Chapter 4- Time series features

🔗 Forecasting Book Club

Dr. Bahman Rostami-Tabar

Outline

- Some simple statistics
- ACF features
- STL Features
- Other features
- Exploring Australian tourism data

Time series feature

- are numerical summaries computed from the series, e.g. autocorrelation, trend.
- The `feasts` package in R includes functions for computing FEatures And Statistics from Time Series (hence the name).

When time series feature are useful?

- If you work with many time series
- If you don't know anything about features/characteristics of your time series (one or many)
- We can compute many different features on many different time series, and use them to explore the properties of the series.

Some simple statistics

Extract features using `feast` package

```
your_data %>%
features(your_measurement,
function)
```

```
tourism %>% features(Trips,
mean)
```

mean

```
tourism %>% features(Trips, mean)
```

```
## # A tibble: 304 x 4
##    Region         State               Purpose    ...4
##    <chr>          <chr>               <chr>     <dbl>
##  1 Adelaide       South Australia     Business 156.
##  2 Adelaide       South Australia     Holiday  157.
##  3 Adelaide       South Australia     Other     56.6
##  4 Adelaide       South Australia     Visiting 205.
##  5 Adelaide Hills South Australia     Business   2.66
##  6 Adelaide Hills South Australia     Holiday   10.5
##  7 Adelaide Hills South Australia     Other      1.40
##  8 Adelaide Hills South Australia     Visiting  14.2
##  9 Alice Springs  Northern Territory  Business  14.6
## 10 Alice Springs  Northern Territory  Holiday   31.9
## # … with 294 more rows
```

mean

```
tourism %>%
  features(Trips, list(mean=mean)) %>%
  arrange(mean)
```

```
## # A tibble: 304 x 4
##    Region          State               Purpose   mean
##    <chr>           <chr>               <chr>    <dbl>
##  1 Kangaroo Island South Australia     Other    0.340
##  2 MacDonnell      Northern Territory  Other    0.449
##  3 Wilderness West Tasmania            Other    0.478
##  4 Barkly          Northern Territory  Other    0.632
##  5 Clare Valley    South Australia     Other    0.898
##  6 Barossa         South Australia     Other    1.02
##  7 Kakadu Arnhem   Northern Territory  Other    1.04
##  8 Lasseter        Northern Territory  Other    1.14
##  9 Wimmera         Victoria            Other    1.15
## 10 MacDonnell      Northern Territory  Visiting 1.18
## # … with 294 more rows
```

Five summary statistics

```
tourism %>%
  features(Trips,
           quantile,
           prob=seq(0,1,by=0.25))
```

```
## # A tibble: 304 x 8
##    Region         State              Purpose    `0%`   `25%`
##    <chr>          <chr>              <chr>     <dbl>   <dbl>
##  1 Adelaide       South Australia    Busine…  68.7    134.    1
##  2 Adelaide       South Australia    Holiday 108.     135.    1
##  3 Adelaide       South Australia    Other    25.9     43.9
##  4 Adelaide       South Australia    Visiti… 137.     179.    2
##  5 Adelaide Hills South Australia    Busine…   0        0
##  6 Adelaide Hills South Australia    Holiday   0        5.77
##  7 Adelaide Hills South Australia    Other     0        0
##  8 Adelaide Hills South Australia    Visiti…   0.778    8.91
##  9 Alice Springs  Northern Territo… Busine…   1.01     9.13
## 10 Alice Springs  Northern Territo… Holiday   2.81    16.9
## # … with 294 more rows
```

ACF features

ACF features

- All the autocorrelations of a series
- the sum of the first ten squared autocorrelation coefficients is a useful summary of how much autocorrelation there is in a series, regardless of lag.

- autocorrelations of transformations of a time series.

  - "difference" the data and create a new time series consisting of the differences between consecutive observations. Then we can compute the autocorrelations of this new differenced series.

- Occasionally, we may compute the differences of the differences.

- Another related approach is to compute seasonal differences of a series.

`feat_acf()` function

The `feat_acf()` function will return the following features:

- the first autocorrelation coefficient from the original data;
- the sum of square of the first ten autocorrelation coefficients from the original data;
- the first autocorrelation coefficient from the differenced data;
- the sum of square of the first ten autocorrelation coefficients from the differenced data;
- the first autocorrelation coefficient from the twice differenced data;
- the sum of square of the first ten autocorrelation coefficients from the twice differenced data;
- For seasonal data, the autocorrelation coefficient at the first seasonal lag is also returned.

