COMPUTER-AIDED ENGINEERING OF NONLINEAR MICROWAVE CIRCUITS: YIELD-DRIVEN DESIGN

Final Report

OSA-92-OS-7-R

July 29, 1992

TABLE OF CONTENTS

1. Title of the Project

COMPUTER-AIDED ENGINEERING OF NONLINEAR MICROWAVE CIRCUITS: YIELD-DRIVEN DESIGN

2. IRAP Project Code

20557U

3. IRAP Funding

July 1, 1991 - March 31, 1992:	$45,800.00
April 1, 1992 - June 30, 1992:	$15,200.00
TOTAL IRAP FUNDING	$61,000.00

4. Project Team

P1.	J.W. Bandler	Project Leader
P2.	R.M. Biernacki	Project Engineer
P3.	S.H. Chen	Project Engineer

5. Project Objectives

The purpose of the project was to research the area of statistical design centering for computer-aided design of nonlinear RF and microwave circuits, in particular for design of high frequency analog integrated circuits. The ultimate goal of this project was to develop a software system for yield-driven circuit design in the presence of device, process and other statistics. Based on the research results we were to develop yield-driven design features to be included in our new, marketable packaging of OSA's technology: software system OSA90™. A number of user-oriented features were to be created in order to enable the users to utilize their existing software within the power of our new system.

6. Achievement of Project Objectives

All goals set for this project have been achieved. In June 1992, we released Version 2.0 of our general purpose CAD software system OSA90/hope™ which contains all the features which we planned to develop.

We have decided to market the newly developed features principally within the software system OSA90/hope™ (where hope which stands for harmonic optimization personal environment). OSA90/hope™ is our new general purpose software system for computer-aided design of high frequency and microwave circuits. OSA90™ is the mathematical engine of OSA90/hope™. An option of delivering the OSA90™ engine to customers not interested in circuit design capabilities is still open, as the parts of the whole OSA90/hope™ system that are outside the engine can be easily disabled.

The new software system has been very well received by the attendees of the 1992 IEEE MTT-S International Microwave Symposium and Exhibition, held in Albuquerque, NM, in June 1992, and the 1992 Design Automation Conference, held in Anaheim, CA, also in June 1992. Quite a few companies have recently acquired a demonstration copy and believe that the yield optimization feature will enhance their design capabilities for both mass production and for the first-pass design success.

We have also carried out the necessary research needed to successfully complete different tasks such as development and implementation of relevant algorithms, thorough testing, etc.

Our research results have been included in a few publications, in addition to the internal OSA reports. First, a comprehensive paper [1] has been accepted for publication in the Special Issue of *IEEE Transactions on Microwave Theory Techniques* on Process-Oriented Microwave CAD and Modeling. It is the most reputable journal in the field. A copy of that paper is attached to this report. Also, two new scientific papers [2] and [3] were presented at the 1992 IEEE MTT-S International Microwave Symposium in Albuquerque, NM, in June 1992.

Research and development carried out within the framework of the project are outlined in the following subsections.

Research towards Yield-Driven Design System

1. Study and develop theory for yield optimization of nonlinear microwave circuits operating in the steady state under large-signal excitations.

   We have studied theoretical aspects of yield optimization in conjunction with the harmonic balance simulation technique. We have identified individual error functions for different harmonics and for different circuit responses w.r.t. design specifications. This allowed us to investigate different objective functions for yield optimization.

2. Study and implement objective function formulations for yield-driven design of nonlinear microwave circuits operating in the steady state under large-signal excitations.

   We have studied and implemented a two stage formulation of the objective function where all the error functions for individual outcomes are assembled into a single generalized ℓ_p function, and then the resulting functions for all the outcomes are used to formulate a single one-sided ℓ_1 function. We have also investigated a novel "yield probability function" which has certain smoothing properties. Our experiments show that this objective function behaves better than the one-sided ℓ_1 function in many instances. Efficiency of the two techniques is similar and, at least at the moment, the one-sided ℓ_1 function appears to be more robust.

3. Study and implement the best approach to gradient estimation to facilitate the powerful gradient based optimizers.

   Our previously published FAST technique has been adapted and utilized.

4. Study the optimal time sample selection for generalized Discrete Fourier Transform in two-tone mixer and intermodulation harmonic balance analysis.

   As originally planned, we have devised an algorithm based on a subsidiary optimization process which aims at improving the condition number of the DFT matrix. In the meantime, we studied a very recent publication by Ngoya *et al.* (*IEEE Trans. Circuits and Systems*, 1990, pp. 1339-1355). They proposed an efficient semi-analytical algorithm which we implemented and observed that its robustness is not satisfactory for general purpose applications. Therefore, to achieve a satisfactory level of robustness, both algorithms have been implemented with appropriate switching conditions.

5. Study and implement efficient modeling techniques that can take advantage of available gradient information in constructing reliable approximate models which then will be used instead of actual circuit simulations.

   Multidimensional quadratics are used to model both the circuit responses and their gradients. Only statistical variables are utilized as the model parameters. Non-statistical design parameters (optimization variables) are not involved, thus dimensionality of the model is reduced.

6. Study and develop new methods and numerical algorithms aimed at improving efficiency of statistical design of analog microwave circuits.

   A number of issues have been investigated. Most of them were related to time-saving organization of relevant calculations, skipping unnecessary calculations, saving the results of already performed simulations for retrieving during a subsequent session, etc.

7. Apply statistical design techniques to practical analog microwave circuits reflecting fabrication parameter spreads.

   Several practical circuits have been optimized for maximum yield to test the techniques and our implementation. The most significant one was yield optimization of a three stage X-band MMIC amplifier originally described by Kermarrec and Rumelhard (in *GaAs MESFET Circuit Design*, R. Soares, Ed., Boston, Artech House, 1988, ch. 8).

8. Interact with process and fabrication engineers for measurement data and process information.

   We have tested yield optimization on circuits containing devices that were statistically characterized from real measurement data obtained from Plessey Research Caswell Ltd. We have verified predictability of the model by comparing results of simulation based on this statistical model with the results of simulation that utilized the device data directly.

9. Develop a framework for a comprehensive yield-driven design system for monolithic integrated microwave circuits.

   This involved all previous tasks and resulted in the yield optimization feature available in OSA90/hope™, as originally planned.

Modules for Yield-Driven Design System

1. Module for User Interface and Command Processor for Yield Optimization.

   We have developed the input file syntax for yield optimization to allow for statistical analysis and design optimization to be performed simultaneously. We have expanded global data structures and the user interface layout, including menus and windows.

2. Parser Module for Yield Optimization Information in the Circuit File.

   The input file Specification Block parser of OSA90 has been modified to allow for both statistical analysis and design optimization to be performed simultaneously.

3. Module for Error Function Formulation for Yield Optimization.

   The new module assembles all the error functions for an individual outcome into a single generalized ℓ_p function, and then assembles the ℓ_p functions for all outcomes into a single objective function for yield optimization.

4. Driver Module for Yield Optimization.

   The new driver is invoked when the user requests yield optimization by selecting the corresponding menu option. The module also recognizes whether the quadratic modeling feature has been requested or not, and invokes model creation, model evaluation or actual simulation, respectively.

