INTERNAL REPORTS IN

SIMULATION, OPTIMIZATION AND CONTROL

No. SOC-110

EFFICIENT, AUTOMATED DESIGN CENTERING AND TOLERANCING

J.W. Bandler, P.C. Liu and H. Tromp

October 1975

FACULTY OF ENGINEERING
McMASTER UNIVERSITY
HAMILTON, ONTARIO, CANADA

EFFICIENT, AUTOMATED DESIGN CENTERING AND TOLERANCING

J.W. Bandler, Senior Member, IEEE , P.C. Liu, Student Member, IEEE, and H. Tromp

Abstract We present an efficient approach to the optimal assignment of component tolerances along with centering and, eventually, tuning. The main objective is to automate the process without sacrificing computational efficiency. The development of selection schemes for critical vertices of the tolerance region is discussed in detail. As the process proceeds vertices can be added or purged automatically. The exploitation of symmetry in design problems is considered at length. The presentation is illustrated by a five-section transmission-line lowpass filter, where both characteristic impedances and section lengths are toleranced and parasitic junction effects are simulated. Finally, capacitive tuning is also considered. The paper contains numerous tables comparing the effectiveness and efficiency of different schemes.

This work was supported by the National Research Council of Canada under Grant A7239 and by a Graduate Fellowship of the Rotary Foundation to H. Tromp.

J.W. Bandler is with the Group on Simulation, Optimization and Control and Department of Electrical Engineering, McMaster University, Hamilton, Canada, L8S 4L7.

P.C. Liu was with the Group on Simulation, Optimization and Control and Department of Electrical Engineering, McMaster University, Hamilton, Canada. He is now with Bell-Northern Research, Verdun, P.Q., Canada.

H. Tromp was with the Group on Simulation, Optimization and Control and Department of Electrical Engineering, McMaster University, Hamilton, Canada. He is now with the Laboratory of Electromagnetism and Acoustics, University of Ghent, Ghent, Belgium.

I. INTRODUCTION

This paper presents an efficient approach to the optimal assignment of component tolerances along with centering and, eventually, tuning. The main objective is to automate the process without sacrificing efficiency.

The historical development of the general area is indicated by a number of relevant papers [1-7]. The authors have already demonstrated the benefits of allowing the nominal design to be variable while optimizing the tolerances [6,7], and the way in which tuning can be brought in [8,9]. The principle drawback with the previous work is the interaction necessary by the designer to reduce the computational effort.

A typical problem solving sequence is shown in Fig. 1. This sequence can also be started as an intermediate problem. Problem 1 is the conventional problem of performance optimization. Its result, or else a design obtained by analytical methods, can be used as a starting point for the next steps in the sequence. Problem 2 is also a performance optimization, but now the objective function or performance criterion should be formulated in terms of the network response for all, or a selected number , of possible outcomes of the parameter vector. This problem allows the designer to find out what the optimal design is and whether the specifications can be met, when the tolerances are given and fixed. The solution of problem 2 helps the designer in deciding what to do next. If the specifications cannot be met, even with minimal tolerances, tuning can be introduced. This leads to problem 4, eventually, with fixed tolerances. If no tuning elements are wanted, we can proceed to calculate the yield. That can be used again in a yield optimization.

Even if problem 2 is not relevant practically, its result usually provides a good starting point for problem 3. This is the tolerance optimization problem as described above. From problem 3 we can proceed to problem 4, the

tolerancing-tuning problem.

In this paper, we shall mainly deal with the efficient automatic solution of problem 3, but most of the methods and results will also be useful in problems 2 and 4, which have essentially the same structure.

II. IMPLEMENTATION OF THE TOLERANCE PROBLEM

The methods to be described are illustrated by the results for the 5-section transmission-line stepped impedance lowpass filter, shown in Fig. 2. Relevant element data is shown in Table I. The nominal characteristic impedances are fixed and have a uniform fixed relative tolerance. ℓ_i, $i = 1,\ldots,5$, are the line lengths, normalized w.r.t. the quarter wavelength at the passband edge. They have variable nominal values and a uniform variable absolute tolerance. The parasitic capacitors C_i, $i = 1,\ldots,6$, were calculated from formulas given by Marcuvitz [10] for a step in the inner conductor of a coaxial line at the edge of the passband and with an inner diameter of the outer conductor of 1 inch.

The specifications and the objective function are given in Table II. The parameter vector in this case is given by

$$\underset{\sim}{\phi} = [\ell_1\ \ell_2\ \ell_3\ \ell_4\ \ell_5\ Z_1\ Z_2\ Z_3\ Z_4\ Z_5]^T. \tag{1}$$

In this case there are $2^{10} = 1024$ vertices of the tolerance region [6,7] given, in general, by

$$R_\varepsilon \triangleq \{\underset{\sim}{\phi} \mid \phi_i = \phi_i^0 + \varepsilon_i\mu_i,\ -1 \leq \mu_i \leq 1,\ i \in I_\phi\}, \tag{2}$$

where superscript 0 distinguishes the nominal parameter values, ε_i the tolerance on the ith component and, for k components,

$$I_\phi \triangleq \{1,2,\ldots,k\}. \tag{3}$$

In principle, the network response should be calculated for all possible values of the parameter vector; or at least at the 2^k vertices of the tolerance region, when a one-dimensional convexity condition is satisfied [5]. In order to reduce the problem, we need a vertex selection method, which finds the worst vertex or at least gives an accurate prediction of the vertices which are critical in the optimization.

The tolerance optimization problem can be solved by the general scheme of Fig. 3. Note that the tolerancing-tuning problem can be solved by the same scheme. The minimax approximation problem with fixed tolerances has also the same basic structure, the only difference being that the selected vertices are not used in the calculation of constraint functions, but for the error functions, and enter the objective function. The critical step in running the scheme of Fig. 3 efficiently is the vertex selection. Several methods of vertex selection have been previously used. The one which the authors had found acceptable involves changing each parameter one at a time from the nominal point and examining the partial derivatives of the response or constraint functions [7]. The disadvantage, in general, is that sometimes only one vertex is selected, which is often insufficient if the starting point is far from the solution. On the other hand, too many can be selected, most of which are not critical.

A large number of selected constraints makes the whole process inefficient or even impossible to run. That means that a further reduction is needed, based, for example, on the constraint values at the selected vertices. This often involves a large number of response evaluations and, at some frequency points, all the vertices have to be evaluated. Moreover, the old schemes are not foolproof. In our example, even at the optimal solution, the one-at-a-time scheme misses some of the critical constraints. That means

that after each optimization, extensive testing of the results is needed, which involves at least the evaluation of all vertices at all frequencies and is extremely time-consuming. It is clear from these considerations that, with the known vertex selection methods, the problem can hardly be solved efficiently and the process cannot be automated. Interference of the designer is continually needed relying on his insight and his experience. Usually, several test runs are needed.

Some results for the lowpass filter are summarized in Table III for future reference. They were obtained by the one-at-a-time vertex selection method, followed by manual selection and testing of all vertices after each optimization. The minimax optimization with fixed tolerances was started with the parameter values obtained from the nominal minimax approximation. Two runs were needed using MINOPT [11]. Figs. 4 and 5 show the frequency response in passband and stopband, respectively, of the nominal minimax approximation. Figs. 4 and 6 show the nominal response and the worst response at each frequency, in the passband and stopband, respectively, for the minimax approximation with fixed tolerance on all lengths ($\varepsilon_\ell = 0.001$). Analogous results for $\varepsilon_\ell = 0.002$ show that the specifications cannot be satisfied in that case. The passband ripple of the nominal response of the nominal minimax approximation was 0.00123 dB. The worst response of the minimax approximation with fixed tolerances is 0.0125 dB if $\varepsilon_\ell = 0.001$, and 0.02007 dB when $\varepsilon_\ell = 0.002$.

The tolerance optimization was started from the parameter values resulting from minimax approximation with $\varepsilon_\ell = 0.001$. Six runs of DISOPT [12] were needed, with intermediate testing of vertices. The final run took 60 secs on a CDC 6400 using 228 function evaluations. The number of variables is 4 and the number of constraints 49. Figs. 7 and 8 show the nominal response and the upper and lower bounds over all vertices for the resulting parameter values in the passband and stopband, respectively.

The symmetry of the problem was exploited. There is a double mirror-symmetry in the parametervector $\underset{\sim}{\phi}$, which means that the insertion loss $L(\underset{\sim}{\phi})$ is symmetric in the parameter space:

$$L(\underset{\sim}{\phi}) = L(\underset{\sim}{S}\underset{\sim}{\phi}), \tag{4}$$

where the symmetry operator $\underset{\sim}{S}$ is given by

$$\underset{\sim}{S} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & & & & & \\ 0 & 0 & 0 & 1 & 0 & & & & & \\ 0 & 0 & 1 & 0 & 0 & & \multicolumn{3}{c}{\multirow{3}{*}{$\underset{\sim}{0}$}} & \\ 0 & 1 & 0 & 0 & 0 & & & & & \\ 1 & 0 & 0 & 0 & 0 & & & & & \\ & & & & & 0 & 0 & 0 & 0 & 1 \\ & & & & & 0 & 0 & 0 & 1 & 0 \\ & & \multicolumn{3}{c}{\multirow{3}{*}{$\underset{\sim}{0}$}} & 0 & 0 & 1 & 0 & 0 \\ & & & & & 0 & 1 & 0 & 0 & 0 \\ & & & & & 1 & 0 & 0 & 0 & 0 \end{bmatrix} . \tag{5}$$

The number of vertices which are essentially different (in response value) is reduced from 1024 to 544.

As a result of the symmetry, the optimum $\underset{\sim}{\phi}^0$ should lie in the plane of symmetry. If the nominal point is forced to be symmetric throughout the optimization, both the number of parameters and the number of constraints is reduced. When symmetry is not exploited, care should be taken in the vertex selection, to ensure that of each pair of symmetric vertices, both should be used as constraints or error functions. As an example, the vertices used in the final run of the minimax approximation with fixed $\varepsilon_\ell = 0.001$ of the lowpass filter are shown in Table IV, when symmetry is and is not used. The savings made by the use of symmetry are clear from Table V. CPU times given apply to the CDC 6400.

III. EFFICIENT VERTEX SELECTION

The vertex selection method which we propose is a based on an iterative solution of a necessary condition for the worst vertex, derived from the Kuhn-Tucker conditions.