ACF features with Australian tourism data

```
tourism %>%
  features(Trips, feat_acf)
```

```
## # A tibble: 304 x 10
##    Region State Purpose     acf1 acf10 diff1_acf1 diff1_acf1
##    <chr>  <chr> <chr>      <dbl> <dbl>      <dbl>       <dbl
##  1 Adela… Sout… Busine…  0.0333  0.131     -0.520       0.46
##  2 Adela… Sout… Holiday  0.0456  0.372     -0.343       0.61
##  3 Adela… Sout… Other    0.517   1.15      -0.409       0.38
##  4 Adela… Sout… Visiti…  0.0684  0.294     -0.394       0.45
##  5 Adela… Sout… Busine…  0.0709  0.134     -0.580       0.41
##  6 Adela… Sout… Holiday  0.131   0.313     -0.536       0.50
##  7 Adela… Sout… Other    0.261   0.330     -0.253       0.31
##  8 Adela… Sout… Visiti…  0.139   0.117     -0.472       0.23
##  9 Alice… Nort… Busine…  0.217   0.367     -0.500       0.38
## 10 Alice… Nort… Holiday -0.00660 2.11      -0.153       2.11
## # … with 294 more rows, and 2 more variables: diff2_acf10 <d
## #   season_acf1 <dbl>
```

STL Features

STL features: trend and seasonality

- The STL decompositions discussed in Chapter 3.

A time series decomposition can be used to measure the strength of trend and seasonality in a time series. Recall that the decomposition is written as:
$y_t = T_t + S_t + R_t,$

- measure of the strength of the trend: $F_T = \max\left(0, 1 - \frac{\text{Var}(R_t)}{\text{Var}(T_t + R_t)}\right).$

- measure of the strength of the seasonality:
  $F_S = \max\left(0, 1 - \frac{\text{Var}(R_t)}{\text{Var}(S_t + R_t)}\right).$

feat_stl() function

The `feat_stl()` function returns several more features:

- *`seasonal_peak`* measures timing of peaks — which season (e.g.month or quarter) contains the largest seasonal component
- *`seasonal_troughs`* measures timing of troughs — which season (e.g.month or quarter) contains the smallest seasonal component.
- *`spikiness`* measures the prevalence of spikes in the remainder component
- *`linearity`* measures the linearity of the trend component of the STL decomposition.
- *`curvature`* measures the curvature of the trend component of the STL decomposition.
- *`stl_e_acf1`* is the first autocorrelation coefficient of the remainder series.
- *`stl_e_acf10`*: is the sum of squares of the first ten autocorrelation coefficients of the remainder series.

