Auto-Organização da População em Sistemas Imunológicos Artificiais aplicada ao *docking* de proteínas

Helder Ken Shimo

Dissertação apresentada ao programa
interunidades em bioinformática da
Universidade de São Paulo para obtenção do
grau de Mestre em Ciências

Área de concentração: Bioinformática

Orientador: Prof. Dr. Renato Tinós

Ribeirão Preto, Maio de 2012

AGRADECIMENTOS

Ofereço meus agradecimentos, em primeiro, lugar a todos aqueles que estão próximos de mim: minha mãe, meus velhos amigos e meus amigos apresentados pela grande USP (Universidade de São Paulo), em especial aos paubistas, por simplesmente estarem lá.

Agradeço também ao Prof. Renato Tinós, pela orientação em todos dos mais importantes trabalhos que desenvolvi na universidade.

Agradeço à Fundação de Amparo à Pesquisa do Estado de São Paulo pelo auxílio financeiro, fundamental para minha manutenção.

RESUMO

Vários problemas do mundo real podem ser analisados como problemas de otimização. Na bioinformática, em especial, como exemplos podem ser citados o alinhamento múltiplo de sequências, a filogenia, a predição de estruturas de proteínas e RNA, entre outros. As Meta-heurísticas Populacionais (MhP) são técnicas baseadas em interações de conjuntos de soluções candidatas, como elementos de uma população, utilizadas na otimização de funções. Seu uso é especialmente interessante na otimização de problemas onde há conhecimento parcial ou nenhum do espaço de busca. O objetivo deste trabalho é investigar o uso de auto-organização da população de um sistema imunológico artificial (AIS) a fim de aplicá-lo no problema de *docking*, que pode ser visto como um problema de otimização multimodal complexo. O AIS é um tipo de MhP inspirado na microevolução do sistema imunológico adaptativo de organismos complexos. Neste, as soluções candidatas representam células do sistema imunológico que busca se adaptar para a eliminação de um patógeno. O desenvolvimento do algoritmo foi baseado no opt-aiNet, que utiliza dos princípios das teorias de seleção clonal e maturação de afinidade para realizar a otimização de funções. Adicionalmente, o opt-aiNet, inspirado na teoria de redes imunológicas, realiza uma etapa de supressão, que busca eliminar soluções semelhantes, aumentando assim a diversidade populacional. Esta etapa é computacionalmente custosa, dado que é feito o cálculo da distância entre todos os possíveis pares de células (soluções) afim de eliminar aquelas próximas de acordo com um dado critério. A proposta deste trabalho é o desenvolvimento de um algoritmo de supressão auto-organizável, inspirado no fenômeno da criticalidade auto-organizada, buscando diminuir a influência da seleção de parâmetros e a complexidade da etapa de supressão. O algoritmo proposto foi testado em um conjunto de funções contínuas conhecidas e comumente utilizadas pela comunidade de computação evolutiva. Os resultados obtidos foram comparados com aqueles de uma implementação do opt-aiNet. Em adição, foi proposta a utilização de operadores de mutação com distribuição q-gaussiana nos AISs desenvolvidos. O algoritmo foi também aplicado no problema de *docking* rígido baseado em complementaridade de superfícies e minimização de colisões, especificamente no *docking* de proteínas. Os resultados foram comparados com aqueles de um algoritmo genético, resultando em um melhor desempenho obtido pelo algoritmo proposto.

ABSTRACT

Many real world problems can be described as optimization problems. In bioinformatics in special, there is multiple sequence alignment, filogeny and RNA and Protein structure prediction, among others. Population based metaheuristics are techniques based in the interaction of a set of candidate solutions as elements of a population. Its use is specially interesting in optimization problems where there is little or no knowledge of the search space. The objective of this work is to study the use of self-organization of population in an artificial immune system for use in the docking problem, considered a complex multimodal optimization problem. The artificial immune system is a type of population based methaheuristics inspired in the microevolution of the adaptive immune system of complex organisms. Candidate solutions represent cells of the immune system adapting its antibodies to eliminate a pathogen. The development of the algorithm was based in the opt-aiNet, based in the principles of clonal selection and affinity maturation for function optimization. Additionally, the opt-aiNet, inspired in theories of immune network, makes a suppression stage to eliminate similiar solutions and control diversity. This stage is computationally expensive as it calculates the distance between every possible pair of cells (solutions) eliminating those closer than a threshold. This work proposes a self-organized suppression algorithm inspired in the self-organized criticality, looking to minimize the influence of parameter selection and complexity of the suppression stage in opt-aiNet. The proposed algorithm was tested in a set of well-known functions in the evolutionary computation community. The results were compared to those of an implementation of the opt-aiNet. In addition, we proposed a mutation operator with q-Gaussian distribution for the artificial immune systems. The algorithm was then applied in the rigid protein docking problem based in surface complementarity and colision avoidance. The results were compared with a genetic algorithm and achieved a better performance.

SUMÁRIO

Agradecimentos	i
Resumo	iii
Sumário	vii
Lista de figuras	ix
Lista de tabelas	xi
Lista de abreviaturas	xiii
1. Introdução	1
1.1. Objetivos	3
1.2. Organização do documento	3
2. Algoritmos evolutivos e criticalidade auto-organizada	5
2.1. Criticalidade auto-organizada	5
2.2. algoritmos evolutivos	7
2.3. Criticalidade auto-organizada aplicada a Algoritmos evolutivos	9
3. Sistemas imunológicos artificiais em otimização contínua	13
4. <i>Docking</i> molecular	17
4.1. Classificações de problemas de <i>docking</i>	21
4.2. Tipos de funções de <i>score</i>	22
4.3. Computação evolutiva aplicada no <i>docking</i> molecular	23
5. Metodologia proposta	27
5.1. Algoritmo proposto: <i>socopt-aiNet</i>	27
5.2. Algoritmo proposto: SOCAIN	29
5.2.1. Aplicação de mutação com distribuição q-gaussiana	30
5.3. Avaliação de Algoritmos Evolutivos	32
5.4. Codificação do problema de <i>docking</i>	34
5.4.1. Codificação da superfície excluída ao solvente	34
5.4.2. Função de <i>score</i>	35
5.4.3. Algoritmo genético de [GARDINER <i>et al.</i> , 2001]	38
6. Resultados e discussão	41
6.1. Experimentos com otimização de funções contínuas	41
6.1.1. Experimentos de <i>benchmark</i> do SOCOpt-aiNet	41
6.1.2. Experimentos de <i>benchmark</i> do SOCAIN	44
6.1.3. Experimentos com mutação q-gaussiana	49
6.1.4. Verificação de auto-organização da população	54
6.1.5. Avaliação do tempo de execução dos AISs	56
6.1.6. Comparação com outros algoritmos	58
6.2. Experimentos com <i>docking</i>	61
6.2.1. Estudo do efeito da restrição de nicho no SOCAIN	63
6.2.2. Aplicação do (q)SOCAIN no <i>docking</i>	65
6.2.3. Experimentos com o complexo proteico 1AVX	68
7. Conclusões	75
8. Referências bibliográficas	79
Apêndice	85
A.1. Experimentos de seleção de parâmetros do SOCOpt-aiNet	85

A.2.	Tabelas dos resultados dos experimentos de <i>benchmark</i>	96
A.3.	Experimentos iniciais de <i>docking</i>	125

