MONASH University

MONASH BUSINESS SCHOOL

ETW3420: Principles of Forecasting and Applications

Topic 8: Advanced Forecasting Methods

Dr. Jason Ng

Outline

Regression with ARIMA errors

Regression models

$$y_t = \beta_0 + \beta_1 x_{1,t} + \cdots + \beta_k x_{k,t} + \varepsilon_t,$$

- y_t modeled as function of k explanatory variables $x_{1,t}, \ldots, x_{k,t}$.
- In regression, we assume that ε_t was WN.
- Now we want to allow ε_t to be autocorrelated.

Regression models

$$y_t = \beta_0 + \beta_1 x_{1,t} + \cdots + \beta_k x_{k,t} + \varepsilon_t,$$

- y_t modeled as function of k explanatory variables $x_{1,t}, \ldots, x_{k,t}$.
- In regression, we assume that ε_t was WN.
- Now we want to allow ε_t to be autocorrelated.

Example: ARIMA(1,1,1) errors

$$y_t = \beta_0 + \beta_1 x_{1,t} + \cdots + \beta_k x_{k,t} + \eta_t,$$

$$(1 - \phi_1 B)(1 - B)\eta_t = (1 + \theta_1 B)\varepsilon_t,$$

where ε_t is white noise.

Residuals and errors

Example: η_t = ARIMA(1,1,1)

$$y_t = \beta_0 + \beta_1 x_{1,t} + \cdots + \beta_k x_{k,t} + \eta_t,$$

$$(1 - \phi_1 B)(1 - B)\eta_t = (1 + \theta_1 B)\varepsilon_t,$$

Residuals and errors

Example: η_t = ARIMA(1,1,1)

$$y_t = \beta_0 + \beta_1 x_{1,t} + \cdots + \beta_k x_{k,t} + \eta_t,$$

$$(1 - \phi_1 B)(1 - B)\eta_t = (1 + \theta_1 B)\varepsilon_t,$$

- Be careful in distinguishing η_t from ε_t.
- Only the errors ε_t are assumed to be white noise.
- In ordinary regression, η_t is assumed to be white noise and so $\eta_t = \varepsilon_t$.

Estimation

If we minimize $\sum \eta_t^2$ (by using ordinary regression):

1. Estimated coefficients $\hat{\beta}_0, \dots, \hat{\beta}_k$ are biased as some information ignored;
2. Statistical tests associated with the model (e.g., t-tests on the coefficients) are incorrect.
3. *p*-values for coefficients usually too small ("spurious regression").
4. AIC of fitted models misleading.

Estimation

If we minimize $\sum \eta_t^2$ (by using ordinary regression):

1. Estimated coefficients $\hat{\beta}_0, \dots, \hat{\beta}_k$ are biased as some information ignored;
2. Statistical tests associated with the model (e.g., t-tests on the coefficients) are incorrect.

3. *p*-values for coefficients usually too small (“spurious regression”).
4. AIC of fitted models misleading.

- Minimizing $\sum \varepsilon_t^2$ avoids these problems.
- Maximizing likelihood is similar to minimizing $\sum \varepsilon_t^2$.

Stationarity

Regression with ARMA errors

$$y_t = \beta_0 + \beta_1 x_{1,t} + \cdots + \beta_k x_{k,t} + \eta_t,$$

where η_t is an ARMA process.

- All variables in the model must be stationary.
- If we estimate the model while any of these are non-stationary, the estimated coefficients can be incorrect.
- Difference variables until all stationary.
- If necessary, apply same differencing to all variables.

Stationarity

Model with ARIMA(1,1,1) errors

$$y_t = \beta_0 + \beta_1 x_{1,t} + \cdots + \beta_k x_{k,t} + \eta_t,$$

$$(1 - \phi_1 B)(1 - B)\eta_t = (1 + \theta_1 B)\varepsilon_t,$$

Stationarity

Model with ARIMA(1,1,1) errors

$$y_t = \beta_0 + \beta_1 x_{1,t} + \cdots + \beta_k x_{k,t} + \eta_t,$$

$$(1 - \phi_1 B)(1 - B)\eta_t = (1 + \theta_1 B)\varepsilon_t,$$

Equivalent to model with ARIMA(1,0,1) errors

$$y'_t = \beta_1 x'_{1,t} + \cdots + \beta_k x'_{k,t} + \eta'_t,$$

$$(1 - \phi_1 B)\eta'_t = (1 + \theta_1 B)\varepsilon_t,$$

where $y'_t = y_t - y_{t-1}$, $x'_{t,i} = x_{t,i} - x_{t-1,i}$ and $\eta'_t = \eta_t - \eta_{t-1}$.

Regression with ARIMA errors

Any regression with an ARIMA error can be rewritten as a regression with an ARMA error by differencing all variables with the same differencing operator as in the ARIMA model.

Any regression with an ARIMA error can be rewritten as a regression with an ARMA error by differencing all variables with the same differencing operator as in the ARIMA model.

Original data

$$y_t = \beta_0 + \beta_1 x_{1,t} + \cdots + \beta_k x_{k,t} + \eta_t$$

$$\text{where} \quad \phi(B)(1-B)^d \eta_t = \theta(B)\varepsilon_t$$

Regression with ARIMA errors

Any regression with an ARIMA error can be rewritten as a regression with an ARMA error by differencing all variables with the same differencing operator as in the ARIMA model.

