Paper SAS2981-2019

# Optimizing Product Assortment with Total Unduplicated Reach and Frequency Analysis in SAS/OR®

Jay Laramore, SAS Institute

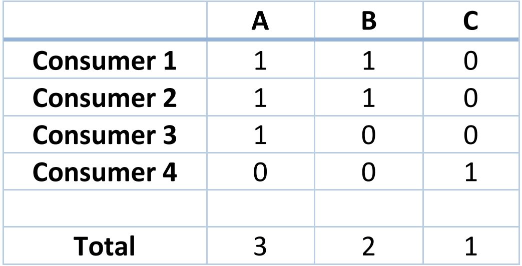
## ABSTRACT

Within the market research industry, Total Unduplicated Reach and Frequency (TURF) analysis has become an increasingly popular technique used to determine which combination of products will appeal to the greatest number of consumers. For companies that rely on optimal product assortment to help drive profitability, the output from a well-designed TURF analysis is critical for understanding product cannibalization and for evaluating the tradeoffs associated with adding or removing specific products from a product line. Conventional approaches for TURF analyses have involved calculating all possible product combinations, only to then recommend the one optimal, or the few near-optimal, solutions. This exhaustive approach is computationally inefficient and does not scale to commercial sized problems, which can often involve dozens of products, thousands of consumers, and tens of millions of product combinations. Other approaches using a greedy heuristic fail to guarantee an optimal solution. A more accurate and commercially viable approach to TURF analysis can instead be constructed as a mixed-integer linear programming (MILP) problem using SAS/OR® software. This paper details the modeling approach, data requirements, desired output, scenario analysis, and stationarity considerations. A detailed example along with sample code using SAS/IML®, SAS/OR®, and SAS/STAT® is provided. This paper introduces both the business problem and the analytical solution, so anyone with a background in retail analytics or market research can use this approach.

## INTRODUCTION

TURF analysis is a technique used in the food, beverage, retail, and marketing industries to optimize product assortment by finding the combination of products that will reach the most consumers. Miaoulis, Free, and Parsons (1990) first proposed the analysis by applying the technique to the design of communication plans. However, when formulated as a mixed-integer linear programming problem, TURF has its equivalent in facility location analysis, more generally referred to as the Maximal Covering Location Problem (Church and ReVelle, 1974). The general form of this problem makes it applicable to a wide range of products and scenarios.

Consider the following example. A company is testing three brand new product varieties (A, B, and C), with plans to launch only two varieties to the market. Due to the absence of historical transactional data, the company has surveyed and recorded the responses from four potential consumers who have indicated whether they would purchase each product (1=yes, 0=no). The objective of this TURF analysis is to find the two product varieties that will appeal to, or reach, the most consumers.



| | A | B | C |
|---|---|---|---|
| **Consumer 1** | 1 | 1 | 0 |
| **Consumer 2** | 1 | 1 | 0 |
| **Consumer 3** | 1 | 0 | 0 |
| **Consumer 4** | 0 | 0 | 1 |
| | | | |
| **Total** | 3 | 2 | 1 |

**Figure 1. Introductory Example**

If the goal was to maximize the reach of each product individually, the company would launch products A and B since the sum of their combined individual reach, 5, is larger than any other two-product combination. However, the goal of TURF analysis is to maximize the unduplicated reach across the entire product line, not the sum of each product's individual reach. Every consumer surveyed would purchase either product A or C, yielding an unduplicated reach of 4, or 100% of those surveyed. Since product combinations B and C and A and B only produce an unduplicated reach of 3, A and C is the optimal two-product combination.

A more intuitive way to visualize the concept of unduplicated reach is through the Venn diagram shown in Figure 2. The three circles represent the three new product varieties being tested (A, B, and C). The numbers represent each consumer's unique identifier from the table in Figure 1, and the location of each consumer within the circle(s) represents their preferences among the different product varieties.

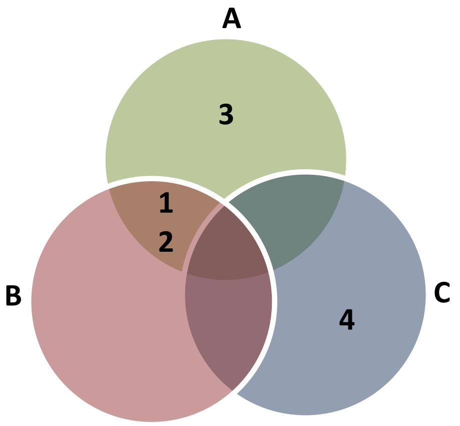


**Figure 2. Introductory Example Using a Venn Diagram**

If product B is removed from the lineup, every consumer would still prefer at least one of the two remaining product varieties. However, removing either product A or C would result in failing to reach one consumer. Thus, the optimal two-product combination that maximizes unduplicated reach is A and C.

Given the number of products in a set being tested (**n**) and the number of products from the set chosen to launch to the market (**v**), the combination formula below calculates the number of product combinations.

$$\frac{n!}{v!\,(n-v)!}$$

For example, a commercial-sized problem might include thousands of consumers surveyed, twenty-six products being tested, and plans to launch the optimal thirteen-product combination from the twenty-six-product set that appeals to the most consumers. This scenario produces 10,400,600 thirteen-product combinations. Increasing the number of products in the set by seven, from twenty-six to thirty-five, increases the number of thirteen-product combinations to more than 1.4 *billion*. Calculating every single combination is at best an inefficient and time-consuming approach, and at worst an exercise in futility.

The following provides a function for calculating the number of product combinations given a user-defined set in SAS/IML.

```
proc iml;
 start Combinations(n); /*begin function creation*/
  tot=j(n,1,.);
   do v=1 to nrow(tot) by 1;
    tot[v]=fact(n)/(fact(v)*fact(n-v));
   end;
```

```
     return(t(1:nrow(tot))||tot);
  finish; /*end function creation*/

  product_combos=Combinations(26); /*use function with user-defined set*/
  print product_combos[format=comma20.]; /*print results*/
 quit;
```

The benefits of using optimization in lieu of exhaustive enumeration (i.e. grid search) is widely recognized. Thus, the remainder of this paper focuses on building and applying the mixed-integer linear programming model in the context of TURF analysis to answer critical business questions and provide insight to those tasked with making key decisions regarding product assortment.