LISTA DE FIGURAS

Figura 1: Exemplo do desenvolvimento do modelo de interação de espécies.	6
Figura 2: Complexo (em azul) alfa-quimotripsina A (de <i>Bos taurus</i>) – (em verde) inibidor ovomucóide (de <i>Meleagris gallopavo</i>), código pdb 1CHO. As proteínas em representação em fita e em transparência a superfície excluída ao solvente.	18
Figura 3: Estágios do <i>docking</i> adaptado de [SMITH & STERNBERG, 2002].....	19
Figura 4: Formas da distribuição q-gaussiana para diferentes valores de q [TINÓS & YANG, 2010]. Com $q=1$ temos a distribuição normal padrão, e com $q=2$, a distribuição Cauchy é reproduzida.	31
Figura 5: Representação da construção da superfície excluída ao solvente (adaptado de [SANNER <i>et al.</i> , 1996]).	35
Figura 6: Comparativo da frequências em que o SOCOpt-aiNet superou e foi superado pelo opt-aiNet para $N_D = 2$ (a esquerda) e 10 (a direita). Sinal + indica a frequência em que o algoritmo superou e S+ a frequência em que ocorre com significância estatística.	43
Figura 7: Comparativo da frequências em que o SOCOpt-aiNet superou e foi superado pelo opt-aiNet para $N_D = 30$ (a esquerda) e 50 (a direita).	44
Figura 8: Comparativo da frequências em que o SOCAIN superou e foi superado pelo SOCOpt-aiNet para $N_D = 2$ (a esquerda) e 10 (a direita).	45
Figura 9: Comparativo da frequências em que o SOCAIN superou e foi superado pelo SOCOpt-aiNet para $N_D = 30$ (a esquerda) e 50 (a direita).	46
Figura 10: Evolução do erro médio da função f_{25} em uma das execuções. Comparativo SOCOpt-aiNet com os SOCAIN.	47
Figura 11: Comparativo da frequências em que o SOCAIN superou e foi superado pelo opt-aiNet para $N_D = 2$ (a esquerda) e 10 (a direita).	48
Figura 12: Comparativo da frequências em que o SOCAIN superou e foi superado pelo opt-aiNet para $N_D = 30$ (a esquerda) e 50 (a direita).	49
Figura 13: Comparativo da frequências em que o (q)opt-aiNet superou e foi superado pelo opt-aiNet para $N_D = 2$ (a esquerda) e 10 (a direita).	50
Figura 14: Comparativo da frequências em que o (q)opt-aiNet superou e foi superado pelo opt-aiNet para $N_D = 30$ (a esquerda) e 50 (a direita).	50
Figura 15: Comparativo da frequências em que o (q)SOCAIN superou e foi superado pelo SOCAIN para $N_D = 2$ (a esquerda) e 10 (a direita).	52
Figura 16: Comparativo da frequências em que o (q)SOCAIN superou e foi superado pelo SOCAIN para $N_D = 30$ (a esquerda) e 50 (a direita).	52
Figura 17: Comparativo da frequências em que o (q)SOCAIN superou e foi superado pelo (q)opt-aiNet para $N_D = 2$ (a esquerda) e 10 (a direita).	53
Figura 18: Comparativo da frequências em que o (q)SOCAIN superou e foi superado pelo (q)opt-aiNet para $N_D = 30$ (a esquerda) e 50 (a direita).	53
Figura 19: Comparativo do <i>fitness</i> médio e ocorrência de extinções do (q)SOCAIN para uma execução da função f_1 com $N_D = 50$	55
Figura 20: Comparativo do <i>fitness</i> médio e ocorrência de extinções do (q)SOCAIN para uma execução da função f_9 com $N_D = 50$	55
Figura 21: Superfície da função Esfera (f_1) com $N_D=2$. As diferentes cores no gráfico são os valores de <i>fitness</i> como definidos na barra a direita.	56
Figura 22: Superfície da função Rastrigin (f_9) com $N_D = 2$	56
Figura 23: Representação gráfica da análise de complexidade do opt-aiNet e SOCAIN.	58
Figura 24: Comparativo do erro médio dos algoritmos de otimização para $N_D=10$	59

Figura 25: Comparativo do erro médio dos algoritmos de otimização para $N_D=30$	60
Figura 26: Comparativo do erro médio dos algoritmos de otimização para $N_D=50$	60
Figura 27: Comparativo das melhores execuções do AG de [GARDINER et al. 2001] sem (cima) e com restrição de nicho (baixo).....	64
Figura 28: Comparativo das melhores execuções do SOCAIN sem (acima) e com restrição de nicho (abaixo).	64
Figura 29: Comparativo das melhores execuções do (q)SOCAIN (acima) e SOCAIN (abaixo) na otimização do complexo 1CHO.	66
Figura 30: Melhor resultado do (q)SOCAIN para o complexo 1CHO.....	67
Figura 31: Melhor resultado do SOCAIN para o complexo 1CHO.	67
Figura 32: Melhor resultado do AG de [GARDINER et al., 2001] para o complexo 1CHO.....	68
Figura 33: Complexo (em azul) tripsina A (de <i>Sus scrofa</i>) – (em verde) inibidor de tripsina (de <i>Glycine max</i>), código pdb 1AVX.....	69
Figura 34: Comparativo das melhores execuções do (q)SOCAIN (acima) e SOCAIN (abaixo) na otimização do complexo 1AVX.	70
Figura 35: Melhor resultado do (q)SOCAIN para o complexo 1AVX.	71
Figura 36: Melhor resultado do SOCAIN para o complexo 1AVX.	71
Figura 37: Comparação da conformação nativa (em vermelho) com o de maior <i>score</i> do (q)SOCAIN (em verde).	72
Figura 38: Conformações obtidas pelos experimentos do <i>docking</i> do complexo 1AVX através do (q)SOCAIN.	73
Figura 39: Detalhe dos sítios de ligação do complexo nativo de 1CHO (esquerda) e 1AVX (direita). O ligante está em pontilhado verde e em vermelho o receptor.	73
Figura A.1: Evolução do erro médio da função f_{25} de uma execução do SOCOpt-aiNet e opt-aiNet.	85
Figura A.2: Evolução do erro médio da função f_{25} em uma das execuções. Comparativo dos parâmetros antigos (1), com os parâmetros novos (2).	96
Figura A.3: Melhor resultado da implementação do AG de [GARDINER et al., 2001] no experimento de validação.	126
Figura A.4: Melhor resultado do SOCOpt-aiNet no experimento de validação.	127
Figura A.5: Evolução do <i>fitness</i> do <i>docking</i> utilizando o SOCOpt-aiNet.	127
Figura A.6: Evolução do <i>fitness</i> do <i>docking</i> utilizando o AG de [GARDINER, et al., 2001].	127

LISTA DE TABELAS

Tabela 1: Comparativo entre AIS e AG [CASTRO & TIMMIS, 2002].	15
Tabela 2: Descrição das funções de <i>benchmark</i> [SUGANTHAN <i>et al.</i> , 2005; TINÓS & YANG, 2007].	33
Tabela 3: Tabela de complementaridades para tipo de superfície.	37
Tabela 4: Tabela de complementaridades para potencial de ligações de hidrogênio.	37
Tabela 5: Análise de complexidade do opt-aiNet e SOCAIN.	57
Tabela 6: Comparativo dos resultados de <i>fitness</i> obtidos de cinco execuções do AG e SOCAIN com e sem restrição de nicho.	63
Tabela 7: Comparativo dos resultados obtidos de cinco execuções do (q)SOCAIN com o (n)GA e SOCAIN na otimização do complexo 1CHO.	65
Tabela 8: Tempo médio de avaliação dos complexos 1CHO e 1AVX.	69
Tabela 9: Comparativo dos resultados obtidos de cinco execuções do (q)SOCAIN e SOCAIN na otimização do complexo 1AVX.	70
Tabela A.1: Resultados da seleção de parâmetros do intervalo de supressão (g_s) com $N_D=30$.	86
Tabela A.2: Resultados da seleção de parâmetros do intervalo de supressão (g_s) com $N_D=50$.	87
Tabela A.3: Resultados da seleção de parâmetros do limiar convergência (l_c) com $N_D=30$.	87
Tabela A.4: Resultados da seleção de parâmetros do limiar de convergência (l_c) com $N_D=50$.	88
Tabela A.5: Resultados da seleção de parâmetros do limiar de supressão (σ_s) com $N_D=30$.	88
Tabela A.6: Resultados da seleção de parâmetros do limiar de supressão (σ_s) com $N_D=50$.	89
Tabela A.7: Resultados dos erros $ f(x)-f(x^*) $ referentes às execuções com $N_D = 2$. Comparação dos parâmetros antigos com os selecionados.	89
Tabela A.8: Resultados dos erros $ f(x)-f(x^*) $ referentes às execuções com $N_D = 10$. Comparação dos parâmetros antigos com os selecionados.	91
Tabela A.9: Resultados dos erros $ f(x)-f(x^*) $ referentes às execuções com $N_D = 30$. Comparação dos parâmetros antigos com os selecionados.	92
Tabela A.10: Resultados dos erros $ f(x)-f(x^*) $ referentes às execuções com $N_D = 50$. Comparação dos parâmetros antigos com os selecionados.	94
Tabela A.11: Resultados dos erros $ f(x)-f(x^*) $ referentes às execuções com $N_D = 2$. Comparação do opt-aiNet com SOCOpt-aiNet.	96
Tabela A.12: Resultados dos erros $ f(x)-f(x^*) $ referentes às execuções com $N_D = 10$. Comparação do opt-aiNet com SOCOpt-aiNet.	98
Tabela A.13: Resultados dos erros $ f(x)-f(x^*) $ referentes às execuções com $N_D = 30$. Comparação do opt-aiNet com SOCOpt-aiNet.	99
Tabela A.14: Resultados dos erros $ f(x)-f(x^*) $ referentes às execuções com $N_D = 50$. Comparação do opt-aiNet com SOCOpt-aiNet.	100
Tabela A.15: Resultados dos erros $ f(x)-f(x^*) $ referentes às execuções com $N_D = 2$. Comparação do SOCOpt-aiNet com o SOCAIN.	101
Tabela A.16: Resultados dos erros $ f(x)-f(x^*) $ referentes às execuções com $N_D = 10$. Comparação do SOCOpt-aiNet com o SOCAIN.	102