Original data

$$y_t = \beta_0 + \beta_1 x_{1,t} + \cdots + \beta_k x_{k,t} + \eta_t$$

$$\text{where} \quad \phi(B)(1 - B)^d \eta_t = \theta(B)\varepsilon_t$$

After differencing all variables

$$y'_t = \beta_1 x'_{1,t} + \cdots + \beta_k x'_{k,t} + \eta'_t.$$

$$\text{where} \quad \phi(B)\eta_t = \theta(B)\varepsilon_t$$

$$\text{and} \quad y'_t = (1 - B)^d y_t$$

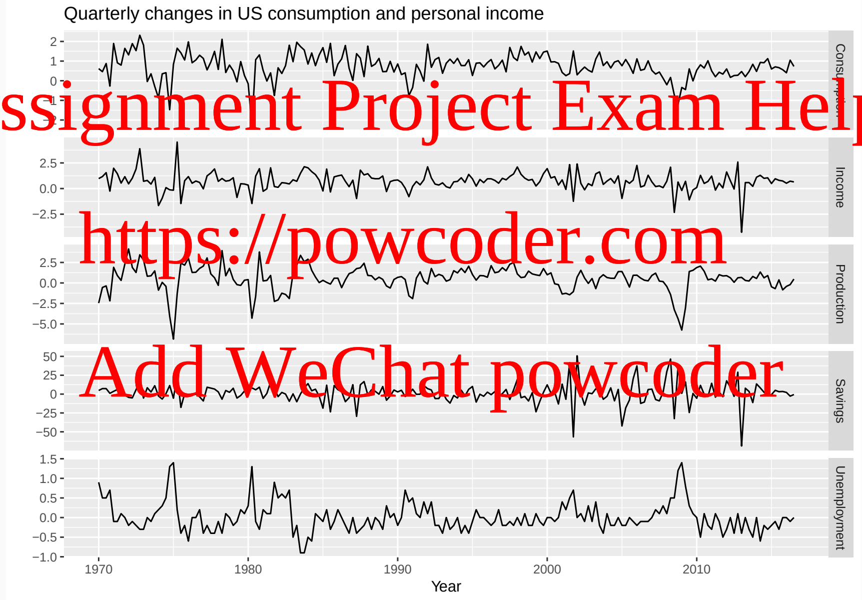
Model selection

- Check that all variables are stationary. If not, apply differencing. Where appropriate, use the same differencing for all variables to preserve interpretability.
- Fit regression model with automatically selected ARIMA errors.
- Check that ε_t series looks like white noise.

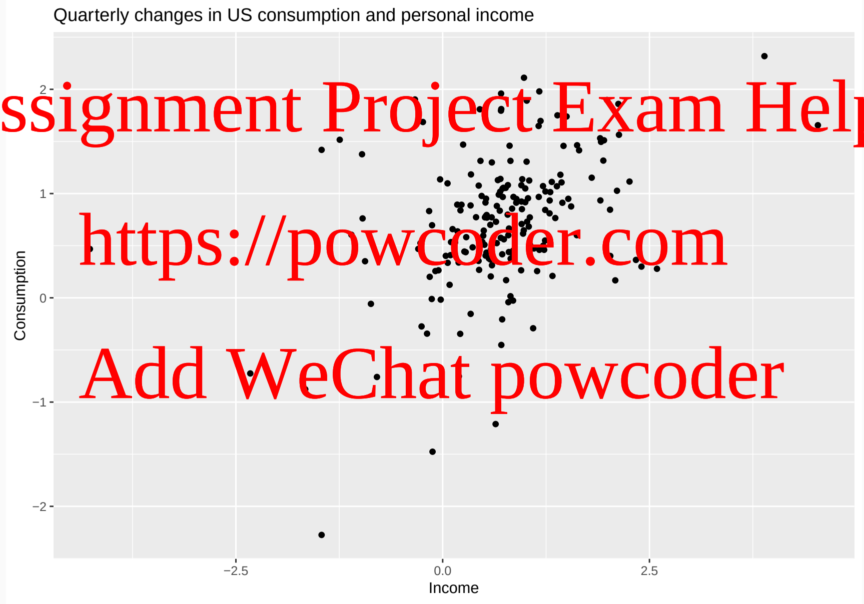
Selecting predictors

- AICc can be calculated for final model.
- Repeat procedure for all subsets of predictors to be considered, and select model with lowest AIC value.

US personal consumption and income



US personal consumption and income



US personal consumption and income

- No need for transformations or further differencing.
- Increase in income does not necessarily translate into instant increase in consumption (e.g., after the loss of a job, it may take a few months for expenses to be reduced to allow for the new circumstances). We will ignore this for now.

