™

OSA

DESIGN OPTIMIZATION

WITH EXTERNAL SIMULATORS

OSA-91-CM-9-V

December 20, 1991

Optimization Systems Associates Inc.

Dundas, Ontario, Canada

DESIGN OPTIMIZATION

WITH EXTERNAL SIMULATORS

OSA-91-CM-9-V

December 20, 1991

Presented at Eighth COMPUMAG Conference on the Computation of Electromagnetic Fields, Sorrento, Italy, July 1991

DESIGN OPTIMIZATION

WITH EXTERNAL SIMULATORS

J.W. Bandler, R.M. Biernacki and S.H. Chen

Optimization Systems Associates Inc.
P.O. Box 8083, Dundas, Ontario
Canada L9H 5E7

Outline

concepts of design optimization

design as an abstract optimization problem

error functions for design goals

combining error functions into a single objective function

a novel and powerful approach to CAD software architecture suitable for distributed calculations and interactions between independent programs

Design Optimization

enables designers to adjust designable parameters to meet design specifications

the nature of the designed object is irrelevant to the optimizer; computer simulation of the object must be available to process a number of input parameters into corresponding responses

it is extremely desirable that the simulator is capable of calculating partial derivatives (gradient) of the responses w.r.t. the designable parameters

the simulator should be efficient enough for repeated calculations

efficiency of the algorithms as well as organization of software are of utmost importance

software modularity must be facilitated and modules of different origin need to be accommodated

Specifications and Responses

the response functions of interest to the designer may involve a combination of frequency domain responses, time domain responses, space domain responses, frequency spectra of periodic functions, and their functions such as power, etc.

a specification is typically imposed on a range of domain values; this leads to an infinite number of specifications

it becomes necessary to discretize the domain and consider only a finite subset of representative frequency, time or space points

after discretization, the *j*th specification is

S_{uj} for upper specification
$S_{\ell j}$ for lower specification

the corresponding responses are denoted by R_j

Error Functions

ϕ denotes the vector of designable parameters

the individual error functions $e_j(\phi)$ are defined as

$$e_j(\phi) = R_j(\phi) - S_{uj} \qquad \text{for upper specification}$$

or

$$e_j(\phi) = S_{\ell j} - R_j(\phi) \qquad \text{for lower specification}$$

the error vector $\mathbf{e}(\phi)$ combines all error functions

$$\mathbf{e}(\phi) = [e_1(\phi) \;\; e_2(\phi) \; . \; . \; . \; e_M(\phi)]^T$$

where M is the total number of errors

The Acceptability Region

negative error values indicate that the corresponding specifications are satisfied

positive error values indicate that the corresponding specifications are violated

the acceptability region is defined in the parameter space as

$$A = \{\phi \mid e_j(\phi) < 0 \;\; j = 1, 2, ..., M\}$$

all specifications are satisfied if the designable parameters fall into A

at least one specification is violated if that point falls outside the acceptability region A

Objective Functions

all the errors $e_j(\phi)$ have to be combined into a single objective function

three important objective functions are ℓ_1, ℓ_2 (least squares) and minimax

the generalized ℓ_p function $v(\phi)$ from $\mathbf{e}(\phi)$ takes the form

$$v(\phi) = \begin{cases} \left[\sum_{j \in J(\phi)} (e_j(\phi))^p\right]^{1/p}, & \text{if } \phi \notin A, \\ -\left[\sum_{j=1}^{M} (-e_j(\phi))^{-p}\right]^{-1/p}, & \text{if } \phi \in A, \end{cases}$$

where

$$J(\phi) = \{j \mid e_j(\phi) \geq 0\}$$

Variations of ℓ_p Functions

(a) the one-sided ℓ_p function

(b) the ℓ_p norm

(c) the minimax function corresponding to $p \rightarrow \infty$, or

$$v(\phi) = \max_j (e_j(\phi))$$

Typical Applications of Different Objective Functions

minimax is the objective of choice for performance-driven design optimization; it leads to equi-ripple solutions

the ℓ_2 norm or one-sided ℓ_2 function are also commonly used for performance-driven design optimization

the ℓ_1 function is uniquely useful in modeling and in yield optimization

specialized, robust algorithms exist for minimization of each of the aforementioned objective functions