Let $g(\underset{\sim}{\phi})$ be the constraint function, generated by the network response at a certain frequency point. The problem of finding the worst vertex at that frequency is equivalent to the nonlinear programming problem:

$$\underset{\underset{\sim}{\mu}}{\text{Min}} \quad g(\underset{\sim}{\phi}^0 + \underset{\sim}{E}\underset{\sim}{\mu}) \tag{6}$$

subject to

$$1 - \mu_i \geq 0, \; 1 + \mu_i \geq 0, \; i \in I_\phi, \tag{7}$$

where $\underset{\sim}{E}$ is a diagonal matrix with ε_i as the ith element.

Let $\check{\underset{\sim}{\mu}}$ be the minimum of (6). The Kuhn-Tucker conditions are

$$\underset{\sim}{\nabla}_\mu \, g(\underset{\sim}{\phi}^0 + \underset{\sim}{E}\check{\underset{\sim}{\mu}}) = \sum_{i=1}^{k} u_i^- \; \underset{\sim}{\nabla}_\mu (1 - \check{\mu}_i) + \sum_{i=1}^{k} u_i^+ \; \underset{\sim}{\nabla}_\mu (1+\check{\mu}_i) \tag{8}$$

$$u_i^- \; (1 - \check{\mu}_i) = 0 \; , \; i \in I_\phi \tag{9}$$

$$u_i^+ \; (1 + \check{\mu}_i) = 0 \; , \; i \in I_\phi \tag{10}$$

$$u_i^- \geq 0 \; , \; i \in I_\phi \tag{11}$$

$$u_i^+ \geq 0 \; , \; i \in I_\phi \tag{12}$$

where $\underset{\sim}{\nabla}_\mu$ is the first partial derivative operator w.r.t. $\underset{\sim}{\mu}$.

Equation (8) is reduced to

$$\left.\frac{\partial g}{\partial \mu_i}\right|_{\underset{\sim}{\mu}=\check{\underset{\sim}{\mu}}} = \varepsilon_i \left.\frac{\partial g}{\partial \phi_i}\right|_{\underset{\sim}{\phi}=\underset{\sim}{\phi}^0+\underset{\sim}{E}\check{\underset{\sim}{\mu}}} = u_i^+ - u_i^- \; , \; i \in I_\phi. \tag{13}$$

If the tolerance region is one-dimensionally convex, the worst point in it should be a vertex. That means that any component $\check{\mu}_i$ of $\check{\underset{\sim}{\mu}}$ has two possible values:

1) If $\check{\mu}_i = 1$, then from (10)

$$u_i^+ = 0 \tag{14}$$

and from (13) and (11)

$$\varepsilon_i \left.\frac{\partial g}{\partial \phi_i}\right|_{\underset{\sim}{\phi}=\underset{\sim}{\phi}^0+\underset{\sim}{E}\check{\underset{\sim}{\mu}}} = -u_i^- \leq 0. \tag{15}$$

2) If $\check{\mu}_i = -1$, then from (9)

$$\mu_i^- = 0 \tag{16}$$

and from (13) and (12)

$$\varepsilon_i \left.\frac{\partial g}{\partial \phi_i}\right|_{\underset{\sim}{\phi}=\underset{\sim}{\phi}^0+E\check{\underset{\sim}{\mu}}} = u_i^+ \geq 0. \tag{17}$$

Equations (15) and (17) can be summarized in the following necessary condition for the worst vertex:

$$\check{\underset{\sim}{\mu}} = -\text{sgn}\,[\nabla_{\underset{\sim}{\phi}} g(\underset{\sim}{\phi}^0 + E\check{\underset{\sim}{\mu}})], \tag{18}$$

where $\nabla_{\underset{\sim}{\phi}}$ is the first partial derivative operator w.r.t. $\underset{\sim}{\phi}$.

Equation (18) can be solved iteratively for $\check{\underset{\sim}{\mu}}$. If $\underset{\sim}{\mu}^{(k)}$ is the kth approximation for $\check{\underset{\sim}{\mu}}$, then

$$\underset{\sim}{\mu}^{(k)} = -\text{sgn}\,[\nabla_{\underset{\sim}{\phi}} g(\underset{\sim}{\phi}^0 + E\underset{\sim}{\mu}^{(k-1)})] \quad . \tag{19}$$

A suitable starting point would be the nominal point, i.e.,

$$\underset{\sim}{\mu}^{(0)} = \underset{\sim}{0}.$$

Note that the iterative solution (19) of (18) amounts to successive linear approximations of the constraint function.

Table VI illustrates the performance of the vertex selection methods for the lowpass filter. Results are shown for the parameter values corresponding to the optimally toleranced solution. Symmetry of the problem was exploited. The results obtained by the one-at-a-time method confirm the criticism made earlier. A large number of vertices is generated and at two sample points no vertices were eliminated. The number of function evaluations required to get a vertex table and the corresponding constrained values (needed for further selection) can be found as: six response evaluations with gradient calculation + the number shown in column 6 of Table VI response evaluations without gradient calculation. It is clear that the method is not efficient. Moreover, it fails to find the worst vertex at the sample frequencies and gives a wrong vertex as the worst at one sample frequency.

Some data on efficiency and accuracy are summarized in Table VII. Since (18) is only a necessary condition, more than one vertex will satisfy it, only one among them being the worst vertex, generally. This is clear from column 3 in Table VI. The first iteration, starting at the nominal point, does generally not lead to the worst vertex, which is illustrated by column 7 in Table VI. This is especially true when there is symmetry. In the case of mirror symmetry, the gradient at the nominal point lies on the axis or in the plane of symmetry and the first iteration leads to a symmetrical vertex. This is confirmed by the results for our example (see columns 3 and 7 of Table VI): the first iteration gives always the worst symmetrical vertex. That means that additional testing is necessary.

It should be noted that in our example, only one iteration was needed in each initial iterative search. In the iterative searches to be mentioned further, there were never more than two iterations needed.

More than one vertex can satisfy (18) due to concavity of the constraint function. This fact suggests an obvious scheme for further vertex selection: the testing of adjacent vertices. Adjacent vertices are vertices differing in only 1 component of $\underset{\sim}{\mu}$. Two testing schemes were used, both starting with an iterative search from the nominal point.

Testing scheme 1

Each time a vertex is found satisfying (18) all adjacent vertices are tested to see whether they satisfy (18).

Testing scheme 2

Each time a vertex is found satisfying (18) the iterative search is restarted from all adjacent vertices.

Both schemes have the structure of a branching process. The process stops when all possibilities are exhausted. A particular branch can also stop when a vertex, or its symmetrically equivalent one, is encountered for the second time. When this occurs, however, in the initial iterative search it indicates that the constraint region is not one-dimensionally convex and the worst point within the tolerance region is not a vertex. One action we can take in such a case is to neglect that particular frequency. In all practical problems solved until now this never happened at a critical frequency, though it often happens at other frequencies.

A complication is caused by the presence in many problems of <u>related parameters</u>. Two or more parameters are called related if they have to be changed simultaneously, when going from one vertex satisfying (18) to another one, both vertices of course taken for the same frequency. In our example, the characteristic impedance and the normalized length of the same transmission line section were found to be related. For the nominal parameter values of interest: considering the vertices in column 3 of Table VI, for a particular frequency, μ_{Z_i} and μ_{ℓ_i} (same i) have to be changed simultaneously to go from one to another.

It is clear that testing scheme 1 will not work in most cases where there are related parameters. In our example, a vertex adjacent to a vertex satisfying (18) may never satisfy (18). Testing scheme 2 will work, but generally not reliably. If we know a priori that there are related parameters, we can use both testing schemes if we reinterpret the term "adjacent": adjacent vertices are vertices differing only in the μ_i's corresponding to a set of related parameters.

Table VI, columns 8-13, gives results for both testing schemes, assuming and not assuming relationship. Table VII summarizes some data on the performance of the different schemes. It is clear that testing scheme 2 is more accurate than scheme 1 and requires only slightly more effort. Assuming related parameters improves both accuracy and efficiency.

If we are not a priori sure about the existence of relationships between the parameters, it is possible, in the general process of Fig. 3, to do the first vertex selection without assuming any relationship. The generated vertex table can then be tested for related parameters, which can easily be done automatically. If there is an indication for a relationship, this can be assumed in subsequent vertex selections. Finally, the constraint set of the last optimization has to be tested again, to see whether the assumed relationship still exists.

From the results in Tables VI and VII, we can conclude that the proposed vertex selection scheme has the following advantages:

- it requires fewer function evaluations than the one-at-a-time scheme, to get the same information, i.e., a vertex table and the corresponding constraint values, needed for further processing;
- the accuracy is at least the same, and usually better;
- the effort required depends on the conditioning of the constraint function at a particular frequency.

This makes the scheme suitable for automatic tolerance optimization according to Fig. 3.

The same tolerance optimization problem as before was solved with the new vertex selection scheme. Testing scheme 2 was used, assuming related parameters. The parameters in the optimization were the same. With the

same starting point, the solution was found in only 1 cycle of the process of Fig. 3. A total of 40 sec of CPU time was needed on the CDC 6400, and 157 function evaluations in the single optimization. The number of constraints was 33 (fewer than before). This performance compares extremely favourably with the 6 runs needed before, requiring an average of 60 sec each, plus 2 to 3 minutes for vertex selection and intermediate testing.

IV. EFFICIENT AUTOMATIC TOLERANCING

As a test problem for the automatic tolerancing process to be discussed we have chosen the starting point

$$\ell_1^0 = \ell_2^0 = \ell_3^0 = \ell_4^0 = \ell_5^0 = 0.1, \quad \varepsilon_\ell = 0.001$$

and used DISOPT [12] to perform the optimization.

Initial approximate centering

The scheme of Fig. 3 can directly be used to solve the test problem. All subsequent results were found with vertex testing scheme 2, assuming related parameters. The box labelled optimization in Fig. 3 consists of 2 steps: a feasibility check interrupted when all constraints are satisfied and the main optimization process.

When using the scheme of Fig. 3, care must be taken to ensure proper convergence. New constraints will be selected even when the solution is already reached. The optimization will needlessly be restarted. This is due to the fact that the vertex selection method selects more vertices than strictly the worst. To get proper convergence, the amendment of limited vertex insertion is needed. A newly selected vertex is only added to the constraint table if it is worse than the vertices already selected for that particular frequency.