US personal consumption and income

```
(fit <- auto.arima(uschange[,1], xreg=uschange[,2]))
```

```
## Series: uschange[, 1]
## Regression with ARIMA(1,0,2) errors
##
## Coefficients:
##          ar1      ma1     ma2  intercept    xreg
##       0.6922  -0.5758  0.1984     0.5990  0.2028
## s.e.  0.1159   0.1301  0.0756     0.0884  0.0461
##
## sigma^2 estimated as 0.3219:  log likelihood=-156.95
## AIC=325.91   AICc=326.37   BIC=345.29
```

US personal consumption and income

```r
(fit <- auto.arima(uschange[,1], xreg=uschange[,2]))
```

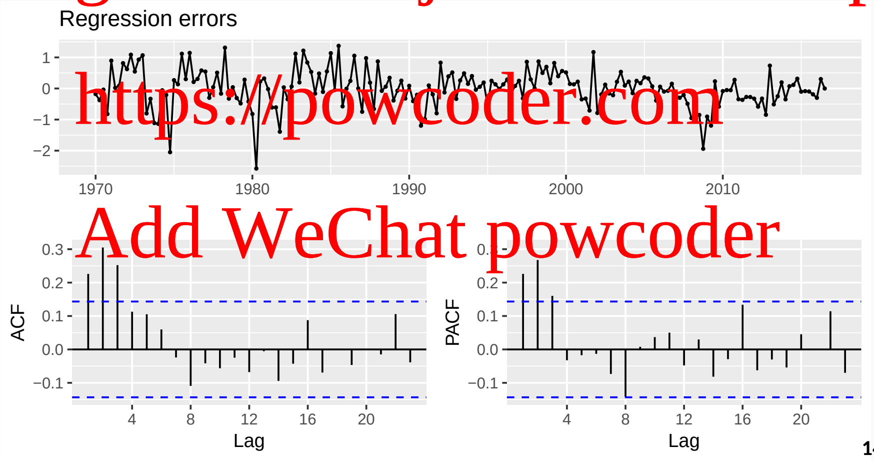


```
## Series: uschange[, 1]
## Regression with ARIMA(1,0,2) errors
##
## Coefficients:
##          ar1      ma1     ma2  intercept    xreg
##       0.6922  -0.5758  0.1984     0.5990  0.2028
## s.e.  0.1159   0.1301  0.0756     0.0884  0.0461
##
## sigma^2 estimated as 0.3219:  log likelihood=-156.95
## AIC=325.91   AICc=326.37   BIC=345.29
```


Write down the equations for the fitted model.

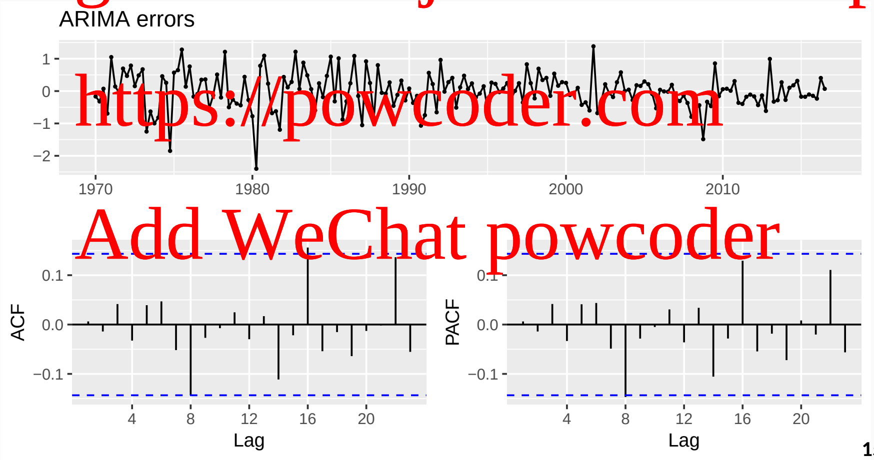
US personal consumption and income

```
ggtsdisplay(residuals(fit, type='regression'),
  main="Regression errors")
```

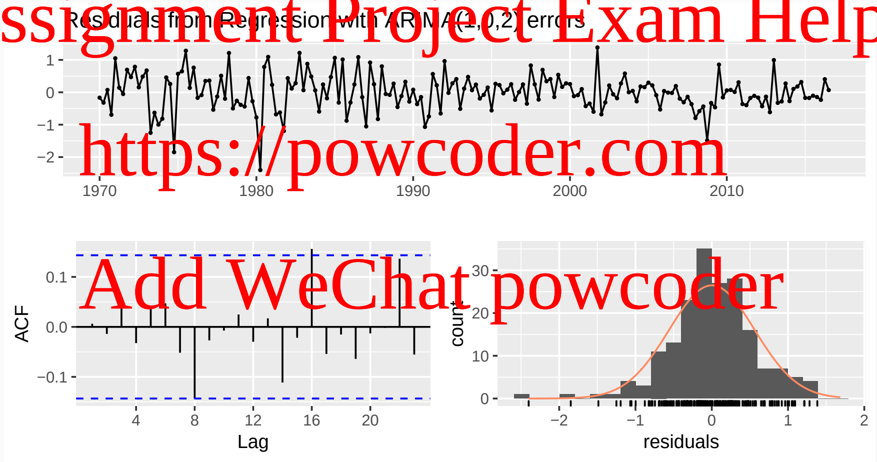
US personal consumption and income

```
ggtsdisplay(residuals(fit, type='response'),
  main="ARIMA errors")
```

