ABSTRACT
J.D. Power & Associates, a market research firm, produces a forecast of car and truck sales each month. There are about 300 different vehicles. Each is forecasted by month, for five years beyond the current year. However, each vehicle must be individually treated as to such things as capacity, promotion, introduction, phase-out, seasonality, etc. This is in addition to economic factors and historical trends. The challenge was to create a robust, automated system that handles a tremendous amount of data and computations, yet easily allows dynamic user input and control during its execution, is self-contained, and produces extensive and standardized output. This system uses only base SAS®. It runs on a 486 PC under OS/2®.

This paper demonstrates several techniques of dealing with system limitations from a large job, such as unavailable memory and storage. There are other constraints, such as users unfamiliar with SAS, constantly changing input parameters, and the need for rapid fine-tuning of the output. In addition, this paper illustrates a method of easily documenting and maintaining a system that contains over 13,000 lines of code.

INTRODUCTION
J.D. Power & Associates has been primarily a market research firm, known for its consumer satisfaction studies. My charter was to develop a forecast. Its scope was ambitious—a sales forecast, by model year, and the number of selling days in each month, by looking at forecasted sales, slated production, and previous inventory. Capacity is a very real constraint in the market. The forecast must not exceed capacity if it is to have credibility.

Seasonality is an important consideration in the automotive industry. Sales vary greatly with the time of year. Also, this variation is profoundly different among different vehicle segments. If that isn’t enough to keep track of, there are separate seasonal factors, put out by the Dept. of Commerce, for domestic vehicles and foreign vehicles. Industry standards dictate that these factors be used in determining total vehicle sales. The system computes seasonal factors by segment and origin. It then simulates the adjusted sales and adjusts the factors to match the simulated sales using the Dept. of Commerce factors. The result is seasonally adjusted total industry sales in accordance with industry standards, but with each segment maintaining its seasonal identity. In addition to time of year, the system must account for such things as position in the model year, and the number of selling days in each month.

An additional control provided to the forecaster is an adjustable tolerance. The system provides an invaluable conceptual and computational benefit by separating econometrics from the more micro considerations of individual vehicles. This is done by dealing only with market shares within each vehicle segment. This, however, exacts a price. When shares are manipulated individually, they no longer sum to 100%. Of course the final allocation must total 100%, so a normalization is necessary. But many vehicles are pulled in opposite directions in that whenever one vehicle’s share is reduced, the real-world effect is to increase all the others. Were this to be done naively, the final result may not reflect the direction of the drivers and constraints that the forecaster envisioned. So a simple mathematical normalization will not suffice. Hence the concept of tolerance. The tolerance is set by the forecaster (batch or interactive). A tolerance of 1% means that the final share after normalization may not deviate from the user determined share by more that 1%. For example: a vehicle has a previous share of 10% of its segment. User input and extrapolation bring it to 12%. Normalization with all the other vehicles brings it back to 10.8%. But the forecaster had set the tolerance at 5%. This means that the final result cannot deviate from 12% by more than 6% (5% of 12). Therefore the final result will be 11.4%, not 10.8%. The forecaster can set the tolerance at 0%, or set no tolerance. And tolerances can be set differently for each vehicle. This allows the forecaster to control the forecast output in accordance with targeted expectations.
USER INTERFACE

The forecaster may enter all parameters either before running the system, or as the programs run, while receiving interim information. In practice, the complete forecast is run once or twice with lots of gross changes. The forecaster typically analyzes the complete system, or as the programs run, while receiving interim information. The forecaster may enter all parameters either before running the forecast, inputs the parameters during the day, then runs the system overnight. The fine tuning phase follows, with the forecaster running specific segments for specific months while entering the revised parameters interactively so as to observe the effects.

For interactive entry, there are up to five screens that can be requested to appear, in sequence, for interactive input:

1. initial inventory
2. multiplicative factors
3. additive factors
4. overtime or added shifts
5. tolerance

The program creates files of the parameters in a user-friendly format, sometimes with information from processing done up to that point. The X command is used to open these files and show them in the OS/2 editor. Figure 1 shows a portion of the tolerance screen. It shows interim results and the default (or previously set) tolerances. The forecaster simply overtypes any tolerance to be changed, saves the file, closes the window and the SAS program resumes.

