TREIZIÈME COLLOQUE GRETSI - JUAN-LES-PINS DU 16 AU 20 SEPTEMBRE 1991

SIPREX: AN EXPERT SYSTEM FOR DIGITAL FILTER DESIGN

J. M. de CARVALHO
M. C. PEQUENO
A. M. SILVA FILHO
M. T. HATTORI

Laboratório de Automação e Processamento de Sinais - LAPS
Departamento de Engenharia Elétrica
Departamento de Computação - UFC
Departamento de Sistemas e Computação
Universidade Federal da Paraíba
Caixa Postal 10105
58.100, Campina Grande, PB, Brazil

RÉSUMÉ

Résumé. Le programme SIPREX est un système expert qui integre le calcul numérique et symbolique dans an outil intelligent destiné a l'usage des ingénieurs de traitement numérique des signaux. Le système opére de façon similaire a un expert sur les techniques de synthèse et d'analyse de filtres numérique qui choisit les méthodes et les parametres importantes pour une application particulieère. Le programme SIPREX est capable d'orienter l'utilisateur pendant l'elaboration du cahier des charges e même de deécider par lui même a partir des informations sur les applications du traitement numérique de signaux déjà stockes dans la base de connaissance. Commme résultat l'execution du programme SIPREX, le filtre numérique obtenu est celui qui s'adapte a l'application en question.

ABSTRACT

Abstract. SIPREX is an expert system which couples symbolic and numeric computation developed to be used as an intelligent tool by digital signal processing engineers. The system works as an expert on different techniques for analysis and synthesis of digital filters, deciding on the choices of methods and trade-offs among relevant design parameters. SIPREX is also capable of directing the user, or even deciding by him, on the choice of design specifications for the problem at hand, based on information about application areas of DSP available in its knowledge base. The final result is a filter design which represents the best solution for the problem at hand.

1. INTRODUCTION

Expert systems or, more generally, knowledge based systems, represent one of the first examples of applied Artificial Intelligence. They appeared initially in the 70's when researchers in the field started giving up the quest for machines with generalized intelligence and decided to concentrate in the solution of practical problems. Since then, this class of systems has been employed in a broad range of functions such as data interpretation, forecasting, monitoring, control, detection and correction of failures in other systems, and as a designing tool for diferent areas of engineering[1].

Of particular interest to engineers and other users of computacional mathematics have been the so called Coupled Expert Systems, which integrate symbolic and numerical computation[2]. This interest is explained by the fact that research on numerical algorithms has been concerned mainly with complexity and accuracy, and not enough with making them ease to use and understand. Tipically, it is left to the users of numerical methods the choice of the best algorithm to be used in a given situation, as well as the analysis and interpretation of the results obtained. To be performed in a optimal way these tasks require a degree of specialized knowledge not always present in a tipical user and, as a result, inadequate decisions and wrong conclusions are likely to come out of the process.

One example of the situation above described is the process of specifying and designing a digital filter. Each stage of this process involves a series of decisions, like choice of critical frequencies and tolerances, phase and group delay behaviour, type of filter (FIR or IIR), structure to be used in the implementation and size of wordlenght, among others, with the objective of obtaining a final design which best suits the problem at hand[3]. Ideally, these choices would require from the designing engineer detailed knowledge about digital filters and the different techniques for synthesizing them. In practice, however, this knowledge is usually not present and the designer most of the times utilizes techniques which are familiar to him or for which friendly softwre packages are available. This situation can be easily understood if one considers that the existing methods for digital filter design use concepts and tools of numerical calculus, optimization theory and linear programming, for example, which the normal engineer is usually familiar with only in a superficial way. Consequently, the performance of the resulting filter, although sometimes acceptable, is usually way below the optimal possible for that given application.