5. Library Modules for new Active Device and Passive Component Models.

   We have combined the Khatibzadeh and Trew DC model with the small-signal Ladbrooke model and created an extremely promising statistical model for GaAs MESFETs. It is based on physical parameters such as gate length, gate width, doping density, etc. We have also expanded our linear library, most noticeably by a number of microstrip circuit component models. The latter were developed outside of the scope of this project.

6. Modules for Inter-Process Communication.

   In addition to previously available FUN and FDF protocols we have developed the SIM and COM protocols which allow the user to pre- or post-process the input and/or output of the user's programs connected to OSA90 through pipes. We have expanded handling of FUN and FDF Datapipe outputs to be included in yield optimization. We have also developed the means to invoke child processes on different networked machines.

7. Output Data Processor for Yield Optimization.

   The output data processor for nominal optimization has been extended to handle output of yield optimization. We have also developed a postprocessing capability for yield sensitivity option after Monte Carlo analysis. It allows for sorting multiple statistical outcomes for displaying partial yield.

8. Graphics Interface for Yield Optimization.

   All graphical displays developed previously for displaying statistical responses have been enhanced by adding features such as user-defined titles, labelling, etc. A new feature of creating a plotter HP-GL file for graphics hardcopies has been added. We have also developed a new module for plotting partial yield versus sweep parameters to graphically illustrate their influence on satisfying or violating design specifications for multiple statistical circuit outcomes.

7. Protection of Technology

All title and rights to the technology resulting from this project will be retained by OSA. In particular, ownership of the software systems OSA90™ and OSA90/hope™ will not be passed on to any of OSA's customers. Following the common practice of other CAD software vendors the customers will be licensed by OSA only to use the product, and no ownership or other rights will be given to them.

No patents arising from this projects are envisaged. The technology will be protected by distributing executable code only. No customer will be given access to the source code. In addition to respective clauses in the licensing agreement, the product is protected against copying by means of a built-in security module checking the site details, whether authorized by OSA or not, expiry date, etc. Demonstration versions, limited in both time and the size of the problems handled, have been created for free distribution.

8. Follow-up Technical Activities

Several areas of follow-up technical activities are clear at the point of completion of this project. The first one is the development of statistical models for predictable yield optimization. While there exist reliable techniques for statistical modeling, including the one based on multi-outcome parameter extraction and statistical postprocessing developed by OSA and featured in its software system HarPE, the usefulness of equivalent circuit models of devices such as the MESFET is quite limited. Physics based models, such as the Khatibzadeh and Trew model or the Ladbrooke model, are more promising for proper statistical characterization. Physics based models are also needed for other than MESFET devices, including HEMT and HBT.

Further research and development is also needed on multi-level simulation, optimization, yield optimization, and other aspects of general purpose CAD software. With the rapid progress in computer hardware, in particular power and speed of modern workstations, certain techniques previously considered too slow or too academic for industrial applications, for example exact field simulation of microstrip structures, are now gaining wide acceptance. Integration of such techniques into circuit-level simulation and optimization has already been experimented with by OSA. Such pioneering work is urgently needed.

Also, market response to the product and customer feedback will play an important role in future developments.

9. Expected Benefits

It is expected that this project will substantially contribute to profitability and development of OSA. Consequently, the growth of the company will generate jobs in Canada. Also, the algorithms and software modules developed for this project will be extremely valuable in future products, thus reducing their development cost.

The marketing of the product developed within the framework of this project should contribute to Canadian competitiveness in the microwave CAE arena and in the high-tech industry in general, thus far dominated by the Americans.

10. Technical Description of the Product

Summary

Our work on the statistical design centering and analysis capabilities of OSA90/hope™ has concentrated on the following features, originally planned for the final product.

1. Handling window specifications for any combination of DC, small-signal and large-signal steady-state circuit responses.

2. Handling a number of predefined statistical distributions including uniform, normal, exponential and lognormal distributions.

3. Handling user-defined statistical distributions including discrete distributions.

4. Handling user-supplied random number generators.

5. Handling multi-level hierarchical statistical models including wafer, chip and device statistics described by multidimensional normal distributions with correlations.

6. Handling multiple nonlinear devices and multi-tone excitations.

7. Handling inter-process communication with user's own simulators, parametric black-boxes of user's components, etc.

8. Graphical display of statistical responses (for multiple outcomes) together with the window specifications imposed on the responses.

9. User-friendly features including a menu-driven command system, flexible input file syntax, interactive graphics for displaying statistical responses, user-formatted output, etc.

10. State-of-the-art yield-oriented one-sided ℓ_1 optimization technique.

11. Numerical efficiency achieved by implementing advanced adjoint sensitivity analysis, state-of-the-art optimizers and efficient approximation to circuit responses.

All of these features, with the exception of (4), have been successfully implemented and are now available in OSA90/hope™. As we have implemented some of the best random number generators, we feel that Feature 4 would not be particularly advantageous to the user.

This section provides a brief technical description of the statistical design centering and yield optimization capabilities of OSA90/hope™ developed within this project, as well some other relevant statistical analysis features partially developed or enhanced in the course of this project. We begin by describing the significant feature of inter-process communication, called Datapipe, which allows the users to apply all statistical design capabilities of OSA90/hope™ to their own simulators, thus creating potential for further applications of this work.

Some features described in this section are available in OSA90/hope™ only, as they invoke the internal simulators of the system. This is the case whenever there is a reference to DC, small-signal AC, or harmonic balance (HB) simulation. OSA90™, as a mathematical engine, deals only with the abstract problems defined by expressions and/or external simulators connected through Datapipe. Nevertheless, we use the OSA90 name throughout this section.

Inter-Process Communication with User's Simulators

Traditional CAD programs are closed systems in the sense that all the features are built-in for a preconceived application. They offer little flexibility for customization or user-designed extensions. These aspects are now effectively addressed in OSA90.

OSA90's Datapipe utilizes UNIX's interprocess pipe communication facility to establish high speed data connection between OSA90 and one or more external programs. The input parameters to the external programs can be defined and preprocessed in OSA90. The outputs from the external programs can be postprocessed, displayed and optimized. The user can functionally integrate multiple external programs by making the outputs of one the inputs of another. The Datapipe schematic is illustrated in Fig. 1 where child programs refer to external programs which communicate with OSA90 via Datapipe connections. Each child includes a pipe server, which is a small set of functions supplied by OSA for reading data from and writing data to Datapipe.

