

SÍNTESE DE FILTROS ANALÓGICOS UTILIZANDO TÉCNICAS EVOLUTIVAS  
E SÓCIO-COGNITIVAS

Claudio Eduardo Csura Szendrodi

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Aprovada por:

---

Prof. Antonio Carneiro de Mesquita Filho, Dr. d'État

---

Prof. José Vicente Calvano, D.Sc.

---

Prof. José Franco Machado do Amaral, D.Sc.

---

Prof. Luiz Wagner Pereira Biscainho, D.Sc.

RIO DE JANEIRO, RJ - BRASIL

ABRIL DE 2005

SZENDRODI, CLAUDIO EDUARDO CSURA

Síntese de Filtros Analógicos Utilizando Técnicas  
Evolutivas e Sócio-Cognitivas [Rio de Janeiro] 2005

XVII, 100 p. 29,7 cm (COPPE/UFRJ, M.Sc.,  
Engenharia Elétrica, 2005)

Tese - Universidade Federal do Rio de  
Janeiro, COPPE

1. Síntese de circuitos analógicos
2. Filtros analógicos
3. Algoritmos genéticos
4. *Particle Swarm*

I. COPPE/UFRJ II. Título ( série )

*À minha esposa Patrícia e ao meu filhinho Ricardo,*

*O amor que sinto por vocês não poderia ser resumido em tão poucas linhas.*

*Mas mesmo assim, gostaria de falar do fundo de minha alma:*

*Vocês são o motivo de minhas lutas e o porto seguro de minha vida;*

*Vocês são o meu céu azul anil, sem nuvens ou turbulências, e onde meu vôo encontra a Paz de Deus; e*

*Somente vocês sabem o quanto os meus momentos de ausência, ou por estar estudando, ou por estar fazendo estes trabalho, fizeram falta...*

*Portanto, somente poderia dedicar estas palavras e este trabalho a vocês, com todo meu amor!*

*A Deus, a Jesus e a todos os meus amigos e guias espirituais,*

*Dedico este trabalho em agradecimento a vocês por toda a luz e o auxílio fraterno nos meus momentos de aflição, quando pensei que não iria conseguir cumprir esta missão!*

## **Agradecimentos**

Não gostaria de ser injusto com alguém por colocar uma ordem neste espaço de agradecimentos. Por isso, saibam que todos de alguma maneira, a sua maneira, me ajudaram durante minha vida profissional, e que se, de algum modo, consegui subir mais este degrau, foi porque tive a ajuda e o carinho de vocês:

A minha esposa Patrícia e ao meu filhinho Ricardo, que me deram muito amor e compreensão, principalmente ao final desta caminhada, quando usei um período de férias na Marinha, não para ficar com vocês, mas para finalizar este trabalho;

Ao meu falecido e querido sogro, Sr. Antônio, a minha querida Sogra, Dna. Solange, e a Avó de minha esposa, Dna. Ilka, obrigado por tudo que vocês fizeram e fazem por mim, pela Patrícia e pelo Ricardo. Sem a ajuda e o incentivo diários de vocês, nada disso teria sido possível;

Aos meus Pais, Ildi e Jorge, por terem me dado a oportunidade da vida, e aos meus Irmãos, Débora e Rafael, por tudo que aprendemos vivendo juntos;

Aos meus Professores do Liceu de Artes e Ofícios, na pessoa de meu primeiro Mestre, Professor Hélio Serafini, durante meu curso técnico, aos meus Professores do Departamento de Engenharia Eletrônica, durante minha graduação, e aos meus Professores no PEE da COPPE, durante este mestrado, por todos os ensinamentos que consegui adquirir. Em especial, gostaria de agradecer aos meus amigos e professores Luiz Wagner Biscainho, que me deu a honra de participar de minha banca, e Juarez, que sempre me deu palavras de incentivo durante o curso. Por vocês terem acreditado em mim desde o início, assinando minha carta de recomendação para o mestrado;

À Marinha do Brasil, na figura do Centro de Eletrônica da Marinha, por ter me permitido utilizar seus recursos computacionais nos períodos fora do expediente para realizar minhas simulações.

Aos Senhores Diretores, meus Comandantes e ex-Comandantes, do CETM, Capitão-de-Mar-e-Guerra Gusmão, Capitão-de-Mar-e-Guerra(RM1) Costa Neto e Capitão-de-Mar-e-Guerra Marcos José, pelo incentivo e pelas horas que fui liberado para cursar as cadeiras do mestrado. Ao Comandante Gusmão agradeço de coração a oportunidade que me foi dada pelo senhor, que sempre perguntava: "Quando você vai começar o mestrado rapaz? Não perca tempo!";

Aos Meus Vice-Diretores, Capitão-de-Fragata Araújo Motta e Capitão-de-Fragata José Augusto, que sempre souberam compreender minhas saídas mais cedo ou regressos atrasados, devido às atividades deste curso;

Aos Meus Chefes de Departamento da Produção, Capitão-de-Fragata(EN-RM1) Eliana Ferret, Capitão-Tenente(EN-RM2) Moraes Junior e Capitão-Tenente(EN) Auro, pela amizade e por acreditarem no meu trabalho, incentivando e ajudando de todas as formas possíveis para que conseguisse chegar até aqui. À Comandante Eliana Ferret que possibilitou que minha carga de trabalho burocrático fosse reduzida e que sempre lutou junto a Direção pela continuidade do meu curso, o meu agradecimento especial;

Aos meus Amigos da Divisão Técnica, CT(EN) Paulo Rocha, CT(EN) Savioli, 1T(EN) Magalde, 1SG-ET Jorge Luiz, 2SG-DT Andrade, Madjer, 3SG-ET (FAB) Leandro Peixoto, Josenilton, Valdinei, Vitor e Pedro, agradeço o companheirismo, a lealdade, e sobre tudo a amizade. Ao meu grande amigo CT(EN) Carlos Eduardo de Freitas Savioli, contemporâneo de graduação no DEL-UFRJ, companheiro do mestrado, não existem palavras que possam ser utilizadas para agradecer toda sua ajuda e

irmandade. Somente posso dizer que tenho muito orgulho de saber que você é meu AMIGO de verdade! Obrigado de coração!;

Aos Amigos de Praça D'Armas do CETM, em especial ao CC(IM) Maia, pela amizade e pelos conselhos seguros;

Aos Colegas Oficiais Intermediários e Subalternos do CETM, por terem ajudado quando precisei realizar trocas dos dias em que estava na escala de serviço;

Ao meu Amigo, Orientador, Professor, Doutor, Capitão-de-Corveta(EN) José Vicente Calvano, que conseguiu me trazer de volta ao mestrado, depois de um momento difícil, e que me disse: “Em nossa vidas, muitas “TSUNAMIS” passam! O importante é conseguirmos voltar a superfície para voltarmos a vida!”. Se não fosse sua ajuda, sua orientação séria, precisa, cuidadosa, inteligente e, acima de tudo, amiga, não teria conseguido chegar a escrever estas palavras nesta Tese. Muito obrigado, de coração!;

Ao meu Amigo e Orientador, Professor, Doutor, Antonio Carneiro de Mesquita Filho, que acreditou em mim desde o início, quando aceitou minha inscrição e me concedeu a honra de sua Orientação, me ajudando e incentivando, mesmo nos meus piores momentos, agradeço sinceramente por tudo que aprendi com o senhor!;

Finalmente, agradeço ao meu Bom Deus e ao meu Amigo Jesus, que me protegem nos momentos mais difíceis, me orientam nos momentos certos e que acalmam minha alma quando os caminhos se tornam sinuosos.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

## SÍNTESE DE FILTROS ANALÓGICOS UTILIZANDO TÉCNICAS EVOLUTIVAS E SÓCIO-COGNITIVAS

Claudio Eduardo Csura Szendrodi

Abril/2005

Orientadores: Prof. Antonio Carneiro de Mesquita Filho  
Prof. José Vicente Calvano

Programa: Engenharia Elétrica

Este trabalho apresenta um estudo comparativo do desempenho de duas técnicas de inteligência artificial, a evolutiva e a sócio-cognitiva, na resolução de problemas de síntese de filtros analógicos ativos. A técnica evolutiva é implementada por meio de algoritmos genéticos, e a sócio-cognitiva, por meio do algoritmo “*Particle Swarm*”. Ambas as técnicas são empregadas na busca de mínimos e máximos de funções-objetivo de uma ou mais variáveis para as quais não se conhece, *a priori*, sua forma analítica. Foram realizados testes comparativos de síntese de três filtros analógicos com funções de aproximação conhecidas e seus resultados são analisados.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

ANALOG FILTER SYNTHESIS USING EVOLUTIONARY AND  
SOCIOCOGNITIVE TECHNIQUES

Claudio Eduardo Csurá Szendrodi

April/2005

Advisors: Prof. Antonio Carneiro de Mesquita Filho  
Prof. José Vicente Calvano

Department: Electrical Engineering

This work presents a comparative study between the performance of two artificial intelligence techniques, namely evolvable and sociocognitive, when applied in solving issues of analog filter synthesis. The evolvable techniques are implemented by means of a genetic algorithm, while for the sociocognitive technique a particle swarm algorithm was chosen. Both algorithms are usually used when the goal is to search for the minimum or the maximum of the objective functions having one or more variables, for which there is not a known analytical form. Three well-known analog filters were synthesized and its results are then analyzed.



## Índice

<b>1. Introdução</b> .....	<b>1</b>
<b>2. Técnicas Evolutivas e Sócio-Cognitivas</b> .....	<b>3</b>
2.1. <i>Técnicas Evolutivas</i> .....	3
2.1.1. Histórico, Definições e Conceitos Básicos.....	3
2.2. <i>Técnicas Sócio-Cognitivas</i> .....	8
2.2.1. Histórico, Definições e Conceitos Básicos.....	8
2.3. <i>Comentários adicionais sobre a revisão bibliográfica e o estado da arte</i> .....	11
<b>3. A SÍNTESE AUTOMÁTICA DE FILTROS ANALÓGICOS UTILIZANDO ALGORITMOS GENÉTICOS E PARTICLE SWARM</b> .....	<b>14</b>
3.1. <i>Objetivo da Tese</i> .....	14
3.2. <i>Pressupostos Empregados na Proposta de Solução do Problema</i> .....	14
3.2.1. Emprego de Técnicas Evolutivas e Sócio-Cognitivas.....	14
3.2.2. Síntese de Filtros Analógicos .....	16
3.3. <i>Descrição do Método</i> .....	20
3.3.1. Geração de circuitos de filtros analógicos por Algoritmos Genéticos .....	21
3.3.2. Geração de circuitos de filtros analógicos por <i>Particle Swarm</i> .....	37
<b>4. Resultados</b> .....	<b>49</b>
4.1. <i>Butterworth passa-baixas de 2a. ordem</i> .....	52
4.1.1. Topologias fixas e Valores de Componentes Variáveis por 30 gerações.....	52
4.1.2. Topologias fixas e Valores de Componentes Variáveis por 60 gerações.....	56
4.1.3. Topologias fixas e Valores de Componentes Variáveis por mais 30 gerações depois de 30 gerações iniciais .....	60
4.2. <i>Sallen-Key passa-baixas de 2a. ordem</i> .....	65
4.2.1. Topologias fixas e Valores de Componentes Variáveis por 30 gerações.....	65

4.2.2.	Topologias fixas e Valores de Componentes Variáveis por 60 gerações .....	69
4.2.3.	Topologias fixas e Valores de Componentes Variáveis por mais 30 gerações depois de 30 gerações iniciais .....	73
4.3.	<i>Sallen-Key passa-faixa de 2a. ordem</i> .....	78
4.3.1.	Topologias fixas e Valores de Componentes Variáveis por 30 gerações .....	78
4.3.2.	Topologias fixas e Valores de Componentes Variáveis por 60 gerações .....	82
4.3.3.	Topologias fixas e Valores de Componentes Variáveis por mais 30 gerações depois de 30 gerações iniciais .....	86
4.3.4.	Análise de Robustez – Monte Carlo .....	91
4.3.5.	Tempo de Execução dos Algoritmos por 60 gerações em um mesmo computador ...	92
<b>5.</b>	<b>Conclusões</b> .....	<b>93</b>
<b>6.</b>	<b>Sugestões para trabalhos futuros</b> .....	<b>94</b>
<b>7.</b>	<b>Referências Bibliográficas</b> .....	<b>96</b>
<b>APÊNDICE I.</b>	<b>Artigos</b> .....	<b>100</b>

## Índice de Figuras

Figura 2.1: Características dos algoritmos genéticos.....	7
Figura 2.2: Fluxograma-exemplo de um algoritmo genético. ....	7
Figura 2.3: Topologia em estrela.....	9
Figura 2.4: Topologia em roda.....	9
Figura 2.5: Topologia em círculo.....	9
Figura 2.6: Topologia randômica.....	9
Figura 2.7: Características dos algoritmos particle swarm.....	11
Figura 2.8 Filtro <i>Butterworth</i> de 3a ordem.....	13
Figura 3.1: Diagrama de Bode de um filtro passa-faixa de segunda ordem.....	16
Figura 3.2: Fluxograma do algoritmo comparativo AG/PS implementado. ....	20
Figura 3.3: Dinâmica da formação de gerações em um algoritmo genético. ....	29
Figura 3.4: Método de seleção - roleta.....	31
Figura 3.5: Operação de crossover entre dois indivíduos. ....	33
Figura 3.6: Operação de mutação. ....	34
Figura 3.7: Gráfico de evolução do AG após 60 gerações, para um filtro <i>Sallen-Key</i> passa-faixa de segunda-ordem com ganho máximo de 45dB. ....	35
Figura 3.8: Intervalo de valores inteiros - topologias e valores de componentes variáveis. .....	40
Figura 3.9: Intervalo de valores inteiros - topologias fixas e valores de componentes variáveis. ....	41
Figura 3.10: Partículas de um processo sócio-cognitivo qualquer.....	43
Figura 3.11: Mudança de posição da partícula i.....	45
Figura 3.12: Implementação do limite de velocidade. ....	46
Figura 4.1: Diagrama de Bode do filtro <i>Butterworth</i> passa-baixas de 2a. ordem.....	52

Figura 4.2.a: $A_{v\text{máx}} = +30$ dB - gráficos dos testes com 30 gerações evoluindo continuamente. ....	53
Figura 4.2.b: $A_{v\text{máx}} = 0$ dB - gráficos dos testes com 30 gerações evoluindo continuamente. ....	54
Figura 4.2.c: $A_{v\text{máx}} = -30$ dB - gráficos dos testes com 30 gerações evoluindo continuamente. ....	55
Figura 4.3.a: $A_{v\text{máx}} = +30$ dB - gráficos dos testes com 60 gerações evoluindo continuamente. ....	57
Figura 4.3.b: $A_{v\text{máx}} = 0$ dB - gráficos dos testes com 60 gerações evoluindo continuamente. ....	58
Figura 4.3.c: $A_{v\text{máx}} = -30$ dB - gráficos dos testes com 60 gerações evoluindo continuamente. ....	59
Figura 4.4.a: $A_{v\text{máx}} = +30$ dB - gráficos dos testes com mais 30 gerações depois de 30 iniciais.....	61
Figura 4.4.b: $A_{v\text{máx}} = 0$ dB - gráficos dos testes com mais 30 gerações depois de 30 iniciais.....	62
Figura 4.4.c: $A_{v\text{máx}} = -30$ dB - gráficos dos testes com mais 30 gerações depois de 30 iniciais.....	63
Figura 4.5: Diagrama de Bode do filtro <i>Sallen-Key</i> passa-baixas de 2a. ordem.....	65
Figura 4.6.a: $A_{v\text{máx}} = +45$ dB - gráficos dos testes com 30 gerações evoluindo continuamente. ....	66
Figura 4.6.b: $A_{v\text{máx}} = +5$ dB - gráficos dos testes com 30 gerações evoluindo continuamente. ....	67
Figura 4.6.c: $A_{v\text{máx}} = -30$ dB - gráficos dos testes com 30 gerações evoluindo continuamente. ....	68

Figura 4.7.a: $A_{v\text{máx}} = +45$ dB - gráficos dos testes com 60 gerações evoluindo continuamente. ....	70
Figura 4.7.b: $A_{v\text{máx}} = +5$ dB - gráficos dos testes com 60 gerações evoluindo continuamente. ....	71
Figura 4.7.c: $A_{v\text{máx}} = -30$ dB - gráficos dos testes com 60 gerações evoluindo continuamente. ....	72
Figura 4.8.a: $A_{v\text{máx}} = +30$ dB - gráficos dos testes com mais 30 gerações depois de 30 iniciais.....	74
Figura 4.8.b: $A_{v\text{máx}} = 0$ dB - gráficos dos testes com mais 30 gerações depois de 30 iniciais.....	75
Figura 4.8.c: $A_{v\text{máx}} = -30$ dB - gráficos dos testes com mais 30 gerações depois de 30 iniciais.....	76
Figura 4.9: Diagrama de Bode do filtro <i>Sallen-Key</i> passa-faixa de 2a. ordem. ....	78
Figura 4.10.a: $A_{v\text{máx}} = +45$ dB - gráficos dos testes com 30 gerações evoluindo continuamente. ....	79
Figura 4.10.b: $A_{v\text{máx}} = +5$ dB - gráficos dos testes com 30 gerações evoluindo continuamente. ....	80
Figura 4.10.c: $A_{v\text{máx}} = -30$ dB - gráficos dos testes com 30 gerações evoluindo	
Figura 4.11.a: $A_{v\text{máx}} = +45$ dB - gráficos dos testes com 60 gerações evoluindo continuamente. ....	83
Figura 4.11.b: $A_{v\text{máx}} = +5$ dB - gráficos dos testes com 60 gerações evoluindo continuamente. ....	84
Figura 4.11.c: $A_{v\text{máx}} = -30$ dB - gráficos dos testes com 60 gerações evoluindo continuamente. ....	85

Figura 4.12.a: $A_{v\text{máx}} = +45$ dB - gráficos dos testes com mais 30 gerações depois de 30 iniciais.....	87
Figura 4.12.b: $A_{v\text{máx}} = +5$ dB - gráficos dos testes com mais 30 gerações depois de 30 iniciais.....	88
Figura 4.12.c: $A_{v\text{máx}} = -30$ dB - gráficos dos testes com mais 30 gerações depois de 30 iniciais.....	89
Figura 4.13.c: Análise de Robustez – Monte Carlo. ....	91
Figura 6.1: Fluxograma do algoritmo para a técnica mista proposta.....	94

## Índice de Tabelas

Tabela 3.1: Instruções para componentes passivos. ....	22
Tabela 3.2.a : Instruções para componentes ativos e topologias possíveis (continua na próxima página). ....	24
Tabela 3.2.b : (continuação) Instruções para componentes ativos e topologias possíveis. ....	25
Tabela 3.3 : Codificação do gene .....	26
Tabela 3.4 : Codificação dos valores dos componentes passivos.....	27
Tabela 3.5: Aptidão dos indivíduos em uma dada geração. ....	30
Tabela 3.6: Aptidão e aptidão relativa dos indivíduos em uma dada geração. ....	31
Tabela 3.7: Valores das partículas do exemplo da figura 3.10.....	44
Tabela 4.1: Resumo dos testes com 30 gerações evoluindo continuamente. ....	52
Tabela 4.2: Resumo dos testes com 60 gerações evoluindo continuamente. ....	56
Tabela 4.3: Resumo dos testes com mais 30 gerações depois de 30 iniciais.....	60
Tabela 4.4: Resumo dos testes com 30 gerações evoluindo continuamente. ....	65
Tabela 4.5: Resumo dos testes com 60 gerações evoluindo continuamente. ....	69
Tabela 4.6: Resumo dos testes com mais 30 gerações depois de 30 iniciais.....	73
Tabela 4.7: Resumo dos testes com 30 gerações evoluindo continuamente. ....	78
Tabela 4.8: Resumo dos testes com 60 gerações evoluindo continuamente. ....	82
Tabela 4.9: Resumo dos testes com mais 30 gerações depois de 30 iniciais.....	86
Tabela 4.10: Tempo de execução para 60 gerações (horas). ....	92

## Convenções Tipográficas

As convenções seguintes são utilizadas nesta tese para identificar certos tipos de informação:

CONVENÇÃO	DESCRIÇÃO
MAIÚSCULAS	siglas ou acrônimos
<i>itálico</i>	palavras ou expressões em língua inglesa ou palavras em português com significado ligeiramente diferente do habitual já definidas anteriormente no texto
<u>sublinhado</u>	variáveis
CAIXA ALTA	termos ou expressões sob definição



## **Nomenclatura**

*ASIC – Application-specific Integrated Circuit*

*AmpOp – Amplificador Operacional*

*IA – Inteligência Artificial*

*AG – Algoritmo Genético*

*PS – Particle Swarm*

## 1. Introdução

Os estudos sobre a síntese automática de circuitos eletrônicos começaram junto com a criação dos primeiros computadores, ganhando impulso com o surgimento dos primeiros processadores digitais integrados. Desde que se conseguiu modelar matematicamente o comportamento das junções semicondutoras e *Schottky* para dispositivos eletrônicos, todos os esforços voltaram-se para a criação de metodologias que permitissem o desenvolvimento das infinitas aplicações que modificaram para sempre a vida do homem, seu modo de agir e de pensar.

Estas metodologias para o desenvolvimento de projetos de circuitos eletrônicos são conhecidas como *métodos clássicos* de concepção de circuitos, para os quais, dependendo da aplicação a que se destinam, existem inúmeras referências na literatura [2-6]. Todos os métodos clássicos, de uma forma ou outra, baseiam-se em métodos analíticos para o desenvolvimento dos circuitos. Isto significa, grosso modo, que, partindo-se de uma determinada função que um circuito deve desempenhar e uma especificação, o engenheiro procura escolher e reunir elementos de circuito elétrico ou eletrônico em uma certa topologia, de forma que todo o conjunto de componentes realize a função que dele se espera. Antes da escolha dos componentes e da topologia que formarão, são analisados os modelos matemáticos de cada um deles em separado e em seu conjunto total.

Entretanto, com o advento da computação e da disponibilidade cada vez maior de recursos de alta velocidade a baixos custos, o conceito de síntese de circuitos parece ganhar uma nova modalidade de realização. Com o auxílio de programas simuladores, que armazenam em si os modelos matemáticos dos componentes eletrônicos, é possível criar rotinas que automaticamente geram circuitos eletrônicos, atribuindo-lhes uma topologia e valores de seus componentes de forma aleatória. Assim, em vez de se projetar analiticamente um circuito eletrônico e realizá-lo fisicamente, poder-se-ia, aproveitando-se do poder de processamento dos computadores atuais, determinar a especificação de um circuito, gerar automática e aleatoriamente milhares de possíveis soluções para ele e, por meio de um critério de seleção, escolher aquele que desempenha a função desejada. Isto representa, de certa forma, uma mudança de paradigma de projeto, que, em alguns casos, poderia deixar de ser analítico para tornar-se estocástico.

Entretanto, ainda que o poder computacional acessível atualmente seja bastante poderoso, ele ainda não é capaz de gerar e avaliar em tempo razoável milhões de circuitos eletrônicos que poderiam ser criados com um dado número de componentes, de modo a encontrar algum que atenda à especificação desejada. Este é o motivo pelo qual se deve adotar uma metodologia de busca baseada em técnicas de inteligência artificial que otimize o processo de seleção do melhor circuito.

O objetivo desta Tese é propor um método para a síntese automática de circuitos eletrônicos analógicos contínuos no tempo para filtros analógicos ativos, e a seguir apresentar uma análise sobre a aptidão das técnicas evolutivas e/ou sócio-cognitivas.

Este trabalho visa, então, realizar um estudo comparativo do desempenho de duas técnicas de inteligência artificial empregadas na síntese de filtros analógicos: as técnicas evolutivas e sócio-cognitivas. As técnicas evolutivas têm como substrato os postulados estabelecidos pela teoria de seleção natural proposta por Charles Darwin em seu clássico livro “*On the Origin of the Species* (1859)”, e, ainda, na genética molecular. A implementação das técnicas evolutivas foi realizada por meio dos algoritmos genéticos.

Por sua vez, as técnicas sócio-cognitivas têm como motivação principal a simulação do comportamento social de animais que andam em bandos, cardumes ou enxames, em busca de alimento ou fugindo de predadores. Estão implementadas nesta tese por meio dos algoritmos chamados de *Particle Swarm*. Os princípios de funcionamento de ambos os métodos são apresentados no capítulo dois.

No capítulo três é realizada a descrição do método que foi implementado nesta tese, a partir de um circuito-exemplo.

O capítulo quatro apresenta os resultados obtidos para a especificação de três filtros analógicos com funções de aproximação conhecidas.

Os capítulos cinco e seis apresentam, respectivamente, as conclusões e comentários adicionais, bem como sugestões para trabalhos futuros. Por fim, são apresentadas as referências bibliográficas utilizadas neste trabalho.

## **2. Técnicas Evolutivas e Sócio-Cognitivas**

Neste capítulo estão reunidas, de modo sucinto, as considerações, conceitos, definições e termos comumente associados às técnicas evolutivas e sócio-cognitivas de busca e otimização.

### **2.1. Técnicas Evolutivas**

#### **2.1.1. Histórico, Definições e Conceitos Básicos**

As técnicas evolutivas possuem sua origem em um ramo da INTELIGÊNCIA ARTIFICIAL (IA), que “*objetiva o desenvolvimento de paradigmas ou algoritmos que requeiram máquinas para realizar tarefas cognitivas, para as quais os humanos são atualmente melhores*” [24]. As técnicas evolutivas consistem em algoritmos de busca e otimização baseados na metáfora biológica da teoria de seleção natural, proposta por Charles Darwin no século XIX, após sua viagem às Ilhas Galápagos, em sua clássica obra “*On the Origin of the Species (1859)*” e, ainda, na genética molecular, que fraciona a célula em unidades menores, que seriam os cromossomos e os genes.

A teoria proposta pelo cientista inglês define que, na natureza, o processo de evolução de uma espécie, que é um conjunto de INDIVÍDUOS capazes de realizarem entre si a REPRODUÇÃO, ocorre quando quatro condições são satisfeitas:

1. um indivíduo é capaz de se perpetuar deixando descendentes (reprodução);
2. existe uma POPULAÇÃO formada por indivíduos da mesma espécie no mesmo AMBIENTE (meio ambiente);
3. existe alguma variedade (pequenas diferenças) entre os indivíduos da mesma população no meio ambiente; e
4. a capacidade de sobrevivência dos indivíduos no ambiente está diretamente relacionada à variedade individual de cada um, pois serão estas diferenças que farão com que os mesmos se adaptem melhor ao meio ambiente.

Suas conclusões foram baseadas nas suas observações realizadas em sua viagem às Ilhas Galápagos, um arquipélago existente no Oceano Pacífico, distante dos

continentes, e sem influência com os mesmos, e onde existem espécies que somente são encontradas nessas ilhas.

Naquele ambiente com escassez de recursos, indivíduos de uma mesma espécie que possuíam maior facilidade de conseguir alimento e fugir dos predadores tinham maior chance de sobrevivência e, também, maior probabilidade de deixar descendentes.

Com o passar das GERAÇÕES, observou ainda que a incidência dos indivíduos com as características físicas que os tornavam melhor adaptados ao meio ambiente aumentava em frequência, enquanto a frequência de incidência daqueles menos adaptados diminuía. Então, assegurou-se de que as diferenças físicas entre indivíduos de uma mesma espécie eram transmitidas aos seus descendentes no ato da reprodução e seriam, portanto, variáveis no tempo e condicionadas pelo ambiente.

No século XX, surgiu a teoria da genética molecular, que veio a corroborar as idéias de Charles Darwin. Por essa teoria, definiu-se, que o conjunto de características de cada indivíduo estaria armazenado em seu CROMOSSOMO, que, por sua vez, seria composto pela união dos GENES. Uma célula de um ser humano possui 46 cromossomos, à exceção das células de reprodução, que são o óvulo, na mulher, e o espermatozóide, no homem, as quais possuem apenas metade do número de cromossomos, ou seja, 23. O número total de genes dos seres humanos e suas funções estão sendo mapeados em um projeto mundial de pesquisa científica denominado “genoma humano”.

Pelo que já foi estudado até hoje, cada gene é, efetivamente, o responsável pela existência desta ou daquela característica física no indivíduo e, no momento da reprodução, os cromossomos pertencentes aos pais seriam partidos em pontos aleatórios - *porém formando partes complementares entre si* - e combinados para criar o filho, com número de cromossomos igual ao dos pais, numa operação conhecida como CROSSOVER, ou cruzamento, o que garantiria a variedade genética em uma população. Também foi verificada a existência de um fenômeno físico-químico que, após o cruzamento, seria capaz de modificar as características do cromossomo de um indivíduo após a reprodução: a MUTAÇÃO. Esta consiste em uma alteração aleatória, por ação de agentes físicos, químicos ou físico-químicos, como a ação de elementos radioativos e a radiação solar, entre outros, do valor de apenas um gene no cromossomo, e não está relacionada com a atividade reprodutiva.

Os conceitos da teoria de evolução e da genética foram, então, utilizados por John Holland, da Universidade de Michigan, para criar programas que simulassem o

processo de evolução de sistemas naturais. Estavam criados os ALGORITMOS GENÉTICOS [22]. Estes algoritmos mostraram-se extremamente eficientes na resolução de problemas para os quais não há solução analítica conhecida, de otimização e busca, particularmente aqueles em que a complexidade e o tamanho do conjunto de soluções possíveis é tão grande que impede o emprego de métodos de busca exaustivos, e logo se tornaram extremamente populares.

São algoritmos que, de acordo com uma especificação de ALVO desejado, i.e., a especificação da solução requerida, transformam um conjunto inicial ou população inicial de objetos matemáticos, os indivíduos, representados por estruturas semelhantes às encontradas nos cromossomos, em outros conjuntos de indivíduos derivados do conjunto inicial, a partir de reproduções integrais de seus membros, operações de cruzamento entre eles e de mutações que ocorrem com uma probabilidade estabelecida, perfazendo uma contínua operação de refinamento até o surgimento do indivíduo cujas características mais se aproximam do alvo especificado. As funções que realizam no algoritmo as operações de cruzamento ou *crossover*, reprodução e mutação são denominadas OPERADORES GENÉTICOS. A exemplo do que ocorre na natureza, a probabilidade de um indivíduo reproduzir-se ou casar-se com outro da mesma população dependerá de um critério que estabelece a sua proximidade ou semelhança em relação ao alvo desejado, denominada *FITNESS* ou aptidão.

As questões de maior relevância no desenvolvimento de um algoritmo genético, e que estão intrinsecamente relacionadas com a natureza do problema escolhido, concentram-se na representação, na avaliação da população e na seleção dos indivíduos que participarão dos cruzamentos ou que serão integralmente reproduzidos, e que estão melhor descritas abaixo:

1. A REPRESENTAÇÃO [7,22] consiste em codificar na forma de uma estrutura de dados parecida com a de um cromossomo, ou seja, em uma seqüência ordenada de números, representados em uma base decimal, hexadecimal ou binária, soluções de problemas de engenharia tais como um motor automotivo, um circuito eletrônico analógico ou uma reação química de produção de uma resina;
2. A AVALIAÇÃO [7,22], por sua vez, é o processo mediante o qual o algoritmo atribui a cada indivíduo da população corrente um número real, que representa sua proximidade com as características do indivíduo-alvo ou padrão, que é denominado valor de *fitness*. Geralmente, essa atribuição dá-se

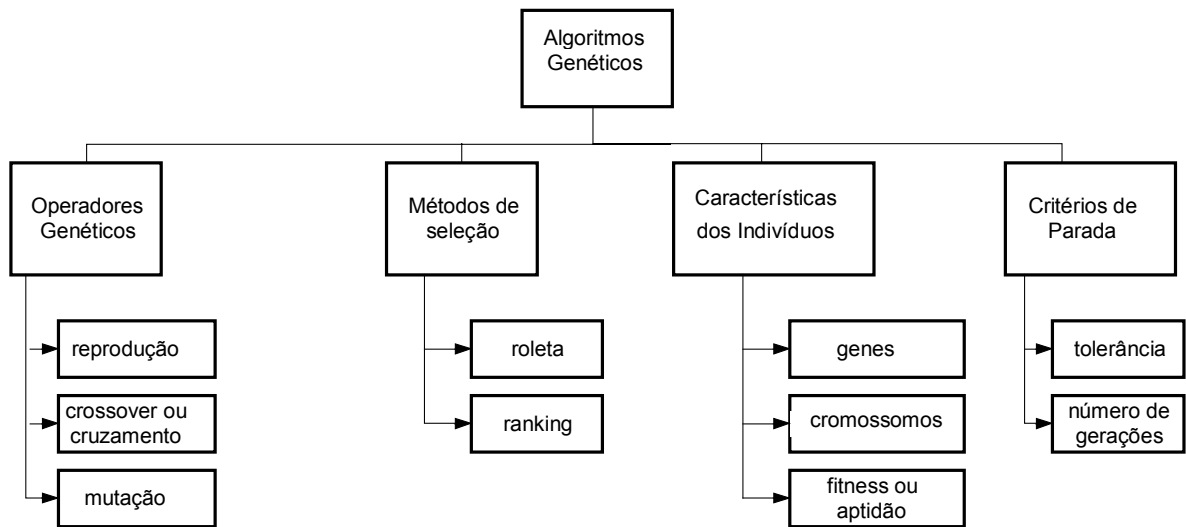
por meio de uma função cujas entradas são a resposta esperada do indivíduo-alvo e a resposta do indivíduo sob avaliação, razão pela qual é comumente chamada de FUNÇÃO-OBJETIVO. O objetivo pode ser não somente a proximidade com uma resposta de saída do alvo, como também com várias outras de suas respostas características. Neste caso, tem-se a realização de uma busca com MÚLTIPLOS OBJETIVOS; e

3. A SELEÇÃO [7,22] dos indivíduos que serão casados ou reproduzidos para a formação da próxima geração depende do método empregado. O mais difundido deles é conhecido como o da ROLETA, pois, tal como em uma roleta de cassino, os indivíduos a serem reproduzidos ou casados são sorteados aleatoriamente. Entretanto, diferentemente do jogo, as probabilidades de sorteio não são iguais, mas proporcionais ao valor de *fitness* do indivíduo, para que sejam sempre escolhidos os indivíduos mais adaptados. Existem ainda outros critérios de seleção, entre os quais, podemos citar o *ranking* [14], mas que não serão discutidos aqui, pois não foram utilizados nesta Tese. Em todos os métodos de seleção existe, ainda, a possibilidade de se optar pelo ELITISMO, que consiste na reprodução incondicional do indivíduo melhor adaptado ao meio de uma geração para a geração seguinte, de modo que não haja risco de se perder a melhor solução já alcançada até aquele momento.