With this amendment to the basic scheme, the solution of the test problem is found in 86 sec on the CDC 6400. Three optimizations were needed. Details of the process are shown in Table VIII. It is clear from Table VIII that the first feasibility check brings us close to the solution. In fact, it replaces approximately the steps 1 and 2 of Fig. 1. The next optimization and feasibility check do not improve the solution essentially.

These considerations lead us to a scheme with initial approximate centering of Fig. 9. We do the feasibility check only as long as new violated constraints are found by the vertex selection. Its performance is illustrated by Table IX. Only two optimizations and 71 sec on the CDC 6400 were needed.

Purging schemes

From Tables VIII and IX it is clear that in each optimization there is still a relatively large number of constraints. We would like to reduce them. This is especially important if we go on to the tolerancing-tuning problem, where the number of slack variables increases with the number of constraints. Uncritical constraints should then definitely be avoided, because they lead to variables, which are indifferent to the optimization. Because we can solve the nonlinear programming problem only approximately, these indifferent variables will slightly interfere in the optimization and can make the whole process highly inefficient. In some examples about half the time needed for optimization was spent in changing the indifferent variables, without any improvement in the solution. For these reasons, the constraint table should be purged before each optimization.

The following basic purging scheme was used. The user specifies a purging percentage P. Let g_u and g_ℓ be the highest and the lowest constraint values, respectively, taken over the whole constraint table (and also over all frequencies). The purging limit is then defined as

$$g_c = g_u - .01\ P\ (g_u - g_\ell). \tag{20}$$

All constraint values greater than g_c are purged. The following amendment has to be adopted to ensure meaningful purging.

1st amendment

The user specifies a maximum constraint value g_m. This can be chosen rather pessimistically. All constraints higher than g_m are not considered in determining g_u and are therefore also automatically purged. This amendment is especially useful for eliminating completely uncritical sample frequencies. In our example we used $g_m = 10$, i.e., we consider that all constraints larger than 10 dB will never be critical.

We would like to use high purging percentages in order to limit the number of constraints as much as possible. We should, however, prevent too much purging, which might make the optimization unbounded. In order to ensure that there remain enough constraints to limit the solution, we need the following.

2nd amendment (minimal band requirements)

The user specifies frequency bands and the minimum number of constraints needed per band to limit the solution. The program will then make sure that these minimal requirements are met. If the user does not know very much about the critical frequency bands for his problem, he can specify them conservatively. This amendment anyway allows the user to use his insight and knowledge of the problem. The more detailed his knowledge is, the higher the P he can use, the lower the minimum number of frequency points per band and the more efficiently the program will run. Even if the designer has no knowledge at all, there are usually some obvious frequency bands.

The minimal band requirements used in our test problem can be found in Table X. Such band requirements can eventually be found by roughly drawing or calculating an equi-ripple response of the desired order, which gives some idea about the critical frequencies.

The basic purging scheme described is incorporated in purging process 1, as shown in Fig. 10. There is a safeguard against too much purging. A negative purging limit may indicate that P is too high. In that case, P is decreased and the last step is discarded.

The performance of purging process 1 in the test problem, with 10% purging, is illustrated in Table XI. The 2nd amendment was not used, while the 1st amendment eliminated the sample point at 10 GHz. The performance with P = 30% is shown in Table XII. The second amendment was used in the first purging step, to keep one constraint in the stopband (band 4).

The initial centering steps in Tables XI and XII are the same as in Table IX. Higher purging percentages were less efficient, because the program decreased P again, and the discarded optimizations were lost. We would still like to use higher purging percentages P. If we could find the solution with a larger number of optimizations, but with a lower number of constraints per optimization, such that the total time needed remains the same as in purging process 1, the process would work more efficiently in the tolerancing-tuning problem. Each optimization will run far more efficiently with fewer constraints. The discarded optimizations are equivalent to the test runs but action will be taken automatically. This leads to the idea of adaptive minimal band requirements, as realized in purging process 2, shown in Fig. 11.

The minimal band requirements are adapted as follows: if any vertex purged before the previous optimization reappears again in the vertex selection with a negative constraint value we discard the last optimization. We make sure that the vertex will not be purged in the next purging step and increase the minimum number of constraints required for that particular band by one. Moreover, in the next purging step, the vertex that has unjustifiably been purged

will not be counted towards determining whether the minimum band requirements are met. This means that one vertex more has to be found for that band.

Another difference with purging process 1 is the treatment of negative purging limits g_c. If g_c is negative we do not discard the previous optimization but we increase g_c. We also make sure that the new g_c is positive, so that the constraints which were active in the last optimization are not purged.

The performance of purging process 2 is illustrated in Table XIII, for P = 90%. It is slightly more efficient than purging scheme 1 with 30% purging. (One more optimization is needed.) It is, however, expected to be more efficient when we introduce tuning, as can be seen from the number of constraints in each optimization.

Some results on tuning are summarized in Table XIV. Tuning was introduced by two symmetrically placed tuning capacitors, C_{t_3} and C_{t_4} (Fig. 12), allowed to vary from 0 to 10 pF. The nominal minimax approximation and the minimax approximation with fixed tolerances gave the same results as before with tuning capacitors + 0, as was expected. The automatic process with purging was also implemented and, as was claimed above, purging process 2 was more efficient than process 1. Process 2 with P = 90% found the solution in 1025 sec on a Siemens 4004, while process 1 with P = 30% did not find a solution in less than 1200 sec. It is clear from these figures that a high purging percentage is essential, if no good starting point is available.

It should also be mentioned that the vertex selection was done with $C_{t_3} = C_{t_4} = 5$ pF, since the actual values of the tuning capacitors for each vertex are not known a priori. That means that we cannot completely rely on the convergence of the process and that testing of the final results (all vertices)

is necessary. It turned out that one more optimization was needed.

Different starting points gave slightly different results, which indicates that the tolerancing-tuning problem is extremely ill-conditioned. At any rate, the usefulness of the ideas involved in automatic efficient tolerancing for the tolerancing-tuning problem was fully illustrated: a problem was solved which was impractical by conventional means in one computer run. Further investigation of the peculiarities of the tuning problem is still needed.

V. CONCLUSIONS

An approach to automating the optimal assignment of component tolerances along with centering has been presented. A new vertex selection method for the tolerance region has been proposed with the aim of avoiding intermediate testing of all vertices. It is felt that a program based on this work would exploit the insight of a designer to improve computational efficiency without requiring detailed knowledge of the program or the algorithms used. The information required of the designer relates to the physical properties of the problem.

REFERENCES

[1] J.F. Pinel and K.A. Roberts, "Tolerance assignment in linear networks using nonlinear programming", IEEE Trans. Circuit Theory, vol. CT-19, Sept. 1972, pp. 475-479.

[2] J.F. Pinel, "Tolerance assignment and network alignment of linear networks in the frequency domain", IEEE Short Course on Computer Aided Network Design, 73-SC-06, Mar. 1973, pp. 17-25.

[3] J.F. Pinel, K.A. Roberts and K. Singhal, "Tolerance assignment in network design", Proc. IEEE Int. Symp. Circuits and Systems (Newton, Mass., April 1975), pp. 317-320.

[4] B. Karafin, "The general component tolerance assignment problem in electrical networks", Ph.D. Thesis, University of Pennsylvania, 1974.

[5] J.W. Bandler, "Optimization of design tolerances using nonlinear programming", J. Optimization Theory and Applications, vol. 14, July 1974, pp. 99-114.

[6] J.W. Bandler and P.C. Liu, "Automated network design with optimal tolerances", IEEE Trans. Circuits and Systems, vol. CAS-21, Mar. 1974, pp. 219-222.

[7] J.W. Bandler, P.C. Liu and J.H.K. Chen, "Worst case network tolerance optimization", IEEE Trans. Microwave Theory Tech., vol. MTT-23, Aug. 1975.

[8] J.W. Bandler, P.C. Liu and H. Tromp, "Practical design centering, tolerancing and tuning", Proc. IEEE Int. Symp. Circuits and Systems (Newton, Mass., April 1975), pp. 206-209.

[9] P.C. Liu, "A theory of optimal worst-case design embodying centering, tolerancing and tuning, with circuit applications", Ph.D. Thesis, McMaster University, 1975.

[10] N. Marcuvitz, Waveguide Handbook, M.I.T. Rad. Lab. Ser., vol. 10. New York: McGraw-Hill, 1951.

[11] J.W. Bandler, C. Charalambous and J.H.K. Chen, "MINOPT - an optimization program based on recent minimax results", Proc. Midwest Symp. Circuits and Systems (Montreal, Canada, Aug. 1975), pp. 520-524.

[12] J.W. Bandler and J.H.K. Chen, "DISOPT - a general programme for continuous and discrete non-linear programming problems", Int. J. Systems Science, vol. 6, 1975, pp. 665-680.

TABLE I DATA FOR LOWPASS FILTER

Band edge	1.0	GHz
$Z_1^0 = Z_3^0 = Z_5^0$	0.2	Ω
$Z_2^0 = Z_4^0$	5.0	Ω
ε_{Z_i}, i = 1,...,5	2	%
$C_1 = C_6$	15.6	pF
$C_2 = C_3 = C_4 = C_5$	28.5	pF
Terminations	1.0	Ω

TABLE II SPECIFICATIONS FOR LOWPASS FILTER

Band	Insertion loss specification
0 - 1 GHz	0.02 dB
2.5 - 10 GHz	25.0 dB
Parameter constraints	$\ell_1^0 = \ell_5^0$
	$\ell_2^0 = \ell_4^0$
	$\varepsilon_{\ell_1} = \varepsilon_{\ell_2} = \varepsilon_{\ell_3} = \varepsilon_{\ell_4} = \varepsilon_{\ell_5} = \varepsilon_\ell$
Cost function	$\frac{1}{\varepsilon_\ell}$

TABLE III RESULTS FOR LOWPASS FILTER

Problem	$\ell_1^0 = \ell_5^0$	$\ell_2^0 = \ell_4^0$	ℓ_3^0	ε_ℓ
Nominal minimax approximation	.0343	.1440	.1207	-
Minimax approximation with fixed tolerances	.0402	.1433	.1252	.001
Tolerance optimization	.0423	.1426	.1274	.00196

TABLE IV VERTICES USED IN MINIMAX APPROXIMATION WITH $\varepsilon_\ell = 0.001$ FIXED (LAST RUN)

Frequency GHz	Vertices when using symmetry†	Additional vertices when not using symmetry
.25	22	-
.35	22	-
.40	22	-
.50	22	-
.65	119,460,507,508	475,782,972,988
.75	476	906
.80	243,476	906
.85	243,476	906
.90	212,243	410,906
.95	119,243	782,906
1.0	615	813
2.5	673	-
10.0	687	-

TABLE V DATA FOR LAST RUN IN SOLUTION OF MINIMAX APPROXIMATION WITH $\varepsilon_\ell = 0.001$ FIXED

	Not using symmetry	Using symmetry
Number of variables	5	3
Number of constraints	32	20
CPU time (sec)	36.0	18.6
Number of function evaluations	252	185

† Vertices are numbered as $r = 1 + \sum_{j=1}^{k} \left[\frac{\mu_j^r + 1}{2} \right] 2^{j-1}$, $\mu_j^r \in \{-1, 1\}$.