## BUSINESS OBJECTIVES AND INSIGHTS

Generally, the objective of a TURF analysis is not to recommend one optimal solution, but to instead recommend many optimal solutions under various scenarios. For example, given a set of twenty-six product varieties under consideration, it's often of interest to compare the additional reach gained from the optimal *v+1*-product bundle versus the optimal *v*-product bundle to assess whether the marginal reach gained justifies the marginal cost associated with launching one more variety to the market.

As the number of products in the optimal product bundle increases, unduplicated reach will generally increase by a smaller amount each time, ultimately converging to 100% when the number of products in the optimal product bundle equals the number of products in the set. This diminishing marginal returns relationship is expected when conducting a TURF analysis, although the percentages and speed of convergence will vary depending on the probabilities and correlation of the input data (i.e. survey responses). Originally named the *cost-effectiveness curve* (Church and ReVelle, 1974), the graph below shows the results from a simulated example of 15,000 surveyed consumers from a set of twenty-six products under consideration.

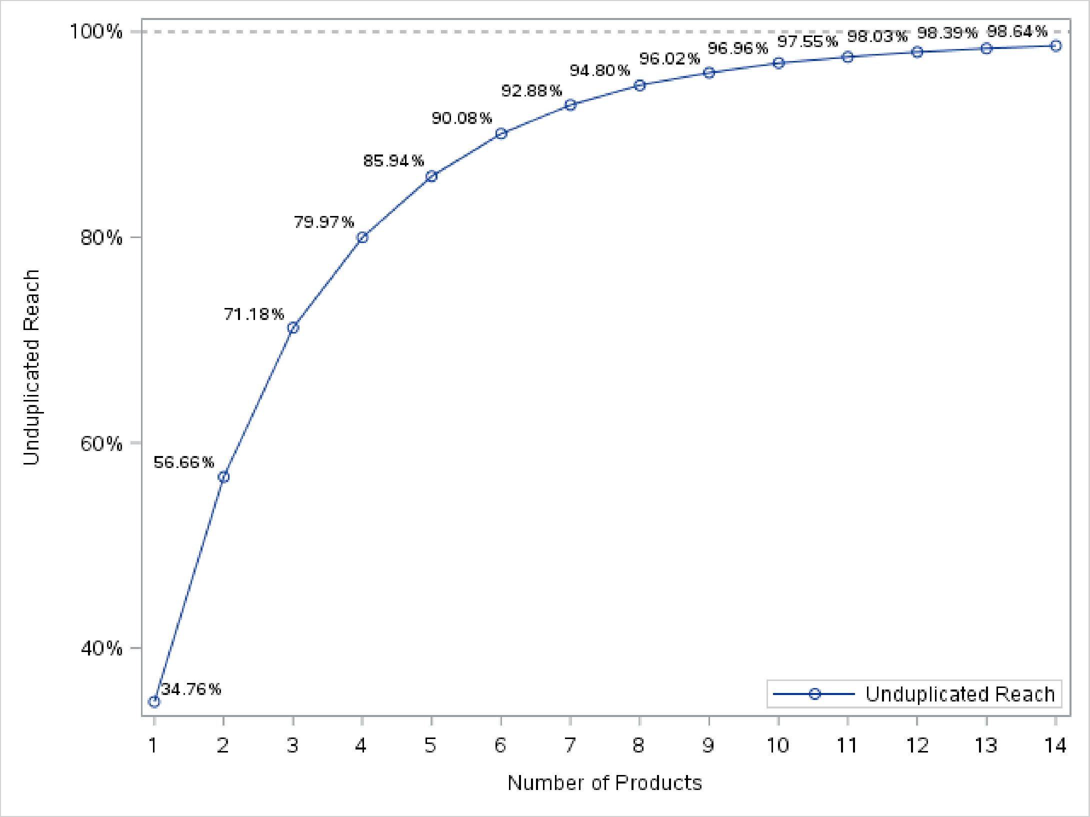


**Figure 3. Cost-Effectiveness Curve**

Along with a list indicating which products are included in each of the fourteen optimal product bundles, numerous comparisons can be drawn from the cost-effectiveness curve in Figure 3 to determine which optimal product bundle and number of product varieties align closest with the strategic and business objectives of the company.

For example, nine out of ten surveyed consumers would purchase at least one of the six product varieties in the optimal six-product bundle. Doubling the number of products from six to twelve, however, increases unduplicated reach by only about eight percent.

Producing the cost-effectiveness curve requires running the TURF analysis algorithm numerous times to evaluate the impact and tradeoffs among optimal product bundles with varying product amounts. This requires a computationally efficient, flexible, and robust algorithm for producing these insights in a timely manner.

## TURF EXAMPLE

You're throwing a Gatsbyesque dinner party at your home for 20,000 of your closest friends. Assume that all 20,000 people accept your invitation to attend, due to, of course, your prominent status in the community, distinguished sense of style, good-natured disposition, and generally quick-witted sense of humor.

On the party invitations, each attendee is asked to choose from up to ten different types of wine they'd prefer to be offered at the party. Based on their individual wine preferences, each attendee can choose to select zero wines, all ten wines, or any number of wine varieties in between.

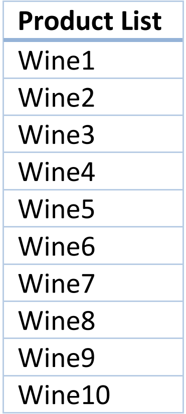
You're initially planning to serve four wine varieties but are uncertain which four-wine combination will appeal to the greatest number of partygoers. It's also unclear *how many* partygoers will prefer at least one of the four varieties in the optimal four-wine combination, so you want to create a cost-effectiveness curve to understand how much additional reach will be gained by including more varieties in the optimal product bundle.