US personal consumption and income

```r
checkresiduals(fit, test=FALSE)
```



US personal consumption and income

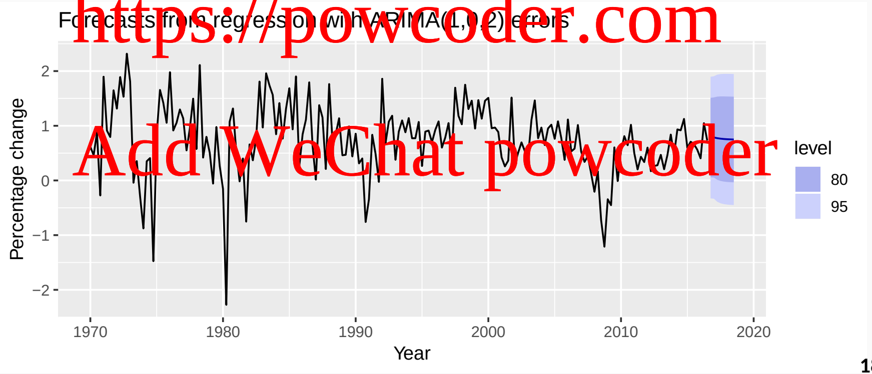

```
checkresiduals(fit, plot=FALSE)
```



```
##
##  Ljung-Box test
##
## data:  Residuals from Regression with ARIMA(1,0,2) errors
## Q* = 5.8916, df = 3, p-value = 0.117
##
## Model df: 5.   Total lags used: 8
```

US personal consumption and income

```r
fcast <- forecast(fit,
  xreg=rep(mean(uschange[,2]),8), h=8)
autoplot(fcast) + xlab("Year") +
  ylab("Percentage change") +
  ggtitle("Forecasts from regression with ARIMA(1,0,2) errors")
```



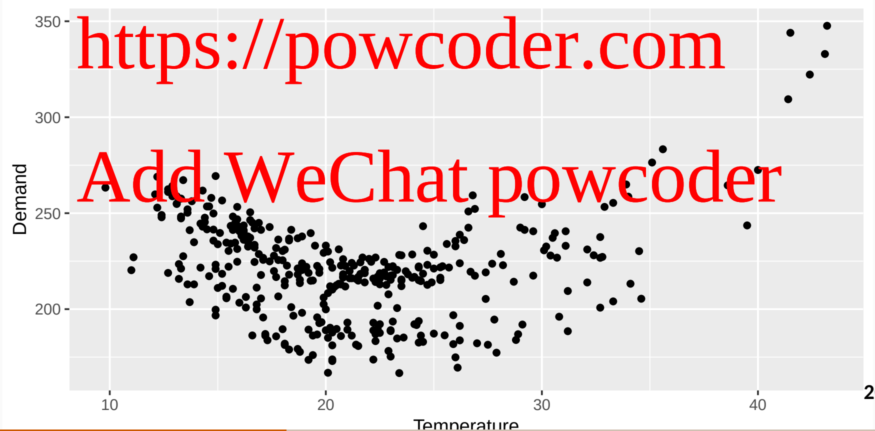
Forecasting

- To forecast a regression model with ARIMA errors, we need to forecast the regression part of the model and the ARIMA part of the model and combine the results.
- Some predictors are known into the future (e.g., time, dummies).
- Separate forecasting models may be needed for other predictors.
- Forecast intervals ignore the uncertainty in forecasting the predictors.

Daily electricity demand

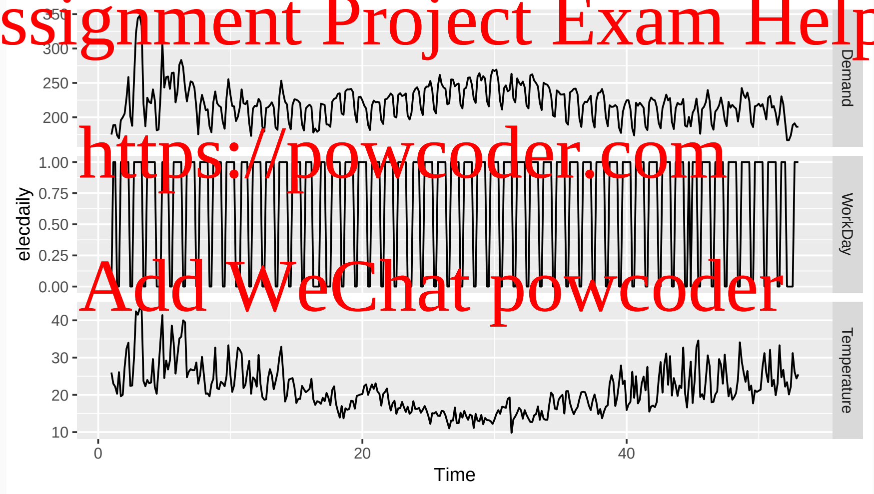
Model daily electricity demand as a function of temperature using quadratic regression with ARMA errors.


```
qplot(elecdaily[,"Temperature"], elecdaily[,"Demand"]) +
  xlab("Temperature") + ylab("Demand")
```