Tabela A.17: Resultados dos erros $ f(x)-f(x^*) $ referentes às execuções com $N_D = 30$. Comparação do SOCOpt-aiNet com o SOCAIN.	103
Tabela A.18: Resultados dos erros $ f(x)-f(x^*) $ referentes às execuções com $N_D = 50$. Comparação do SOCOpt-aiNet com o SOCAIN.	105
Tabela A.19: Resultados dos erros $ f(x)-f(x^*) $ referentes às execuções com $N_D = 2$. Comparação do opt-aiNet com o SOCAIN.	106
Tabela A.20: Resultados dos erros $ f(x)-f(x^*) $ referentes às execuções com $N_D = 10$. Comparação do opt-aiNet com o SOCAIN.	107
Tabela A.21: Resultados dos erros $ f(x)-f(x^*) $ referentes às execuções com $N_D = 30$. Comparação do opt-aiNet com o SOCAIN.	108
Tabela A.22: Resultados dos erros $ f(x)-f(x^*) $ referentes às execuções com $N_D = 50$. Comparação do opt-aiNet com o SOCAIN.	109
Tabela A.23: Resultados dos erros $ f(x)-f(x^*) $ referentes às execuções com $N_D = 2$. Comparação do opt-aiNet com o (q)opt-aiNet.	110
Tabela A.24: Resultados dos erros $ f(x)-f(x^*) $ referentes às execuções com $N_D = 10$. Comparação do opt-aiNet com o (q)opt-aiNet.	112
Tabela A.25: Resultados dos erros $ f(x)-f(x^*) $ referentes às execuções com $N_D = 30$. Comparação do opt-aiNet com o (q)opt-aiNet.	113
Tabela A.26: Resultados dos erros $ f(x)-f(x^*) $ referentes às execuções com $N_D = 50$. Comparação do opt-aiNet com o (q)opt-aiNet.	114
Tabela A.27: Resultados dos erros $ f(x)-f(x^*) $ referentes às execuções com $N_D = 2$. Comparação do SOCAIN com o (q)SOCAIN.	115
Tabela A.28: Resultados dos erros $ f(x)-f(x^*) $ referentes às execuções com $N_D = 10$. Comparação do SOCAIN com o (q)SOCAIN.	116
Tabela A.29: Resultados dos erros $ f(x)-f(x^*) $ referentes às execuções com $N_D = 30$. Comparação do SOCAIN com o (q)SOCAIN.	118
Tabela A.30: Resultados dos erros $ f(x)-f(x^*) $ referentes às execuções com $N_D = 50$. Comparação do SOCAIN com o (q)SOCAIN.	119
Tabela A.31: Resultados dos erros $ f(x)-f(x^*) $ referentes às execuções com $N_D = 2$. Comparação do (q)opt-aiNet com o (q)SOCAIN.	120
Tabela A.32: Resultados dos erros $ f(x)-f(x^*) $ referentes às execuções com $N_D = 10$. Comparação do (q)opt-aiNet com o (q)SOCAIN.	121
Tabela A.33: Resultados dos erros $ f(x)-f(x^*) $ referentes às execuções com $N_D = 30$. Comparação do (q)opt-aiNet com o (q)SOCAIN.	122
Tabela A.34: Resultados dos erros $ f(x)-f(x^*) $ referentes às execuções com $N_D = 50$. Comparação do (q)opt-aiNet com o (q)SOCAIN.	123
Tabela A.35: Melhores <i>fitness</i> obtidos de cinco execuções do AG e SOCOpt-aiNet. ...	125

LISTA DE ABREVIATURAS

AE – Algoritmo Evolutivo;

AG – Algoritmo Genético;

AGL – Algoritmo Genético Lamarckiano;

AIS – Sistema Imunológico Artificial (*Artificial Immune System*);

CAPRI – Avaliação Crítica de Predição de Interações (*Critical Assessment of PRediction of Interactions*);

MhP – Meta-heurística Populacional;

SES – Superfície Excluída ao Solvente;

SOC – Criticalidade Auto-Organizada (*Self-Organizing Criticality*);

SOCAIN – Sistema Imunológico Artificial com Criticalidade Auto-Organizada (*Self-Organizing Criticality Artificial Immune System*);

SORIGA – Algoritmo Genético com Imigrantes Aleatórios Auto-Organizáveis (*Self-Organizing Random Immigrants Genetic Algorithm*).

1. INTRODUÇÃO

Sistemas Imunológicos Artificiais (*Artificial Immune Systems* – AISs) representam uma classe de Meta-heurísticas Populacionais (MhPs) que, diferentemente dos métodos tradicionais inspirados na evolução das espécies, como os Algoritmos Evolutivos (AEs), são baseados na microevolução do sistema imunológico dos organismos complexos.

As MhPs podem ser consideradas um tipo de busca aleatória guiada, que tenta explorar o espaço de busca como um todo e, paralelamente, procura também realizar uma busca local encontrando soluções sub-ótimas. Tal característica existe devido à abordagem de população, o que permite um alto paralelismo e, adicionalmente, confere robustez ao processo de busca obtendo várias soluções candidatas rapidamente [PAL *et al.*, 2006], um compromisso entre exploração (busca global) e exploração (busca local).

O CLONALG [CASTRO & VON ZUBEN, 2002], desenvolvido inicialmente para realizar reconhecimento de padrões, inspirado nas teorias da imunologia, demonstrou também a possibilidade da utilização de AISs em problemas de otimização [CASTRO & TIMMIS, 2002]. Utilizando as estratégias propostas no CLONALG, associado à teoria de redes imunológicas, foi desenvolvido o algoritmo de otimização contínua opt-aiNet [CASTRO & TIMMIS, 2003].

Uma característica interessante observada no opt-aiNet é a inserção de soluções aleatórias (imigrantes aleatórios) quando o *fitness* populacional médio converge com o passar das gerações. Esta técnica permite o aumento da diversidade da população para melhor explorar o espaço de busca. Adicionalmente, buscando evitar a exploração excessiva de soluções já conhecidas, o método utiliza um mecanismo de supressão, que elimina indivíduos semelhantes. A supressão é realizada pelo cálculo da distância euclidiana entre todos os possíveis pares de soluções, descartando aquelas com menor *fitness* no caso de soluções com distância menor que um dado limiar.

A técnica de supressão e adição de imigrantes aleatórios é determinista e, portanto, dependente dos parâmetros utilizados, além de ser custoso computacionalmente calcular a distância entre cada par de soluções. Neste trabalho, foi desenvolvida uma nova técnica de supressão em substituição àquela utilizada no opt-aiNet.

Inspirado no algoritmo SORIGA (*Self-Organizing Random Imigrants Genetic Algorithm*) [TINÓS & YANG, 2007], a técnica proposta seleciona aleatoriamente uma solução da população para ser substituída por uma solução aleatória. Esta, em seguida, seleciona mais duas que, se possuírem *fitness* suficientemente semelhantes são também marcadas para supressão, e assim sucessivamente. Desta forma, cada célula marcada para supressão seleciona recursivamente mais duas soluções aleatórias até que não se encontrem soluções com *fitness* semelhante e, portanto, não haja mais soluções selecionadas para supressão. Esta solução simples pode levar a um comportamento auto-organizável, que permite que mais ou menos indivíduos sejam substituídos de acordo com a diversidade da população, sem que seja necessário um controle explícito desta e diminuindo assim a sensibilidade ao ajuste de parâmetros, como o número de soluções substituídas.