Open Architecture Optimization Software Systems

IPPC (inter-program pipe communication) - a recent, advanced technique for open software architecture facilitates high speed numerical interaction between independent programs

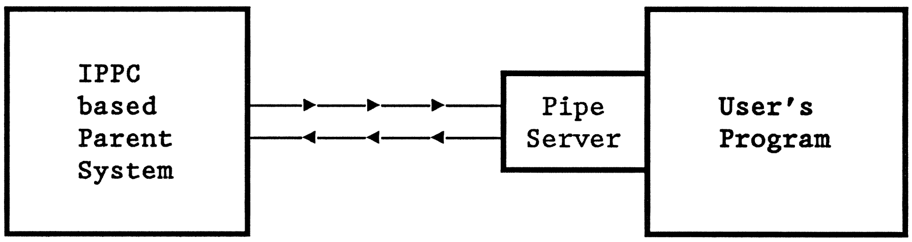
it allows highly repetitive data communication

it allows an unlimited number of non-predetermined and new software modules to be added to existing software systems with no modification, no re-compilation and no re-linking of the existing systems

the user can add new modules to an existing IPPC-based system, allowing the existing system's optimizers, statistical drivers, etc., to interact iteratively with his own modules

the user's modules are separate executable programs

Basic Form of the IPPC



in forking the child process, two inter-process pipes are created

Application Details of the IPPC

no modification to the IPPC-based parent program

only minor modification to the child program are needed: the user attaches the IPPC server to his or her program to generate a pipe-ready version

during simulation or optimization involving the child, the parent executes the child as a separate process

communication between one parent and several children and grandchildren is possible

experiments have been conducted on our new CAD system OSA90™

the overhead CPU cost in practical situations is found to be negligible - it typically adds only about 1% to the conventional approach of subroutine calls

2D Field MESFET Simulation

two-dimensional numerical model based on

Reiser, *"A two-dimensional numerical FET model for DC, AC, and large-signal analysis"* (1973), and

Snowden *et al., "Large-signal modeling of GaAs MESFET operation"* (1983)

drift-diffusion relation is used: the electron mobility is assumed as a function of electric field and the diffusion coefficient is determined by the Einstein relation

velocity-overshoot not considered

two-dimensional Poisson equation and current continuity equation are solved to simulate the internal device physics

finite difference technique used to solve two partial differential equations

Input File for 2D Field MESFET Simulation

```
! fld2.ckt
! field simulation
! 2D MESFET model through Datapipe

Expression

   k = 0;
```

define voltages

```
   ! bias sweep
   vd1 = 0.25;     ! drain voltage - lower value
   vd2 = 3.5;      ! drain voltage - upper value
   ivdp = 14;      ! total number of points
   vd = vd1 * k;   ! drain voltage - actual values
   vg = -3.0;      ! define label here

   vvv[1:4] = [vg, vd1, vd2, ivdp];
```

Input File for 2D Field MESFET Simulation

define physical parameters

```
T0 = 350;                 ! temperature
PSI_B = 0.8;              ! gate barrier potential
E0 = 4.E5;                ! critical electric field

Width = 3.0E-04;          ! gate width

S_length =  0.15E-6;      ! source length
G_S = 0.5E-06;            ! gate to source gap
G_length = 0.5E-06;       ! gate length
G_D = 0.60E-06;           ! gate to drain gap
D_length = 0.15E-6;       ! drain length

A_layer = 1.5E-07;        ! active layer
B_layer = 2.0E-07;        ! buffer layer
S_layer = 0.5E-7;         ! substrate layer

D_posi = 1.55E-6;         ! drain position
S_posi = 0.4E-6;          ! source position

N0 = 1.5E+23;        ! doping density
NS = 1.E19;          ! doping density at substrate
NC = 3.7E23;         ! doping density at drain and source contacts
NG = 1.0E10;         ! doping density at gate contact
```

