

Appendix E

Macros for the Design and Analysis of Experiments

Appendix Table of Contents

OVERVIEW	1873
Software Requirements	1874
Structure Files for the Macro Collection	1874
Calling the Macros	1874
GENERAL MACROS: ADXGEN FILE	1876
ADXCODE: Code a Design for Analysis	1876
ADXDCODE: Decode a Design	1877
ADXINIT: Set Global Macro Variables	1878
ADXQMOD: Set Up a Second-Order Model	1879
ADXRPT: Randomize Design and Display a Data Collection Report	1880
ADXTRANS: Determine an Optimal Box-Cox Power Transformation	1881
FRACTIONAL FACTORIAL DESIGN MACROS: ADXFF FILE	1883
ADXALIAS: Compute the Alias Structure	1884
ADXFFA: Analyze Fractional Factorial Designs	1885
ADXFFD SAS Data Set	1886
ADXFFD: Construct Fractional Factorial Designs	1886
ADXPBD: Construct Plackett-Burman Designs	1887
ADXPFF: List Available Fractional Factorial Designs	1887
CENTRAL COMPOSITE DESIGN MACROS: ADXCC FILE	1889
ADXADCEN: Add Center Points	1889
ADXCCD: Construct Central Composite Designs	1890
ADXPCC: List Available Central Composite Designs	1891
MIXTURE DESIGN MACROS: ADMIX FILE	1892
ADXFILL: Filling in the Design Region	1892
ADMAMD: Construct McLean-Anderson Mixture Designs	1893
ADXSCD: Construct Simplex-Centroid Designs	1894
ADXSLD: Construct Simplex-Lattice Designs	1895
ADXXVERT: the XVERT Algorithm	1896
EXAMPLES	1897

Appendix E

Macros for the Design and Analysis of Experiments

Overview

The SAS System provides the tools to give you a complete software solution for constructing and analyzing experimental designs. Moreover, you can use these tools in a variety of ways. If you are comfortable with SAS programming, the FACTEX and OPTEX procedures, described in Part 3, “The FACTEX Procedure” and Part 6, “The OPTEX Procedure” are fundamental tools for constructing many types of experimental designs; and the analytic procedures of SAS/Stat[®] Software enable you to compute the statistical results of your experiments. Alternatively, you can use the point-and-click ease of the ADX Interface to construct and analyze designs for your experiments. The macros discussed in this appendix provide a third, intermediate alternative: you use them in SAS programming, but they bundle much of the syntax required for the general procedures into macro arguments. The macros draw on various SAS software tools:

- two-level factorial and fractional factorial designs for as many as 128 runs and 11 factors. This includes designs with and without blocking.
- two-level screening designs (Plackett-Burman designs) for as many as 47 factors
- orthogonal and rotatable central composite designs (Box-Wilson designs) for as many as 8 factors. This includes designs with and without blocking.
- mixture designs for either constrained or unconstrained components, with no limit on the number of factors. This includes simplex-centroid, simplex-lattice, and McLean-Anderson designs.

You can also use the macros to

- decode a design into units meaningful for your application
- randomize the design and display a data collection form
- perform a power-transformation analysis for the response variable
- analyze fractional factorial designs. The analysis includes a listing of the alias structure and a normal plot for the effects.

Note again that the macros in this collection are primarily intended to provide a *programming* interface to common experimental design tasks, as an alternative to using the underlying procedures directly. For users who are not experienced with SAS programming, the ADX Interface may be a more appropriate tool. The ADX Interface, which has been completely revised in Version 7, is designed primarily for engineers and researchers who require a point-and-click solution for the entire experimental process, from building the designs through determining significant effects to optimization and reporting. Information about the ADX Interface can be found at <http://www.sas.com/rnd/app/qc/newadx/newadx.html>. The ADX Interface is documented in *Getting Started with the ADX Interface for Design of Experiments*.

Software Requirements

This collection of SAS macros uses features in base SAS, SAS/STAT, and SAS/QC software, as shown in the following table:

SAS Software Module	Procedures Used
Base SAS	CONTENTS, CORR, DATASETS, FORMAT PLOT, PRINT, SORT, TRANSPOSE
SAS/STAT	REG
SAS/QC	FACTEX

To use all the features of the macros, you need to have these procedures installed.

Structure Files for the Macro Collection

The macros are contained in four files that cover the various types of designs. When you install SAS/QC software, these files are placed in the SASMACRO subdirectory. Figure E.1 summarizes the structure of the collection.

Calling the Macros

There are 19 macros, as shown in Figure E.1. The macros are organized into four files according to their function. To use the macros, you need to define them in each SAS session. Since these macro files are included in the AUTOCALL library you can include a file simply by typing the filename preceded by the % symbol, as shown below.

You only need to include a file once for each SAS session. You must include ADXGEN first since it contains general macros that may be called by macros contained in the other three files. In addition, once you have included ADXGEN, you must call the ADXINIT macro whenever you start a new design.

For example, if you want to work with the macros found in the ADXFF file, you must first submit:

```
%adxgen      Define general macros
%adxff       Define macros for fractional factorial designs
%adxinit     Initialize for a new design
```

The rest of this chapter consists of four major sections, corresponding to the four files that contain macros. The ADXGEN file is covered first, since it must be included before the other files.