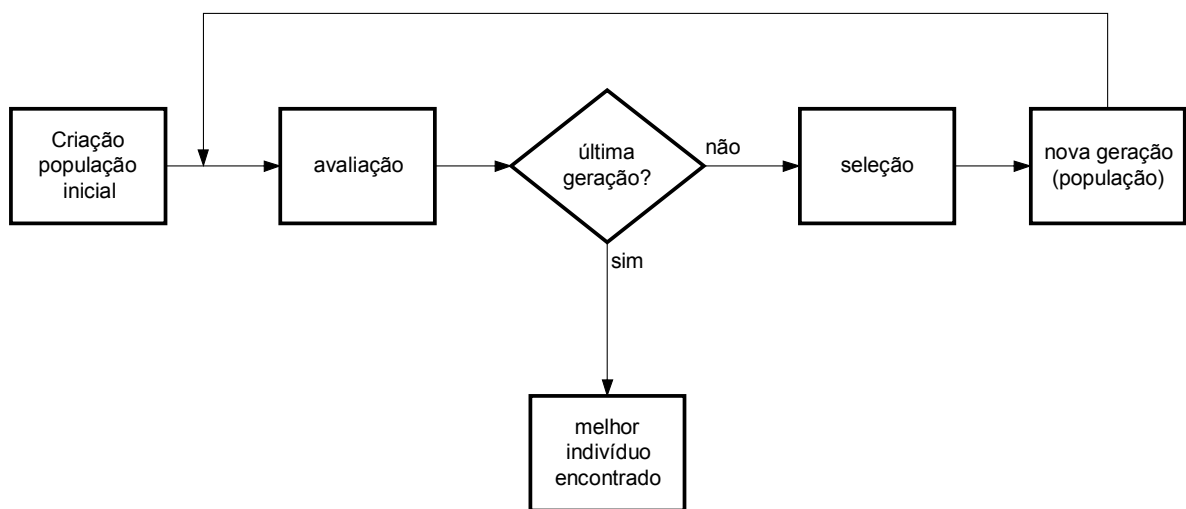
Finalmente, nos algoritmos genéticos podem ser estabelecidos dois critérios de parada, segundo os quais a evolução já teria completado seu objetivo: número de gerações ou tolerância. Na TOLERÂNCIA, a execução do algoritmo encerra-se após a constatação da existência de um indivíduo com as mesmas características do indivíduo-alvo, dentro de uma faixa de tolerância estabelecida. No NÚMERO DE GERAÇÕES, o encerramento do algoritmo condiciona-se à execução de um determinado número prévio de gerações, seja qual for a proximidade do alvo em que estiver o melhor indivíduo no fim da execução. Trata-se este, pois, de um critério *a priori* e aquele de um critério *a posteriori*. Neste trabalho, foi adotado o segundo critério de parada tendo em vista que os recursos computacionais existentes são limitados e também porque, caso o melhor indivíduo encontrado não chegasse à tolerância pré-estabelecida no primeiro critério, o algoritmo poderia continuar a ser executado indefinidamente.

A figura 2.3 apresenta uma síntese das características do algoritmo genético e a figura 2.4 uma representação em fluxograma de sua execução, segundo o critério de

parada de número de gerações. As especificações e as considerações pertinentes relativas ao desenvolvimento do algoritmo incluído neste trabalho serão descritas com detalhes no capítulo quatro.



**Figura 2.1: Características dos algoritmos genéticos.**



**Figura 2.2: Fluxograma-exemplo de um algoritmo genético.**



## 2.2. Técnicas Sócio-Cognitivas

### 2.2.1. Histórico, Definições e Conceitos Básicos

As técnicas sócio-cognitivas também derivam da inteligência artificial e são metodologias de busca e de otimização introduzidas por James Kennedy e Russel Eberhart [17] em 1995. A metáfora utilizada para compor o substrato da teoria é a do comportamento dos grupos de animais que se movimentam em bandos, cardumes ou enxames. Estudando os métodos pelos quais estes animais procuram alimento ou fogem de predadores, os autores perceberam que o sucesso na fuga ou na obtenção de comida por cada indivíduo dentro de sua coletividade depende da experiência do próprio indivíduo, i.e., a cultura por ele adquirida, em sua existência, nas atividades desempenhadas para seu sustento e preservação, acrescida ainda da influência das atitudes dos seus vizinhos mais próximos, estas também produtos de suas culturas individuais. Estabeleceram, então, os pressupostos daquilo que vieram a batizar de *Swarm Intelligence* [16]:

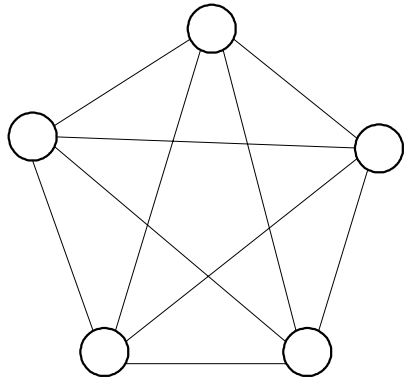
1) A Inteligência é fruto da interação social: ela nasce da avaliação, comparação e imitação por todos das atitudes consideradas bem-sucedidas de alguns; e

2) Cultura e Cognição são conseqüências inseparáveis da socialização dos indivíduos: o surgimento da cultura, aqui entendida como uma padronização no modo de agir, e, por conseqüência, da apreensão do conhecimento sobre algo, só foi possível graças às interações sociais dos indivíduos.

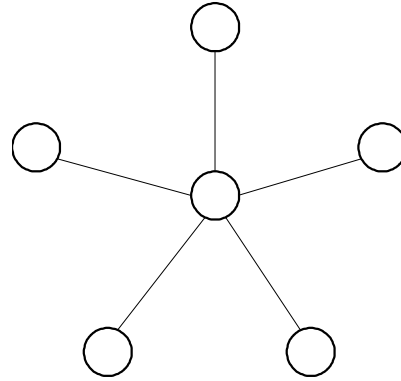
Dessa forma, criaram a técnica computacional chamada de *PARTICLE SWARM* [16] -*Enxame de Partículas, em tradução livre*, que, a exemplo dos algoritmos genéticos, também objetivam a otimização na busca por soluções adequadas em espaços onde o número de soluções possíveis tornaria proibitiva a utilização de técnicas de busca exaustivas. Nesta técnica, a partir de uma especificação de solução desejada, o alvo, é criado um conjunto aleatório de PARTÍCULAS, no qual a POSIÇÃO no espaço ocupada por cada uma delas representa uma possível solução para o alvo. Como a inteligência é fruto da interação social, são estabelecidas *a priori* para cada partícula um conjunto de relações com os outros membros da sociedade, i.e., o modo como as partículas influenciar-se-ão mutuamente. Este esquema de influência é chamado de TOPOLOGIA. Assim, as partículas movimentar-se-ão, de acordo com a topologia definida, dentro do

espaço de soluções, procurando concentrar-se nas regiões onde as soluções estão mais próximas do alvo.

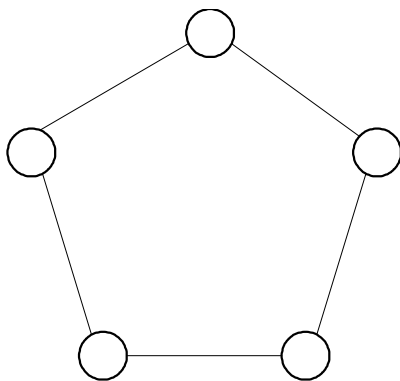
São apresentadas nas figuras de 2.6 a 2.9 os tipos de topologia normalmente empregados em *Particle Swarm*: estrela, roda, círculo e randômica.



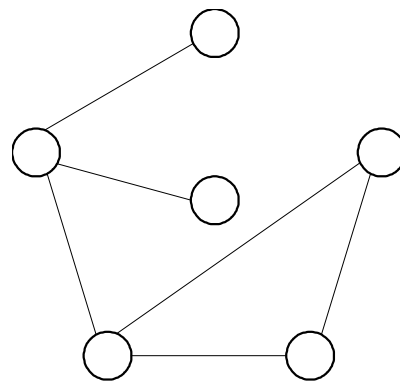
**Figura 2.3: Topologia em estrela.**



**Figura 2.4: Topologia em roda.**



**Figura 2.5: Topologia em círculo.**



**Figura 2.6: Topologia randômica.**

As questões mais importantes na implementação desta classe de algoritmos de busca concentram-se nas atividades de avaliação, comparação e imitação. A AVALIAÇÃO é, tal como nos algoritmos genéticos, o processo mediante o qual o algoritmo atribui a cada partícula um número real representativo da proximidade absoluta de sua posição com a do alvo, seu valor de aptidão, sua POSIÇÃO COGNITIVA. Essa atribuição consubstancia-se na função-objetivo, a qual, dependendo da natureza do problema, deseja-se maximizar ou minimizar.

A COMPARAÇÃO, por sua vez, é o processo mediante o qual o algoritmo estabelece como a aptidão dos vizinhos de uma partícula vizinha influenciará o seu

movimento futuro em direção a uma região mais próxima do alvo desejado, i.e., determina a sua POSIÇÃO SOCIAL.

Finalmente, a IMITAÇÃO é uma ponderação das posições cognitivas e sociais de cada partícula, de modo a determinar-lhe seu movimento futuro, a sua VELOCIDADE, em módulo, direção e sentido. A velocidade, segundo a definição dos próprios autores da técnica, representa a diferença entre duas posições no espaço observadas entre duas iterações consecutivas [16].

Após a imitação, aplica-se a velocidade calculada à posição da partícula, forçando-a a assumir uma nova posição no espaço de soluções. Neste momento, o algoritmo estará pronto para realizar novamente uma nova ITERAÇÃO, i.e., um novo processo de avaliação, comparação e imitação relativos à posição de suas partículas.

Os critérios de parada dos algoritmos *particle swarm* são exatamente os mesmos dos algoritmos genéticos: número de iterações e tolerância em relação ao alvo.

Algebricamente, a implementação desta classe de algoritmos dá-se pela criação de dois vetores: “posição” e “velocidade”. O índice de cada vetor posição contém os dados das coordenadas da posição que uma dada partícula está ocupando numa dada iteração. O vetor velocidade é o valor das coordenadas que serão adicionadas à posição de uma determinada partícula de modo a estabelecer-lhe a nova posição, ou a posição futura. Assim, tem-se a expressão algébrica da posição:

$$x_i(t) = x_i(t-1) + v_i(t-1), \quad (2.1)$$

Onde:

$x_i(t) \Rightarrow$  Posição atual da partícula

$v_i(t-1) \Rightarrow$  Velocidade anterior da partícula

$x_i(t-1) \Rightarrow$  Posição anterior da partícula.

Como a determinação da velocidade de uma partícula depende da sua posição cognitiva e da sua posição social, pode-se dizer que a posição da partícula é uma função de quatro argumentos:

$$x_i(t) = f\left(x_i(t-1), v_i(t-1), p_i, p_g\right), \quad (2.2)$$

Onde  $p_g$  é a coordenada da partícula que possui a melhor posição absoluta global e  $p_i$  é a coordenada da partícula, dentre os vizinhos de  $i$  que o influenciam, que possui a melhor posição absoluta.

Finalmente, tem-se o conjunto de equações que descrevem o deslocamento das partículas:

$$\begin{cases} v_i(t) = v_i(t-1) + \varphi_1 [p_i - x_i(t-1)] + \varphi_2 [p_g - x_i(t-1)] \\ x_i(t) = x_i(t-1) + v_i(t) \end{cases} \quad (2.3)$$

Na expressão acima,  $\varphi_1$  e  $\varphi_2$  são pesos estocásticos cuja função é atribuir algum grau de aleatoriedade ao sistema, de modo a evitar a prisão das partículas em máximos ou mínimos locais da função-objetivo [16], o que comprometeria o desempenho da busca. Estes pesos estocásticos desempenham função semelhante à desempenhada pela mutação no AG.

A figura 2.7 sintetiza as principais características dos algoritmos *particle swarm* aqui expostas.

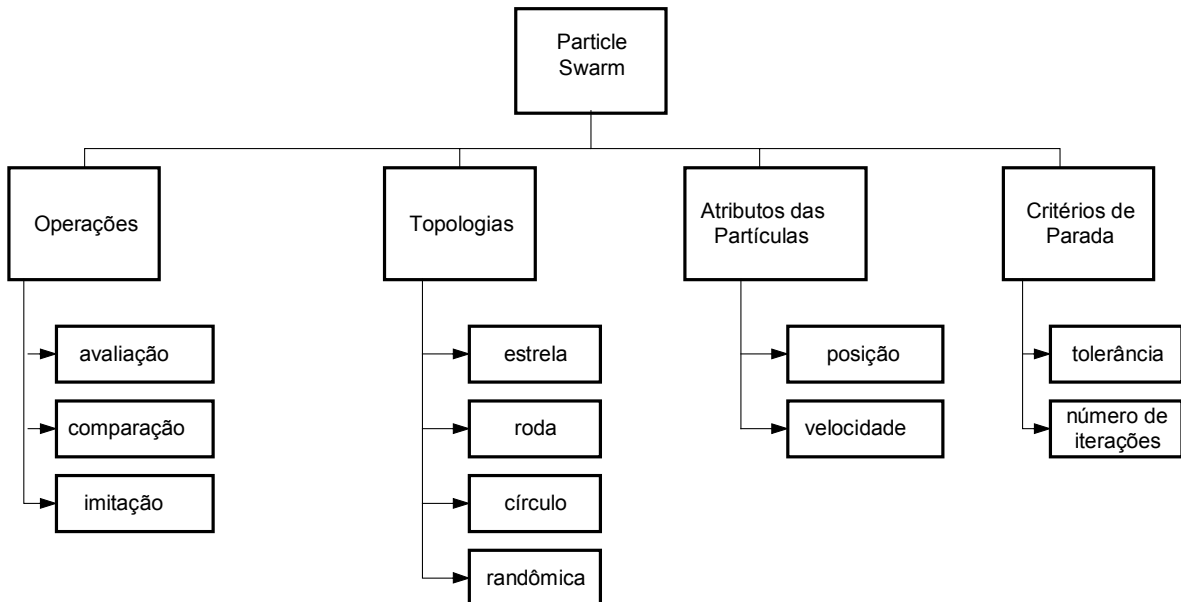


Figura 2.7: Características dos algoritmos *particle swarm*.

### 2.3. Comentários adicionais sobre a revisão bibliográfica e o estado da arte

Embora já tenham sido citadas ao longo do texto algumas referências bibliográficas, bem como apresentados alguns conceitos importantes, persiste a necessidade de que sejam feitos comentários e observações adicionais, ainda que de

forma sucinta, sobre a revisão bibliográfica do tema da tese, bem como sobre o estado da arte.

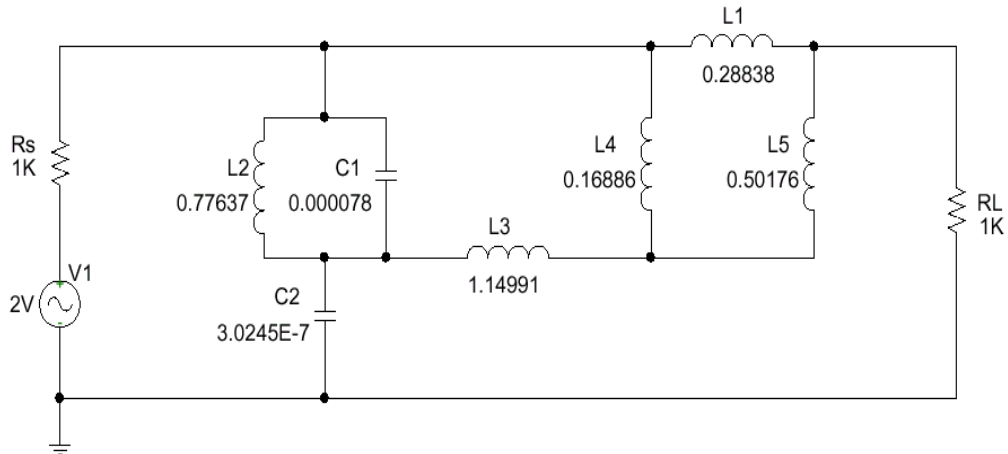
Considerando a realidade brasileira sobre a síntese de circuitos eletrônicos, de uma forma geral, quase que se pode resumir as atividades acadêmicas e de pesquisa, com registro em anais de eventos, revistas e periódicos de acesso ao público em geral, ao *Journal of Integrated Circuits and Systems* (publicado pela Sociedade Brasileira de Computação), ao Simpósio Brasileiro de Concepção de Circuitos Integrados (SBCCI), e ao *Latin American Test Workshop* (LATW). A consulta a estes registros mostrou que o uso de técnicas evolutivas e sócio-cognitivas é explorada por um número reduzido de autores, e ainda representa um vasto campo de atividades.

Todavia, em se considerando a comunidade internacional, considera-se o interesse sobre o tema, bastante pronunciado, e inúmeros eventos, periódicos e livros-texto podem ser encontrados sobre o assunto, sendo que um exemplo significativo de evento onde se apresenta o estado da arte, reunindo autores bastante seletos, é o *NASA/DoD Conference on Evolvable Hardware*.

Desta forma, a seguir, são apresentados os principais artigos, relacionados ao tema desta Tese, apresentados em algumas conferências e periódicos, nos últimos anos.

Uma fonte interessante de consulta no que se refere à relevância do tema, assim como a sua atualidade, pode ser encontrada em [29, 30]. Como referências clássicas para o tema poderíamos citar [7, 16, 17, 22, 24].

Uma curiosidade sobre o tema pode ser constatada em [33], um artigo considerado de grande impacto na época, que propõe uma técnica para representação de circuitos. Na figura 2.8 é apresentado um *Evolved 3rd-order Butterworth low-pass filter with units in ohms, farads, and henries* [33]. Observando-se os componentes do circuito, pode-se considerar, no mínimo, irrealistas os valores calculados pelo algoritmo do autor. Neste ponto fica clara a relevância desta tese, que busca realizar circuitos mais próximos da realidade.



**Figura 2.8 Filtro *Butterworth* de 3a ordem.**

Em [31] foi proposta uma ferramenta para síntese de células analógicas, baseada em técnicas de otimização usando algoritmos genéticos. O procedimento se utiliza de uma linguagem formal para descrição do problema de otimização, que é capaz de descrever funções de razoável complexidade.

Um bom exemplo de metodologia para síntese de circuitos envolvendo múltiplos objetivos pode ser encontrado em [32], onde é proposta uma abordagem hierárquica para síntese de circuitos em larga escala. Nesse caso o autor mostra de forma clara a necessidade da formulação dos objetivos a serem alcançados, e a métrica envolvida para a consecução desses objetivos.

Outros aspectos têm sido abordados por diversos autores, como, por exemplo, a recuperação de falhas [34], e a reutilização de soluções topológicas de redes elétricas [35].

Por fim, é dever destacar como Instituições Nacionais participantes desse esforço envolvendo novas técnicas para síntese de circuitos a Universidade Federal do Rio de Janeiro, a Universidade do Estado do Rio de Janeiro e a Pontifícia Universidade Católica do Rio de Janeiro.

### **3. A SÍNTESE AUTOMÁTICA DE FILTROS ANALÓGICOS UTILIZANDO ALGORITMOS GENÉTICOS E *PARTICLE SWARM***

#### **3.1. Objetivo da Tese**

O objetivo desta Tese é propor um método para a síntese automática de circuitos eletrônicos analógicos contínuos no tempo para filtros analógicos ativos, e a seguir apresentar uma análise sobre a aptidão das técnicas evolutivas e/ou sócio-cognitivas.

O processo de síntese desses circuitos através do uso de técnicas evolutivas e sócio-cognitivas propõe uma sistemática de codificação dos circuitos, sua simulação e sua avaliação. O método utiliza apenas resistores, capacitores e amplificadores operacionais como componentes, dada uma característica de resposta no domínio da frequência de um filtro analógico conhecido. O processo de análise utiliza métricas de desempenho adequadas para comparar as duas técnicas.

Como em qualquer problema de engenharia, não existe apenas uma solução na resolução do mesmo, mas sim várias soluções, que dependem, *a priori*, dos pressupostos que foram pré-estabelecidos. Portanto, faz-se necessário estabelecer os mesmos, para que o desenvolvimento do presente trabalho ocorra de maneira contínua, de modo a conseguir ao seu final, responder as questões indagadas inicialmente.

#### **3.2. Pressupostos Empregados na Proposta de Solução do Problema**

##### **3.2.1. Emprego de Técnicas Evolutivas e Sócio-Cognitivas**

As técnicas utilizadas na presente tese são duas, dentre várias, das mais recentes ferramentas de inteligência artificial empregadas com o objetivo de buscar uma solução sem o uso de técnicas de busca exaustiva, também conhecida por “força bruta”.

As técnicas propostas são mais indicadas do que a busca exaustiva devido ao fato de que esta última, além de necessitar de um esforço computacional enorme, perde desempenho à medida que a complexidade do circuito aumenta (número de nós e de componentes). Como o trabalho de síntese está intimamente ligado com a topologia do circuito, faz-se necessário também utilizar de maneira adequada ferramentas de criação

dos circuitos, de modo a colocar uma certa ordem no caos, ou seja, buscar a montagem de circuitos que realmente tenham uma topologia que permita obter resultados consistentes.

As técnicas derivadas da Inteligência Artificial permitem uma busca otimizada e robusta, utilizando uma capacidade computacional menor.

Poderíamos pensar também em não buscar tão somente o “primeiro organismo unicelular”, ou seja, um circuito eletrônico simples apenas formado por resistores, capacitores e transistores MOS, mas em tecidos mais especializados, como o formado por “células do tecido conjuntivo” ou por “células dos músculos cardíacos”, que poderiam ser os blocos de componentes (macros) formados pelos mesmos elementos.

A única mudança seria no caso dos transistores, que seriam substituídos por amplificadores operacionais, elementos de ganho que amplificam de DC até algumas centenas de kHz, sem a necessidade de haver preocupação com a polarização ou com a análise DC. No caso dos circuitos gerados com transistores como componentes ativos, este fato que deve ser verificado constantemente, através da técnica de análise DC, visto que o circuito gerado pode aparentemente possuir uma boa resposta AC, mas a análise DC mostra que o mesmo está pessimamente polarizado.

A escolha dos métodos evolutivos e sócio-cognitivos, dentre as diversas técnicas de inteligência artificial, se deve ao fato de que ambos apresentam resultados satisfatórios em diversos problemas estudados. Entretanto, ainda não existiu uma comparação acerca de qual seria o melhor neste tipo de problema, e se as melhores características de cada um poderiam ser empregadas na construção de um possível novo método misto.



### 3.2.2. Síntese de Filtros Analógicos

Os filtros analógicos são uma classe de circuitos cujo comportamento é caracterizado de modo quase completo pelo estudo de suas respostas no domínio da frequência por meio de suas curvas traçadas no diagrama de Bode [18].

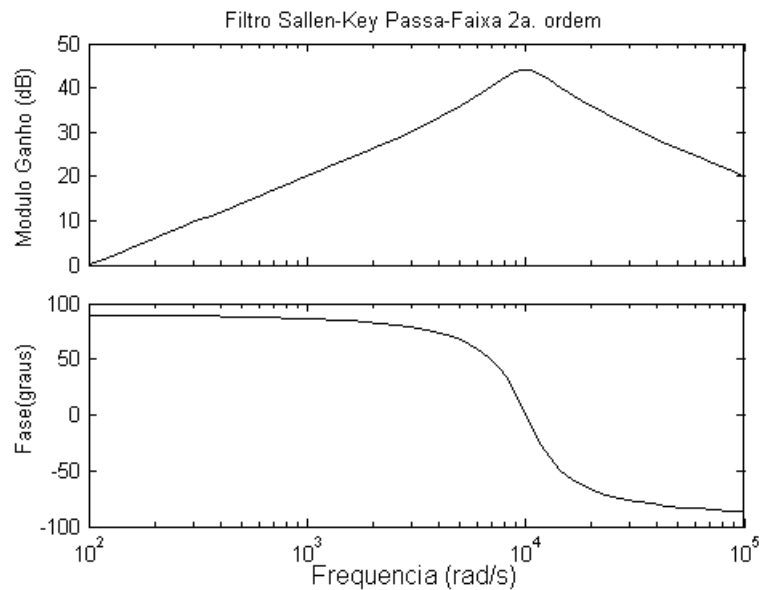


Figura 3.1: Diagrama de Bode de um filtro passa-faixa de segunda ordem.

#### 3.2.2.1 Emprego do Ganho no Espectro em Frequência

O diagrama de Bode representa, através das curvas acima citadas, as variações do módulo do ganho e da fase com a frequência para um determinado circuito com uma determinada função de transferência. A escolha da função-objetivo padrão apenas com relação ao módulo do ganho se deve a razões de cunho prático, pois os equipamentos de medida existentes em laboratório trabalham com este tipo de componente, sendo a fase obtida indiretamente. Os dados de fase são apresentados apenas para aumentar o número de informações sobre o circuito.

### 3.2.2.2 Função de Transferência do Filtro

A princípio, não é necessário que se disponha previamente de qualquer tipo de expressão da função de transferência do circuito para a implementação da metodologia proposta. No lugar da mesma, o circuito que foi sintetizado torna-se conhecido por meio de sua lista de nós (*netlist*). A caracterização completa do circuito é conseguida através da simulação AC (no domínio da frequência) no mesmo. A resposta de saída do circuito simulado passa, então, a ser conhecida. Como o conhecimento do comportamento do circuito depende unicamente dos dados de simulação, a pressuposição existente é de que o método proposto seja do tipo SBT (“*simulation before test*”).

### 3.2.2.3 Utilização somente de Resistores, Capacitores e Amplificadores

#### Operacionais

No presente trabalho, foi feita a opção de apenas empregar, como componentes destes circuitos, resistores, capacitores e amplificadores operacionais, doravante chamado de AmpOp’s. Para os dois primeiros, foram utilizados valores discretos extraídos da antiga tabela RETMA 10% [23], antigo nome da associação EIA [26]– *Electronic Industries Alliance* –, que congrega diversas entidades representativas da indústria mundial de componentes eletrônicos. Afiliada à EIA está a ECA [27] – *Electronic Components, Assemblies, and Materials Association* –, que representa a indústria mundial de componentes passivos e manteve a padronização dos valores da antiga tabela RETMA.

A utilização destes valores tem o objetivo de aproximar nossas simulações em computador do “mundo real da eletrônica”: não se pode simplesmente arbitrar valores para componentes de circuitos usando componentes discretos, ou até mesmo circuitos integrados, onde, dependendo do processo, a precisão para valores de componentes não é alta.

Outro aspecto relevante é o uso de componentes ativos de razoável complexidade. Foi empregado um macromodelo modificado do macromodelo original do amplificador operacional OP-07, que é disponibilizado pelo fabricante *Analog Devices* em seu *site*. A modificação do modelo visa tão somente limitar o ganho elevado do OP-07, que, em alguns casos, provoca erros de convergência numérica no simulador SPICE, além de impedir topologias com realimentação positiva do AmpOp.

Considerando os elementos passivos, foi acrescentado um resistor de valor elevado em paralelo com os mesmos (resistores e capacitores), visando eliminar ciclos capacitivos e problemas de simulação do SPICE pela existência de nós flutuantes no circuito. Cabe ressaltar, que o SPICE realiza uma análise estrutural da topologia do circuito, antes de iniciar a simulação, e que não simula o mesmo, caso existam esses tipos de problema.

Como o problema proposto é o de sintetizar ou, em outras palavras, gerar ou criar, circuitos com as mais diversas topologias, em sua maioria pouco ou nada “ortodoxas“, muitos circuitos que são criados aleatoriamente e possuem uma topologia bem próxima da ideal ou, digamos, do modelo teórico aceito, e que poderiam possuir uma resposta bem próxima da função-objetivo padrão, são sumariamente descartados da simulação por possuírem, às vezes, apenas um componente passivo com um nó flutuante. Seria como, utilizando-se novamente a biologia como metáfora, um ser vivo, que, ao nascer em uma sociedade qualquer, possuísse um apêndice a mais em um dos seus membros, sem função definida, e que não interferisse em nada no desempenho do ser. No entanto, uma lei imaginária qualquer dessa sociedade impede que seres vivos com deformações continuem vivos após o nascimento. Sendo assim, o mesmo seria eliminado, mesmo que dele pudessem surgir resultados melhores do que de qualquer outro elemento, dito normal, desta sociedade.

Os indivíduos somente serão eliminados caso suas respostas sejam piores que a dos outros membros dessas populações, mesmo assim dependendo das características do algoritmo selecionado.

De todo modo, tanto as técnicas evolutivas como as técnicas sócio-cognitivas possuem seus mecanismos para preservar os melhores indivíduos.

Para podermos realizar uma comparação das técnicas, foi desenvolvido um algoritmo que “roda” uma geração da técnica evolutiva e uma iteração da técnica sócio-cognitiva, ambas as técnicas partindo da mesma população inicial, sob as mesmas condições. Deste modo, seria como pensar em dois gêmeos univitelinos do “elo perdido”, parente distante que foi o divisor de águas do homem e do macaco. Do mesmo ponto inicial, surgiram os seres humanos e os primatas. Caso contrário, as duas técnicas poderão gerar dois seres vivos iguais, ou muito parecidos, passadas  $n$  gerações.

### **3.2.2.4 Utilização de Topologias Variáveis e Valores de Componentes**

#### **Variáveis para Algoritmos Genéticos**

Devido ao processo de codificação criado, que será detalhado mais à frente, e à característica principal do Algoritmo Genético de perpetuar os melhores indivíduos de uma população (elitismo), os novos indivíduos criados a partir da metade mais apta da população inicial, após os processos de reprodução e mutação, podem sofrer variação tanto em termos de topologia quanto no valor de seus componentes, sem perder, entretanto, sua característica de perpetuar os mais aptos.

### **3.2.2.5 Utilização de Topologias Variáveis e Valores de Componentes**

#### **Variáveis para o *Particle Swarm***

O Particle Swarm não é elitista e, portanto, pode perder seu melhor indivíduo, caso exista uma quantidade de indivíduos (vizinhos do melhor indivíduo) com baixo valor de *fitness* e que desloquem o melhor de sua posição atual. Nesse caso, o indivíduo pode sofrer variações que alterem sensivelmente o valor de seus componentes ou, se for o caso, até mesmo sua topologia. Foram adotados estes dois tipos de variação para o algoritmo de PS, a fim de verificar se podem coexistir mudanças de topologia e de valores de componentes, de modo idêntico ao que ocorre com o AG. Este último, por sinal, foi codificado de modo a somente permitir instruções possíveis de formar um grafo completo entre entrada e saída, sem a existência de nós flutuantes no interior do circuito, ou com falta de elementos que levem o sinal de entrada para a saída. Já o PS não guarda nem possui qualquer tipo de relação com a topologia dos circuitos gerados. Este também foi um outro motivo para utilizar a população inicial gerada através de AG para o desenvolvimento dos circuitos através de PS.

### **3.2.2.6 Utilização de Topologias Fixas e Valores de Componentes**

#### **Variáveis para o *Particle Swarm***

De modo a comparar os resultados obtidos no item anterior, além de compará-los com os do AG, foi fixada a topologia das partículas, variando apenas os valores dos componentes com o PS. Desta maneira, evita-se que a primeira população gerada através de um mecanismo que impede o aparecimento de circuitos que possuam erros

topológicos, como o de grafos sem componentes que acoplem a entrada à saída, seja perdida em parte ou totalmente. Assim, o PS será empregado como método de *tuning* dos valores dos componentes dos circuitos da população inicial.

### 3.3. Descrição do Método

O método proposto está subdividido da seguinte forma: a evolução, a cada geração, de uma população inicial através de AG e em seguida a evolução desta mesma geração através de PS. Ou seja, o algoritmo processa uma geração de AG e em seguida uma iteração de PS. Somente após o processamento de uma nova iteração através de AG, o algoritmo executa uma nova geração de PS. Naturalmente, os membros da população inicial podem sofrer alteração em sua topologia e valores de componentes no caso do AG, caso o valor de *fitness* do melhor indivíduo da geração corrente seja superior ao do melhor indivíduo que foi gerado anteriormente. No caso do PS, antes do início do processamento, é feita a opção, pelo usuário, do método implementado, se deseja fixar ou não a topologia dos circuitos gerados a cada nova iteração, a partir da primeira população.

A figura 3.2 apresenta uma representação do fluxograma do algoritmo comparativo AG/PS implementado, que executa uma geração do AG e em seguida uma iteração do PS, segundo o critério de parada de número de gerações/iterações.

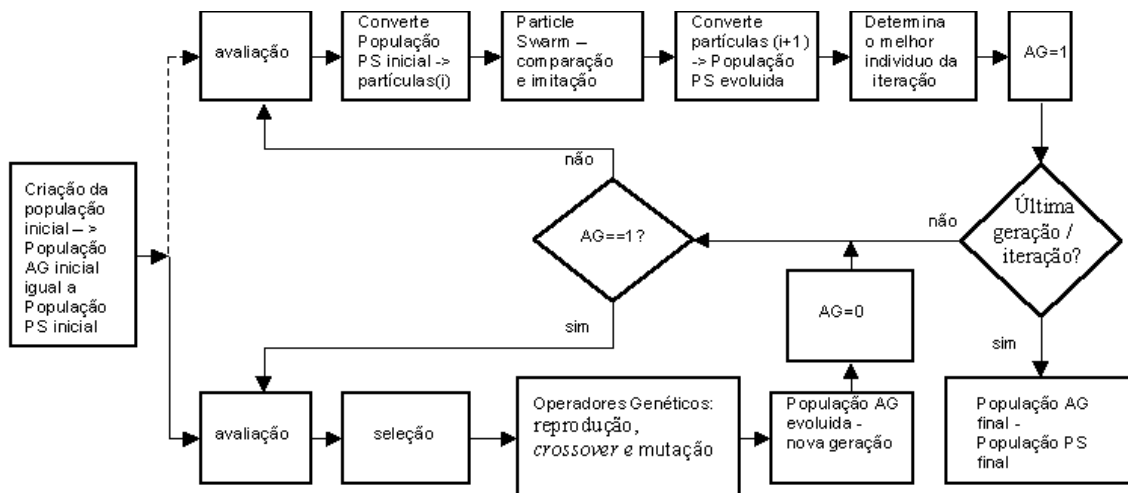


Figura 3.2: Fluxograma do algoritmo comparativo AG/PS implementado.

A linha tracejada esclarece que a população inicial que é criada é duplicada de modo a criar uma população inicial para AG que é igual à população inicial de PS. Saindo então do mesmo ponto inicial, deseja-se comparar os dois métodos através de algumas medidas, como as que se seguem:

O *fitness* máximo final alcançado pelas populações evoluídas por um e por outro método;

O tempo de execução de um *loop* pelos métodos;

A capacidade de evolução ou regressão dos métodos;

O número de gerações que os métodos levam para conseguir convergir;

A estabilidade dos circuitos obtidos pelos métodos através da análise de Monte Carlo.

### **3.3.1. Geração de circuitos de filtros analógicos por Algoritmos Genéticos**

Uma vez que o objetivo da geração automática é sintetizar um circuito de um filtro analógico com uma função de transferência e, conseqüentemente, uma determinada curva característica representativa do módulo do ganho, traçada no diagrama de Bode e a mais próxima possível da curva característica de um filtro analógico padrão, escolhido previamente, com um número de componentes previamente determinado, tem-se para resolução um problema de busca por combinações possíveis.