The remaining sections show how to construct a TURF analysis using binary integer linear programming to provide clarity around an otherwise intimidating product assortment problem.

## DATA REQUIREMENTS

Two input data sets are required to run the TURF analysis algorithm in SAS/OR®. An example of each is provided below.

The first data set, PRODUCT_LIST, is a one-column list of distinct product names. In the dinner party example, these are the ten different wine varieties.



| Product List |
|---|
| Wine1 |
| Wine2 |
| Wine3 |
| Wine4 |
| Wine5 |
| Wine6 |
| Wine7 |
| Wine8 |
| Wine9 |
| Wine10 |

**Figure 4. PRODUCT_LIST Data Set**

The second data set, TURF_RAW, contains unique consumer IDs, consumer weights, and binary survey responses for each product under consideration (1=would purchase/consume product j, 0=would not purchase/consume product j). Survey weights can be calculated to weight the responses of some consumers higher than others based on their responses to other survey questions and/or past consumption behavior (for example, recency, frequency, and monetary statistics; Customer Lifetime Value scores; demographic information; geographic location; and so on) that distinguishes purchasing intent.

For example, out of the 20,000 dinner party attendees, the wine preferences of some attendees might carry more influence, or weight, than others. As the host of the party, it would be wise, for example, to weight the responses from your spouse higher than the responses from your crazy Uncle Charlie, who still hasn't returned your pressure washer that he borrowed last summer. In this instance, weights would not be calculated based on purchasing intent, but rather on strategic, political, or personal motivations that would otherwise be left unaccounted for in the model. If all surveyed consumers are weighed equally however, each receives a value of 1 in this column. This paper does not address the technical considerations of survey weighting, but will instead show where survey weights factor into the construction of the optimization model.

| Consumer ID | Consumer Weight | Wine1 | Wine2 | … | Wine10 |
|---|---|---|---|---|---|
| 1 | 1 | 1 | 0 | … | 0 |
| 2 | 1 | 1 | 1 | … | 0 |
| 3 | 1 | 0 | 1 | … | 1 |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| 20000 | 1 | 0 | 1 | … | 0 |

**Figure 5. TURF_RAW Data Set**

In the dinner party example, as well as in commercial retail, beverage, and food service applications, data are commonly gathered through surveys. To demonstrate the TURF algorithm in SAS/OR, the survey responses for the 20,000 dinner party attendees are simulated.

To simulate consumer responses for each type of wine, the methodology proposed in Emrich and Piedmonte (1991) is used to simulate correlated multivariate binary variables. Given a probability vector and correlation matrix for the products under consideration, the RandMVBinary() function in SAS/IML® simulates the survey responses. The result is a 20,000 x 10 matrix of correlated binary survey responses.

The following provides the probability vector and correlation matrix used to simulate the survey response data.

| Wine1 | Wine2 | Wine3 | Wine4 | Wine5 | Wine6 | Wine7 | Wine8 | Wine9 | Wine10 |
|---|---|---|---|---|---|---|---|---|---|
| 30% | 27% | 31% | 37% | 35% | 45% | 36% | 31% | 25% | 20% |

**Figure 6. Probability Vector**

| | Wine1 | Wine2 | Wine3 | Wine4 | Wine5 | Wine6 | Wine7 | Wine8 | Wine9 | Wine10 |
|---|---|---|---|---|---|---|---|---|---|---|
| **Wine1** | 1 | 0.254 | -0.021 | -0.05 | -0.078 | -0.049 | 0.091 | 0.219 | -0.046 | 0.109 |
| **Wine2** | 0.254 | 1 | -0.019 | -0.034 | -0.098 | -0.039 | 0.098 | 0.105 | -0.063 | 0.212 |
| **Wine3** | -0.021 | -0.019 | 1 | 0.145 | 0.196 | 0.158 | 0.10 | 0.041 | 0.076 | 0.028 |
| **Wine4** | -0.05 | -0.034 | 0.145 | 1 | 0.121 | 0.554 | 0.201 | -0.036 | 0.135 | 0.039 |
| **Wine5** | -0.078 | -0.098 | 0.196 | 0.121 | 1 | 0.255 | 0.085 | -0.047 | 0.153 | -0.010 |

|  | Wine1 | Wine2 | Wine3 | Wine4 | Wine5 | Wine6 | Wine7 | Wine8 | Wine9 | Wine10 |
|---|---|---|---|---|---|---|---|---|---|---|
| Wine6 | -0.049 | -0.039 | 0.158 | 0.554 | 0.255 | 1 | 0.11 | -0.031 | 0.082 | 0.065 |
| Wine7 | 0.091 | 0.098 | 0.10 | 0.201 | 0.085 | 0.11 | 1 | 0.071 | 0.070 | 0.191 |
| Wine8 | 0.219 | 0.105 | 0.041 | -0.036 | -0.047 | -0.031 | 0.071 | 1 | -0.015 | 0.070 |
| Wine9 | -0.046 | -0.063 | 0.076 | 0.135 | 0.153 | 0.082 | 0.070 | -0.015 | 1 | 0.005 |
| Wine10 | 0.109 | 0.212 | 0.028 | 0.039 | -0.010 | 0.065 | 0.191 | 0.070 | 0.005 | 1 |