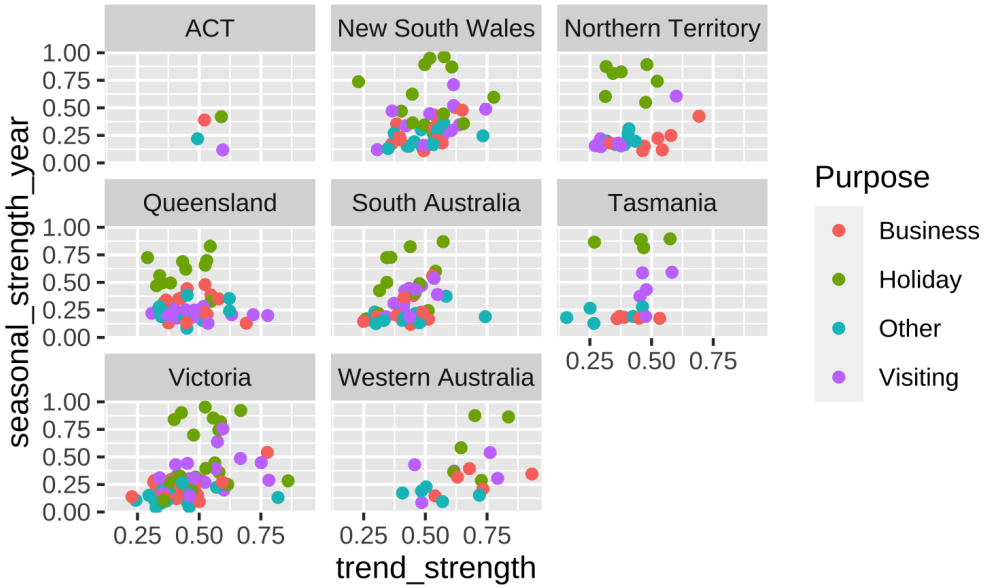
STL features with Australian tourism data

```
tourism %>%
  features(Trips, feat_stl)
```

```
## # A tibble: 304 x 12
##    Region State Purpose trend_strength seasonal_streng… seas
##    <chr>  <chr> <chr>             <dbl>             <dbl>
##  1 Adela… Sout… Busine…           0.451             0.380
##  2 Adela… Sout… Holiday           0.541             0.601
##  3 Adela… Sout… Other             0.743             0.189
##  4 Adela… Sout… Visiti…           0.433             0.446
##  5 Adela… Sout… Busine…           0.453             0.140
##  6 Adela… Sout… Holiday           0.512             0.244
##  7 Adela… Sout… Other             0.584             0.374
##  8 Adela… Sout… Visiti…           0.481             0.228
##  9 Alice… Nort… Busine…           0.526             0.224
## 10 Alice… Nort… Holiday           0.377             0.827
## # … with 294 more rows, and 6 more variables: seasonal_troug
## #   spikiness <dbl>, linearity <dbl>, curvature <dbl>, stl_e
## #   stl_e_acf10 <dbl>
```

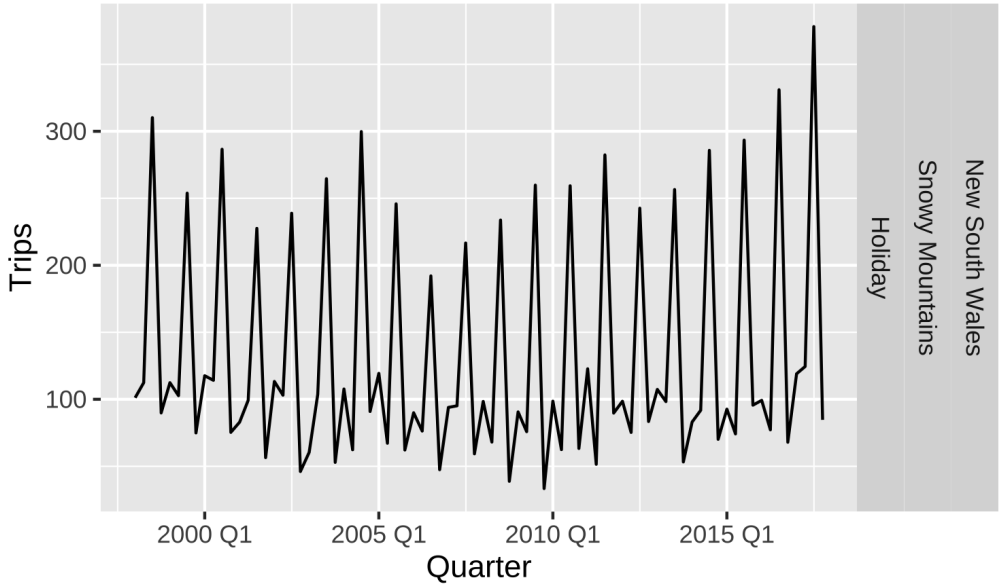
STL features with Australian tourism data

```
tourism %>%
  features(Trips, feat_stl) %>%
  ggplot(aes(x=trend_strength,
             y=seasonal_strength_year,
             col=Purpose)) +
    geom_point() +
  facet_wrap(vars(State))
```



Identify most seasonal series with Australian tourism data

```
tourism %>%
  features(Trips, feat_stl) %>%
  filter(seasonal_strength_year ==
           max(seasonal_strength_year)) %>%
  left_join(tourism,
            by = c("State","Region","Purpose"))
  ggplot(aes(x = Quarter, y = Trips)) +
  geom_line() +
    facet_grid(vars(State,Region,Purpose))
```