For batch entry, there are resident files of the parameters. The forecaster may edit these files with direct edit or SAS code. They can be edited either in SAS, or in OS/2, or off-site with any text editor. The system automatically converts the text file to a SAS file used in the processing. Alternatively, the parameters are in SAS files. Simple code can be used to make global changes to these parameters (e.g., for a certain vehicle, reduce all of 1994-1997 by 5%). Users who don't know SAS, can easily supply this code, as shown in Figure 2.

OUTPUT

The system outputs reports that are extensively used in interacting with the data and the results of the forecast to produce a credible and comprehensive forecast. These include:

1. simulation forecast vs. base-line
2. summary of volumes and shares by segment
3. market shares by make
4. degree of over-constraint
5. cars made in Japan

Figure 3 shows a section of a simulation report, which compares the current forecast to last month's (or other requested forecast).

The system also produces numerous reports to meet the needs of a diverse group of decision makers. Most of these reports are produced only after the forecast is finalized. Some of these are:

1. forecast of units sold
2. seasonally adjusted forecast of units sold
3. forecast of market shares
4. constraint flags
5. model detail flow report

Many reports show at least six months of history for analysis purposes. This means that during the first half of the year they show the last six months of the prior year. During the second half of the year they start with January of the current year. This is easily maintained by having two sets of report programs, in different directories. One need only change the directory name every six months, so the correct report programs are called.

A key robust feature is that one can request any or all reports for any or all segments, either as part of the job stream or stand-alone. Figure 4 shows the code for choosing which reports and segments to run.

OPERATIONAL REQUIREMENTS

1. Can run automatically.
   The entire system can run without user intervention, i.e., in the background. The system uses %INCLUDE members within SAS programs and strings together SAS jobs with OS/2 batch files. Figure 5 illustrates the batch file coding.

2. Can give interim results.
   As seen in Figure 1, the preliminary normalization is shown on the same screen where the forecaster enters and edits the input parameters (e.g., tolerance). Hence, the result of the pre-determined input can be seen before constraints are introduced.

3. Can read files from other environments.
   Historical data comes from an AS/400 database and segment volume data comes from an Excel spreadsheet. Both data streams can be converted to a comma delimiter format. SAS can easily read these files with the DLM option after the INFILE statement.

   The forecaster can easily parameterize which segments and time periods to run. The results of limited time-period simulations are available momentarily (taking about 1 minute per month). The program accomplishes this by a series of update commands on the data files.

5. Allows for easy change of program parameters.
   System parameters such as time periods to run, which editing screens to come up, which reports to run, and default tolerances are normally left on default, but need to be changed for fine-tuning. Figure 6 shows the screen where the forecaster can override the default parameters.

OVERCOMING RESOURCE LIMITATIONS

One SAS resource limitation encountered was the log window. With a job this size, the log window eventually fills up, requiring user intervention to save the log or clear the window. This is circumvented by routing the log to a file with PROC PRINTTO (LOG=). Since the huge log is unwieldy and hard to browse, it is partitioned into separate program logs, and a master or "calling" log.

The forecast takes a long time to run, due to the tremendous number of calculations performed. The average time was about 19 hours on a FAT drive. This was on a 486-33 machine with 16 meg ram. Hardware upgrades and maximally efficient coding made only marginal improvements. However, running it on an HPFS formatted partition on the hard drive made a phenomenal improvement, reducing the time to 12 hours!

Even 16 meg of RAM was insufficient after five or six hours of continuous processing. The memory window would fill up, requiring user intervention. The problem was solved by running each segment in a separate SAS session from a batch file, with the "no display manager" option. The command would resemble:

SAS: C:\SAS\PROGRAMS\4CSTTDS.SAS -NODES

Figure 5 illustrates the use of this command in the OS/2 batch file which executes the entire forecast.

567
OVER-CONSTRAINT

One particularly interesting manifestation of the algorithm is an occasional over-constraint condition. It can happen that after meeting capacity and tolerance constraints the shares do not add to 100%. And only by breaking a constraint can they be made to total 100%. In fact, this happens about 15% of the time. The forecaster can elect to have the program report the discrepancy and continue, or have the system break the constraints in the most equitable way and “solve” the segment in all cases. In practice, the latter is usually chosen. Of course capacities cannot be violated, nor can there be negative shares. So only the tolerances can change.