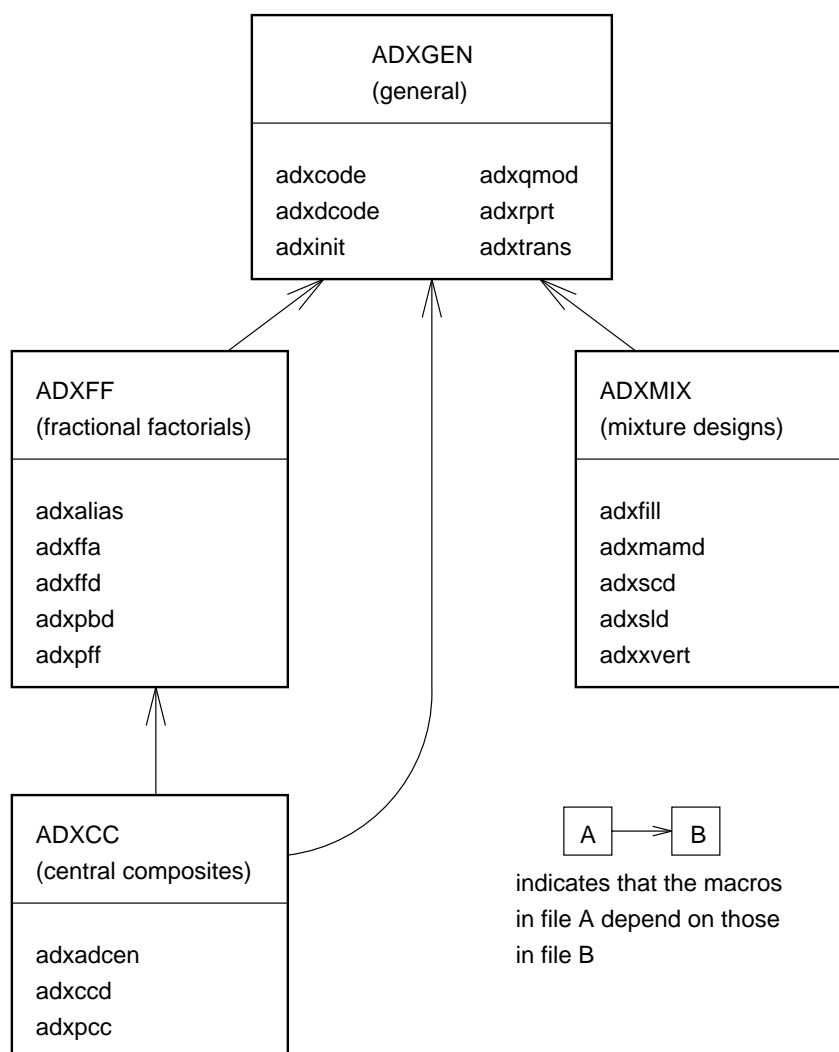


Figure E.1. Structure of Files for the Macro Collection

General Macros: ADXGEN File

The ADXGEN file contains six general-purpose macros. These macros can be used on all types of designs created by macros in other parts of the collection. The ADXGEN file also contains macros that are called by other parts of the collection. As a result, **you must include the ADXGEN file before you include any of the other files.**

The macros contained in ADXGEN are as follows:

ADXC <small>ODE</small>	codes the factor values for a design to standard values (for example, ± 1 for two-level factors).
ADXDC <small>ODE</small>	decodes a design from standard values like -1 and +1 to values that you specify.
ADXIN <small>IT</small>	sets several global macro variables. You should call the ADXINIT macro whenever you start a new design.
ADXQ <small>MOD</small>	adds terms for a second-order model to the design data set and writes the second-order regression model to a macro variable.
ADXR <small>PRT</small>	randomizes a design and displays a data collection report.
ADXTR <small>ANS</small>	finds an optimal Box-Cox power transformation for the response variable.

The next six sections cover these macros in detail.

ADXCODE: Code a Design for Analysis

%adxcode(*dsin*, *dsout*, *vlst*, *bvar*)

where

<i>dsin</i>	names the SAS data set that contains the design to be coded.
<i>dsout</i>	names the output SAS data set that contains the coded design. The value of <i>dsout</i> can be the same as <i>dsin</i> ; this simply replaces the uncoded design with the coded design.
<i>vlst</i>	lists the design factors.
<i>bvar</i>	names the block factor. If the design does not contain a block factor, omit <i>bvar</i> .

The ADXCODE macro translates uncoded values of design and block factors into coded values. Once an experiment has been run and you are ready to analyze the resulting data, values for the design factors must be coded into the standard values of -1, 0, +1, and so on. This makes the estimated effects of different factors directly comparable. Note that the ADXFFA macro, which performs analysis of fractional factorial designs, assumes that the data are coded.

For example, suppose you have an uncoded design stored in the data set REACTOR. The design factors are TIME, ROTOR, OVEN, GLUE, and TEMP. The block factor is DAY. You can code the factor values with the following macro call:

```
%adxgen
%adxcode(reactor,reactor,time rotor oven glue temp,day)
```

In the call above, the new, coded data set is named REACTOR. The uncoded data is no longer available.

The example above uses factor names other than the default names (T1, T2, and so on). To rename factors from the defaults, use the ADXDCODE macro, which is discussed below. The ADXCODE macro performs the opposite function of the ADXDCODE macro, except that the ADXCODE macro does not reset factor names to the defaults.

ADXDCODE: Decode a Design

```
%adxrcode (ds, desc, block)
```

where

ds is the name of the data set that contains the design. By default, the name of the last design constructed is used.

desc is a description of how to decode the design factors. Each description gives the old factor name, the new factor name, the low value of the factor, and the high value of the factor as follows:

old-name new-name low-value high-value

The low value corresponds to the coded value of -1, and the high value corresponds to the coded value of +1. Descriptions for different factors are separated by slashes. For character values, enclose each character string in angle brackets (< >). To simply rename factors, you can omit the low and high values in the description. The factors are not decoded in this case.

block is a description of how to decode the block factor. The description gives the old name for the block factor, the new name for the block factor, and the values for the factor. For character values, enclose each character string in angle brackets (< >). If the design does not contain a block factor, omit *block* from the call to ADXDCODE.

The ADXDCODE macro customizes designs by changing the factor names and values from the defaults to ones relevant to your application. The default names for design factors are T1, T2, T3, and so on; the default name for a block factor is BLOCK. Once you use the ADXDCODE macro, the design data set no longer contains these default factors.

