SEMI-INFINITE OPTIMIZATION IN ENGINEERING DESIGN

by

E. Polak

Memorandum No. UCB/ERL M82/50

30 November 1981

SEMI-INFINITE OPTIMIZATION IN ENGINEERING DESIGN

by

E. POLAK

Memorandum No. UCB/ERL M82/50

30 November 1981

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

SEMI-INFINITE OPTIMIZATION IN ENGINEERING DESIGN

by

E. Polak

Department of Electrical Engineering
and Computer Sciences
and the Electronics Research Laboratory
University of California
Berkeley, Ca. 94720

ABSTRACT.
It is shown that a large class of engineering design problems are expressible as nonlinear semi-infinite programming problems. Methods for the solution of these SIP problems are surveyed.

1. INTRODUCTION.

Many industries are facing rapid growth in the cost of raw materials and energy. More and more frequently, competitive product design is becoming an ever more time-consuming and difficult task because design complexity is increasing more rapidly than the number of design variables. All these factors contribute to increasing costs of design and manufacture. Computing power is the only important industrial resource which is declining in cost. Not surprisingly, the engineer is turning more and more frequently to the computer as a way of coping with his tasks.

The growing importance of computer-based tools for the analysis and design of devices, structures, machines and systems has resulted in the evolution of a relatively new interdisciplinary engineering specialty which is commonly referred to as computer-aided design (CAD). Evolution of the more sophisticated forms of CAD capability requires a team effort involving engineering design specialists, computer scientists, optimization experts, and numerical analysts. CAD techniques developed in one area of engineering are often of substantial relevance to other areas.

Both combinatorial and parametric optimization are used in CAD. The most commonly occuring parametric optimization problems in engineering design are usually expressed either as differentiable or as non-differentiable semi-infinite programming (SIP) problems. In this paper we shall illustrate by example how typical SIP problems arise in engineering design and survey some of the more promising methods for their solution.

2. FORMULATION OF ENGINEERING DESIGN PROBLEMS IN SIP FORM.

In this section we shall give four examples of engineering design problems which can be solved in SIP form: one in structural design, one in single-input single-output control system design, one in multi-input multi-output control system design, and one in electronic circuit design.

2.1. Seismic Resistant Design of Structures.

One of the conceptually simplest examples of SIP in engineering design is found in the design of braced frame buildings which are expected to withstand small earthquakes with no damage and large ones with repairable damage. A simple braced frame is shown in Fig. 1a, where the components of the vector design parameter x are the stiffnesses of the members indicated. The horizontal floors are assumed to be rigid and to concentrate the mass of the structure. The displacements of the three floors and "roof" form the components of the displacement vector y. This lumped parameter model Fig. 1b obeys a vector second order differential equation of the form:

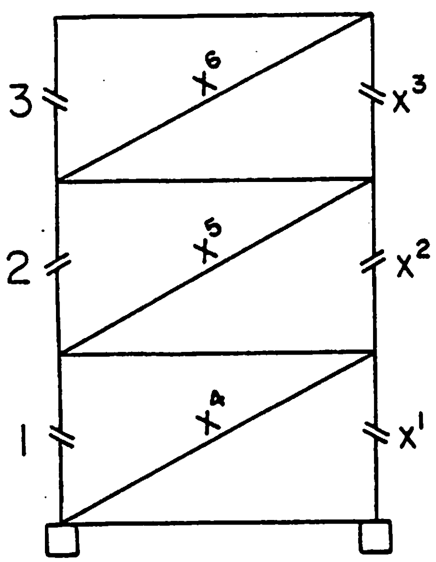


Fig. 1a

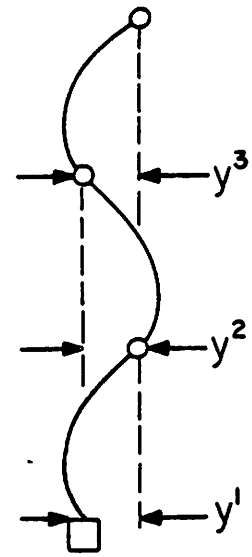


Fig. 1b

$$M\ddot{y}(t,x)+D(\dot{y},y,x)\dot{y}(t,x)+K(\dot{y},y,x)y(t,x) = F(t) \tag{2.1}$$

where F(t) represents the seismic forces. For reasonable values of x, when F is small, i.e., it represents a small earthquake, (2.1) is a linear differential equation, but when F is large, the bending of steel introduces gross nonlinearities due to its hysteretic behavior. It is common to consider a whole family of excitatons $\{F_k\}$, both large and small in carrying out a design. A simple cost to minimize is the cross section of the frame members, while the constraints are introduced to limit the relative floor displacements over the entire duration of a sample earthquake. Thus, we obtain a SIP of the form