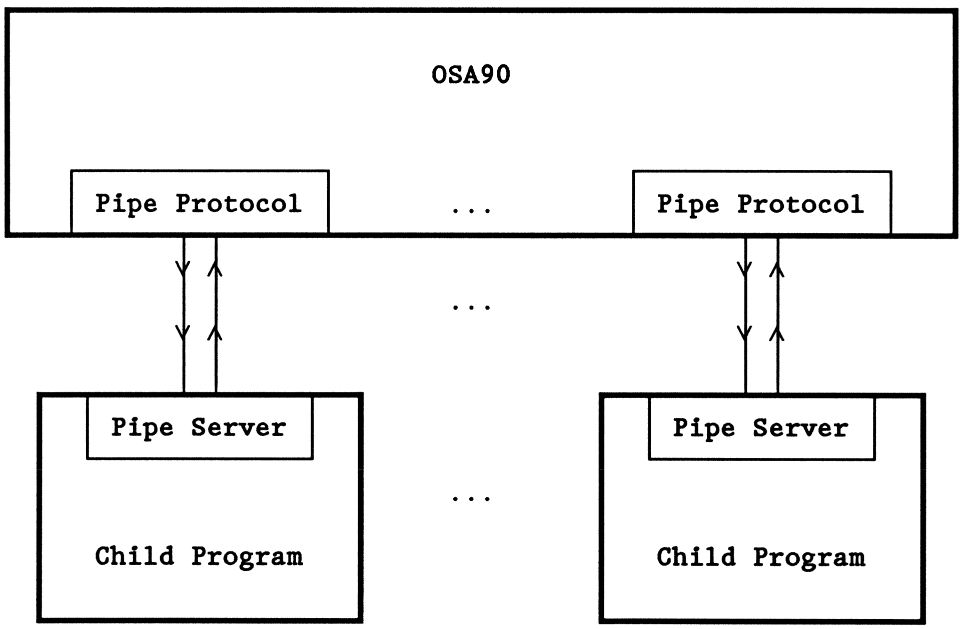

Fig. 1 Datapipe Schematic

When the outputs from a child program are needed during simulation or optimization, OSA90 forks a new process and requests the operating system to execute the child program in the new process. At the same time, a pair of interprocess communication pipes are established between OSA90 and the child process. Once the Datapipe connection is established, OSA90 and the child program can exchange data. OSA90 computes the inputs to the child program according to their definitions (variables, labels, etc.) and writes the data to Datapipe. The child program reads the inputs from Datapipe, carries out the necessary calculations, and sends back the outputs to OSA90.

OSA90 is equipped with a number of predefined protocols. In each specific case, the user selects an appropriate protocol and defines all specific details for Datapipe communication directly in the EXPRESSION or the MODEL block of the OSA90 input file. This is illustrated by the following example of the SIM protocol.

```
Datapipe:  SIM  FILE="sim_prog"
           N_INPUT=3  INPUT=(3.5, FREQ, ?10cm?)
           N_OUTPUT=2  OUTPUT=(Gain, Efficiency);
```

It defines a Datapipe using the SIM protocol to connect OSA90 to an external program named sim_prog. It indicates that sim_prog expects three input parameters, and specifies these parameters by a constant 3.5, the label FREQ and an optimization variable ?10cm?. The program sim_prog returns two values which will be stored as labels Gain and Efficiency. The names Gain and Efficiency must be unique (i.e., not used before). They represent two new labels created as the outputs of sim_prog (via Datapipe). After the Datapipe statement these two labels can be used just like any other labels, including expressions (postprocessing).

Menu Options

OSA90 communicates with the user by means of an input file and menu options. The input file contains all predefined information about a specific problem the user wants to solve and can be created either outside of OSA90 or using its built-in full screen editor. Operation of the program is then controlled by menu options shown in Fig. 2 and described in Table 1.

TABLE 1 OSA90 MENU OPTIONS

Menu Option	Function
OSA90.File	reads, edits, parses and saves files
OSA90.Display	calculates and displays responses
OSA90.Optimize	performs nominal or yield optimization
OSA90.Macro	switches control of operation to a macro file
OSA90.Sensitivity	calculates and displays sensitivities
OSA90.MonteCarlo	performs statistical (Monte Carlo) analysis
OSA90.Learn	creates a macro command file
OSA90.Display.Xsweep	displays responses versus parameter sweeps
OSA90.Display.Parametric	displays parametric plots of responses
OSA90.Display.Array	displays elements of arrays
OSA90.Display.Waveform	displays time-domain waveforms
OSA90.Display.Smith	displays Smith Chart and polar plots
OSA90.MonteCarlo.Xsweep	displays statistical sweep responses
OSA90.MonteCarlo.Parametric	displays statistical parametric plots
OSA90.MonteCarlo.Histogram	plots histograms of individual responses
OSA90.MonteCarlo.RunChart	plots run charts of individual responses
OSA90.MonteCarlo.Yield	displays yield from Monte Carlo analysis
OSA90.MonteCarlo.Sensitivity	displays partial yield versus parameter sweeps