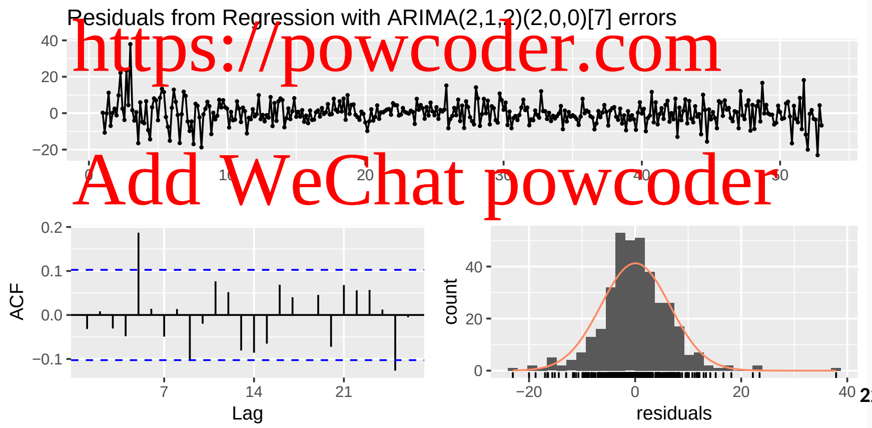
Daily electricity demand

```r
autoplot(elecdaily, facets = TRUE)
```



Daily electricity demand

```
xreg <- cbind(MaxTemp = elecdaily[, "Temperature"],
              MaxTempSq = elecdaily[, "Temperature"]^2,
              Workday = elecdaily[, "WorkDay"])
fit <- auto.arima(elecdaily[, "Demand"], xreg = xreg)
checkresiduals(fit)
```

Daily electricity demand


```
##
##  Ljung-Box test
##
## data:  Residuals from Regression with ARIMA(2,1,2)(2,0,0)[7] errors
## Q* = 28.229, df = 4, p-value = 1.121e-05
##
## Model df: 10.   Total lags used: 14
```

Daily electricity demand

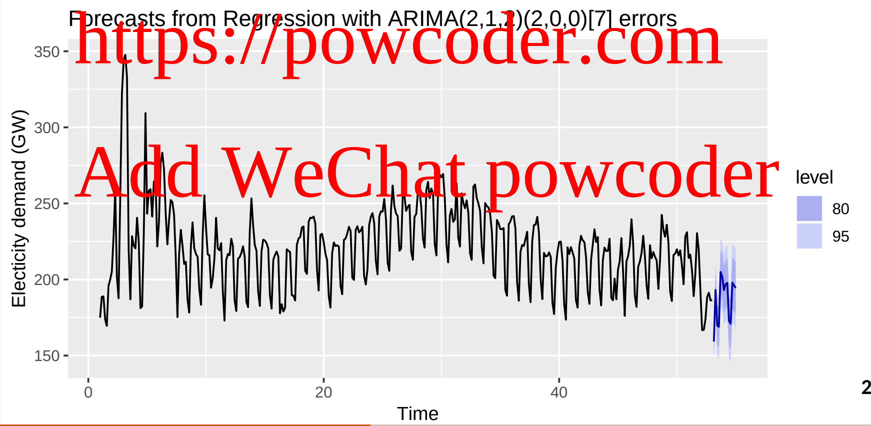

```r
# Forecast one day ahead
forecast(fit, xreg = cbind(26, 26^2, 1))
```



```
##          Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## 53.14286        189.769 181.2954 198.2427 176.8096 202.7284
```

Daily electricity demand

```
fcast <- forecast(fit,
  xreg = cbind(rep(26,14), rep(26^2,14),
    c(0,1,0,0,1,1,1,1,1,0,0,1,1,1)))
autoplot(fcast) + ylab("Electicity demand (GW)")
```

Outline

Forecast Combinations

- It turns out that combining multiple forecasting methods can lead to improved predictive performance.

- This fact has been well known since at least 1969, when Bates and Granger wrote their famous paper on "The Combination of Forecasts", that combining forecasts often lead to better forecast accuracy.

- 20 years later, Clemen (1989) wrote

> The results have been virtually unanimous: combining multiple forecasts leads to increased forecast accuracy. In many cases one can make dramatic performance improvements by simply averaging the forecasts.

Forecast Combinations

- In the million-dollar Netflix Prize contest in 2009, the winning team, Bellkor's Pragmatic Chaos, combined results from several teams to produce their winning forecasts.
- In a 2010 article the Netflix Prize winners described the power of their ensemble approach:

> "An early lesson of the competition was the value of combining sets of predictions from multiple models or algorithms. If 2 prediction sets achieved similar RMSEs, it was quicker and more effective to simply average the 2 sets than to try to develop a new model that incorporated the best of each method. Even if the RMSE for one set was much worse than the other, there was almost certainly a linear combination that improved on the better set."

Forecast Combinations

- There has been considerable research on using weighted averages, with packages developed for estimating these weights (e.g. `opera`: "Online Prediction by ExpeRt Aggegation").
- However, using a simple average has proven hard to beat.