Neste trabalho é ainda proposta a aplicação de um operador de mutação com distribuição q-gaussiana em substituição àquele com distribuição normal. O interesse de sua utilização é devido à possibilidade, através do parâmetro q , de ajustar a forma da distribuição das mutações, permitindo controlar a severidade das mutações. O operador de mutação aplicado nos AISs é baseado na proposta de Tinós & Yang [2010], que utiliza uma técnica com q auto-adaptável. Esta característica é interessante pois nem sempre é adequado utilizar para diferentes problemas e em diferentes estágios do processo de otimização uma mesma forma de distribuição de mutação (*e.g.*, a gaussiana). A mutação q-gaussiana permite que diferentes tipos de distribuição de mutação sejam reproduzidas durante o processo de busca.

Uma série de problemas em bioinformática podem ser abordados através do uso de MhPs, o que justifica o interesse em sua aplicação na área [FOGEL & CORNE, 2003; PAL *et al.*, 2006]. Aplicações podem ser das mais variadas como a filogenia, que busca encontrar relações ancestrais entre diferentes espécies a fim de estudar o histórico evolutivo; a classificação de funções de proteínas que, a partir de informações extraídas da sequência de aminoácidos de uma proteína, busca definir sua função; a predição de estrutura tridimensional de proteínas que, de posse de uma sequência de aminoácidos, procura prever sua estrutura tridimensional (estrutura terciária), o que deve permitir o estudo de sua função; e o *docking*, abordado neste trabalho, que busca prever a conformação tridimensional de complexos moleculares (estrutura quaternária). Este

problema pode ser visto como um problema de otimização desafiador que requer a robustez de MhPs, como o AIS.

Embora existam outros algoritmos de *docking* com funções de *score* mais sofisticadas, neste trabalho foi utilizada uma função baseada em complementaridade pela simplicidade de sua implementação e também como um esforço inicial no desenvolvimento do primeiro passo do *docking* como proposto em [SMITH & STERNBERG, 2002].

1.1. OBJETIVOS

A proposta deste trabalho é o desenvolvimento de um MhP baseado na implementação de Sistemas Imunológicos Artificiais descrito em [CASTRO & TIMMIS, 2002], o opt-aiNet, em conjunto com os princípios de auto-organização inspirados nos conceitos relacionados ao fenômeno de criticalidade auto-organizada de Bak [1997]. Espera-se que a técnica auto-organizável implementada, primeiramente, diminua a influência da escolha de parâmetros referentes à etapa de supressão no desempenho do processo de otimização, e em seguida, que o algoritmo implementado diminua a complexidade do procedimento de supressão do opt-aiNet. Foi realizado em seguida um estudo para sua aplicação no problema de *docking* rígido de proteínas.

A fim de validação, os algoritmos desenvolvidos, chamados SOCOpt-aiNet e SOCAIN, foram avaliados e comparados com o opt-aiNet, utilizando as funções de *benchmark* descritas em [SUGANTHAN *et al.*, 2005], um trabalho muito conhecido na comunidade de MhPs que fornece uma metodologia para a avaliação e comparação de algoritmos de otimização. Verificados seus desempenhos na otimização de funções multimodais, o algoritmo foi aplicado no *docking* de proteínas.

1.2. ORGANIZAÇÃO DO DOCUMENTO

Este documento está organizado da seguinte forma: no Capítulo 2 é descrito o conceito de criticalidade auto-organizada e sua inspiração para a utilização em AEs; no Capítulo 3 são apresentados os conceitos envolvendo AISs e o algoritmo opt-aiNet; o Capítulo 4 apresenta algumas aplicações de AEs em bioinformática e descreve em detalhes o problema de *docking*; no Capítulo 5 é detalhada a metodologia utilizada no desenvolvimento dos algoritmos propostos e na função de avaliação do *docking*; o Capítulo 6 contém a discussão dos resultados obtidos; o Capítulo 7 apresenta as conclusões do trabalho e; o Capítulo 8 as referências bibliográficas.

2. ALGORITMOS EVOLUTIVOS E CRITICALIDADE AUTO-ORGANIZADA

A criticalidade auto-organizada (do inglês, *self-organizing criticality* - SOC) e a aplicação de seus princípios no controle de parâmetros em algoritmos evolutivos (AEs) são descritos neste capítulo.

2.1. CRITICALIDADE AUTO-ORGANIZADA

Sistemas com diversos componentes interagentes podem apresentar um comportamento auto-organizável quando encontrados em um estado crítico [BAK, 1997]. O sistema tende a se auto-organizar a um ponto crítico em resposta a estímulos externos. No estado crítico o sistema é capaz de, com uma pequena perturbação, causar uma reação em cadeia atingindo vários componentes, organizando-o em um estado estável sem necessidade de controle externo. As distribuições estatísticas da duração da reação em cadeia e o número de componentes afetados são governadas por leis de potência [BAK, 1997] nas formas:

$$P(s) \sim s^{-\tau} \quad , \quad P(d) \sim d^{-\alpha} \quad (1)$$

sendo s o número de componentes afetados pela perturbação, d a duração da reação em cadeia, e τ e α constantes reais positivas.

Um exemplo de sistema que apresenta este comportamento auto-organizável é o modelo de pilhas de areia descrito em [BAK, 1997], que considera um autômato onde cada célula representa uma pilha de grãos de areia. Em cada instante de tempo t , um grão é adicionado a uma posição aleatória da grade. No momento em que uma pilha recebe grãos suficientes para passar de um determinado limiar (em geral 3), estes são distribuídos entre as células vizinhas (vizinhança de *von Neumann*), evento chamado de avalanche. No caso em que uma célula recebe um grão decorrente de uma avalanche, e também ultrapassa o limiar, este também dividirá os grãos com suas células vizinhas aumentando o tamanho da avalanche. Este processo ocorre recursivamente até que todas as pilhas estejam estáveis (possuam um número de grãos abaixo do limiar).

Outro modelo, embora extremamente simplista, é o de interação de espécies, que indica que o processo de co-evolução pode apresentar SOC. Neste caso, a extinção de uma espécie, por exemplo, uma presa, pode afetar a sobrevivência das espécies que a predam, que por sua vez afeta as dos predadores desta, o que pode levar a uma cascata

de extinções. O modelo simplificado considera um vetor de valores reais representando as espécies e seus valores de *fitness*, cada uma interagindo com as duas adjacentes no vetor. No instante inicial da simulação, cada espécie possui um valor de *fitness* gerado aleatoriamente. Em cada instante t , a espécie com o menor valor de *fitness* e as duas adjacentes são substituídas por valores aleatórios. O evento de substituição dos valores de *fitness* é chamado extinção.

A Figura 1 apresenta um gráfico radial em três instantes de tempo de uma simulação do modelo de interação de espécies simulado neste trabalho, como proposto em [BAK, 1997]. Cada linha radial representa um indivíduo e, em azul, seu *fitness*. No gráfico, existem três estágios: o inicial (A), onde as espécies possuem *fitness* aleatórios, um em que as espécies se encontram próximo de um estado crítico (B), e um após uma extinção de tamanho oito (C). Observa-se, neste caso, a extinção de várias espécies, vizinhas entre si, decorrente da extinção de uma única espécie.

Figura 1: Exemplo do desenvolvimento do modelo de interação de espécies.

No modelo de pilhas de areia, o estado crítico ocorre quando várias pilhas estão próximas do limiar. Neste caso, uma pequena perturbação é mais propícia a causar uma grande avalanche. De forma análoga, no modelo de interação de espécies, com o passar do tempo os valores de *fitness* tendem a ficar com valores próximos e altos. No estado crítico é mais provável que as espécies substituídas apresentem valores menores de *fitness* que o restante. Desta forma, em instantes de tempo seguintes, é mais provável que as novas substituições ocorram nas posições onde ocorreram anteriormente, aumentando assim o tamanho e a duração das extinções. Repare que este é um processo auto-organizado. Quando a diversidade da população é alta, (*e.g.* como no início da simulação) pequenas extinções (entre espécies vizinhas) ocorrem com frequência. Entretanto, quando a diversidade é baixa (no estado crítico) a probabilidade de ocorrência de grandes extinções é maior.