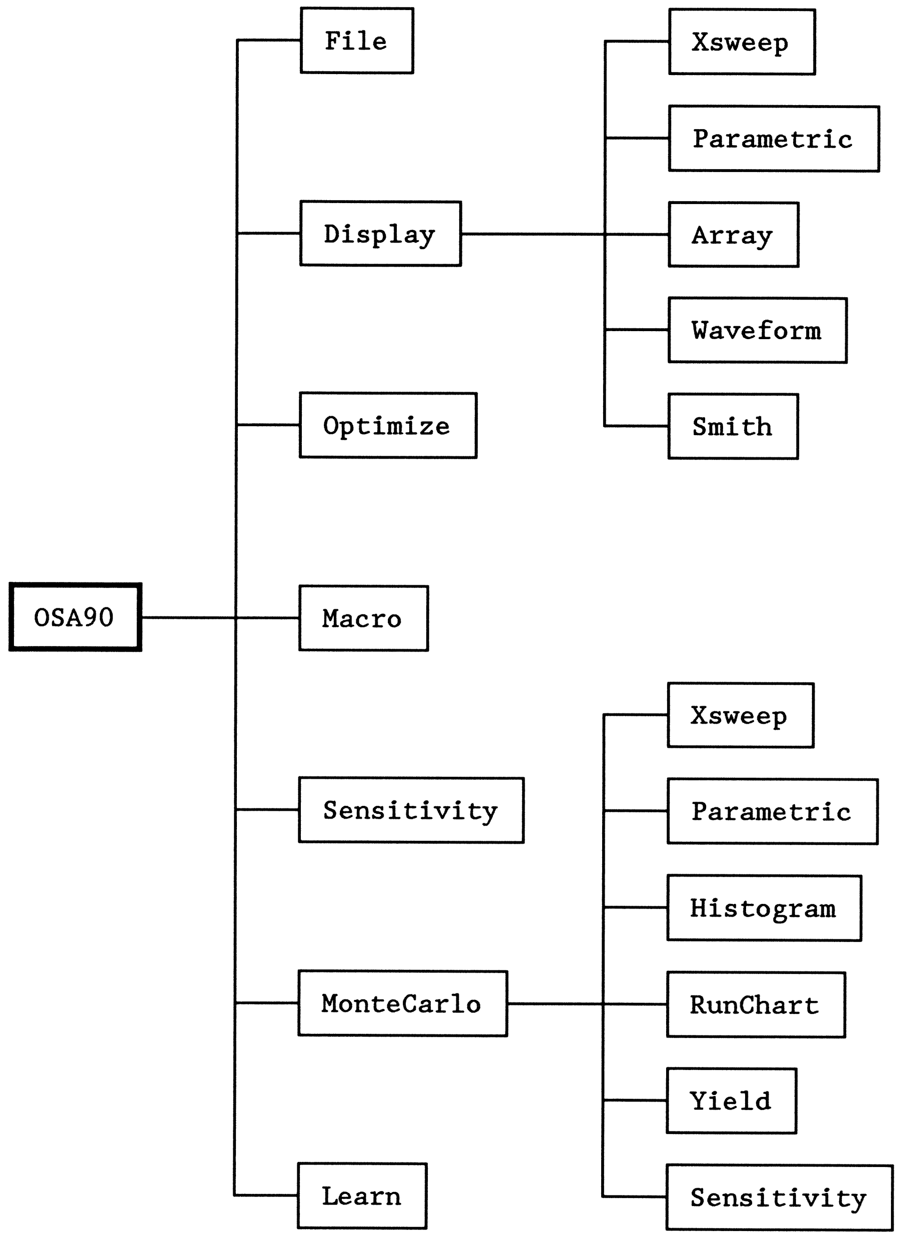

Fig. 2 OSA90 menu overview.

These menu options are presented to the user during interactive operation of the program. Selecting a menu option invokes the corresponding actions. If necessary, a pop-up window appears on the screen to allow the user to select certain specific features or set some parameters.

Our work for this project has concentrated on the parts of OSA90 corresponding to the following menu options: `OSA90.File` (input file parser), `OSA90.Optimize` (yield optimization), and `OSA90.MonteCarlo` (different displays).

Defining Specifications for Statistical Analysis

Statistical analysis, yield estimation and various graphical displays of statistical responses are defined by the user in the MONTECARLO block of the OSA90 input file. The syntax for the MONTECARLO block is:

```
MonteCarlo

    sweep_set  N_Outcomes=n;
    sweep_set;
    ...
    sweep_set;

End
```

where N_Outcomes defined with the first sweep set tells the program how many statistical outcomes are to be considered in statistical analysis; the same number is applicable to all sweep sets. Each statistical outcome is generated by perturbing all statistical parameters according to their distributions.

In each sweep set the user can define up to 2 sweep labels (parameters that are swept in a deterministic manner within user defined range) and 16 dependent labels (parameters that are not independently swept but their values may functionally depend on sweep labels), up to 128 responses of interest (output labels), the simulation type (must be consistent with the requested responses), and - if yield is to be evaluated - the goals which together with the responses define design specifications. For example, the following MONTECARLO block

```
MonteCarlo
    AC: FREQ: from 2GHZ to 18GHZ step=1GHZ
        MS21 > 2.2, MS21 < 2.6
        N_Outcomes=200;

    HB: Signal_Power: -10dBm -5dBm 0dBm
        FREQ: from 5GHz to 12GHz step=1GHz
        Omega=(2.0 * PI * FREQ)
        Bias_Voltage=0.75V
        Pout_dBm[2],  Efficiency > 0.75;
End
```

defines two sweep sets: the first one for small-signal simulation (AC), and the second for harmonic balance large-signal simulation (HB). Both simulations are to be performed for 200 statistically generated outcomes. In the first sweep set there is one sweep label FREQ (frequency) and one output label MS21 (the magnitude of S_{21}) which together with two goals defines two specifications for yield estimation. The second sweep set contains two sweep labels: Signal_Power and FREQ, two dependent labels Omega and Bias_Voltage, and two output labels Pout_dbm[2] (with no goal), and Efficiency which together with the goal "> 0.75" defines the third specification for yield. Since the output label Pout_dbm[2] has no goal it does not contribute to yield, but it will be simulated and available for display.

In the case of two-tone excitations, FREQ specifies the first tone frequency, and the second tone frequency is specified by the keyword FREQ2. For example, if

```
HB:  FREQ=900MHZ  FREQ2=907MHZ  ...;
```

then the two tone frequencies are specified as 900MHz and 907MHz. The OSA90 syntax allows for synchronized sweep of both tones, a feature particularly useful in mixer analysis. For example,

```
FREQ: from 1GHZ to 15GHZ step=2GHZ
FREQ2 = (FREQ + IM_Freq)
```

where IM_Freq is a user-defined label representing the difference between the two tone frequencies.

In order to estimate yield by Monte Carlo analysis the user must define at least one specification in the MONTECARLO block. He/she can define upper specifications, such as

```
output_label < goal
```

and lower specifications, such as

```
output_label > goal
```

respectively. The yield estimated by Monte Carlo analysis is given by

```
Yield = N_Pass / N_Outcomes
```

where N_Pass represents the number of "passed" outcomes, and N_Outcomes is the total number of outcomes. A "passed" outcome means that for that outcome all the responses (output labels) which have associated goals, from all the sweep sets in the MONTECARLO block, satisfy the specifications at all the sweep points. If any of the output labels associated with a goal from any of the sweep sets violates the specification at any of the sweep points, then that outcome is considered failed. Equality specifications are not allowed because such exact equality specifications will almost always lead to a zero yield which does not provide any useful information. Any single specification in nominal design should be replaced by a pair of upper and lower specifications to allow a "window" appropriate for the response.

Defining Specifications for Yield Optimization

Specifications for yield optimization, as for nominal optimization (performance-driven design), are defined by the user in the SPECIFICATION block of the OSA90 input file. The syntax for the SPECIFICATION block is:

```
Specification

   specification_set;
   specification_set;
   ...
   specification_set;

End
```

where in each specification set the user can define up to 2 sweep labels (parameters that are swept in a deterministic manner within user defined ranges) and 16 dependent labels (parameters that are not independently swept but their values may functionally depend on sweep labels), up to 128 responses of interest (output labels) which together with the goals define design specifications, and the simulation type (must be consistent with the requested responses). For example, the following SPECIFICATION block

```
Specification
   AC: FREQ: from 2GHZ to 18GHZ step=1GHZ
       MS21 > 2.2, MS21 < 2.6;

   HB: Signal_Power: -10dBm -5dBm 0dBm
       FREQ: from 5GHz to 12GHz step=1GHz
       Omega=(2.0 * PI * FREQ)
       Bias_Voltage=0.75V
       Pout_dBm[2] < -30,  Efficiency > 0.75;
End
```

defines two specification sets: the first one for small-signal operation (AC) and the second for harmonic balance large-signal operation (HB). In the first specification set there is one sweep label FREQ (frequency) and one output label MS21 (the magnitude of S_{21}) which together with two goals defines the first two specifications for yield estimation. The second set contains two sweep labels Signal_Power and FREQ, two dependent labels Omega and Bias_Voltage, and two output labels Pout_dbm[2] and Efficiency which together with the corresponding goals "< -30" and "> 0.75", define the third and fourth window specifications. The equality specifications, though allowed in the SPECIFICATION block, should not be used for yield optimization.