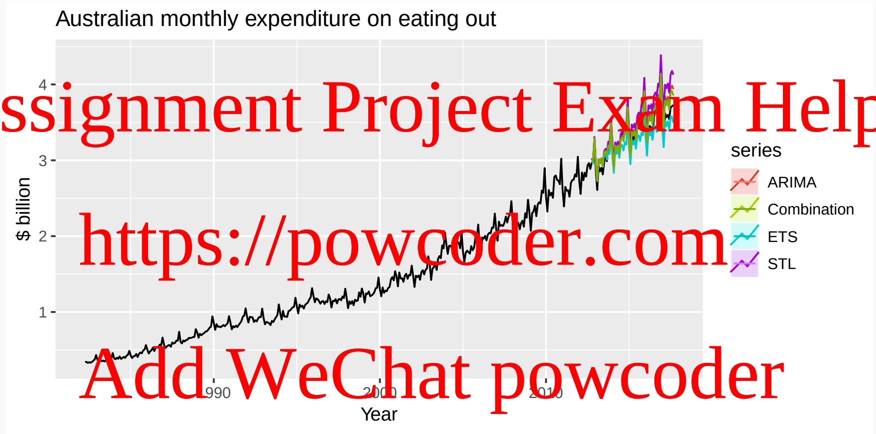
Monthly Expenditure on eating out in Australia

```
train <- window(auscafe, end=c(2012,9))
h <- length(auscafe) - length(train)

ETS <- forecast(ets(train), h=h)
ARIMA <- forecast(auto.arima(train, lambda=0, biasadj=TR
  h=h)
STL <- stlf(train, lambda=0, h=h, biasadj=TRUE)

Combination <- (ETS[["mean"]] + ARIMA[["mean"]] +
  STL[["mean"]])/3
```

Monthly Expenditure on eating out in Australia


```
autoplot(auscafe) +
  autolayer(ETS, series="ETS", PI=FALSE) +
  autolayer(ARIMA, series="ARIMA", PI=FALSE) +
  autolayer(STL, series="STL", PI=FALSE) +
  autolayer(Combination, series="Combination") +
  xlab("Year") + ylab("$ billion") +
  ggtitle("Australian monthly expenditure on eating out"
```

Monthly Expenditure on eating out in Australia



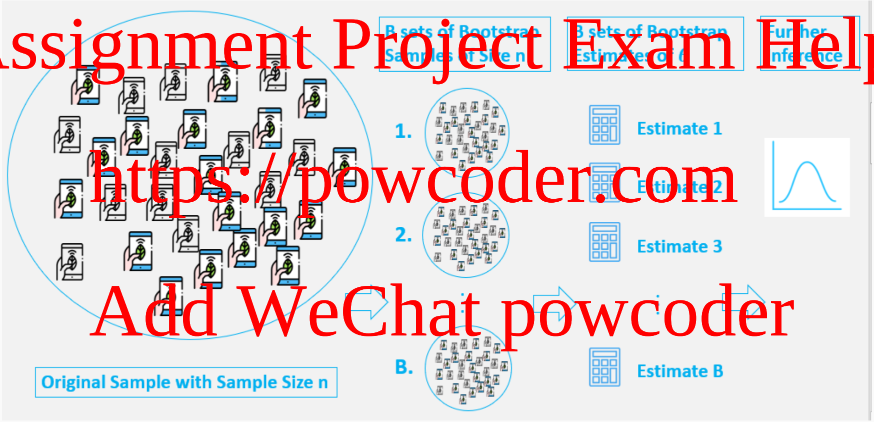
Out-of-sample MAPE:

ETS	ARIMA	STL	COMBINATION
3.682	3.036	5.201	2.388

Outline

- The basic idea of bootstrap is to make **inference** about a estimate (such as sample mean) for a population parameter θ (such as population mean) on sample data.
- It is a resampling method by independently **sampling with replacement** from an existing sample data with same sample size n, and performing inference among these resampled data.

Bootstrap


B sets of Bootstrap Samples of Size n
B sets of Bootstrap Estimates of
Further Inference
1.
2.
B.
Estimate 1
Estimate 2
Estimate 3
Estimate B
Original Sample with Sample Size n

Bootstrap

- In a time series context, bootstrapping can be performed to **generate new time series** that are similar to our observed series.

- However, bootstrapping is a little tricky to implement with time series data due to its ordering and autocorrelation.

- Solution: Block bootstrap.

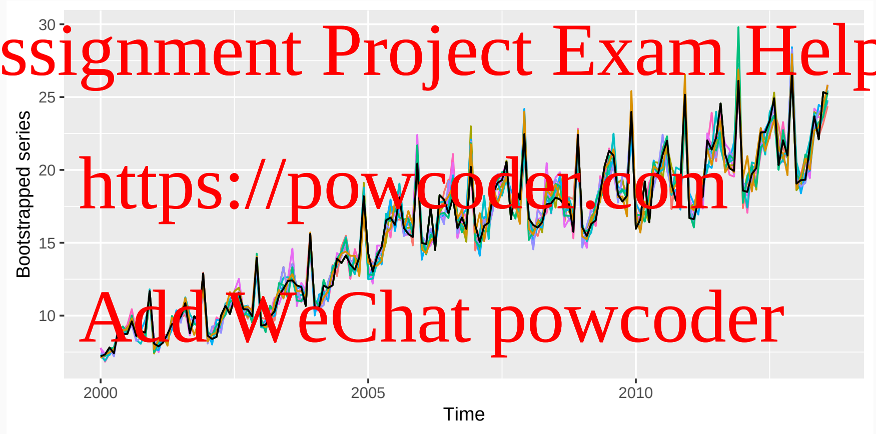
Block Bootstrap

1. The time series is Box-Cox transformed, and decomposed into trend, seasonal and remainder components using STL.
2. Obtain shuffled versions of the remainder component to get bootstrapped remainder series.
   (a) Cannot simply use the re-draw procedure due to the presence of autocorrelation present in an STL remainder series.
   (b) Instead, contiguous sections of the remainder time series are selected at random and joined together.
3. The boostrapped remainder series are added to the trend and seasonal components, and the Box-Cox transformation is reversed to give variations on the original time series.