2.2. ALGORITMOS EVOLUTIVOS

AEs são meta-heurísticas populacionais (MhPs) aplicadas tipicamente à busca e otimização de funções [MITCHELL, 1996]. Sua utilização é especialmente interessante em casos onde o conhecimento sobre o problema é limitado, ou em que o espaço de busca se altera com o tempo de forma imprevisível. Existem inúmeros MhPs com diferentes inspirações. A mais utilizada é inspirada, provavelmente, nos processos adaptativos descritos pelas teorias de Darwin sobre a evolução biológica, que são os AEs. Dentre os algoritmos que seguem tal inspiração, os Algoritmos Genéticos (AGs), proposto por Holland [1975], são um dos mais populares. O AG padrão utiliza operadores inspirados no processo de seleção natural, como a seleção dependente do *fitness*, mutação e cruzamento.

As MhPs consideram um conjunto de soluções candidatas para a função que se deseja otimizar, chamada neste contexto de função de *fitness* (ou de avaliação), representando elementos de uma população. Estes interagem entre si para gerar novas soluções, potencialmente mais promissoras. A representação das soluções candidatas para o problema que se deseja otimizar é chamada codificação. Soluções para funções são comumente representadas por um (ou mais, em alguns problemas específicos) vetor de valores, que são tipicamente binários, mas são comuns implementações reais, inteiras ou até mesmo, em casos especiais, caracteres ou tipos mistos. A codificação deve ainda definir o espaço de busca, limitando o conjunto dos valores possíveis que cada entrada da função pode assumir.

No caso dos AGs, as soluções candidatas representam indivíduos de uma população, cada um com seu cromossomo representando o conjunto de valores de entrada da função de acordo com a codificação. A saída da função de *fitness* é chamada de valor de *fitness* (ou aptidão).

O operador de seleção é utilizado para selecionar indivíduos para herdarem suas características para compor as soluções da próxima geração. O operador de seleção implementado pelo AG padrão é conhecido como método da roleta (*roulette wheel*, em inglês), que seleciona um indivíduo com probabilidade dada pela razão do valor do *fitness* do indivíduo com o somatório do *fitness* populacional. Desta forma a chance de seleção de um indivíduo é proporcional ao seu valor de *fitness* (inversamente proporcional no caso de minimização).

A reprodução é aplicada para gerar novos indivíduos a partir dos indivíduos obtidos pelo operador de seleção. No AG padrão, os operadores de reprodução utilizados são o *crossover* e a mutação [GOLDBERG, 1989]. No primeiro, são obtidos dois indivíduos pelo operador de seleção, e estes têm alguns trechos de cromossomo trocados. A chance de utilizar o *crossover* para a geração de novos indivíduos é definida por uma taxa p_c , chamada de taxa de *crossover*. No segundo, cada indivíduo selecionado tem alguns de seus genes alterados aleatoriamente. O número de genes alterados por mutação é definido através de uma taxa p_m , chamada de taxa de mutação.

O pseudo-código do AG padrão é apresentado no Quadro 1. Após sucessivas gerações, o *fitness* médio da população tende a melhorar, sendo que uma solução é retornada quando um critério de convergência é alcançado.

Quadro 1: Pseudo-código do AG padrão

```
1 Início
2   inicialize a população
3   avalie a população inicial
4   enquanto ( critério de convergência não satisfeito ) faça
5       selecione indivíduos para a nova população
6       aplique mutação e cruzamento nos indivíduos selecionados
7       avalie os indivíduos da nova população
8   fim enquanto
9   Fim
```

Um problema muito comum em AEs é a convergência prematura da população para ótimos locais, o que torna difícil nestas situações, com os operadores padrão, explorar melhor o espaço de soluções. Desta forma, é importante que haja um operador que controle a diversidade populacional para permitir uma busca mais ampla pelo espaço de soluções [EIBEN *et al.*, 1999; VAVAK & FOGARTY, 1996; COBB & GREFENSTETTE, 1993].

Uma forma simples de aumentar a diversidade em AGs é a inserção de imigrantes aleatórios na população [COBB & GREFENSTETTE, 1993]. Neste caso, alguns indivíduos da população são substituídos por novos gerados aleatoriamente a cada geração. Porém, é comum que indivíduos gerados aleatoriamente possuam *fitness* ruim, quando comparados ao restante da população. Portanto, quando a população em geral converge para um ótimo local, o operador de seleção tende a manter os indivíduos mais aptos, eliminando os imigrantes e, assim, dificultando a exploração do espaço de busca pelos novos indivíduos. Desta forma, a efetividade deste método é dependente do

número de indivíduos substituídos e da política de substituição (e.g. indivíduos de pior *fitness*).

2.3. CRITICALIDADE AUTO-ORGANIZADA APLICADA A ALGORITMOS EVOLUTIVOS

Os autores de [TINÓS & YANG, 2007] propõem um método auto-organizável de operador de imigrantes aleatórios inspirado no modelo de interação de espécies para AGs, chamado SORIGA (*Self-Organizing Random Immigrants Genetic Algorithm*). Neste, considera-se que um indivíduo representa uma espécie do modelo de interação entre espécies de [BAK, 1997]. Desta forma, a população do AG fica organizada em um vetor circular, onde cada indivíduo possui uma interação interdependente com os dois vizinhos, de forma análoga ao modelo descrito anteriormente, no capítulo 2.1.

Quadro 2: Pseudo-código do SORIGA adaptado de [TINÓS & YANG, 2007].

```
1  Início
2      selecione indivíduo de índice  $i$  com menor fitness
3      substitua indivíduos  $x_{i-1}$ ,  $x_i$  e  $x_{i+1}$  por imigrantes aleatórios
4
5      se  $i \geq i_{min}$  e  $i \leq i_{max}$  então
6           $ext \leftarrow ext+1$ 
7
8          se  $i = i_{min}$  ou  $i = i_{max}$  então
9              atualize  $i_{min}$  e  $i_{max}$ 
10         fim se
11     senão
12          $i_{min} \leftarrow i-1$ 
13          $i_{max} \leftarrow i+1$ 
14          $ext \leftarrow 1$ 
15     fim se
16 Fim
```

O Quadro 2 apresenta o pseudo-código do método de substituição utilizado no SORIGA. Em cada geração do AG, o indivíduo de índice i , de pior *fitness*, e seus vizinhos adjacentes, de índices $i-1$ e $i+1$ no vetor circular, são substituídos por imigrantes aleatórios. Para que os imigrantes não sejam eliminados pelo operador de seleção, estes são mantidos em uma subpopulação isolada contendo os indivíduos extintos. O tamanho da subpopulação cresce quando o indivíduo com pior *fitness* é de índice i_{min} ou i_{max} , índices dos indivíduos das extremidades da subpopulação. A duração das extinções, ext , são computadas como o número de gerações em que o indivíduo i de menor *fitness* encontra-se na subpopulação. No caso em que o indivíduo de pior *fitness*

não pertença à subpopulação de imigrantes, então a subpopulação e a duração são reiniciados.

O isolamento dos indivíduos extintos é feito através de uma modificação no operador de seleção. Considerando que a cada geração do AG são gerados N indivíduos, na inserção do indivíduo de índice i na nova população, se i representar o índice de um imigrante na subpopulação corrente (*i.e.*, se $i_{min} \leq i \leq i_{max}$), então i será selecionado através do operador de seleção utilizado, dentre os indivíduos somente desta subpopulação. Senão, o indivíduo i deve ser selecionado de acordo com o método de seleção padrão, considerando a população como um todo. Este método isola os imigrantes aleatórios e seus descendentes de forma que, indivíduos dentro da subpopulação, somente interagem entre eles, não recebendo genes dos indivíduos externos a ela, mantendo a variabilidade fornecida pelos imigrantes aleatórios. Uma vantagem deste método é que não é necessário determinar o número de indivíduos extintos, podendo o tamanho da subpopulação variar ao longo das gerações.

Em outro exemplo de AE inspirado em sistemas que apresentam SOC, [FERNANDES *et al.*, 2008] descreve o *Sandpile Mutation*, um operador de mutação para AGs inspirado no modelo de pilhas de areias. O autômato, que representa as pilhas de areia, é uma grade de tamanho N por N_D , sendo N o número de indivíduos e N_D o tamanho do cromossomo. Desta forma, pode-se dizer que cada gene possui uma pilha de areia associada.

