Paper 3351-2015

Analyzing Parking Meter Transactions Using SAS® Procedures

Anders Milhøj, University of Copenhagen, Denmark

ABSTRACT

The analysis is based upon data for all transactions at four parking meters within a small area in central Copenhagen for a period of four years. The observations are the exact time of the transaction and the number of minutes for parking bought for each transaction. These series of at most 80,000 transactions are aggregated to hours, days, weeks and months using TIMESERIES procedure. The aggregated series of parking times and numbers of transactions are then analyzed for seasonality and interdependence by X12, UCM and VARMAX procedures.

INTRODUCTION

The availability of fast modern computer equipment has made more efficient predicting methods possible for practical use. These facilities include the possibility of analyzing series of time stamped data with many observations and the possibility to efficiently fit statistical models with lots of parameters.

In Copenhagen City more than 2000 parking meters are placed in the streets. The idea is that a motorist has to buy parking time for the period he wants to park. The parking meters receive money or more modern methods for payment like credit cards and mobile pay. But regardless of the method of payment the notice printed on paper by the parking meter has to be placed in the front window of the car. If the note is absent a corps of controllers will issue a fee.

The transactions for each parking meter are stored and published on the Internet. In this presentation data from just four parking meters is used. They are placed close to each other in the center of Copenhagen - in fact the four parking meters closest to the Danish SAS® Office are used.

AGGREGATION

The original data set parking_meter_everything includes the variable tlPayDateTime for the exact time for the transaction as a SAS® date time variable to the precision of minutes. It also includes the time for parking, parkering_time, bought for that transaction. The variable parking_time is aggregated by PROC TIMESERIES.

In the application the option interval=hour tells that the aggregation is to the level of hours. The option accumulate=total tells that the aggregation is performed by summation, so in the new dataset the variable parkering_time now denotes the total time for parking bought by all motorists within the hour. The variable dummy is just the number 1 for each transaction. When aggregated this gives the number of transactions at the parking meter within an hour.

```
proc timeseries data=wrk8.parking_meter_everything
      out= parking_meter_ plots=all;
   by id;
   id tlPayDateTime interval=hour accumulate=total setmiss=0
      start='01jan2010:00:00:00'dt
      end  ='31dec2013:00:00:00'dt;
   var dummy parking_time;
run;
```

Aggregation to other levels like day, week and month is performed in a similar way. The dataset parking_meter_month includes for all 48 months in the years 2010 to 2013 the number of parking meter transactions and the total parking time bought by these transactions. The parking time is measured in minutes. The data sets parking_meter_day and parking_meter_week similarly gives the values of the same variables at the level of days and weeks.

ANALYZING THE MONTHLY SERIES USING PROC X12

PROC X12 is a procedure in the ETS package designed to perform seasonal adjustment by the Census X12 method. This method is the most recent version of the series of adjustment methods derived by the Bureau of the Census. The basic part of the seasonal adjustment is however exactly the same as the previous version as found in PROC X11 but some improvement has been made. The minor enhancements are useful also if seasonal adjustment is out of focus because these new features in an simple way give the analyst some feeling of the behavior of the time series. PROC X12 is in fact useful for time series analysis, see Milhøj(2013).

The code presents an extended application of PROC X12 for the number of transactions for one parking meter; the variable number and for the variable average_parking_time . The series of the monthly total numbers of transactions for one parking meter and for the average parking time bought within a month both have a constant mean. Trends are often present in macroeconomic time series, but this particular series has no trend. In the application the statement x11 mode=add specifies that a basic seasonal adjustment is performed by the additive version of the X11 method. The default is a multiplicative version which is more appropriate for series with trends because the multiplicative version considers seasonality as a relative feature and not an additive. The multiplicative version is mainly of interest for series with a trend.

The output statement specifies where to store the adjusted series together with some series of interest for the adjustment algorithm. The remaining statements are mainly the enhancements for PROC X12 which are discussed later.

```
proc x12 data=Parking_3443_month date=date;
   var number;
   x11 mode=add;
   regression predefined=td;
   automdl;
   outlier;
   forecast lead=12 ;
   output out=out_number a1 d11 d10 d12 d13 c17;
run;
proc x12 data=Parking_3443_month date=date;
   var average_parking_time;
   x11 mode=add;
   regression predefined=td;
   automdl;
   outlier;
   forecast lead=12 ;
   output out=out_time a1 d11 d10 d12 d13 c17;
run;
```

The output series are plotted by PROC SGPLOT. The codes for plotting for all series are like the following which plots the original series, denoted `number_A1` in the output dataset, and the seasonally adjusted series, denoted `number_D11` in the output dataset, for the series of the total number of transactions.

```
PROC SGPLOT data=out_number;
series x=date y=number_A1/markers;
series x=date y=number_D11/markers;
xaxis values=('1jan2010'd to '1jan2014'd by qtr2);
run;
```