TABLE VI VERTEX SELECTION AT OPTIMALLY TOLERANCED SOLUTION OF LOWPASS FILTER†

Frequency GHz	Nominal constraint value	Vertices satisfying necessary condition		Vertices selected by one-at-a-time scheme		One iteration from nominal point
		No.+	Constraint value	Satisfying necessary condition	Number	
.25	.0140	22	.00671	22	1	22
.35	.0123	22	.00107	22	1	22
.40	.0124	22	.00000145	22	1	22
.50	.0146	22	.00213	22	1	22
.65	.0193	53	.01095	53	36	-
		119	.01144	-		-
		549	.01147	549		549
		507	.01786	-		-
		476	.01851	-		-
.75	.0199	243	.009968	243	544	-
		119	.01060	119		-
		476	.01242	476		476
		507	.01341	-		-
		549	.01522	549		-
		53	.01560	53		-
.80	.0195	243	.006660	-	36	-
		476	.008507	476		476
		119	.009267	-		-
		549	.01638	-		-
.85	.0190	243	.003388	-	10	-
		476	.005286	476		476
		119	.007627	-		-
		549	.01689	-		-
.90	.0189	243	.0009800	-	10	-
		212	.001882	212		-
		476	.004051	476		476
		119	.005923	-		-
		549	.01662	-		-
.95	.0194	212	.0000004761	212	20	-
		243	.0001738	-		-
		119	.003988	-		-
		476	.005918	476		476
		549	.01506	-		-
1.0	.0200	615	.000001250	615	544	-
		119	.0005760	119		-
		212	.0006256	212		-
		243	.001076	243		-
		549	.01093	549		549
		476	.01108	476		-
		879	.01144	879		-
2.5	1.507	673	.00007964	673	1	673
10.0	28.31	687	25.98	687	1	687

† Of each pair of symmetric vertices, only the lowest vertex number is mentioned. Also in column 6, the figure mentioned is the number of essentially different vertices.

+ Underlined vertex numbers denote vertices in the plane of symmetry.

++ At 0.80 GHz the one-at-a-time scheme found vertex no. 211 as worst vertex. It does not satisfy the necessary condition.

Vertex selection assuming related parameters				Vertex selection not assuming related parameters, testing scheme 2	
Testing scheme 1		Testing scheme 2			
Vertices	FE	Vertices	FE	Vertices	FE
22	5	22	5	22	8
22	5	22	5	22	8
22	5	22	5	22	8
22	5	22	5	22	8
53	12	53	12	53	24
119		119		119	
549		549		549	
-		-		-	
-		-		-	
243	15	243	15	243	24
119		119		-	
476		476		476	
507		507		507	
549		549		-	
53		53		-	
-	5	243	11	-	8
476		476		476	
-		119		-	
-		-		-	
-	5	243	11	243	16
476		476		476	
-		119		-	
-		-		-	
243	14	243	15	243	24
212		212		212	
476		476		476	
119		119		-	
-		-		-	
212	14	212	14	212	24
243		243		243	
119		119		-	
476		476		476	
-		-		-	
615	16	615	17	615	32
119		119		119	
212		212		212	
243		243		243	
549		549		549	
476		476		-	
879		879		-	
673	5	673	5	673	8
687	5	687	5	687	8

FE = number of function evaluations (in this case calculation of network response and gradient at a single frequency)

TABLE VII PERFORMANCE OF VERTEX SELECTION METHODS FOR OPTIMALLY TOLERANCED LOWPASS FILTER

	No. of sample frequencies where failure	CPU time on CDC 6400 (sec)	No. of vertices generated
One-at-a-time	3	12.3	1196
Iterative			
Assuming related parameters			
Testing scheme 1	2	1.65	32
Testing scheme 2	0	2.07	36
Not assuming related parameters			
Testing scheme 2	1	3.09	26

TABLE VIII PERFORMANCE OF BASIC AUTOMATIC SCHEME IN TEST PROBLEM

Step	No. of constraints in optimization	Resulting parameters $\ell_1^0=\ell_5^0$	$\ell_2^0=\ell_4^0$	ℓ_3^0	ε_ℓ	CPU time on CDC 6400 (sec)
Vertex selection						
Feasibility check	12	.0394	.1400	.1300	.00034	9.6
Optimization		.0283	.1683	.1310	.00653	
Vertex selection						
Feasibility check	37	.0380	.1477	.1235	.00161	
Optimization		.0416	.1442	.1287	.00254	31.7
Vertex selection						
Feasibility check	47	.0410	.1439	.1249	.00156	
Optimization		.0423	.1426	.1274	.00196	40.1
Total CPU time = 86 sec						

TABLE IX PERFORMANCE OF INITIAL APPROXIMATE CENTERING SCHEME IN TEST PROBLEM

Step	No. of constraints in optimization	Resulting parameters $\ell_1^0=\ell_5^0$	$\ell_2^0=\ell_4^0$	ℓ_3^0	ε_ℓ	Effort FE = function evaluations, CPU times on CDC 6400
Vertex selection Feasibility check	12	.0394	.1400	.1300	.00034	22 FE
Vertex selection Optimization	24	.0427	.1430	.1268	.00217	33.5 sec
Vertex selection Feasibility check Optimization	40	.0421 .0423	.1427 .1426	.1271 .1274	.00186 .00196	31.2 sec
Total CPU time = 70.7 sec						

TABLE X MINIMAL BAND REQUIREMENTS FOR PURGING IN TEST PROBLEM

Band No.	Frequencies (GHz)	Minimum number of constraints
1	0 - 0.55	1
2	0.60 - 0.96	1
3	0.97 - 1.0	1
4	2.5 -10.0	1

TABLE XI PERFORMANCE OF PURGING SCHEME 1, P = 10%, IN TEST PROBLEM

Step	Number of constraints	Resulting parameters $\ell_1^0=\ell_5^0$	$\ell_2^0=\ell_4^0$	ℓ_3^0	ε_ℓ	CPU time (sec) CDC 6400	Siemens 4004
Initial centering	24	.0427	.1430	.1268	.00034		
Purging							
Optimization	18	.0437	.1438	.1252	.00270	20.1	89.5
Vertex selection	32						
Purging							
Feasibility check	30	.0422	.1427	.1273	.00193	22.9	96.2
Optimization		.0423	.1426	.1274	.00196		
	Total					49.3	212.9

TABLE XII PERFORMANCE OF PURGING SCHEME 1, P = 30%, IN TEST PROBLEM

Step	Number of constraints	Resulting parameters $\ell_1^0=\ell_5^0$	$\ell_2^0=\ell_4^0$	ℓ_3^0	ε_ℓ	CPU time (sec) CDC 6400	Siemens 4004
Initial centering	24	.0427	.1430	.1268	.00034		
Purging- second amendment used in band 4.							
Optimization	18	.0437	.1438	.1252	.00270	20.1	89.5
Vertex selection	32						
Purging							
Feasibility check	24	.0423	.1424	.1274	.00188	18.9	79.8
Optimization		.0423	.1426	.1274	.00196		
	Total					45.3	188.6

TABLE XIII PERFORMANCE OF PURGING PROCESS 2, P = 90%, IN TEST PROBLEM

Step	Number of constraints	Resulting parameters $\ell_1^0=\ell_5^0$	$\ell_2^0=\ell_4^0$	ℓ_3^0	ε_ℓ	CPU time (sec) CDC 6400	Siemens 4004
Initial centering	24	.0427	.1430	.1268	.00034		
Purging Optimization	6	.0361	.1633	.1078	.00361	9.02	34.9
Band 4, minimum from 1 to 4, discard.							
Purging Optimization	12	.0427	.1430	.1268	.00217	10.2	39.8
Vertex selection	30						
Purging (increased g_c)							
Feasibility check	13						
Optimization		.0423	.1426	.1274	.00196	10.9	41.5
	Total					38.3	148.8

TABLE XIV PERFORMANCE OF AUTOMATIC SCHEMES ON TEST PROBLEM

	Basic scheme	Approximate initial centering	Purging 1, 30%	Purging 2, 90%
No. of vertex selections	4	4	4	4
No. of feasibility checks	3	2	2	2
No. of optimizations	3	2	2	2
No. of constraints per optimization	12,37,47	24,40	18,30	6,12,13
CPU time(sec) CDC 6400	86.0	70.7	45.3	38.3