Estas combinações, de topologia e/ou de valores de componentes, os quais podem ser resistores, capacitores e AmpOp's, seguem diretrizes previamente traçadas, para que ao fim de um determinado critério de parada (número de gerações), encontre-se o melhor circuito possível, podendo este ser de topologia o mais próxima possível do circuito clássico, ou não.

O grande motivo de realizarmos a opção pelo emprego de algoritmos genéticos nesta busca é a crença de que os resultados que possivelmente serão alcançados por esta metodologia não o serão por fruto de uma simples busca por força bruta, totalmente cega e exaustiva, que demandariam uma quantidade infinita de recursos computacionais, além de uma grande combinação de fatores de sorte. Sem dúvida alguma, no entanto, o método contrabalança uma componente probabilística, devido ao fato de que existe o sorteio dos componentes, com fatores de inteligência, uma vez que a técnica se vale de experiências passadas para direcionar seus movimentos futuros.

Os aspectos de maior relevância na busca pela solução de um problema por meio de algoritmos genéticos passam, necessariamente, tanto pela representação dos indivíduos quanto pela geração da população inicial, e também pela correta definição dos objetivos a serem alcançados. A seleção dos indivíduos e as operações genéticas neles realizadas são também de importância fundamental, e não poderiam ser deixadas de fora do presente trabalho.

### 3.3.1.1 A Representação dos Indivíduos

A correta representação e modelagem são talvez metade da solução de um problema em diversos campos da ciência. No caso dos indivíduos de uma modelagem por algoritmos genéticos, ela consiste em codificar na forma de uma estrutura de dados parecida com a de um cromossomo, ou seja, em uma seqüência ordenada de números representada em uma base decimal, hexadecimal ou binária, soluções de problemas de engenharia tais como um motor automotivo, um circuito eletrônico analógico ou uma reação química de produção de uma resina.

Neste trabalho, o indivíduo, ou circuito eletrônico, foi codificado do seguinte modo:

Um circuito é um cromossomo que por sua vez é formado por 15 genes, ou componentes eletrônicos, que podem ser passivos (resistores e capacitores) ou ativos (amplificadores operacionais).

Um gene de um componente passivo pode ser criado segundo seis tipos de instruções, representadas na tabela 3.1.

**Tabela 3.1: Instruções para componentes passivos.**

Número da Instrução	Instrução	Próximo nó do circuito	Nó ativo no circuito
0	<i>move-to-new</i>	nó atual + 1	nó atual
1	<i>cast-to-previous</i>	nó atual – 1	nó atual
2	<i>cast-to-ground</i>	Nó 0 (zero) – <i>gnd</i>	nó atual
3	<i>cast-to-input</i>	Nó de entrada – <i>input</i>	nó atual
4	<i>cast-to-output</i>	Nó de saída – <i>output</i>	nó atual
5	<i>cast-to-vcc</i>	Nó de alimentação - <i>vcc</i>	nó atual

Cabe ressaltar que em termos de análise AC, quando um componente passivo é ligado tanto ao nó *vcc* quanto ao nó *vee* a contribuição dos mesmos para a resposta AC do circuito será a mesma, uma vez que na análise AC as fontes de tensão DC são “aterradas”. Deste modo, optou-se por não criar a instrução *cast-to-vee*, de modo a não criar uma instrução a mais que teria a mesma função em termos de análise AC da instrução *cast-to-vcc* .

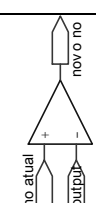
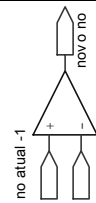
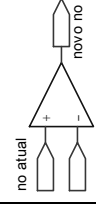
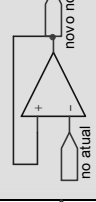
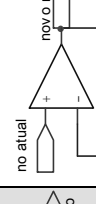
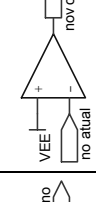

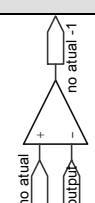
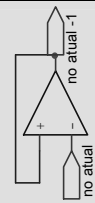
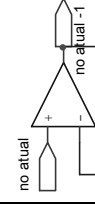
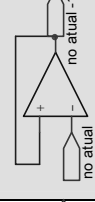
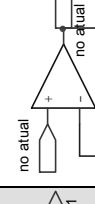


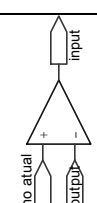
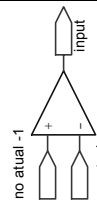
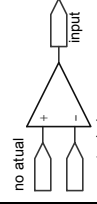
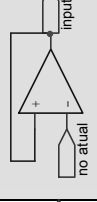
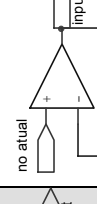

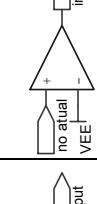
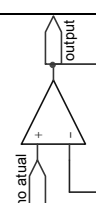
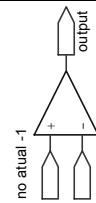
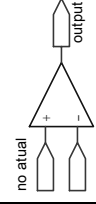
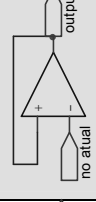
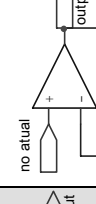


Um gene de um componente ativo pode ser criado segundo quatro tipos de instruções, representadas pelas tabelas 3.2a e 3.2b das duas próximas páginas:



Tabela 3.2.a : Instruções para componentes ativos e topologias possíveis (continua na próxima página).

		Número da Topologia						
		1	2	3	4	5	6	7
Número da Instrução	Instrução							
0	<i>move-to-new</i>							
1	<i>cast-to-previous</i>							
2	<i>cast-to-input</i>							
3	<i>cast-to-output</i>							
						<b>Não permitido</b>		<b>Não permitido</b>

Tabela 3.2.b : (continuação) Instruções para componentes ativos e topologias possíveis.

		Número da Topologia						
		8	9	10	11	12	13	14
Número da Instrução	Instrução	v- to output	v+ to previous	v- to previous	v+ to vout	v- to vout	v+ to vee	v- to vee
0	<i>move-to-new</i>							
1	<i>cast-to-previous</i>							
2	<i>cast-to-input</i>							
3	<i>cast-to-output</i>							
			<b>Não permitido</b>		<b>Não permitido</b>			
			<b>Não permitido</b>		<b>Não permitido</b>			

O gene de um componente, tanto passivo quanto ativo, foi dividido em quatro números, que possuem uma codificação descrita a seguir:

**Tabela 3.3 : Codificação do gene**

<i>first_num</i>	<i>second_num</i>	<i>third_num</i>	<i>fourth_num</i>
------------------	-------------------	------------------	-------------------

*first\_num* – recebe o número da instrução que é sorteada no momento da criação da população. Se for um componente passivo, segue a tabela 3.1; se for ativo, segue as tabelas 3.2.a e 3.2.b. No caso do primeiro gene, ou componente eletrônico, do indivíduo, a primeira instrução não pode ser *cast-to-previous*, ou a instrução *cast-to-input*, e no caso do último, é dependente das ocorrências anteriores da instrução *cast-to-output*, que deve pelo menos ter sido gerada uma única vez, para que seja possível montar um circuito que possua uma topologia passível de ser simulada. Para qualquer outro gene, compreendido entre o primeiro e o último, o sorteio da instrução é livre, desde que já tenha ocorrido, pelo menos uma vez, a instrução *move-to-new*. Caso contrário, as instruções *cast-to-previous* e *cast-to-input* não podem ser sorteadas;

*second\_num* – determina o tipo de componente que foi sorteado. Se for um resistor, o valor de *second\_num* é igual a 0 (zero). Caso seja um capacitor, assume o valor 1 (um) e se for um AmpOp, seu valor será igual a 2 (dois);

*third\_num* – Para o caso dos componentes ativos (AmpOp), assume sempre valor igual a 0 (zero). No caso dos componentes passivos, os valores inteiros entre 0 e 12, inclusive. Cada um destes valores está associado a um valor principal da tabela RETMA 10%, que está na tabela 3.4; e

*fourth\_num* – No caso dos componentes ativos, seu valor assume valores inteiros entre 0 e 14 inclusive, que representam as topologias existentes nas tabelas 3.2.a e 3.2.b, sendo que quando os indivíduos são criados, os casos de topologias não permitidos, que estão grafados em cinza, não são sorteados, visto que nos mesmos existem realimentações positivas, que provocariam resultados inconsistentes nas simulações. Desse modo, tentou-se impedir que o algoritmo criasse circuitos que não teriam qualquer chance de sobrevivência e apenas aumentariam o tempo de processamento, além de impedirem o surgimento de um indivíduo com *fitness* melhor do que o restante da população, que é finita. Para os componentes passivos, o valor de *fourth\_num* assume valores inteiros entre 0 e 6, inclusive, e representa os multiplicadores do valor dos componentes da tabela RETMA 10%.

A tabela 3.4 melhor exemplifica os valores dos componentes passivos segundo a tabela RETMA 10%

**Tabela 3.4 : Codificação dos valores dos componentes passivos.**

<i>third_num</i>	Valor	<i>fourth_num</i>	Multiplicador para Resistor	Multiplicador para Capacitor
0	1	0	$10^0$	$10^{-12}$
1	1,2	1	$10^1$	$10^{-11}$
2	1,5	2	$10^2$	$10^{-10}$
3	1,8	3	$10^3$	$10^{-9}$
4	2,2	4	$10^4$	$10^{-8}$
5	2,7	5	$10^5$	$10^{-7}$
6	3,3	6	$10^6$	$10^{-6}$
7	3,9			
8	4,7			
9	5,6			
10	6,8			
11	8,2			
12	9,1			

A geração dos indivíduos, tanto para algoritmo genético quanto para *particle swarm*, seguem esta mesma regra de formação, existindo, apenas para o segundo caso e que mais a frente será melhor descrito, uma função que transforma a codificação dos genes em partículas. Após o processamento do *particle swarm*, uma função que faz a transformada inversa da codificação converte as partículas em genes no formato acima, para posteriormente serem criadas as *netlists* de modo idêntico ao que é feito para o algoritmo genético.

### 3.3.1.2 A Geração da População Inicial

A geração da população inicial de indivíduos é o ponto de partida de todo algoritmo genético. A população inicial é formada por um número previamente estabelecido de circuitos eletrônicos que podem representar uma solução em potencial para um determinado problema de otimização.

Este número de circuitos eletrônicos (indivíduos) que formarão a população inicial e também o número de componentes eletrônicos (genes) que formam um circuito eletrônico dependem de vários fatores, sendo um dos principais o grau de complexidade do filtro desejado e os recursos computacionais, *hardware* e *software*, disponíveis. A busca que será efetuada pelo algoritmo no universo total de possíveis soluções dependerá, em termos de tempo de processamento, destas variáveis.

Esta geração da população inicial é levada a cabo através de uma rotina que determina de maneira randômica, obedecendo algumas regras de formação descritas no item anterior e uma distribuição uniforme [25], os genes de cada indivíduo.

### 3.3.1.3 Definição dos Objetivos

A modelagem matemática de um universo que será vasculhado por um algoritmo genético que objetiva a obtenção do melhor resultado possível após um dado critério de parada, é, na verdade, um problema que consiste no encontro dos pontos de máximo ou de mínimo de uma função da qual não se tem a expressão analítica, mas tão-somente o domínio. Essas funções, que são denominadas de *funções-objetivo*, são geralmente multidimensionais e não-lineares. Sua formulação matemática é dada pela expressão (3.1).

$$d = f(\text{alvo}, p_1, \dots, p_n) \quad (3.1)$$

Onde  $d$  é um número real que representa o quão próximo um dado conjunto de parâmetros de entrada  $p_1, \dots, p_n$ , está das características que compõem o padrão (indivíduo-alvo). O sucesso da busca depende, prioritariamente, de uma definição correta das especificações do alvo. Quando o algoritmo perseguir múltiplos objetivos, deverão ser determinados os parâmetros em que a característica de proximidade é mais sensível.

No caso desta tese, foram adotados como padrões, obtidos a partir da função *BODE* do *MATLAB*, diversas respostas no domínio da frequência de filtros analógicos conhecidos. Esta lista de pontos é formada pelos vetores com o valor em dB do ganho do filtro, com o valor da fase e com o valor das frequências, em rad/s, em que estes pontos foram amostrados.

A partir dos indivíduos obtidos, são geradas *netlists* que serão simuladas pelo *PSPICE*. Para essas simulações, será utilizado o mesmo vetor com valores de

frequências, em rad/s, que foi gerado pela função *BODE*. Desse modo, poderão ser calculadas, ponto a ponto, as diferenças entre o valor esperado e o valor simulado.

O inverso do erro médio quadrático dos pontos de mesma frequência será então o valor de *fitness* ou aptidão de cada indivíduo. Essa expressão (3.2) está abaixo representada:

$$d = \frac{1}{\frac{1}{n} \sum_{j=1}^n erro_j^2} \quad (3.2)$$

O objetivo a ser alcançado é o de maximizar o valor de  $\underline{d}$ , que pode ser interpretado da seguinte maneira: a aptidão de um indivíduo será tão melhor quanto maior for o valor obtido em sua função-objetivo.

### 3.3.1.4 Seleção dos Indivíduos

A seleção dos indivíduos é o processo por meio do qual os indivíduos de uma dada geração, de acordo com seus valores de aptidão, serão selecionados para serem integralmente reproduzidos na composição de parte da geração seguinte de indivíduos – *nesse caso, netlists de circuitos eletrônicos* – e ainda para serem combinados, entre si, através do *crossover* e também para sofrerem mutação, de modo a formar novos indivíduos que comporão o resto da geração. A figura 3.3 apresenta um esquema ilustrativo da dinâmica de composição das gerações de indivíduos ao longo do processamento de um algoritmo genético.

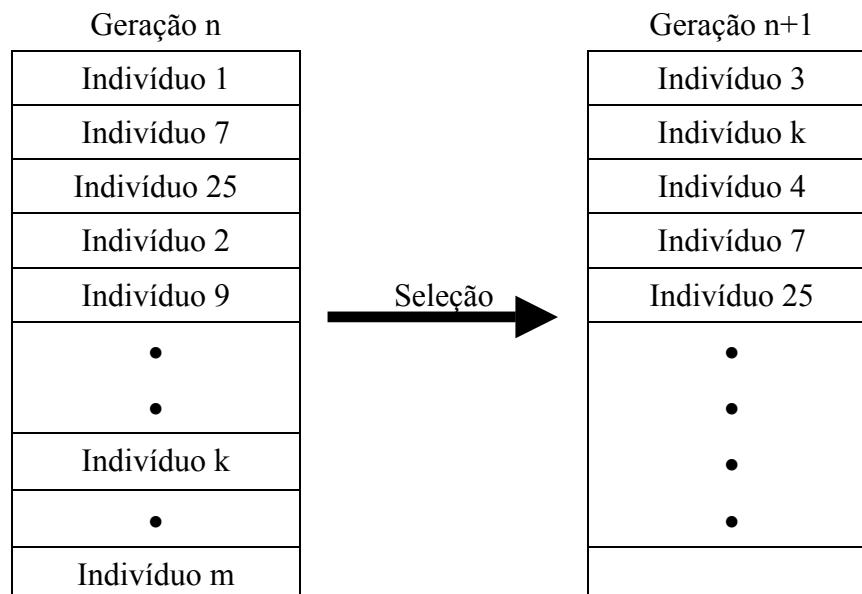


Figura 3.3: Dinâmica da formação de gerações em um algoritmo genético.

Pela figura da página anterior, observa-se que parte da geração seguinte,  $n+1$ , dos circuitos eletrônicos, é formada pela reprodução integral de alguns indivíduos da geração atual  $n$ . A parte restante da população será preenchida por meio da criação de novos indivíduos a partir de cruzamentos e mutações. Pode ser observado, também, que o melhor indivíduo da geração  $n$  era o Indivíduo 1, e que na geração  $n+1$  o indivíduo 3 superou o *fitness* do melhor indivíduo da geração anterior.

Vejam, então, como se dá este processo de seleção dos indivíduos de uma população. Esse mecanismo de seleção é dependente da TAXA DE REPRODUÇÃO escolhida, do método de seleção estabelecido e do emprego ou não do elitismo. A taxa de reprodução é definida *a priori* e determina a percentagem de indivíduos da população da geração atual que serão integralmente reproduzidos na geração seguinte. Para este trabalho, a taxa de reprodução foi estabelecida em 50%, ou seja, sempre a metade de uma população sobrevive integralmente a cada passagem de geração.

O método de seleção escolhido para esta Tese, por sua vez, foi o da roleta [1,2], o qual atribui a cada indivíduo, de acordo com seu respectivo valor de aptidão, uma probabilidade de ser aleatoriamente escolhido. Seja um exemplo de uma população hipotética composta por cinco indivíduos na tabela 3.5 com seus respectivos valores de aptidão em uma dada geração.

**Tabela 3.5: Aptidão dos indivíduos em uma dada geração.**

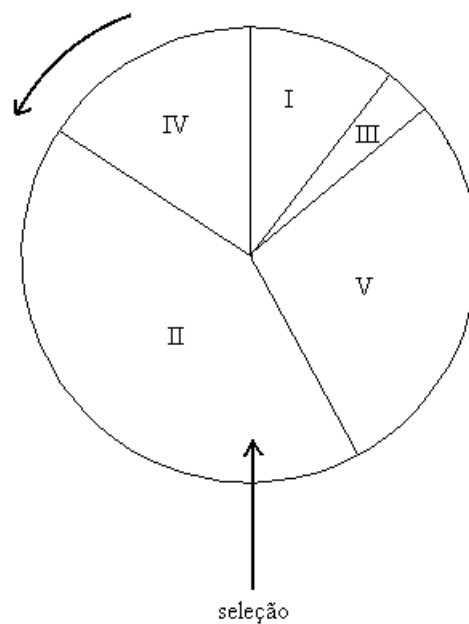
<i>Indivíduo</i>	<i>Aptidão</i>
<i>I</i>	0,025
<i>II</i>	0,344
<i>III</i>	0,00103
<i>IV</i>	0,07
<i>V</i>	0,112
<i>Total</i>	0,55203

A APTIDÃO TOTAL de uma dada população é o somatório dos valores de aptidão de seus indivíduos. A APTIDÃO RELATIVA é a percentagem com que cada indivíduo contribui para a formação da aptidão total. Seja a tabela 3.6, uma reprodução da tabela 3.5, porém acrescida de uma nova coluna de aptidão relativa de cada indivíduo.

**Tabela 3.6: Aptidão e aptidão relativa dos indivíduos em uma dada geração.**

<i>Indivíduo</i>	Aptidão	Aptidão Relativa
<i>I</i>	0,025	4,529%
<i>II</i>	0,344	62,3%
<i>III</i>	0,00103	0,187%
<i>IV</i>	0,07	12,7%
<i>V</i>	0,112	20,3%
<i>Total</i>	0,55203	100%

A aptidão relativa é estabelecida, dessa forma, como a probabilidade de sorteio de um determinado indivíduo em uma população para que este possa ser reproduzido na população da geração seguinte. A metáfora da seleção por roleta pressupõe que a roleta da figura 3.4 seja girada e que, quando parar, seja admitido na nova geração o indivíduo para quem estiver sendo apontada a seta de seleção. No entanto, não seria uma roleta “honesta”, como a que imaginamos em um cassino, mas uma roleta viciada, pois a probabilidade de sorteio não é igual para todos os indivíduos da população.



**Figura 3.4: Método de seleção - roleta.**



Um dos possíveis problemas que essa forma de seleção pode provocar no sucesso da busca é a possibilidade de ocorrer o surgimento de uma nova geração com vários indivíduos rigorosamente iguais, que casualmente podem possuir uma probabilidade de seleção muito elevada, bem maior que a dos demais indivíduos de sua população. Em pouco tempo, esta situação pode causar uma espécie de saturação do algoritmo devido à perda de diversidade de sua população. Deste modo, os valores de aptidão permanecem sempre os mesmos porque os indivíduos são todos iguais. Por isso, existe um mecanismo que é normalmente empregado nos algoritmos genéticos para evitar, ou ao menos atenuar, a probabilidade de ocorrência desta estagnação do *fitness* dos indivíduos de uma população: a mutação, que é um operador genético que será visto mais adiante.

Entretanto existe um outro tipo de entrave no sucesso da busca pela melhor solução, que ocorre devido a uma causa contrária à anterior: a perda do melhor indivíduo da geração anterior.

Devido ao fato de a escolha dos indivíduos ser executada através de um sorteio, existe a possibilidade de ocorrer perda de informação na passagem de uma geração para a geração seguinte, significando um retrocesso na busca.

Como forma de impedir este tipo de problema, neste trabalho, empregou-se o elitismo, que é a reprodução incondicional do melhor indivíduo, o membro da população com mais alto valor de aptidão, na população da geração seguinte.

### **3.3.1.5 Operações Genéticas**

As operações genéticas proporcionam variedade a uma população, visto que possibilitam diversificar aleatoriamente a mesma. Esta diversidade de características entre os indivíduos de uma população é fundamental para o êxito na procura do padrão (indivíduo-alvo), porque as diferenças proporcionam que se descubra, de modo idêntico ao que acontece no meio ambiente, qual dentre os seres vivos de uma determinada espécie possui maior aptidão para se adaptar a um certo habitat.

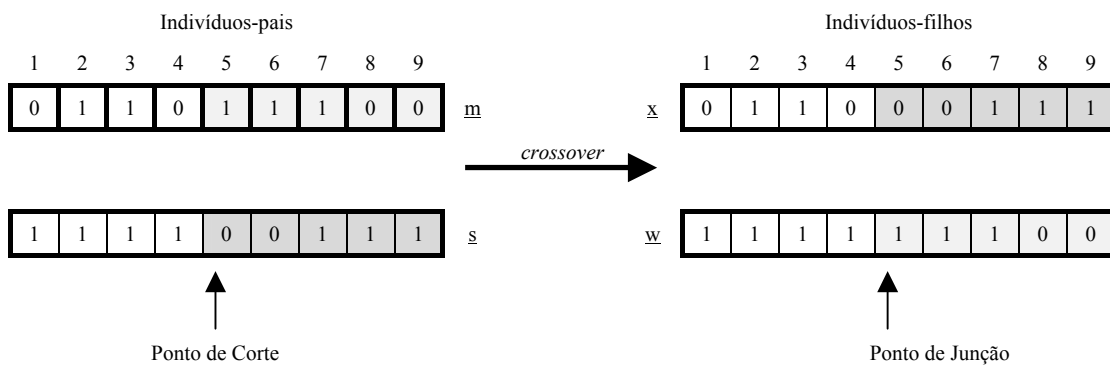
As operações genéticas são realizadas através de operadores genéticos. Os operadores genéticos que foram implementados no presente trabalho são o *crossover* e a mutação.

O *crossover* é o operador que realiza o cruzamento entre dois indivíduos (cromossomos) sobreviventes da geração atual para que o número de indivíduos que

compõem a população da geração seguinte fique completo, ou seja, as populações sempre possuirão o mesmo número de indivíduos da primeira população.

São escolhidos aleatoriamente dois indivíduos-pais, a partir dos sobreviventes da geração atual, para, também de modo aleatório, definir um ponto de corte dos cromossomos. Estes ficarão divididos em quatro partes, as quais serão unidas de modo cruzado, gerando dois filhos, com o mesmo número de genes dos pais, além de possuírem parte das características dos progenitores.

Por facilidade didática, consideremos um exemplo hipotético de uma população de indivíduos que possuem seus 9 genes representados sob forma de números binários. Digamos que os membros desta população realizem a operação de *crossover*, entre os indivíduos-pais  $\underline{m}$  e  $\underline{s}$ , dando origem a dois filhos, ditos  $\underline{x}$  e  $\underline{w}$ . Seja o ponto de corte, escolhido aleatoriamente, no gene de número 5. Na figura 3.5 está representada esta operação genética.



**Figura 3.5: Operação de *crossover* entre dois indivíduos.**

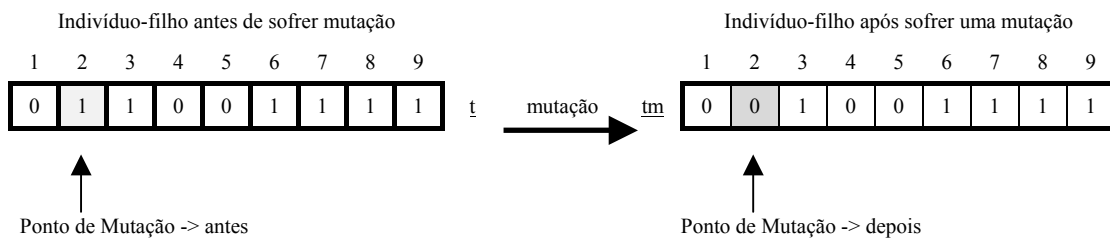
Desta forma, a cada novo cruzamento ou *crossover*, dois indivíduos filhos são gerados e inseridos na população de indivíduos da atual geração, até que toda a metade restante da população seja completamente preenchida. É importante salientar que foi feita a opção de não se alterar as características dos indivíduos-pais, ou seja, seus genes não sofrem qualquer tipo de modificação na geração atual, mantendo, assim, seus valores de *fitness* inalterados.

O operador genético de mutação é aquele que altera, com uma probabilidade predefinida no algoritmo, apenas um dos genes, escolhido ao acaso, dos indivíduos-filhos. Ela é também definida como **PROBABILIDADE DE MUTAÇÃO**. Desempenha a função de impedir a perda de diversidade entre os indivíduos de uma mesma população, que pode ser observado, como já dito, pela “saturação” do *fitness*.

Como um simples exemplo, consideremos a hipótese de um indivíduo filho  $t$  com os mesmos 9 genes, representados binariamente, sofrer mutação. Digamos que o mesmo tenha sido sorteado aleatoriamente, e que este sorteio foi realizado com probabilidades iguais para todos os filhos desta nova população.

Digamos que, também de forma aleatória, foi sorteado o ponto de mutação no gene de número 2. Deste modo, este gene, por somente possuir dois estados lógicos, muda de  $1$  para  $0$ .

A figura 3.6 que está abaixo representada representa o que foi dito sobre a mutação que sofreu o cromossomo  $t$ , passando a ser o cromossomo  $tm$ .



**Figura 3.6: Operação de mutação.**

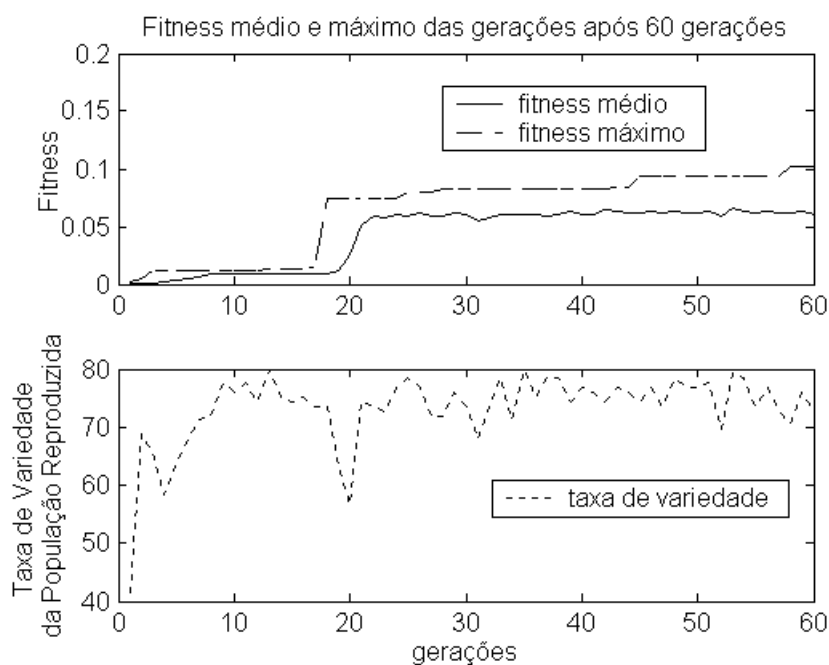
Cabe ressaltar que, se a base fosse outra qualquer ao invés da binária, o gene 2 poderia assumir qualquer valor no intervalo de valores possíveis para este gene. Caso o mesmo fosse subdividido em mais de uma parte, como é o caso implementado nesta Tese, a mudança poderia ocorrer em qualquer uma das quatro subdivisões, visto que é sorteado um indivíduo-filho, e para ele é feito um novo sorteio de qual dos seus genes será alterado. A chance é extremamente baixa de neste sorteio o gene permanecer inalterado, ou seja, de o novo gene criado ser exatamente igual ao anterior.

Neste trabalho, a probabilidade de mutação dos indivíduos-filhos foi definida em 60%, i.e., a cada 100 indivíduos filhos gerados, é provável que sessenta deles sofram mutação. Para o gene sorteado neste indivíduo filho, a probabilidade é igual para todos os genes.

### 3.3.1.6 Análise de Gráficos de Evolução

A observação do gráfico de evolução de um processo de busca efetuado por meio de um algoritmo genético é uma maneira direta de se verificar se esta evolução está ocorrendo de maneira correta. Nesse gráfico são representados, ao longo das gerações, os valores de aptidão médios de cada geração e o valor de aptidão máximo do melhor indivíduo.

Um exemplo desse tipo de gráfico está representado na figura 3.7, que apresenta um processo evolucionário por um total de sessenta gerações, e onde foi adotado o elitismo.



**Figura 3.7: Gráfico de evolução do AG após 60 gerações, para um filtro *Sallen-Key* passa-faixa de segunda-ordem com ganho máximo de 45dB.**

Está representada em um outro gráfico, também na figura 3.7, a taxa de variedade, ao longo das gerações. Esta taxa dá uma medida de quanto os indivíduos de uma população, em determinada geração são diferentes entre si quando passam para a geração seguinte. Uma taxa de variedade de 100% significa que todos os indivíduos sorteados que passam para a metade da população seguinte são diferentes. Em sentido oposto, se esta taxa de variedade for igual a 0%, um mesmo indivíduo foi sorteado todas as vezes e passou para metade da população seguinte.

Portanto, se durante a execução do método de seleção pela roleta, um elemento com alta aptidão relativa é sorteado várias vezes, existe a tendência de o mesmo ser repetido pelo mesmo número de vezes na população da geração subsequente, ou seja, existirão vários indivíduos iguais ao sorteado naquela nova população. Com isso, o algoritmo terá a tendência de saturar.

Pode ser observado também no mesmo gráfico que pela conjunção de diversos fatores, quer sejam os *crossovers*, as mutações ou as combinações de ambos, em determinadas gerações, ocorrem saltos de qualidade de seus indivíduos devido ao aumento em degrau da aptidão máxima. Entretanto, por um grande número de gerações o melhor indivíduo permanece inalterado, conforme pode ser observado pelo valor de *fitness* máximo que permanece em um determinado valor fixo.

Quando por muitas gerações esta imutabilidade do melhor indivíduo ocorrer, pode ser necessário, por exemplo, um aumento da probabilidade de mutação, ou talvez um outro novo método de aumento da variedade da população, como, por exemplo, a geração por sorteio de uma nova segunda metade da população, ou ainda a aplicação de uma outra técnica que realize o *tuning* dos componentes em termos de seus valores.

Estas são algumas das conclusões a que se pode chegar depois de realizarmos a análise dos dados de um processo evolucionário cujo objetivo é gerar indivíduos cujas respostas mais se aproximem em valor absoluto do padrão (indivíduo-alvo).

### 3.3.2. Geração de circuitos de filtros analógicos por *Particle Swarm*

Os algoritmos genéticos foram utilizados nesta Tese como um método robusto de busca cujo objetivo é o de sintetizar um circuito de um filtro analógico com uma função de transferência e, conseqüentemente, uma determinada curva característica, representativa do módulo do ganho, traçada no diagrama de Bode, que se aproxime o mais possível da curva característica de um filtro analógico padrão.

Entretanto, a ciência evolui com a produção de novos conhecimentos que permitem a constante mudança de idéias pré-estabelecidas. Na tentativa de encontrar novas soluções para o mesmo problema, foram também implementados nesta Tese os algoritmos do tipo *Particle Swarm*, ou técnicas sócio-cognitivas, aproveitando parte da codificação e da modelagem já implementadas para os algoritmos genéticos, de modo a permitir a realização de comparações entre o desempenho dos algoritmos do tipo *Particle Swarm* com o dos algoritmos genéticos.

Muitas das considerações adotadas para a construção dos algoritmos genéticos também o foram na implementação do *Particle Swarm*. Assim sendo, para não tornar a esta dissertação repetitiva e cansativa, com a descrição pormenorizada de toda a metodologia, esta classe de algoritmos será descrita comparativamente, com base nas suas principais diferenças com relação às técnicas evolutivas.

A implementação do algoritmo de *Particle Swarm* é razoavelmente simples, mas no entanto seus resultados podem até ser comparáveis com os obtidos das técnicas evolutivas, dependendo do tipo de problema que se deseja resolver. No entanto, o fato de não possuir o elitismo e de depender de uma população inicial com indivíduos com alguma aptidão, pode levar o algoritmo a ficar preso em mínimos e máximos locais da função-objetivo, que geralmente é multidimensional e não-linear, sem uma forma analítica conhecida.

As partículas serão vetores de dimensão  $\underline{n}$ , onde  $\underline{n}$  é igual ao número de genes (componentes) da população a ser convertida, e cada componente do vetor deve possuir um intervalo contínuo de valores inteiros para a correta representação dos índices das partículas. Deste modo, cada cromossomo, que representa um circuito eletrônico no AG, passará a ser, após devidamente convertido, uma partícula no PS. De maneira semelhante, cada gene do cromossomo, que no AG representa um componente eletrônico, será uma dimensão do espaço ou “hiperespaço” que as partículas poderão

ocupar, no caso do PS. A cada componente do vetor dá-se o nome de índice da partícula.

Relembrado o que já foi dito na representação dos indivíduos no AG, a codificação de um componente passivo ou ativo é realizada pela subdivisão do gene em quatro números: *first\_num*, *second\_num*, *third\_num* e *fourth\_num*. A combinação dos mesmos determina um e somente um componente interligado a outros através de uma instrução, ou seja, existe uma relação unívoca entre um componente (tipo, valor e instrução, que define a topologia de acordo com a qual o mesmo está associado aos outros componentes do circuito) e seu gene. De maneira inversa, existe uma relação unívoca entre um gene e o seu componente. Esta propriedade será melhor exemplificada quando for detalhada a operação de conversão de uma população em uma partícula.

Realizando uma comparação com os algoritmos genéticos, a geração da população inicial e a definição dos objetivos são iguais. A principal diferença concentra-se naquilo que nos algoritmos genéticos é conhecido por operadores genéticos, que são utilizados durante a seleção dos indivíduos. Essas fases são substituídas na nova técnica pela avaliação, comparação e imitação. As técnicas sócio-cognitivas não necessitam de uma codificação decimal, hexadecimal ou binária para cada indivíduo, uma vez que não existem as operações genéticas de *crossover* e mutação.