The meaning and use of different definitions in the SPECIFICATION block are similar to those for the MONTECARLO block, as already discussed. In particular, the syntax allows for two-tone harmonic balance simulation, including synchronized sweep of both tone frequencies. Additionally, the SPECIFICATION block allows the user to assign weighting factors to individual specifications, thus emphasizing the importance of some specifications while downplaying the others. For example

```
Specification
   AC: ...  MS21_dB > 7  MS21_dB < 9  MS11 < 0.1 W=10;
   ...
End
```

assigns a weight of 10 for the specification on MS11, while the specifications on MS21 are not weighted (equivalent to the weight of 1).

Each specification set can include output labels (responses) of only one simulation type. The simulation type determines the simulator involved in the calculation of the output labels and is defined by one of the keywords: DC - for DC analysis, AC - for small-signal AC analysis, HB - for large-signal harmonic balance analysis, or Datapipe - if the error functions are calculated by an external program connected to OSA90 via Datapipe. If no keyword is specified then the default simulation type is abstract, that is neither circuit simulators nor Datapipe are involved in the evaluation of the responses.

Different specification sets can be of different simulation type. This facilitates one of the most important features of OSA90: it allows us to simultaneously handle specifications defined in different domains or involving more than one simulator.

The simulation type Datapipe is used for specifications which contain error functions calculated by external programs and directly supplied for optimization. Such external programs have to be connected to OSA90 via Datapipe using the FUN protocol (functions only) or the FDF protocol (functions and derivatives). In order to utilize this feature, first FUN or FDF Datapipe is defined in the MODEL or EXPRESSION block, for example, as

```
Expression
   ...
   Datapipe:  Fxx  FILE="sim_errf"
              N_INPUT=4   INPUT=(X1, X2, 2.0, FREQ)
              N_OUTPUT=6  NAME=error_set1;
   ...
End
```

where Fxx denotes FUN or FDF, and then it is referred to in the SPECIFICATION block as

```
Specification
   Datapipe: FREQ: from 5 to 20 step=2
             error_set1 W=10;
   ...
End
```

where error_set1 identifies the set of 6 error functions to be calculated by the child program sim_errf, as shown in the Datapipe definition. The weighting factor defined as 10 is applied to all 6 error functions. For yield optimization it is important to define the error functions such that they become negative for acceptable designs (window specification).

Defining Optimization Variables

There is no distinction between optimization variables for nominal design (performance driven) and optimization variables for yield optimization. Optimization variables, are defined in the MODEL and/or EXPRESSION block of the OSA90 input file by means of user-defined labels or circuit element parameters with values enclosed in the pair of question marks, as

```
label_name: ?initial value?;
```

for example,

```
X1: ?2.5?;
```

or

```
RES 1 2 R = ?10.0?;
```

In addition to the initial value which is required, optional bounds can be assigned to an optimization variable, as

```
label_name: ?lower_bound  initial value  upper_bound?;
```

This forces the value of the variable to remain within lower_bound and upper_bound during optimization. Relative bounds can also be specified as

```
label_name: ?lower%  initial value  upper%?;
```

The effective bounds are then calculated by the file parser as

```
lower_bound = initial_value - ABS(lower * initial_value) / 100

upper_bound = initial_value + ABS(upper * initial_value) / 100
```

After optimization is finished, each optimization variable as well as the corresponding text in the input file, is automatically updated with the optimized value.

Defining Statistical Parameters and Built-In Distributions

Statistical parameters are defined in the MODEL and EXPRESSION blocks of the OSA90 input file as

```
label_name = nominal { distribution keyword=value ... keyword=value }
```

where label_name represents a keyword or a label, and nominal is the nominal (mean) value. Enclosed in a pair of curly braces is the definition of the distribution type, tolerance, correlation, etc., using the keywords listed in Table 2.

TABLE 2 STATISTICAL PARAMETER KEYWORDS

Distribution Type	Keywords
UNIFORM	TOL, HIGH, LOW
NORMAL	SIGMA, HIGH, LOW, CORRELATION, DDF
EXPONENT	TOL, HIGH, LOW
LOGNORMAL	TOL, HIGH, LOW

For example, in the following definitions

```
V1: 50 {Uniform TOL=10};

X1: ?2.5pF? {Normal SIGMA=5%};
```

the label V1 represents a statistical parameter with a constant nominal value of 50 and a uniform distribution with a tolerance of 10 and the label X1 represents another statistical parameter with normal distribution. The mean value of X1 is defined as an optimization variable with an initial value of 2.5pF. The standard deviation of X1 is defined as 5 percent of its mean value.

In addition to labels, statistical parameters can also be defined in circuit element models. For example:

```
SRC  1 2  R=50 {Uniform TOL=10}  C=?2.5pF? {Normal SIGMA=5%};
```

During Monte Carlo analysis and yield optimization, statistically perturbed values are generated for each statistical parameter defined in the input file. The process of generating the statistically perturbed values is described as follows. First, OSA90 generates *normalized deviates* (statistical perturbations) according to the distribution type specified (UNIFORM, NORMAL, etc.). These normalized deviates have a zero mean value and a unit tolerance. If truncation limits are specified using the LOW and HIGH keywords, the normalized deviates are truncated to be within the interval [LOW, HIGH]. These are called the *truncated deviates*. Then, the deviates are scaled by the value specified for the TOL or SIGMA keyword. They are called *scaled deviates*. Finally, the nominal value is taken into account to produce an *outcome*, as

```
outcome = nominal + scaled_deviate
```

for an absolute tolerance, or

```
outcome = nominal × (1 + scaled_deviate)
```

for a relative (percentage) tolerance, i.e., if the percent sign follows the TOL or SIGMA keyword.

A discrete density function (DDF) approximation is provided to represent non-Gaussian distributions. The random deviates are generated first from a normal distribution and then they are mapped into a discrete distribution through a histogram. The histogram serves as a discrete density function approximation of a non-Gaussian distribution.

To obtain the discrete density function (DDF) from the histogram of a statistical sample, the sample outcomes are divided into *N* bins (subintervals), where *N* is typically between 10 and 20. The number of outcomes which fall into each bin is recorded, and the numeric record is used as the DDF. For example, the following defines a non-standard distribution.

```
V1:  5  {Normal SIGMA=10%  DDF=2 8 15 16 18 18 12 4 2 4};
```

The syntax for statistical parameters, as well as the internal structure of OSA90, allow the user to define sophisticated hierarchical (multi-level) distributions. For instance,

```
V_Low_Level:  5  {Normal SIGMA=10%};

V_High_Level:  V_Low_Level  {Normal SIGMA=15%};
```

This is a two-level hierarchical distribution. The nominal of V_High_Level is itself a statistical parameter with the normal distribution specified by V_Low_Level. V_Low_Level may represent, for example, a "global" statistical uncertainty, and V_High_Level may model a "local" uncertainty on top of the global uncertainty. This feature is particularly useful in modeling chip, wafer and device statistics.