For example, suppose you have created a fractional factorial design for 5 factors in 16 runs and 4 blocks in the data set DESIGN, using the following commands:

```
%adxgen
%adxff
%adxinit
%adxffd(design,5,16,4)
```

Now suppose you want to rename and decode the 5 design factors as follows:

Factor Name	Low Value	High Value
TIME	1	3
ROTOR	on	off
OVEN	1	2
GLUE	0	1
TEMP	250	400

In addition to changing values for the design factors, you want to change the name of the block factor to DAY and give it the values *Mon*, *Tue*, *Wed*, and *Thu*. The following macro call performs these decoding tasks:

```
%adxrcode(design, t1 time 1 3
            /t2 rotor <on> <off>
            /t3 oven 1 2
            /t4 glue 0 1
            /t5 temp 250 400,
            block day <Mon> <Tue> <Wed> <Thu>)
```

ADXINIT: Set Global Macro Variables

%adxinit

The following macro variables are used for global communication between the various macros:

ADXDS	the name of the data set that contains the last design constructed
ADXFIT	the correct model to fit to a fractional factorial design. ADXFIT contains the right-hand side of the model that you can use to analyze the design with a regression procedure. ADXFIT contains a model with main-effect terms, and possibly, two-factor interactions (depending on the resolution of the design). The variable does not contain models with three-factor or higher interactions.
ADXNB	the number of blocks in the current design

ADXNF	the number of factors in the current design
ADXNFIT	the number of terms in the correct model for a fractional factorial design
ADXRES	the resolution of a fractional factorial design
ADXVLST	the list of factor names for the current design

Whenever you start a new design, you must initialize these variables to null values by calling the ADXINIT macro. Note that the ADXINIT macro does not need to be used each time you use one of the other macros; use it only when you start a new design.

ADXQMOD: Set Up a Second-Order Model

`%adxqmod(dsin, dsout, vlst, ORDER2)`

where

<i>dsin</i>	names the SAS data set that contains the original design.
<i>dsout</i>	names the output SAS data set that contains the original design and new variables for second-order terms in the model. The value of <i>dsout</i> can be the same as <i>dsin</i> ; in this case, the original data set is replaced.
<i>vlst</i>	lists the design factors. These must all be numeric.
ORDER2	adds the quadratic terms to the model. If you omit ORDER2, cross-product terms (interactions) are added, but quadratic terms are not.

The ADXQMOD macro adds variables that represent cross-product terms to a design data set, optionally adds quadratic terms to a design data set, and sets up the second-order regression model in the global macro variable ADXFIT.

The names of the new variables are formed by concatenating the names of their factors. For example, the name of the variable for the cross-product of A and B is AB. You should verify that this convention produces a valid variable name in all cases. To concatenate names:

- Each combination of two factor names should have a length of 8 or less.
- The constructed name for a cross-product should not conflict with a previously defined variable in the design.

For example, the variable names A and B meet these criteria. The variable names TEMPERAT and PRES do not.

The ADXQMOD macro makes it easier for you to use the REG procedure* to analyze data for a second-order model. The REG procedure does not accept terms of the form $X1*X2$, so normally you need to create new variables for quadratic and interaction terms in a DATA step.

The following statements create a central composite design with the ADXCCD macro and output the design to a SAS data set named CENC1. Then the ADXQMOD macro adds new variables for quadratic terms and outputs the original design and new variables to a SAS data set named CENC2.

```
%adxgen  
%adxff  
%adxcc  
%adxinit  
%adxccd(cenc1,6,32,14,2.3784)  
%adxqmod(cenc1,cenc2,t1 t2 t3 t4 t5 t6,order2)
```

ADXRPT: Randomize Design and Display a Data Collection Report

%adxprrt(*ds*, *resp*, *bvar*)

where

ds is the name of the data set that contains the design. By default, the name of the last design constructed is used.

resp is the name of the response variable to add to the design.

bvar is the name of the block factor for the design. Omit *bvar* if your design does not have a block factor.

The ADXRPT macro randomizes the design and displays a data collection form, taking the block structure into account if *bvar* is specified. This form is listed in the output window and contains the randomized design with the addition of a column of blanks for entering the response variable. The form contains a column for entering response values. To obtain this column, the macro adds a numeric variable to the design (with the name you specify) with missing values formatted as underscores.

Suppose you want a data collection form for a design stored in the data set DESIGN. The block factor is named DAY, and you want to add a column for the response variable YIELD. Use the following call:

```
%adxgen  
%adxprrt(design,yield,day)
```

*Refer to the *SAS/STAT User's Guide* for details of the REG procedure.

ADXTRANS: Determine an Optimal Box-Cox Power Transformation

`%adxtrans(dsin, dsout, resp, model, intvllo, intvlhi, numintvl)`

where

<i>dsin</i>	names the SAS data set that contains the coded design and the original, untransformed values of the response variable.
<i>dsout</i>	names the output SAS data set to contain the coded design and the transformed values of the response. The value for <i>dsout</i> can be the same as <i>dsin</i> . In this case the original values of the response variable are replaced by the transformed values.
<i>resp</i>	names the response variable for analysis. The Box-Cox family of transformations requires all values of <i>resp</i> to be positive. If <i>resp</i> has zero or negative values but you still want to estimate an optimal transformation, add an amount <i>c</i> to each response, where <i>c</i> is greater than the absolute value of the most negative value of <i>resp</i> .
<i>model</i>	lists the independent variables for analysis.
<i>intvllo</i>	is the bottom end of the range for computing the likelihood. The default value is -2.
<i>intvlhi</i>	is the top end of the range for computing the likelihood. The default value is 2.
<i>numintvl</i>	is the number of intervals tested in the range for computing the likelihood. The default value is 21.

The ADXTRANS macro uses maximum likelihood theory to estimate an optimal transformation within the class of *power transformations* of the form

$$z = y^{\lambda}$$

When $\lambda = 0$, a limit argument justifies using the transformation $z = \log(y)$. (Refer to Box and Cox, 1964.) The algorithm computes the likelihood of the data for several values of λ in the test range and takes the value for which the likelihood is maximized as the estimated optimal transform. By default, the test range is $\lambda = -2, -1.8, -1.6, \dots, 1.8, 2$.