$$\min\{f(x) \mid |y^j(t,x,F)-y^{j-1}(t,x,F)| \le d^j(F),\ \forall t \in [0,T], \tag{2.2}$$

$$j = 1,2,3,\ \forall F \in \{F_k\};\ x \ge L\}$$

In choosing a SIP algorithm for solving structural problems, one must bear in mind that the differential equations are highly nonlinear and quite difficult to integrate. For example, a simple four story frame may require several minutes of cpu time using a VAX 11/780 computer to simulate, while a quadratic programming search direction finding subprocedure requires only fractions of a second. Also, since simulation has to be carried out using standard computer codes, one finds that derivatives must be computed by finite differences, making it difficult to use second order algorithms. The algorithm in (33) was developed with these problems in mind. For further reading, see (3, 4, 21).

2.2 Design of SISO Control Systems.

Single-input single output (SISO) control systems are particularly easy to design by SIP techniques. Consider the block diagram of Fig. 2, which shows a plant, with transfer function P(s) and a compensator, with transfer function C(x,s), where x is a three components vector of design parameters to be chosen. We assume that this system satisfies the following set of equations:

$$\dddot{y}(t,x)+5\ddot{y}(t,x)+8\dot{y}(t,y)+6y(t,x) = u(t,x) \tag{2.3}$$

$$\dot{u}(t,x)+30u(t,x) = x^1\ddot{e}(t,x)+x^2\dot{e}(t,x)+x^3e(t,x) \tag{2.4}$$

$$e(t,x) = r(t)-y(t,x) \tag{2.5}$$

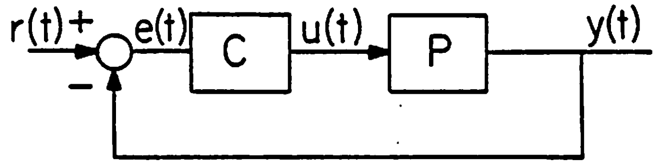


Fig. 2

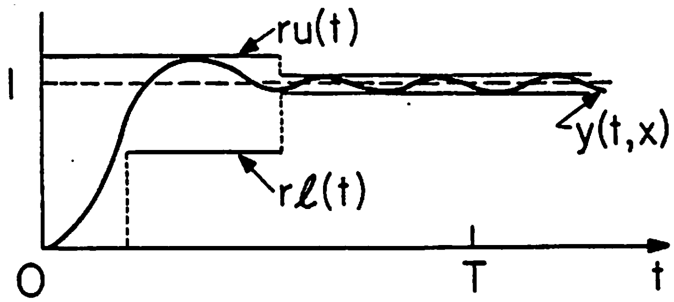


Fig. 3

and hence that

$$P(s) = \frac{1}{(s^3+5s^2+8s+6)} \tag{2.6}$$

$$C(x,s) = \frac{x^1s^2+x^2s+x^3}{s(s+30)} \tag{2.7}$$

In the simplest case, there are three sets of constraints on the design. First, there are positivity and boundedness constraints on x:

$$0 \le x \le b \tag{2.8}$$

Next, there is a time domain constraint on the step response of the closed loop system (i.e., on the response y resulting from r(t) = 1 for $t \ge 0$ and r(t) = 0 for $t < 0$), as shown in Fig. 3. It has the form

$$rl(t \le y(t,x) \le ru(t) \text{ for } all \ t \in [0,T] \tag{2.9}$$

where rl(t) and ru(t) are piece-wise constant functions.

Finally, it may be required that the poles (eigenvalues) of the closed loop system lie in a parabolic region in the complex plane defined by