**Figure 7. Correlation Matrix**

The following provides the SAS/IML® code to simulate the 20,000 survey responses. To include the RandMVBinary.sas program into your SAS® session, follow the instructions in the link below: https://communities.sas.com/t5/SAS-IML-File-Exchange/Simulate-Correlated-Multivariate-Binary-Variables/ta-p/221225.

```
%include "<insert-path>\RandMVBinary.sas";
proc iml;
 nprod=10; /* Number of products */
 nsurveyed=20000; /* Number of surveyed consumers */
 product_prefix='Wine'; /* Generic product prefix */

 ConsID=t(1:nsurveyed);/* Vector of unique consumer IDs */
 Weight=j(nsurveyed,1); /* Vector of weights = 1 */
 product_names=product_prefix+strip(char(1:nprod)); /*Unique prod names*/

 load module=_all_;

 call randseed(27278); /* Seed for reproducible results*/

 p = {.30 .27 .31 .37 .35 .45 .36 .31 .25 .20}; /*Probability vector*/
 R=j(ncol(p),ncol(p),1); /*Create and populate correlation matrix*/

 R[1,2]= 0.254;  R[2,1]= 0.254; R[2,3]=-0.019;  R[3,2]=-0.019;
 R[1,3]=-0.021;  R[3,1]=-0.021; R[2,4]=-0.034;  R[4,2]=-0.034;
 R[1,4]= -0.05;  R[4,1]= -0.05; R[2,5]=-0.098;  R[5,2]=-0.098;
 R[1,5]=-0.078;  R[5,1]=-0.078; R[2,6]=-0.039;  R[6,2]=-0.039;
 R[1,6]=-0.049;  R[6,1]=-0.049; R[2,7]= 0.098;  R[7,2]= 0.098;
 R[1,7]= 0.091;  R[7,1]= 0.091; R[2,8]= 0.105;  R[8,2]= 0.105;
 R[1,8]= 0.219;  R[8,1]= 0.219; R[2,9]=-0.063;  R[9,2]=-0.063;
 R[1,9]=-0.046;  R[9,1]=-0.046; R[2,10]=0.212;  R[10,2]=0.212;
 R[1,10]=0.109;  R[10,1]=0.109;

 R[3,4]= 0.145;  R[4,3]= 0.145; R[4,5]= 0.121;  R[5,4]= 0.121;
 R[3,5]= 0.196;  R[5,3]= 0.196; R[4,6]= 0.554;  R[6,4]= 0.554;
 R[3,6]= 0.158;  R[6,3]= 0.158; R[4,7]= 0.201;  R[7,4]= 0.201;
 R[3,7]= 0.100;  R[7,3]= 0.100; R[4,8]=-0.036;  R[8,4]=-0.036;
 R[3,8]= 0.041;  R[8,3]= 0.041; R[4,9]= 0.135;  R[9,4]= 0.135;
 R[3,9]= 0.076;  R[9,3]= 0.076; R[4,10]=0.039;  R[10,4]=0.039;
 R[3,10]=0.028;  R[10,3]=0.028;

 R[5,6]= 0.255;  R[6,5]= 0.255; R[6,7]= 0.110;  R[7,6]= 0.110;
 R[5,7]= 0.085;  R[7,5]= 0.085; R[6,8]=-0.031;  R[8,6]=-0.031;
 R[5,8]=-0.047;  R[8,5]=-0.047; R[6,9]= 0.082;  R[9,6]= 0.082;
 R[5,9]= 0.153;  R[9,5]= 0.153; R[6,10]=0.065;  R[10,6]=0.065;
 R[5,10]=-0.01;  R[10,5]=-0.01;
```

```
  R[7,8]= 0.071;  R[8,7]= 0.071; R[8,9]=-0.015;  R[9,8]=-0.015;
  R[7,9]= 0.070;  R[9,7]= 0.070; R[8,10]=0.070;  R[10,8]=0.070;
  R[7,10]=0.191;  R[10,7]=0.191;

  R[9,10]=0.005;  R[10,9]=0.005;

  X = RandMVBinary(nsurveyed,p,R); /*Simulate survey responses*/

  colnames='ConsumerNo'||'ConsumerWeight'||product_names;
  Z=ConsID||Weight||X;

  /*Create required data sets*/
  create work.turf_raw from Z[colname=colnames];
  append from Z;
  close work.turf_raw;

  product_names=t(product_names);

  create work.product_list from product_names[colname="ProductList"];
  append from product_names;
  close work.product_list;
quit;
```

Out of the 20,000 dinner party attendees, it's likely that some will not select any of the ten wines because they either do not drink wine or do not prefer any of the varieties in the ten-product set. Since no combination will appeal to these attendees, they're removed from further analysis. Figure 8 below counts the number of attendees and groups them by the total number of wines chosen.

| No. Wines Chosen | No. Dinner Party Attendees |
|---|---|
| 0 | 1,401 |
| 1 | 2,726 |
| 2 | 3,646 |
| 3 | 4,044 |
| 4 | 3,400 |
| 5 | 2,399 |
| 6 | 1,410 |
| 7 | 669 |
| 8 | 226 |
| 9 | 64 |
| 10 | 15 |

**Figure 8. Attendee Count by Number of Wines Selected**

Thus, the TURF analysis model focuses solely on the remaining 18,599 dinner party attendees who selected at least one type of wine. The following code removes the 1,401 attendees who selected zero wine varieties.

```
data work.turf_raw;
 set work.turf_raw;
 if sum(of Wine1-Wine10) gt 0;
run;
```

The PRODUCT_LIST and TURF_RAW data sets are now sufficiently prepared to be read into the OPTMODEL procedure.

## TURF MODEL DEVELOPMENT

Repurposed from the original Maximal Covering Location Problem (Church and ReVelle, 1974), the model maximizes the number of unduplicated consumers who prefer at least one product from the optimal product bundle by first constructing two sets of binary decision variables: one set for each product under consideration ($x_j$), and one set for each consumer in the survey ($y_i$).