Uma das maiores diferenças de fundamento entre as duas técnicas está no conceito de população. Nos algoritmos genéticos, o índice de um indivíduo numa dada população é apenas um espaço por ele circunstancialmente ocupado. Os indivíduos morrem, passam a não constar na população da geração seguinte e, com isso, deixam de ocupar as posições na população. Portanto, cada cromossomo representa um indivíduo único e indistinguível, e o caráter de indistinguibilidade reside em suas próprias características. No *Particle Swarm*, cada índice da população é uma partícula que eventualmente muda suas características, mas nunca deixa de ocupar uma dada posição no índice da população. Seu caráter de indistinguibilidade reside, pois, na posição que a partícula ocupa na população. Assim, uma partícula pode mudar de posição no espaço de busca, o que significa a alteração das suas características, tornando-as mais próximas das especificadas para o indivíduo-alvo.

Determinada a diferença fundamental das metáforas nas quais se espelham os dois métodos de busca, passa-se à descrição daquilo que se pode considerar análogo à seleção nos algoritmos genéticos: avaliação, comparação e imitação.

### 3.3.2.1 Avaliação

A avaliação é, a rigor, o mesmo processo que calcula os valores de aptidão nos algoritmos genéticos, não tendo sido alterada nesta nova implementação.

### 3.3.2.2 Conversão da População PS em Partículas

Para tornar possível aproveitar toda a metodologia de construção de uma população, gerada através de um mecanismo que impede o aparecimento de circuitos que possuam erros topológicos, como o de grafos sem componentes que acoplem a entrada à saída, podendo assim haver a perda parcial ou total desta população, foram implementadas funções que realizam a conversão da população codificada em um enxame de partículas.

#### 3.3.2.2.1 Topologias e Valores de Componentes Variáveis

Como foi mostrado anteriormente, é possível associar cada tipo de componente a um número inteiro, até o limite do número de alternativas possíveis para este componente, que seria a multiplicação da máxima variação do *first\_num*, com o *third\_num* e por fim com o *fourth\_num*, no caso dos componentes passivos, e no caso dos componentes ativos, o número de topologias possíveis existentes nas tabelas 3.2.a e 3.2.b.

Para melhor exemplificar, tomemos o caso dos resistores:

*second\_num* é igual a 0 (zero), pois é o valor associado que indica que o componente é um resistor;

*first\_num* pode assumir valores entre 0 e 5, ou seja, possui 6 (seis) tipos de instruções na tabela 3.1;

*third\_num* pode assumir valores entre 0 e 12, inclusive, ou seja, possui 13 (treze) tipos de valores principais na tabela 3.4; e

*fourth\_num* pode assumir valores entre 0 e 6, inclusive, ou seja, possui 7 (sete) tipos de multiplicadores na tabela 3.4.

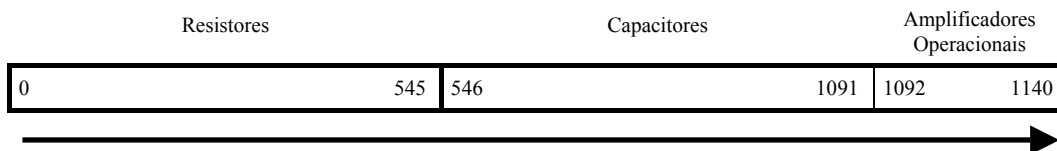
Assim sendo, existem 6 vezes 13 vezes 7, ou seja, 546 possíveis resistores, variando um a um em termos de valores principais, multiplicadores e topologias de associação com outros componentes. Desse modo, cada número no intervalo entre 0 e 545, inclusive, representa um e somente um resistor.



Podemos fazer o mesmo raciocínio para os capacitores. Como no caso destes o *second\_num* é igual a 1 (um), e os demais números possuem os mesmos intervalos de validade, basta considerar os capacitores ocupando o segmento de valores que vai do valor 546 ao 1091, inclusive. Seria como somar um *off-set* de 546 posições aos resultados obtidos para os capacitores.

Da posição 1092 em diante, os valores serão ocupados pelos componentes ativos, que possuem 49 possíveis topologias. No caso dos mesmos, optou-se por somente associar a um número inteiro as topologias possíveis a partir de 1092, da esquerda para a direita, de cima para baixo, nas tabelas 3.2.a e 3.2.b, chegando ao valor máximo de 1140, que corresponde à instrução número 3 (*cast-to-output*) na topologia 14 (v- to vee).

A figura 3.8 representa o intervalo de valores inteiros possíveis para o caso de topologias e valores de componentes variáveis.



**Figura 3.8: Intervalo de valores inteiros - topologias e valores de componentes variáveis.**

### 3.3.2.2.2 Topologias Fixas e Valores de Componentes Variáveis

Para o caso de se manter as topologias fixas, apenas os resistores e os capacitores possuem interesse em termos de seus valores nominais; assim ficam de fora da função de conversão suas topologias, que são definidas pelo código das instruções, e as dos componentes ativos. Estas topologias serão recuperadas pela função que faz a transformada inversa desta conversão, e que será mais à frente elucidada, pois será extraída da população anterior ao deslocamento das partículas a informação topológica de cada indivíduo, sendo o mesmo formado de componentes passivos ou ativos. Assim sendo, apenas os valores dos resistores e dos capacitores podem sofrer variação.

É, portanto, possível associar cada tipo componente passivo a um número inteiro, até o limite do número de alternativas possíveis para este componente, que seria a multiplicação da máxima variação do *third\_num* com o *fourth\_num*.

Para melhor exemplificar, tomemos o caso dos resistores:

*second\_num* é igual a 0 (zero), pois é o valor associado que indica que o componente é um resistor;

*third\_num* pode assumir valores entre 0 e 12, inclusive, ou seja, possui 13 (treze) tipos de valores principais na tabela 3.4; e

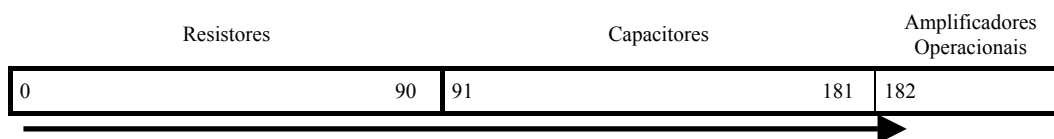
*fourth\_num* pode assumir valores entre 0 e 6, inclusive, ou seja, possui 7 (sete) tipos de multiplicadores na tabela 3.4.

Assim sendo, existem 13 vezes 7, ou seja, 91 possíveis resistores, variando um a um em termos de valores principais, multiplicadores e topologias de associação com outros componentes. Desse modo, cada número no intervalo entre 0 e 90, inclusive, representa um e somente um resistor.

Podemos fazer o mesmo raciocínio para os capacitores. Como no caso destes *second\_num* é igual a 1 (um), e os demais números possuem os mesmos intervalos de validade, basta considerar os capacitores ocupando o segmento de valores que vai do valor 91 ao 181, inclusive. Seria como somar um *off-set* de 91 posições aos resultados obtidos para os capacitores.

Como já foi dito, os componentes ativos ficam de fora desta conversão, pois somente as topologias dos mesmos sofrem variação, e nesse caso elas foram mantidas fixas. Para possibilitar que a função que implementa a transformada inversa desta conversão reconheça os componentes ativos quando ainda sob a forma de índices de partículas, foi associado somente o número inteiro 182 aos componentes ativos para o caso de topologias fixas e valores de componentes variáveis.

A figura 3.9 representa o intervalo de valores inteiros possíveis para o caso de topologias fixas e valores de componentes variáveis.



**Figura 3.9: Intervalo de valores inteiros - topologias fixas e valores de componentes variáveis.**

### 3.3.2.3 Comparação

Na comparação entre as partículas, surge um aspecto importante que não estava presente na técnica evolutiva: a topologia de influência entre os elementos da população. Conforme foi visto no item 2.2.1, as partículas podem sofrer influência de

seus vizinhos mais próximos e da melhor partícula dentre todas, segundo alguns esquemas que foram apresentados nas figuras 2.3 a 2.6. Assim sendo, a direção e a distância que uma partícula percorre entre uma iteração e outra dependem, exclusivamente, de dois fatores: do comportamento de seus vizinhos e do comportamento da partícula mais apta dentre todas.

Nesta tese foi empregada a topologia em círculo, que está apresentada na figura 2.5, pelo fato de ser a mais eficaz em evitar a convergência para máximos locais, visto que os indivíduos distantes daquele que se achar num máximo local não sofrerão influência imediata por este, tendendo a continuar a busca por outras regiões do “hiperespaço” [15,16,17]. Dessa forma, cada partícula pode sofrer influência, localmente, por apenas dois vizinhos: os que lhe são imediatamente anterior e posterior. Esses vizinhos são comparados e armazena-se aquele com maior valor de aptidão. Globalmente, a partícula sempre será influenciada pela partícula mais apta.

Assim, dado o conjunto de expressões que descrevem o deslocamento das partículas, reproduzido abaixo, tem-se, nesta fase, a determinação de  $p_i$  e  $p_g$  na expressão (3.3) do sistema abaixo.

$$\begin{cases} v_i(t) = v_i(t-1) + \varphi_1 [p_i - x_i(t-1)] + \varphi_2 [p_g - x_i(t-1)] & (3.3) \\ x_i(t) = x_i(t-1) + v_i(t) & (3.4) \end{cases}$$

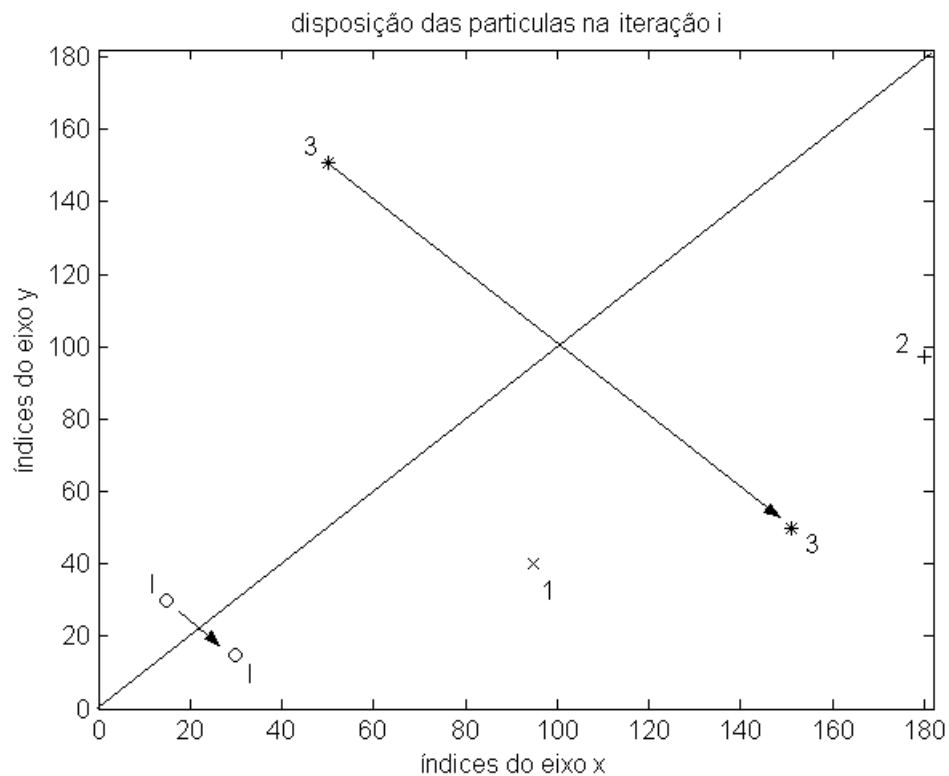
### 3.3.2.4 Imitação

A consolidação dos novos valores de posição e de velocidade que serão atribuídos às partículas de uma determinada população ocorre no processo de imitação. Uma forma de representação dos circuitos eletrônicos precisa ser utilizada nesta etapa, conforme foi visto anteriormente quando da conversão da população em partículas, embora não se requeira a complexidade da codificação que foi empregada no algoritmo genético. No entanto, este fator é demeritório para a técnica sócio-cognitiva, uma vez que, ao contrário da técnica evolutiva, as informações de topologia existentes na codificação dos cromossomos acabam sendo perdidas, caso não sejam mantidas quando da conversão para as partículas e depois, quando da aplicação da transformada inversa desta conversão. As técnicas evolutivas, por possuírem elitismo e a capacidade de selecionar e reproduzir através do método de roleta os melhores indivíduos, acabam intrinsecamente preservando as topologias melhores.

No caso das técnicas sócio-cognitivas, ao se permitir mexer com as topologias e os valores simultaneamente, podem surgir grafos sem componentes que acoplem a entrada à saída. Neste caso, as simulações realizadas no *SPICE* apresentariam erros, diminuindo a massa de indivíduos simulados e, conseqüentemente, a população que realmente contribui para a evolução.

A representação utilizada para a partícula foi por índices. Relembrando, um circuito eletrônico que foi transformado em uma partícula é composto por componentes passivos e ativos, que foram transformados em índices destas partículas, conforme representado nas figuras 3.8 e 3.9.

Para que se possa visualizar a dinâmica do processo sócio-cognitivo adotada neste trabalho, do qual a imitação é seu último passo, seja o exemplo da figura 3.10, que apresenta quatro partículas representativas de circuitos eletrônicos com dois componentes passivos ou ativos cada circuito, em um processo sócio-cognitivo onde somente os valores dos componentes passivos variam, ficando a topologia fixa.



**Figura 3.10: Partículas de um processo sócio-cognitivo qualquer.**

Os eixos representam os índices de cada partícula, que no caso é um circuito eletrônico. A tabela 3.7 apresenta os valores das partículas na figura, com seus valores de aptidão já calculados.

**Tabela 3.7: Valores das partículas do exemplo da figura 3.10.**

<i>Partícula</i>	<i>Circuito Eletrônico Correspondente</i>	<i>Índices da Partícula</i>	<i>Aptidão</i>
I	pop(I,1) = [0 0 2 1] pop(I,2) = [4 0 4 2]	[15 30]	3
1	pop(1,1) = [0 1 4 0] pop(1,2) = [3 0 1 3]	[95 40]	3.5
2	pop(2,1) = [0 1 11 6] pop(2,2) = [2 1 6 0]	[180 97]	3.7
3	pop(3,1) = [0 0 11 3] pop(3,2) = [0 1 8 4]	[50 151]	4

A partícula  $\underline{i}$  tem como seus vizinhos, considerando-se uma topologia em círculo, as partículas um e dois. A partícula três é a mais apta da população. Considerando-se que cada componente do circuito eletrônico pode assumir somente um valor inteiro entre 1 e 182 e, ainda, que a função-objetivo é insensível à ordem dos índices da partícula –*por exemplo*,  $f([50 \ 151]) = f([151 \ 50])$ , delimita-se o domínio da função-objetivo por meio da diagonal traçada na figura 3.10.

Uma vez que a partícula vizinha mais apta de  $\underline{i}$ , ou seja,  $p_i$ , é a partícula dois, aplica-se a expressão (3.3) para calcular as componentes  $x$  e  $y$  de sua velocidade, assumindo-se que sua velocidade anterior era zero e que os valores estocásticos  $\varphi_1$  e  $\varphi_2$  foram estabelecidos em, respectivamente, 0.76 e 0.29:

$$\left\{ \begin{array}{l} v_i(t) = v_i(t-1) + \varphi_1 [p_i - x_i(t-1)] + \varphi_2 [p_g - x_i(t-1)] \end{array} \right. \quad (3.3)$$

$$\left\{ \begin{array}{l} x_i(t) = x_i(t-1) + v_i(t) \end{array} \right. \quad (3.4)$$

$$\left\{ \begin{array}{l} v_{ix}(t) = 0 + 0.76[180 - 15] + 0.29[50 - 15] \end{array} \right. \quad (3.5)$$

$$\left\{ \begin{array}{l} v_{iy}(t) = 0 + 0.76[97 - 30] + 0.29[151 - 30] \end{array} \right. \quad (3.6)$$

Dá se chega ao valor da velocidade que a partícula alcançará:

$$v_i(t) = [135.55 \quad 86.01] \quad (3.7)$$

Uma vez que a próxima posição que a partícula  $i$  alcançará é formada por números inteiros, assume-se apenas a parte inteira da velocidade. Assim:

$$v_i(t) = [135 \quad 86] \quad (3.8)$$

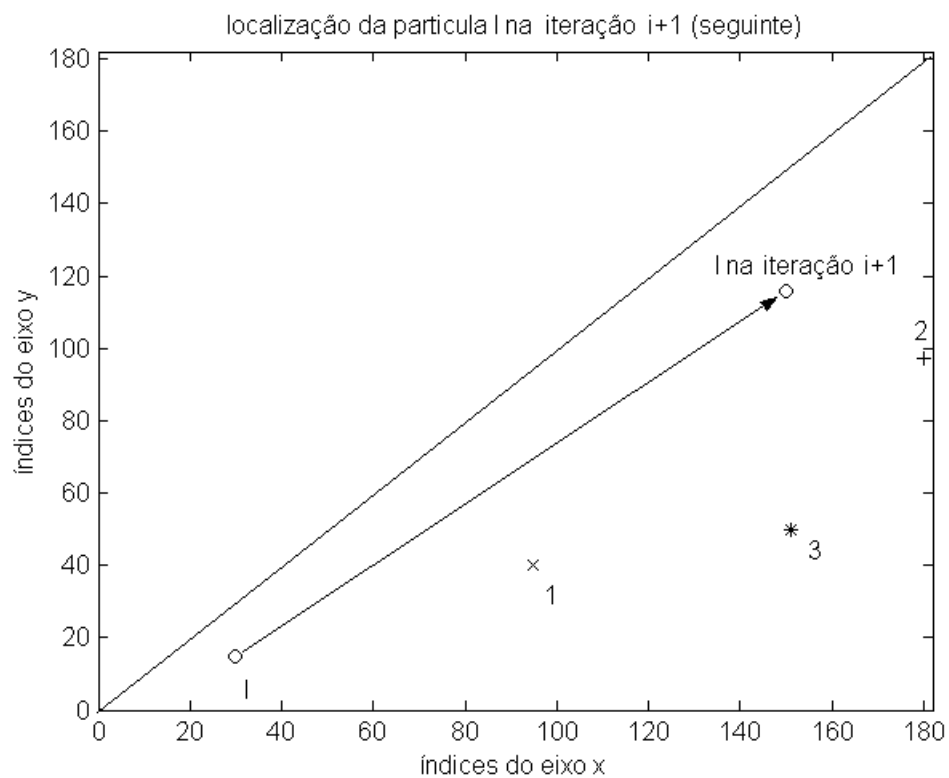
Aplicando-se (3.4), a nova posição ocupada pela partícula  $i$  será, então:

$$x_i(t) = [150 \quad 116] \quad (3.9)$$

E esta posição corresponde ao circuito eletrônico:

$$\begin{aligned} \text{pop}(I,1) &= [0 \quad 1 \quad 7 \quad 4] \\ \text{pop}(I,2) &= [4 \quad 1 \quad 12 \quad 1] \end{aligned} \quad (3.10)$$

A figura 3.11 apresenta a mudança de posição da partícula  $i$  na iteração seguinte.



**Figura 3.11: Mudança de posição da partícula  $i$ .**

Na implementação desta classe de algoritmo, deve-se tomar cuidado para que o valor da velocidade não seja tão grande a ponto de movimentar a partícula para fora do domínio de interesse da função-objetivo, o que a tornaria não mais válida. Por este motivo, cria-se um LIMITE DE VELOCIDADE. Esse limite objetiva diminuir o número de partículas que, numa dada iteração, possam exceder simultaneamente o domínio da

função-objetivo. Costuma-se implementar o limite de velocidade da seguinte forma, apresentada no quadro 3.1.

```
If  $v_i > V_{max}$ 
     $v_i = V_{max}$  ;
else
    If  $v_i < -V_{max}$ 
         $v_i = -V_{max}$  ;
```

**Quadro 3.1: Implementação do limite de velocidade.**

O limite de velocidade estabelecido neste trabalho foi de 10% do tamanho do domínio em cada eixo, ou seja, se o domínio de cada eixo está compreendido entre 1 e 182, a velocidade máxima será sempre 18 ou -18.

O limite de velocidade reduz a probabilidade de ultrapassagem dos limites do domínio por uma partícula, mas não a evita por completo. Portanto, quando uma partícula projeta-se para fora do domínio válido, o algoritmo a traz de volta, atribuindo-lhe o valor máximo permitido do índice que teve seu valor ultrapassado. Assim, por exemplo, se a posição de uma partícula é calculada em [-2 113], o algoritmo a modifica para [0 113].

### **3.3.2.5 Conversão das Partículas Evoluídas em População PS Evoluída**

Como já foi informado anteriormente, as topologias são recuperadas por uma função que faz a transformada inversa da conversão das Partículas Evoluídas em População PS Evoluída.

Faz-se necessário continuar a aproveitar toda a metodologia de construção de uma população, gerada através de um mecanismo que impede o aparecimento de circuitos que não seriam simulados, por conter erros de topologia em sua estrutura.

Na seção 3.3.2.2, foi visto que a codificação de um componente passivo ou ativo é realizada pela subdivisão do gene em quatro números e que a combinação dos mesmos determina um e somente um componente interligado a outros através de uma instrução. Portanto, existe uma relação unívoca entre um componente (tipo, valor e instrução, que define a topologia e acordo com a qual o mesmo está associado aos outros componentes do circuito) e seu gene; e de maneira inversa, existe uma relação unívoca entre um gene e o seu componente. Esta última relação será utilizada agora para implementar a

recuperação de uma população que será simulada, sob a forma de *netlists*, na iteração seguinte.

### **3.3.2.5.1 Topologias e Valores de Componentes Variáveis**

Como foi mostrado anteriormente, associou-se cada tipo componente a um número inteiro, até o limite do número de alternativas possíveis para este componente, que seria a multiplicação da máxima variação do *first\_num*, com o *third\_num* e por fim com o *fourth\_num*, no caso dos componentes passivos, e no caso dos componentes ativos, o número de topologias possíveis existentes nas tabelas 3.2.a e 3.2.b..

Para implementar a transformada inversa, no caso dos componentes passivos, basta tomar cada número que foi gerado como explicado acima e realizar a divisão inteira do mesmo pelo número de possíveis componentes passivos, que neste caso é de 546. O valor inteiro encontrado no intervalo entre 0 e 2 será o *second\_num* do componente. O resto de *second\_num* será então dividido, como divisão inteira, pelo intervalo de variação de cada componente passivo, que no caso é igual a 13 vezes 7, ou seja, 91. O valor encontrado será o *first\_num* para os componentes passivos. Novamente o resto de *first\_num* será utilizado em uma divisão inteira para agora encontrar o *fourth\_num* do componente, sendo o divisor inteiro igual a 13. Por fim, o resto de *fourth\_num* encontrado será o *third\_num* do componente. Como visto, podemos encontrar novamente o gene do componente no formato utilizado nesta Tese.

Para os componentes ativos, optou-se por fazer esta transformada inversa através de uma tabela, uma vez que existem apenas 49 possibilidades.

Vale ressaltar que durante o processamento desta função, são realizados testes para que os números gerados não fiquem fora dos intervalos de existência determinados para cada um. Isso poderia ocorrer caso uma ou mais partículas saíssem do intervalo válido do “hiperespaço”, devido a um mínimo fora do mesmo. Mesmo com a limitação de velocidade implementada, podem ocorrer situações em que isso ocorreria, e é importante reconhecer e corrigir esses problemas nesta função.

### **3.3.2.5.2 Topologias Fixas e Valores de Componentes Variáveis**

Foi implementado de modo análogo ao caso anterior, sendo que a diferença está no fato de que a população anterior ao deslocamento deve ser entregue à função de conversão inversa para que as informações de topologia sejam recuperadas.



Para implementar a transformada inversa, no caso dos componentes passivos, basta pegar cada número que foi gerado, como acima explicado, e realizar a divisão inteira do mesmo pelo número de possíveis componentes passivos, que neste caso é de 91. O valor inteiro encontrado no intervalo entre 0 e 2, inclusive, será o *second\_num* do componente. Caso o valor encontrado seja igual a 0 ou 1, teremos um componente passivo. O resto de *second\_num* será utilizado em uma divisão inteira para agora encontrar o *fourth\_num* do componente, sendo o divisor inteiro igual a 13. Por fim, o resto de *fourth\_num* encontrado será o *third\_num* do componente. Podemos, portanto, encontrar novamente o gene do componente no formato utilizado nesta Tese.

Se o valor encontrado para o *second\_num* for igual a 2, o componente encontrado será ativo.

Em ambos os casos, de componentes passivos ou ativos, o *first\_num* é obtido diretamente do *first\_num* da população anterior.

### **3.3.2.6 Determinação do Melhor Indivíduo da Iteração Corrente**

A seleção do melhor indivíduo é realizada através da determinação do máximo *fitness* entre as partículas e a atribuição do *status* de melhor à partícula que o possui.

Como forma de verificar se a seleção do melhor indivíduo pela técnica sócio-cognitiva estava funcionando de modo correto, foi utilizada parte da função de seleção implementada para as técnicas evolutivas, sendo retirada da mesma toda a parte de sorteio pelo método de roleta, e aplicada à população PS evoluída após a transformada inversa, que recuperou a codificação sob a forma de genes. Os resultados encontrados foram iguais.

#### 4. Resultados

Objetivando realizar a comparação do desempenho das técnicas evolutiva e socio-cognitiva através da metodologia descrita até o presente capítulo, foram selecionadas como funções-objetivo as respostas do módulo do ganho no domínio da frequência, obtidas diretamente do diagrama de Bode, de três tipos de filtros analógicos de segunda-ordem conhecidos, cada qual com três valores de ganho máximo.

Esses filtros possuem espectros de frequência entre  $10^2$  e  $10^5$  rad/s e com diferentes características: *passa-baixas* e *passa-faixa*.

Os processos de geração dos indivíduos da população inicial, que é utilizada tanto para a técnica evolutiva quanto para a socio-cognitiva, utilizam as seguintes condições, para cada um dos exemplos: emprego de 250 indivíduos ou partículas; cada indivíduo da população é um cromossomo formado de 15 genes, ou seja, componentes passivos ou ativos; e o número de gerações do AG é igual ao número de iterações do PS, levando-se em conta os recursos computacionais disponíveis.

Devido a estes recursos limitados, foi feita a opção de não rodar continuamente o algoritmo até a quantidade de gerações desejada, uma vez que, caso ocorresse um problema de *software*, *hardware* ou até mesmo de fornecimento de energia elétrica, as simulações poderiam ser perdidas. Optou-se, então, por trabalhar com blocos de simulações por um número  $n$  de gerações previamente definido. Ao final de cada bloco, os dados são salvos e as últimas populações (ou partículas) geradas são usadas no bloco de simulações de  $m+n$  gerações que se deseja simular.

Foram simuladas: 30 gerações contínuas; mais 30 gerações totalizando 60; e 60 gerações contínuas, para todos os filtros. Para o filtro *Sallen-Key* passa-faixas de segunda ordem, que apresentou os melhores resultados, foram feitas as seguintes simulações adicionais: mais 60 gerações, totalizando 120; e 120 gerações contínuas (ainda em execução até o presente momento!).

Como o *Particle Swarm* não é elitista e trabalha somente com a mesma população inicial de partículas, foi efetuado o teste de manter ou não a topologia fixa, de modo a verificar se a quantidade de circuitos não simulados aumenta no segundo caso devido a perda das topologias construídas segundo o método de criação da população inicial que impede o surgimento de indivíduos com problemas topológicos.

Também no caso do *Particle Swarm*, foram efetuados dois testes de verificação da capacidade evolutiva desta técnica: um desses testes consistia em fazer funcionar o algoritmo de modo normal, entregando ao mesmo uma população não especializada igual à que fora criada para o Algoritmo Genético; e de modo inverso, era disponibilizada para o *Particle Swarm* uma população já evoluída pelo Algoritmo Genético após  $n$  gerações.

Para cada uma das técnicas empregadas, foi salva a *netlist* do melhor indivíduo de cada geração, além da *netlist* dos melhores indivíduos gerados ao final das  $n$  gerações definidas para cada bloco de simulações.

Foram realizadas medidas de desempenho do tempo de duração do Algoritmo Genético de uma geração, do *Particle Swarm* de uma iteração, além do tempo total de execução do método.

Por fim, foram rodadas simulações no *SPICE* empregando o método de Monte Carlo para verificação de robustez dos melhores indivíduos (circuitos eletrônicos) gerados tanto pela técnica evolutiva quanto pela socio-cognitiva. Foram geradas variações gaussianas de 10% nos valores nominais dos componentes passivos e ativos dos circuitos, um tipo de componente de cada vez. Deste modo, estas simulações foram geradas para variações ora dos resistores, ora dos capacitores, e por fim dos amplificadores operacionais.

A tabela 4.1 apresenta um resumo dos experimentos efetuados para o teste do método.

Tabela 4.1: Resumo dos testes efetuados.

<i>Filtro</i>	<i>Ganho máx. em dB</i>	<i>Métodos de busca empregados e Topologia</i>	<i>Número de Gerações ou interações.</i>				
			n ou m Gerações Contínuas		PopPS = PopPS anterior.	PopPS = PopAG anterior.	PopPS = PopAG anterior.
			n	m	+n	+n	+m
<i>Butterworth</i> passa- baixas de 2 <sup>a</sup> . ordem	+30	AG, PS variável	70	X	X	X	X
<i>Butterworth</i> passa- baixas de 2 <sup>a</sup> . ordem	+30	AG, PS fixa	30	60	X	60	X
	0	AG, PS fixa	30	60	X	60	X
	-30	AG, PS fixa	30	60	X	60	X
<i>Sallen-Key</i> passa- baixas de 2 <sup>a</sup> . ordem	+45	AG, PS fixa	30	60	X	60	X
	+5	AG, PS fixa	30	60	X	60	X
	-35	AG, PS fixa	30	60	X	60	X
<i>Sallen-Key</i> passa-faixa de 2 <sup>a</sup> . ordem	+45	AG, PS fixa	30	60	X	60	* → 120
	+5	AG, PS fixa	30	60	X	X	* → 120
	-35	AG, PS fixa	30	60	X	X	* → 120

\* → 120 gerações contínuas, simulações ainda em execução até o presente momento

#### 4.1. Butterworth passa-baixas de 2a. ordem

Um diagrama de Bode para este filtro está ilustrado na figura 4.1.

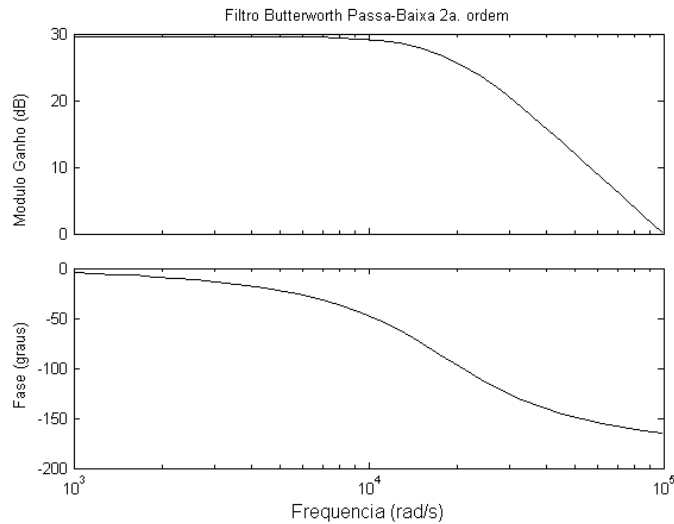


Figura 4.1: Diagrama de Bode do filtro *Butterworth* passa-baixas de 2a. ordem.

##### 4.1.1. Topologias fixas e Valores de Componentes Variáveis por 30 gerações

A população inicial de 250 indivíduos de 15 genes cada um foi evoluída inicialmente por 30 gerações continuamente, para cada um dos ganhos máximos pré-estabelecidos para a função-objetivo em questão. A tabela 4.1 e as Figuras 4.2.a, 4.2.b e 4.2.c mostram os resultados alcançados. O índice K significa Geração/Iteração.

Tabela 4.1: Resumo dos testes com 30 gerações evoluindo continuamente.

<i>Métodos empregados e Topologia</i>	<i>Ganho máx. (dB)</i>	<i>fitness máximo</i>	<i>Duração loop (s)</i>	<i>Método Evoluiu, ficou estável ou regrediu?</i>	<i>Converge em quantas gerações?</i>
AG	+30	0,04	188	evoluiu	10
	0	0,044	210	evoluiu	5
	-30	0,021	496	evoluiu	5
PS com topologia fixa	+30	0,013	186	regrediu	3
	0	0,021	208	ficou estável	-
	-30	0,018	489	regrediu	2

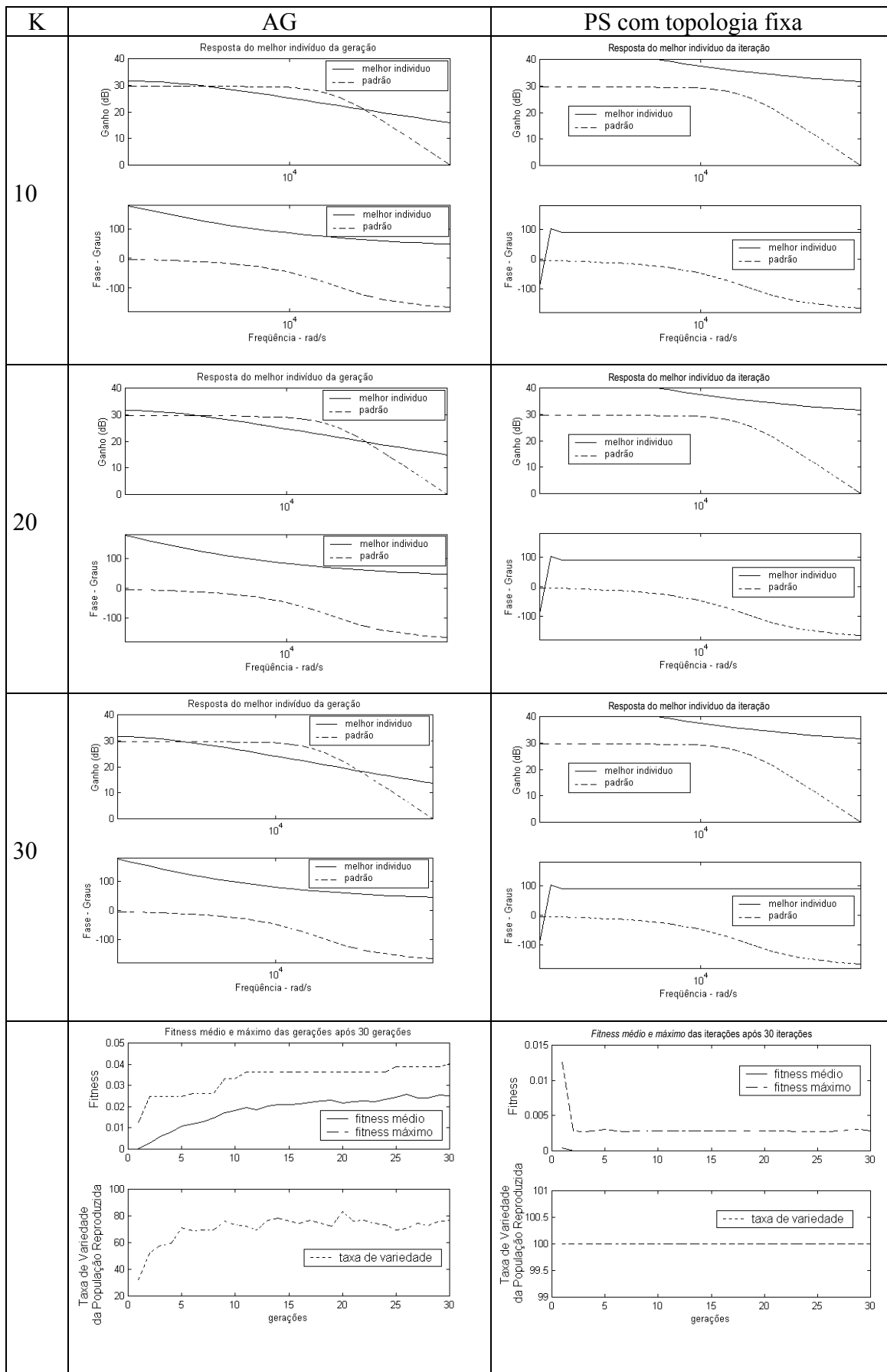


Figura 4.2.a:  $A_{v\max} = +30$  dB - gráficos dos testes com 30 gerações evoluindo continuamente.