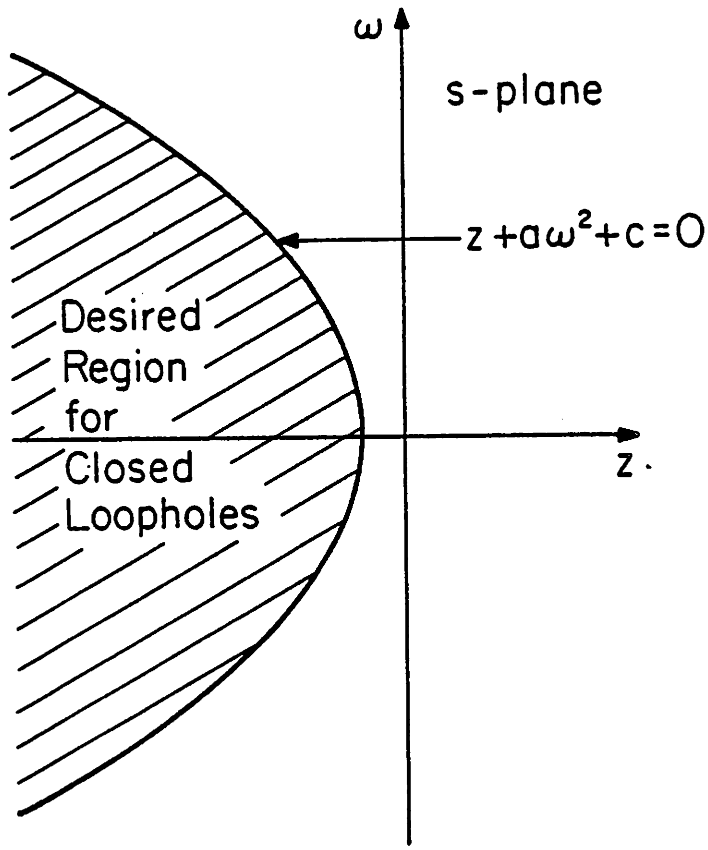
$$z+aw^2+c \le 0, \tag{2.10}$$

where s = z + iw, and a,c $\ge$ 0 (see Fig. 4a). This may be achieved by making use of a modified Nyquist stability criterion, as follows. We may write P(s) = 1/d(s) and C(x,s) = n(x,s)/m(s), where d, n and m are polynomials in s. Then the characteristic polynomial of the closed loop system is

$$X(x,s) = n(x,s)+d(s)m(s) \tag{2.11}$$

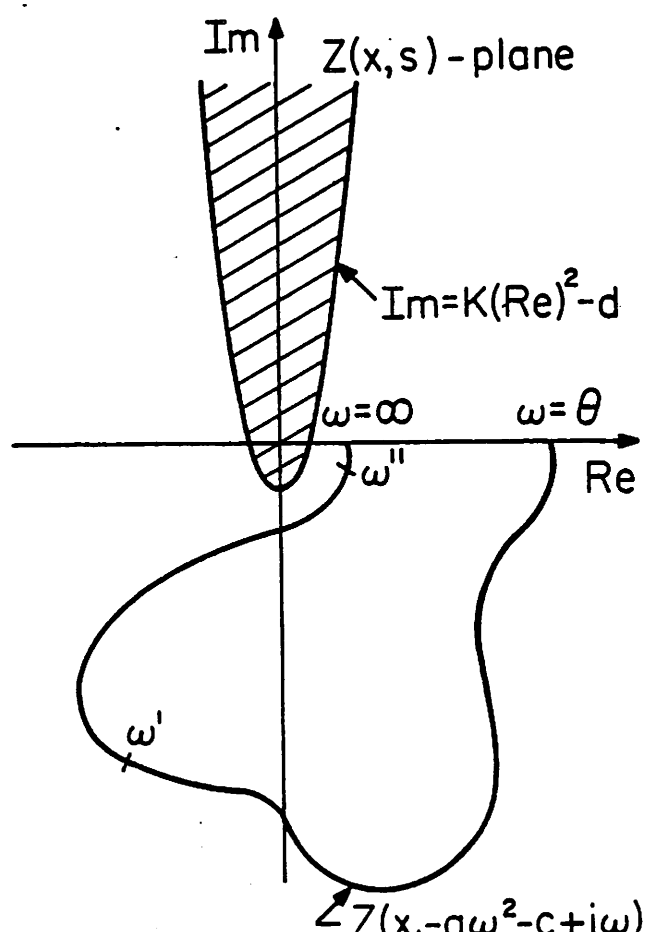
and its zeros are the same as those of the normalized rational function

$$Z(x,s) = X(x,s)/(s+50)^5 \tag{2.12}$$



ω
s-plane
$z + a\omega^2 + c = 0$
Desired
Region
for
Closed
Loopholes
z

Fig. 4a



Im
Z(x, s) - plane
Im=K(Re)^2-d
ω=∞
ω=θ
Re
ω''
ω'
$Z(x_1 - a\omega^2 - c + i\omega)$

Fig. 4b

It follows from arguments as for the Nyquist stability criterion, that a sufficient condition that for the zeros s* of Z(x,s) to satisfy $z^* + aw^{*2} + c \leq 0$ is that the following semi-infinite inequality hold (see Fig.4b):

$$d - KRe\,(Z(x,-aw^2-c+iw))^2 + Im\,(Z(x,-aw^2-c+iw)) \leq 0 \quad \text{for all } w \in [w',w''] \tag{2.13}$$

where d,K > 0, Re and Im denote the real and imaginary parts, respectively, of Z, and w', w" define a critical range of values in the complex plane. It is seen (see Fig. 4b) that (2.13) ensures that the image under $Z(x,\cdot)$ of the modified Nyquist locus given by (2.10) does not encircle the origin.

Finally, as a performance criterion, one could use

$$f(x) = \int_0^T u(t,x)^2 dt \tag{2.14}$$

for r(t) a unit step.

Problems such as these led to the development of the algorithms (9, 5, 20). For other examples of SISO control design via optimization, see (12, 15, 18, 34, 35).