The ADXTRANS macro is useful in situations where the original form of the measurements for the response variable is not the best one to use when analyzing the data. For example, in many situations the original data are not normally distributed, but after applying a log transformation, the transformed data are normally distributed.

Suppose the RESULT data set contains factors T1, T2, and T3 along with values for a response variable BURST. To estimate an optimal Box-Cox power transformation using the defaults for the number of intervals and the ends of the range, use the following statements:

```
%adxgen
%adxtrans(result,tresult,burst,T1 T2 T3)
```

The design with the transformed values for the response is stored in the TRESULT data set.

Output from ADXTRANS

The ADXTRANS macro produces an output listing as well as the ADXREG output data set. The ADXREG data set contains the following variables for each value of λ in the test range:

ADXCONF	a character variable of length 1. The value of ADXCONF is an asterisk (*) if the associated value of λ is within a 95% confidence interval of the estimated optimum. Otherwise, the value of ADXCONF is a blank.
ADXLAM	the value of λ .
ADXLIKE	the log-likelihood based on the fit of the model to the transformed response.
RMSE	the root mean squared error based on the fit of the model to the transformed response.
<i>effect</i>	<i>t</i> -values for estimates of parameters for effects in the model. The names for <i>effect</i> depend on the model. If the parameters in the model are T1 and T2, the ADXREG data set contains new variables T1 and T2, whose values are the <i>t</i> -values for the parameter estimates. The variable that contains <i>t</i> -values for the intercept parameter is named INTERCEP.

The ADXTRANS macro lists

- the values of λ (ADXLAM)
- the root mean squared error (_RMSE_)
- the confidence interval indicator (ADXCONF), which is either an asterisk or blank, as described for the ADXREG data set above
- a plot of *t*-values against λ

The data require a transformation if the confidence interval does not contain $\lambda = 1$. A plot of _RMSE_ against λ should dip fairly steeply with a minimum in the region of the optimum value. Typically, many of the parameter estimates might appear to be significant outside the region of the optimum λ , but near it only a few will be highly significant and the rest will be insignificant.

Fractional Factorial Design Macros: ADXFF File

The ADXFF file contains five macros and the ADXFFD SAS data set. These macros are used to construct and manipulate orthogonal fractional factorial designs for two-level factors. Two kinds of orthogonal designs are available: orthogonally confounded designs and Plackett-Burman screening designs.

In orthogonally confounded designs, any two effects (main effects or interactions) are either orthogonal or totally confounded. Effects that are totally confounded cannot be estimated separately. As designs decrease in size, fewer and fewer effects are estimable. One goal of design is to keep the effects of interest unconfounded with each other.

The resolution of a design indicates what kinds of effects are confounded with each other. In resolution 5 designs, main effects and two-factor interactions are all estimable and unconfounded with each other. In resolution 4 designs, two-factor interactions are confounded with each other, but they are free of main effects. In resolution 3 designs, main effects are confounded with two-factor interactions. Choose a design with maximum resolution, subject to your particular constraints on the number of runs you can make. See the “Resolution” section on page 491 in Part 3, “The FACTEX Procedure,” for more details about design resolution.

The macros in the ADXFF file enable you to construct orthogonal fractional factorial designs for as many as 11 treatments, as many as 128 runs, optionally in blocks with as few as 2 runs. If you require more specialized fractional factorial designs, you should use the FACTEX procedure; see Part 3, “The FACTEX Procedure.”

You can also use the macros to construct Plackett-Burman designs for up to 47 two-level factors. Plackett-Burman designs are used to estimate the main effects of a large number of factors in as few runs as possible. They are sometimes known as *screening designs* because they are used at the initial stages of experimentation to identify important factors. However, there is no protection against two-factor interactions, so these designs should be used with caution.

The macros contained in ADXFF are as follows:

- ADXALIAS computes the alias structure of fractional factorial designs.
- ADXFFA analyzes fractional factorial designs.
- ADXFFD constructs fractional factorial designs.
- ADXPBD constructs Plackett-Burman designs.
- ADXPFF lists available fractional factorial designs.

Additionally, the ADXFFD data set is created in your work library. This data set contains the parameters for all possible orthogonally confounded designs of 128 or fewer runs.

The next six sections describe the macros and data set introduced above.

ADXALIAS: Compute the Alias Structure

`%adxalias(ds, vlst, nv, res, nb, NOPRINT)`

where

<i>ds</i>	gives the name of the SAS data set that contains the design. The design must be in coded form, either as originally constructed or after recoding with the ADXCODE macro.
<i>vlst</i>	lists the design factors.
<i>nv</i>	gives the number of design factors. By default, <i>nv</i> is constructed from <i>vlst</i> .
<i>res</i>	gives the resolution for the design.
<i>nb</i>	gives the number of blocks in the design.
NOPRINT	suppresses listing the alias structure. This is useful when you want only the output data sets.

By default, if *ds*, *vlst*, *res*, or *nb* are omitted, the values from the last design constructed are used.

In orthogonally confounded fractional factorial designs, certain effects are confounded or *aliased* with each other. The alias structure of the design is the pattern of this confounding.

The ADXALIAS macro

- lists the alias structure and stores it in the ADXALIAS data set. The alias structure is listed as a series of *alias chains* of effects. Two effects are listed as equal to each other if they are aliased with one another in the design. Effects are listed as equal to “(BLOCKS)” if they are confounded with the block effect. Two aliased effects cannot be simultaneously estimated, so the model constructed in the ADXFIT variable (see below) contains one effect from each alias chain and no effect from a chain that is aliased with blocks.
- creates the ADXEFF data set, which contains the design and additional variables for cross-product terms. This data set is generally used only by other macros in the collection.
- creates the macro variables ADXFIT and ADXNFIT. ADXFIT is the right-hand side of the appropriate model to use to analyze the design with PROC REG (in SAS/STAT software) when ADXEFF is the input data set to the REG procedure. ADXNFIT is the number of terms for the model given by ADXFIT.