In addition to one-dimensional distributions, the user can also define multi-dimensional distributions by means of defining correlations between statistical parameters. Applicable to the NORMAL distributions only (but including DDF, if desired), parameter correlations can be defined through the use of the STATISTICS block in the input file and the keyword CORRELATION for the parameter statistics. The STATISTICS block in the circuit file contains correlation matrices. It is structured as

```
Statistics

   correlation_matrix
   ...
   correlation_matrix

End
```

Each correlation matrix defines the correlations among a group of statistical variables. It must have the header

```
CORRELATION: name  DIMENSION = n  [FORMAT = format];
```

where name is an arbitrary character string which uniquely identifies the correlation matrix. The DIMENSION keyword must be followed by a positive integer which specifies the number of rows (columns) of the matrix. The dimension is equal to the number of statistical parameters whose correlations are defined by the matrix.

The elements in the correlation matrix are the correlation coefficients. The element at the position (i,j) is the correlation coefficient between the ith and the jth parameter. The value of the coefficients must be between -1 and +1. The matrix must be symmetrical, semi-positive definite and all the diagonal elements must be 1. The entries can be organized in one of the four available formats, namely, FULL, UPPER, LOWER and SPARSE. For example:

```
CORRELATION: Cor_Mat  Dimension=4  Format=FULL;
   1.0   0.2   0.4   0.6
   0.2   1.0   0.8   0.0
   0.4   0.8   1.0   0.1
   0.6   0.0   0.1   1.0
```

The CORRELATION keyword is used to link a set of statistical parameters to a correlation matrix defined in the STATISTICS block. For example, the following two statements

```
V1:  5  {Normal SIGMA=10%  CORRELATION=Cor_Mat[2]};
V2:  2  {Normal SIGMA=5%   CORRELATION=Cor_Mat[3]};
```

indicate that the correlation coefficients of V1 and V2 are given by the 2nd and 3rd rows (or columns), respectively, of the correlation matrix defined in the STATISTICS block and identified by the string name Cor_Mat. The correlation coefficient between V1 and V2, for instance, is the element at position (2,3).

Graphical Displays of Statistical Analysis Results

The menu option OSA90.MonteCarlo is designed to manage statistical simulation and the resulting displays. The available options were summarized in Table 1.

The menu option OSA90.MonteCarlo.Xsweep displays the statistical response of one output label versus a sweep parameter. Each curve on the display represents the value of the output label versus the sweep parameter for one random outcome. The total number of curves that are displayed is equal to the number of outcomes specified in the MONTECARLO block, as illustrated in Fig. 3.

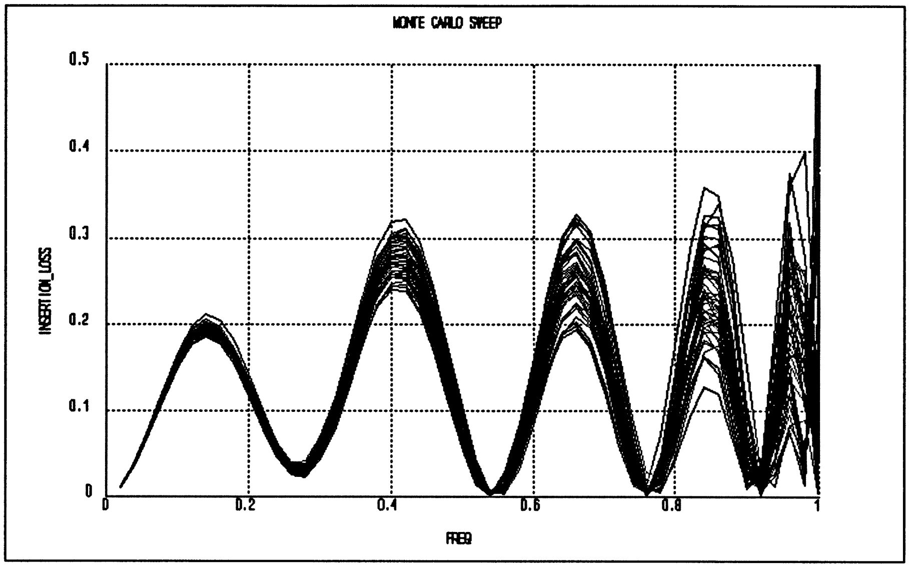

Fig. 3 Monte Carlo Xsweep Display.

The parametric display featured by the menu option OSA90.MonteCarlo.Parametric plots one output label versus another output label for all the statistical outcomes.

The menu option OSA90.MonteCarlo.Histogram displays the histograms of individual output labels at a selected sweep point. A histogram divides the values of an output label into a number of evenly spaced intervals (bins). The number of values (outcomes) which fall into each bin is counted, and the result is displayed as a bar chart, as illustrated in Fig. 4.

In addition to graphical display, the menu options OSA90.MonteCarlo.Xsweep and OSA90.MonteCarlo.Histogram can display the statistical responses numerically. The values of the sweep label and output labels are printed in ASCII text format, and the result is loaded into OSA90's file editor. The program automatically switches into the editing mode, which allows the user to view, edit and save the numerical outputs.

The menu option OSA90.MonteCarlo.RunChart displays the run charts of individual output labels at a selected sweep point. A run chart plots the value of an output label versus the outcome index. The X-axis is the outcome index ranging from 1 to *n*, where *n* is the number of outcomes. The Y-axis represents the output label, as illustrated in Fig. 5.

By default, graphics displays are scaled to the range of the output values. The user can change the display scale (zooming) using the "Zoom scale" option.

The menu option OSA90.MonteCarlo.Yield prints yield numerically on the screen. Yield is estimated and printed for every 10 outcome interval. The percentage of passed outcomes from the first 10 outcomes is calculated and printed, followed by the percentage from the first 20 outcomes, and so on. Only those output labels for which goals are specified are taken into account in the yield calculation.

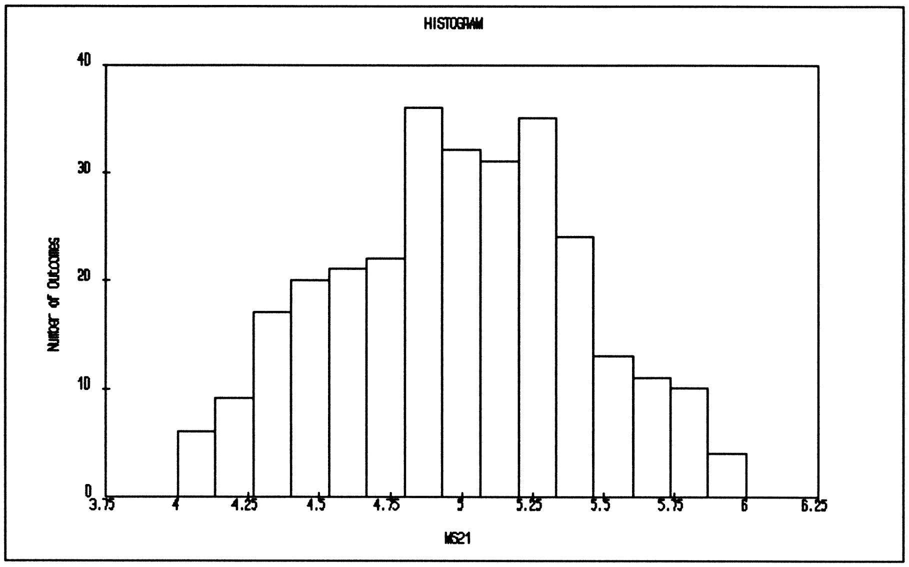

Fig. 4 Histogram.

