Structured, Large-Scale Statistical Forecasting Using SAS® Forecast Server
Contents

Introduction................................................................................. 1
A Streamlined Approach to Large-Scale Forecasting .................................................. 1
  Extract, Transform and Load................................................. 2
  Forecast Control Tables.................................................... 2
  Forecasting Using a Forecast Control Table (FCT)......... 3
  Forecasting by Exception.................................................. 4
  Extensions of the Basic Approach.................................. 4
Conclusion.................................................................................. 4
Appendix – Sample Code Demonstrating a Forecast Control Table ........................................ 5
Introduction

With seemingly unlimited data flowing from countless sources, a growing list of stakeholders and all the variables included in analysis, forecasting for larger organizations can resemble a minefield rather than a gold mine. And to do it effectively, large-scale forecasting cannot change from month to month - it must be conducted seamlessly and automatically so it becomes a part of the process rather than a task to be performed. Implementing this type of forecasting can often require more time, staff or financial investment.

SAS® Forecast Server is a powerful statistical forecast engine designed to diagnose and estimate a statistical forecast model for individual time series. Once preferences are created, either the engine can use default settings, or users can tweak the process to fit a specific need. For example, different business units may use different promotional vehicles or need to produce forecasts with different time intervals. Also, some forecast models benefit from creating separate seasonal and non-seasonal versions, and so on. This kind of flexible segmentation approach to statistical forecasting can come at a cost. However, enforcing structure on SAS Forecast Server settings while keeping them easy to adjust and maintain does not have to involve expert programming resources.

Here, we present a structured approach to statistical forecasting that is both flexible and uses the full power of SAS Forecast Server without compromising user friendliness and manageability. It is intended for non-technical business analysts who want to accommodate complex business forecasting problems, and for more technical users who need input on implementing SAS Forecast Server. Therefore, this paper does not cover consumption of the statistical forecast downstream - for instance in a demand planning process.

A Streamlined Approach to Large-Scale Forecasting

The streamlined approach to large-scale forecasting that we will present is based on two proven theories of this type of analytic forecasting:

- **One size does not fit all.** Different time series and segments of time series benefit from different approaches to modeling and forecasting.
- **Forecasting by exception is the exception.** Most forecasts are generated automatically; only a small subset of all series requires manual intervention.

From a purely technical perspective, the approach we’re presenting requires a division of labor between producing forecasts using parameterized SAS batch code and forecasts generated manually using the SAS Forecast Studio graphical user interface (GUI). The setup of the parameterized SAS code is a one-time effort that occurs during implementation. Once in production, the parameters are controlled outside of SAS.

Since setting up a large-scale statistical forecasting system at the outset can seem a daunting task, the following presents a high-level process flow of the steps involved.

From a process perspective, the approach can be depicted as shown in Figure 1.
Extract, Transform and Load

The preliminary steps of the process (source data -> import data) can be described as standard data integration tasks - accessing and extracting the relevant data. With the focus of this paper on large-scale statistical forecasting from an analytical viewpoint, this topic will not be included in-depth here. Instead, we will now discuss the subsequent steps in the process.

Segmentation of the data to be forecast

Following our process flow chart, the third step – segmenting data - is one of the two key tasks included in our overall approach, and it has two levels.

Level one

With SAS Forecast Server, not all series can be forecast using out-of-the-box time series forecasting methods. Therefore, we must identify those series where statistical forecasting makes sense. For instance, we should remove series that are considered “retired.” It also means we must identify series with very little or no historical information, as they cannot easily be forecast using the out-of-the-box methods available in SAS Forecast Server.

In a similar manner, you should exclude series with intermittent demand patterns. The output from intermittent demand models is not directly comparable with the output from the other time series forecasting methods in SAS Forecast Server. This means that reconciling a hierarchy involving the output from intermittent demand models produces meaningless results. Secondly, completely random demand is often confused with intermittent demand, making the use of methods exploiting systematic behavior difficult.

Level two

The second level of segmentation uncovers the related data that is valid for statistical forecasting and can use the same forecasting settings. For instance, you can opt to segment your data based on geography or business area.

Alternatively, you can segment data based on behavior such as promotional intensity, seasonality, etc. Independent of your choice of segmentation, the segment needs to be identifiable in the data - for instance, as part of a data set name or as values in a column. This is why segmentation based on information already existing in master data can be easier to implement, since that information just needs to be integrated into the forecast data.