It is a good idea to examine the alias structure of your design before decoding it, and to name factors so that effects you want to estimate are estimable and not aliased with one another. Alternatively, you can use the FACTEX procedure to construct and decode the design; in this procedure you can directly specify which effects to estimate

and which effects to consider nonnegligible. See Part 3, “The FACTEX Procedure,” for more details on using the FACTEX procedure.

Suppose you create a fractional factorial design for 5 factors in 16 runs and 4 blocks and you output it to the data set DESIGN with the following statements:

```
%adxgen
%adxff
%adxinit
%adxffd(design,5,16,4)
```

To examine the alias structure of this design, submit the statement

```
%adxalias(design,t1 t2 t3 t4 t5,5,4,4)
```

The statement above uses the default variable names for treatment factors. If you call the ADXALIAS macro immediately after constructing the design, you can leave all parameters blank. Thus, the following statements produce the same output as the two sets of statements above:

```
%adxgen
%adxff
%adxinit
%adxffd(design,5,16,4)
%adxalias()
```

ADXFFA: Analyze Fractional Factorial Designs

```
%adxffa(ds, resp, vlst, res, nb)
```

where

ds gives the name of the SAS data set that contains the design. **The design must be in coded form, either as originally constructed or after recoding with the ADXCODE macro.**

resp gives the name of the response variable (dependent variable) for analysis.

vlst lists the design factors.

res gives the resolution for the design.

nb gives the number of blocks in the design. If the design does not use blocking, *nb* can be omitted.

By default, if *ds*, *vlst*, *res*, or *nb* are omitted, the values from the last design constructed are used.

The ADXFFA macro computes a standard analysis for orthogonal two-level fractional factorial designs. The analysis lists effect estimates and significance levels, and plots the estimates against the quantiles of a normal distribution. The analysis provided by the ADXFFA macro considers only main effects and two-factor interactions. In the

plot, the horizontal axis (labeled “EFFEST”) is the effect estimate. To interpret the plot, use the values from the horizontal axis to identify effects.

If there are any degrees of freedom available to estimate error in the design, then the significance levels given for the effect estimates are based on tests using the estimated error term. Otherwise, a conservative estimate of error is constructed for each effect by pooling all the other effect estimates. In this situation, the normal plot of effects is a more useful analysis tool than the significance tests. The normal plot is based on the assumption that only a few of the effects will be significant and the rest are all due to noise. The nonsignificant effects should fall roughly on a line, with significant effects deviating from the line. Refer to Box, Hunter, and Hunter (1978) for more information on interpreting normal plots of effects.

For example, to obtain an analysis for a resolution 4 design contained in the DESA data set, with three factors WATER, FERT, and SPECIES and the response variable YIELD, use the following statements:

```
%adxgen  
%adxff  
%adxinit  
%adxffa(desa,yield,water fert species,4)
```

ADXFFD SAS Data Set

The parameters for all possible orthogonally confounded designs of 128 runs or fewer are stored in the data set ADXFFD, which is created in the work library when you use %ADXFF to include the ADXFFD file.

ADXFFD: Construct Fractional Factorial Designs

```
%adxffd(ds, nf, nr, nb)
```

where

ds is the name of the output data set for the constructed design. This data set contains variables for design factors and for a block factor, if the design uses blocking. Default names are used. The default names for design factors are T1, T2, T3, and so on; the default name for a block factor is BLOCK. You can change these default names using the ADXDCODE macro, described in the “General Macros: ADXGEN File” section on page 1876.

nf is the number of design factors.

nr is the number of runs in the design.

nb is the number of blocks in the design. Omit *nb* if the design does not involve blocking.

The ADXFFD macro constructs a fractional factorial design and saves the design in a data set. If you want to construct a fractional factorial design but need to see a list of available designs, use the ADXPFF macro to display the list.

Suppose you want a design for 5 factors. You can first use the ADXPFF macro to find that there is a design in 16 runs and 4 blocks which suits your needs. Then you can construct this design using the ADXFFD macro, as in the following statements:

```
%adxgen  
%adxff  
%adxinit  
%adxpff( (ntmts=5) )  
%adxffd(ff5,5,16,4)
```

The design is contained in data set FF5. You can use the default factor levels (-1 and +1), or you can use the ADXDCODE macro to customize the design.

ADXPBD: Construct Plackett-Burman Designs

%adxpbd(*ds*, *nf*)

where

ds gives the name of the output data set for the constructed design.

nf gives the number of factors in the design.

The ADXPBD macro constructs Plackett-Burman designs. These are orthogonal main-effect screening designs with a minimum number of runs, for 1 to 47 factors. The designs were first discussed by Plackett and Burman (1947).

The number of runs in these designs will be the smallest multiple of 4 that is greater than the number of factors. These are main-effects-only designs. The designs give no protection against two-factor interactions and have few (if any) degrees of freedom available for error. Thus, these designs should be used with caution. They are most appropriate in situations where a very large number of factors need to be screened and a minimum-run design is absolutely necessary.

To construct a design for 11 two-level factors and store it in the data set A, use the following statements:

```
%adxgen  
%adxff  
%adxinit  
%adxpbd(a,11)
```

ADXPFF: List Available Fractional Factorial Designs

%adxpff(*expression*)

where *expression* is a valid comparison expression involving one or more of the following design parameters:

Part 10. The CAPABILITY Procedure

NTMTS the number of design factors, where $2 \leq NTMTS \leq 11$.

NRUNS the number of experimental runs, where $4 \leq NRUNS \leq 128$.

NBLKS the number of blocks in the design.

K the size of each block.

RES the resolution of the design, where $RES \geq 3$.

Each part of *expression* is a comparison involving one or more of the above design parameters. Each part of *expression* must be enclosed in parentheses, and the entire expression must also be enclosed in parentheses. For example, a simple *expression* that requests designs with 5 or more treatments is shown below:

```
%adxgen  
%adxff  
%adxpff((ntmts ge 5))
```

You can use AND and OR operators to form a complex *expression*. The symbols & and | can also be used for AND and OR, respectively. For example, the following statements list all regular fractions for six factors with either less than 80 runs or exactly 8 blocks:

```
%adxgen  
%adxff  
%adxpff((ntmts=6) & ((nruns < 80) or (nblks=8)))
```

The ADXPFF macro is useful when you do not know the exact parameters of the design you want to construct. The ADXPFF macro also lists the valid ADXFFD macro call for each design. Once you have chosen a design from the list, you can simply submit the associated ADXFFD macro call, filling in a SAS data set name for the value **data-set-name** shown in the listing.