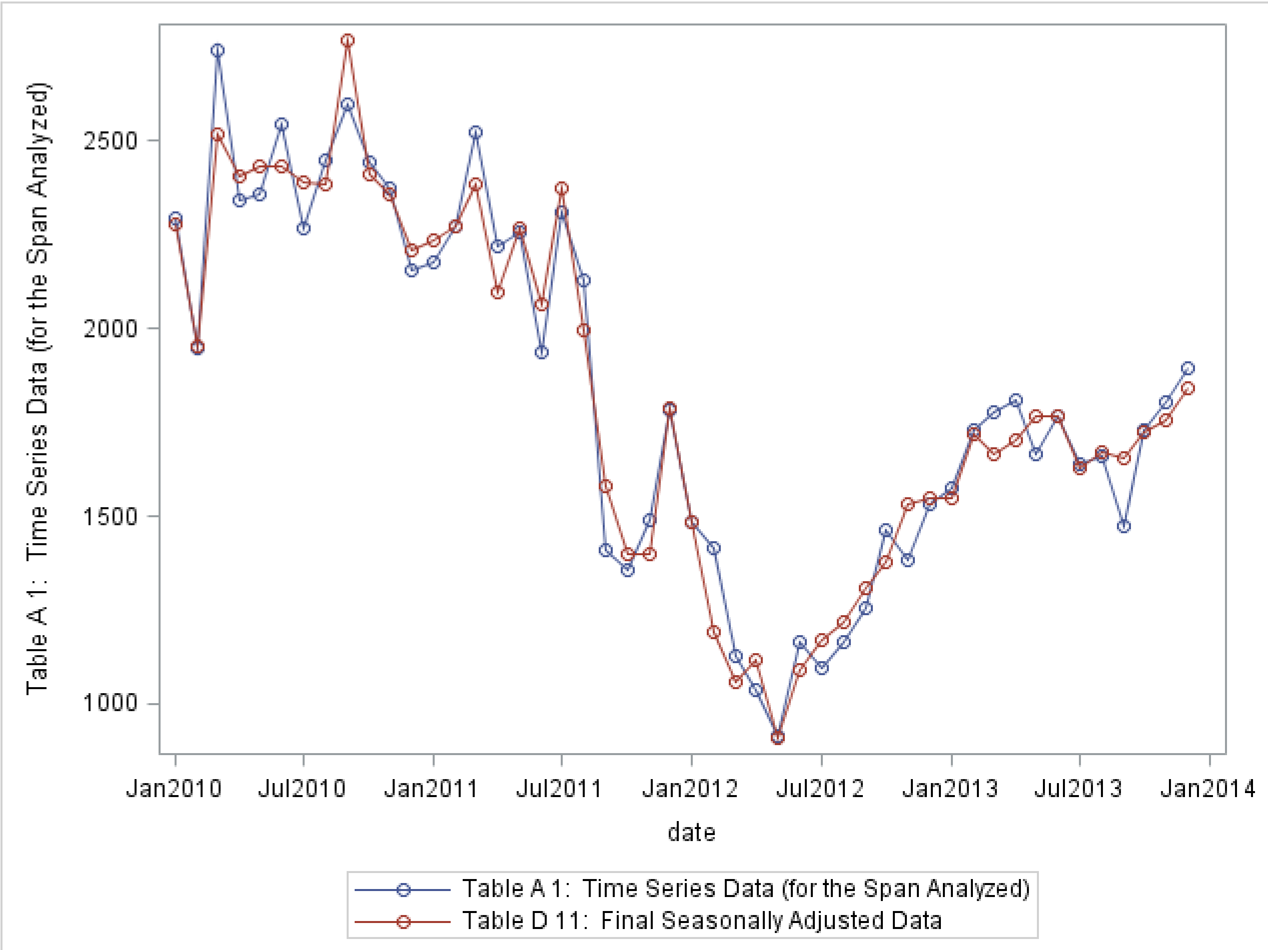

Figure 1 The original and the seasonally adjusted series of number of transactions

Figure 2 gives a plot of the seasonal component for the series of average parking time; the output series denoted d10 in the output dataset by PROC X12. It is seen that the average parking time is shortest in the summer time probably because short period of parking is bought by people who are out shopping or sightseeing while longer parking times are bought by people who go to work or attend meetings which last longer than a small visit at say a clothing shop.