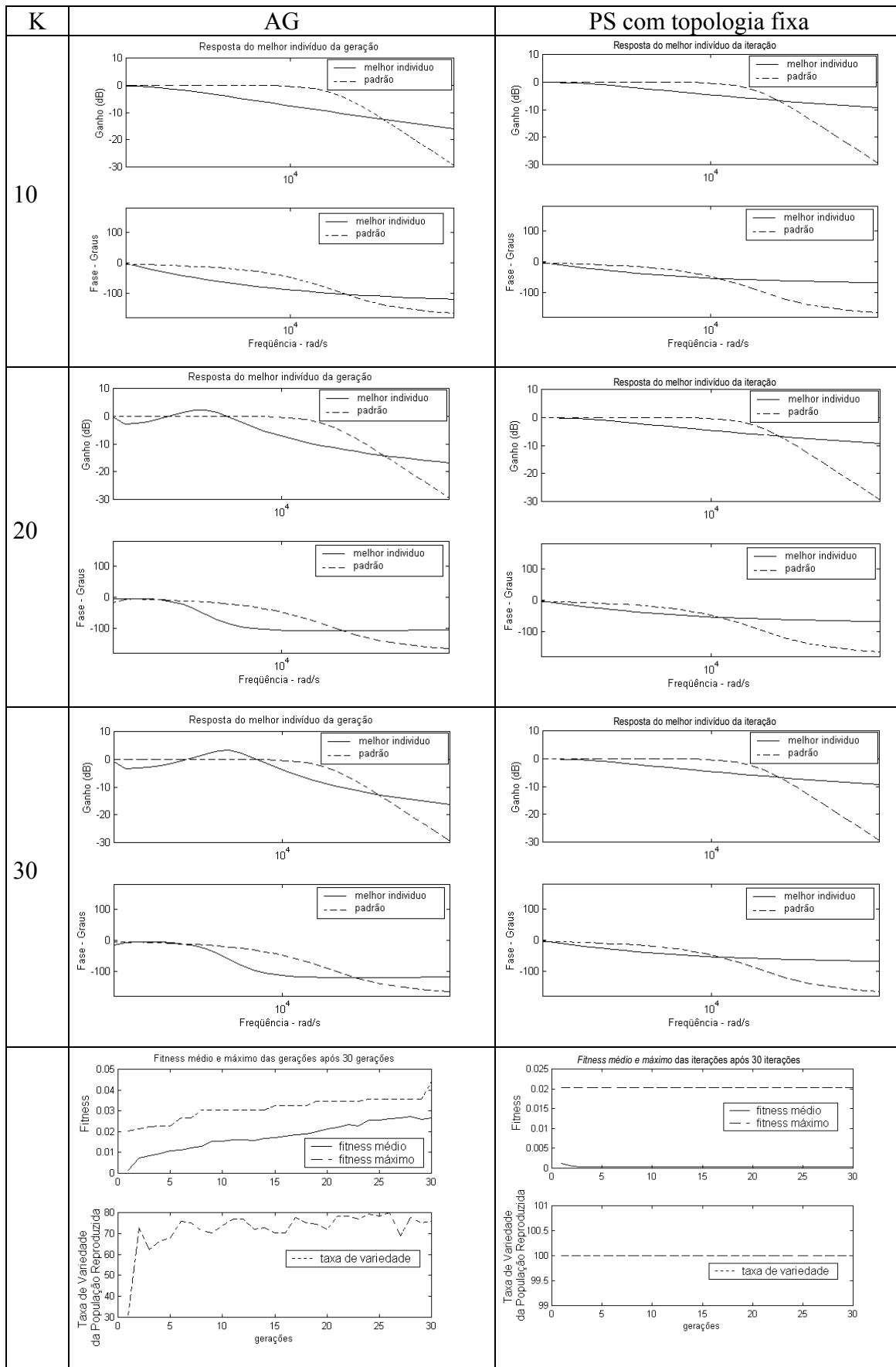


Figura 4.2.b:  $A_{v\max} = 0$  dB - gráficos dos testes com 30 gerações evoluindo continuamente.

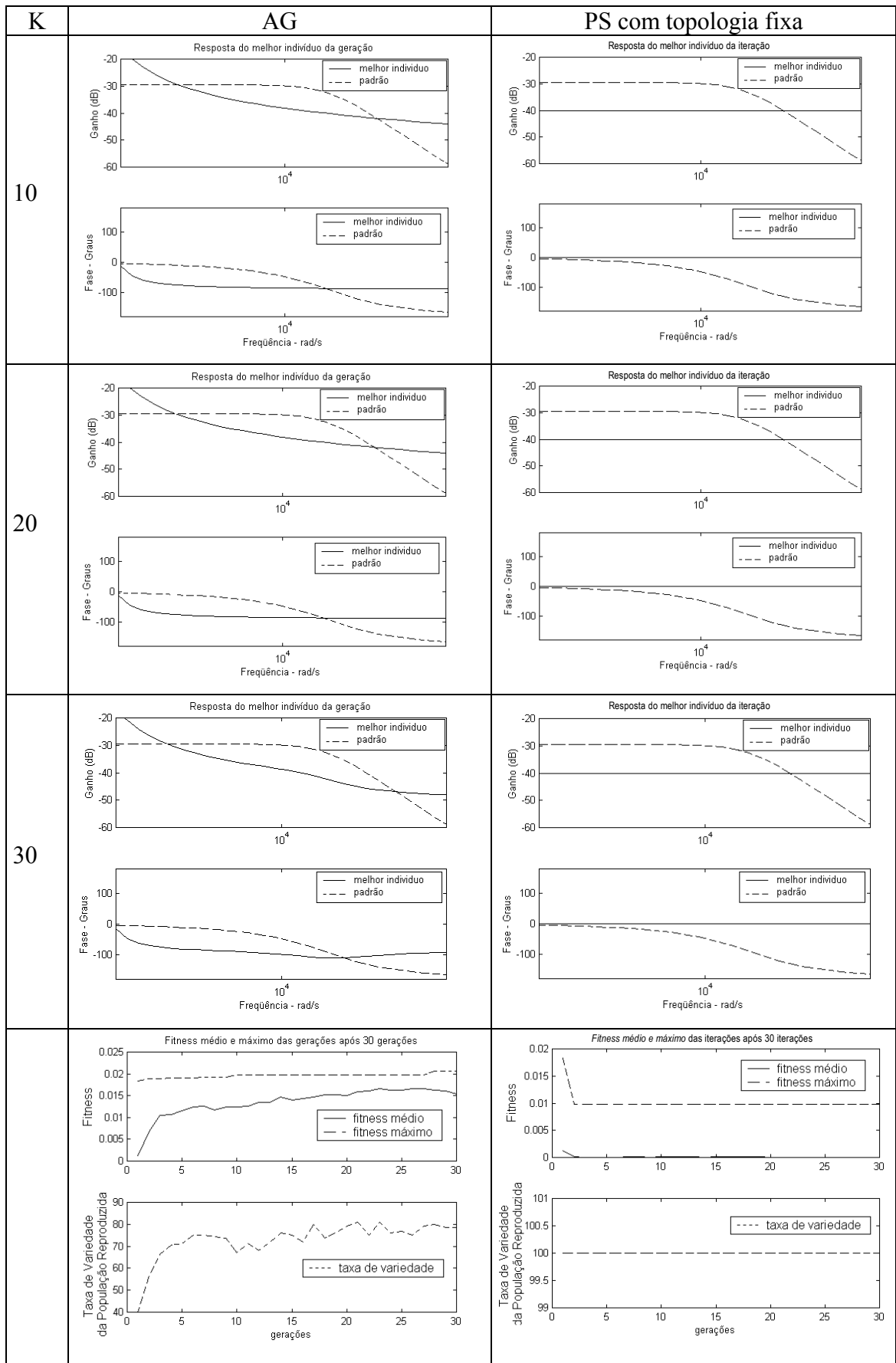


Figura 4.2.c:  $A_{v\max} = -30$  dB - gráficos dos testes com 30 gerações evoluindo continuamente.



Pode ser observado dos resultados anteriores que o AG possui características evolutivas mais adequadas que o PS, que, no entanto, apresenta uma convergência mais rápida que o AG, além de possuir um tempo entre iterações menor que o do AG.

Na figura 4.2.a, o gráfico de aptidão do PS apresenta pequenas oscilações em torno de um valor médio de *fitness*. Podemos interpretar este fato como uma característica do não elitismo do PS, que pode apresentar pequenas oscilações na posição da partícula mais apta, devido à influência dos vizinhos desta sobre a mesma. Como podem existir muitos vizinhos com aptidão baixa, eles podem atrair a melhor partícula que ficaria oscilando entre mínimos locais.

#### 4.1.2. Topologias fixas e Valores de Componentes Variáveis por 60 gerações

Posteriormente foram realizadas simulações com 60 gerações, também continuamente, para uma população inicial de 250 indivíduos com 15 genes cada um, para cada uma das três funções-objetivo. A tabela 4.2 e as Figuras 4.3.a, 4.3.b e 4.3.c mostram os resultados alcançados. O índice K significa Geração/Iteração.

Tabela 4.2: Resumo dos testes com 60 gerações evoluindo continuamente.

<i>Métodos empregados e Topologia</i>	<i>Ganho máx. (dB)</i>	<i>fitness máximo</i>	<i>Duração loop (s)</i>	<i>Método Evoluiu, ficou estável ou regrediu?</i>	<i>Converge em quantas gerações?</i>
AG	+30	0,049	184	evoluiu	5
	0	0,046	197	evoluiu	3
	-30	0,021	384	evoluiu	5
PS com topologia fixa	+30	0,013	182	alternado	2
	0	0,020	195	ficou estável	-
	-30	0,018	380	regrediu	2

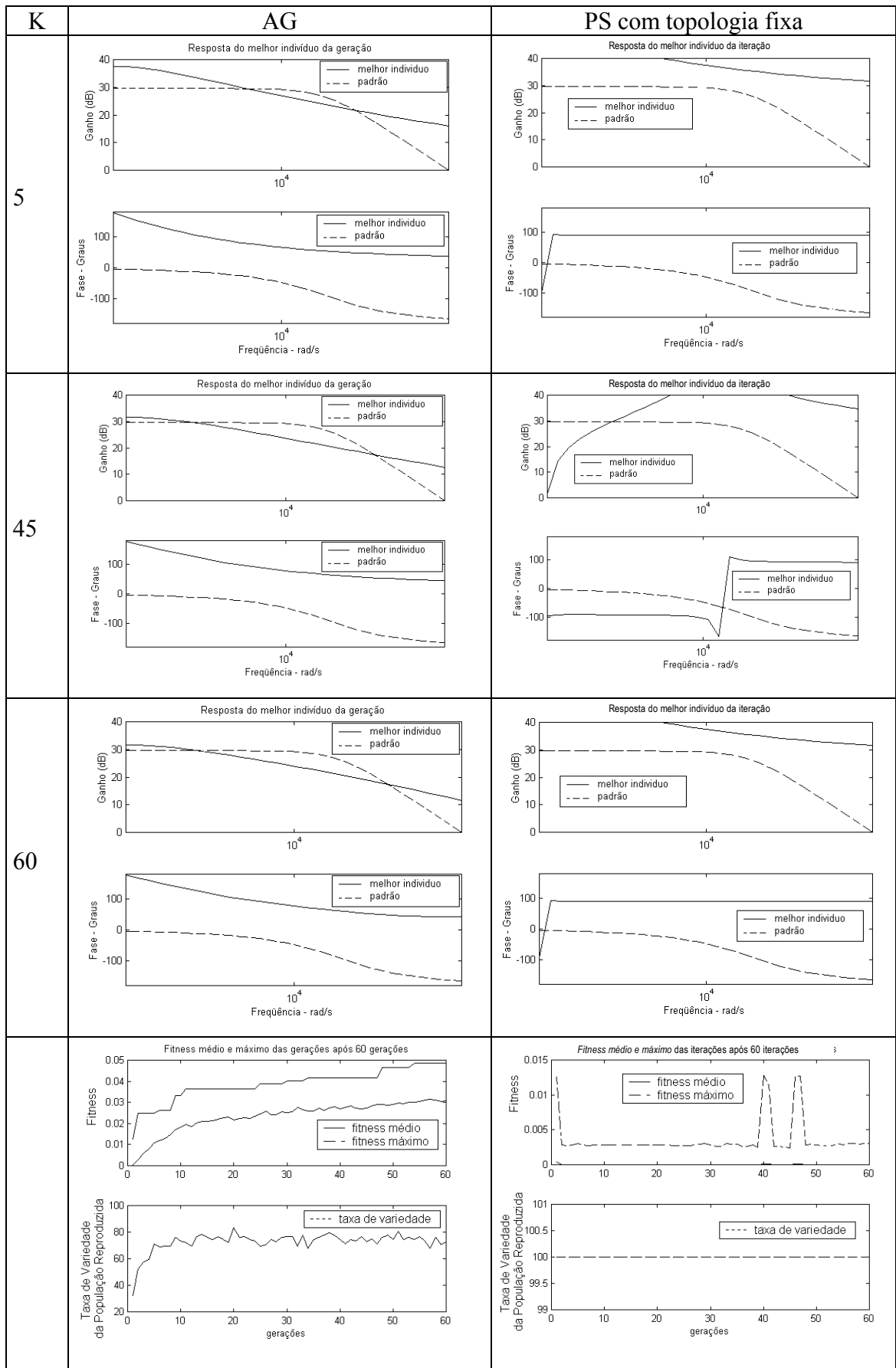


Figura 4.3.a:  $A_{v\max} = +30$  dB - gráficos dos testes com 60 gerações evoluindo continuamente.

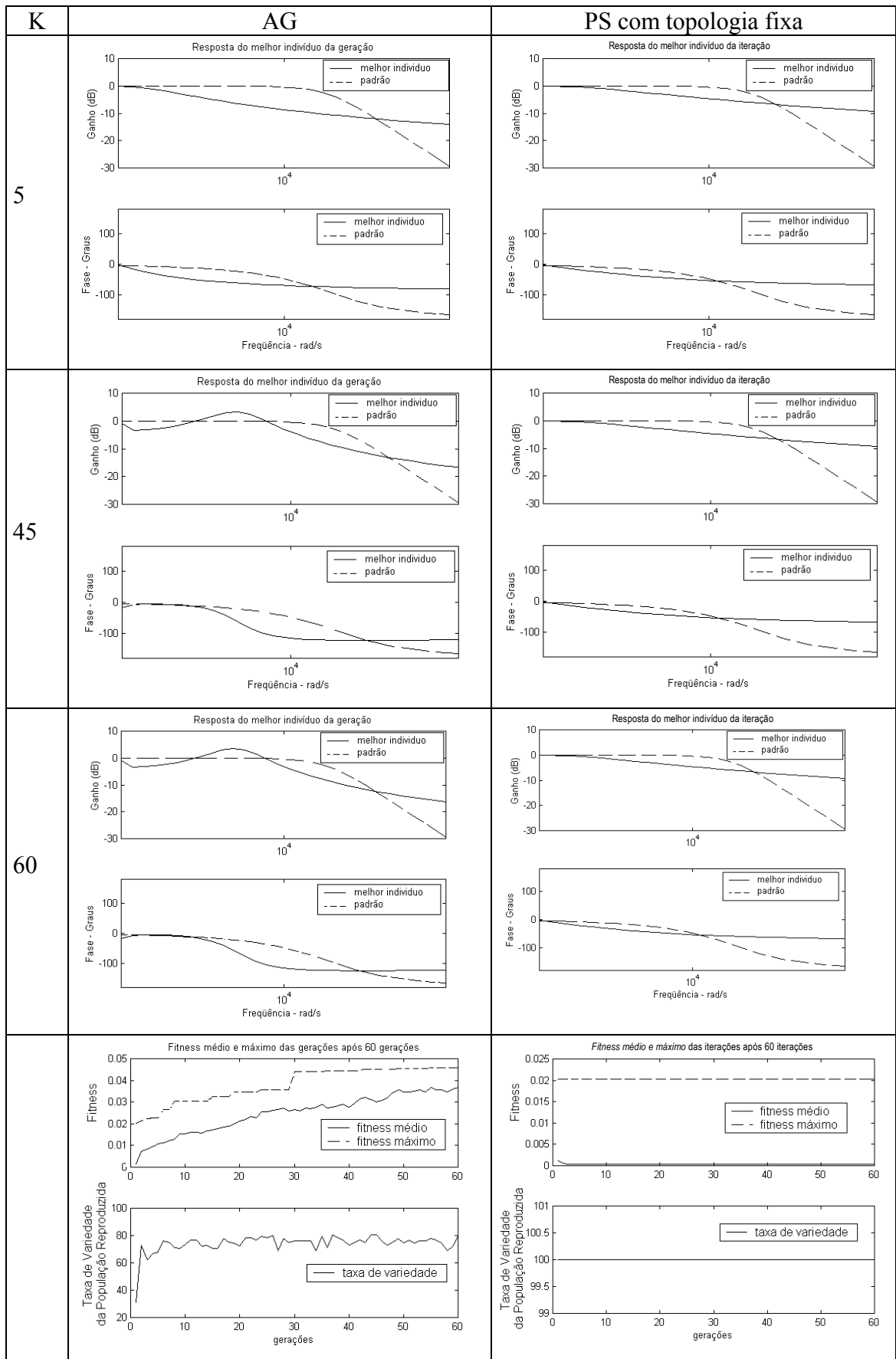


Figura 4.3.b:  $A_{v\max} = 0$  dB - gráficos dos testes com 60 gerações evoluindo continuamente.

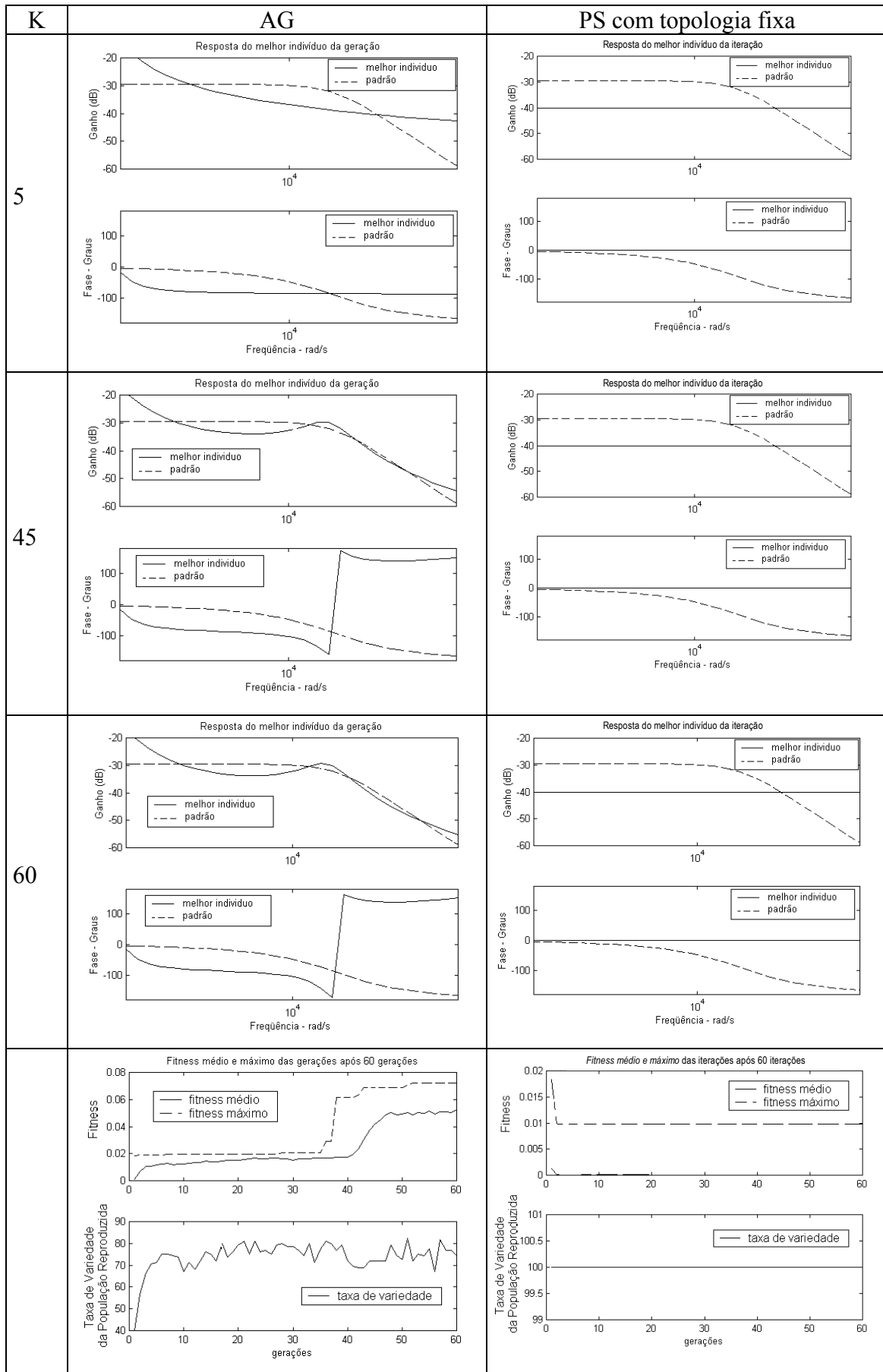


Figura 4.3.c:  $A_{v\max} = -30$  dB - gráficos dos testes com 60 gerações evoluindo continuamente.

Novamente, pode ser observado dos resultados apresentados que o AG possui características evolutivas melhores que o PS, sendo que este último apresenta uma convergência mais rápida que o AG, além de possuir um tempo entre iterações menor que o do AG.

Na figura 4.3.b, o gráfico de aptidão do PS apresenta comportamento constante em torno de um valor médio de *fitness*, depois de apresentar uma queda após a terceira geração. Este fato sugere que podem existir muitos vizinhos com aptidão baixa, atraindo a melhor partícula para um mínimo local.

#### 4.1.3. Topologias fixas e Valores de Componentes Variáveis por mais 30 gerações depois de 30 gerações iniciais

Finalmente foram realizadas novas simulações, tendo em vista os resultados encontrados para as técnicas sócio-cognitivas, só que agora entregando como a última população simulada para o *Particle Swarm* a população já evoluída pelo Algoritmo Genético após 30 gerações, e que foi analisada na Tabela 4.2. Esta população também possui 250 indivíduos com 15 genes cada um. A tabela 4.3 e as Figuras 4.4.a, 4.4.b e 4.4.c mostram os resultados alcançados. O índice K significa Geração/Iteração.

Tabela 4.3: Resumo dos testes com mais 30 gerações depois de 30 iniciais.

<i>Métodos empregados e Topologia</i>	<i>Ganho máx. (dB)</i>	<i>fitness máximo</i>	<i>Duração loop (s)</i>	<i>Método Evoluiu, ficou estável ou regrediu?</i>	<i>Converge em quantas gerações?</i>
AG	+30	0,05	190	evoluiu	10
	0	0,045	180	evoluiu	5
	-30	0,028	357	evoluiu	8
PS com topologia fixa	+30	0,04 máx 0,02 fim	187	evoluiu e regrediu	2
	0	0,05 máx 0,04 méd 0,03 mín	178	evoluiu e regrediu oscilando	oscila a uma frequência média de 2 gerações
	-30	0,018	349	regrediu	2

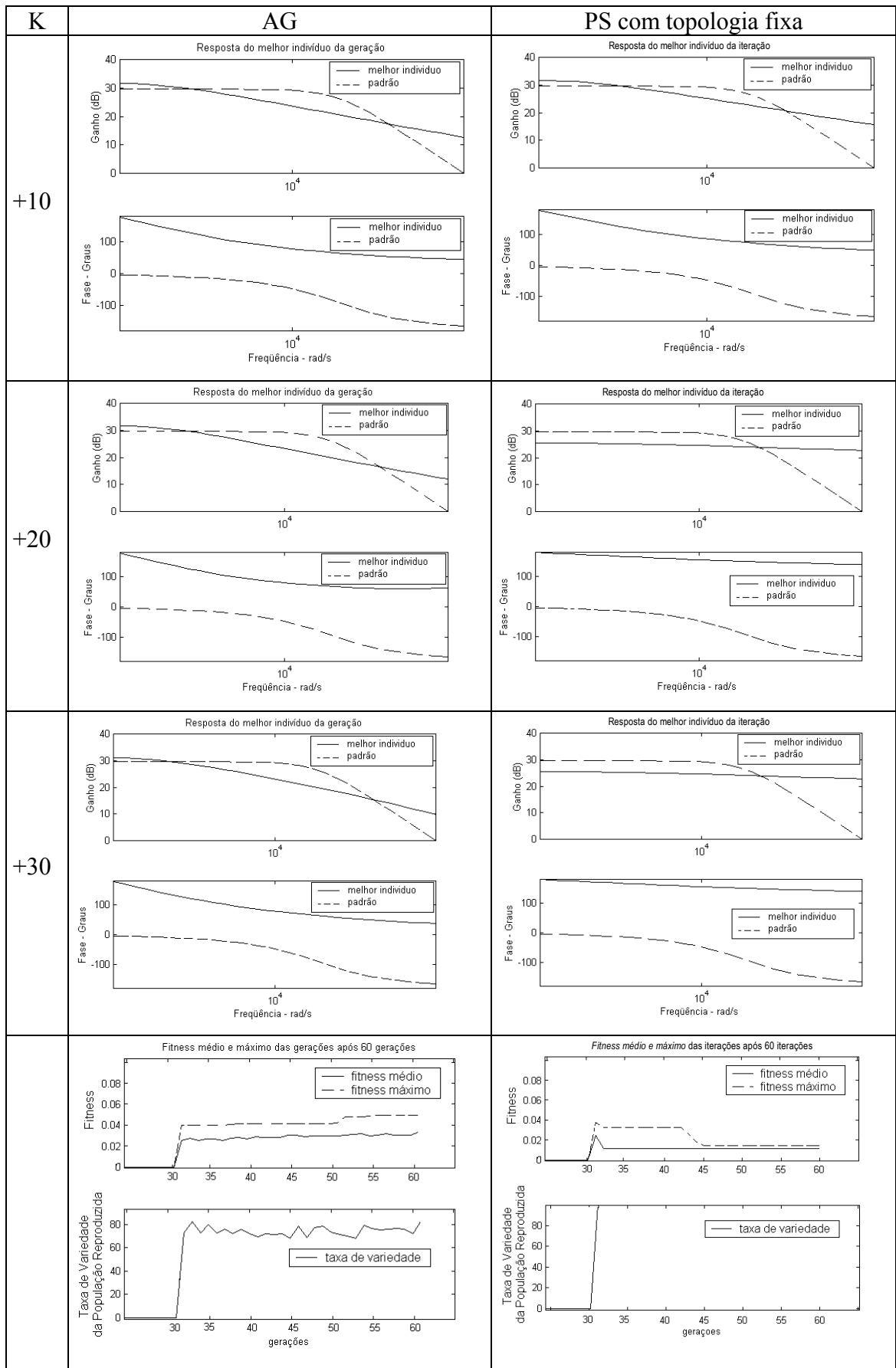


Figura 4.4.a: Avmáx = +30 dB - gráficos dos testes com mais 30 gerações depois de 30 iniciais.

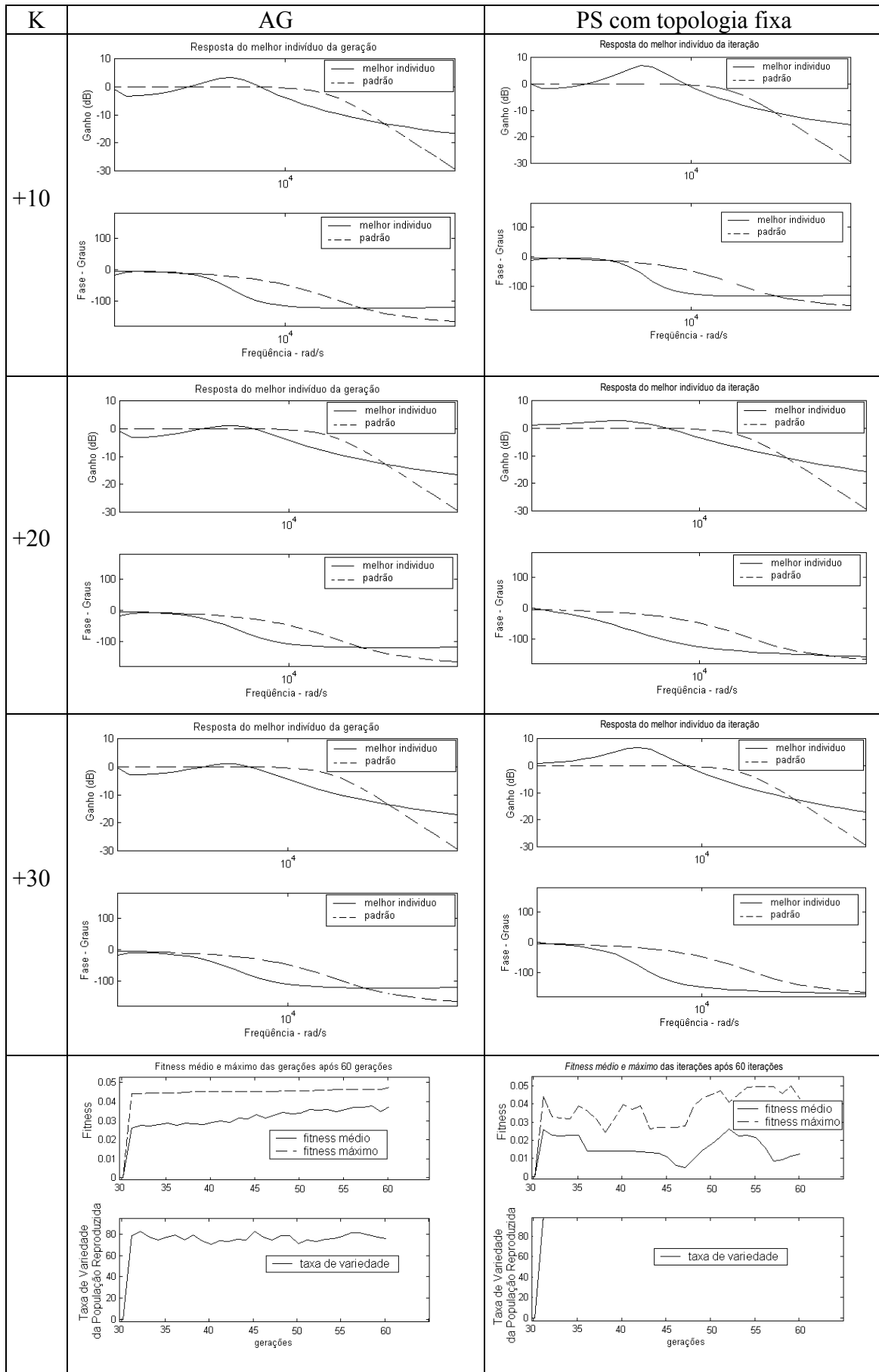


Figura 4.4.b:  $A_{v\max} = 0$  dB - gráficos dos testes com mais 30 gerações depois de 30 iniciais.

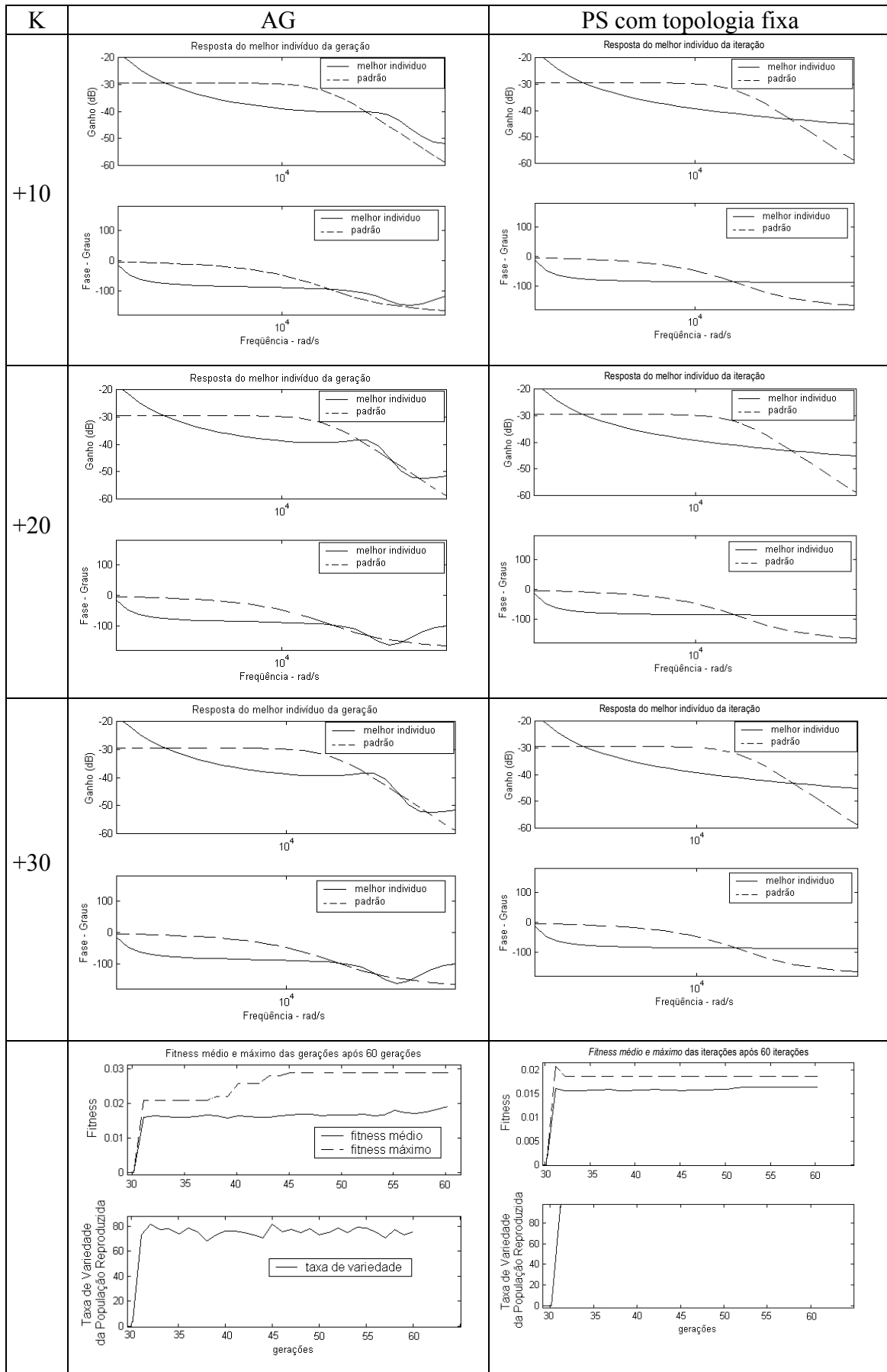


Figura 4.4.c:  $A_{v\max} = -30$  dB - gráficos dos testes com mais 30 gerações depois de 30 iniciais.