A cada geração g grãos são lançados em posições aleatórias da grade e as pilhas são atualizadas recursivamente. A mutação ocorre com probabilidade igual a $1 - f_{norm}(\mathbf{x}_i)$ quando uma pilha alcança o limiar, onde $f_{norm}(\mathbf{x}_i)$ é o *fitness* normalizado na forma

$$f_{norm}(\mathbf{x}_i) = \frac{f(\mathbf{x}_i) - \min_{j=1:N}(f(\mathbf{x}_j))}{\max_{j=1:N}(f(\mathbf{x}_j)) - \min_{j=1:N}(f(\mathbf{x}_j))} \quad (2)$$

sendo *min* o mínimo e *max* o máximo *fitness* da população corrente. A equação 2 considera um problema de maximização. Nos casos de minimização, deve-se utilizar $f_{norm_{min}}(\mathbf{x}_i) = 1 - f_{norm}(\mathbf{x}_i)$. Caso a mutação não ocorra devido ao teste de probabilidade, o grão é descartado. Isto evita que grandes alterações ocorram em indivíduos com melhores *fitness*. Uma vez ocorrida a mutação, aquela pilha é considerada “ativa”, para discernir genes já mutados, evitando que ocorram diversas

mutações no mesmo gene. Para que não haja um período inicial em que não ocorrem mutações, as pilhas de areia são iniciadas com tamanhos aleatórios abaixo do limiar. No caso de uma mutação em um gene na borda da grade, considera-se que alguns grãos são descartados para fora da grade. O algoritmo do *Sandpile Mutation* é apresentado em forma de pseudo-código no Quadro 3.

Quadro 3: Pseudo-código do *Sandpile Mutation* [FERNANDES *et al.*, 2008]

```
1  Início
2  para g grãos faça
3      lance um grão em posição aleatória (i, j) do autômato
4      calcule  $f_n(x_i)$ 
5
6      se  $\text{rand}(0, 1) > f_n(x_i)$  e posição(i, j) não ativa
7          mute gene j de  $x_i$ 
8          compute avalanche
9          atualize recursivamente toda pilha
10  fim para
11  Fim
```

Os métodos descritos anteriormente [TINÓS & YANG, 2007; FERNANDES *et al.*, 2008] são técnicas auto-organizáveis de manutenção de diversidade em algoritmos evolutivos. Nestes modelos é comum a ocorrência de pequenas alterações na população em geral. Porém, em raros eventos, quando atingido o ponto crítico, ocorrem grandes alterações na população levando a um aumento na diversidade.

No SORIGA o ponto crítico ocorre quando a população em geral converge para um ótimo local, onde cada indivíduo possui um *fitness* próximo da média geral, de forma semelhante ao modelo de interação de espécies descrito em [BAK, 1997]. Neste as melhores soluções devem ser mantidas por métodos elitistas para que não sejam substituídas por imigrantes aleatórios. No *Sandpile Mutation* grandes alterações ocorrem quando existem várias pilhas com valores próximos do limiar. Porém, um fator adicional permite a manutenção das melhores soluções, onde soluções com melhores *fitness* possuem menor probabilidade de serem mutadas.

Já Krink *et al.* [2000] descreve operadores de mutação e seleção baseados em distribuições de leis de potência. Este utiliza o modelo de pilhas de areia para a geração de valores aleatórios com tal distribuição. O operador de mutação utiliza da distribuição em lei de potência para definir a variância da mutação. Inicialmente um teste de probabilidade é feito para verificar se a mutação deve ou não ocorrer em um determinado indivíduo. Caso o teste de probabilidade seja bem sucedido, a mutação

ocorre em todos os genes com distribuição gaussiana com variância definida pelo valor da distribuição de lei de potência multiplicado por uma constante de escala.

O operador de seleção utiliza também a distribuição em lei de potência, desta vez para obter uma taxa de extinção. Esta taxa é definida por $ext_size = \frac{sN}{100}$, sendo s (equação 1) obtido pelo tamanho da avalanche do modelo de pilhas de areia, e N o tamanho da população. Em cada geração, ext_size indivíduos são selecionados da população corrente, que são substituídos por versões mutadas dos indivíduos remanescentes. A seleção é feita aleatoriamente e a mutação é realizada com distribuição gaussiana com parâmetros fixos.

Em outro trabalho, Krink & Thomsen [2001] apresentam um modelo de extinção em massa auto-organizável que utiliza um valor aleatório com distribuição em lei de potência para definir um número de regiões de extinção. Este valor é obtido utilizando os tamanhos de avalanches do modelo de pilhas de areia.

O modelo de extinção em massa auto-organizável considera que os indivíduos do AE estão organizados em uma grade de tamanho $N_A \times N_B$. Em cada geração, um valor obtido da distribuição em lei de potência é utilizado para definir o número de regiões de extinção. Estas são regiões de tamanho 3×3 com centro em uma posição selecionada aleatoriamente.

Os indivíduos dentro das regiões de extinção são substituídos por versões mutadas do melhor indivíduo. No caso em que o melhor indivíduo pertença à uma zona de extinção, este não é substituído. Por utilizar distribuições em lei de potência, este método por diversas vezes substitui um pequeno número de indivíduos, porém em eventos raros ocorrem um grande número de zonas de extinção substituindo grande parte da população.

3. SISTEMAS IMUNOLÓGICOS ARTIFICIAIS EM OTIMIZAÇÃO CONTÍNUA

AISs são um tipo de MhP inspirados nas teorias do sistema imunológico adaptativo. Os autores em [CASTRO & TIMMIS, 2002] descrevem um AIS utilizado para a otimização de funções, chamado opt-aiNet. Este algoritmo funciona com base em três mecanismos das teorias da imunologia: seleção clonal, maturação de afinidade e redes imunológicas.

O sistema imunológico adaptativo, presente em organismos complexos, como o homem, de forma simplista age reconhecendo certas estruturas moleculares das membranas de microorganismos invasores, como as bactérias infecciosas. Tais estruturas, no contexto da imunologia, são conhecidas como antígenos. O reconhecimento destes é feito através de um tipo de proteína, chamada anticorpo, secretada pelas células do sistema imunológico. Os anticorpos agem interagindo com os antígenos, inibindo-os. Devido à interação entre proteínas ocorrer de forma específica, como um mecanismo de “chave e fechadura”, associado ao fato de, em geral, agentes externos serem desconhecidos ao sistema imunológico do organismo invadido, o processo de reconhecimento dos anticorpos deve passar por uma fase de adaptação [CASTRO & VON ZUBEN, 2002].

Considerando que as células se reproduzem por mitose, criando duas células idênticas, o princípio de seleção clonal busca selecionar as células mais aptas a reconhecer os antígenos, gerando assim um maior número de clones destas.

A adaptação, de fato, ocorre em um processo conhecido como maturação de afinidade. Na clonagem das células do sistema imunológico, ocorrem mutações adaptativas, de forma que células mais aptas a reconhecerem os antígenos sejam reproduzidas com menos alterações, e células menos aptas são mutadas mais intensamente. Aquelas que possuem maiores afinidades, eventualmente se diferenciam em células de memória, o que permite uma resposta imunológica mais rápida em caso de uma segunda invasão de um mesmo tipo de microorganismo.

A teoria de redes imunológicas indica que no sistema imunológico podem ocorrer interações entre células do próprio organismo invadido, inclusive de forma inibitória, por exemplo, para controle da proliferação, formando uma rede de células [CASTRO & TIMMIS, 2003].

Quadro 4: Pseudo-código do opt-aiNet [CASTRO & TIMMIS, 2002].

```

1  Início
2      gere população inicial aleatoriamente
3      enquanto critério de convergência não satisfeito faça
4          determine fitness de cada célula de memória
5          para cada célula de memória faça
6              gere  $N_c$  clones mutados proporcional ao fitness
7              determine fitness de cada clone
8              diferencie clone com melhor fitness para célula
                                                de memória
9          fim para
10         se fitness médio não alterou significativamente então
11             determine afinidade para cada par de células
12             suprima células com afinidade acima do limiar  $\sigma_s$ 
13             introduza d% de células geradas aleatoriamente
14         fim se
15     fim enquanto
16 Fim
    
```

O Quadro 4 apresenta o pseudo-código do opt-aiNet. Nele, as células de memória são as soluções candidatas. O princípio de seleção clonal é aplicado na geração dos N_c clones mutados de cada célula de memória. Porém, este princípio é aplicado de forma simplificada, uma vez que todas as células produzem o mesmo número de clones.