For each product (e.g. Wine1 – Wine 10), Product j is part of the optimal product bundle: Yes=1 or No=0.

$$x_j = (0,1), \quad j = 1, \dots, J$$

For each surveyed consumer (e.g. 1 – 18,599), Consumer i will purchase at least one product from the optimal product bundle: Yes=1 or No=0.

$$y_i = (0,1), \quad i = 1, \dots, I$$

The objective function maximizes unduplicated reach, where Z is unduplicated reach, and $f_i$ is the consumer weight.

$$Max\ Z = \sum_{i=1}^{I} f_i\, y_i$$

When the algorithm decides that a product is part of the optimal product bundle ($x_j$ =1), all consumers who indicated from their survey responses ($N_i$) that they would purchase that product are considered buyers and their $y_i$ equals 1. The problem seeks to maximize $y_i$, so if $y_i$ can satisfy the constraint below as a 0 or a 1, it will always choose 1. However, when the algorithm determines that none of the products chosen by a consumer are part of the optimal product bundle ($\sum x_j$ =0), that consumer is not considered a buyer and their $y_i$ is forced to equal 0. This is the first of two mandatory constraints.

$$s.t. \quad \sum_{j \in N_i} x_j - y_i \geq 0, \quad i = 1, \dots, I$$

The second mandatory constraint limits the number of total products in the optimal product bundle. For example, out of the ten wine varieties, if the goal is to find the optimal four-product combination that maximizes unduplicated reach, V=4. To create the cost-effectiveness curve, this algorithm is run multiple times, replacing the value of V by one each time and holding everything else constant.

$$\sum_{j=1}^{J} x_j = V$$

Formulated as a mixed integer linear programming problem, TURF analysis can be easily extended to accommodate and control for other important variables such as the production cost of each product, time required to produce each product, total capacity constraints, budget constraints, and so on for a more comprehensive analysis.

In addition, new constraints can be readily added to force specific products into or out of the optimal product bundle. These constraints are common when extending an existing product line, since a certain number of products in the set are currently on the market and plan to remain on the market, so the business objective has shifted from determining which combination of products maximizes unduplicated reach, to which combination of new products *alongside the products already on the market* maximizes unduplicated reach. In this case, constraints can be added to force each product already on the market into the optimal product bundle.

## TURF MODEL DEVELOPMENT IN SAS/OR®

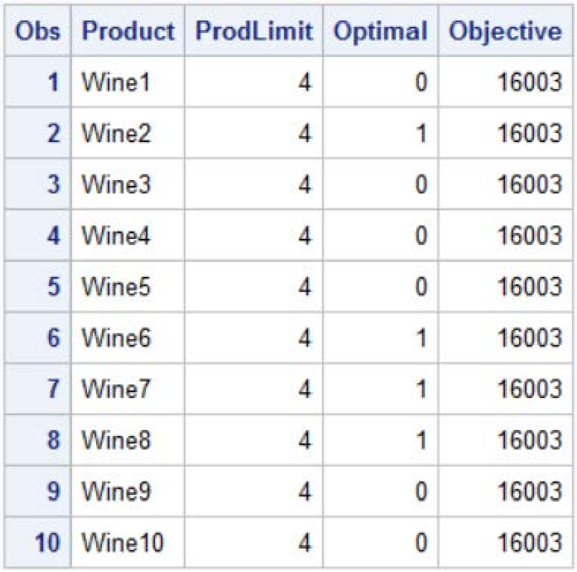
The OPTMODEL procedure in SAS/OR® is a flexible optimization programming language allowing users to build and customize optimization models of varying scale and complexity. TURF analysis, formulated as a mixed integer linear programming problem, can be built in fewer than twenty lines of code and executed to completion in under a few seconds.

```
%let V=4;
proc optmodel;
 set <num> CONSUMERS;
 set <str> PRODUCTS;

 num responses{CONSUMERS,PRODUCTS};
 num ConsumerWeight{CONSUMERS};

 read data work.product_list into PRODUCTS=[ProductList];
 read data work.turf_raw into CONSUMERS=[ConsumerNo] ConsumerWeight
   {j in PRODUCTS} <responses[ConsumerNo,j]=col(j)>;

 var x{PRODUCTS} binary;
 var y{CONSUMERS} binary;

 impvar UnduplicatedReach=sum{i in CONSUMERS} ConsumerWeight[i]*y[i];

 max Z = UnduplicatedReach;

 con UnduplicatedConsumers{i in CONSUMERS}:
    sum {j in PRODUCTS: responses[i,j]=1} x[j]-y[i]>=0;

 con ProductLimit: sum{j in PRODUCTS} x[j] = &V.;

 solve with MILP;

 create data work.product_results_optimal&V.
 from [Product]={j in PRODUCTS} ProdLimit=&V. Optimal=x
       Objective=UnduplicatedReach;

 create data work.consumer_results_optimal&V.
 from [ConsumerNo]={i in CONSUMERS} Purchase=y;
quit;
```

## RESULTS

The optimal four-product bundle contains Wine2, Wine6, Wine7, and Wine8 and reaches approximately 86% of the dinner party attendees who selected at least one type of wine. No other four-product combination exceeds this value.

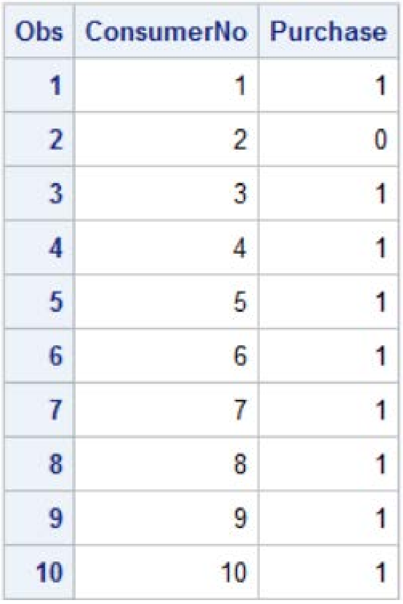
In the PRODUCT_RESULTS_OPTIMAL4 output data set, each product, along with the product limit (V), optimal product bundle binary indicator ($x_j$), and objective value (Z) identify key pieces of output from the TURF optimization model. Appending subsequent data sets each time as V increments by one provides sufficient information to create the cost-effectiveness curve.