2.3. Design of MIMO Control Systems.

We shall now consider an elementary example from the design of multiple-input multiple-output (MIMO) control systems in the Laplace transform domain, i.e., via a technique which replaces all the differential equations by their Laplace transforms. Referring to to the block diagram in Laplace transform relations given in Fig. 5, the closed loop system is made up of blocks whose inputs and outputs are all m dimensional vector functions. The vector r(s) denotes the Laplace transform of the *closed loop system input* r(t), the m vector y(s) denotes the Laplace transform of the *closed loop system output* y(t), and the m vector u(s) denotes the Laplace transform of the *plant input* u(t). Finally, the m vector d(s) denotes the Laplace transform of the *output disturbance* d(t). The plant transfer function P(s) is an m x m matrix whose elements are proper rational functions in the complex variable s and one is required to design the m x m *compensator* transfer function matrix C(x,s) whose elements are proper rational functions in the complex variable s, with coefficients which depend on the n dimensional *design vector* x. Among other things, the closed loop system is required to be stable and to remain stable under plant perturbations and to be insensitive to the disturbance d(s). Both of these requirements lead to similar, nondifferentiable, semi-infinite inequality constraints and hence we will describe only the latter.

For the purpose of expressing insensitivity to the disturbance d(s), we set r(s) ≡ 0, which leads to the equation

$$y(s) = [I + P(s)C(x,s)]d(s) = Q(x,s)d(s) \tag{2.15}$$

$$u(s) = -C(x,s)Q(x,s)d(s) = R(x,s)d(s). \tag{2.16}$$

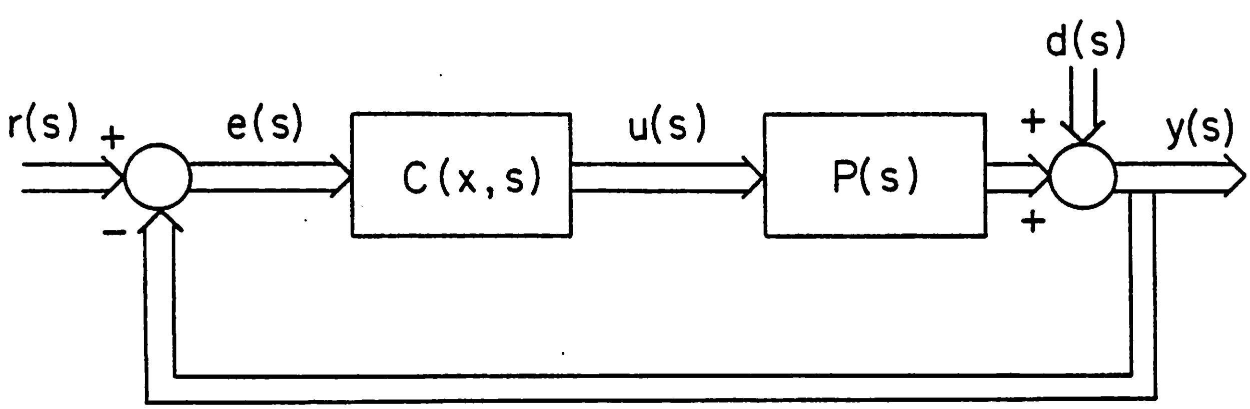


Fig. 5

Let q(x,w) denote the largest singular value of Q(x,iw) and s(x,w) denote the largest singular value of R(x,iw), with w real. The variable w denotes frequency; the matrix Q(x,iw) is complex valued. Since the largest singular values are matrix norms, to make the response y of the system small for a large class of disturbances, without unduly saturating the system as a result of u becoming too large, control system designers strive to keep both q and r small over appropriate frequency ranges. This leads to the following partial formulation of the MIMO control system design problem in SIP form:

$$\textit{minimize } f(x) = \{\max q(x,w) \mid w \in [w',w'']\} \tag{2.17}$$

subject to:

$$s(x,w) \le b(w) \text{ for } \textit{all } w \in [w',w''] \tag{2.18}$$

$$c \le x \le d \tag{2.19}$$

where b(w) is a continuous function of the frequency w.

In addition, there are constraints expressing *decoupling* i.e., the requirement that when only a single component of the input vector is a nonzero function, only the corresponding component of the output vector is nonzero, as well as other constraints on time domain responses, all of which are semi-infinite in form. We note that the singular values q and s are non-differentiable and hence that the SIP problem corresponding to MIMO control system design is considerably more difficult than the SISO problem. For algorithms which solve problems of the fcr (2.17) -(2.19), see (P9). For other examples and a discussion of the use of singular values in control system design, see (8, 13, 26, 32).