The solving algorithm increases the tolerance by the smallest percentage necessary to solve the segment. Since this is done on a relative basis, the forecaster still controls the prioritization. For example: Vehicle A has a share of 6% and a tolerance of 1% while Vehicle B has a share of 2% and a tolerance of 10%. The algorithm determines the segment can be solved by augmenting the tolerance by 5% and increasing the shares. Therefore Vehicle A’s share is 0.063% (.06 * [1 + (.01 * 1.05)]) , and Vehicle B’s share is 2.210% (.02 * [1 + (.1 * 1.05)]). Note the absolute deviation of Vehicle A is smaller because the user-set tolerance is smaller. The system generates an over-constraint report when this occurs.

In addition, as an overview, the system generates a report that shows the overall over-constraint activity. Tolerance can be used as an information tool as well as a control tool through this report. The “index” of over-constraint is derived from the amount the tolerance must be augmented, with the sign being the direction of pull. A preponderance of positive indexes indicates shares are pulled downward, i.e., totaling less than 100%. The conclusion is that the segment volume is too high in that case. Likewise, a preponderance of negative indexes indicates shares above 100% and a volume that is too low. Figure 7 shows a portion of this report. It can be seen that the Upper Middle segment has a preponderance of positive indexes for 1997-1998. This indicates that a lowering of total volume for that segment is in order. Here is a case where analyzing the interrelationship of individual vehicles gives feedback on the validity of a previously derived segment total.

GENERAL APPROACH

The overriding concept behind the design of this system is modularity. The system is exceedingly complex, yet is easy to maintain, enhance and document. Macros and %INCLUDE members are used extensively. Processes at the core, such as the algorithm for iterative normalization, are used repeatedly. One level up, a program such as the capacity updater, is called for each month of the forecast, and it calls the core processes. At a higher level, a database manager is called once for each segment and calls these lower level programs repeatedly. The highest SAS level is simply a %LET statement which sets the segment and calls the main program. These entities operate up a chain to an OS/2 batch file.

The processing for all segments is identical, but with different data, parameters, file names, report titles, and other identifiers. The batch file calls a program that sets the segment name with %LET &SEG = and calls the main program. The designator &SEG appears numerous times in the programs to reference those identifiers.

Reports are each in a separate program. There is a program that runs a report across named segments. There is a program that runs named reports for one segment. And, there is a program that runs all reports. The modular design allows maximum flexibility and convenience for the forecaster.

CONCLUSION

This is an illustration of how the Base SAS System can be used for a large application. I found SAS to be far more conducive to modular construction than I had previously thought. Integrating SAS processing with OS/2 commands, both within SAS programs (the X command), and from the OS/2 operating system (batch files), is an essential component of the system.

This forecasting system has been running successfully for a year and a half, for several users, and with virtually no maintenance. It is a tribute to the versatility of SAS.

All the programs contain about 13,000 lines of code. But the core algorithms contain 1,000 lines and the report programs contain 5,800 lines. The application of modularity is illustrated by the fact that when the system executes in entirety, over a million lines of code are processed.

SAS is a registered trademark or trademark of SAS Institute Inc. in the USA and other countries. OS/2 is a registered trademark or trademark of International Business Machines Corporation. ® indicates USA registration.

Other brand and product names are registered trademarks or trademarks of their respective companies.
## Tolerance Editing Screen