The above example shows that a system capable of storing the knowledge of experts in the different techniques and application areas of Digital Signal Processing (DSP) and integrating it with powerful numerical algorithms, should represent a very valuable tool for the solution of problems involving analysis and synthesis of digital filters. This is the objective of SIPREX, the system presented in this work.

Tipically, a coupled system is composed of a symbolic structure, which stores all the information about the domain of the system and performs the required inference, and a numerical structure, responsible for the calculations involved in the deductive process. Complying with this model the symbolic components of SIPREX are a Control Module, a Central Module, and an Explanation Module. Additionaly, the system is equiped with a Numerical Module, containing the algorithms for filter design, analysis and optimization, and a Graphical Module for communication with the user. These modules and their interaction within the system are illustrated in Figure 1 and described in the next sections.

SIPREX communicates interactively with the user via a menu based interface, receiving data about the problem to be solved and providing, in exchange, information about the synthesis/analysis technique employed. The final design can be visually analyzed by the user via the Graphical Module through its magnitude frequency response, phase, group delay and impulse response curves. In case the result achieved is judged not satisfactory by the user SIPREX tries again with a different method, until an acceptable filter is produced.

3. CENTRAL MODULE

The Central Module of SIPREX is the intelligent part of the system, where the inference process takes place. It is organized in a blackboard architecture and composed by three main sections: Knowledge Base, Inference Unity and Work Memory, as illustrated in Figure 1. These components are described next.

3.1 Knowledge Base

The Knowledge Base is the part of an expert system which holds information about the domain of the system, usually stored in the form of rules. The Knowledge Base of SIPREX is a modular structure composed by two classes of cells, those with information about the algorithms for filter design present in the Numerical Module and those with information about application areas of Digital Signal Processing, where the filters are to be employed. New cells can be added to the structure without affecting the existing ones.