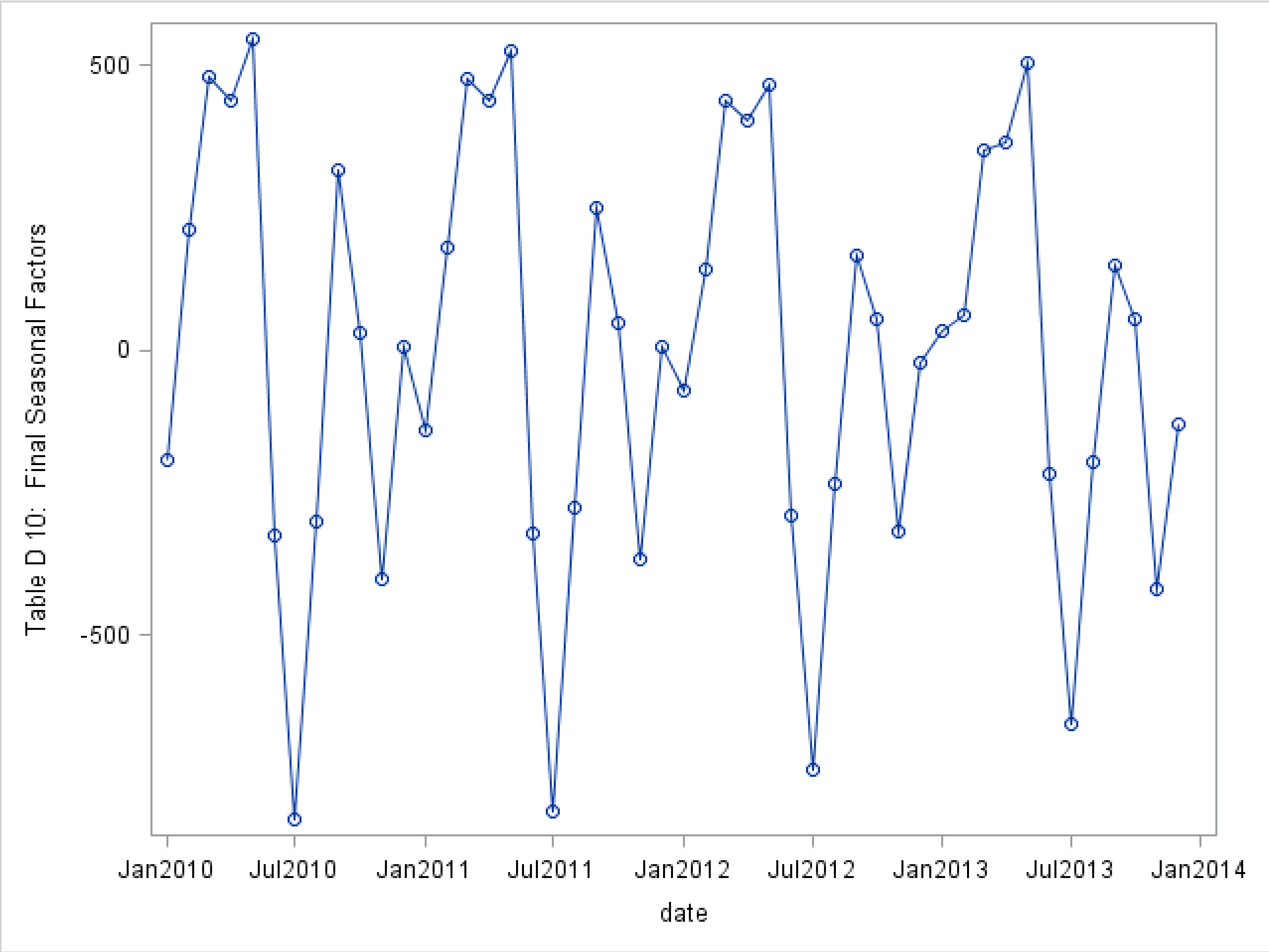

Figure 2 The seasonal component for the average parking time

The regression statement gives the analyst many possibilities for adjusting the series of using deterministic effects to the series. This is important because especially the Easter holiday season could be in March or April (or both). Many such calendar effects are available by options to the predefined option. In this example only a trading day, td, correction is used as it is obvious that the parking behavior varies over the week. Moreover parking is much cheaper or even free on Sundays. This is the reason that the average parking time is larger on Sundays.

The individual corrections for the particular days of the week are hardly significant according to the printed output. But the test for the composite hypothesis that the trading day effect is negligible, clearly tells that the trading day effect is significant as seen in the table. The effect of a leap year is of no importance which is to be expected as the average parking time is independent of the number of days in February.

The outlier statement identifies outliers in the time series. The outliers are indentified by correct statistical outlier tests having a critical value which depends on the number of observations. In Analytical Updates 13.2 even more flexibility is offered to the statement in form of options. The output presented here just reports the standard outlier detection. The result is that two outliers are found both for the month of December.

Regression Model Parameter Estimates						
For Variable average_parking_time						
Type	*Parameter*	*NoEst*	*Estimate*	*Standard Error*	*t Value*	*Pr > |t|*
Trading Day	MON	Est	-76.78213	374.99209	-0.20	0.8389
	TUE	Est	-771.66753	381.98454	-2.02	0.0505
	WED	Est	-114.83640	360.25126	-0.32	0.7516
	THU	Est	246.02217	353.63666	0.70	0.4909
	FRI	Est	94.86286	372.37268	0.25	0.8003
	SAT	Est	-62.78797	368.92555	-0.17	0.8658
	SUN(derived)*	Est	685.18899	359.76832	1.90	0.0644
Leap Year	Leap Year	Est	-1326.4987	1108.10076	-1.20	0.2387
Constant	Constant	Est	13330.7193	221.20539	60.26	<.0001
Automatically Identified	AO DEC2012	Est	7449.00469	980.54969	7.60	<.0001
	AO DEC2013	Est	4188.36481	978.96571	4.28	0.0001