Input File for 2D Field MESFET Simulation

define arrays for Datapipe communication

```
para[1:30] =  [G_length, S_length, D_length, G_S, G_D, A_layer,
               B_layer, S_layer, Width, N0, NS, NC, NG, T0,
               PSI_B, E0, D_posi, S_posi,
               0,0,0,0,0,0,0,0,0,0,0,0];

! controlling parameters
contr[1:30] =  [4, 0.2E-12, .005E-12, 0.5E-12, .01E-12,
                2.0E-12, .015E-12, 12.E-12,.02E-12,
                0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0];

! calculate the total number of input points
TOTAL_IN = 4 + 30 + 30;

! calculate the total number of output points
TOTAL_OUT = 20;
```

call *model* - user's 2D field-based FET simulator

```
    datapipe: SIM  FILE = "model"        ! model is a field based fet
                   N_INPUT = TOTAL_IN   ! simulation program
                   INPUT = (vvv, para, contr)
                   N_OUTPUT = TOTAL_OUT
                   OUTPUT = (ID[1:20]);
End
```

Input File for 2D Field MESFET Simulation

define simulation ranges and outputs

```
Sweep
  vg: from -3 to 0 step=1
  K: from 1 to ivdp step=1
  ID[K], vd
  TITLE = "Two-Dimensional MESFET Model: DC I-V Curves";
End
```

