256
var(cycle1)	.0769678	.0213281	3.61	0.000	.0351656	.11877

```
Note: Model is not stationary.
Note: Tests of variances against zero are one sided, and the two-sided
      confidence intervals are truncated at zero.
```

A Cyclical Model for Unemployment II

- Properties of the cyclical component:

```
. estat period
```

cycle1	Coefficient	Std. err.	[95% conf.	interval]
period	23.13805	4.031086	15.23727	31.03883
frequency	.2715521	.0473095	.1788271	.364277
damping	.9325094	.0184627	.8963233	.9686956

Note: Cycle time unit is quarterly.

- The period is the average length of the cycle in units of the data (23 quarters, 5.75 years)
- Damping, between 0 and 1, indicates how tightly the cycles are clustered around the average period

Table of Contents

Multivariate intro

- So far we have modelled a time series as a function of its own lags, and perhaps in terms of trends or seasons
- This section introduces tools for modeling time series jointly
  - Autoregressive Distributed Lag (ARDL)
  - Vector Autoregression (VAR)

Table of Contents

Autoregressive Distributed Lag models

- Extends AR model to include exogenous variables and their lags
- Autoregressive model:

$$y_t = \alpha_1 y_{t-1} + \cdots + \alpha_p y_{t-p} + e_t$$

  (you've seen this before)
- Autoregressvie distributed lag model:

$$y_t = \alpha_1 y_{t-1} + \cdots + \alpha_p y_{t-p} + \beta_0 x_t + \beta_1 x_{t-1} + \cdots + \beta_r x_{t-r} + e_t$$

- Notice the current value x_t in the y_t equation
- Can be estimated with `regress`
- Let's see it work

Estimating an ARDL model I

- Data:

```
. use quarterly_clean

. tsappend, add(3)

. describe

Contains data from quarterly_clean.dta
 Observations:           320
    Variables:            10                  8 Apr 2025 11:03
---------------------------------------------------------------------------------
Variable      Storage   Display    Value
    name         type    format    label      Variable label
---------------------------------------------------------------------------------
datestr         str6    %9s
dateq           float   %tq
oilprice        float   %9.0g                 Oil Price
cpi             float   %9.0g                 Consumer Price Index
unrate          float   %9.0g                 Unemployment Rate
fedfunds        float   %9.0g                 Interest Rate
ln_cpi          float   %9.0g                 Log of price level
inflation       float   %9.0g                 Inflation Rate
ln_oil          float   %9.0g                 Log of oil price
oil_infl        float   %9.0g                 Oil price inflation
---------------------------------------------------------------------------------
Sorted by: dateq
     Note: Dataset has changed since last saved.
```

Estimating an ARDL model II

- Estimation:

```
. regress inflation l(1/4).inflation L(0/4).oil_infl

      Source |       SS           df       MS      Number of obs   =       261
-------------+----------------------------------   F(9, 251)       =     42.06
       Model |  1835.76469         9  203.973855   Prob > F        =    0.0000
    Residual |  1217.15722       251  4.84923194   R-squared       =    0.6013
-------------+----------------------------------   Adj R-squared   =    0.5870
       Total |  3052.92191       260  11.7420074   Root MSE        =    2.2021

------------------------------------------------------------------------------
   inflation | Coefficient  Std. err.      t    P>|t|     [95% conf. interval]
-------------+----------------------------------------------------------------
   inflation |
         L1. |   .3431796   .0620517     5.53   0.000     .2209712     .465388
         L2. |   .1600333   .0599326     2.67   0.008     .0419983    .2780682
         L3. |   .3598485    .058109     6.19   0.000     .2454051    .4742919
         L4. |  -.0999274   .0578606    -1.73   0.085    -.2138815    .0140267
             |
    oil_infl |
         --. |   .0238986   .0024772     9.65   0.000     .0190198    .0287774
         L1. |   .0019419   .0028655     0.68   0.499    -.0037017    .0075855
         L2. |   .0021146   .0028462     0.74   0.458     -.003491    .0077202
         L3. |  -.0083601   .0027625    -3.03   0.003    -.0138008   -.0029194
         L4. |   .0009038   .0027274     0.33   0.741    -.0044676    .0062753
             |
       _cons |   .6895869   .2284134     3.02   0.003     .2397358    1.139438
------------------------------------------------------------------------------
```

Table of Contents

Vector Autoregression

- Extends AR model to multiple variables
- AR(1) model:

$$y_t = \alpha_1 y_{t-1} + e_t$$

  (you've seen this before)
- VAR(1) model:

$$\begin{aligned} y_t &= \alpha_{11} y_{t-1} + \alpha_{12} x_{t-1} + e_{y,t} \\ x_t &= \alpha_{21} y_{t-1} + \alpha_{22} x_{t-1} + e_{x,t} \end{aligned}$$

- Simultaneous modeling
- No contemporaneous terms (but see `svar`)
- Cross-lags capture dependence across variables

The `var` command and its suite

- Syntax: `var` *varlist* `[if] [in] [, options]`
- Important options:
  - `lags()` – a numlist of lags (not just the max lag)
  - `exog()` – exogenous variables
  - `dfk`, `small`, `vce(robust)` – to adjust standard errors
- Additional commands
  - `varsoc` – lag order selection
  - `varstable` – check stationarity condition
  - `vargranger` – Granger causality (cross-lag significance) tests
- After estimation
  - `predict` for onestep predictions
  - `fcast compute` for dynamic forecasts