The automdl statement means that a Box & Jenkins(1976) seasonal ARIMA model is fitted to the series. This fitted model is used in the seasonal adjustment algorithm to forecast the observed series in order to stabilize the seasonal adjustment of the most recent observations. This extension to the X11 algorithm is denoted X11-ARIMA and it is one of the enhancements offered by PROC X12. The specific form of the model is derived by a model selection algorithm which compares various model orders to a standard model, the famous Box & Jenkins airline model. In this way PROC X12 automatically performs the rather troublesome Box & Jenkins modeling procedure which usually has to be performed manually by PROC ARIMA; see Brocklebank and Dickey(2003) or Milhøj(2013) for applications of PROC VARMAX.

The resulting model for this application for the series of average parking times is a non-seasonal ARIMA(1,1) model. The parameters of this model are estimated by exact maximum likelihood and they are reported in the output as

Exact ARMA Maximum Likelihood Estimation					
For Variable average_parking_time					
Parameter	*Lag*	*Estimate*	*Standard Error*	*t Value*	*Pr > |t|*
Nonseasonal AR	*1*	0.92298	0.10955	8.43	<.0001
Nonseasonal MA	*1*	0.85740	0.15155	5.66	<.0001

The fact that no seasonal components are needed in the best Box & Jenkins model indicates that the seasonality of the series is only of minor importance in contrast to the impression you get from figure 2.

The series is forecasted by the forecast statement. The forecasts are derived a year ahead by the option lead=12. The forecasts are even plotted by the procedure; Figure 3. In the observation period the forecasts, the solid line, is the series of one step ahead forecasts. The forecasts for the average parking

times in 2014 seem to include some seasonality but relative to the level of the series the seasonal fluctuations are only of minor importance. It has to be stressed that these fluctuations are due to the trading day regression as no seasonality is incorporated in the fitted ARMA model.