For example, suppose you want to find a design

- for 5 or more factors
- in 32 or fewer runs
- of resolution 4 or higher

Submitting the statements

```
%adxgen  
%adxff  
%adxpff((ntmts >= 5) & (nruns <= 32) & (res >= 4))
```

returns 51 designs, with as many as 11 factors and with blocks as small as 2 runs per block. You can choose one of these 51 designs and then submit the appropriate ADXFFD call to create the design and output it to a data set.

Central Composite Design Macros: ADXCC File

The ADXCC file contains three macros that construct central composite designs. These designs are also known as Box-Wilson designs and are useful in response surface exploration. These designs have three components: a *factorial portion*, an *axial portion*, and *centerpoints*. The factorial portion is a fractional factorial design of resolution 5 or higher. The axial portion consists of runs with all factors except one at their central levels and the remaining factor at its highest and lowest values, respectively. The centerpoints are runs where all factors are at their central levels.

Central composite designs are especially suited to exploring a second-order response surface over several continuous factors. The designs are generally meant to be used after you have screened for the important factors with a two-level design (a fractional factorial design, for example).

Note that before including the ADXCC file, you must include both the ADXGEN file and the ADXFF file.

The ADXCC file contains three macros as follows:

ADXADCEN adds centerpoints to a central composite design.

ADXCCD constructs central composite designs.

ADXPCC lists available central composite designs.

The next three sections cover these three macros in the order shown in the list above.

ADXADCEN: Add Center Points

%adxadcen(*ds*, *vlst*, *nc*, *bvar*)

where

ds is the name of the SAS data set that contains the design to which you want to add centerpoints.

vlst is the list of design factors.

nc is the number of centerpoints to be added to the design.

bvar is the block variable, if any. If blocking is involved, then the design is sorted by blocks, and *nc* center points are added to each block.

The ADXADCEN macro adds a specified number of centerpoints to a design stored in a SAS data set. Centerpoints are runs where all factors are at their central levels (with coded values of 0). They are often added to a fractional factorial design to allow testing for lack-of-fit, and they form part of a central composite design.

Suppose you use the ADXFFD macro to create a fractional factorial design for 5 factors in 16 runs and 4 blocks and output the design to the data set named DESIGN, keeping the default factor names T1, T2, T3, T4, and T5 and the default block factor

name (BLOCK). You want to add three centerpoints to each block of the design. The following statements create the fractional factorial design, output it to the data set DESIGN, and add three centerpoints to each block:

```
%adxgen
%adxff
%adxcc
%adxinit
%adxffd(design,5,16,4)
%adxadcen(design,t1 t2 t3 t4 t5,3,block)
```

ADXCCD: Construct Central Composite Designs

%adxccd(*ds, nf, nr, nc, alpha, nb*)

where

ds is the name of the output SAS data set for the constructed design. This data set contains variables for design factors and for a block factor, if the design uses blocking. Default names are used. The default names for design factors are T1, T2, T3, and so on; the default name for a block factor is BLOCK.

nf is the number of design factors.

nr is the number of points in the factorial portion of the design and must be a power of 2. There must be a resolution 5 design in *nf* factors for the value of *nr* you specify. For example, if *nf*=5, then *nr* can be 16, but not 8.

nc is the number of centerpoints in the design. If you do not use blocking, *nc* is a nonnegative integer. If you use blocking, *nc* is two integers separated by a slash. The first integer gives the number of centerpoints in each block in the factorial portion of the design, and the second integer gives the number of centerpoints for the axial block.

alpha is the value of the axial extreme.

nb is the number of blocks in the design. Since the blocks for a design are the blocks for the factorial portion together with the axial block, the number of blocks must be of the form $2^n + 1$ for some *n*. If the design does not use blocking, *nb* can be omitted.

The ADXCCD macro constructs a central composite design according to the parameters you specify and outputs the design to a SAS data set.

Suppose you want to examine a response surface over 5 factors. You can use the ADXPCC macro to find that there is a design in 33 runs and 2 blocks. One block is a half-fraction of the full 2^5 design, and the other block is the axial portion of the design. You can produce the output from the ADXPCC macro and construct the design with the following statements:

```

%adxgen
%adxff
%adxcc
%adxinit
%adxpcc(5)
%adxccd(a,5,16,6/1,2.0000,2)

```

The design is output to data set A. Note that 6/1 is used for nc . Since the design involves blocking, this specifies that the factorial block contains 6 centerpoints and the axial block contains 1 centerpoint.

You can produce a design (output to data set B) with just one centerpoint in the axial block with the following statements:

```

%adxgen
%adxff
%adxcc
%adxinit
%adxccd(b,5,16,0/1,2.0000,2)

```

ADXPCC: List Available Central Composite Designs

%adxpcc(nf)

where nf gives the number of factors you want to study, where $2 \leq nf \leq 8$. This macro lists available designs and is useful when you do not know the exact parameters of the central composite design you want to construct.

The ADXPCC macro lists designs of various sizes that may or may not involve blocking. In each design, the values of the axial extreme are chosen so the design is *rotatable*, and the number of centerpoints is chosen so the design is as nearly *orthogonal* as possible and *orthogonally blocked* if blocking is involved. For details on these design characteristics, refer to Myers (1976).

In addition to listing designs, the output from the ADXPCC macro gives statements to construct each design using the ADXCCD macro. In the statements shown on the output, you need to fill in a SAS data set name for the value **data-set-name**. When you have chosen a design, you can simply submit the corresponding statement to construct the design.

When choosing a design, consider the overall number of runs required, the number of runs required in each block, and the number of centerpoints required.