Both levels of segmentation can be done using SAS. SAS Forecast Server includes a dedicated graphical user interface to analyze and segment time series called SAS Time Series Studio. In earlier versions of SAS, the segmentation can be done using either a programming approach or SAS® Enterprise Guide®.

Forecast Control Tables

The next task is defining and setting up forecasting strategies for different segments. The choice of forecasting strategy depends on the problem at hand, and typically covers the settings available in SAS Forecast Server. These settings cover areas such as model families employed, size of holdout sample and forecast horizon, usage of input variables and hierarchical reconciliation. Using different settings for the different segments in your data helps improve forecast accuracy.

Consider the following simple example of forecasting for the Eastern European region.

You would like to:

- Avoid using Unobserved Component Models (UCM).
- Use a holdout sample of four periods.
- Produce forecasts for a forecast horizon of 12 periods.
- Consider only seasonal models.
- Include an input as a required input.
- Use top-down reconciliation.
In tabular format, the strategy could be represented as follows:

<table>
<thead>
<tr>
<th>Region</th>
<th>UCM</th>
<th>Holdout</th>
<th>Horizon</th>
<th>Force seasonality</th>
<th>Required input</th>
<th>Reconciliation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eastern Europe</td>
<td>N</td>
<td>4</td>
<td>12</td>
<td>Y</td>
<td>Easter</td>
<td>Top down</td>
</tr>
</tbody>
</table>

These settings typically factor in a combination of preliminary analysis and business knowledge/requirements. For instance, in the above case, the business could require a 12-period forecast horizon. Also, analysis shows that because data is scarce, the holdout sample cannot be very large. The lack of data also cast doubt over the validity of the seasonality tests, and since the business analysts think the data contains seasonal demand, we consider only seasonal models. This hypothesis is backed by analysis showing plots of a historic pattern.

Building out this table for the entire business spanning the European market (in this example), you could arrive at a table like the following:

<table>
<thead>
<tr>
<th>Region</th>
<th>UCM</th>
<th>Holdout</th>
<th>Horizon</th>
<th>Force seasonality</th>
<th>Required input</th>
<th>Reconciliation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eastern Europe</td>
<td>N</td>
<td>4</td>
<td>12</td>
<td>Y</td>
<td>Easter</td>
<td>Top down</td>
</tr>
<tr>
<td>South Europe</td>
<td>Y</td>
<td>12</td>
<td>12</td>
<td>N</td>
<td>Bottom up</td>
<td></td>
</tr>
<tr>
<td>Scandinavia</td>
<td>N</td>
<td>12</td>
<td>18</td>
<td>N</td>
<td>December</td>
<td>Top down</td>
</tr>
<tr>
<td>UK &amp; Ireland</td>
<td>N</td>
<td>12</td>
<td>12</td>
<td>Y</td>
<td>Bottom up</td>
<td></td>
</tr>
<tr>
<td>Central Europe</td>
<td>N</td>
<td>12</td>
<td>12</td>
<td>Y</td>
<td>Bottom up</td>
<td></td>
</tr>
</tbody>
</table>

Because this table controls how forecasts for different segments are created, the table is called a “Forecast Control Table.” It is created and maintained outside SAS Forecast Server, for instance in Microsoft Excel or Notepad.

**Forecasting Using a Forecast Control Table (FCT)**

There are several ways to create forecasts in SAS Forecast Server using different specifications for different data segments. One method is to create individual SAS Forecast Studio projects for each segment. Another example involves writing different pieces of SAS code for each segment. Using the FCT and SAS macro facility produces a more efficient and easier way to maintain setup outside of SAS. Consider an Eastern European example of changing the size of the data hold-out sample from four to 12. Using the FCT, this is achieved by opening the table in Excel, changing the value and saving upon exit.

The FCT is used to produce forecasts as follows:

1. Set up a SAS macro containing the code that will run.
2. Import the FCT from the outside data source (Excel, CSV, etc.).
3. Count the number of rows in the FCT and store the number in a macro variable.
4. Create a macro do-loop running from one to the number of rows in the FCT.
5. For each iteration of the loop, read the relevant row from the FCT.
6. Create macro variables based on the input from the FCT.
7. Create the relevant forecast code substituting the settings specified in the FCT with the names of the macro variables generated in step five.
8. Append the output from the latest iteration to the overall output.