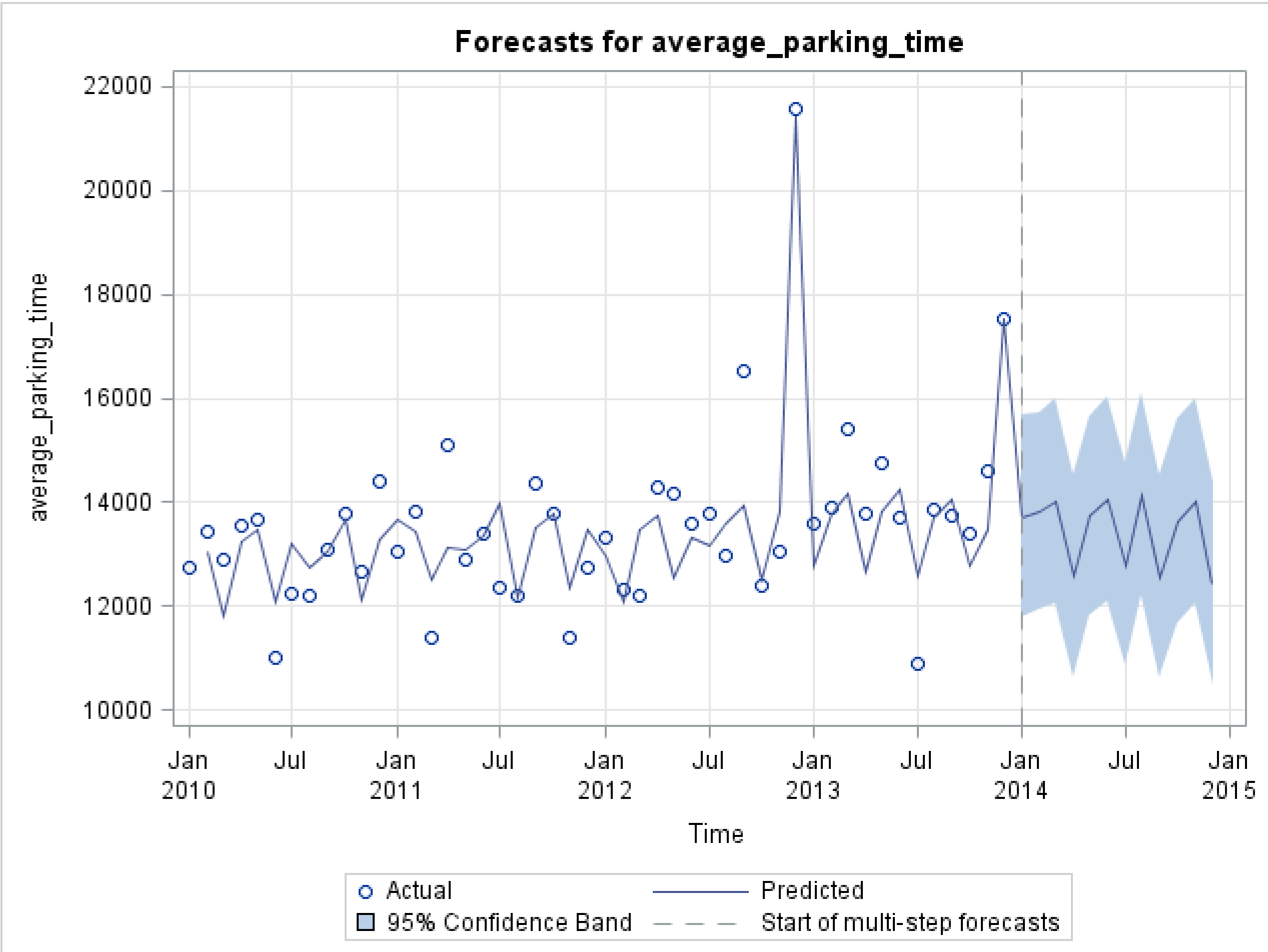

Figure 3 Forecasts of the series of average parking times

PROC VARMAX APPLIED TO A FOUR DIMENSIONAL SERIES

In this section PROC VARMAX is applied to the four dimensional series of numbers of transactions for all four parking meters considered. Models for multivariate time series usually depend on many parameters and so it is only possible to fit them if the time series are rather long, so here we analyze the weekly data series with 624 (full) weeks in the four year observation period.

In the following application a VARMA model is fitted using a model selection algorithm which finds the optimal model order according to an information criterion. The data set log_week includes the logarithmically transformed parking time for the four parking meters as the variables ltime_3443 etc. where the number of the parking meter is used as an identification. The option minic (for Minimum Information Criterion) specifies that all VARMA(p,q) models of orders p and q up to five are fitted and the Akaike Information Criterion is printed for each model order p, q up to 5.

```
proc varmax data=log_week ;
  model ltime_3443 ltime_3853 ltime_3881 ltime_3984/
    minic print=diagnose;
  id date interval=week;
run;
```

Minimum Information Criterion Based on AICC						
Lag	*MA 0*	*MA 1*	*MA 2*	*MA 3*	*MA 4*	*MA 5*
AR 0	-9.904593	-9.817377	-9.805507	-9.708511	-9.62176	-9.501152
AR 1	-9.895414	-9.740381	-9.744825	-9.673181	-9.568744	-9.486063
AR 2	-9.85362	-9.722336	-9.642943	-9.545917	-9.414829	-9.348288
AR 3	-9.731303	-9.652616	-9.566546	-9.44878	-9.381306	-9.276043
AR 4	-9.6251	-9.567784	-9.444907	-9.368137	-9.244066	-9.160931
AR 5	-9.50941	-9.463904	-9.393466	-9.275092	-9.15535	-9.023088

The smallest value of the criterion, that is the optimal model, is found for both the autoregressive order, p, and the moving average order, q, equal zero. The conclusion is that no lagged dependence exists meaning that the average parking time for one particular parking meter in one week is independent of all average paring times for any week at any of the four parking meters including the particular parking meter itself. This conclusion that the series forms a four dimensional white noise series is realistic for the present dataset as it is for stock market deviations.

For the number of transactions a second order vector autoregressive model is selected by the automatic model selection algorithm. The variables lnumber_3443 etc. are the logarithmically transformed number of transactions.

```
proc varmax data=log_week ;
  model lnumber_3443 lnumber_3853 lnumber_3881 lnumber_3984/
    minic print=all;
  id date interval=week;
run;
```

This model has 32 parameters in the autoregressive matrices. The significance of all these fitted parameters is displayed in a schematic diagram.

Schematic Representation of Parameter Estimates			
Variable/Lag	*C*	*AR1*	*AR2*
lnumber_3443	+	+.-.	+.+.
lnumber_3853	+	.+..	....
lnumber_3881	.	..+.	....
lnumber_3984	.	...+	...+
*+ is > 2*std error, - is < -2*std error, . is between, * is N/A*			

It is seen that the first and the last parking meter both have univariate models of order two while the order for the marginal autoregressive model for the other two parking meters is just one. Moreover the first parking meter to some extend (at least significantly at a 5% level) depends on the parking time for the third parking meter at both lag one and two. This last conclusion is perhaps surprising. The p-values are however only p = 1% and the estimated parameters are both numerically equal to 0.08 and hence of very little importance.

The numbers of transactions are correlated at lag zero. This is rather obvious as parking in these four areas, which are situated very close to each other, is expected to behave in parallel.

Cross Correlations of Residuals					
Lag	Variable	lnumber_3443	lnumber_3853	lnumber_3881	lnumber_3984
0	lnumber_3443	1.00000	0.32710	0.36671	0.25635
	lnumber_3853	0.32710	1.00000	0.17717	0.11131
	lnumber_3881	0.36671	0.17717	1.00000	0.07161
	lnumber_3984	0.25635	0.11131	0.07161	1.00000

One problem exists for the model fit: The tests for ARCH effects all show significance; the model assumptions of constant variance is in error and instead volatility clustering is present. This means that the variation in the four series is high for some periods and low for other periods. This is commonly seen for financial data where stock market behavior varies between stable periods and very volatile periods.