The statements below produce a list of four designs, which differ depending on whether they are based on half-replicates, and whether they involve blocking. The output also includes the ADXCCD macro statements for each of these designs.

```

%adxgen
%adxff
%adxcc
%adxpcc(5)

```

Mixture Design Macros: ADXMIX File

The ADXMIX file contains five macros for use in mixture experiments. In a mixture experiment, the independent factors are proportions of different components of a blend. For example, if you want to optimize the tensile strength of stainless steel, then factors of interest might be the proportions of iron, copper, nickel, and chromium in the alloy. The fact that the proportions of the different factors must sum to 100% complicates the design as well as the analysis of mixture experiments.

When the mixture components are subject only to the constraint that they must sum to one, there are standard designs for fitting standard models. When mixture components are subject to additional constraints, such as a maximum and minimum value for each component, nonstandard designs are called for. The ADXMIX file contains macros for both the unconstrained and constrained situations as follows:

ADXFILL	adds interior points to any design with a convex feasible region.
ADXMAMD	constructs McLean-Anderson mixture designs. McLean and Anderson (1966) suggest using the vertices of the feasible region when fitting first- or second-order models.
ADXSCD	constructs simplex-centroid designs. These are standard designs.
ADXSLD	constructs simplex-lattice designs. These are standard designs.
ADXXVERT	constructs extreme vertices designs. Snee and Marquardt (1974) and Snee (1975) suggest generating vertices and centroids of the faces of the constrained region, then choosing the design points from this set. ADXXVERT constructs the set of vertices and face-centroid points. If the set is too large to run as a design, you can use the OPTEX procedure to choose an optimal subset.

The next five sections give details on these macros.

ADXFILL: Filling in the Design Region

%adxfill(*ds*, *nmlst*)

where

ds is the name of the SAS data set holding the design to be filled in.

nmlst is the list of factor names.

The ADXMAMD and ADXXVERT macros construct constrained mixture designs from the vertices and generalized edge centroids of the constrained feasible region. These will be the points from which an optimal design may be chosen for fitting a first or second-order model. However, in practice, experimenters often add a few points spread around the interior of the feasible region. These additional points may protect the fit from bias if the model is incorrect. You can use the ADXFILL macro to add such points to any design with a convex feasible region.

The ADXFILL macro works simply by adding to the design the average of each pair of points. Thus, if the data set BOX contains the four vertices of a box, labeled a, b, c and d, then the following statements fill in the box as shown in Figure E.2.

```
%adxgen
%adxmix
%adxinit
%adxfill(box,x1 x2);
```

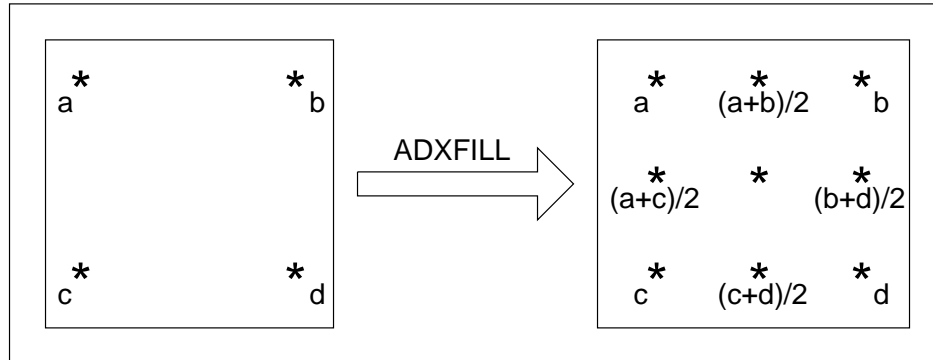


Figure E.2. Filling in a Data Set with the ADXFILL Macro

Note that the centerpoint is the average of more than one pair of points, but it is only included in the filled-in data set once.

You may execute the ADXFILL macro more than once for a design, but note that the number of points can get large very fast, growing exponentially in the number of times you execute the macro. You can use this macro with the uniform coverage criterion of the OPTEX procedure; see Example 24.10 on page 762 for an example.

ADXMAMD: Construct McLean-Anderson Mixture Designs

```
%adxmamd(ds, vlst)
```

where

ds is the name of the SAS data set to contain the constructed design. This data set contains variables for design factors. Default names are used. The default names for design factors are T1, T2, T3, and so on.

vlst is a list of mixture components and their constraints. Separate information for different components with a slash (/) and specify each component as follows:

name loval-hival

where *name* gives the name of the component, *loval* gives the low value for the component, and *hival* gives the high value for the component. If you omit *loval* or *hival*, the defaults are 0 and 1, respectively.

Use the ADXMAMD macro when your mixture factors are subject to constraints of the form

$$\text{low-value} \leq \text{component} \leq \text{high-value}$$

In this situation, you cannot study a response over the entire mixture region, so you cannot use the ADXSCD or ADXSLD macros. You can use the ADXMAMD or the ADXXVERT macros. The ADXMAMD macro constructs a design composed of vertices of the feasible region, as described by McLean and Anderson (1966). If you do not specify any constraints, the McLean-Anderson design is the same as a simplex-lattice design of order 1.

For example, consider an experiment to study the effects of a lubricant additive on three-component lubricant blends. The additive and the three components are subject to restrictions as follows:

Mixture Component	Range of Values
Additive (ADD)	7% to 18%
Component A (A)	0% to 30%
Component B (B)	37% to 70%
Component C (C)	0% to 15%

To construct the McLean-Anderson design and output it to the data set LUB, submit the following statements:

```
%adxgen
%admix
%adxinit
%adxmamd(lub,add .07-.18 /a -.3 /b .37-.7 /c -.15)
```

In the statements above, the low values for lubricant components A and C are omitted; since these low values are 0, you can use the defaults.

ADXSCD: Construct Simplex-Centroid Designs

```
%adxscd(ds, vlst, m)
```

where

ds is the name of the SAS data set for the constructed design. This data set contains variables for design factors. Default names are used. The default names for design factors are T1, T2, T3, and so on.

vlst is a list of mixture components.

m is the degree of the model.