2.4. Electronic Circuit Design.

A well known SIP problem in electronic circuit design is the so called *design centering, tolerancing and tuning* problem. Its formulation is developed in three stages, as follows. First, the responses (see Fig. 6) of the circuit to given inputs must satisfy semi-infinite constraints of the form

$$a(t) \le y(t,x) \le b(t) \text{ for } \textit{all } t \in [t',t''] \tag{2.20}$$

where x is an n dimensional design vector. Next, due to production errors, when the circuit is built, the actual value of the design parameter will not be the *nominal value* x, but x + g(e), where e is a *production error parameter*, usually of much smaller dimension than x. To obtain 100% yield, the design engineer must therefore require that

$$a(t) \le y(t,x+g(e)) \le b(t) \text{ for } \textit{all } t \in [t',t''] \quad \text{for } \textit{all } e \in E \tag{2.21}$$

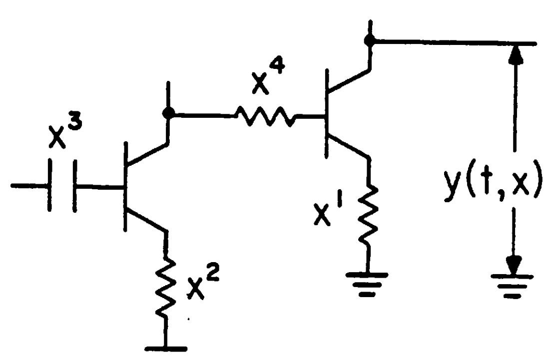


Fig. 6

where E is the production engineer's *tolerance* set. Now, it is not uncommon for the inequalities of the form of (2.21) to define an extremely small, or even empty feasible set and hence engineers have to resort to post manufacture *tuning*, i.e., post manufacture correction of the product. This leads to the introduction of a repair (tuning) parameter r into the picture, so that (2.21) becomes replaced by

$$a(t) \le y(t, x+g(e)+h(r)) \le b(t) \text{for } all \ t \in [t', t''] \tag{2.22}$$

$$\text{for } all \ e \in E, \text{ for } some \ r \in R$$

where R defines the allowable range of the repair parameter. Clearly, (2.22) is an extremely intractable kind of semi-infinite inequality, particularly, since a realistic formulation must replace the single y in (2.22) with the max over all such functions of interest. A certain amount of simplification can be obtained by replacing (2.22) by the following, slightly stricter inequality:

$$a(t) \le \max_{e \in E} \min_{r \in R} \max_{k \in K} y^k(t, x + g(e) + h(r)) \le b(t) \tag{2.23}$$

$$\text{for } all \ t \in [t', t'']$$

Of course, in addition, to the constraint of the form (2.23), the electronic circuit design problem will also involve a number of others that are much more benign. Thus, the electronic circuit designer sees his problem as that of choosing a design center x so that the set x + g(E), resulting from tolerances which must be allowed in production, is as much as possible in the feasible set specified for x, and of making sure that any part of x + g(x), which is outside of the feasible set, can be brought back into it by post-manufacture tuning.

For further reading, see (6, 7, 16, 22, 29).

3. SIP ALGORITHMS FOR ENGINEERING DESIGN.

As we have seen in the previous section, SIP problems in engineering design are of the form

$$\min\{f(x) \mid g^i(x) \le 0 \ \forall i \in I;\ h^j(x,p) \le 0 \ \forall j \in J,\ \forall p \in P; \tag{3.1}$$

$$\max_{e \in E} \min_{r \in R} \max_{t \in T} \max_{k \in K} y^k(x, e, r, t) \le 0\}$$

where f, g , h , and y are, usually, real valued, continuously differentiable functions; P, E, R are compact multidimensional sets, T is a compact interval, and K is a finite set of indices. When the functions in (3.1) are not continuously differentiable, they are semi-smooth (18) or, at least, locally Lipschitz.

We shall now describe two algorithms which we have found to be very reliable in engineering design. The first is due to Gonzaga, Mayne, Polak and Trahan (P1,9) and it solves problems of the form (3.1) without the max min max term, when all functions are continuously differentiable, i.e., it solves problems for the form:

$$\min\{f(x) \mid g^i(x) \le 0 \ \forall i \in I;\ h^j(x, p_j) \le 0 \ \forall p_j \in p_j,\ \forall j \in J\} \tag{3.2}$$