Other Features

The remaining features in the `feasts` package

- *`feat_spectral`* will compute the (Shannon) spectral entropy of a time series, which is a measure of how easy the series is to forecast.
- *`coef_hurst`* will calculate the Hurst coefficient of a time series which is a measure of "long memory". A series with long memory will have significant autocorrelations for many lags.
- *`box_pierce`* gives the Box-Pierce statistic for testing if a time series is white noise, and the corresponding p-value.
- *`ljung_box`* gives the Ljung-Box statistic for testing if a time series is white noise, and the corresponding p-value.
- *`feat_pacf`* function contains several features involving partial autocorrelations
- *`unitroot_kpss`* gives the Kwiatkowski-Phillips-Schmidt-Shin (KPSS) statistic for testing if a series is stationary, and the corresponding p-value.
- *`unitroot_pp`* gives the Phillips-Perron statistic for testing if a series is non-stationary, and the corresponding p-value.

The remaining features in the `feasts` package (continue)

- `unitroot_ndiffs` gives the number of differences required to lead to a stationary series based on the KPSS test.
- `unitroot_nsdiffs` gives the number of seasonal differences required to make a series stationary.
- `var_tiled_mean` gives the variances of the "tiled means" (i.e., the means of consecutive non-overlapping blocks of observations).
- `var_tiled_var` gives the variances of the "tiled variances" (i.e., the variances of consecutive non-overlapping blocks of observations).
- `shift_level_max` finds the largest mean shift between two consecutive sliding windows of the time series.
- `shift_level_index` gives the index at which the largest mean shift occurs.
- `shift_var_max` finds the largest variance shift between two consecutive sliding windows of the time series.

The remaining features in the `feasts` package (continue)

- *`shift_var_index`* gives the index at which the largest mean shift occurs
- *`shift_kl_max`* finds the largest distributional shift (based on the Kulback-Leibler divergence) between two consecutive sliding windows of the time series.
- *`shift_kl_index`* gives the index at which the largest KL shift occurs.
- *`n_crossing_points`* computes the number of times a time series crosses the median.
- *`longest_flat_spot`* computes the number of sections of the data where the series is relatively unchanging.
- *`stat_arch_lm`* returns the statistic based on the Lagrange Multiplier (LM) test of Engle (1982) for autoregressive conditional heteroscedasticity (ARCH).
- *`guerrero`*`computes the optimal`(λ)` value for a Box-Cox transformation using the Guerrero method