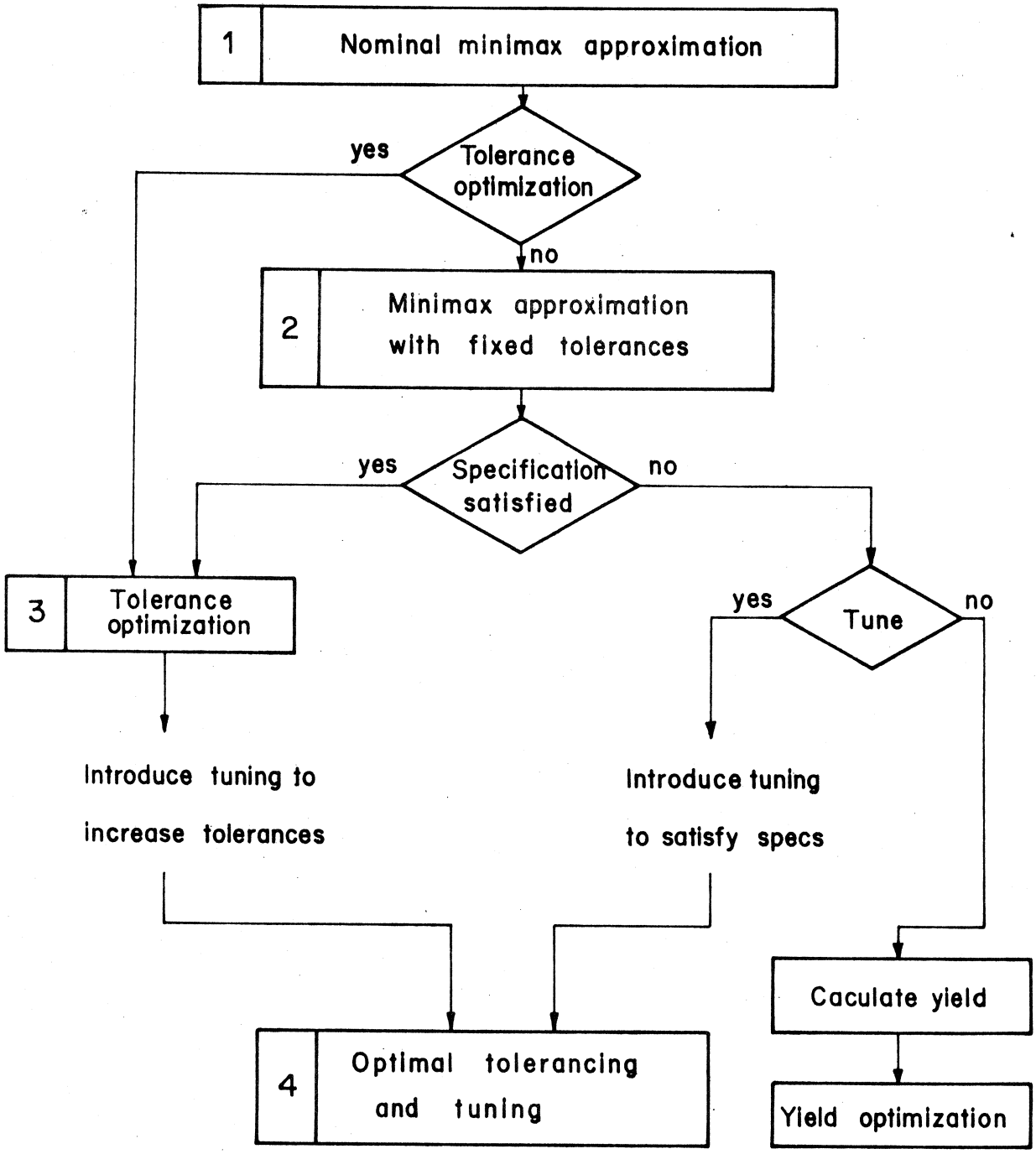

Fig. 1. Typical problem solving sequence.

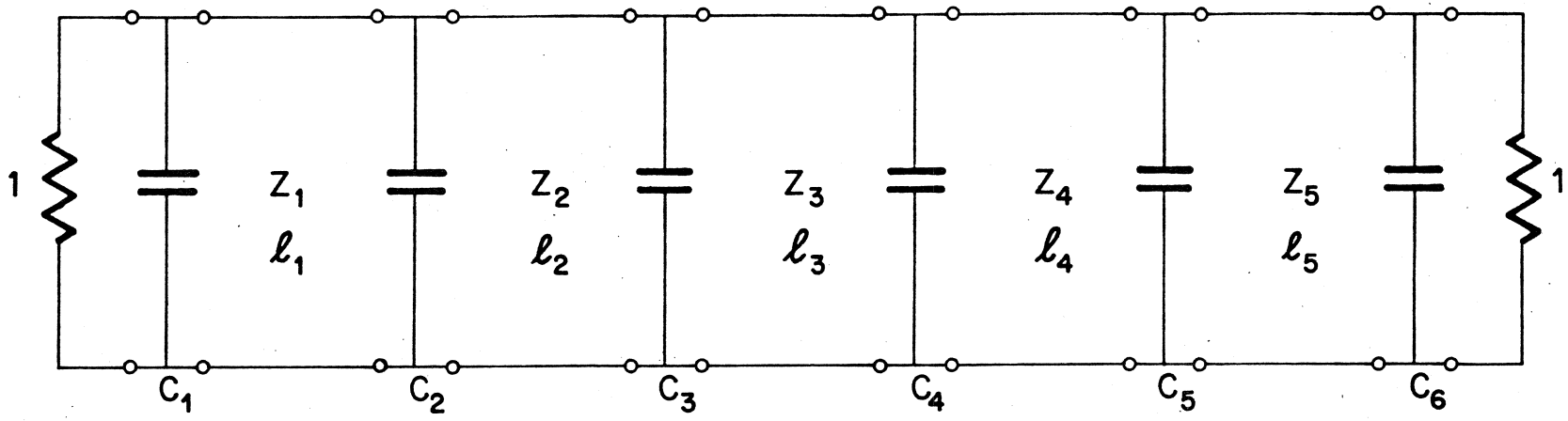

Fig. 2. Five-section transmission-line lowpass filter.

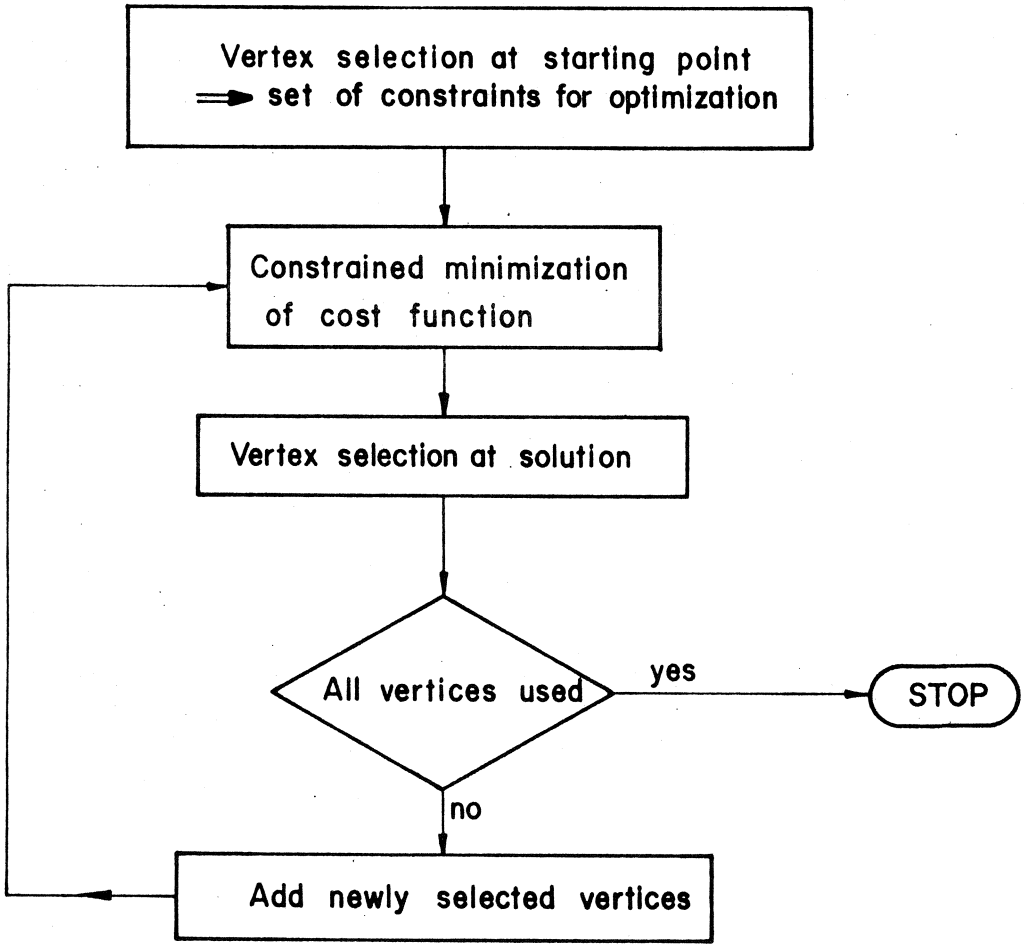

Fig. 3. Basic scheme for tolerance optimization.