The algorithm in (9) is a refinement of the algorithm in (20). In turn, it underwent some refinement, as described in (5), the most significant of which is the introduction of weights w which can be used to improve the conditioning of the problem. For simplicity of exposition, we may as well consider the case of (3.2), where the "simple" inequalities specified by the g^i are not present and there is only one semi-infinite inequality, i.e., J = {1}, so that the index j can be dropped. For this case, we define, with $e \ge 0$, $t \ge 0$, $w \ge 0$,

$$M(x) \max\{h(x,p) \mid p \in p\} \tag{3.3}$$

$$M(x)_+ = \max\{M(x), 0\} \tag{3.4}$$

$$P_e(x) = \{p \in P \mid h(x,P) \ge M(x)_+ - e, \text{ and } p \textit{ is a local maximizer of} \tag{3.5}$$

$$h(x,\cdot) \textit{ in } P\}$$

$$W_e(x) = \min\{tu^f M(x)_+ + (1/2)||u^f \nabla f(x) + \sum_{p \in P_e(x)} u^p \nabla h(x,p)||^2 | w^f u^f + \sum_{p \in P_e(x)} w^p u^p = 1, u \geq 0\} \tag{3.6}$$

and

$$d_e(x) = -[u^f \nabla f(x) + \sum_{p \in P_e(x)} u^p \nabla h(x,p)] \tag{3.7}$$

where u^f, u^p solve (3.6). Finally, with $0 < c < 1$, and $q > 0$, we define

$$E = \{0,1,c,c^2,c^3,....\} \tag{3.8}$$

and

$$e(x) = \max\{e \in E \mid W_e(x) \geq eq\} \tag{3.9}$$

We note that the parameter e in (3.5) controls the number of local maximizers that will be used in the computation of the search direction $d_e(x)$. The parameters w^f, w^p in (3.6) are used to control the angle between the search direction $d_e(x)$ and the corresponding gradients: when w^p is increased, the angle between $-d_e(x)$ and $grh(x,p)$ is decreased. The parameter t controls the phase I - phase II aspect of the algorithm: when t is large, the algorithm places greater emphasis on achieving feasibility than when t is small. Finally, the parameter q is used to control the value of the parameter e that will be used in the computation of the search direction $d_e(x)$: the smaller q, the larger will e(x) be. In practice, this rather large number of parameters turns out to be rather easy to adjust, provided the computations are carried out in an interactive environment.

Finally, introducing two parameters $a,b \in (0,1)$ for the Armijo step size rule, we are ready to state the algorithm which will be recognized as a phase I - phase II method of feasible directions, see (23).

Algorithm 1:

Parameters: $a,b,c \in (0,1)$, $q > 0$, $K > 0$.

Data: x_0.

Step 0: Set $i = 0$.

Step 1: Compute $d_{e(x_i)}(x_i)$.

Step 2: If $M(x_i) > 0$, compute the largest $s \in \{K, bK, b^2K, ...\}$ such that

$$M(x_i + sd_{e(x_i)}(x_i)) - M(x_i)) \leq -saW_{e(x_i)}(x_i). \tag{3.10}$$

Else, compute the largest $s \in \{K, bK, b^2K, ...\}$ such that

$$M(x_i + sd_{e(x_i)}(x_i)) \leq 0 \tag{3.11}$$

and

$$f(x_i + sd_{e(x_i)}(x_i)) - f(x_i) \leq -saW_{e(x_i)}(x_i). \tag{3.12}$$

Step 3: Set

$$x_{i+1} = x_i + sd_{e(x_i)}(x_i) \tag{3.13}$$

set $i = i + 1$, and go to Step 1.

It can be shown (see (9)) that every accumulation point of Algorithm 1 is feasible and satisfies a standard F. John type condition of optimality.

Algorithm 1 has been used as is and also as a vehicle for stabilizing local superlinearly converging algorithms such as sequential quadratic programming (27). The stabilization is accomplished as follows. A simple comparison test is used to determine whether the local version is in its region of convergence. If the local algorithm is in its region of convergence, it is allowed to proceed, otherwise Algorithm 1 is used to drive the iterates into the region of convergence of the local method. The result is an algorithm with mathematically demonstrable global convergence properties (see (27)).

For non-differentiable problems of the form (17) -(19), a modification of Algorithm 1 has been proposed in (28). The algorithm in (28) substitutes e-approximations to the generalized gradients for the gradients used in Algorithm 1, and uses the outer decompositions algorithm (10), to be described below, to replace the intervals $[w',w'']$ with finite sets of points.

Finally, for differentiable problems for which derivative computations have not been incorporated into simulation codes, we find a derivative free version of Algorithm 1 in (33).

An important tool for dealing with problems of the form (3.1), either totally or partially, is the family of outer approximations algorithms, see, e.g (10,14,16). Probably the most elaborate use of this tool can be found in (29), where it is used to "unwind" (3.1) into sequences of differentiable, finitely constrained optimization problems. We shall describe an example of these methods in terms of the simplified problem

$$\min\{f(x) \mid h(x,p) \le 0, \forall p \in P\} \tag{3.14}$$

where f and h are real, at least, locally Lipschitz continuous functions defined on R^n . The idea is to decompose (3.14) into a sequence of problems in which P is replaced by finite sets P , which, hopefully, remain of small cardinality.