The ADXSCD macro can be used to construct *simplex-centroid designs* of any degree *m*. These designs are efficient for fitting an m^{th} -degree polynomial model to mixture data. A simplex-centroid design of degree *m* is composed of mixtures with only one factor present, mixtures with two factors present in equal amounts, mixtures with

three equal factors, and so on up to m ; and the mixture that contains all factors in equal amounts.

The design data set includes the levels of the mixture components for each point in the design. In addition, the data set includes a variable, DIMEN, which gives the dimension of the face from which the corresponding centroid is computed. A 0-dimensional centroid is a vertex, a 1-dimensional centroid is computed from an edge, and so on.

For example, suppose you want a design for an experiment on blends of three fruit juices: watermelon, pineapple, and orange. To construct a simplex-centroid design of order 2, submit the following statements:

```
%adxgen
%adxmix
%adxinit
%adxscd(juice,watermel pineapple orange,2)
```

The design is output to the JUICE data set, which contains the variables WATERMEL, PINEAPPL, and ORANGE. The values of these variables give the proportions of each juice for runs in the experimental design.

ADXSLD: Construct Simplex-Lattice Designs

```
%adxsls(ds, vlst, m)
```

where

ds is the name of the SAS data set for the constructed design. This data set contains variables for design factors. Default names are used. The default names for design factors are T1, T2, T3, and so on.

vlst is a list of mixture components.

m is the reciprocal of the increment.

The ADXSLD macro can construct *simplex-lattice designs* of any degree m . These designs have points positioned uniformly over the feasible region and are thus useful for investigating a response surface over the entire simplex.

For example, suppose you want a design for an experiment on blends of three fruit juices: watermelon, pineapple, and orange. To construct a simplex-lattice design of order 4, submit the following statements:

```
%adxgen
%adxmix
%adxinit
%adxsls(juice2,watermel pineapple orange,4)
```

The design is output to the JUICE2 data set, which contains the variables WATERMEL, PINEAPPL, and ORANGE. The values of these variables give the proportions of each juice for runs in the experimental design.

ADXXVERT: the XVERT Algorithm

%adxxvert(*ds*, *vlst*, *m*)

where

ds is the name of the SAS data set for the constructed design. This data set contains variables for design factors. Default names are used. The default names for design factors are T1, T2, T3, and so on.

vlst is a list of mixture components and their constraints. Separate information for different components with a slash (/) and specify each component as follows:

name loval-hival

where *name* gives the name of the component, *loval* gives the low value for the component, and *hival* gives the high value for the component. If you omit *loval* or *hival*, the defaults are 0 and 1, respectively.

m is the maximum order of centroid to be generated. By default, centroids of all orders are generated.

Use the ADXXVERT macro when your mixture factors are subject to constraints of the form

$$\text{low-value} \leq \text{component} \leq \text{high-value}$$

In this situation, you cannot study a response over the entire mixture region, so you must use the ADXMAMD or ADXXVERT macros instead of the ADXSCD or ADXSLD macros. The ADXXVERT macro generates the points corresponding to the vertices of the feasible region and adds the centroids of the edges and generalized faces of the region as well, up to a specified order *m*. The resulting set of points can be too large for a design, but it provides a good candidate set from which to choose design points. Two additional variables are included in the data set, with the following values for each point:

DIMEN the dimension of the face of which the corresponding point is the centroid

DIST the average distance from the vertices on the face to the centroid

You can try to choose design points directly based in DIMEN and DIST, or you can use the OPTEX procedure to choose an optimal design from the set of candidate points; see Example 24.10 on page 762 for a detailed example.

For example, consider an experiment to study the effects of a lubricant additive on three-component lubricant blends. The additive and the three components are subject to restrictions as follows:

Mixture Component	Range of Values
Additive (ADD)	7% to 18%
Component A (A)	0% to 30%
Component B (B)	37% to 70%
Component C (C)	0% to 15%

To construct the vertices and generalized face-centroids of the feasible region and output these points to the data set LUB, submit the following statements:

```
%adxgen
%adxmix
%adxinit
%adxxvert(lub,add .07 -.18 /a -.3 /b .37 -.7 /c -.15)
```

Since *m* is omitted in the ADXXVERT macro call above, centroids of all orders are generated and output to LUB.

Examples

The following table lists programs in the SAS/QC Sample Library that use the collection of macros described in this appendix. The type of design for each example program is listed, with the macro files that each program uses (in addition to ADXGEN).

Example	Design	Additional Macro Files Used
adxeg1	Resolution 5 fractional factorial	ADXFF
adxeg2	Screening with fold-over	ADXFF
adxeg3	Blocked fractional factorial	ADXFF
adxeg4	Central composite	ADXCC, ADXFF
adxeg5	Simplex-lattice	ADXMIX
adxeg6	Constrained simplex-lattice	ADXMIX
adxeg7	3^3 with optimal transformation	

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *SAS/QC® User's Guide, Version 8*, Cary, NC: SAS Institute Inc., 1999. 1994 pp.

SAS/QC® User's Guide, Version 8

Copyright © 1999 SAS Institute Inc., Cary, NC, USA.

ISBN 1-58025-493-4

All rights reserved. Printed in the United States of America. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, by any form or by any means, electronic, mechanical, photocopying, or otherwise, without the prior written permission of the publisher, SAS Institute Inc.

U.S. Government Restricted Rights Notice. Use, duplication, or disclosure of the software by the government is subject to restrictions as set forth in FAR 52.227-19 Commercial Computer Software-Restricted Rights (June 1987).

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

1st printing, October 1999

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute in the USA and other countries.® indicates USA registration.

IBM®, ACF/VTAM®, AIX®, APPN®, MVS/ESA®, OS/2®, OS/390®, VM/ESA®, and VTAM® are registered trademarks or trademarks of International Business Machines Corporation.® indicates USA registration.

Other brand and product names are registered trademarks or trademarks of their respective companies.

The Institute is a private company devoted to the support and further development of its software and related services.