Univariate Model White Noise Diagnostics					
		Normality		ARCH	
Variable	Durbin Watson	Chi-Square	Pr > ChiSq	F Value	Pr > F
lnumber_3443	2.06414	150.88	<.0001	21.34	<.0001
lnumber_3853	2.04887	19.79	<.0001	16.74	<.0001
lnumber_3881	2.02761	213.21	<.0001	11.12	0.0010
lnumber_3984	1.99345	24.85	<.0001	10.36	0.0015

In the following application the second order autoregressive model is extended by an ARCH model of order 1. This could hopefully also help the problem of non-normality which was also seen from the white noise diagnostic table. As the number of parameters is now very large the numerical estimation a further extension to a GARCH(1,1) model becomes unstable.

```
proc varmax data=log_week ;
  model lnumber_3443 lnumber_3853 lnumber_3881 lnumber_3984/
    p=2 q=0 print=all;
  garch q=1;
  id date interval=week;
run;
```

It is obvious from the schematic presentation of the estimated ARCH parameters that the fitted model includes much too many insignificant parameters as was also the case for the second order autoregressive model. In PROC VARMAX all the superfluous parameters could be restricted to zero leading to a model with only a few parameters. Moreover the specific form of the GARCH model is the BEKK parameterization for a multivariate GARCH which is a very flexible parameterization with many parameters instead of more sparse CCC or DCC parameterizations also offered by PROC VARMAX. These possibilities are however not pursued in the present paper; see Milhøj(2015).

Schematic Representation of GARCH Parameter Estimates		
Variable/Lag	*GCHC*	*ACH1*
h1	++++	++-.
h2	*++.	....
h3	**++	..+.
h4	***+	...+
*+ is > 2*std error, - is < -2*std error, . is between, * is N/A*		

UNOBSERVED COMPONENT MODEL FOR THE DAILY SERIES

In this section the series of the daily number of transactions is considered. It is obvious that a systematic weekly variation exists. In a time series model the weekday effects are modeled by inclusion of dummy variables for Sundays, Mondays etc. The analysis is performed by PROC UCM which decompose the observed time series as a sum of stochastic unobserved components. The unobserved components are estimated for the observed data series by an application of the Kalman filter but it is unnecessary for the user to care about the details in the estimation.

In the application 3 cycles are included in the model to allow for systematic variation over a calendar year; notice however that the problem of such seasonality was only of minor importance in the previous analyses. These cycles are specified as completely deterministic by fixing them by options to the cycle statements. The 3 cycle lengths correspond to the number of days each year, half year and quarter of a year.

The weekday effects are modeled by regressions having a time varying regression coefficients. This allows the effect of a Wednesday, say, to be different in summer times than in the rest of the year. In PROC UCM this is possible by the randomreg statements.

The results are presented by a huge number of plots and output tables. The regression coefficient for Wednesdays is presented in Figure 4. This regression coefficient is rather volatile and almost zero a few weeks; probably because of specific events that prohibit parking these Wednesdays. The forecast of the Wednesday effect is a constant according to the last observation of the regression coefficient.

```
PROC UCM data=Parking_3443_day;
  id date interval=day;
  model number;
  randomreg  Monday /plot=smooth;
  randomreg  Tuesday /plot=smooth;
  randomreg  Wednesday /plot=smooth;
  randomreg  Thursday /plot=smooth;
  randomreg  Friday /plot=smooth;
  randomreg  Saturday/plot=smooth;
  level plot=smooth ;
  cycle period=365.24 rho=1  variance=0
    noest=(rho period variance ) plot=smooth;
  cycle period=182.62 rho=1  variance=0
    noest=(rho period variance) plot=smooth;
  cycle period=91.31 rho=1  variance=0
    noest=(rho period variance) plot=smooth;
  estimate plot=(panel residual);
  forecast lead=62 back=31 plot=all outfor=forecasts;
run;
```

```
proc sgplot data=forecasts;
  series y=forecast x=date/;
  scatter y=number x=date/markerattrs=(symbol=diamond color=red);
  series x=date y=LCL/lineattrs=(pattern=solid color=black);
  series x=date y=UCL/lineattrs=(pattern=solid color=black);
  where (year(date)=2013 and month(date)=12)
    or (year(date)=2014 and month(date)=1);
run;
```