Algorithm 2:

Parameters: t $\in$ (0,1) and a double subscripted sequence $\{s_{jk}\}$ satisfying

(i) $s_{kk} = 0$ for all k and $s_{jk} > 0$ for all $k > j$;
(ii) $s_{jk} \to \hat{s}_j$ as $k \to \infty$.
(iii) $\hat{s}_j \to 0$ as $j \to \infty$.

Data: A finite set $P_0 \in P$.

Step 0: Set k = 0.

Step 1: Compute by means of Algorithm 1 an x_k such that

$$h(x_k,p) \le t^k, \forall p \in P, e(x_k) \le t^k. \tag{3.14a}$$

Step 2: Computer a

$$p_k \in argmax\{h(x,p) \mid p \in P\}. \tag{3.15}$$

Step 3: Set

$$P_{k+1} = \{p\} \cup \{p_j \in P_k \mid h(x_j,p_j) \ge s_{jk}\}, \tag{3.16}$$

set k = k + 1 and go to Step 1.

We note that the parameter t is used to control the precision with which the kth problem is solved. It is necessary to solve the successive problems with progressively greater precision. In (3.14) an exponential increase of precision is specified, however, one may increase the precision at a slower rate. The double subscripted sequence s_{jk} is used for determining which of the p_j , that were included in P_k , need not be carried over into P_{k+1}, i.e., they form part of the constraint dropping scheme. When one wishes to drop lots of constraints, the s should decrease to zero very slowly as k increases. There are many possibilities for such a sequence, for example, one may use $s_{jk} = 100/[1/(1+j)^{1/10} - 1/(1+k)^{1/10}]$. When computing in environment, it is possible to exercise highly intelligent control over the constraint dropping mechanism.

In view of our comments on the effect of parameters, it should be clear that algorithms of the type discussed above are very difficult to use in batch mode in an engineering design where function evaluations consume many minutes of computer time. It has been our experience that the computations can be made much more efficient by parameter adjustment in an interactive computing environment, such as the ones described in (2,19).

4. CONCLUSION.

We hope that this brief survey of SIP problems arrising in engineering and of some of the algorithms that have been used for their solution will stimulate the interest of SIP researchers in the very challenging problems that occur in engineering design.

5. ACKNOWLEDGEMENT: This research was supported by the National Science Foundation under grants ECS-79-13148 and CEE-81-05790 and the Joint Services Electronics Program under grant F49620-79-C-0178.

6. REFERENCES.

(01) Becker, R. G., A. J., Heunis and D. Q. Mayne: "Computer-Aided Design of Control Systems via Optimization," Proc. IEE, vol. 126, no. 6, 1979.

(02) Bhatti, M. A., T. Essebo, W. Nye, K. S. Pister, E. Polak, A. Sangiovanni Vincentelli and A. Tits: "A Software System University of for Optimization Based Interactive Computer Aided Design," Proc. IEEE I.S.C.A.S. Houston, TX, April 1980.

(03) Bhatti, M. A., K. S. Pister, and E. Polak: "Optimal Design of an Earthquake Isolation System," Proc. IUTAM Symp. on Structural Control, Univ. of Waterloo, Waterloo, Ont., Canada, June, 1979.

(04) Bhatti, M. A., K. S. Pister, and E. Polak "Optimization-Based, Interactive CAD of Dynamically Loaded Structures," to appear in Jour. of Structural Dynamics, of the A.S.C.E.

(05) Bhatti, M. A., K. S. Pister, and E. Polak: "Optdyn - A General Purpose Optimization Program for Problems With or Without Dynamic Constraints," Report No. UCB/EERC - 79/16, Earthquake Engineering Research Center, University of California, Berkeley, July 1979.

(06) Brayton, R. K., S. W. Director and G. D. Hachtel: "Yield Maximization and Worst Case Design with Arbitrary Statistical Distributions," IEEE Trans. on Circuits and Systems, Vol. CAS-27, n. 9, (1980).

(07) Bandler, J. W., P. C. Liu and H. Tromp: "Nonlinear Programming Approach to Optimal Design Centering Tolerating and Tuning," IEEE Trans., Vol. CAS-23, (1976).

(08) Davison, E. J., and I. J. Ferguson: "The Design of Controllers for the Multivariable Robust Servomechanism Problem Using Parameter Optimization Methods," Systems Control Report No. 8002, University of Toronto, February 1980.

(09) Gonzaga, C., E. Polak, and R. Trahan: "An Improved Algorithm for Optimization Problems with Functional Inequality Constraints," IEEE Trans., Vol. AC-25, No. 1, (1980).

(10) Gonzaga, C., and E. Polak: "On Constraint Dropping Schemes and Optimality Functions for a Class of Outer Approximations Algorithms," J. SIAM Control and Optimization Vol. 17, (1979).

(11) Hettich, R., "Semi-Infinite Programming," Springer-Verlag Lecture Notes in Control and Information Sciences, Vol. 15, 1979.