This paper’s Appendix provides sample code to illustrate these basic principles.
Forecasting by Exception

Automatic forecasting does not always produce satisfactory results for all series, so it might be the case that we must forecast some of the series using manually created models. SAS Forecast Studio is especially useful for editing and creating forecasting models manually at the individual series level. We recommend, therefore, that you route those series with unsatisfactory results to SAS Forecast Studio for further analysis. This can be done by identifying the relevant series, creating a table containing the historic data for those items, and then creating a SAS Forecast Studio project in batch using the %FSCREATE macro. (The %FSCREATE macro is included with SAS Forecast Server. All macros are documented in the SAS Forecast Server Administrator’s Guide.)

It is critical to be able to identify those series with an unsatisfactory forecast. One direct and simple possibility is to create a filter based on the fit statistics in the OUTSTAT data set produced by PROC HPFENGINE. The fit statistics will – in absence of an out-of-sample forecast – only contain measures of in-sample performance, and therefore only be an indirect hint at true forecast accuracy. To get true forecast accuracy you will need to store your forecasts and compare them to actuals over time.

Accuracy measurements cannot stand alone. It might be the case that there is nothing that can be done about a series with low forecast accuracy. Fit statistics do not measure the “forecastability” – or level of forecast accuracy we can expect. Forecastability depends on the degree of systematic variation in the series, so augmenting the fit statistics filter with a measure of forecastability is often recommended. One option is to measure the coefficient of variation (CV), which is calculated using PROC SUMMARY – bearing in mind the limitations of the approach for highly seasonal data and series that vary due to inputs and events.

After manual validation of the extracted series’ forecasts, the resulting forecasts can be combined with the forecasts from the automatic flow. Several SAS tools, including SAS Enterprise Guide, support this process.

Extensions of the Basic Approach

So far, we have focused on using a control table to create statistical forecasts. A natural extension could be a control table that covers the entire process of data preparation, segmentation and data manipulation after forecasts have been generated.

Also, the approach has demonstrated the concept using SAS Forecast Server software alone. Another natural extension uses SAS Add-In for Microsoft Office and runs the automatic process through a stored process launched within Excel.

By default, SAS Forecast Server utilizes just one processor to produce forecasts. Using SAS/CONNECT® software, it is possible to distribute the generation of forecasts across multiple processors. The Forecast Control Table can be set up to contain information on how to use these additional processors.

Finally, users may want to create several different SAS Forecast Studio projects automatically. They can use the control table content to create values that are used as macro variables in the SAS Forecast Server administrative macros.

Conclusion

Improving forecast accuracy continues to be a key focus area in organizations today. Being able to produce high-quality statistical forecasts is an important step, as it supplies downstream planning processes with high-quality input. SAS Forecast Server is designed to automatically deliver high-quality statistical forecasts quickly and efficiently. However, in some cases the size and complexity of the business problem requires some tweaking of the settings available in SAS Forecast Server to cater for the specific characteristics of different parts of the data. When forecasting a large amount of time series data, the challenge is to create a robust and flexible setup that accommodates various business inputs as well as the specifics of the time series. The approach outlined in this paper produces statistical forecasts in a large-scale setting that allows the user to accomplish just this by combining the powerful statistical forecasting engine of SAS Forecast Server with the flexibility of the SAS programming language.
Appendix - Sample Code Demonstrating a Forecast Control Table

* libname statement;
libname fc "<location of data to be forecast>";

* Import the forecast control table, which is assumed to exist in the file and folder mentioned in the infile statement;

data fcast_control_table;
infile "C:\<your folder>\forecast_control_table.csv" delimiter="," firstobs=2 dsd;
input geography : $30.
use_ucm $
holdout
horizon
force_season $
required_input :$30.
recon_dir : $20.;
run;

proc sort data=fcast_control_table;
by geography;
run;

* Determine number of obs in control table;
%macro obsnvars(ds);
%global nobs;
%let dset=&ds;
%let dsid = %sysfunc(open(&dset));
%if &dsid %then
%do;
%let nobs =%sysfunc(attrn(&dsid,NOBS));
%let rc = %sysfunc(close(&dsid));
  %put Open for data set &dset succeeded - number of rows = &nobs;
%end;
%else
  %put Open for data set &dset failed - %sysfunc(sysmsg());
%mend obsnvars;