| Obs | Product | ProdLimit | Optimal | Objective |
|---|---|---|---|---|
| 1 | Wine1 | 4 | 0 | 16003 |
| 2 | Wine2 | 4 | 1 | 16003 |
| 3 | Wine3 | 4 | 0 | 16003 |
| 4 | Wine4 | 4 | 0 | 16003 |
| 5 | Wine5 | 4 | 0 | 16003 |
| 6 | Wine6 | 4 | 1 | 16003 |
| 7 | Wine7 | 4 | 1 | 16003 |
| 8 | Wine8 | 4 | 1 | 16003 |
| 9 | Wine9 | 4 | 0 | 16003 |
| 10 | Wine10 | 4 | 0 | 16003 |

**Output 1. PRODUCT_RESULTS_OPTIMAL4 Data Set**

In the CONSUMER_RESULTS_OPTIMAL4 output data set, each unique consumer ID is listed, along with the Purchase column ($y_i$) indicating whether each attendee would purchase (or in this case, consume) at least one of the products in the optimal product bundle.



| Obs | ConsumerNo | Purchase |
|---|---|---|
| 1 | 1 | 1 |
| 2 | 2 | 0 |
| 3 | 3 | 1 |
| 4 | 4 | 1 |
| 5 | 5 | 1 |
| 6 | 6 | 1 |
| 7 | 7 | 1 |
| 8 | 8 | 1 |
| 9 | 9 | 1 |
| 10 | 10 | 1 |

**Output 2. CONSUMER_RESULTS_OPTIMAL4 Data Set (Partial)**

To create the cost-effectiveness curve for the dinner party example, the OPTMODEL program needs to be executed ten separate times, replacing the value of V each time, so that the ten different product result data sets (e.g. PRODUCT_RESULTS_OPTIMAL1 – PRODUCT_RESULTS_OPTIMAL10) are created.

From there, the following code appends them into one master data set, calculates unduplicated reach as a percentage, and constructs the cost-effectiveness curve.

```
data work.product_results_master;
 set work.product_results_optimal1 - work.product_results_optimal10;
run;
proc sql;
 create table work.graphdata as
 select distinct prodlimit
 ,             objective
 ,             objective/18599 format=percent8.2 as UnduplicatedReach
 from work.product_results_master;
quit;

proc sgplot data=work.graphdata;
 series x=prodlimit y=UnduplicatedReach/ datalabel=UnduplicatedReach
 legendlabel="Optimal Unduplicated Reach" lineattrs=(thickness=1) markers;
 xaxis label="Number of Wine Varieties" type=discrete;
 yaxis label="Unduplicated Reach" max=1;
 keylegend / location=inside position=bottomright;
 refline 1 / axis=y lineattrs=(color=gray pattern=shortdash thickness=1)
 transparency=0.50;
run;
```

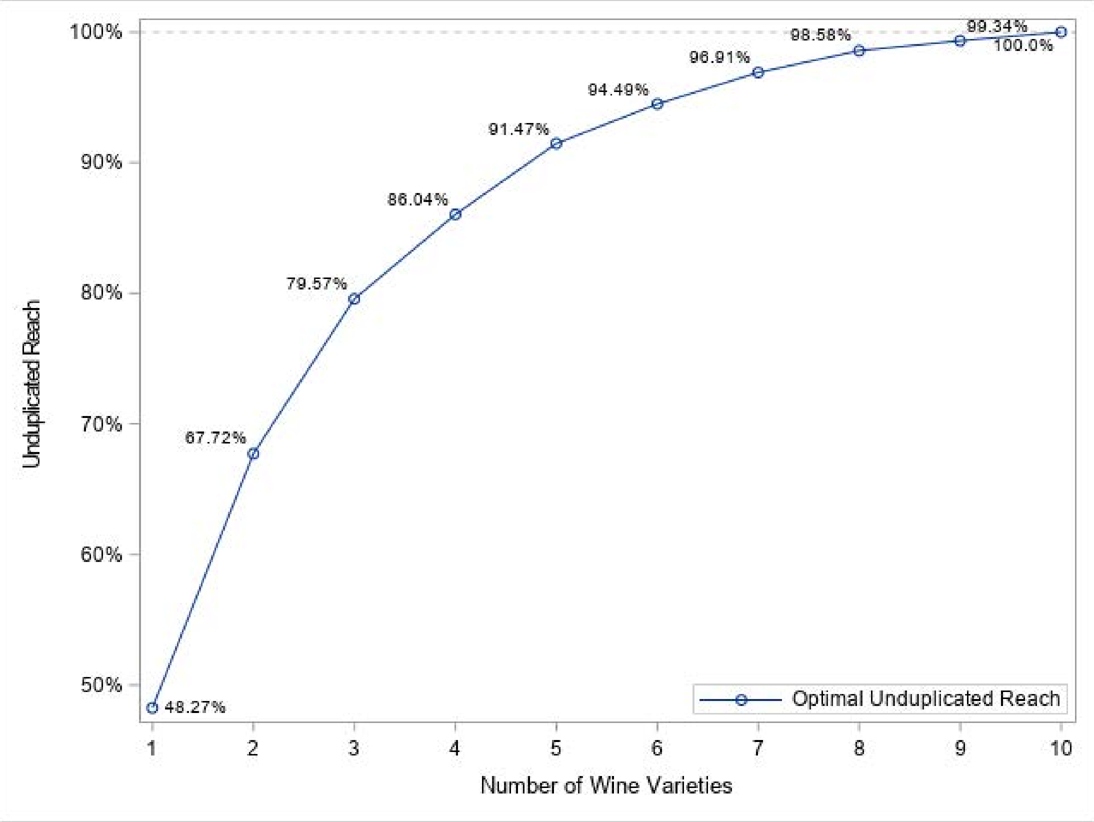


**Figure 9. Cost-Effectiveness Curve – Wine Example**

The following table lists information from the PRODUCT_RESULTS_MASTER data set to show the optimal product bundles that are displayed in the cost-effectiveness curve above.