A maturação de afinidade é aplicada na mutação e seleção do clone com melhor *fitness*. A mutação é realizada em todos os genes da célula, adicionando um valor aleatório com distribuição gaussiana de média zero, na forma:

$$\mathbf{c}' = \mathbf{c} + \alpha \text{Rand}_g(0,1) \quad (3)$$

sendo $\mathbf{c}'$ a célula mutada, $\mathbf{c}$ a célula clone original, $\text{Rand}_g(0, 1)$ um vetor de números aleatórios com distribuição gaussiana de média zero e desvio padrão $\sigma = 1$ e

$$\alpha = \left(\frac{1}{\beta}\right) \exp(-f_n(\mathbf{c})) \quad (4)$$

onde β é um parâmetro que controla o decaimento da exponencial inversa e f_n o *fitness* normalizado como descrito na equação 2. Repare que, de acordo com a equação 4, na equação 3, quanto pior o desempenho da solução candidata, maior será a variância do componente aleatório incrementado na mutação.

No opt-aiNet, a teoria de redes imunológicas é aplicada na fase de supressão. Esta etapa é realizada na detecção de estagnação no *fitness* médio da população, que ocorre na seguinte condição:

$$\frac{|\bar{f} - \bar{f}_0|}{|\bar{f}_0|} < l_c \quad (5)$$

onde $\bar{f}$ é a média da geração atual, $\bar{f}_0$ é a média dos *fitness* médios de cada geração das últimas g_c gerações, e l_c o limiar de convergência para que ocorra a supressão. A verificação da condição em (5) ocorre em intervalos de g_c gerações.

Na satisfação da condição em (5), é determinada a afinidade entre cada par de células de memória. O conceito de afinidade entre as células de memórias remete ao conceito de similaridade entre as células. O algoritmo padrão, porém, utiliza a distância euclidiana para tanto, um critério de dissimilaridade. Desta forma, as células suprimidas são aquelas que possuem a distância abaixo de um limiar. No caso do uso de um critério de similaridade a afinidade deve ter um valor maior que o limiar.

Ao final do processo de supressão, uma porcentagem d de células geradas aleatoriamente são adicionadas às células de memória.

Tabela 1: Comparativo entre AIS e AG [CASTRO & TIMMIS, 2002].

	Sistema Imunológico Artificial (AIS)	Algoritmo Genético (AG)
Solução candidata	<i>Célula do sistema imunológico</i>	<i>Indivíduo da população</i>
Reprodução	<i>Clonagem</i>	<i>Crossing-over</i>
Mutação	<i>Ocorre em todos os genes da célula com distribuição normal proporcional ao fitness</i>	<i>Ocorre com determinada probabilidade p_m, geralmente baixa</i>
Codificação	<i>Real</i>	<i>Tradicionalmente binária.</i>
Evolução	<i>Teorias do sistema imunológico adaptativo inspira o desenvolvimento</i>	<i>Teoria evolutiva inspira o desenvolvimento</i>

Nota-se uma série de similaridades do AIS com o AG. Ambos partem de soluções aleatórias e, então, através de processos de reprodução, seleção e mutação, buscam evoluir as soluções candidatas para a otimização de uma função. A Tabela 1 apresenta uma comparativo com as principais diferenças entre os algoritmos.

Uma das principais diferenças entre os algoritmos é a etapa de supressão que ocorre nos AISs. Nesta etapa o algoritmo é capaz de suprimir soluções semelhantes, eliminando buscas desnecessárias em regiões próximas. A adição de uma porcentagem de soluções aleatórias (equivalente aos imigrantes aleatórios em AGs) em seguida permite um controle do tamanho da população.

4. DOCKING MOLECULAR

A bioinformática trata do desenvolvimento e aplicação de métodos computacionais para a extração e análise de informações de dados biológicos. É uma área de conhecimento multidisciplinar que envolve conhecimento em biologia, computação, matemática e estatística [PAL *et al.*, 2006].

Embora possam existir variações, problemas em bioinformática são frequentemente relacionados a estudos de sequências biológicas, como o DNA (ácido desoxirribonucleico), RNA (ácido ribonucleico) e proteínas. Desta forma, a pesquisa em bioinformática envolve, como exemplos, a filogenia, que investiga o relacionamento ancestral entre diferentes espécies com base, por exemplo, em similaridade nas sequências de DNA; a predição de estrutura tridimensional de proteínas, e de complexos proteicos, que tem como objetivo realizar a predição da conformação da cadeia de aminoácidos em proteínas; o reconhecimento de genes e regiões promotoras em sequências de DNA; a classificação de função de proteínas; entre outras. A referência [FOGEL & CORNE, 2003] apresenta uma série de problemas de bioinformática tratando especialmente da aplicação de AEs na área.

Aqui, estamos interessados em aplicar um AIS modificado no problema de *docking* de proteínas. Neste problema busca-se predizer a estrutura quaternária de proteínas, isto é, a conformação de diferentes proteínas interagindo formando um complexo proteico.

Moléculas, quando se aproximam, possuem interações energéticas que podem levar, em situações favoráveis, à ligação entre elas [PAL *et al.*, 2006]. O *docking* é a atividade de modelar o processo de ligação de duas moléculas (uma receptora e uma ligante), e predizer a estrutura do complexo (receptor-ligante).

Um dos mais importantes tipos de moléculas estudadas em *docking* são as proteínas. Sua importância é devida à sua presença em grande parte dos processos biológicos nos mais diferentes organismos (*e.g.*, nas sinapses entre neurônios para a recepção de neurotransmissores, na divisão celular separando a fita dupla de DNA e replicando novas fitas).

As interações entre moléculas são extremamente fracas, quando comparadas com às ligações covalentes entre os átomos de uma molécula. Desta forma, para que as ligações entre moléculas sejam fortes, estas devem ocorrer de forma específica,

maximizando o número de interações. Na modelagem do *docking* utilizando métodos computacionais, a avaliação da especificidade da ligação é feita através de funções de *score*. Um exemplo, provavelmente o mais simples, se tratando de *docking*, é a complementaridade das superfícies das moléculas, onde se busca maximizar a superfície de contato entre elas [SMITH & STERNBERG, 2002]. Outras, mais complexas, buscam utilizar de campos de força e outras características energéticas das moléculas, como as interações de van der Waals, ligações de hidrogênio ou outras interações eletrostáticas [AZEVEDO JR. & DIAS, 2008; HEBERLÉ & AZEVEDO JR., 2011; KITCHEN *et al.*, 2004; SMITH & STERNBERG, 2002]. A Figura 2 exemplifica duas estruturas moleculares ligadas, neste caso, um complexo proteico.

Figura 2: Complexo (em azul) alfa-quimotripsina A (de *Bos taurus*) – (em verde) inibidor ovomucóide (de *Meleagris gallopavo*), código pdb 1CHO. As proteínas em representação em fita e em transparência a superfície excluída ao solvente.

A natureza da especificidade exigida para a ligação das moléculas requer, como consequência, uma especificidade estrutural delas. Desta forma, o conhecimento das estruturas tridimensionais das moléculas ou, no caso do *docking*, do complexo molecular, é importante para o estudo de suas funções, sendo essencial no domínio da bioquímica [SMITH & STERNBERG, 2002] para o entendimento dos microprocessos que regem o funcionamento dos organismos. De conhecimento da estrutura molecular

isolada, um modelo ideal de *docking* deve permitir ainda apoiar os estágios iniciais do desenvolvimento de drogas, possibilitando verificar se uma determinada droga é capaz de se ligar com sucesso, por exemplo, a uma proteína receptora, previamente à realização de experimentos *in vitro* [AZEVEDO JR. & DIAS, 2008; GARDINER *et al.*, 2001; HEBERLÉ & AZEVEDO JR., 2011; KITCHEN *et al.*, 2004; PAL *et al.*, 2006].

Os autores de [SMITH & STERNBERG, 2002] descrevem o problema de *docking* de uma forma genérica dividindo o processo em três estágios, nesta ordem: busca global, *rescoring* e introdução de flexibilidade; como visto na Figura 3.