Novamente, pode ser observado dos resultados ora apresentados que o AG possui características evolutivas melhores que o PS, sendo que este último apresenta uma convergência mais rápida que o AG, além de possuir um tempo entre interações menor que o do AG.

Na figura 4.4.b, o gráfico de aptidão do PS apresenta comportamento oscilatório em torno de um valor médio de *fitness*, durante todo o tempo de simulação. Provavelmente existem vários mínimos locais e as partículas são atraídas por elas. Quando um grupo de partículas é atraído para alguns desses mínimos, o restante pode ser “arrastado” junto para esta região do “hiperespaço”.

## 4.2. Sallen-Key passa-baixas de 2a. ordem

Um diagrama de Bode para este filtro está ilustrado na figura 4.5.

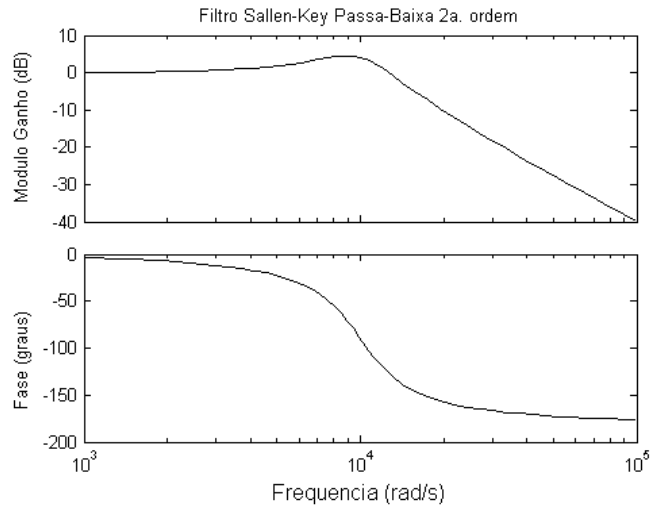


Figura 4.5: Diagrama de Bode do filtro *Sallen-Key* passa-baixas de 2a. ordem.

### 4.2.1. Topologias fixas e Valores de Componentes Variáveis por 30 gerações

De modo idêntico ao que foi feito para o filtro anterior, a população inicial foi evoluída por 30 gerações continuamente, para cada um dos ganhos máximos pré-estabelecidos para a função-objetivo em questão. A tabela 4.4 e as Figuras 4.6.a, 4.6.b e 4.6.c mostram os resultados alcançados.

Tabela 4.4: Resumo dos testes com 30 gerações evoluindo continuamente.

<i>Métodos empregados e Topologia</i>	<i>Ganho máx. (dB)</i>	<i>fitness máximo</i>	<i>Duração loop (s)</i>	<i>Método Evoluiu, ficou estável ou regrediu?</i>	<i>Converge em quantas gerações?</i>
AG	+45	0,01	475	evoluiu	8
	+5	0,017	520	evoluiu	5
	-30	0,014	558	evoluiu	5
PS com topologia fixa	+45	0,001	466	oscilou	2
	+5	0,008	518	regrediu	1
	-30	0,006	557	oscilou	2

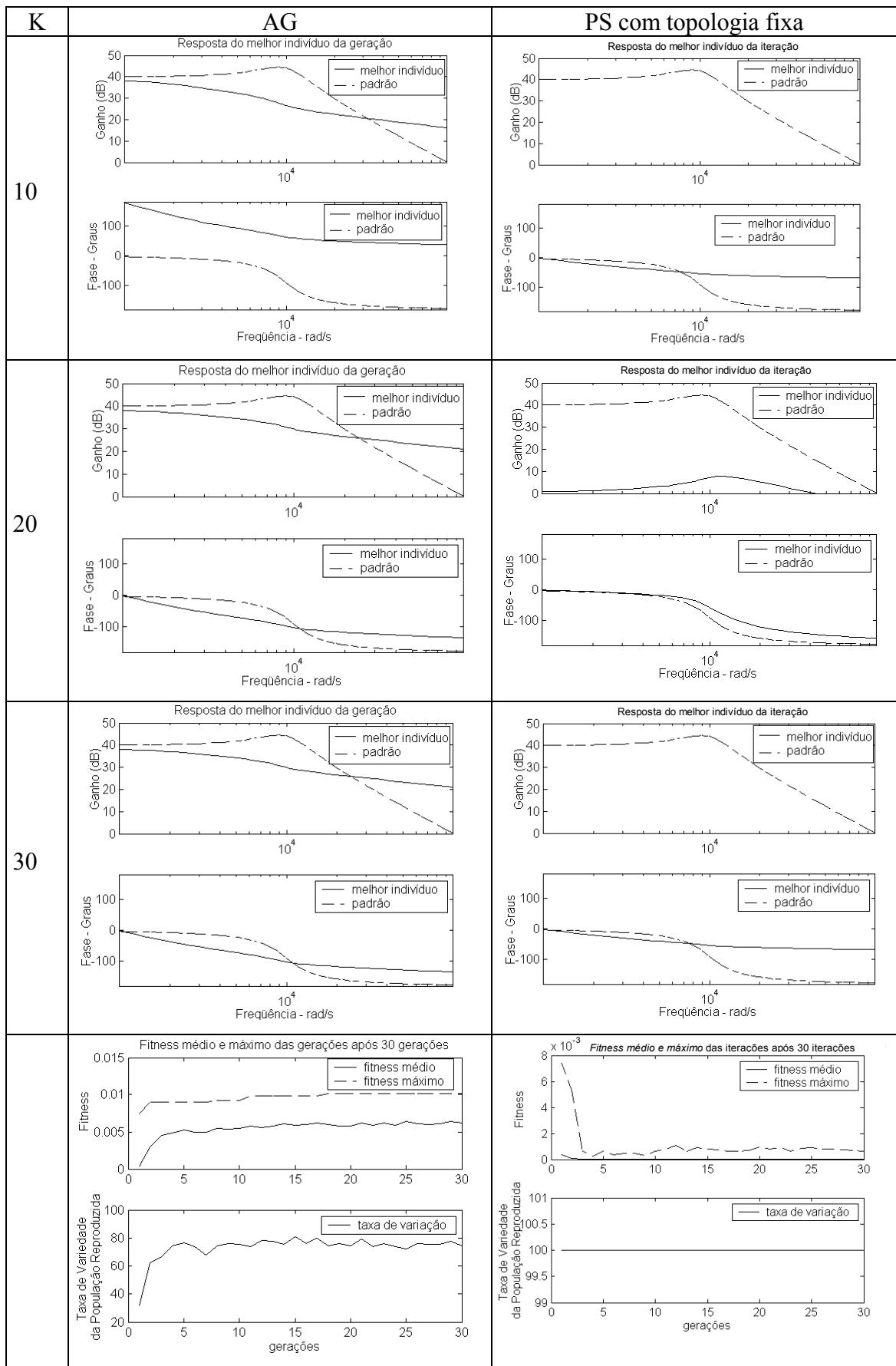


Figura 4.6.a:  $A_{v\max} = +45$  dB - gráficos dos testes com 30 gerações evoluindo continuamente.

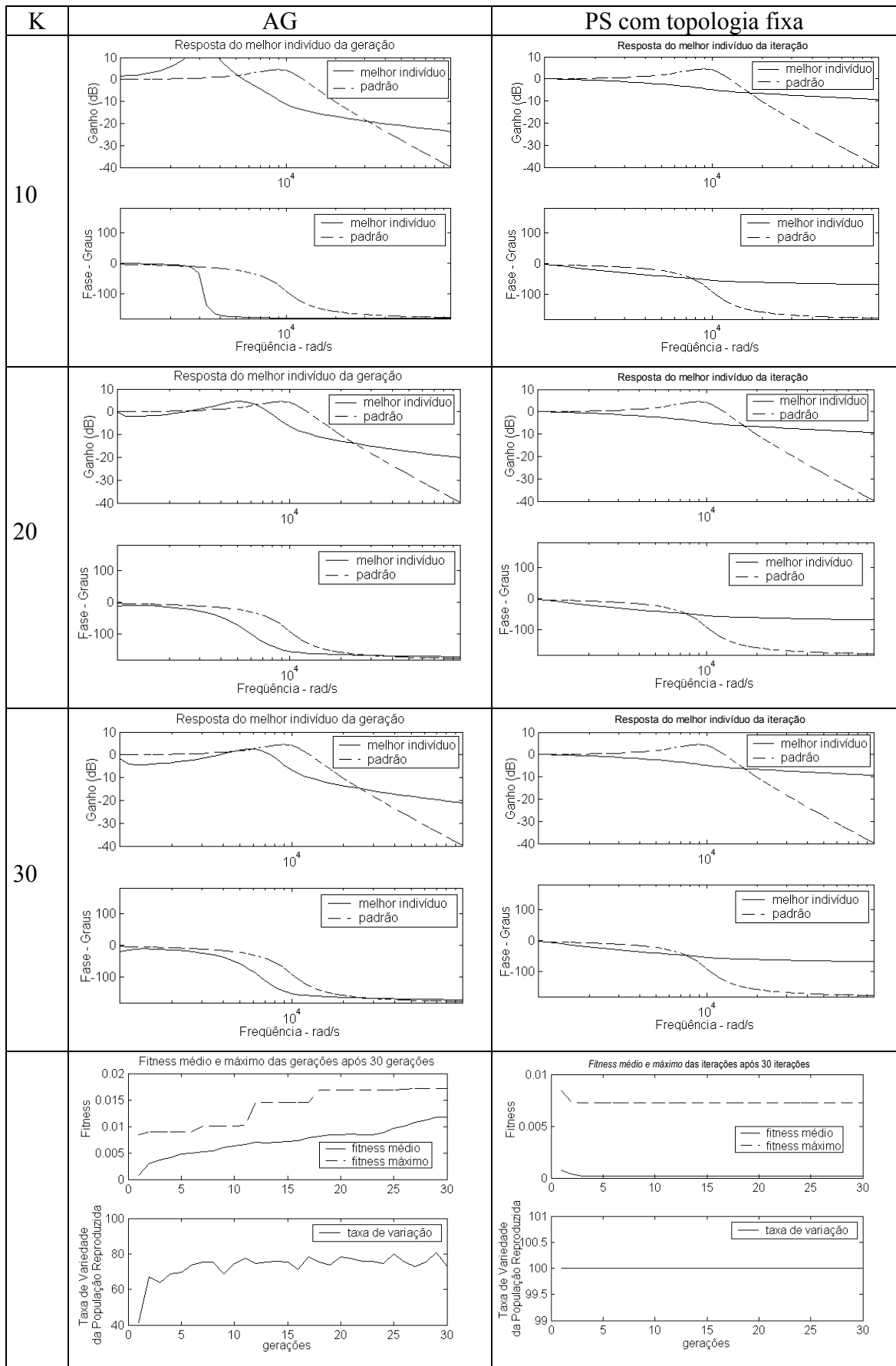


Figura 4.6.b:  $A_{v\max} = +5$  dB - gráficos dos testes com 30 gerações evoluindo continuamente.

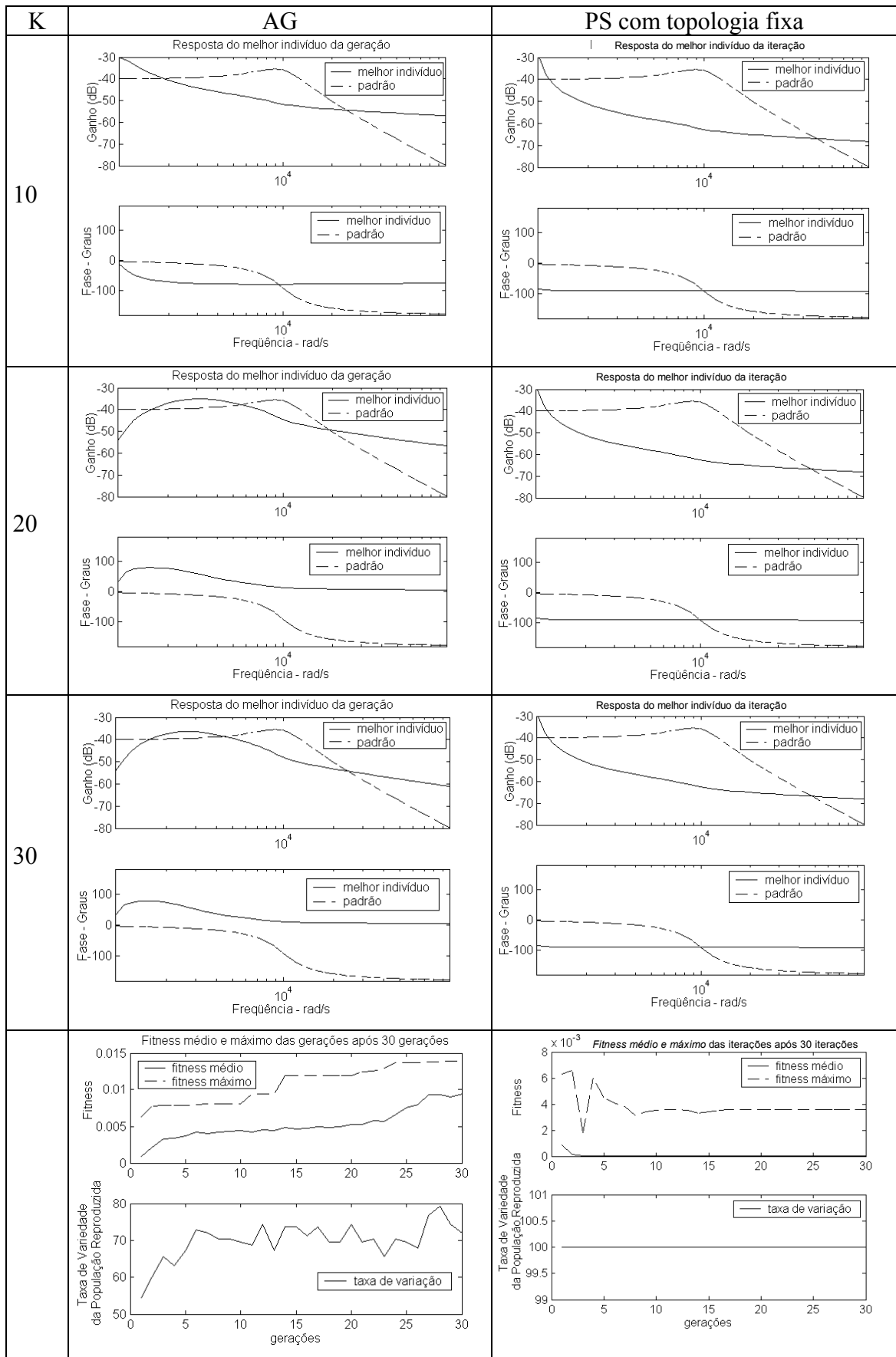


Figura 4.6.c:  $A_{v\max} = -30$  dB - gráficos dos testes com 30 gerações evoluindo continuamente.

Pode ser observado dos resultados anteriores que, mais uma vez, o AG possui características evolutivas mais adequadas que o PS, que, no entanto, apresenta uma convergência mais rápida que o AG, além de possuir um tempo entre iterações menor que o do AG.

Os resultados da figura 4.6.b apresentam um fato importante com relação aos circuitos evoluídos através de AG: as curvas de ganho e de fase bem próximas das curvas de um circuito padrão.

Nas figura 4.6.a e 4.6.c, os gráficos de aptidão do PS apresentam pequenas oscilações em torno de um valor médio de *fitness*. Podemos interpretar novamente isto como uma característica do não elitismo do PS, que pode apresentar pequenas oscilações na posição da partícula mais apta, devido à influência dos seus vizinhos. Como podem existir muitos vizinhos com aptidão baixa, eles podem atrair a melhor partícula, que ficaria oscilando entre mínimos locais.

#### 4.2.2. Topologias fixas e Valores de Componentes Variáveis por 60 gerações

Posteriormente foram realizadas simulações com 60 gerações, também continuamente, para uma população inicial de 250 indivíduos com 15 genes cada um, para cada uma das três funções-objetivo. A tabela 4.5 e as Figuras 4.7.a, 4.7.b e 4.7.c mostram os resultados alcançados. O índice K significa Geração/Iteração.

**Tabela 4.5: Resumo dos testes com 60 gerações evoluindo continuamente.**

<i>Métodos empregados e Topologia</i>	<i>Ganho máx. (dB)</i>	<i>fitness máximo</i>	<i>Duração loop (s)</i>	<i>Método Evoluiu, ficou estável ou regrediu?</i>	<i>Converge em quantas gerações?</i>
AG	+45	0,013	475	evoluiu	8
	+5	0,01	520	evoluiu	5
	-30	0,013	558	evoluiu	5
PS com topologia fixa	+45	0,001	466	oscilou	2
	+5	0,007	518	regrediu	1
	-30	0,006	557	oscilou	2

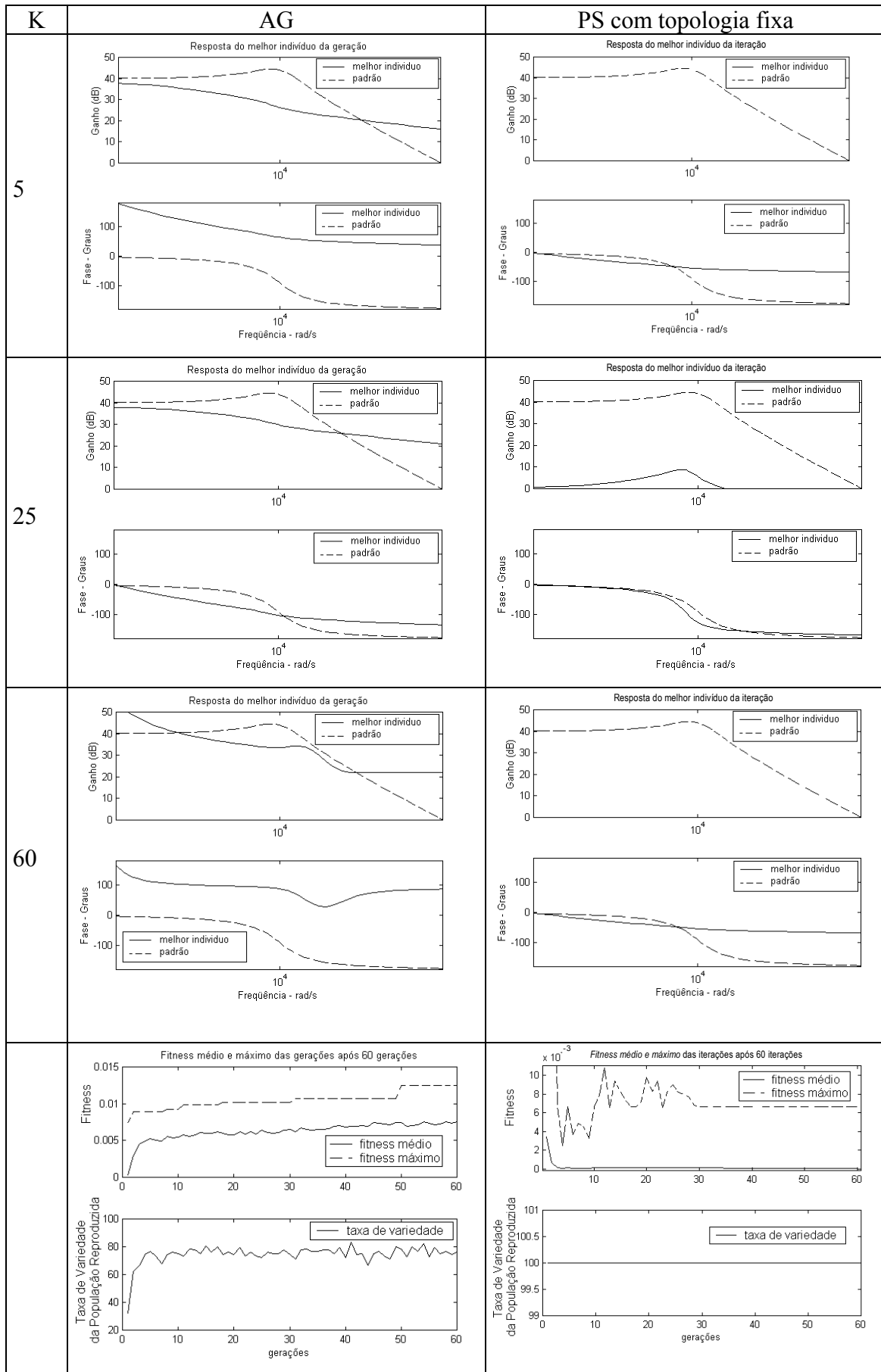


Figura 4.7.a:  $A_{v\max} = +45$  dB - gráficos dos testes com 60 gerações evoluindo continuamente.

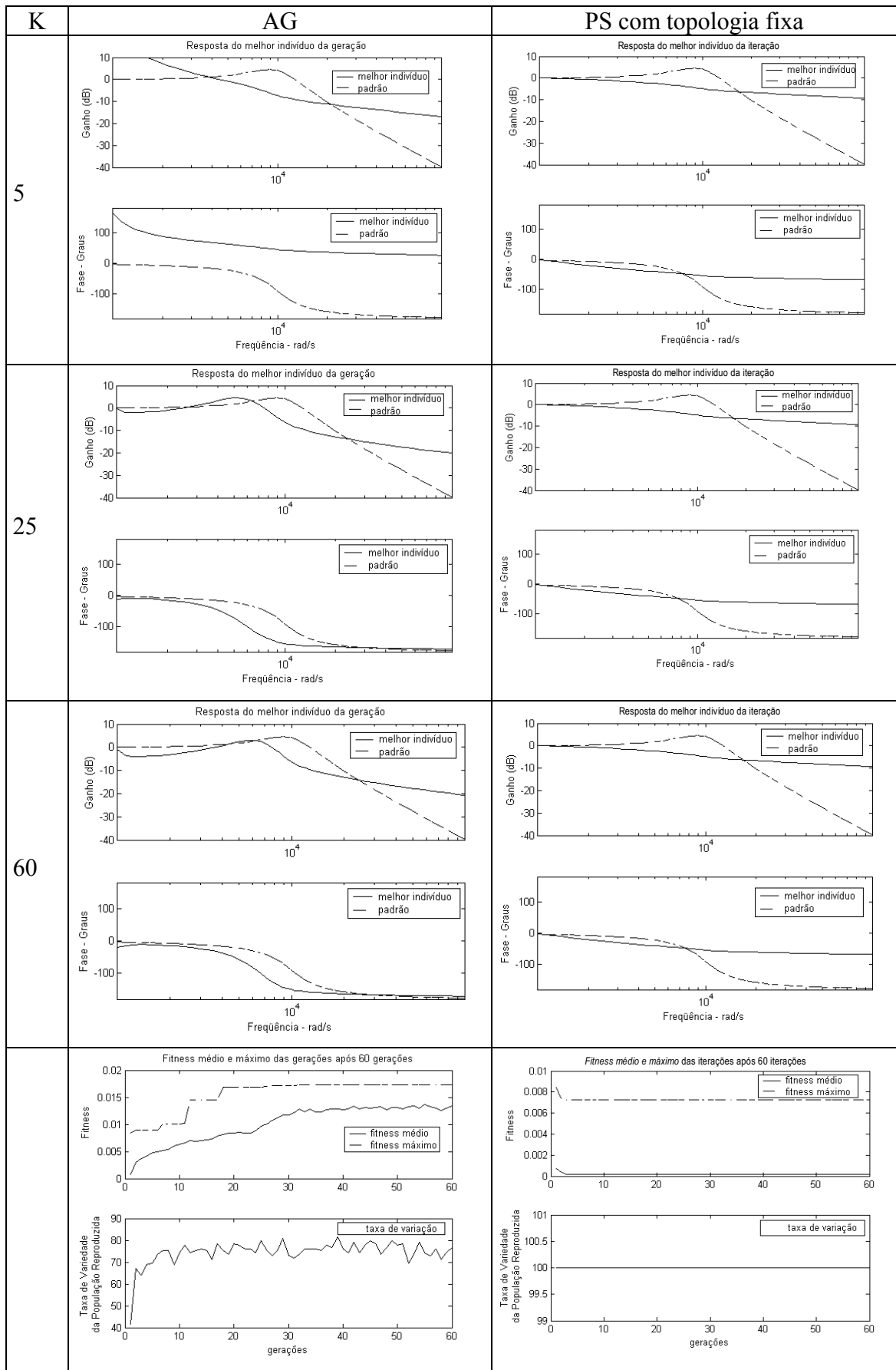


Figura 4.7.b:  $A_{v\max} = +5$  dB - gráficos dos testes com 60 gerações evoluindo continuamente.



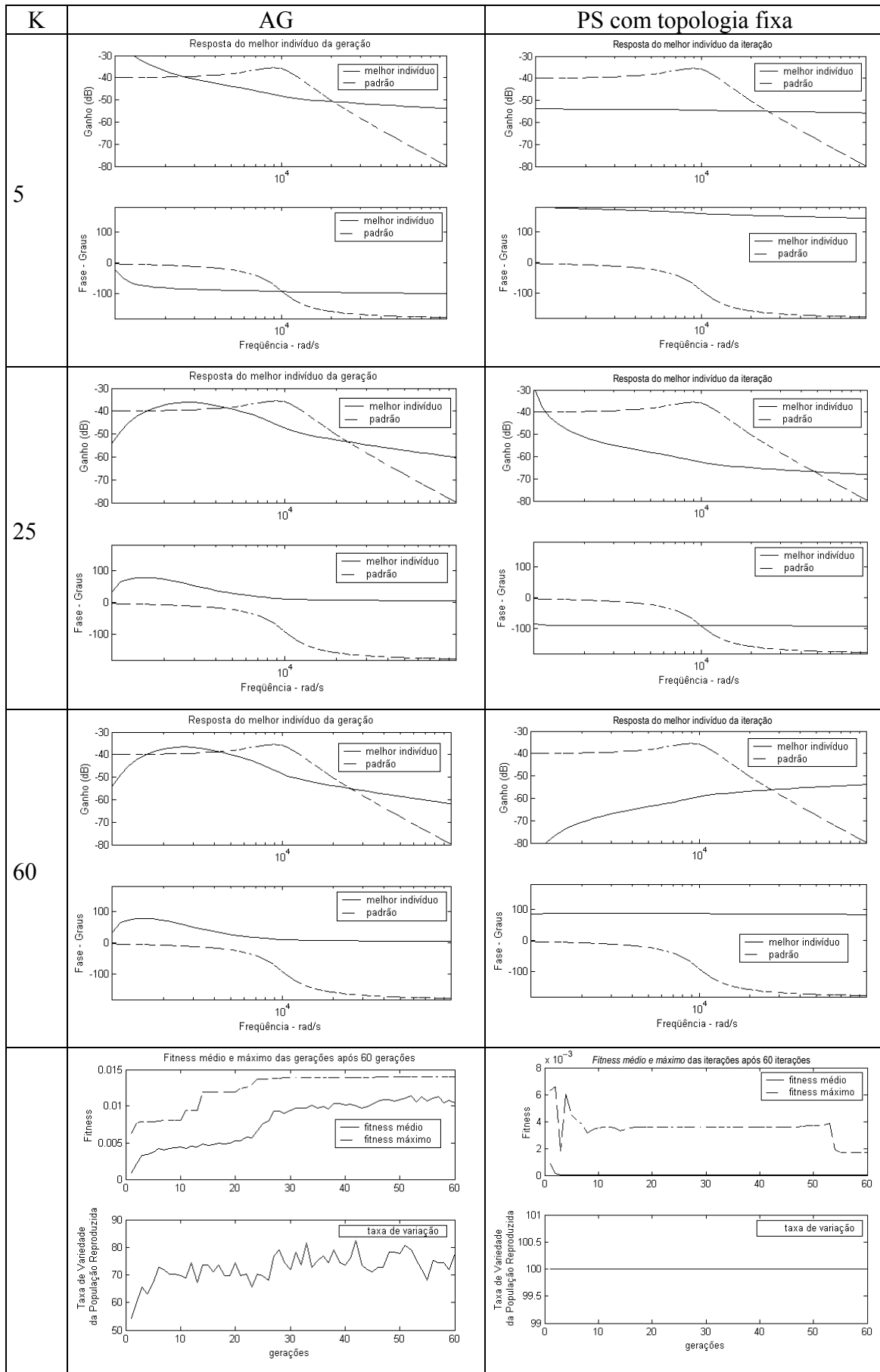


Figura 4.7.c:  $A_{v\max} = -30$  dB - gráficos dos testes com 60 gerações evoluindo continuamente.

Novamente, pode ser observado dos resultados obtidos, que o AG possui características evolutivas melhores que o PS, sendo que este último apresenta uma convergência mais rápida que o AG, além de possuir um tempo entre iterações menor que o do AG.

Na figura 4.7.b, o gráfico de aptidão do PS apresenta comportamento constante em torno de um valor médio de *fitness*, depois de apresentar uma queda após a terceira geração. Este fato sugere que podem existir muitos vizinhos com aptidão baixa, atraindo a melhor partícula para um mínimo local.

#### 4.2.3. Topologias fixas e Valores de Componentes Variáveis por mais 30 gerações depois de 30 gerações iniciais

Finalmente foram realizadas novas simulações, tendo em vista os resultados encontrados para as técnicas sócio-cognitivas, só que agora entregando como a última população simulada para o *Particle Swarm* a população já evoluída pelo Algoritmo Genético após 30 gerações, e que foi analisada na Tabela 4.2. Esta população também possui 250 indivíduos com 15 genes cada um. A tabela 4.6 e as Figuras 4.8.a, 4.8.b e 4.8.c mostram os resultados alcançados. O índice K significa Geração/Iteração.

Tabela 4.6: Resumo dos testes com mais 30 gerações depois de 30 iniciais.

<i>Métodos empregados e Topologia</i>	<i>Ganho máx. (dB)</i>	<i>fitness máximo</i>	<i>Duração loop (s)</i>	<i>Método Evoluiu, ficou estável ou regrediu?</i>	<i>Converge em quantas gerações?</i>
AG	+45	0,0013	188	evoluiu	6
	+5	0,016	210	ficou estável	-
	-30	0,023	192	evoluiu	5
PS com topologia fixa	+45	0,0013	186	oscilou	2
	+5	0,016	208	regrediu	1
	-30	0,001	188	oscilou	2

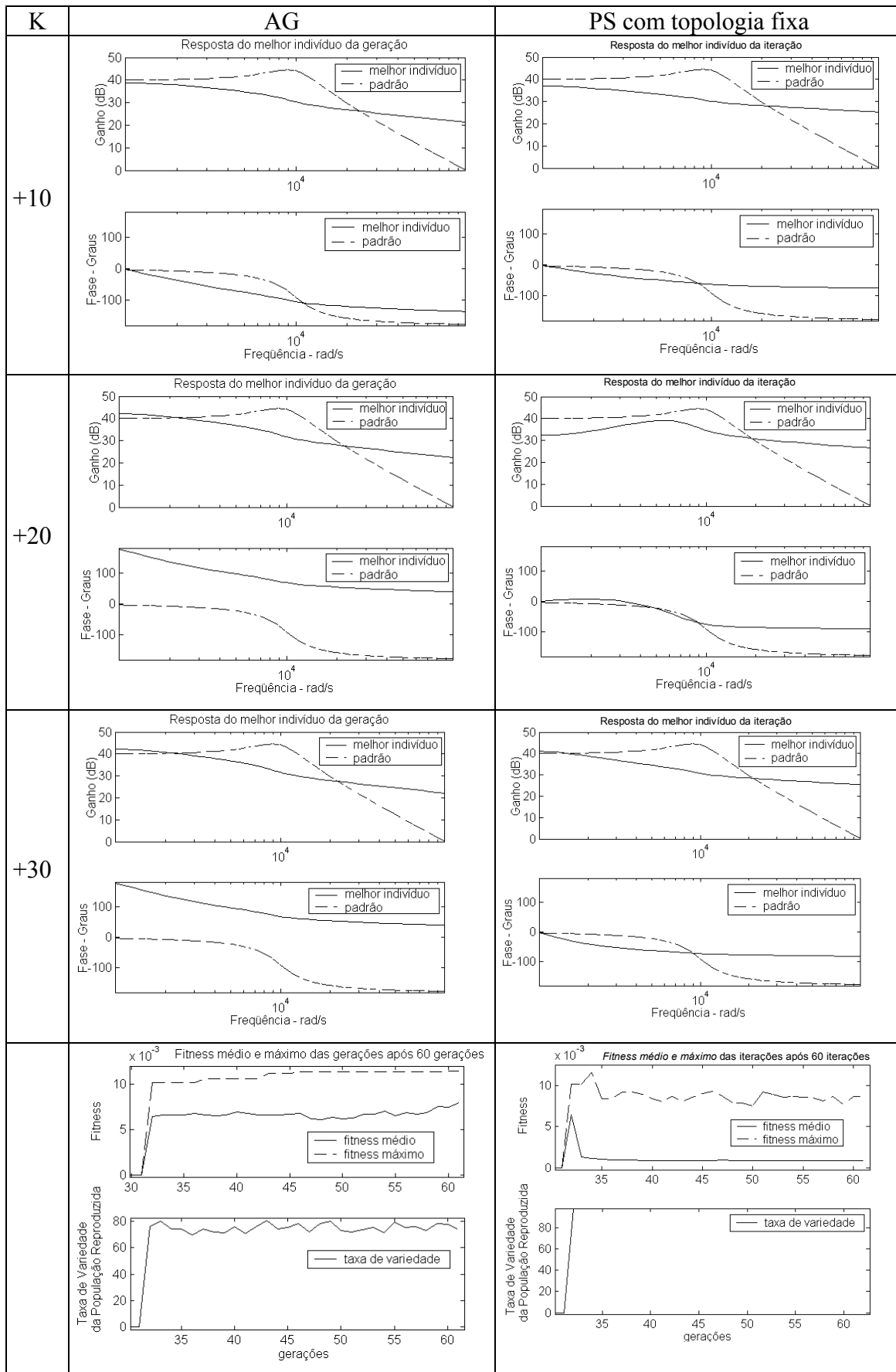


Figura 4.8.a:  $A_{v\max} = +30$  dB - gráficos dos testes com mais 30 gerações depois de 30 iniciais.