| No. Wine Varieties | Optimal Product Bundle | Optimal Unduplicated Reach | Optimal Unduplicated Reach % |
|---|---|---|---|
| **1** | Wine 6 | 8,977 | 48.27% |
| **2** | Wines 6, 8 | 12,596 | 67.72% |
| **3** | Wines 6, 7, 8 | 14,800 | 79.57% |
| **4** | Wines 2, 6, 7, 8 | 16,003 | 86.04% |
| **5** | Wines 2, 5, 6, 7, 8 | 17,013 | 91.47% |
| **6** | Wines 2, 5, 6, 7, 8, 9 | 17,575 | 94.49% |
| **7** | Wines 1, 2, 5, 6, 7, 8, 9 | 18,025 | 96.91% |
| **8** | Wines 1, 2, 3, 5, 6, 7, 8, 9 | 18,335 | 98.58% |
| **9** | Wines 1, 2, 3, 5, 6, 7, 8, 9, 10 | 18,476 | 99.34% |
| **10** | Wines 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 | 18,599 | 100.0% |

**Figure 10. Optimal Product Bundles**

As the host of the party, you now have the information needed to compare the reach across every optimal v-product combination to determine the number of varieties to serve at the party.

For example, should you decide to serve five wine varieties instead of four, you'll reach approximately 1,000 more dinner party attendees and 91.47% of all attendees who prefer at least one type of wine from the ten-wine set.

The following section assumes you've decided to serve the optimal four-wine combination, although in practice, any optimal v-wine combination can be used for the upcoming scenario analysis.

## SCENARIO ANALYSIS

The types of scenario analyses that can be performed within a TURF analysis are effectively limitless and bound only by the scope of the project and one's imagination. This section introduces one type of scenario analysis that is widely applicable across many TURF analyses, especially when the surveyed participants represent a sample of the total population of potential consumers.

The objective is to test the robustness of the optimal four-product bundle with other "challengers", or sub-optimal four-product bundles, to determine whether the difference in unduplicated reach is significantly different. In general, the two product bundles being compared are the following:

1. The optimal v-product bundle (e.g. Wine 2, Wine 6, Wine 7, Wine 8; v=4)
2. A sub-optimal v-product bundle where one or more products have been forced into or out of the optimal bundle

For example, suppose you want to compare the difference in unduplicated reach when Wine 9 is forced into the four-product bundle ($x_9$=1). This requires modifying the original optimization model by adding an additional constraint, renaming the output data sets, ensuring that the macro variable V is equal to four, and re-running the model.

```
%let V=4;

con ForceIn: x['Wine9']=1;
```

```
   create data work.product_results_suboptimal&V.

   create data work.consumer_results_suboptimal&V.
```

Figure 11 compares the results of the two product bundles. For the sub-optimal product bundle, Wine9 replaced Wine2 and unduplicated reach dropped by 0.83%, effectively reaching 154 fewer attendees. Keep in mind that just because the other three wines (6, 7, and 8) are in the optimal product bundle doesn't guarantee their inclusion into the sub-optimal bundle when another product is forced in, but they happen to be in this case.

| Product Bundle | Products | Unduplicated Reach | Unduplicated Reach % |
|---|---|---|---|
| **Optimal** | Wines 2, 6, 7, 8 | 16,003 | 86.04% |
| **Sub-optimal** | Wines 6, 7, 8, 9 | 15,849 | 85.21% |

**Figure 11. Optimal and Sub-Optimal Bundle Comparison**

Using the two consumer output data sets from the different model runs, a 2x2 table is constructed to identify marginal proportions and discordant pairs.

```
  proc sql;
   create table work.scenario_analysis as
   select     a.ConsumerNo
   ,          a.Purchase as Optimal
   ,          b.Purchase as Suboptimal
   from work.consumer_results_optimal4 as a
   left join work.consumer_results_suboptimal4 as b
    on a.ConsumerNo=b.ConsumerNo
   order by a.ConsumerNo;
  quit;

  proc freq data=work.scenario_analysis;
   tables Optimal*Suboptimal /agree nocol norow;
  run;
```

|  |  | **(Sub-optimal "Challenger")<br>Wines 6, 7, 8, 9** |  |  |
|---|---|---|---|---|
|  | Consume | 0 | 1 | Total |
| **(Optimal)<br>Wines 2, 6, 7, 8** | 0 | 1,730<br>(9.30%) | 866<br>(4.66%) | 2,596<br>(13.96%) |
|  | 1 | 1,020<br>(5.48%) | 14,983<br>(80.56%) | 16,003<br>(86.04%) |
|  | Total | 2,750<br>(14.79%) | 15,849<br>(85.21%) | 18,599<br>(100.00%) |

**Output 3. Model Comparison 2x2 Contingency Table**

Output 3, created from the FREQ procedure, is used to test the difference between marginal proportions using McNemar's test. McNemar's test is appropriate when data are being analyzed from repeated measures with a binary response. In the dinner party example, McNemar's test is being used to test whether the difference in unduplicated reach between the two product bundles (0.83%) is significantly different, or statistically equivalent.

The marginal proportions are 2,596/18,599 (13.96%) and 2,750/18,599 (14.79%). In other words, 13.96%, or $P_A$, of attendees will not consume from the optimal four-product bundle, and 14.79%, or $P_B$, will not consume from the sub-optimal four-product bundle. Included in both $P_A$ and $P_B$ are the number of attendees who wouldn't consume from either four-product bundle, 1,730 (9.30%), so the difference between $P_A$ and $P_B$ boils down to the difference between the two discordant cells. The null hypothesis of the test is marginal homogeneity.