(12) Karmarkar J. S., and D. D. Siljak: "Maximization of Absolute Stability Regions byy Mathematical Programming Methods," Regelungtechnik, No. 2, (1975).

(13) Laub, A. J.: "An Inequality and Some Computations Related to the Robust Stability of Linear Dynamic Systems," IEEE Trans. on Automatic Control, Vol. AC-24, No. 2, (1979).

(14) Mayne, D. Q., E. Polak, and R. Trahan: "An Outer Approximations Algorithm for Computer-Aided Design Problems," JOTA, Vol. 28, No. 3, (1979).

(15) Mayne, D. Q., and E. Polak: "A Quadratically Convergent Algorithm for Solving Infinite-Dimensional Inequalities," Memo No. UCB/ERL M80/11, University of California, Berkeley, 1980.

(16) Mayne, D. Q., E. Polak, and A. Voreadis: "A Cut Map Algorithm for Design Problems with Tolerances," IEEE Trans., Vol. CAS-29, No. 1, (1982).

(17) Mayne, D. Q., E. Polak, and A. J. Heunis: "Solving Nonlinear Inequalities in a Finite Number of Iterations," JOTA Vol. 33, No. 2, (1981).

(18) Mifflin, R., "Semismooth and Semiconvex Functions in Constrained Optimization," SIAM J. Control and Optimization, Vol. 15, No. 6, (1977).

(19) Nye, W., E. Polak, A. Sangiovanni Vincentelli, and A. Tits: "DELIGHT: An Optimization-Based Computer-Aided-Design System," Proc. IEEE Int. Symp. on Circuits and Systems, Chicago, IL, April 24-27, 1981.

(20) Polak, E., and D. Q. Mayne: "An Algorithm for Optimization Problems with Functional Inequality Constraints," IEEE Trans., Vol. AC-21, No. 2, (1976).

(21) Polak, E., K. S. Pister, and D. Ray: "Optimal Design of Framed Structures Subjected to Earthquakes," Eng. Optimization, Vol. 12, (1976).

(22) Polak, E., and A. Sangiovanni Vincentelli "Theoretical and Computational Aspects of the Optimal Design Centering, Tolerancing and Tuning Problem," IEEE Trans., Vol. CAS-26, No. 9 (1979).

(23) Polak, E., R. Trahan, and D. Q. Mayne: "Combined Phase I - Phase II Methods of Feasible Directions," Math. Programming, Vol. 17, No. 1, (1979).

(24) Polak, E., and D. Q. Mayne: "Algorithmss for Computer Aided Design Aided Design of Control Systems by the Method of Inequalities," Proc. 18th IEEE Conference on Decision and Control, Fort Lauderdale, FL, December 12-14, 1979.

(25) Polak, E.: "Algorithms for a Class of Computer Aided Design Problems: a review," Automatica, Vol. 15, (1979).

(26) Polak, E., and D. Q. Mayne: "Design of Nonlinear Feedback Controllers," IEEE AC Trans. on Automatic Control, Vol. AC-26, No. 3, (1981).

(27) Polak, E., and A. Tits: "A Recursive Quadratic Programming Algorithm for Semi-Infinite Optimization Problems," University of California, Berkeley, ERL Memo No. UCB/ERL M80/50, 22 September, 1980.

(28) Polak, E., and Y. Y. Wardi: "A Nondifferentiable Optimization Algorithm for the Design of Control Systems Subject to Singular Value Inequalities over a Frequency Range," Proceedings IFAC/81 Congress, Kyoto, Japan, August 24-28, 1981.

(29) Polak, E.: "An Implementable Algorithm for the Design Centering, Tolerancing and Tuning Problem," JOTA, Vol. 35, No. 3, (1981).

(30) Safonov, M. G., A. J. Laub, and G. L. Hartman: "Feedback Properties of Multivariable Systems: The Role and Use of the Return Difference Matrix," IEEE Trans. on Control, Vol. AC-26, (1981).

(31) Sandel, N. R.: "Robust Stability of Systems with Applications to Singular Value Perturbations," Automatica, Vol. 15, (1979).

(32) Taiwa, O.: "Design of a Multivariable Controller for a High Order Turbofan Engine Model by Zakian's Method of Inequalities," IEEE Trans., Vol. AC-23, No. 5, (1978).

(33) Trahan, R., and E. Polak: "A Derivative Free Algorithm for a Class of Infinitely Constrained Optimization Problems," IEEE Trans. Vol. AC-25, No. 1, (1979).

(34) Zakian, V., and L. Al-Naib: "Design of Dynamical and Control Systems by the Method of Inequalities," Proc. IEE, 120 (11), (1973).

(35) Zakian, V.: "New Formulation for the Method of Inequalities," Proc. IEE, 126 (6), (1979).