Exploring Australian tourism data

Cardiff Business School - Dr. Bahman Rostami-Tabar

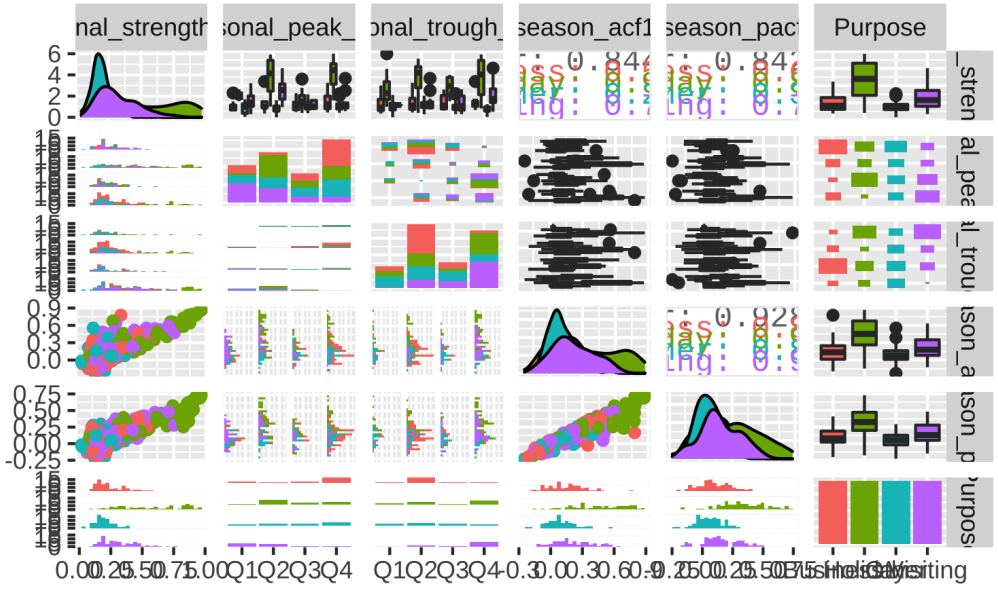
Exploring Australian tourism data

```
tourism_features <- tourism %>%
  features(Trips,
           feature_set(pkgs="feasts"))
tourism_features
```

```
## # A tibble: 304 x 51
##    Region State Purpose trend_strength seasonal_streng… seas
##    <chr>  <chr> <chr>             <dbl>             <dbl>
##  1 Adela… Sout… Busine…           0.451             0.380
##  2 Adela… Sout… Holiday           0.541             0.601
##  3 Adela… Sout… Other             0.743             0.189
##  4 Adela… Sout… Visiti…           0.433             0.446
##  5 Adela… Sout… Busine…           0.453             0.140
##  6 Adela… Sout… Holiday           0.512             0.244
##  7 Adela… Sout… Other             0.584             0.374
##  8 Adela… Sout… Visiti…           0.481             0.228
##  9 Alice… Nort… Busine…           0.526             0.224
## 10 Alice… Nort… Holiday           0.377             0.827
## # … with 294 more rows, and 45 more variables: seasonal_trou
## #   spikiness <dbl>, linearity <dbl>, curvature <dbl>, stl_e
## #   stl_e_acf10 <dbl>, acf1 <dbl>, acf10 <dbl>, diff1_acf1 <
## #   diff1_acf10 <dbl>, diff2_acf1 <dbl>, diff2_acf10 <dbl>,
## #   pacf5 <dbl>, diff1_pacf5 <dbl>, diff2_pacf5 <dbl>, seaso
## #   zero_run_mean <dbl>, nonzero_squared_cv <dbl>, zero_star
## #   zero_end_prop <dbl>, lambda_guerrero <dbl>, kpss_stat <d
## #   kpss_pvalue <dbl>, pp_stat <dbl>, pp_pvalue <dbl>, ndiff
## #   nsdiffs <int>, bp_stat <dbl>, bp_pvalue <dbl>, lb_stat <
## #   lb_pvalue <dbl>, var_tiled_var <dbl>, var_tiled_mean <db
## #   shift_level_max <dbl>, shift_level_index <dbl>, shift_va
## #   shift_var_index <dbl>, shift_kl_max <dbl>, shift_kl_inde
## #   spectral_entropy <dbl>, n_crossing_points <int>, n_flat_
## #   coef_hurst <dbl>, stat_arch_lm <dbl>
```

all the features involving seasonality

```
tourism_features %>%
  select_at(vars(contains("season"),
                 Purpose)) %>%
  mutate(
    seasonal_peak_year =
      glue::glue("Q{seasonal_peak_year+1}"),
    seasonal_trough_year =
      glue::glue("Q{seasonal_trough_year+1}"),
  ) %>%
  GGally::ggpairs(mapping =
                    aes(colour=Purpose))
```