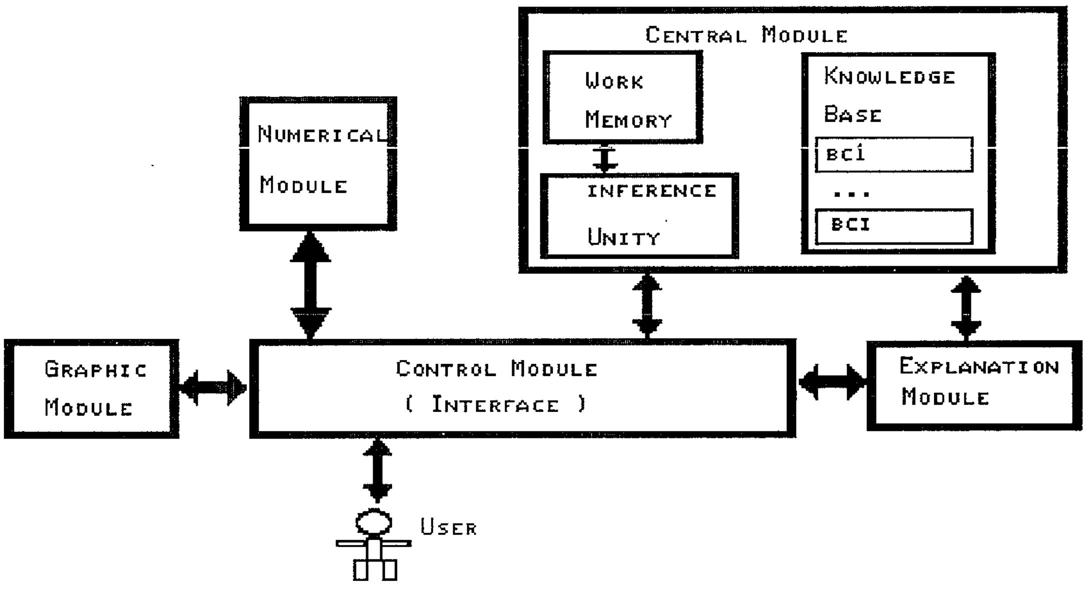

Figure 1. Structure of SIPREX

2. CONTROL MODULE

The Control Module of SIPREX activates and control the other modules in the system, as well as the flow of information between them. This is done as a function of the problem being solved, as specified by the user, and the intermediate results obtained, which are analyzed by this module. It is also responsible for interacting with the user via a friendly interface, formulating questions and proposing alternatives with the objective of obtaining an initial data set which characterizes the problem as precisely as possible. During this interaction, on-line help is available to the user for clarification about the questions being formulated. The raw data thus obtained is analyzed by the system and, through a deductive process, originates new data which is transfered to the Work Memory of the Central Module and used in the solution of the problem.

As a result of the inference process which takes place in the Central Module programs implementing the numerical algorithms will be activated by the Control Module. Finally, the Control Module presents to the user via the interface the final result (filter design) obtained. Information about the choice of algorithms and rules employed in the deductive process can be obtained by activation of the Explanation Module. The final design can also be visualized by the curves of magnitude and phase frequency response, group delay and impulse rsponse, upon request by the user and activation of the Graphical Module.

Internally, each cell in the Knowledge Base contains four files:

- Rules File: stores the rules representing the knowledge of the system.
- Counter File: holds the count of the number of conditional elements in the rules and indicates the type of conective between them.
- Influence File: indicates the conditional elements (facts and diagnostics) which have influence over a given rule.
- Priority File: indicates the priority of a given rule.

As an example, consider the following rule stored in the Rules File of a cell:

rule(4,if(filter(lowpass),structure(fir)) then save_d(fir1))

The register corresponding to this rule in the Counter File would be:

count(4,0,2)

where the numbers 4,0 and 2 correspond to the identifier of the rule, the type of conective used (0 = AND,1 = OR) and the number of conditional elements to be satisfied, respectively. In the Influence File the following registers would contain the elements influencing this rule:

inf(4, filter(lowpass))
inf(4, structure(fir)).

Finally, one register would be present in the Priority File:

pri(4,0)

where priority zero has been assigned to this rule.

3.2 Work Memory

The Work Memory is a data base which holds information related to the current problem. This includes the information provided by the user via the interface and processed by the Control Module, the corresponding cells transferred from the Knowledge Base by the Inference Unity, and the intermediate results obtained by the previous operation. The objective of this memory is to ease the task of the Inference Unity, by restricting the data field to be searched and tested during the inference process.

3.3 Inference Unity

The Inference Unity is the component of the system in charge of all the search, matching, and reasoning involved in the inference process. It is composed by three blocks: Supervisor, Scheduler and Inference Engine.

The Supervisor analyzes the contents of the Knowledge Base, loads the Work Memory with the cells containing information relevant to the current problem and activates the Scheduler for operation. It also alters the contents of the Work Memory when new facts are generated by execution of the rules. Upon activation the Scheduler tests the conditions of the rules in the Work Memory and prepares a set of executable rules ordered by priority. This set of rules is passed to the Inference Engine which uses a forward chaining strategy to generate new facts and eventually reach a diagnostic.

4. NUMERICAL MODULE

This module implements the algorithms for digital filter design, responsible for all numerical calculations involved in the process. The algorithms utilized are those selected by the Digital Signal Processing Committee of the IEEE ASSP Society which are divided in two classes: programs for Infinite Impulse Response (IIR or recursive) and programs for Finite Impulse Response (FIR or non-recursive) digital filter design and optimization[4]. The original programs were substancially modified with the objective of increasing their efficiency and robustness, and adapting them to SIPREX environment. This includes standardization of input/output data specification and format and suiting them to perform in the context of the system. Also available in the Numerical Module are a program for frequency response magnitude error analysis, another for estimating the order of a FIR filter, and a third to calculate the number and location of the poles and zeros of an IIR filter.