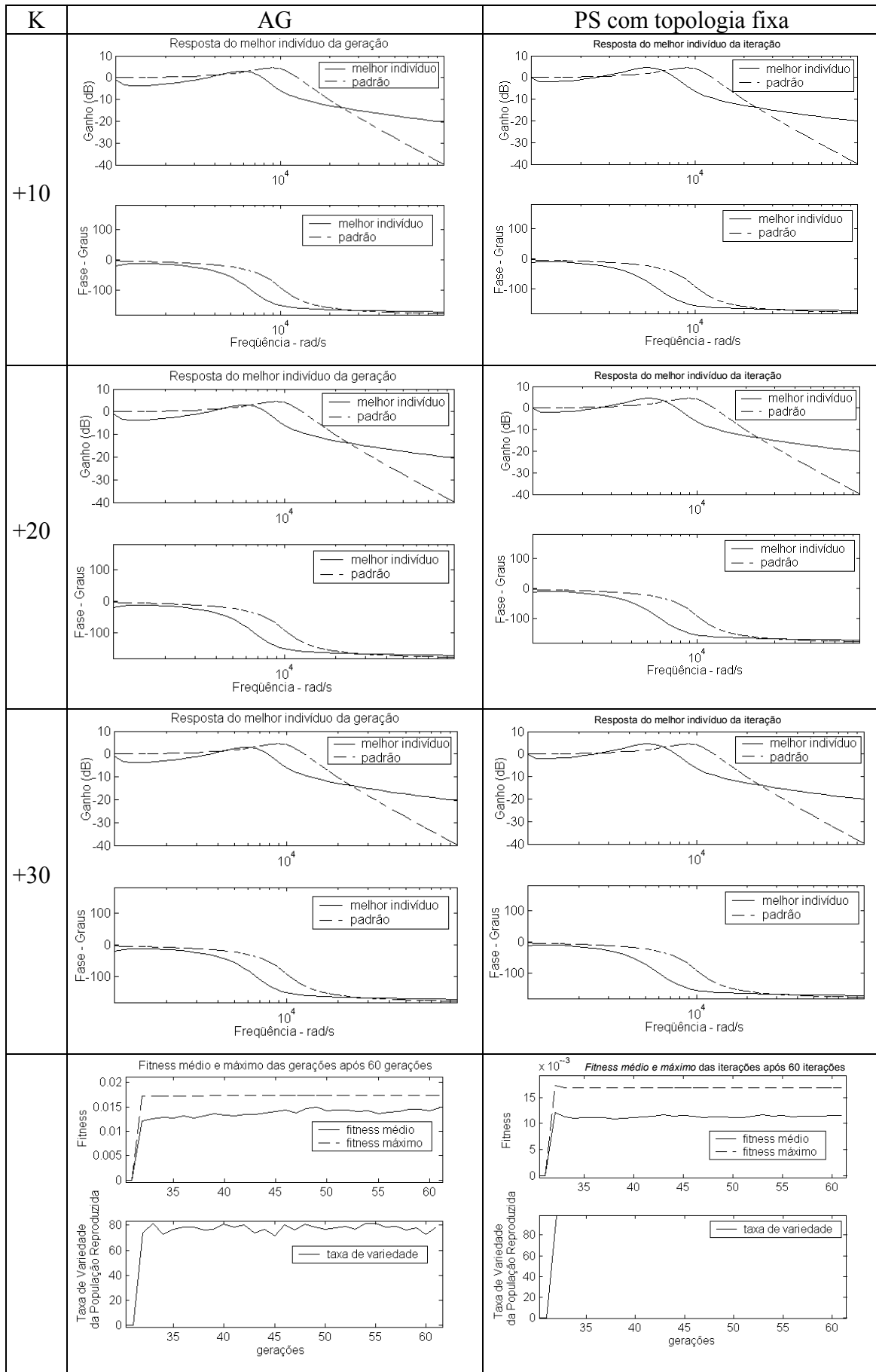


Figura 4.8.b:  $A_{v\max} = 0$  dB - gráficos dos testes com mais 30 gerações depois de 30 iniciais.

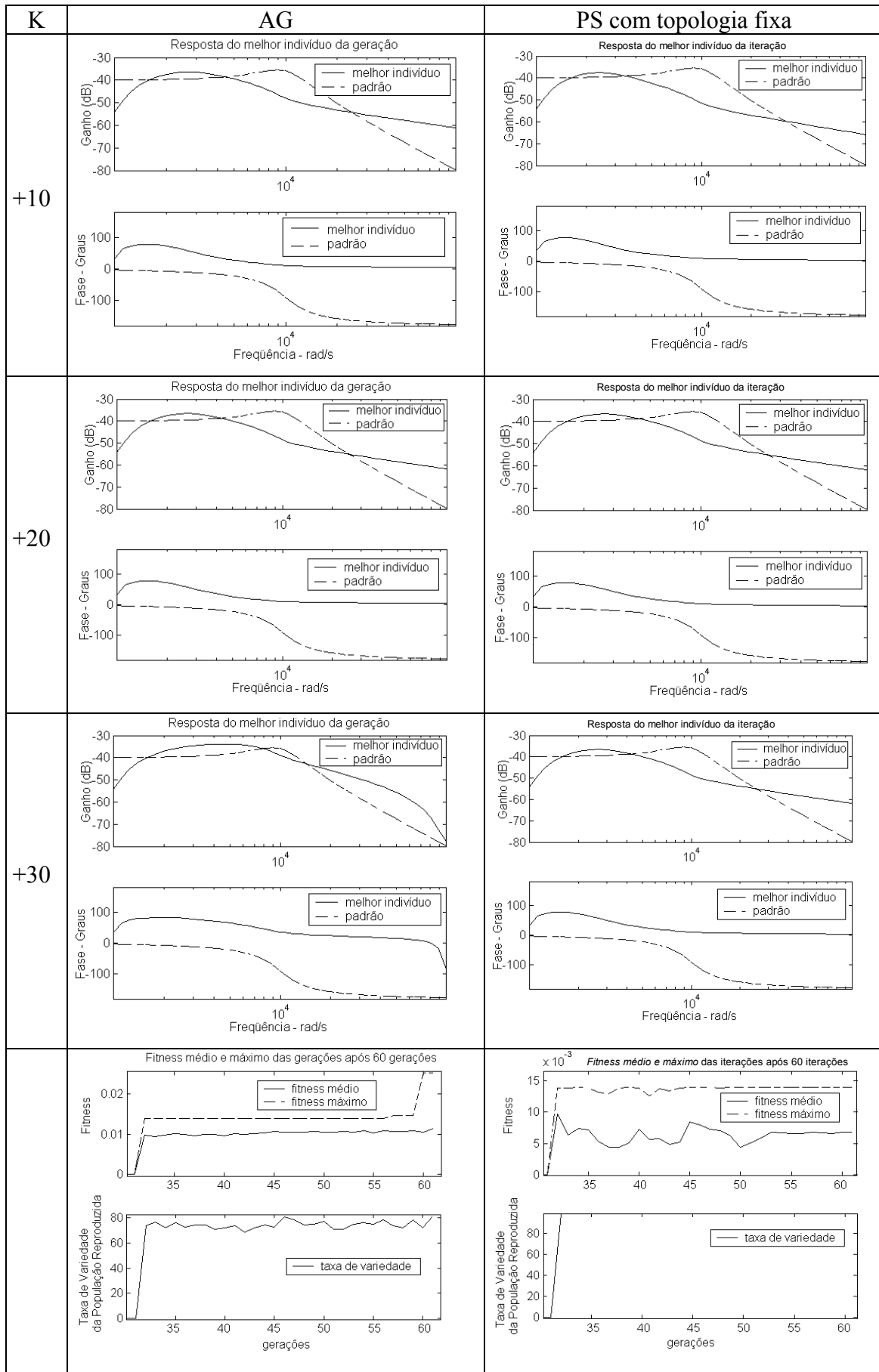


Figura 4.8.c:  $A_{v\max} = -30$  dB - gráficos dos testes com mais 30 gerações depois de 30 iniciais.

Novamente, pode ser observado dos resultados apresentados que o AG possui características evolutivas melhores que o PS, sendo que este último apresenta uma convergência mais rápida que o AG, além de possuir um tempo entre iterações menor que o do AG.

Na figura 4.8.a, o gráfico de aptidão do PS apresenta comportamento oscilatório em torno de um valor médio de *fitness*, durante todo o tempo de simulação. Provavelmente existem vários mínimos locais e as partículas são atraídas pelos mesmos. Quando um grupo de partículas é atraído para alguns desses mínimos, o restante pode ser “arrastado” junto para esta região do “hiperespaço”.

As respostas mais próximas do padrão foram obtidas pelo circuitos gerados através do AG, e estão representados na figura 4.8.b. Tanto em termos do módulo do ganho quanto em termos da fase, as curvas ficaram bastante próximas, o que indica que circuito gerado possui uma função de transferência bem parecida com a do padrão. Podemos observar que o gráfico de evolução do AG ficou quase que estabilizado, indicando que o AG chegou a um ponto de quase saturação.

### 4.3. Sallen-Key passa-faixa de 2a. ordem

Um diagrama de Bode para este filtro está ilustrado na figura 4.9.

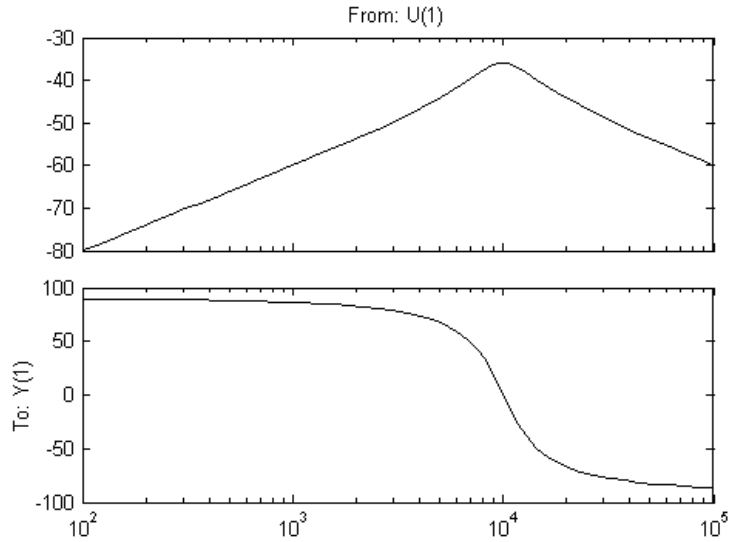


Figura 4.9: Diagrama de Bode do filtro *Sallen-Key* passa-faixa de 2a. ordem.

#### 4.3.1. Topologias fixas e Valores de Componentes Variáveis por 30 gerações

De modo idêntico ao que foi feito para o filtro anterior, a população inicial foi evoluída por 30 gerações continuamente, para cada um dos ganhos máximos pré-estabelecidos para a função-objetivo em questão. A tabela 4.7 e as Figuras 4.10.a, 4.10.b e 4.10.c mostram os resultados alcançados.

Tabela 4.7: Resumo dos testes com 30 gerações evoluindo continuamente.

<i>Métodos empregados e Topologia</i>	<i>Ganho máx. (dB)</i>	<i>fitness máximo</i>	<i>Duração loop (s)</i>	<i>Método Evoluiu, ficou estável ou regrediu?</i>	<i>Converge em quantas gerações?</i>
AG	+45	0,08	475	evoluiu	7
	+5	0,023	520	evoluiu	5
	-30	0,016	203	evoluiu	5
PS com topologia fixa	+45	0,001	466	regrediu	2
	+5	0,007	518	regrediu	1
	-30	0,01	201	oscilou	2

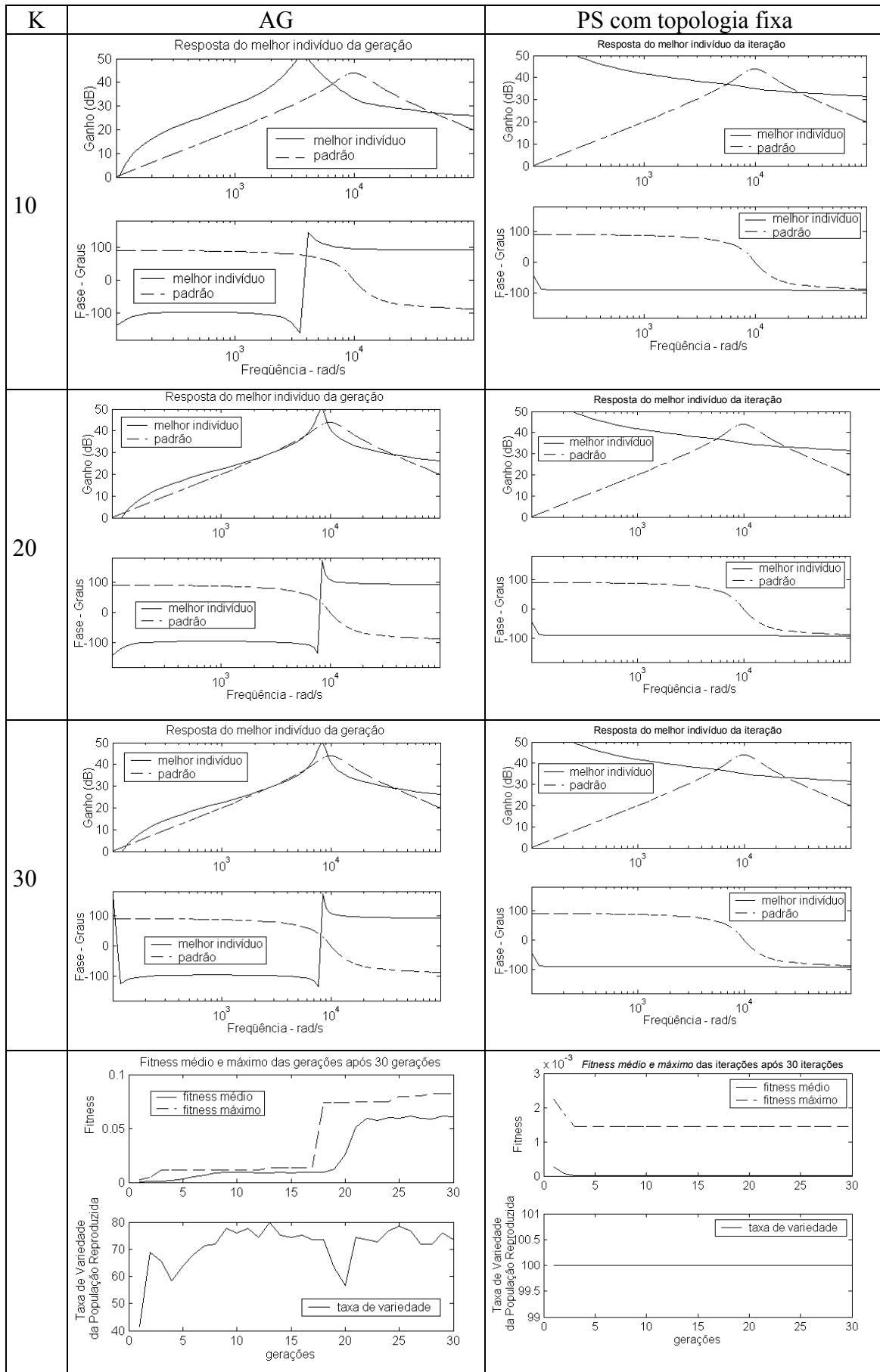


Figura 4.10.a:  $A_{v\max} = +45$  dB - gráficos dos testes com 30 gerações evoluindo continuamente.



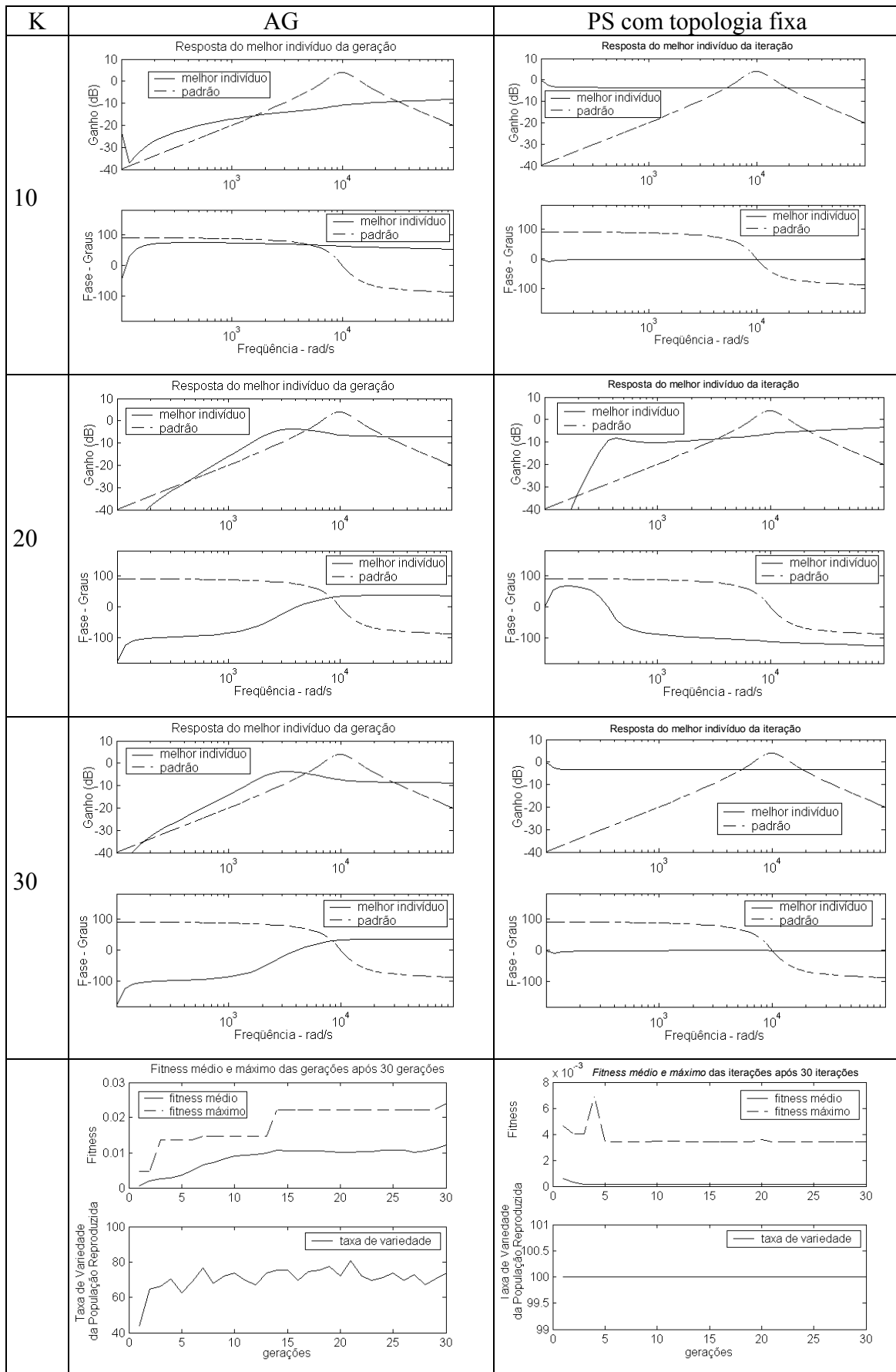


Figura 4.10.b:  $A_{v\max} = +5$  dB - gráficos dos testes com 30 gerações evoluindo continuamente.

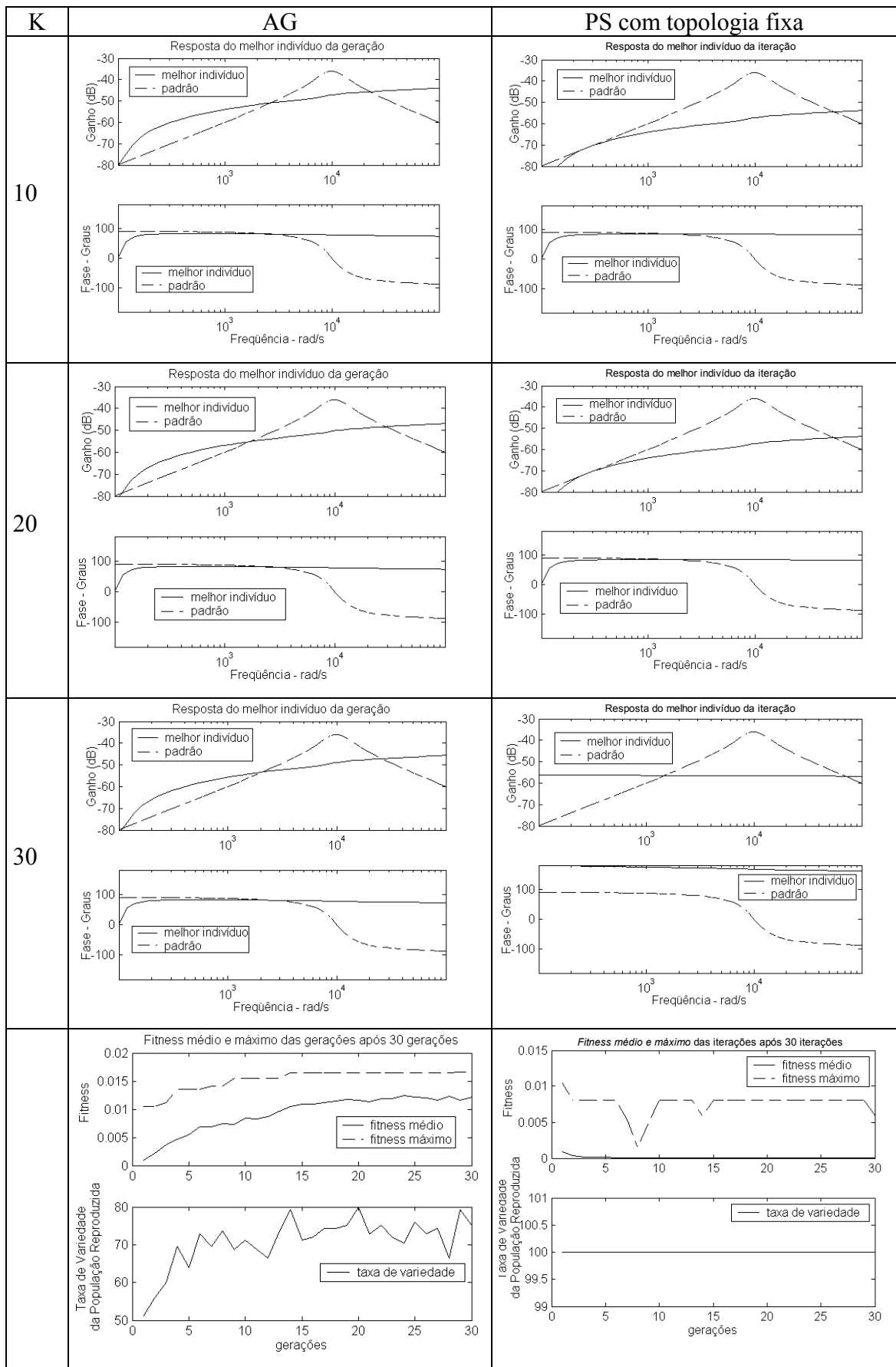


Figura 4.10.c:  $A_{v\max} = -30$  dB - gráficos dos testes com 30 gerações evoluindo continuamente.

Pode ser observado dos resultados anteriores que o AG possui características evolutivas mais adequadas que o PS, que, no entanto, apresenta uma convergência mais rápida que o AG, além de possuir um tempo entre iterações menor que o do AG.

Na figura 4.10.a, o gráfico de aptidão do PS apresenta pequenas oscilações em torno de um valor médio de *fitness*. Podemos interpretar esse fato como uma característica do não-elitismo do PS, que pode apresentar pequenas oscilações na posição da partícula mais apta, devido à influência dos seus vizinhos. Como podem existir muitos vizinhos com aptidão baixa, eles podem atrair a melhor partícula, que ficaria oscilando entre mínimos locais.

#### 4.3.2. Topologias fixas e Valores de Componentes Variáveis por 60 gerações

Posteriormente foram realizadas simulações com 60 gerações, também continuamente, para uma população inicial de 250 indivíduos com 15 genes cada um, para cada uma das três funções-objetivo. A tabela 4.8 e as Figuras 4.11.a, 4.11.b e 4.11.c mostram os resultados alcançados. O índice K significa Geração/Iteração.

Tabela 4.8: Resumo dos testes com 60 gerações evoluindo continuamente.

<i>Métodos empregados e Topologia</i>	<i>Ganho máx. (dB)</i>	<i>fitness máximo</i>	<i>Duração loop (s)</i>	<i>Método Evoluiu, ficou estável ou regrediu?</i>	<i>Converge em quantas gerações?</i>
AG	+45	0,1	185	evoluiu	7
	+5	0,062	211	evoluiu	5
	-30	0,016	194	evoluiu	5
PS com topologia fixa	+45	0,0015	182	regrediu	2
	+5	0,012	207	oscilou	1
	-30	0,001	189	oscilou	2

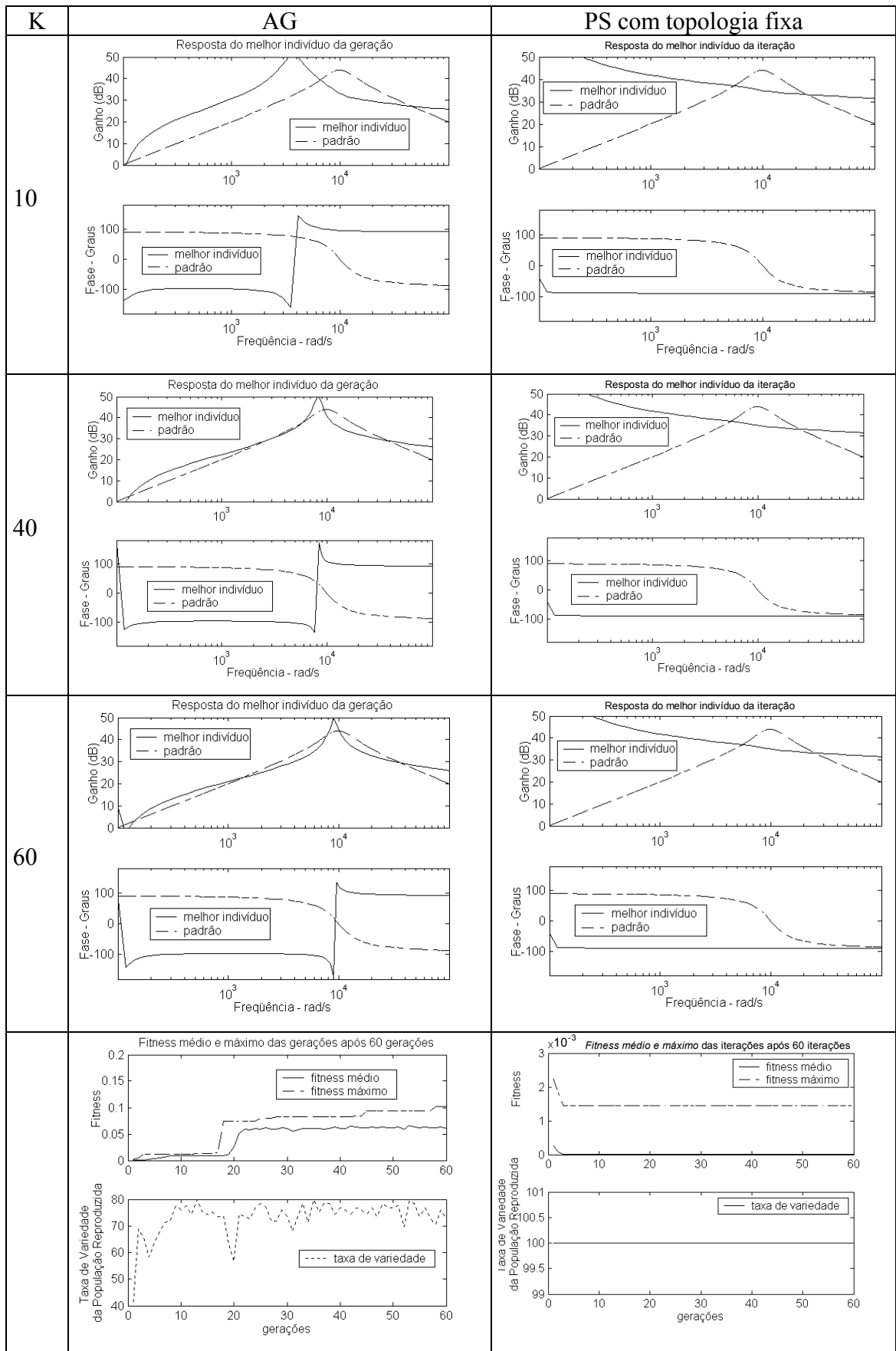


Figura 4.11.a:  $A_{v\max} = +45$  dB - gráficos dos testes com 60 gerações evoluindo continuamente.

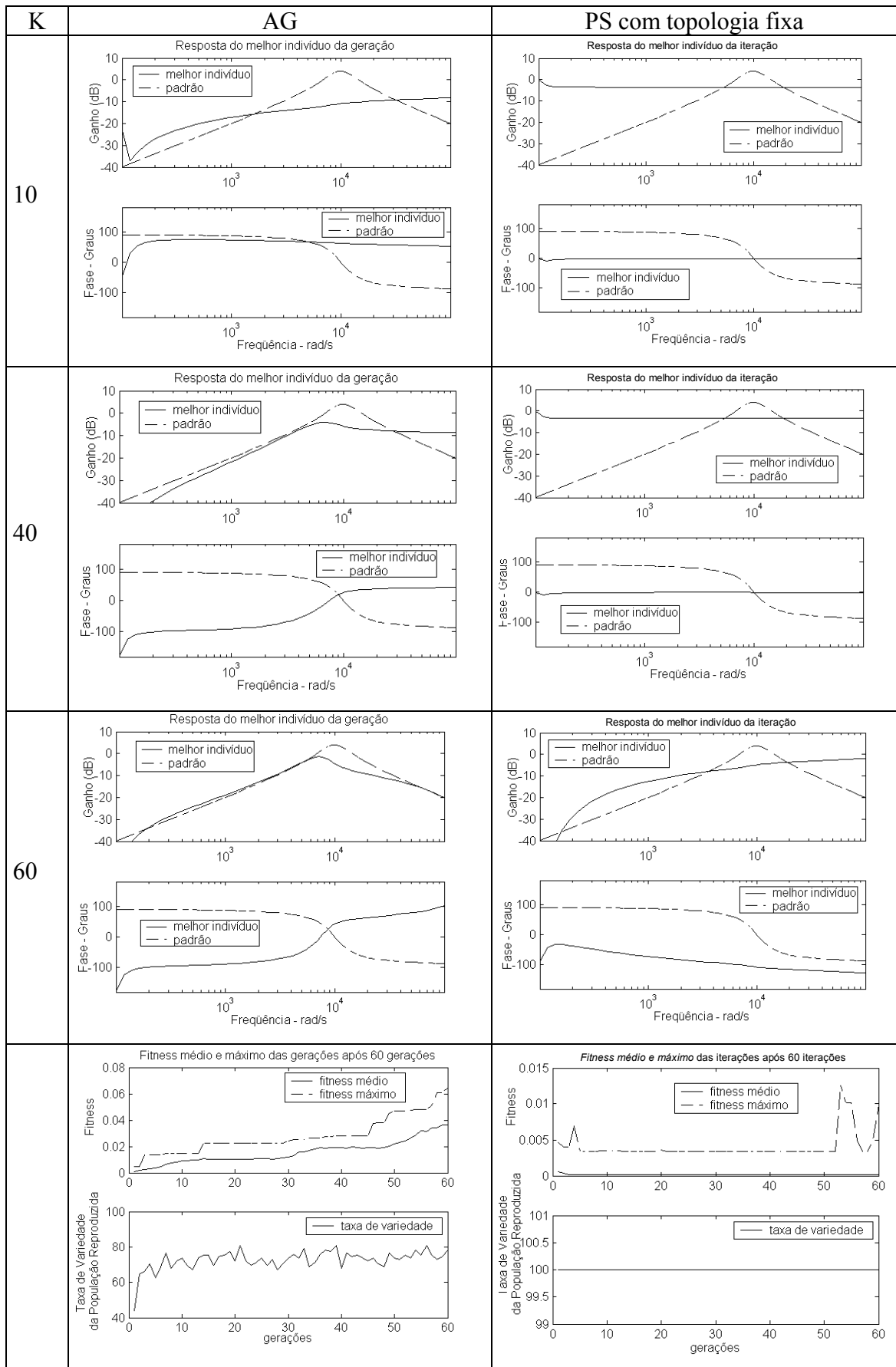


Figura 4.11.b: Avmáx = +5 dB - gráficos dos testes com 60 gerações evoluindo continuamente.