Tolerance editing screen

<table>
<thead>
<tr>
<th>Year</th>
<th>Jan</th>
<th>Feb</th>
<th>Mar</th>
<th>Apr</th>
<th>May</th>
<th>Jun</th>
<th>Jul</th>
<th>Aug</th>
<th>Sep</th>
<th>Oct</th>
<th>Nov</th>
<th>Dec</th>
</tr>
</thead>
<tbody>
<tr>
<td>1993</td>
<td>1.2%</td>
<td>1.4%</td>
<td>1.3%</td>
<td>1.4%</td>
<td>1.6%</td>
<td>1.5%</td>
<td>2.0%</td>
<td>2.2%</td>
<td>1.5%</td>
<td>1.7%</td>
<td>1.7%</td>
<td>1.8%</td>
</tr>
<tr>
<td>1994</td>
<td>1.8%</td>
<td>1.8%</td>
<td>1.7%</td>
<td>1.8%</td>
<td>1.8%</td>
<td>1.8%</td>
<td>1.8%</td>
<td>1.8%</td>
<td>1.9%</td>
<td>1.9%</td>
<td>1.9%</td>
<td>1.9%</td>
</tr>
<tr>
<td>1995</td>
<td>1.9%</td>
<td>1.9%</td>
<td>1.8%</td>
<td>1.9%</td>
<td>1.9%</td>
<td>1.9%</td>
<td>1.9%</td>
<td>1.9%</td>
<td>1.9%</td>
<td>1.9%</td>
<td>1.9%</td>
<td>1.9%</td>
</tr>
<tr>
<td>1996</td>
<td>2.1%</td>
<td>2.1%</td>
<td>2.0%</td>
<td>2.0%</td>
<td>2.0%</td>
<td>2.0%</td>
<td>2.0%</td>
<td>2.0%</td>
<td>2.0%</td>
<td>2.1%</td>
<td>2.1%</td>
<td>2.1%</td>
</tr>
<tr>
<td>1997</td>
<td>2.0%</td>
<td>2.0%</td>
<td>2.0%</td>
<td>2.0%</td>
<td>2.0%</td>
<td>2.0%</td>
<td>2.0%</td>
<td>2.0%</td>
<td>2.0%</td>
<td>2.0%</td>
<td>2.0%</td>
<td>2.0%</td>
</tr>
<tr>
<td>1998</td>
<td>2.2%</td>
<td>2.2%</td>
<td>2.2%</td>
<td>2.2%</td>
<td>2.2%</td>
<td>2.2%</td>
<td>2.2%</td>
<td>2.2%</td>
<td>2.2%</td>
<td>2.2%</td>
<td>2.2%</td>
<td>2.2%</td>
</tr>
</tbody>
</table>

* Input segment designator.
* Input model, period, factor change.

```sas
* Input segment designator.
%LET SEG = CLU;

DATA FILE.AFAC&SEG;
SET FILE.AFAC&SEG;
BY NAME MODEL;
ARRAY AF{72};

* Input model, period, factor change.
IF UPCASE(MODEL) = 'VIPER' /* MODEL */
THEN DO i = 13 TO 60 /* PERIOD */;
   AF{i} = AF{i} * .95 /* CHANGE */;
END;
RUN;
```

---

Figure 1 Tolerance Editing Screen

Figure 2 SAS Code to Change Parameters

---

**Choose reports** (Enter N to prevent running of report):
- RUNSIRPT = Y: Simulation
- RUNURRPT = Y: Units (Raw)
- RUNRUNRPT = Y: Multiplicative Factors
- RUNRRRPT = Y: Additive Factors
- RUNRUNFRT = Y: Shares
- RUNRUNRT = Y: Constraint Flags
- RUNRUNAFT = Y: Units (Adjusted, Annualized)
- RUNRUNFORPT = Y: Model Flow

---

Figure 4 Requesting Report(s) for All Segments
### Figure 7: Over-Constraint Index Report