Simulation Details

56 DC bias points

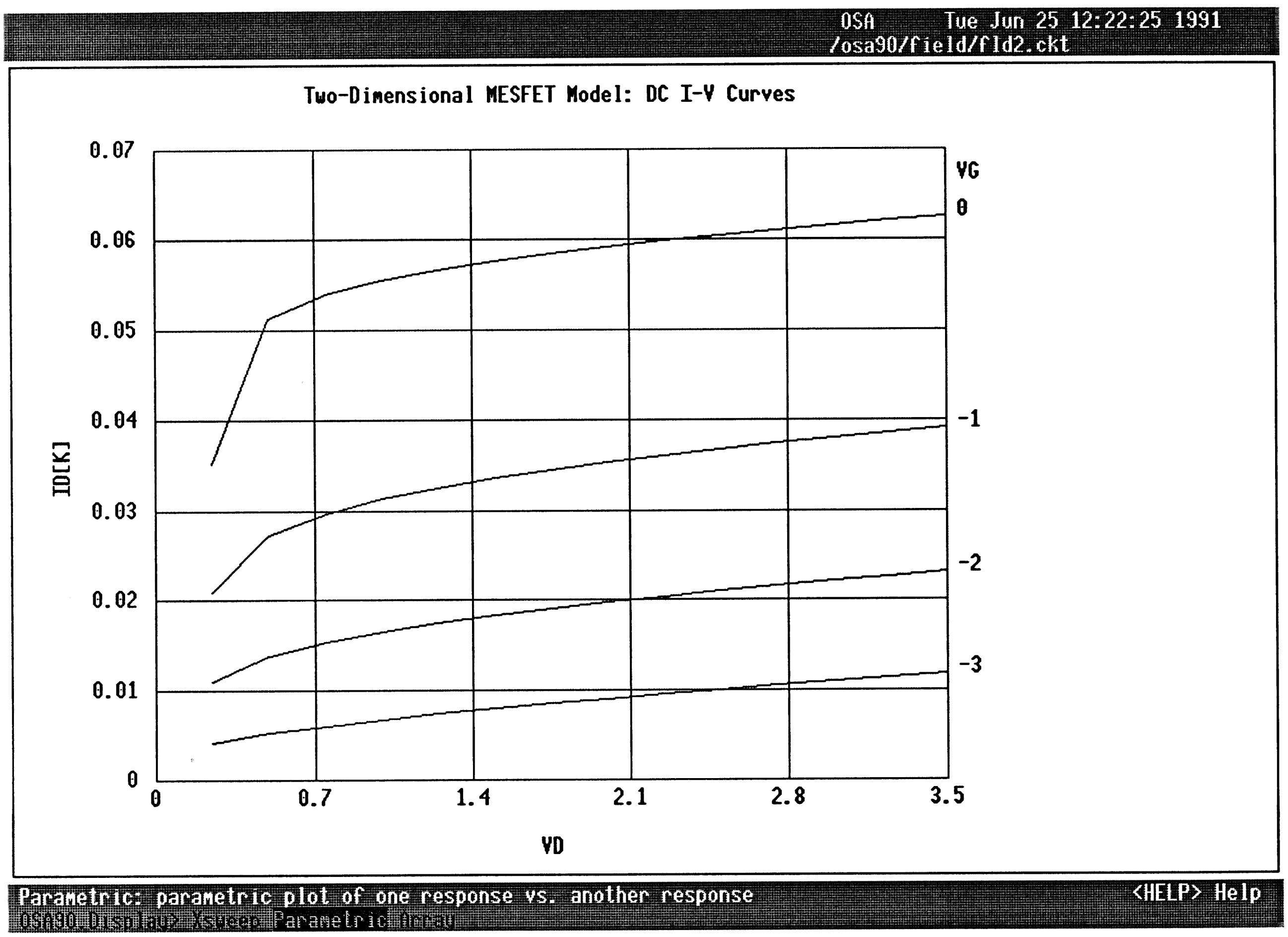
Sun SPARCstation 1

CPU time - 14 hours 10 minutes

16 minutes per bias point

Results of 2D Field MESFET Simulation

DC I-V characteristics: drain current vs. drain voltage for four different values of gate voltage



Matching Plessey Model to 2D Field MESFET Simulation

create an equivalent circuit model suitable for fast, repeated simulations to be used in place of the field simulator