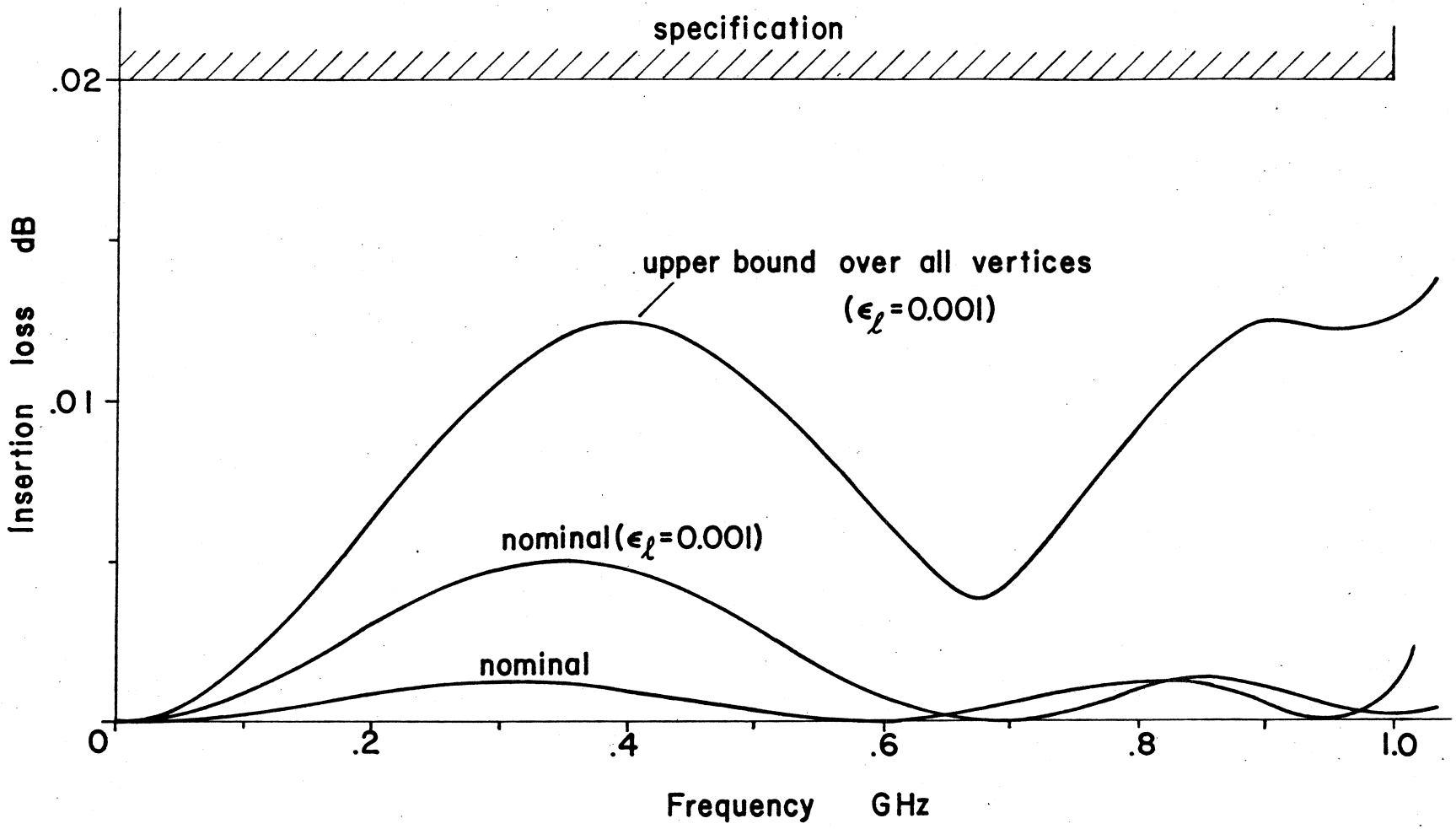

Fig. 4. Minimax approximation (passband) with no tolerance and with fixed tolerances.

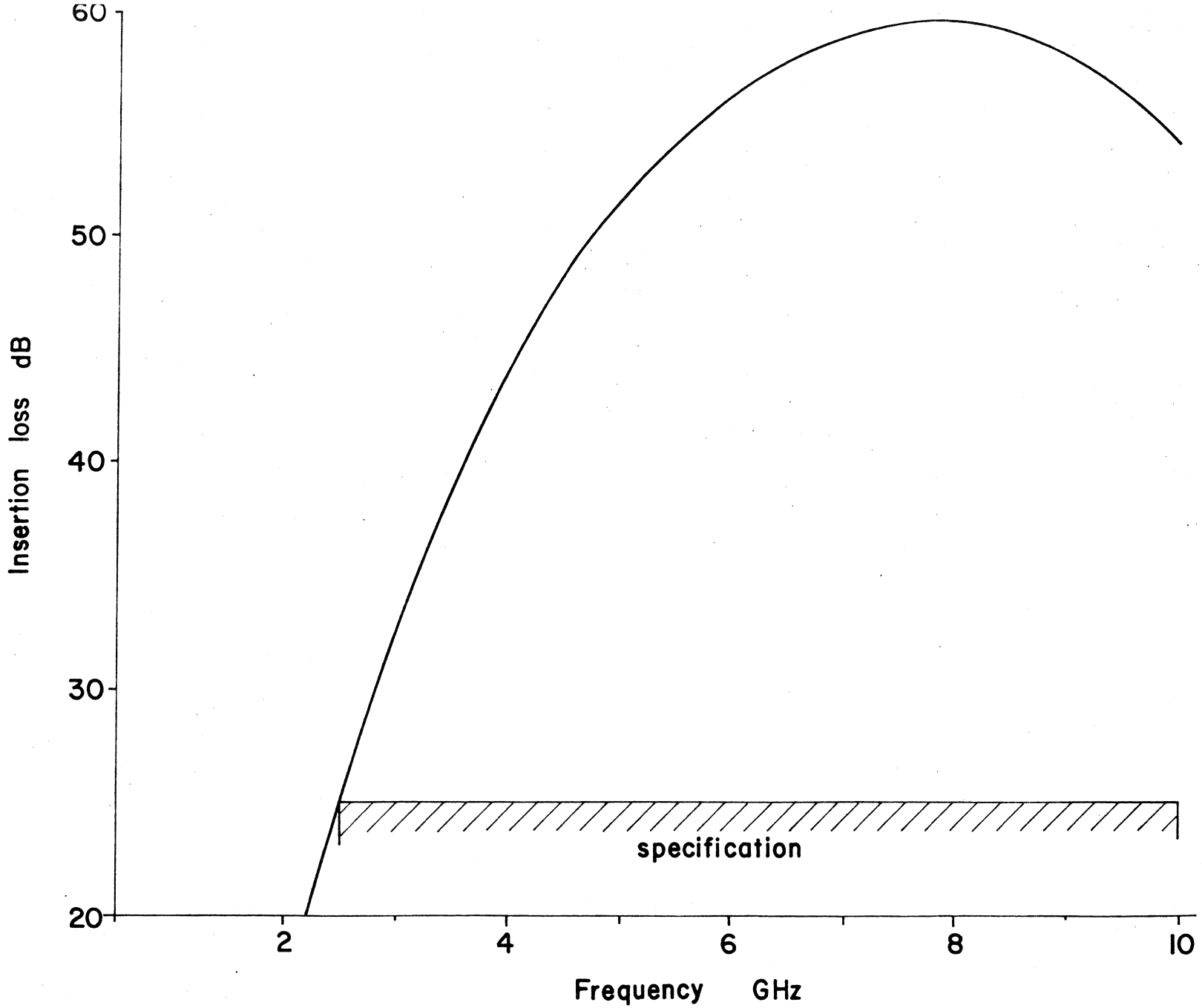

Fig. 5. Stopband response of the nominal minimax approximation without tolerances.

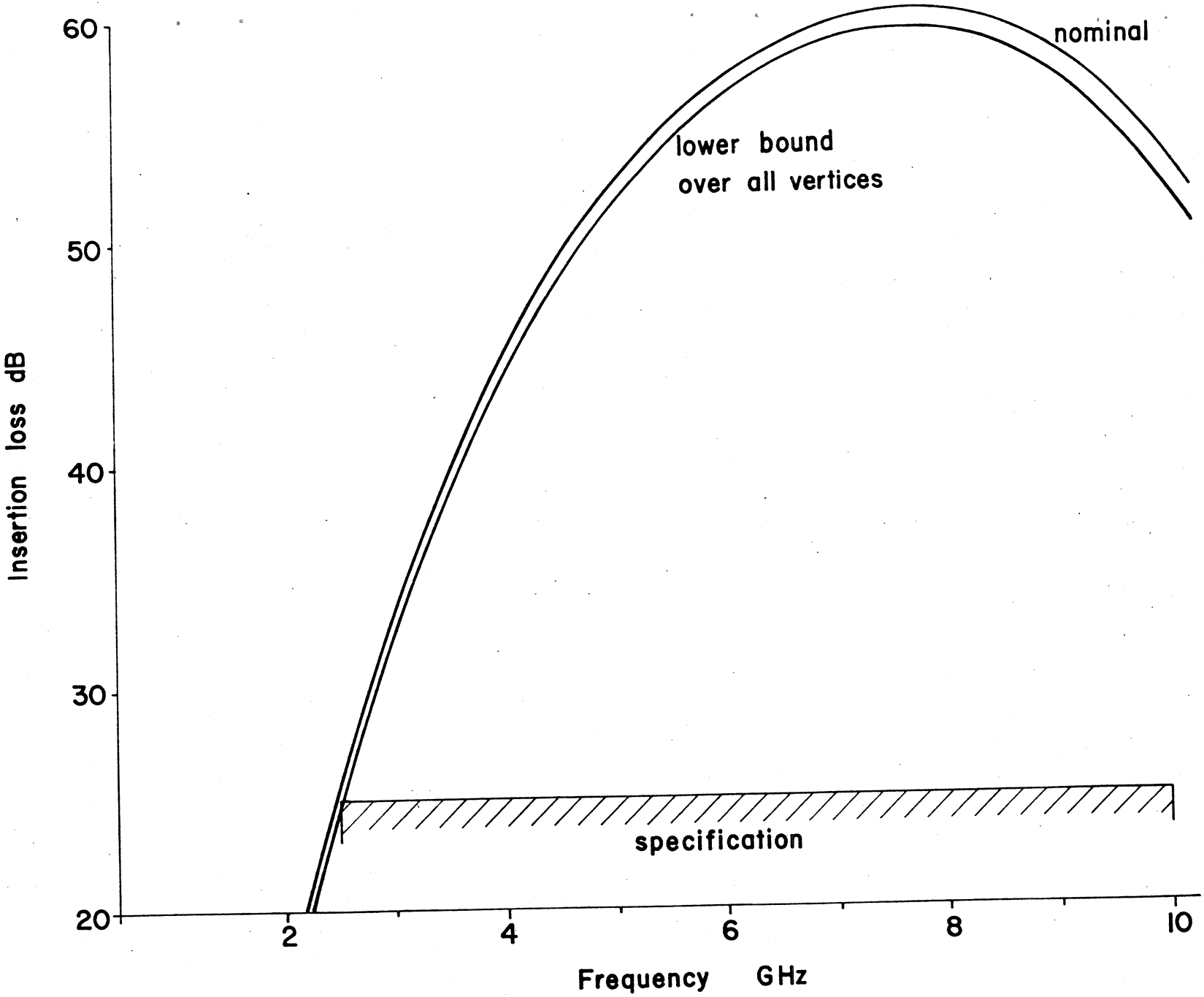

Fig. 6. Stopband response of the minimax approximation with fixed tolerances ($\varepsilon_\ell = 0.001$).