%obsnvars(fcast_control_table);

* Build macro;
%macro run_fc;

%do i=1 %to &nobs;
* Create macro variables based on content in the forecast control table;
data _null_;  
set fcast_control_table;
   if _n_ = &i.;
      call symput('geo',geography);   
call symput('short',substr(geography,1,2));
call symput('ucm',use_ucm);
call symput('holdout',holdout);
call symput('horizon',horizon);
call symput('season',force_season);
call symput('req',required_input);
call symput('rec_dir',recon_dir);
run;

* Sort input data;
proc sort data=fc.emea_region_sales_data out=sales_data_sort;
   by region type item date;
run;

* Aggregate detail data to geography level;
proc timeseries data=sales_data_sort out=agg_level;
   by region;
   id date interval=month notsorted;
   var units / accumulate=total;
   var price / accumulate=average;
   var december / accumulate=max;
   var easter / accumulate=max;
run;

* Run HPFDIAGNOSE using parameters read from the forecast control table;
proc hpfdiagnose data=agg_level (where=(region="&geo.")
   holdout=&holdout
   modelrepository=work.mod_rep_&short.
   EXCEPTIONS=CATCH
   outest=diag_&short. ;
   by region;
   id date interval=month;
   forecast units;
   %if &season=Y %then %do;
      esm method=bests;
      trend sdiff=1;
   %end;
   %else %do;
      esm;
   %end;
   %end;
   arimax identify=both;
   %if &ucm=Y %then %do;
      ucm;
   %end;
   %if &req ne %then %do;
      input &req. / required=yes;
   %end;
run;
* Create forecasts;
proc hpfengine data=agg_level(where=(region="&geo."))
  lead=&horizon
  modelrepository=work.mod_rep_&short.
  inest=diag_&short.
  outfor=fc_agg_&short. out=_null_
  errorcontrol=(severity=HIGH, stage=(PROCEDURELEVEL))
  EXCEPTIONS=CATCH;
  by region;
  id date interval=month;
  forecast units;
  %if &req ne %then %do;
    input &req.;
  %end;
run;

* Detail level;
proc hpfdiagnose data=sales_data_sort(where=(region="&geo."))
  holdout=&holdout
  modelrepository=work.mod_rep_&short.
  EXCEPTIONS=CATCH
  outest=diag_&short. ;
  by region type item;
  id date interval=month;
  forecast units;
  %if &season=Y %then %do;
    esm method=bests;
    trend sdiff=1;
  %end;
  %else %do;
    esm;
  %end;
  arimax identify=both;
  %if &ucm=Y %then %do;
    ucm;
  %end;
  %if &req ne %then %do;
    input &req. / required=yes;
  %end;
run;

proc hpfengine data=sales_data_sort(where=(region="&geo."))
  lead=&horizon
  modelrepository=work.mod_rep_&short.
  inest=diag_&short.
  outfor=fc_detail_&short. out=_null_
  errorcontrol=(severity=HIGH, stage=(PROCEDURELEVEL))
  EXCEPTIONS=CATCH;
  by region type item;
  id date interval=month;
  forecast units;
  %if &req ne %then %do;
    input &req.;
  %end;
run;
* Reconciliation step;
proc hpfreconcile sign=positive
   aggdata=fc_agg_&short.
   disaggdata=fc_detail_&short.
   outfor=rec_fc_&short.
   %if %lowcase(%substr(&rec_dir.,1,3))=top %then %do;
      dir=td;
   %end;
   %else %do;
      dir=bu;
   %end;
by region type item;
id date interval=month;
run;

* The final forecasts at the lowest level of the hierarchy;
data final_fc_&short.;
   %if %lowcase(%substr(&rec_dir.,1,3))=top %then %do;
      set rec_fc_&short. (drop=_reconstatus_);
   %end;
   %else %do;
      set fc_detail_&short.;
   %end;
run;

* Create one table containing all forecasts;
%if &i=1 %then %do;
data all_forecasts_emea;
   set final_fc_&short.;
run;
%end;
%else %do;
   proc append base=all_forecasts_emea data=final_fc_&short.;
   run;
%end;
%end;
%mend;
%run_fc;