Estimating a VAR model I

- Data:

```
. use quarterly_clean

. tsappend, add(3)

. describe

Contains data from quarterly_clean.dta
 Observations:           320
    Variables:            10                  8 Apr 2025 11:03
------------------------------------------------------------------------------------
Variable      Storage   Display    Value
    name         type    format    label      Variable label
------------------------------------------------------------------------------------
datestr         str6    %9s
dateq           float   %tq
oilprice        float   %9.0g                 Oil Price
cpi             float   %9.0g                 Consumer Price Index
unrate          float   %9.0g                 Unemployment Rate
fedfunds        float   %9.0g                 Interest Rate
ln_cpi          float   %9.0g                 Log of price level
inflation       float   %9.0g                 Inflation Rate
ln_oil          float   %9.0g                 Log of oil price
oil_infl        float   %9.0g                 Oil price inflation
------------------------------------------------------------------------------------
Sorted by: dateq
     Note: Dataset has changed since last saved.
```

Estimating a VAR model II

- Lag order selection:

```
. varsoc inflation unrate fedfunds if tin(, 2022q2)
Lag-order selection criteria
   Sample: 1955q3 thru 2022q2                          Number of obs = 268
```

Lag	LL	LR	df	p	FPE	AIC	HQIC	SBIC
0	-1872.36				240.227	13.9952	14.0114	14.0354
1	-1239.33	1266.1	9	0.000	2.28103	9.33825	9.40283	9.49904
2	-1207.26	64.124	9	0.000	1.92043	9.16615	9.27916	9.44753*
3	-1187.62	39.283*	9	0.000	1.77394*	9.08673*	9.24818*	9.48871
4	-1181.27	12.712	9	0.176	1.80952	9.10646	9.31635	9.62903

```
   * optimal lag
   Endogenous: inflation unrate fedfunds
    Exogenous: _cons
```

- Estimation:

```
. var inflation unrate fedfunds, lag(1/2)
```
(*output omitted*)

Estimating a VAR model III

- Estimation results in a `table`:

```
. table () (coleq result) if tin(, 2022q2), nformat(%7.2f)   ///
>         command(coef=_r_b SE=_r_se:                        ///
>                 var inflation unrate fedfunds, lags(1/2))
```

	Inflation Rate		Unemployment Rate		Interest Rate	
	coef	SE	coef	SE	coef	SE
L.inflation	0.32	0.06	-0.01	0.01	-0.02	0.03
L2.inflation	0.31	0.06	0.02	0.01	0.08	0.03
L.unrate	0.15	0.25	0.98	0.06	-0.30	0.12
L2.unrate	-0.14	0.25	-0.06	0.06	0.27	0.12
L.fedfunds	0.69	0.13	-0.04	0.03	0.85	0.06
L2.fedfunds	-0.57	0.14	0.07	0.03	0.08	0.06
Intercept	0.69	0.56	0.33	0.14	0.29	0.27

Estimating a VAR model IV

- Granger causality:

```
. vargranger
```

Granger causality Wald tests

Equation	Excluded	chi2	df	Prob > chi2
inflation	unrate	.36831	2	0.832
inflation	fedfunds	28.456	2	0.000
inflation	ALL	28.675	4	0.000
unrate	inflation	2.7058	2	0.258
unrate	fedfunds	6.2375	2	0.044
unrate	ALL	15.977	4	0.003
fedfunds	inflation	7.9656	2	0.019
fedfunds	unrate	6.4066	2	0.041
fedfunds	ALL	14.28	4	0.006

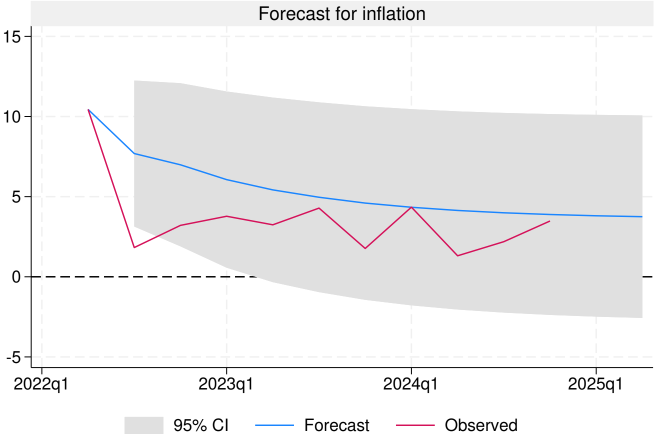
- Notes: H_0 is that all coefficients on the excluded variable are 0 Rejection implies that the excluded variable Granger-causes the dependent variable

Estimating a VAR model V

- Make a forecast

```
. fcast compute f_, step(12) dynamic(tq(2022q3))
. fcast graph f_inflation , yline(0) observed legend(rows(1))
```

Estimating a VAR model VI



Other tasks with VARs

- Impulse response analysis
- What is the effect of an increase in one variable in the VAR on the other variables over time?
- Stata: `irf` suite
- Structural VARs: adding contemporaneous relationships based on theory
- Structural VARs with instrumental variables
- Panel VARs

A Summing Up

- Stata's time-series environment
- Models for stationary data
- Models for trends, seasons, and cycles
- Models for multivariate time series