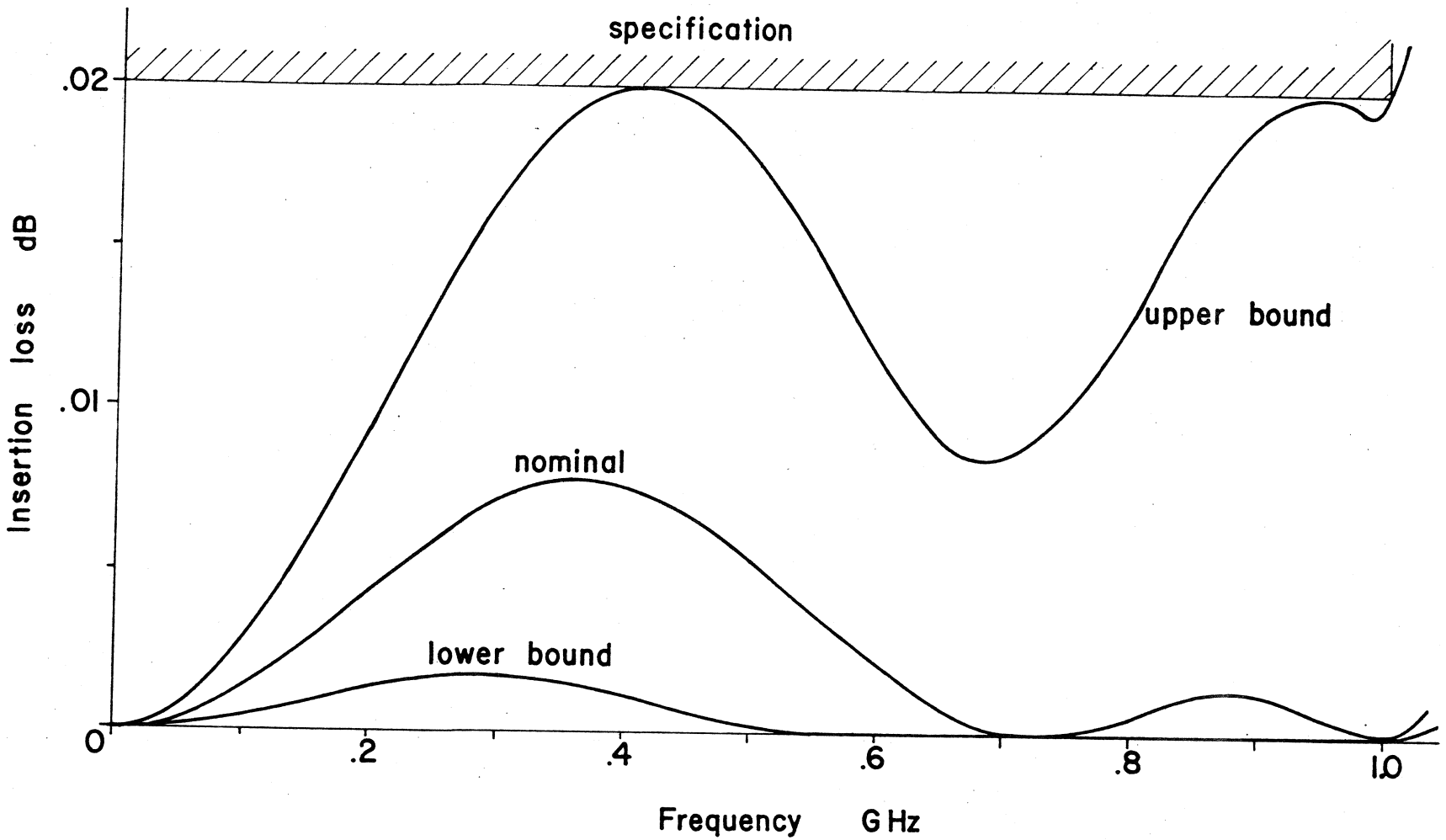

Fig. 7. Passband response of the optimally toleranced filter.

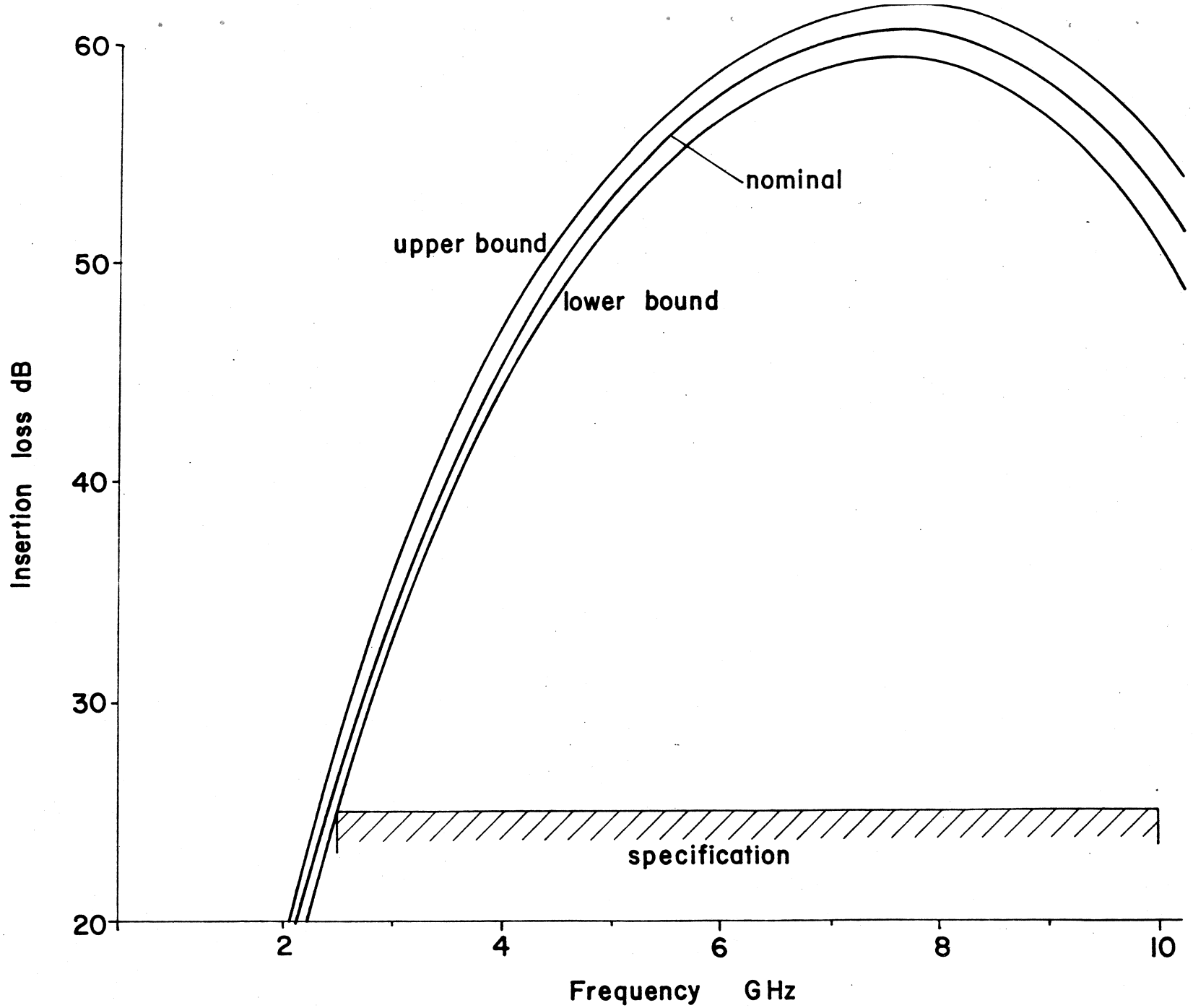

Fig. 8. Stopband response of the optimally toleranced filter.

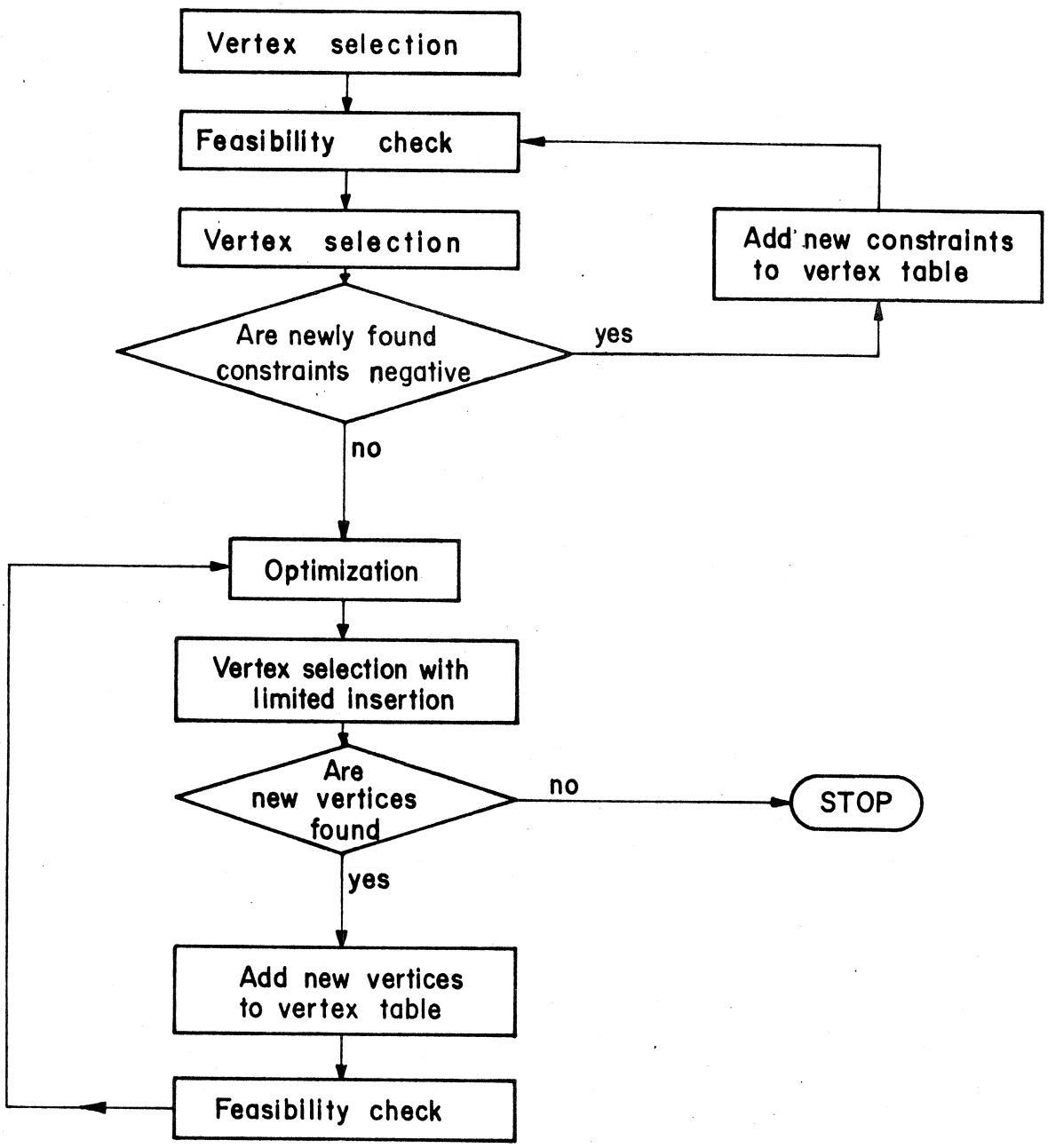

Fig. 9. Approximate initial centering scheme.