Monthly expenditure on retail debit cards in Iceland


```
bootseries <- bld.mbb.bootstrap(debitcards, 10) %>%
         as.data.frame() %>%
         ts(start=2000, frequency=12)

autoplot(debitcards) +
  autolayer(bootseries, colour=TRUE) +
  autolayer(debitcards, colour=FALSE) +
  ylab("Bootstrapped series") + guides(colour="none")
```

Monthly expenditure on retail debit cards in Iceland



Uses of bootstrapping

1. Get a better measure of forecast uncertainty
2. Improve point forecasts using "bagging"

Bagging

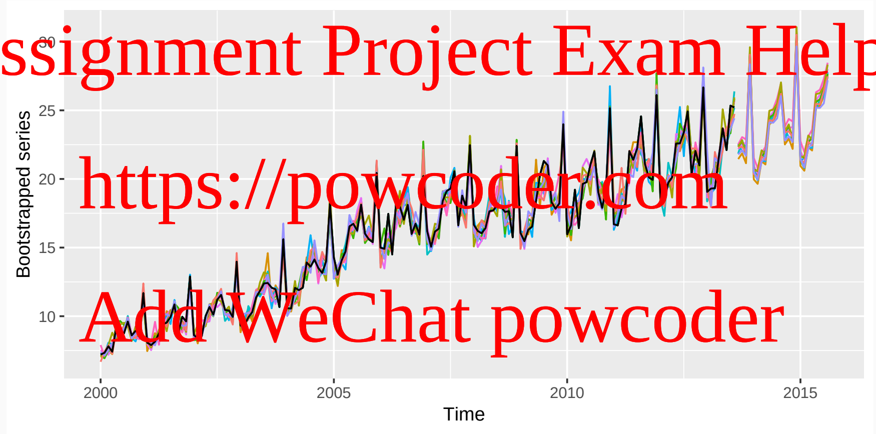
- Bootstrapped time series can be used to improve forecast accuracy.

- Produce forecasts from each of the additional time series $\rightarrow$ average the forecasts $\rightarrow$ better forecasts.
- This procedure is called “bagging”, acronym for “**b**ootstrap **agg**regat**ing**”.

Bagging with ETS

```r
sim <- bld.mbb.bootstrap(debitcards, 10) %>%
        as.data.frame() %>%
        ts(start = 2000, freq = 12)
fc <- purrr::map(as.list(sim),
           function(x){forecast(ets(x))[["mean"]]
        as.data.frame() %>%
        ts(freq = 12, start = c(2013,9))
autoplot(debitcards) +
        autolayer(sim, colour = TRUE) +
        autolayer(fc, colour = TRUE) +
        autolayer(debitcards, colour = FALSE) +
        ylab("Bootstrapped series") +
        guides(colour = "none")
```

Bagging with ETS



`baggedETS()`

- The above procedure can be handled with the `baggedETS()` function, whereby the average of the ETS forecasts gives the bagged forecasts of the original data.

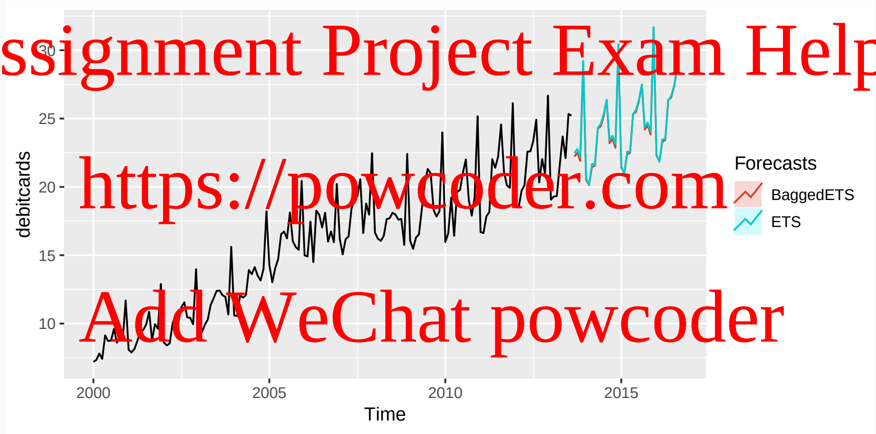
- By default, 100 bootstrapped series are used; length of the blocks used for obtaining bootstrapped residuals is set to 24 for monthly data.

- The more general function is `baggedModel()` whereby you can specify the forecasting function (e.g. “ets”, “auto.arima”,…).

baggedETS()

```
etsfc <- debitcards %>% ets() %>% forecast(h=36)

baggedfc <- debitcards %>% baggedETS() %>%
  forecast(h=36)

autoplot(debitcards) +
  autolayer(baggedfc, series="BaggedETS", PI=FALSE) +
  autolayer(etsfc, series="ETS", PI=FALSE) +
  guides(colour=guide_legend(title="Forecasts"))
```

baggedETS()



Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder
debitcards
30
25
20
15
10
2000
2005
2010
2015
Time
Forecasts
BaggedETS
ETS

Bagging

- In this case, it makes little difference.
- Bergmeir, Hyndman, & Benítez (2016) show that, on average, bagging gives better forecasts than just applying `ets()` directly.