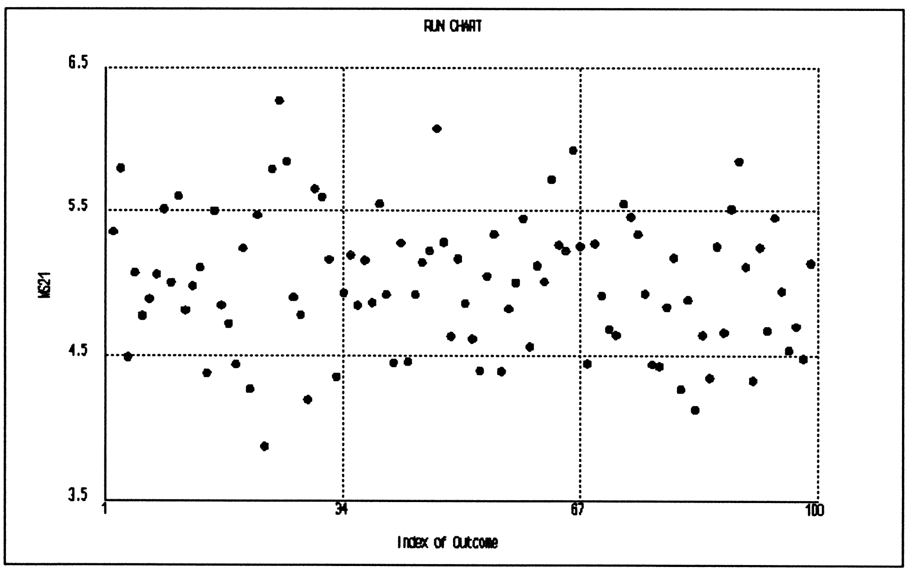

Fig. 5 Run chart.

Monte Carlo analysis can be time-consuming. To avoid duplicated analyses, a feature is built into OSA90 to save the results in an encoded disk file, which can be retrieved and displayed in a subsequent session. The encoded file is named after the input file by replacing the extension ".ckt" with ".mca".

Yield Sensitivity Analysis

A unique feature of yield sensitivity is available in OSA90 through the menu option OSA90.MonteCarlo.Sensitivity. It displays the partial yield versus a sweep parameter. For this option to be available, the MONTECARLO block must contain at least one sweep label and at least one specification. The values of the output labels from the Monte Carlo analysis are separated into groups, each group corresponding to one of the values of the sweep label. A partial yield is calculated from each group, and the result is plotted versus the sweep label. For example if the MONTECARLO block is defined as

```
MonteCarlo
   AC: FREQ: 1 25 50 75 100
       MS21 > 2.8  MS21 < 3.2  MS11 < 0.4  MS22 < 0.4
       N_Outcomes=500;
End
```

the corresponding display of the partial yield versus the sweep label FREQ would be as shown in Fig. 6. It clearly shows that FREQ=1 corresponds to the lowest partial yield. Since the overall yield cannot exceed the lowest partial yield, this means that FREQ=1 is the most critical value in terms of its impact on the yield.

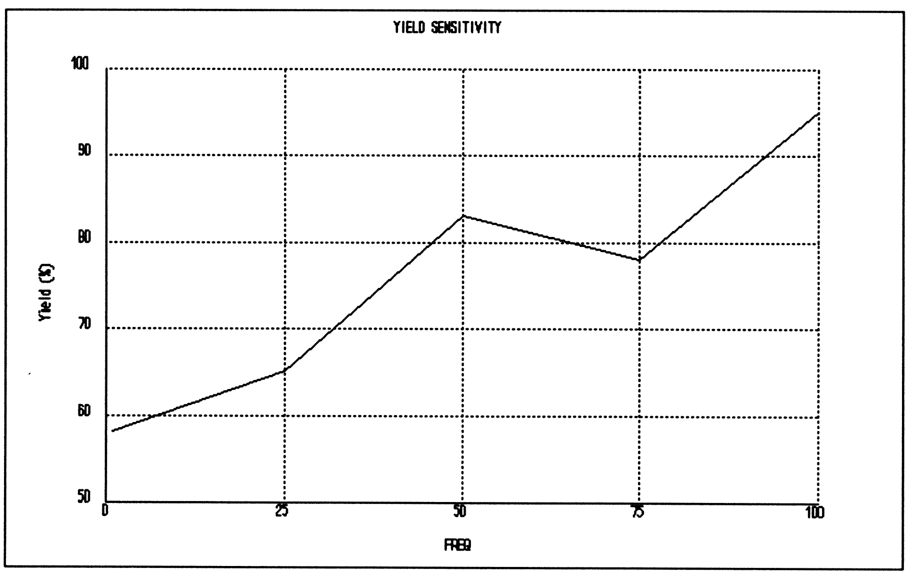

Fig. 6 Partial yield versus FREQ.

The yield sensitivity feature can be used to investigate the sensitivity of the overall yield with respect to design variables. This can be achieved by adding a design variable, say X, to the MONTECARLO block. For the preceding example it would lead to

```
MonteCarlo
   AC: X: from 20 to 28 step=0.5
       FREQ: 1 25 50 75 100
       MS21 > 2.8  MS21 < 3.2  MS11 < 0.4  MS22 < 0.4
       N_Outcomes=500;
End
```

and then the "partial" yield for each value of X is actually equivalent to the "total" yield in the preceding example. In other words, displaying the "partial" yield versus X is equivalent to displaying the yield defined in the preceding example versus the design variable X. The display is illustrated in Fig. 7.

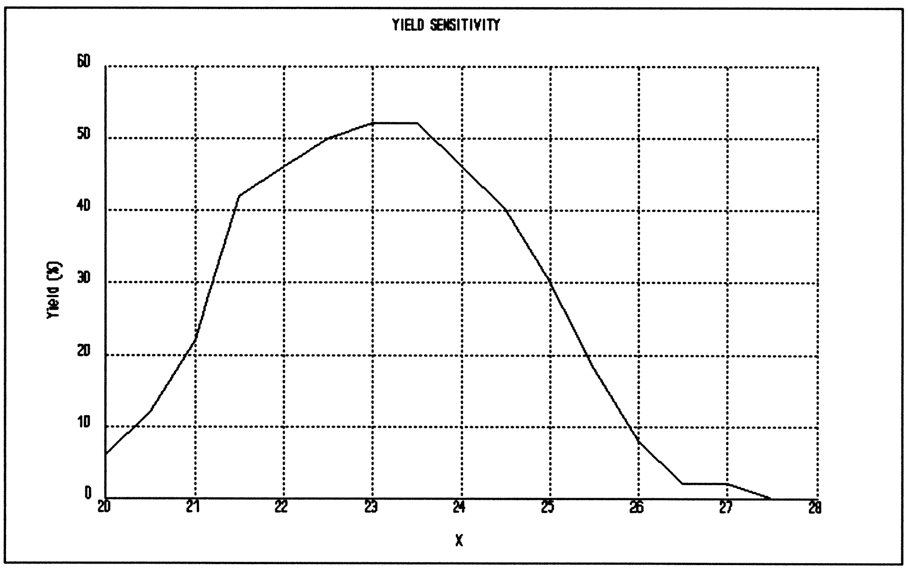

Fig. 7 Yield versus the design variable X.