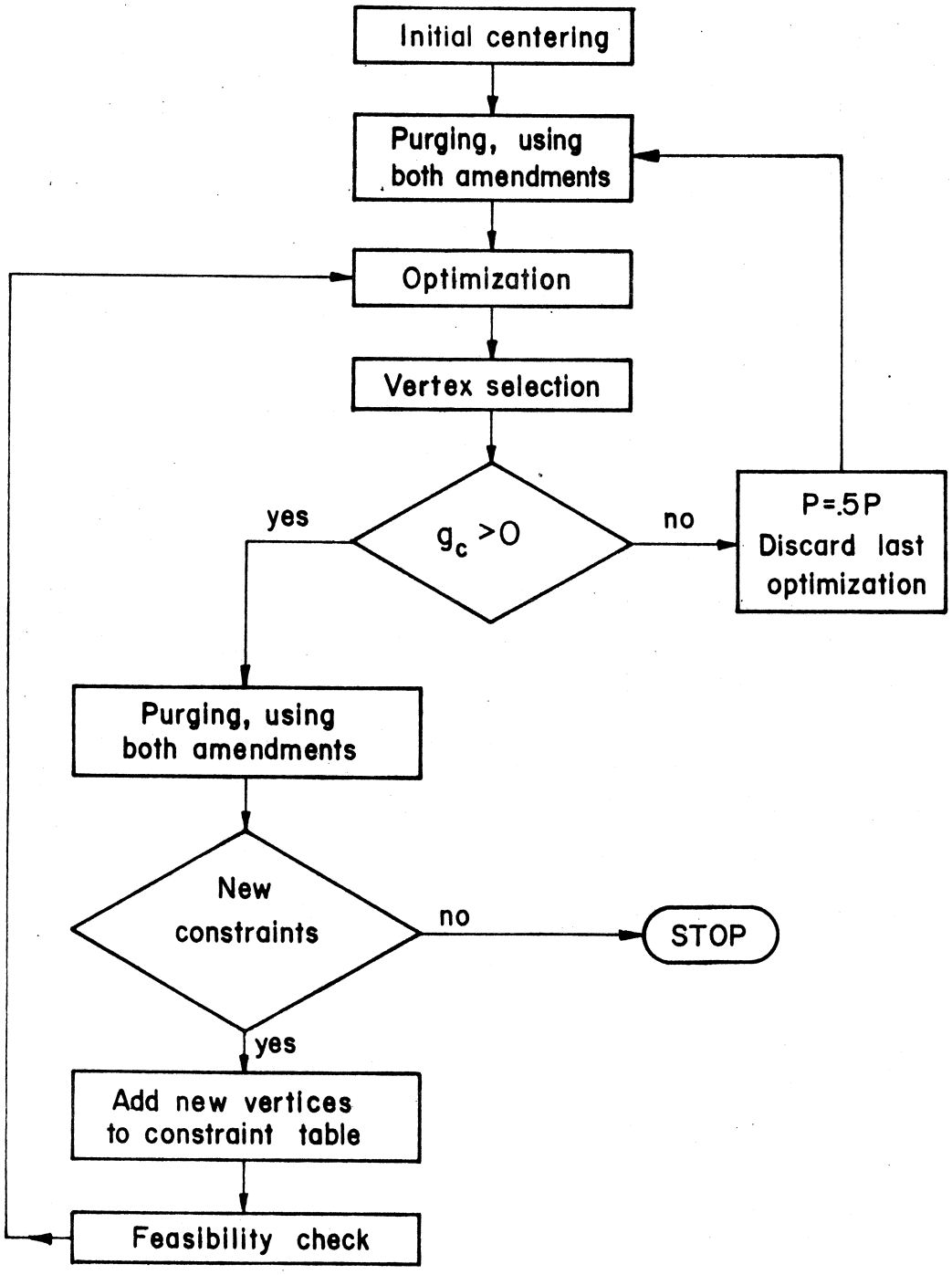

Fig. 10. Purging process 1.

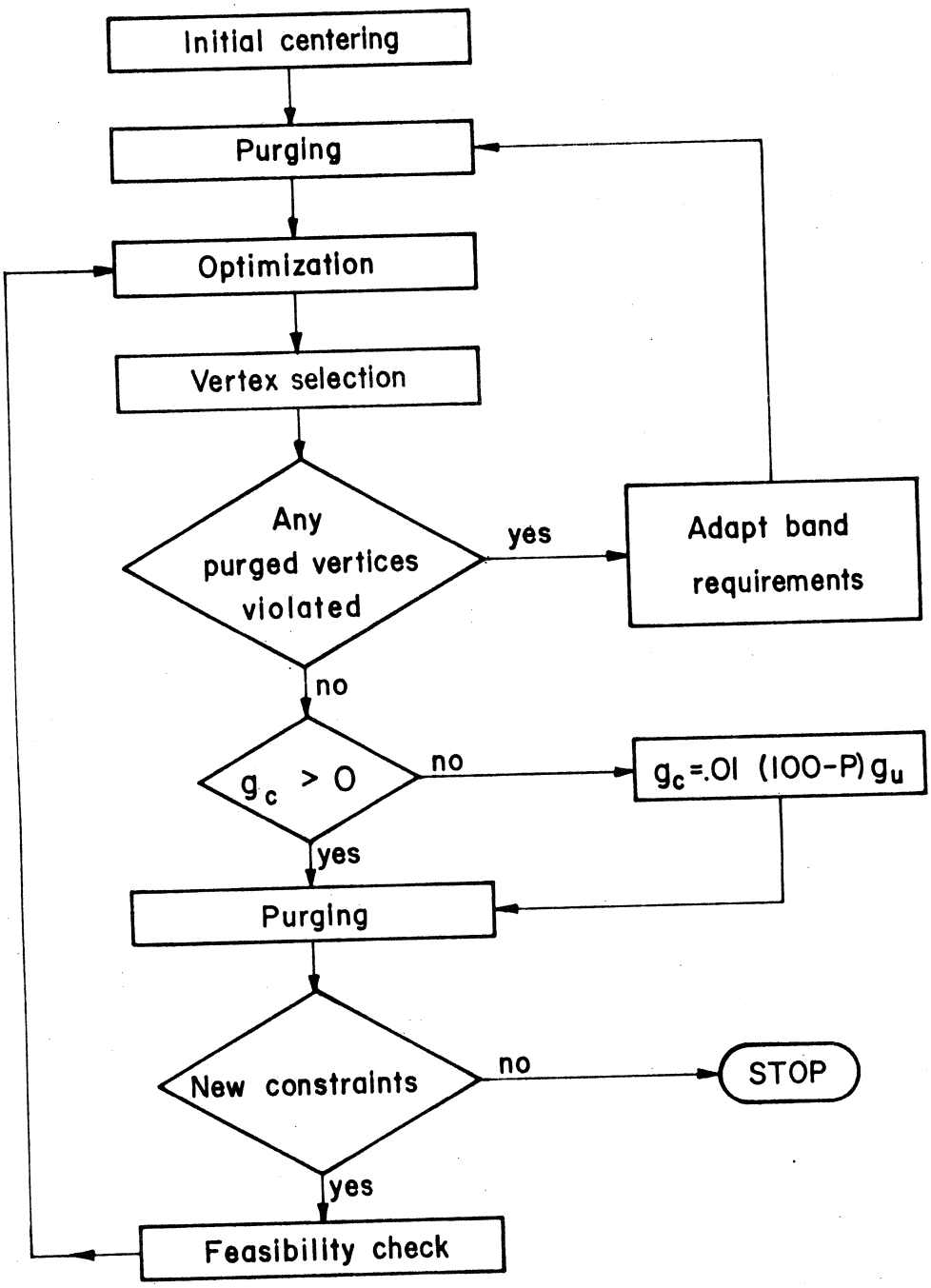

Fig. 11. Purging process 2.

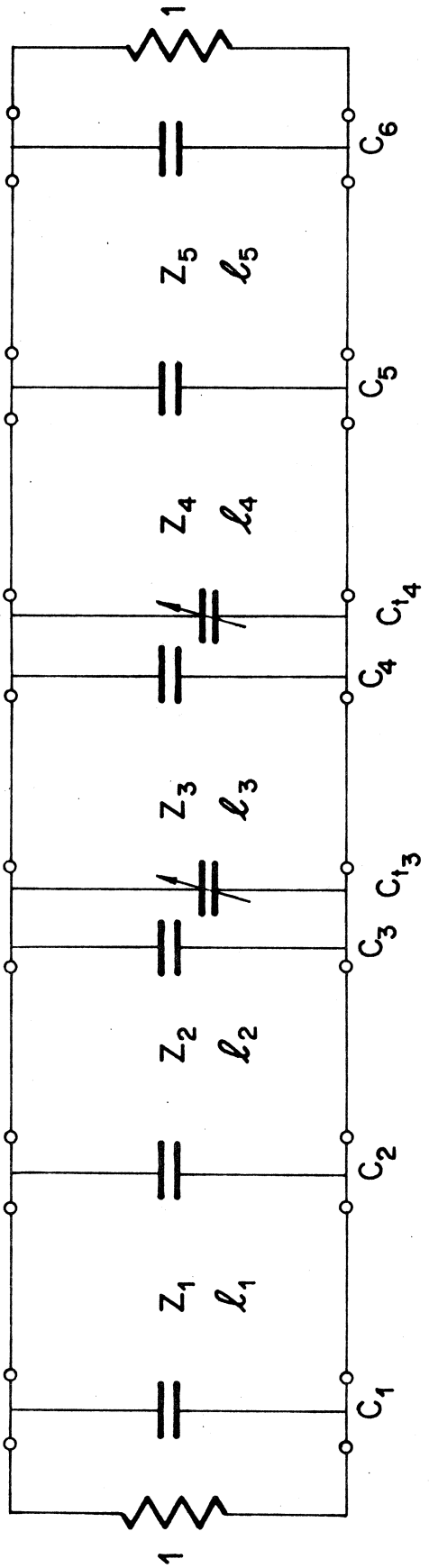

Fig. 12. The filter with tuning capacitors.