4.1 Programs for IIR Filter Design

Four programs are available for this purpose, each performing a distinct but complementary task. Program IIR1 produces an initial design from the user specifications. The output of IIR1 is submitted to program IIR2 to produce a more refined filter. IIR2 can also be used for group delay equalization of an already existent filter. In case further refinement is needed the output of IIR2 is submitted to program IIR3. The last program, IIR4 has the job of quantizing the filters designed by previous programs to a defined wordlenght. Each of these programs is briefly described next.

Based in the DOREDI program by Dehner[4,section 6.1], IIR1 solves the classical approximation problem for designing frequency selective recursive filters of the types lowpass, highpass, bandpass and bandstop. The method used produces initially an analog design which approximates the tolerance scheme obtained from the user specifications, followed by a mapping to the z-plane by the bilinear transformation to produce the desired digital filter. Possible approximations are Butterworth, Chebyshev types I and II, and elliptic [3]. In addition to the filter coefficients, IIR1 also calcultes the number of poles and zeros and their location with respect to the unity circle in the z-plane, assuming realization as a cascade of second order sections. This information is utilized as input to IIR2.

Based on the LPIIR program by Deczky, IIR2 produces an optimal digital filter with arbitrary frequency characteristics, using a minimum weighted p-error criterium and the Fletcher and Powell optimization technique[4,section 6.2]. The quadratic error is the default (p = 2). The system function of the filter is represented in terms of its poles and zeros and the error can be minimized for either the frequency response magnitude or group delay approximation, or for joint magnitude and group delay approximation. This program can either optimize an initial design produced by IIR1 or perform phase equalization in any other filter via the auxiliary program EQUALZ.

Based on the OPTIIR program by Dolan and Kaiser, which uses the method of penalty function, IIR3 assumes that the filter is realized as a cascade of second order sections[4,section 6.3]. The program varies the coefficients of this realization untill the arbitrary frequency response magnitude specifications are met. A minimax optimization criterium is employed, i.e., if a filter is acceptable then maximize the minimum amount by which it exceeds the specifications, if it is not acceptable then minimize the maximum amount by which it fails.

Program IIR4 is based on the FWIIR program by Steiglitz and Ladendorf[4,section 6.4]. From a digital filter design with high precision coefficients it produces a filter with coefficients quantized to a finite wordlenght, which still meets the frequency response magnitude specifications. The program initially rounds the coefficients of the original filter to the given wordlenght, after which a randomized version of the Hooke and Jeeves search algorithm [5] is utilized for coefficient optimization. After convergency is obtained the program checks for poles and zeros which may have resulted outside the unity circle and replaces them by their inverse, thus preserving stability.

4.2 Programs for FIR Filter Design

Four programs are presently available in the Numerical Module of SIPREX for designing FIR (non-recursive) linear phase digital filters, operating independently from each other and to be used in distinct situations. The choice of program is made by SIPREX based on the specifications.

Program FIR1 is based on the EQFIR program by McClellan, Parks and Rabiner which produces an optimal Chebyshev approximation utilizing the Remez exchange algorithm[4,section 5.1]. The approximation is optimal under a minimax criterion where within the frequency bands of interest a frequency response is obtained which minimizes the maximum weighted approximation error. An weighting function provides for weighting the approximation errors differently in the different bands. This method can be used to design all the classical frequency selective digital filters plus differentiators and Hilbert transformers.

The popular technique of windowing for FIR filter design is implemented by the FIR2 program, based on the FWFIR program by Rabiner and McGonegal[4,section 5.2]. Seven types of window are available for this purpose, rectangular, triangular, Hamming, generalized Hamming, Hanning, Chebyshev and Kaiser for designing lowpass, highpass, bandpass and bandstop filters.