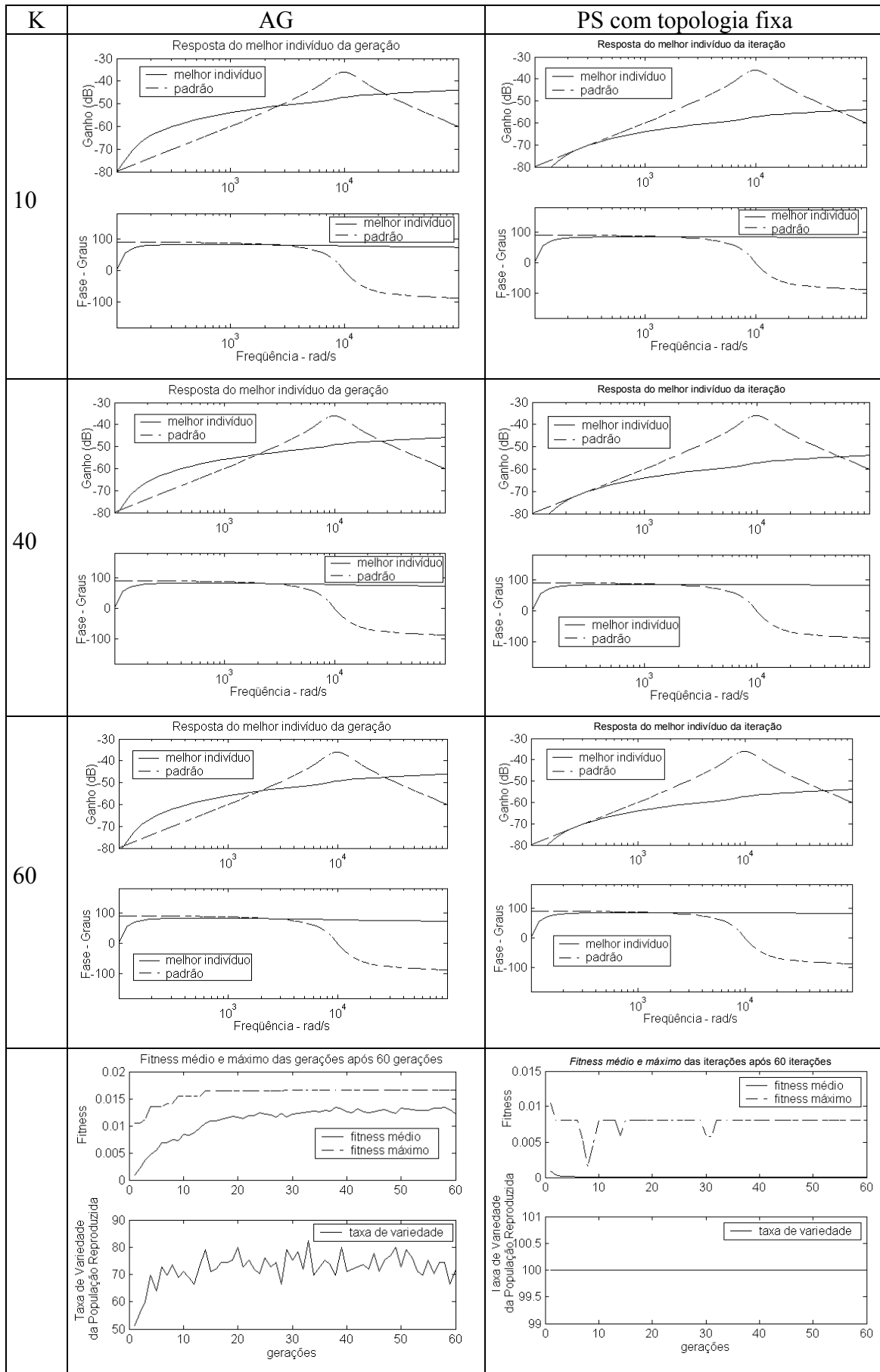


Figura 4.11.c:  $A_{v\max} = -30$  dB - gráficos dos testes com 60 gerações evoluindo continuamente.

Novamente, pode ser observado dos resultados apresentados que o AG possui características evolutivas melhores que o PS, sendo que este último apresenta uma convergência mais rápida que o AG, além de possuir um tempo entre iterações menor que o do AG.

Na figura 4.11.b, o gráfico de aptidão do PS apresenta comportamento constante em torno de um valor médio de *fitness*, mas no final ocorre um grande aumento no valor de *fitness*, seguido de uma queda e um novo aumento. Este fato sugere que podem existir muitos vizinhos com aptidão baixa, atraindo a melhor partícula para um mínimo local. No entanto, se um grupo de vizinhos apresentar em uma iteração um valor de aptidão mais alto, juntamente com um aumento da aptidão do melhor indivíduo, pode ocorrer a saída do mínimo local.

#### 4.3.3. Topologias fixas e Valores de Componentes Variáveis por mais 30 gerações depois de 30 gerações iniciais

Finalmente foram realizadas novas simulações, tendo em vista os resultados encontrados para as técnicas sócio-cognitivas, só que agora entregando como a última população simulada para o *Particle Swarm* a população já evoluída pelo Algoritmo Genético após 30 gerações, e que foi analisada na Tabela 4.2. Esta população também possui 250 indivíduos com 15 genes cada um. A tabela 4.9 e as Figuras 4.12.a, 4.12.b e 4.12.c mostram os resultados alcançados. O índice K significa Geração/Iteração.

**Tabela 4.9: Resumo dos testes com mais 30 gerações depois de 30 iniciais.**

<i>Métodos empregados e Topologia</i>	<i>Ganho máx. (dB)</i>	<i>fitness máximo</i>	<i>Duração loop (s)</i>	<i>Método Evoluiu, ficou estável ou regrediu?</i>	<i>Converge em quantas gerações?</i>
AG	+45	0,08	199	ficou estável	-
	+5	0,05	221	evoluiu	15
	-30	0,016	192	ficou estável	-
PS com topologia fixa	+45	0,08	197	oscilou	2
	+5	0,04	217	oscilou	1
	-30	0,035	190	oscilou	15

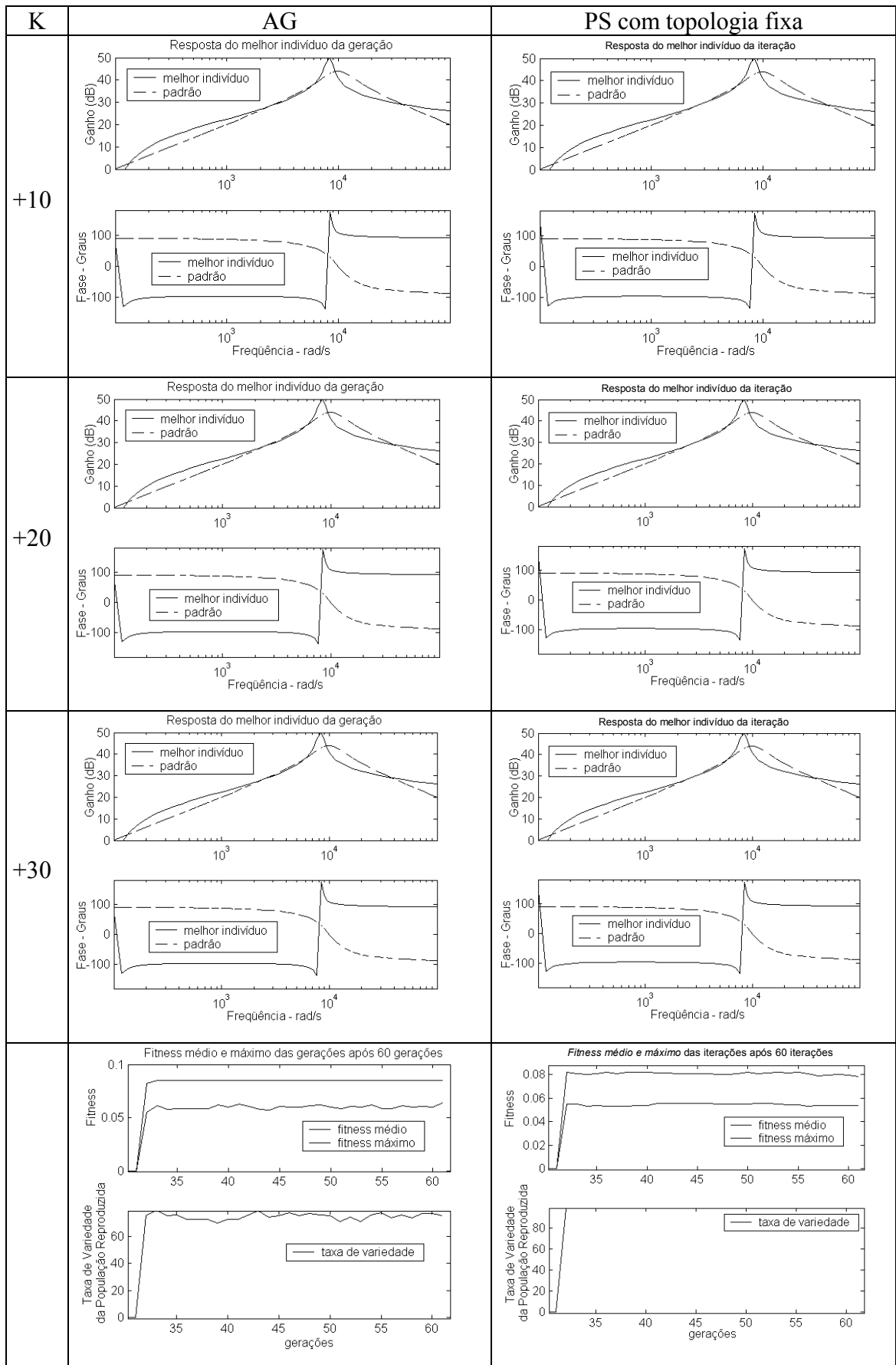


Figura 4.12.a:  $A_{v\max} = +45$  dB - gráficos dos testes com mais 30 gerações depois de 30 iniciais.



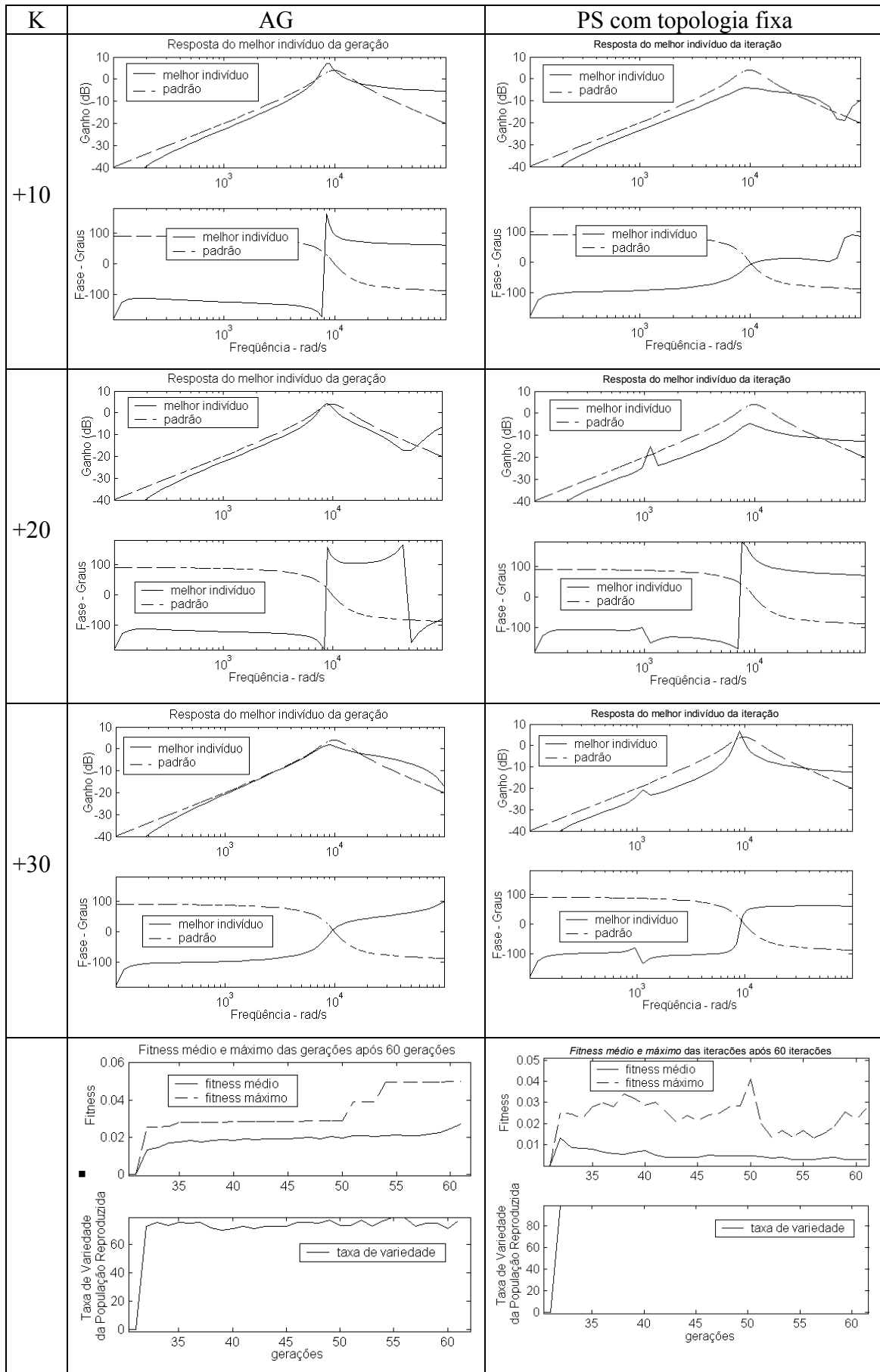


Figura 4.12.b:  $A_{v\max} = +5$  dB - gráficos dos testes com mais 30 gerações depois de 30 iniciais.

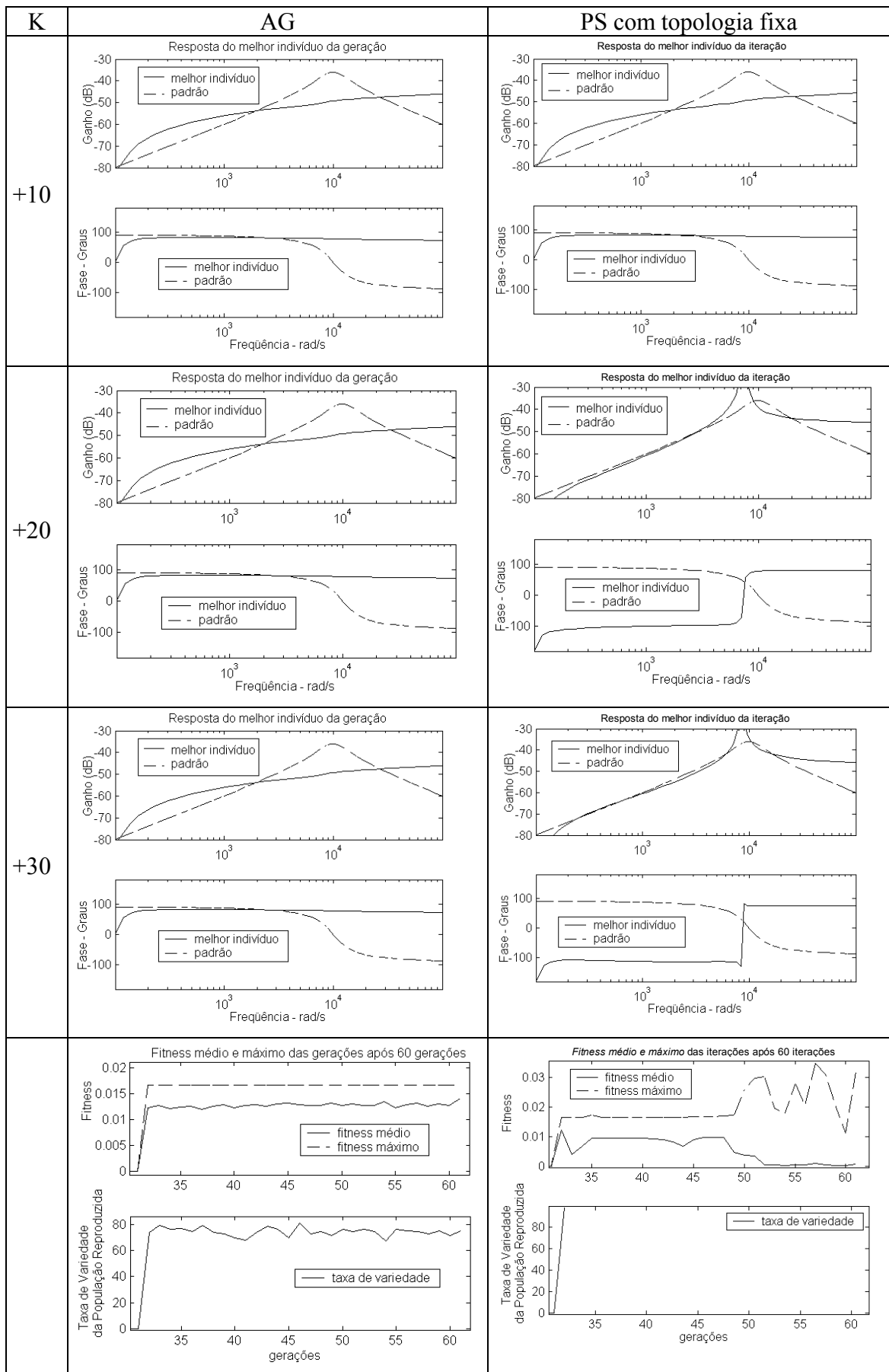


Figura 4.12.c:  $A_{v\max} = -30$  dB - gráficos dos testes com mais 30 gerações depois de 30 iniciais.

Somente neste caso pode ser observado dos resultados apresentados que o PS possui características evolutivas melhores que o AG, que ficou estável por duas vezes. Ainda, o PS apresenta uma convergência mais rápida que o AG, além de possuir um tempo entre iterações menor que o do AG.

Na figura 4.12.c, o gráfico de aptidão do PS apresenta comportamento oscilatório em torno de um valor médio de *fitness*, durante as últimas iterações da simulação. Provavelmente existiam vários mínimos locais e as partículas estavam fortemente atraídas por eles. Quando um grupo de partículas conseguiu aumentar sua aptidão, o restante pôde ser “arrastado” junto para estas novas regiões do “hiperespaço”.

#### 4.3.4. Análise de Robustez – Monte Carlo

A estabilidade dos circuitos obtidos pelos métodos foi aferida através da análise de Monte Carlo.

Foi escolhido para esta análise apenas um tipo de filtro.

Foi estabelecido que a variação percentual dos componentes ficaria em torno de 10%, variando em cada simulação apenas um tipo de componente, em um dos seu parâmetros.

No caso dos componentes passivos, foi efetuada a variação, segundo uma curva gaussiana, dos seus valores nominais.

Para os componentes ativos, a variação foi no parâmetro Bf dos transistores NPN internos do modelo.

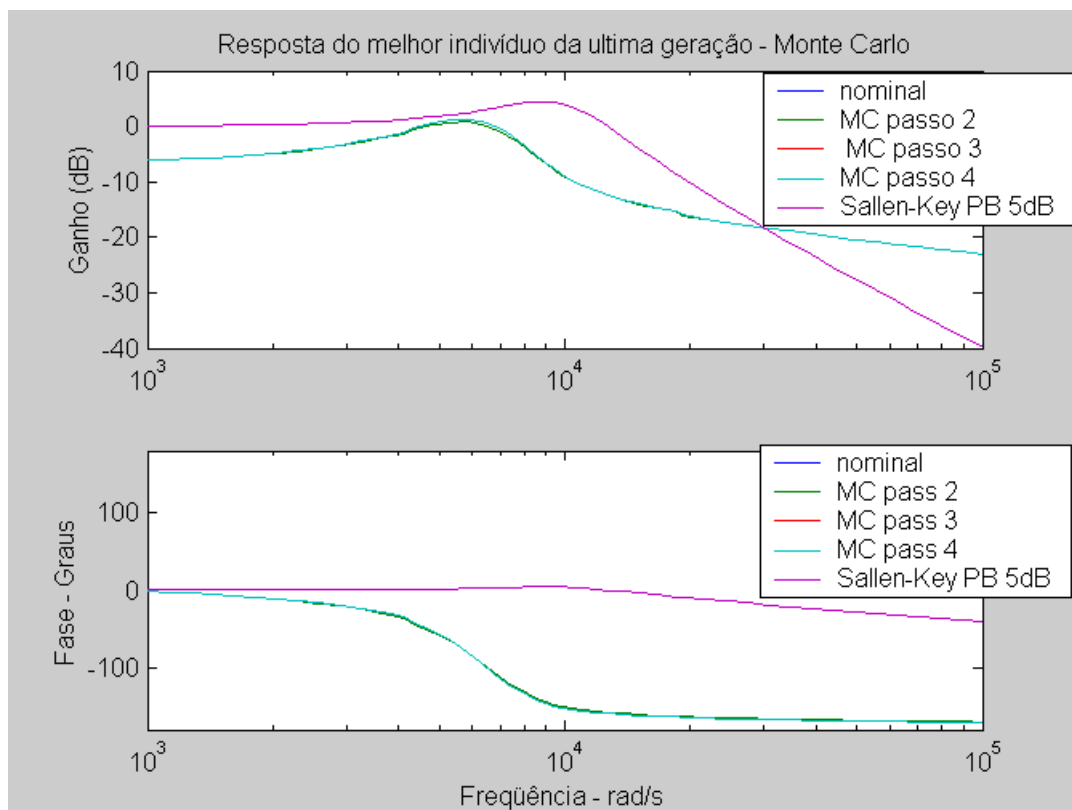


Figura 4.13.c: Análise de Robustez – Monte Carlo.

Pode ser observado que o circuito gerado possui uma característica de robustez e estabilidade, uma vez que os valores do módulo do ganho e da fase apresentam uma pequena variação, quase imperceptível, com relação ao valor nominal.

Um dos motivos para isso foi o fato de ter sido utilizado o amplificador operacional como elemento de ganho. O mesmo possui grande estabilidade e robustez, além de não apresentar problemas de polarização DC, visto que o algoritmo não necessita processar este tipo de informação. Os circuitos gerados que possuem o amplificador operacional como componente, já estão desde o momento de sua criação, com sua polarização bem definida, pois cada AMPOP já está polarizado corretamente nas instruções que o geram em sua codificação, conforme foi visto no capítulo 3.

#### 4.3.5. Tempo de Execução dos Algoritmos por 60 gerações em um mesmo computador

A Tabela 4.10 apresenta o tempo de execução, no formato hora, minuto e segundo, de cada um dos testes efetuados. A máquina utilizada para a execução estava configurada com um processador Pentium IV 1.2 GHz e com 512MB de RAM.

**Tabela 4.10: Tempo de execução para 60 gerações (horas).**

<i>Métodos empregados e Topologia</i>	<i>Ganho máx. em dB</i>	<i>Tempo em horas, minutos e segundos</i>
<i>Butterworth</i> passa-baixas de 2 <sup>a</sup> . ordem	+30	3:00:36
	0	3:02:21
	-30	X
<i>Sallen-Key</i> passa-baixas de 2 <sup>a</sup> . ordem	+45	3:05:12
	+5	3:04:21
	-35	3:03:06
<i>Sallen-Key</i> passa-faixa de 2 <sup>a</sup> . ordem	+45	3:00:51
	+5	3:00:11
	-35	3:01:00

X → Não foi simulado neste computador em questão.

Apesar do tempo de processamento ser, aparentemente, elevado, o algoritmo implementado mostrou-se bastante estável e razoavelmente rápido, sem comparado a outros trabalhos [31,32,33] na área, que necessitam de uma grande quantidade de recursos computacionais.

## 5. Conclusões

Esta Tese propôs um método para a síntese automática de circuitos eletrônicos analógicos contínuos no tempo para filtros analógicos ativos, utilizando técnicas evolutivas e sócio-cognitivas, sendo apresentada a seguir uma análise sobre a aptidão das técnicas usadas.

O método consiste de uma sistemática de codificação dos circuitos, simulação e avaliação das soluções obtidas. O método utiliza apenas resistores, capacitores e amplificadores operacionais como componentes, dada a característica de resposta no domínio da frequência de um filtro analógico desejado. O processo de análise se utiliza de métricas de desempenho adequadas para comparar as duas técnicas

Na avaliação das técnicas propostas, foram utilizados três tipos de filtros analógicos diferentes, do tipo passa-baixas e passa-faixa, e de amplo conhecimento, como funções-objetivo para o algoritmo.

Os resultados demonstram que a técnica evolutiva é superior à sócio-cognitiva no que concerne à capacidade de buscar soluções no espaço de soluções possíveis, sem perder os seus melhores resultados e evoluindo continuamente.

No entanto, pelos resultados obtidos para os filtros sintetizados, conclui-se que a técnica sócio-cognitiva é mais indicada para realizar o ajuste fino dos valores dos componentes passivos do circuito, uma vez que converge mais rapidamente que a técnica evolutiva. A princípio, algumas iterações da técnica sócio-cognitiva já são suficientes para identificação de indivíduos não tão aptos.

As características da vizinhança foram realmente os fatores preponderantes para fazer com que as partículas não ao menos mantivessem seu valor de *fitness*.

A medida de robustez dos melhores circuitos gerados por ambas as técnicas foi avaliada pelo uso do método de Monte Carlo. Os dados obtidos comprovam que o algoritmo é capaz de gerar circuitos robustos, sem problemas de polarização DC e com grandes chances de, quando montados em laboratório, funcionarem do modo esperado, uma vez que foram utilizados componentes passivos com valores e tolerâncias comerciais, bem como um amplificador operacional amplamente utilizado na indústria.

Por fim, conclui-se que a contribuição desta Tese está em apresentar, ainda que embrionariamente, novas idéias e um novo método para síntese de circuitos analógicos,

mesclando capacidades de uma e outra técnica, de modo a conseguir gerar novas topologias de circuitos, ainda não imaginadas, de modo eficiente.

## 6. Sugestões para trabalhos futuros

Certamente, uma proposta interessante seria desenvolver uma técnica mista, utilizando as melhores características de cada técnica. Poderia ser implementado um algoritmo onde o *loop* principal seria da técnica evolutiva, ficando a técnica sócio-cognitiva como um *loop* interno. Assim, para uma determinada população de  $n$  indivíduos de  $g$  genes evoluindo por geração  $G$ , ao final de cada geração seriam rodadas  $i$  iterações do *particle swarm* somente variando os valores dos componentes passivos, e mantendo a topologia fixa. Seria um método que utiliza a técnica evolutiva para mapear o espaço de soluções e que faz uma sintonia fina através da técnica sócio-cognitiva.

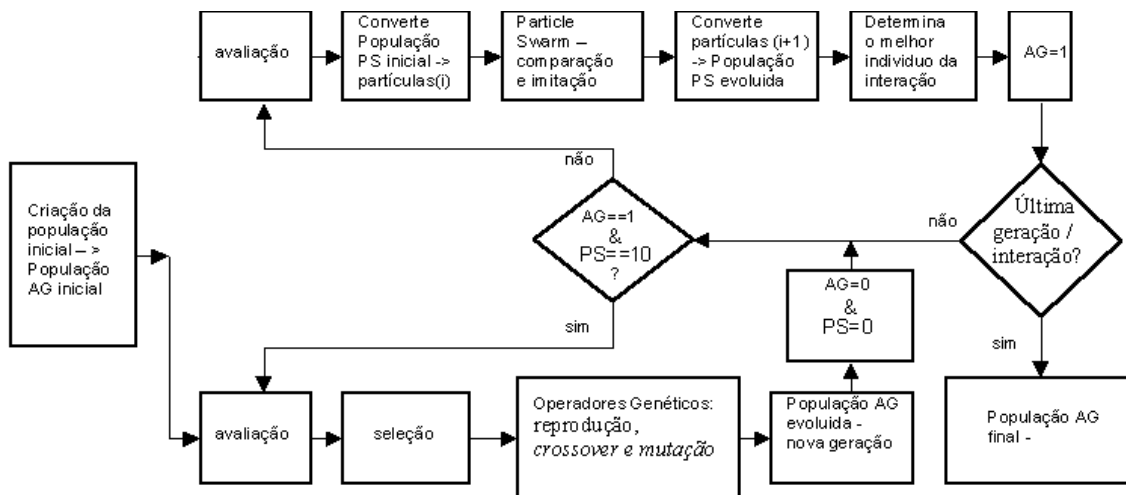


Figura 6.1: Fluxograma do algoritmo para a técnica mista proposta.

Outra possibilidade seria, partindo de uma resposta inicial no domínio da frequência de um filtro conhecido, ir gradualmente variando esta resposta conforme a aptidão ou *fitness* do melhor circuito gerado para um outro filtro desejado, de modo a acelerar ou facilitar a busca. Seria como realizar a conformação da resposta desejada!

Seria possível, quando da geração da população inicial, utilizar os diversos padrões de filtro para calcular o *fitness* de cada indivíduo desta população para cada um destes padrões e, assim, classificar e selecionar os mesmos para criar um primeiro algoritmo classificador. Desse modo, estaríamos aproveitando indivíduos que, sendo

criados ao acaso, poderiam ser eliminados para um determinado padrão de filtro previamente escolhido por apresentar um baixo *fitness* para o mesmo, mas que no entanto podem apresentar um alto *fitness* para outros padrões que não seriam nunca testados.

Deve-se criar uma nova medida para o cálculo do fitness onde seriam incluídos também os valores de fase, e não somente o ganho. O algoritmo deve então ser testado novamente para verificar se ficou muito mais lento que a versão atual.



## 7. Referências Bibliográficas

- [1] Bardeen, J. e Brattain, W. H., Physical Review, v. 74, pp. 230, 1948.
- [2] Aaserud, O. e Nielsen, I., "Trends in current analog design: A panel debate", In: Proceedings of Analog Integrated Circuits and Signal Processing, pp. 5-9, 1995.
- [3] Daryanani, Gobind. Principles of Active Network Synthesis and Design. John Wiley and Sons, 1976.
- [4] Johns, D. A., Martins, K., Analog Integrated Circuit Design, John Wiley & Sons, 1997.
- [5] Sallen, R.P., Key, E.L., A Practical Method of Designing RC Active Filters, IRE Transactions on Circuit Theory, CT-2, May, pp 74-85, 1955.
- [6] Hill, F.J., Peterson, G.R., Introduction to Switching Theory and Logical Design, John Wiley and Sons, 1981.
- [7] Koza, John R., Genetic Programming: on the programming of computers by means of natural selection. The MIT Press, 5<sup>th</sup> printing, 1996.
- [8] Zebulum, Ricardo S., Pacheco, M.A.C. and Vellasco, M.M.B.R.. Evolutionary Electronics: Automatic Design of Electronic Circuits and Systems by Genetic Algorithms. CRC Press, 2002.
- [9] Tuinenga, Paul W.. SPICE: A guide to circuit simulation and analysis using Pspice. 2<sup>nd</sup> ed. Prentice Hall, 1992.
- [10] Filho, Sidnei N.. Filtros Seletores de Sinais. Editora da UFSC, Florianópolis, 1998.
- [11] Hanselman, D., Littlefield, B. MATLAB 5 Versão do Estudante: Guia do Usuário. Makron Books do Brasil, 1999.

- [12] Calvano, José V.. Geração de Testes e Projeto Visando a Testabilidade de Circuitos Analógicos. Tese de Doutorado, COPPE, Rio de Janeiro, 2000.
- [13] Savioli, Carlos Eduardo de Freitas. Geração Automática de Vetores de Teste Utilizando Técnicas Evolutivas e Sócio-Cognitivas. Tese de Mestrado, COPPE, Rio de Janeiro, 2005.
- [14] Savioli, C.E.F., Szendrodi C.E.C., Calvano, J.V., Mesquita, A.C., Lubaszewsky, M.S. , A Rank-Based Genetic Algorithm for Fault Tolerant Analog Circuit Synthesis, Proceedings of the 4<sup>th</sup> IEEE Latin American Test Workshop, pp 142-145, 2003.
- [15] Novaes, U. R., Agrupamento de Dados Através de Algoritmos Swarm, Tese de Mestrado, COPPE, Rio de Janeiro, 2002.
- [16] Kennedy, J., Eberhart, R. C., Swarm Intelligence, Morgan Kaufmann Publishers, 2001.
- [17] Eberhart, R. C., Kennedy, J., A New Optimizer using Particle Swarm Theory, Proceedings of the 6<sup>th</sup> International Symposium on Micro-Machine and Human Science, pp39-43, Nagoya, Japan, 1995.
- [18] Sedra, A., Smith, K., Microeletrônica, volume 2, Makron Books do Brasil, São Paulo, 1995.
- [19] Lathi, B.P, Sistemas de Comunicação, Editora Guanabara Dois, Rio de Janeiro, 1979.
- [20] Savioli, C.E.F., Szendrodi C.E.C., Calvano, J.V., Mesquita, A.C. , ATPG for Fault Diagnosis on Analog Electrical Networks Using Evolutionary Techniques. In: 17th Symposium on Integrated Circuits and System Design - SBCCI 2004, 2004, Porto de Galinhas. Proceedings of the 17th Symposium on Integrated Circuits and System Design, pp 100-104, 2004.

- [21] Savioli, C.E.F, Calvano, J.V., Filho, A.C.M., Detecting Regions of Insensitivity on Parametric Faults in Frequency Domain for Continuous-Time Analog Filters, Submitted to the 6<sup>th</sup> Latin American Test Workshop, Salvador, 2005.
- [22] Holland, J. H., Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Harbor, 1975.
- [23] Malvino, Albert Paul, Eletrônica, Volumes I e II, Editora McGraw-Hill, São Paulo, 2<sup>o</sup> e 4<sup>o</sup> Edições, 1987 e 1995.
- [24] Haykin, Simon, Redes Neurais: princípios e prática, 2<sup>a</sup>. edição, Bookman, Porto Alegre, 2001.
- [25] Larson, H. J., Introduction to Probability Theory and Statistical Inference, John Wiley and Sons, 1982.
- [26] EIA – *Electronic Industries Alliance* - <http://www.eia.org/>
- [27] ECA - *Electronic Components, Assemblies, and Materials Association* - <http://www.ec-central.org/>
- [28] Queiroz, Antônio Carlos Moreirão de, Redes RLC Simétricas e Antimétricas de Baixa Sensibilidade, Tese de Mestrado, COPPE, Rio de Janeiro, 1984.
- [29] Evolvable Hardware for Space Exploration, Benny Toomarian, The Third NASA/DoD Workshop on Evolvable Hardware, 2001.
- [30] Nanotechnology in Information Processing-Oportunities and Challenges, M. Meyyappan, The Third NASA/DoD Workshop on Evolvable Hardware, 2001.

[31] Synthesis of CMOS Analog Cells using AMIGO, Ramy Iskander\*, Mohamed Dessouky, Maie Aly, Mahmoud Magdy, Noha Hassan, Noha Soliman, Sami Moussa, Design, Automation and Test in Europe Conference and Exhibition, 2003

[32] A Multi-Objective Hierarchical Methodology for Synthesis of Large Scale Electronic Designs, Trent McConaghy, The Third NASA/DoD Workshop on Evolvable Hardware, 2001.

[33] Jason D. Lohn, Silvano P. Colombano, A Circuit Representation Technique for Automated Circuit Design, IEEE Transactions on Evolutionary Computation, vol. 3, no. 3, pp. 205-219, 1999.

[34] Fault Recovery in Linear Systems via Intrinsic Evolution, Charles Rice, Garrison W. Greenwood and David Hunter, The 2004 NASA/DoD Conference on Evolvable Hardware (EH2004), 2004.

[35] Analog circuit synthesis based on reuse of topological features of prototype circuits, Hajime Shibata & Nobuo Fujii, The Third NASA/DoD Workshop on Evolvable Hardware, 2001.

[36] Reconfigurable Devices in the 21<sup>st</sup> Century- An Evolutionary Perspective, Steve Trimberger, The Third NASA/DoD Workshop on Evolvable Hardware, 2001.

## **APÊNDICE I. ARTIGOS**

Neste apêndice são apresentados alguns artigos sobre o assunto desta tese que foram apresentados em congressos e workshops, bem como artigos de submissão já aceita.