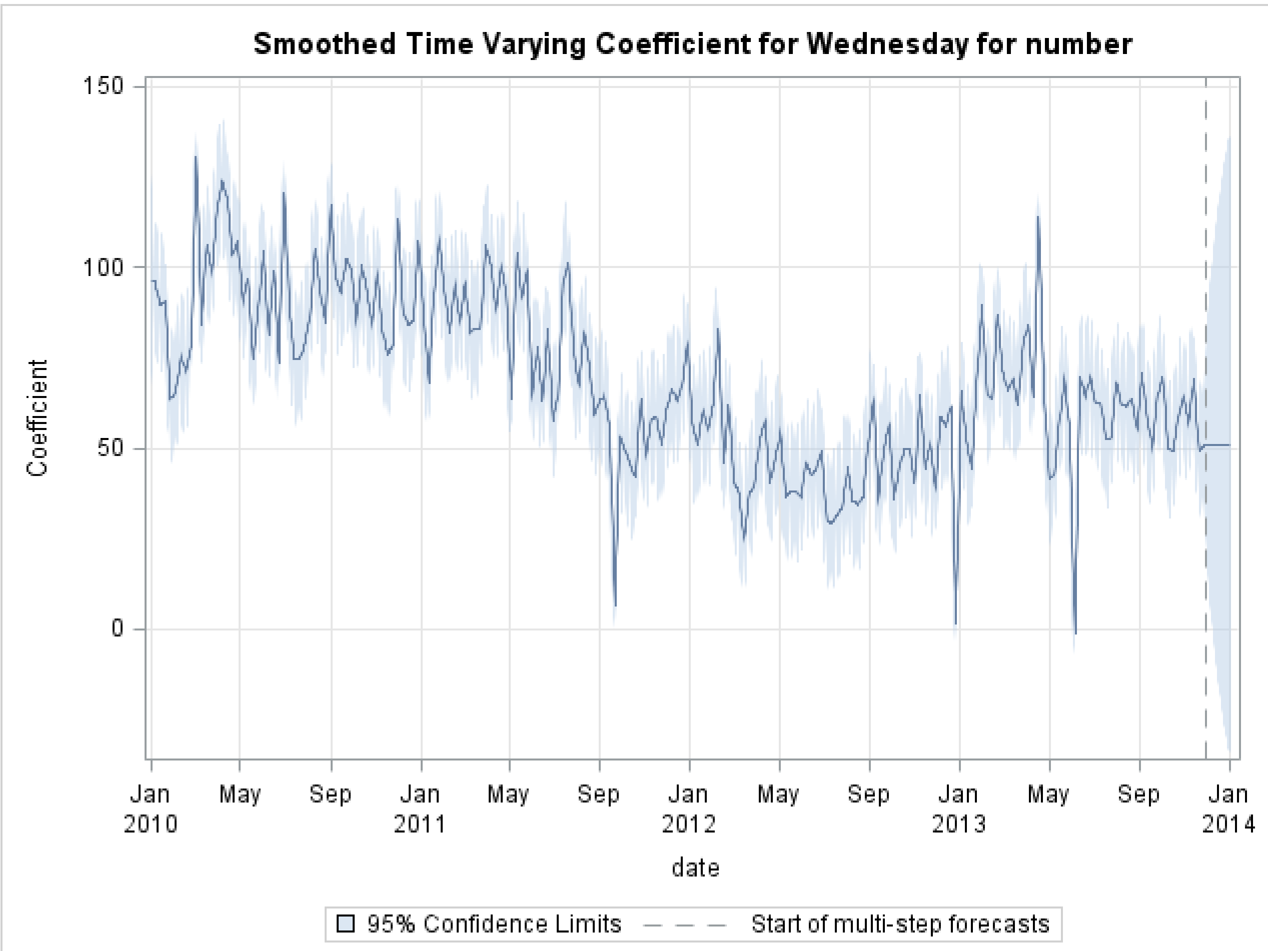

Figure 4 The time varying regression coefficients for Wednesdays

The cycles are of only minor importance. The table of the estimated cycle parameters tells that the cycles variation, the amplitude, is only around 10% of the actual level of the series.

Summary of Cycles								
Name	*Type*	*Period*	*Frequency*	*Damping Factor*	*Final Amplitude*	*Percent Relative to Level*	*Cycle Variance*	*Error Variance*
Cycle_1	*Deterministic*	365.24000	0.01720	1.00000	0.19962	3.82580	.	0
Cycle_2	*Deterministic*	182.62000	0.03441	1.00000	0.53587	10.27001	.	0
Cycle_3	*Deterministic*	91.31000	0.06881	1.00000	0.53065	10.17003	.	0

The number of observations is large, 1461, so plots of the observations have no meaning. It is more convenient to look at the forecasts for shorter periods. By the `forecast` statement the series is predicted for the last months of the estimation period, December 2013, and for the next month, January 2014. This is obtained by the `lead=62` and `back=31` options. The forecasts are saved in the data set named `forecasts`. Finally they are plotted by PROC SGPLOT in the last part of the code.