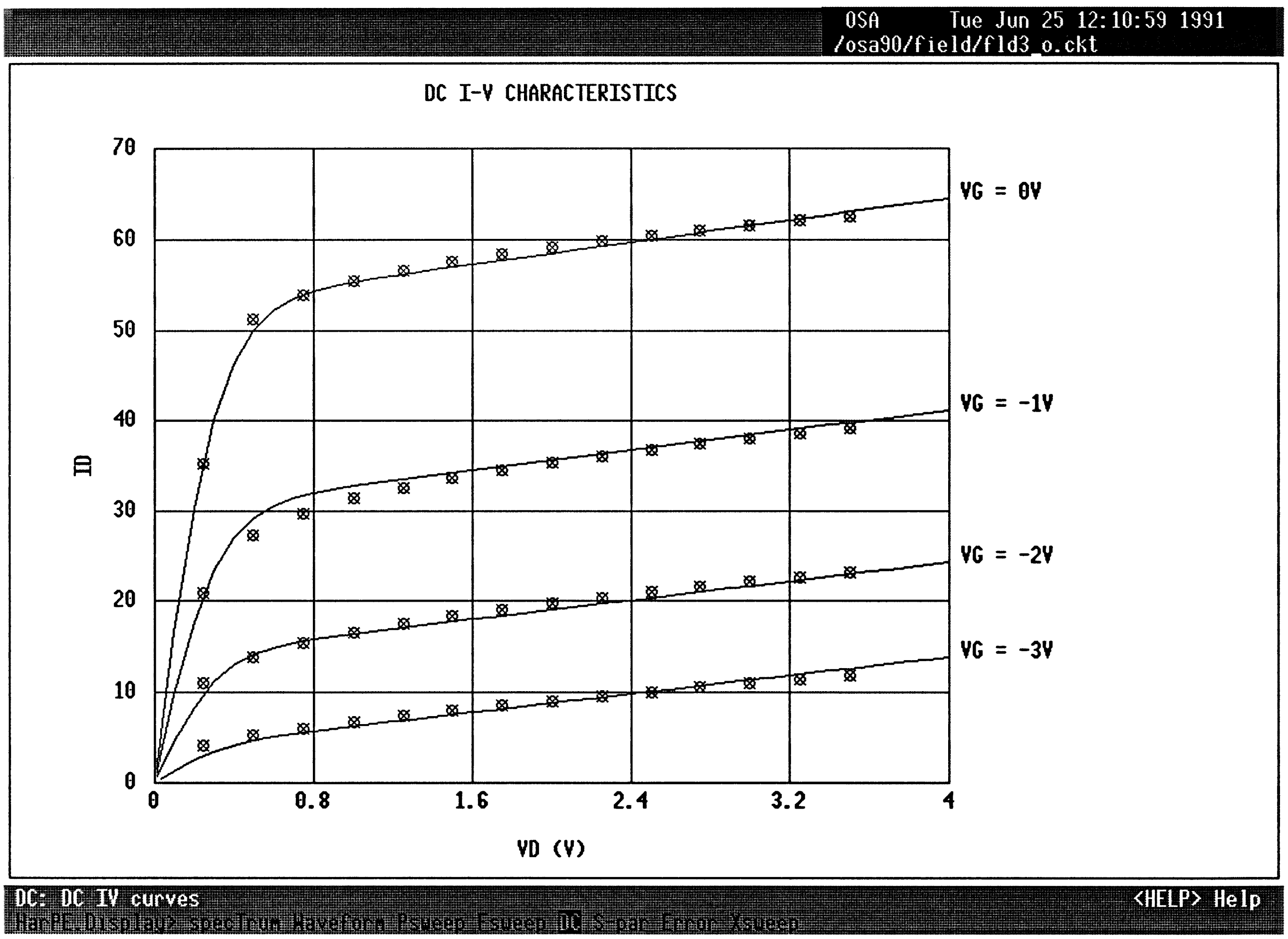
a recent research FET model from Plessey: modified Statz model

field-based simulated results are to be matched by the Plessey model

ℓ_1 optimization used to extract the model parameters

Simulation Results for the Plessey Model

DC I-V characteristics; drain current is swept with higher resolution than requested from the field simulator



points represent results from the field simulator

Gradient Evaluation

PAST (perturbation approximate sensitivity technique) to calculate Jacobians and gradients - very simple to implement but inaccurate and time-consuming

adjoint circuit technique - developed by several authors in the late 1960's and early 1970's - theoretically the most efficient method for sensitivity analysis

EAST (exact adjoint sensitivity technique) - for nonlinear steady state circuits and for hierarchical linear circuits (*Bandler, Zhang and Biernacki, 1988*)

FAST (feasible adjoint sensitivity technique) - expedient implementation combining the efficiency of EAST with the simplicity of PAST; particularly suitable for general purpose programs (*Bandler, Zhang and Biernacki, 1989*)

IGAT (integrated gradient approximation technique) - utilizes the Broyden formula with special iterations of Powell to update the approximate gradients (*Bandler, Chen, Daijavad and Madsen, 1986*)

Conclusions

design as an abstract optimization problem; objects of different nature can be designed

field theoretic analysis can provide simulation results for the optimizer

field theoretic analysis can provide simulated data for modeling in terms of physical parameters, e.g., to develop equivalent circuit models for fast repeated simulations

quadratic approximation, look-up table techniques, etc.

IPPC is particularly useful for independent software development, testing, and maintenance of large software systems

IPPC secures the confidentiality of both commercial optimizers and user's programs