Program FIR3 is based on the MXFLAT program by Kaiser, which generates coefficients for a symmetric FIR lowpass filter with maximally-flat pass and stop bands and odd total number of terms[4,section 5.3]. The input set of tolerances specified for the filter are used by the program to determine the width and central frequency of the desired frequency response magnitude transition band. These parameters are utilized by the program to initially compute the filter order to meet the specifications. The filter coefficients are then determined by uniformly sampling the frequency response and then performing an inverse DFT on the set of samples obtained. The order of the filter is inversely proportional to the square of the desired transition bandwidth.

Finally, program FIR4 is based on the IDEFIR routine by Heute which designs a linear phase FIR filter with quantized coefficients[4,section 5.4]. It can be used for lowpass, highpass, symmetric bandpass and bandstop filters and Hilbert transformers. The program transforms the input specifications to those of a prototype lowpass filter and the minimum wordlenght which satisfies the transformed specifications is determined as a function of the error tolerance in the pass and stop bands and the transition bandwidth. Next, the filter lenght is estimated and the Parks-McClellan algorithm is utilized to obtain a lowpass design. The program works iteratively checking at each step for possible violations of the specified tolerances and making the proper adjustments when that occurs. The resulting lowpass design with quantized coefficients is finally transformed to the desired frequency selective filter.

4.3 Auxiliary Programs

Program ANALIS analyzes a filter designed by any of the other programs by calculating the mean-square error between the magnitude frequency response obtained and the desired value of this function in the pass and stopbands. The result of this analysis is used by the Control Module of SIPREX to decide whether or not activation of an optimization program is needed (in the case of IIR filters).

Program ESTINF utilizes an empiric technique due to Herrmann et. al.[6] to estimate the size of a FIR filter as a function of the transition bandwidth and tolerances in the stop and pass bands. This method can be used for lowpass, highpass, symmetrical bandpass and bandstop filters and Hilbert transformers.

Program EQUALZ is used to determine the number and location of poles and zeros of a given IIR design for which group delay equalization is desired. Output data from this program can be used as input to the IIR2 program, responsible for the equalization.

5. CONCLUSION

An initial version of SIPREX for use on 80386 type personal computers with DOS operating system is now under tests. The Knowledge Base for this version contains cells about four areas of application of digital filters: speech, audio, telecommunications and biomedical signal processing. Initial knowledge for the cells was obtained from study of literature and interviews with experts on the areas. As a result of the tests under way and further interaction with human experts this knowledge will be up dated and expanded, increasing the degree of expertise of the system. Also, the modularity of the base allows for inclusion of new cells without affecting the existing ones.

A module for automatic learning is now under development which will allow SIPREX to modify or create new rules in the Knowledge Base as a result of interaction with the user, whenever the system fails to produce a filter judged satisfactory. In this case the user is asked to explain the reasons of the refuse and a new designing approach will be tried untill an acceptable filter is obtained. The information involved in this process will originate new rules to be added to the Knowledge Base for future use in a similar situation.

Acknowledgements: This work has been partially suported by CNPq-Brazil.

REFERENCES

[1] Waterman, D. A., A Guide to Expert Systems, Addison-Wesley Publishing Company, Inc., USA, 1986.

[2] Kowalik, J.S.,Ed., Coupling Symbolic and Numerical Computing in Expert Systems, Elsevier Science Publishing Co., Amsterdam, The Netherlands, 1986.

[3] Rabiner, L. R., and Gold, B., Theory and Application of Digital Signal Processing, Prentice Hall, Inc., USA, 1975.

[4] DSP Committee, IEEE ASSP Society, Eds., Programs for Digital Signal Processing, IEEE Press, New York, 1979.

[5] Hooke, R., and Jeeves, T. A., Direct Search Solution of Numerical and Statistical Problems, Jour. Assoc. Comput. Mach., Vol. 8, N. 2, pp. 212-229, April 1961.

[6] Herrmann, O., Rabiner, and Chan, D. S. K., Practical Design Rules for Optimum Finite Impulse Response Low-Pass Digital Filters, The Bell System Tech. Jour., Vol. 52, N. 6, pp. 769-799, July 1973.