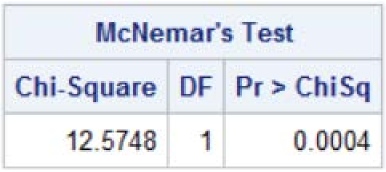
$$H_0 : P_A = P_B$$

$$H_A : P_A \neq P_B$$

Given enough discordant observations, $\chi^2$ has a chi-squared distribution with 1 degree of freedom.

$$\chi^2 = \frac{(1020 - 866)^2}{1020 + 866}$$

The AGREE option in PROC FREQ performs McNemar's test.



| McNemar's Test | | |
|---|---|---|
| Chi-Square | DF | Pr > ChiSq |
| 12.5748 | 1 | 0.0004 |

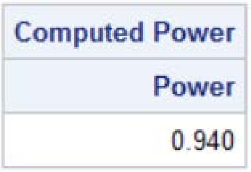
**Output 4. McNemar's Test Results**

The results provide sufficient evidence to reject the null hypothesis and conclude the marginal proportions are significantly different from one another, and thus, the sub-optimal product bundle is not an adequate substitute for the optimal product bundle.

In addition, the POWER procedure can be used to determine whether the test is well powered for detecting true effects. The discordant proportions and number of pairs can be filled in directly from the 2x2 table.

```
proc power;
 pairedfreq dist=normal method=connor
 test=mcnemar
 discproportions = 0.0548 | 0.0466
 npairs = 18599
 power = .;
run;
```

Output 5 displays the results from PROC POWER and confirms the test is well powered with 18,599 subjects and alpha=0.05.

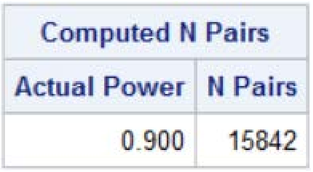


| Computed Power |
|---|
| Power |
| 0.940 |

**Output 5. Computed Power Results**

Given real-world time and budget considerations, a more interesting question at the beginning of the study is to determine the number of survey responses needed to ensure a power of at least a specific, user-defined threshold. The following example uses 0.90.

```
proc power;
 pairedfreq dist=normal method=connor
 test=mcnemar
 discproportions = 0.0548 | 0.0466
 npairs = .
 power =.90;
run;
```



| Computed N Pairs | |
|---|---|
| Actual Power | N Pairs |
| 0.900 | 15842 |

**Output 6. Sample Size Requirement**

For this comparative test, 15,842 subjects need to be surveyed for alpha to equal 0.05 and power to equal 0.90. This one example does not constitute an exhaustive power study, as different product bundle tests with varying discordant proportions will require different sample sizes. Thus, numerous tests of varying discordant proportions should be considered to ensure a sample size large enough so that all tests of interest are well powered.

Domain expertise, or input from someone with domain expertise, is critical in the planning phase for gauging discordant proportions in the absence of survey data that has yet to be collected. Technical considerations, such as alpha and power thresholds, are ultimately left up to the discretion of the researcher.

## STATIONARITY

The general purpose of a TURF analysis is to gain insight into consumer preferences, which leads to optimizing product assortment, and with enough relevant data on the surveyed participants, can be a catalyst for kick starting targeted marketing campaigns. While the underlying mathematical and statistical applications in a TURF analysis are deterministic, consumer preferences are constantly evolving, making the survey responses, and thus the results from a TURF analysis, less reliable over time.

Just like how predictive models need to be refreshed periodically, so too do survey responses, and perhaps even the product varieties in a TURF analysis. Just like how you wouldn’t trust a credit default model today that was built in 2008, you similarly wouldn’t trust survey responses on product varieties gathered during a time period that no longer reflects current market conditions.

## CONCLUSION

TURF analysis is an empirical approach used to gain insights into consumer preferences. Whether optimizing product assortment for your next dinner party, or helping a commercial food, beverage, or retail corporation successfully launch new product varieties to the market, the scalability and broad range of insights gained from a well-designed TURF analysis using SAS software can equip decision makers with key information to make strategic, data-driven decisions.

## REFERENCES

Church, R. and C. ReVelle. 1974. "The Maximal Covering Location Problem." *Papers of the Regional Science Association* 32:101-118. Available http://www.geog.ucsb.edu/~forest/G294download/MAX_COVER_RLC_CSR.pdf.

Emrich, L. J. and M. R. Piedmonte. 1991. "A Method for Generating High-Dimensional Multivariate Binary Variables." *The American Statistician* 45:4, 302-304. doi: 10.1080/00031305.1991.10475828

SAS Institute Inc. 2013. “Tests and Measures of Agreement.” In *Base SAS® 9.4 Procedures Guide: Statistical Procedures*. 2nd ed. Cary, NC: SAS Institute Inc. Available: http://support.sas.com/documentation/cdl/en/procstat/66703/HTML/default/viewer.htm#procstat_freq_details70.htm. (accessed June 16, 2018).

SAS Institute Inc. 2009. “PAIREDFREQ Statement.” In *SAS/STAT® 9.2 User’s Guide*, Second Edition. Cary, NC: SAS Institute Inc. Available: https://support.sas.com/documentation/cdl/en/statug/63033/HTML/default/viewer.htm#statug_power_sect009.htm. (accessed June 16, 2018).

Serra, D. 2013. “Implementing TURF Analysis Through Binary Linear Programming.” *Food Quality and Preference*, 28:1, 382-388.

Wicklin, Rick. “Simulate Correlated Multivariate Binary Variables.” Last modified June 5, 2014. Available https://communities.sas.com/t5/SAS-IML-File-Exchange/Simulate-Correlated-Multivariate-Binary-Variables/ta-p/221225?nobounce. Accessed on October 7, 2018.

## ACKNOWLEDGMENTS

Thank you to Aric LaBarr, Dan Kelly, Chip Wells, Shadi Hassani Goodarzi, and Stu Sztukowski for their technical and substantive review of this material. Their collective feedback made this final version better than it otherwise would have been.

## CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Jay Laramore
SAS Institute Inc.
+1 919-531-2810
Jay.Laramore@sas.com