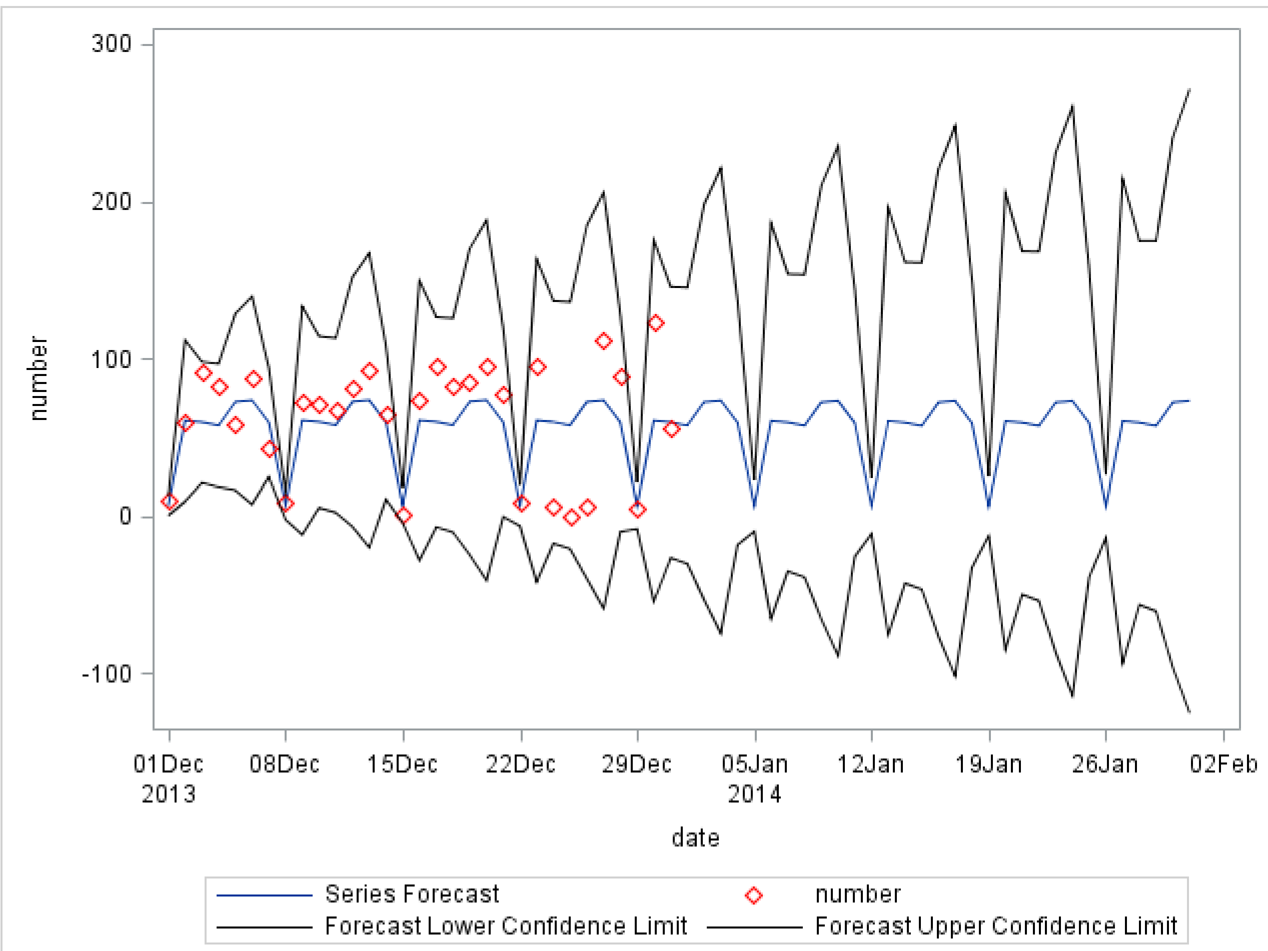

Figure 5 Forecasts of the series of numbers of transactions

CONCLUSION

It is demonstrated how to apply various SAS® procedures to analyze time series data:

- PROC TIMESERIES is applied to aggregate data series in two ways
- PROC X12 is applied to perform a seasonal adjustment
- PROC X12 is applied to fit an ARIMA model to univariate time series
- PROC VARMAX is applied to fit a multivariate time series model
- PROC VARMAX is applied to fit a multivariate GARCH model
- PROC UCM is applied decompose a univariate series in many components with a clear interpretation
- PROC UCM is applied to fit a regression model with time varying regression coefficients

REFERENCES

Box, G.E.P. and Jenkins, G.M. (1976), *Time Series Analysis: Forecasting and Control*, Revised Edition, San Francisco: Holden-Day.

Brocklebank, J. C. and Dickey, D. A. (2003*), SAS for Forecasting Time Series*, Second Edition, , Cary NC: SAS Institute Inc.

Milhøj, A. (2013), *Practical Time Series Analysis Using SAS®*, SAS Press, Cary, NC.

Milhøj, A. (2015), *Time Series Modeling Using the SAS® VARMAX Procedure*, To Appear, SAS Press, Cary, NC.

CONTACT INFORMATION

Anders Milhøj
Department of Economics, University of Copenhagen
Øster Farimagsgade 5
DK1353 Copenhagen K
Denmark
Tlf: +45 35323265
Mail: Anders.milhoj@econ.ku.dk
Web: http://www.econ.ku.dk/milhoj/