**INDEX of OVER-CONSTRAINT---CARS**

**FORECASTS BEGIN & April, 1993**

| YEAR | SEGMENT    | JAN | FEB | MAR | APR | MAY | JUN | JUL | AUG | SEP | OCT | NOV | DEC |
|------|------------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 1993 | Basic Small |     |     |     |     |     |     |     |     |     |     |     |     |     |
|      | Lower Middle| -13 | -5  | -4  | -3  | -2  | -10 | -7  | -1  | -16 | -16 |     |     |     |
|      | Upper Middle| -2  |     |     |     |     |     |     |     |     |     |     |     |     |
|      | Sporty     |     |     |     |     |     |     |     |     |     |     |     |     |     |
|      | Luxury     |     |     |     |     |     |     |     |     |     |     |     |     |     |
| 1994 | Basic Small |     |     |     |     |     |     |     |     |     |     |     |     |     |
|      | Lower Middle| -15 | -15 | -2  | -24 | -9  | -6  | -13 |     |     |     |     |     |     |
|      | Upper Middle| -1  |     |     |     |     |     |     |     |     |     |     |     |     |
|      | Sporty     |     |     |     |     |     |     |     |     |     |     |     |     |     |
|      | Luxury     |     |     |     |     |     |     |     |     |     |     |     |     |     |
| 1995 | Basic Small |     |     |     |     |     |     |     |     |     |     |     |     |     |
|      | Lower Middle| -15 | -15 | -2  | -24 | -9  | -6  | -13 |     |     |     |     |     |     |
|      | Upper Middle| -1  |     |     |     |     |     |     |     |     |     |     |     |     |
|      | Sporty     |     |     |     |     |     |     |     |     |     |     |     |     |     |
|      | Luxury     |     |     |     |     |     |     |     |     |     |     |     |     |     |
| 1996 | Basic Small |     |     |     |     |     |     |     |     |     |     |     |     |     |
|      | Lower Middle| -2  |     |     |     |     |     |     |     |     |     |     |     |     |
|      | Upper Middle| -2  |     |     |     |     |     |     |     |     |     |     |     |     |
|      | Sporty     |     |     |     |     |     |     |     |     |     |     |     |     |     |
|      | Luxury     |     |     |     |     |     |     |     |     |     |     |     |     |     |
| 1997 | Basic Small |     |     |     |     |     |     |     |     |     |     |     |     |     |
|      | Lower Middle| -17 | -17 | -10 | -27 | -8  | -11 | -11 |     |     |     |     |     |     |
|      | Upper Middle| -1  |     |     |     |     |     |     |     |     |     |     |     |     |
|      | Sporty     |     |     |     |     |     |     |     |     |     |     |     |     |     |
|      | Luxury     |     |     |     |     |     |     |     |     |     |     |     |     |     |
| 1998 | Basic Small |     |     |     |     |     |     |     |     |     |     |     |     |     |
|      | Lower Middle| -18 | -18 | -17 | -24 | -8  | -18 | -18 |     |     |     |     |     |     |
|      | Upper Middle| -1  |     |     |     |     |     |     |     |     |     |     |     |     |
|      | Sporty     |     |     |     |     |     |     |     |     |     |     |     |     |     |
|      | Luxury     |     |     |     |     |     |     |     |     |     |     |     |     |     |

The Index = 1/10 the required tolerance percent increases to solve the segment.

A positive index indicates shares < 100%, i.e., the segment volume is too high.
A negative index indicates shares > 100%, i.e., the segment volume is too low.
A dot (.) indicates that the segment was solved within the given constraints.

Figure 7: Over-Constraint Index Report

570
@ECHO OFF
SAS D:\JOBS\STAMP.SAS -NODMS
COPY 4CSTALL.LOG+STAMP.LOG

echo Now Running SEG4CAST
SAS D:\JOBS\SEG4CAST.SAS -NODMS
COPY 4CSTALL.LOG+SEG4CAST.LOG

echo Now Running MFAXUPD
SAS D:\JOBS\MFAXUPD.SAS -NODMS
COPY 4CSTALL.LOG+MFAXUPD.LOG

echo Now Running CBS
SAS D:\JOBS\CSTCBS.SAS -NODMS
COPY 4CSTALL.LOG+CSTCBS.LOG

echo Now Running CLM
SAS D:\JOBS\CSTCLM.SAS -NODMS
COPY 4CSTALL.LOG+CSTCLM.LOG


echo Now Running TFV
SAS D:\JOBS\CSTTFV.SAS -NODMS
COPY 4CSTALL.LOG+CSTTFV.LOG

echo Now Running SUMRPTS
SAS D:\JOBS\SUMRPTS.SAS -NODMS
COPY 4CSTALL.LOG+SUMRPTS.LOG

echo Now Running SUB4CST
SAS D:\JOBS\SUB4CST.SAS -NODMS
COPY 4CSTALL.LOG+SUB4CST.LOG

SAS D:\JOBS\STAMP.SAS -NODMS
COPY 4CSTALL.LOG+STAMP.LOG
COPY 4CSTALL.LOG D:\LOG
DEL * .LOG
COPY D:\LOG\NULL .LOG 4CSTALL .LOG
QPrintP
EXIT

Figure 5 OS/2 Batch File to Run Entire Forecast System