Outline

Missing Values

Functions which can handle missing values

- `auto.arima()`, `Arima()`
- `tslm()`
- `nnetar()`

Models which cannot handle missing values

- `ets()`
- `stl()`
- `stlf()`
- `tbats()`

Missing Values

Functions which can handle missing values

- `auto.arima()`, `Arima()`
- `tslm()`
- `nnetar()`

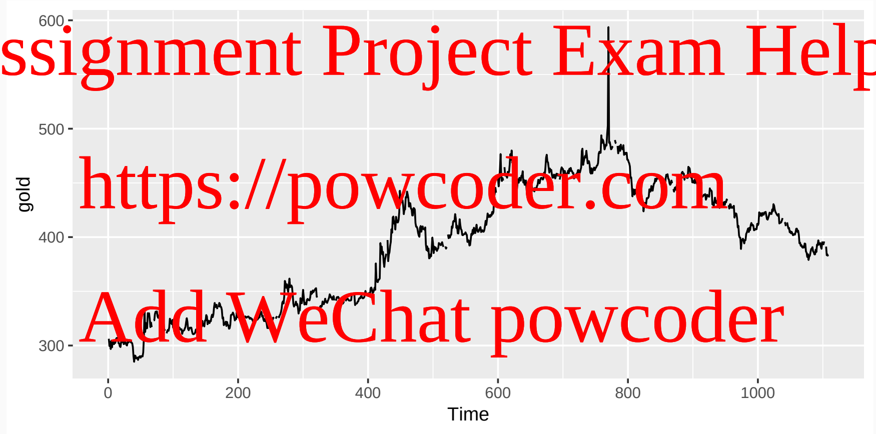
Models which cannot handle missing values

- `ets()`
- `stl()`
- `stlf()`
- `tbats()`

What to do?

1. Model section of data after last missing value.
2. Estimate missing values with `na.interp()`.

Missing values

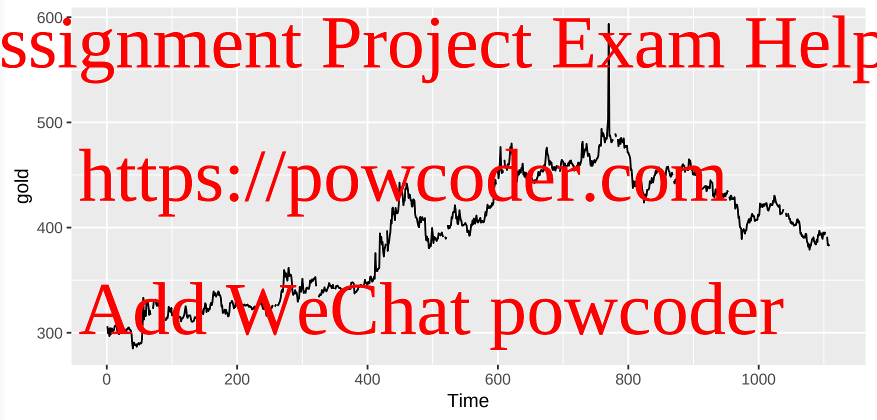


Missing values

```
itp <- gold %>% na.interp()
autoplot(itp, series="Interpolated") +
        autolayer(gold, series="Original") +
        scale_color_manual(
        values=c(Interpolated="red",Original="gray"))
```


Outline

Outliers



Outliers

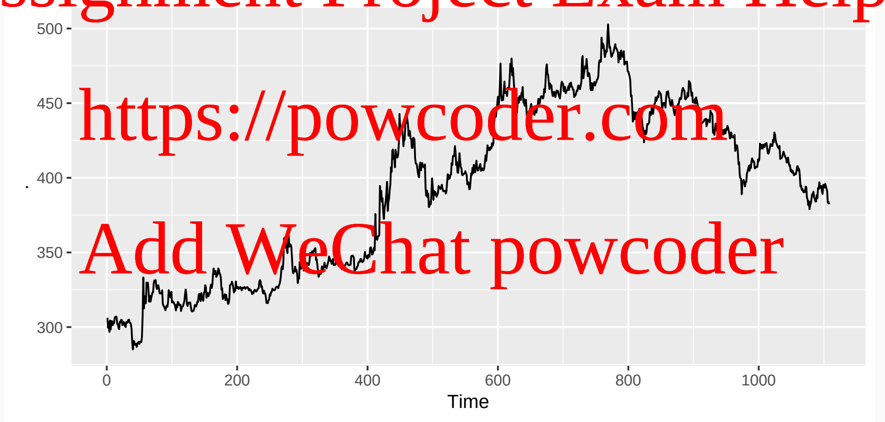

```
tsoutliers(gold)
```



```
## $index
## [1] 770
##
## $replacements
## [1] 494.9
```

Outliers

```
gold %>% tsclean() %>% autoplot()
```

Forecasting with Outliers

```
gold %>% tsclean() %>%
    ets() %>%
    forecast(h = 24) %>%
    autoplot()
```