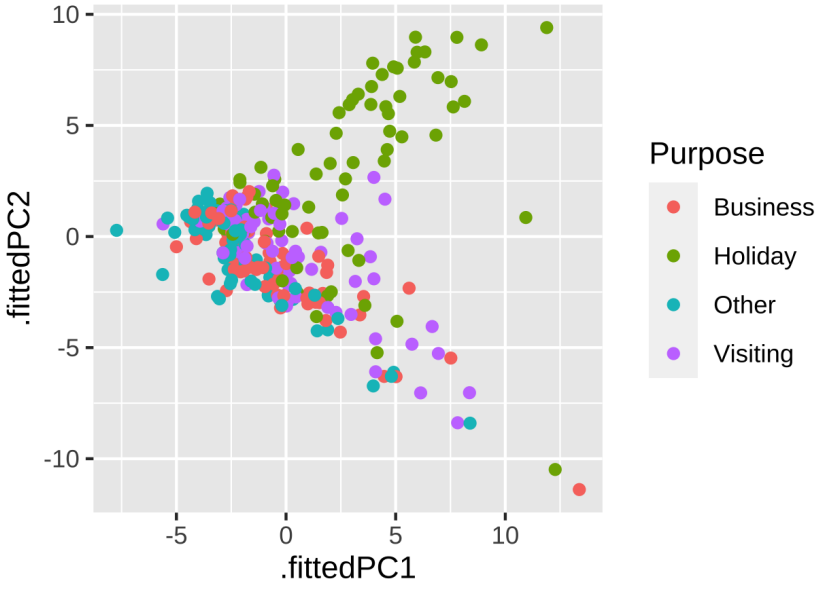


Dimension reduction technique

- It is difficult to explore more than a handful of variables seperately.
- A useful way to handle many more variables is to use a dimension reduction technique such as `principal components`.
- This gives linear combinations of variables that explain the most variation in the original data. We can compute the principal components of the tourism features as follows.

principal components for Australian toursime

```
library(broom)
pcs <- tourism_features %>%
  select(-State, -Region, -Purpose) %>%
  prcomp(scale=TRUE) %>%
  augment(tourism_features)
pcs %>%
  ggplot(aes(x=.fittedPC1,
             y=.fittedPC2,
             col=Purpose)) +
  geom_point() + theme(aspect.ratio=1)
```



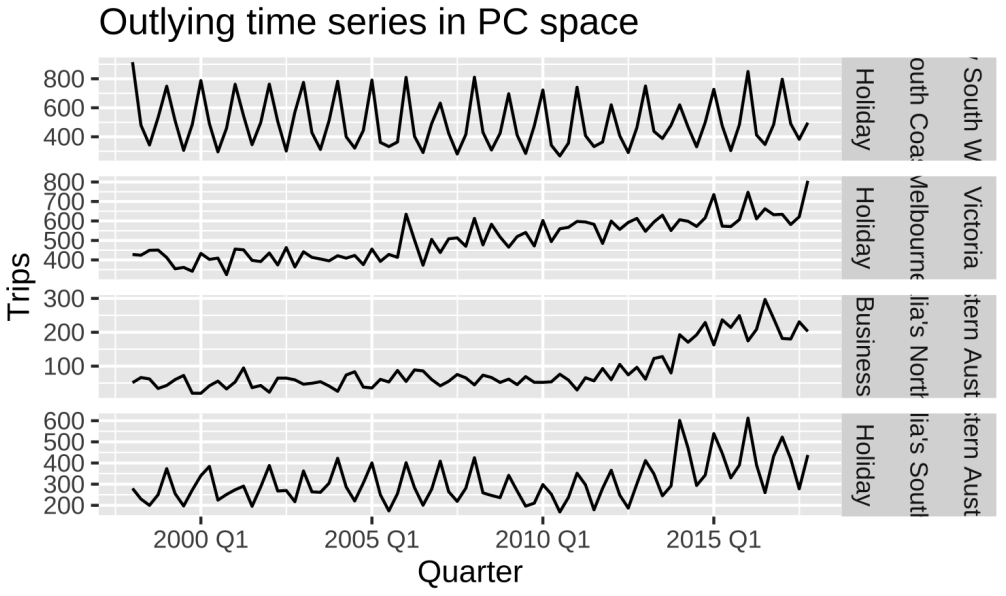
Extract unusual series

```
outliers <- pcs %>%
  filter(.fittedPC1 > 10.5) %>%
  select(Region,
         State,
         Purpose,
         .fittedPC1,
         .fittedPC2)
outliers
```

```
## # A tibble: 4 x 5
##   Region                  State              Purpose
##   <chr>                   <chr>              <chr>
## 1 Australia's North West  Western Australia  Busines
## 2 Australia's South West  Western Australia  Holiday
## 3 Melbourne               Victoria           Holiday
## 4 South Coast             New South Wales    Holiday
```

Plot unusual series

```
outliers %>%
  left_join(tourism,
            by = c("State", "Region", "Purpose"
  mutate(Series =
           glue::glue("{State}", "{Region}", "
                      .sep = "\n\n")) %>%
  ggplot(aes(x = Quarter, y = Trips)) +
    geom_line() +
    facet_grid(Series ~ ., scales='free') +
    ggtitle("Outlying time series in PC space")
```



thank you!

- Slides are available at here
- Email rostami-tabarb@cardiff.ac.uk
- Website www.bahmanrt.com
- Twitter @Bahman_R_T