Yield Optimization

Yield estimated by Monte Carlo simulation is a discrete function (discontinuous with respect to the optimization variables) and thus unsuitable for gradient-based optimization. To overcome this difficulty, OSA developed a sophisticated generalized ℓ_1 centering approach. The generalized ℓ_1 objective function is formulated from the error functions calculated at a number of random outcomes generated according to the statistical parameter distributions. In this way discrete yield is approximated by a continuous objective function suitable for gradient-based optimization. A dedicated gradient-based one-sided ℓ_1 optimizer is specially developed for the minimization of this objective function. A set of weighting factors is devised in such way that optimization leads to a centered solution and enhanced yield.

Yield optimization is invoked through the menu option OSA90.Optimize by selecting the `Yield` optimizer in the pop-up window. The pop-up window looks like:

```
Optimizer:                Yield
objective Function:       Generalized L1
Number of iterations:     30
Accuracy of solution:     0.0001
Display option:           every iteration
Statistical outcomes:     20
Q-gradient:               Yes
Ready to go:

<ENTER> = go  or  <ESC> = cancel
Select item with <UP>/<DOWN>
<HELP> help
```

The pop-up window allows the user to interactively select different options for yield optimization, such as the number of statistical outcomes, maximum number of iterations, desired accuracy, or enable quadratic modeling, etc. Before invoking yield optimization, however, the user has to set up the problem in the OSA90 input file by defining

1. statistical parameters in the MODEL or EXPRESSION block,
2. optimization variables in the MODEL or EXPRESSION block, and
3. the simulation types, sweep ranges and specifications in the SPECIFICATION block,

as outlined in the preceding subsections.

The yield optimization process starts after selecting the Ready to go: item in the pop-up window. During yield optimization, OSA90 displays the iteration count, the current yield estimate and the value of the generalized ℓ_1 objective function. For example:

```
Iteration   1/30  Yield= 60%  Objective=8
Iteration   2/30  Yield= 65%  Objective=5.82925
Iteration   3/30  Yield= 65%  Objective=5.6665
Iteration   4/30  Yield= 70%  Objective=4.89755
...
```

Of the iteration count, the first number is the current iteration and the second one is the maximum number of iterations selected in the pop-up window. For example, 12/30 means that the current iteration is the 12th out of a maximum of 30. The yield is estimated by

$$\text{Yield} = N_{pass} / N$$

where N is the number of outcomes selected in the pop-up window, and N_{pass} is the number of "passed" outcomes.

The choices for the number of statistical outcomes are 10, 20, 50, 100, 200 and 500. The default is 20 outcomes. The accuracy of yield estimated during yield optimization depends on the selected number of outcomes. A larger number of outcomes will better represent the statistical parameter distributions and lead to a more accurate estimate of yield, but it will also demand longer computer time. Typically, for a new problem, the user may start with a choice of 20 or 50 outcomes. Then, at the solution, he/she can increase the number of outcomes (e.g., to 100 or 200 outcomes) and restart yield optimization in order to obtain a more accurate result.

During yield optimization, a bar graphics indicator of the progress is displayed in the upper left-hand corner of the screen, as well as the following message:

```
Optimization ...  Press any key to interrupt
```

While optimization is in progress, the user can interrupt it at any time by pressing any key on the keyboard. Then, the following prompt is displayed:

```
Terminate Operation (Y/<N>)?
```

and the user can decide whether to terminate or to continue. In addition to user interruption, yield optimization is terminated under one of the following conditions.

1. A local minimum of the generalized ℓ_1 objective function is found, i.e., the value of the objective function cannot be further reduced in the vicinity of the solution.

2. The specified accuracy is reached, i.e., the step size separating the current point and the next point suggested by the algorithm is smaller than the requested accuracy. The step size is relative to the norm of the variable vector.

3. The maximum number of optimization iterations is reached.

4. A severe error condition is encountered such as a floating-point divide by zero error.

A message indicating the cause of termination is displayed. Regardless of the cause, the solution presented to the user is always the best set of values of the variables obtained up to the point of termination. The input file is automatically updated with the solution, i.e., the variables are updated with their optimized values.

A unique quadratic modeling technique is offered for yield optimization. It constructs quadratic models of the error functions and derivatives. During yield optimization, the quadratic model is utilized to generate the error functions and derivatives for the Monte Carlo outcomes. For a large number of outcomes, this can lead to a substantial reduction in computation time in comparison with exact analyses.

The `Q-gradient` item in the pop-up window allows the user to choose between the `Yes` and `No` options, to enable or disable the use of quadratic models in yield optimization. If it is enabled, a quadratic model is constructed from the error functions and derivatives computed at a number of so called base points. The quadratic model is then used to generate error functions and derivatives for each statistical outcome, instead of exact calculations that would otherwise be required.

The main effort of the quadratic modeling technique is in constructing the model and it depends on the number of statistical parameters. Once the model is established, using it to generate error functions and derivatives takes relatively little effort. The computational time required by the quadratic modeling does not increase significantly with the number of outcomes. Therefore, a large number of outcomes (100 or 200) is recommended to be chosen in conjunction with the use of quadratic modeling.

It is recommended that the solution obtained using quadratic modeling be verified by Monte Carlo analysis. It is not unexpected to find yield estimated by Monte Carlo analysis different from yield predicted by the quadratic model, because the model is an approximation to the exact functions. If necessary, the user can restart yield optimization with the quadratic modeling feature disabled. Restarting yield optimization (i.e., after a solution is obtained) can also be advantageous in several other situations, e.g., if a relatively small number of outcomes is used or if the solution is far away from the starting point.

11. Impact of Support

OSA had the technical expertise to carry out this project. IRAP funding was crucial for our ability to devote the necessary effort, time and resources to research and development needed for this project, and thus it helped the project to be completed.

OSA is grateful to the National Research Council of Canada for supporting this project and making this development of the state-of-the-art yield optimization features possible.

12. References

[1] J.W. Bandler, R.M. Biernacki, Q. Cai, S.H. Chen, S. Ye and Q.J. Zhang, "Integrated physics-oriented statistical modeling, simulation and optimization," Special Issue of *IEEE Trans. Microwave Theory Tech.* on Process-Oriented Microwave CAD and Modeling, vol. 40, 1992, pp. 1374-1400.

[2] J.W. Bandler, S. Ye, Q. Cai, R.M. Biernacki and S.H. Chen, "Predictable yield-driven circuit optimization," *IEEE MTT-S Int. Microwave Symp. Dig.* (Albuquerque, NM), 1992, pp 837-840.

[3] J.W. Bandler, S.H. Chen and R.M. Biernacki, "A new formulation for yield optimization," *IEEE MTT-S Int. Microwave Symp. Dig.* (Albuquerque, NM), 1992, pp. 1465-1468.