Figura 3: Estágios do *docking* adaptado de [SMITH & STERNBERG, 2002].

No primeiro estágio é realizado o *docking* rígido das moléculas, possivelmente considerando uma tolerância nas superfícies, utilizando uma função de *score* computacionalmente simples. Desta forma a busca é realizada em um espaço de seis dimensões, considerando somente posições e orientações da molécula ligante referente à molécula receptora. Esta foi a etapa estudada neste trabalho.

A etapa de busca global tem como objetivo obter um conjunto de posições e orientações candidatas mais promissoras para análise através de uma função de *score* mais completa, em geral mais computacionalmente custosa, que seja capaz de melhor discriminar soluções mais próximas da real. Este é o estágio de *rescoring* que deve fazer uma análise mais apurada dos complexos formados na primeira etapa eliminando soluções menos promissoras. No primeiro estágio em geral é utilizado uma abordagem baseada em complementaridade de superfícies, enquanto no segundo, adicionam-se componentes de interações eletrostáticas e energias de ligações de hidrogênio, por exemplo.

No estágio final, introdução de flexibilidade, é realizada uma avaliação através de uma função de *score* ainda mais detalhada, considerando um modelo *full-atom* e

adicionando flexibilidade nas cadeias laterais ou, em modelos mais realistas, na cadeia principal.

Os passos 2 e 3 podem ser realizados em conjunto e podem ser interessantes caso a função utilizada no passo 1 tenha uma baixa taxa de falsos positivos. Adicionalmente, como visto na Figura 3, pode-se ainda adicionar conhecimento experimental entre os estágios, buscando, por exemplo, limitar o espaço de busca através da adição de informações referentes a posição do sítio de ligação.

Note que em cada estágio, a função de *score* torna-se mais detalhada buscando permitir uma melhor discriminação de conformações mais próximas da nativa (conformação encontrada na natureza), através da adição de novos termos simulando diferentes tipos de interações (*e.g.*, energia de ligações de hidrogênio, interações iônicas, interações hidrofóbicas) [BÖHM & STAHL, 2002] tornando a simulação mais realista. A obtenção de uma função mais específica tem como consequência, na maioria dos casos, um aumento do custo computacional. A separação do processo de *docking* em três estágios ordenados deve permitir uma limitação do espaço de busca a cada etapa, diminuindo o tempo computacional, realizando a computação do *score* detalhado da última etapa somente em soluções mais promissoras. Por outro lado, é importante citar que não é possível garantir que a busca global (primeira etapa) encontrará a conformação que levará ao complexo nativo nas demais etapas.

No processo de descoberta de drogas, o *docking* é uma parte do processo de *screening* [BÖHM & STAHL, 2002], que buscar reconhecer e selecionar, a partir de uma base de compostos químicos, um grupo de candidatos capazes de se ligar a uma determinada molécula alvo (*e.g.*, moléculas proteicas). O *docking* é utilizado para simular *in silico* a interação dos compostos com as moléculas alvo, previamente a realização de testes laboratoriais, que são afinal realizados para validar as hipóteses geradas nos experimentos computacionais e verificar como o ligante (composto) e receptor (molécula alvo) interagem no ambiente real. No *screening*, mesmo que sejam tratadas moléculas pequenas em geral, o *docking* ainda é a etapa mais custosa de todo o processo [BÖHM & STAHL, 2002].

O problema de *docking* pode ser classificado em função de sua flexibilidade: rígido ou flexível; ou ainda em função do estado inicial das moléculas: *bound* e *unbound*. Já as funções de *score*, utilizadas para avaliar as conformações dos complexos

no *docking*, podem ser funções: baseadas em campos de força, empíricas e baseadas em conhecimento [BÖHM & STAHL, 2002].

4.1. CLASSIFICAÇÕES DE PROBLEMAS DE DOCKING

O problema de *docking* pode ser classificado em função da flexibilidade das moléculas. Neste caso, o *docking* pode ser dividido em: rígido, o caso mais simples, onde considera-se que as moléculas têm conformação fixa e, portanto, não são permitidas modificações em suas estruturas (ângulos de ligação entre os aminoácidos no caso das proteínas, ou também na cadeia principal); e flexível, onde são permitidas mudanças na conformação das moléculas. Modelos flexíveis de *docking* tendem a ser mais realistas uma vez que é comum ocorrer mudanças de conformação nas moléculas quando elas se ligam.

No caso das proteínas, no *docking* flexível é comum a utilização de uma restrição nos ângulos de torção da cadeia principal (φ e ψ), utilizando somente um conjunto de ângulos possíveis [SMITH & STERNBERG, 2002]. Este conjunto é geralmente obtido empiricamente a partir de uma base de proteínas conhecidas. As possíveis combinações podem ser organizadas em um gráfico de φ por ψ que constituem o diagrama de Ramachandran.

Modelos de *docking* podem ainda ser classificados como *bound* e *unbound*. No primeiro, considera-se as moléculas em seus estados ligados. Estas são, então, separadas e passadas para o modelo com o objetivo de realizar o *docking*. Desta forma, esta abordagem exige que se conheça a estrutura do complexo, e devido a isso, é utilizado para testes nos estágios iniciais [GARDINER *et al.*, 2001]. Uma vez que as moléculas a serem ligadas são iniciadas em suas estruturas finais, modelos *bound* não são utilizados em problemas de *docking* flexível. Por outro lado, modelos *unbound* realizam a ligação das moléculas iniciando-se em seus estados nativos isolados. *Unbound docking* permite a realização de *docking* flexível e, portanto, para aplicações em problemas reais, como no caso do desenvolvimento de fármacos, citado anteriormente, é o mais estudado.

Os autores em [MORRIS *et al.*, 1998], embora seja uma classificação incomum atualmente, ainda descrevem os problemas de *docking* como métodos de *matching* ou de simulação. *Matching* são métodos baseados em complementaridade, onde se busca modelar geometricamente o sítio de ligação da molécula receptora (quando conhecido)

e tenta-se “encaixar” a molécula ligante na receptora de forma a ligar as estruturas de forma que suas superfícies sejam complementares. Este método em geral considera as moléculas como corpos rígidos. Métodos de simulação buscam simular o processo de *docking* em maior detalhe, considerando flexibilidade das moléculas, e utilizam funções de energia mais sofisticadas.

Neste trabalho, foi estudado a aplicação de AIS no *docking* rígido de proteínas. Entretanto, a solução pode ser também aplicada no problema do *docking* flexível através da implementação adequada de uma função de *score*.

4.2. TIPOS DE FUNÇÕES DE SCORE

As principais interações intermoleculares que regem a formação de um complexo ligante-receptor são: ligações de hidrogênio, que é a interação entre átomos de hidrogênio em uma molécula (doador de ligação de hidrogênio) com um átomo altamente eletronegativo (receptor de ligação de hidrogênio); interações iônicas, que ocorrem pela atração de grupos eletronegativos por grupos eletropositivos das moléculas ligante e receptora; interações hidrofóbicas, que ocorrem devido a tendência de regiões neutras das moléculas a evitarem contato com meio aquoso; interações cátion- π , que ocorrem devido ao quadrupolo gerado tipicamente por componentes aromáticos (ricos em elétrons) interagindo com cátions e; complexos metálicos, que ocorre na presença de um íon metálico que pode interagir com diversas moléculas ligantes em regiões polares [BÖHM & STAHL, 2002].

Funções de *score* podem ser baseadas em campos de força, empíricas e/ou baseadas em conhecimento [BÖHM & STAHL, 2002; FERRARA *et al.*, 2004; SOUSA *et al.*, 2006]. Funções de *score* baseadas em campos de força utilizam os princípios da mecânica molecular como uma aproximação da afinidade de ligação das moléculas. Desta forma, a energia livre de ligação é calculada como uma soma das energias de interações eletrostáticas e de van der Waals. AMBER [CASE *et al.*, 2005; WEINER & KOLLMAN, 1981] e CHARMM [BROOKS *et al.*, 1983; BROOKS *et al.*, 2009] são exemplos populares de funções baseadas em campos de força.

As funções empíricas utilizam medidas de afinidade de ligação obtidas experimentalmente para obter um valor médio utilizado para aproximar a energia de ligação de complexos desconhecidos. Desta forma, se a medida calculada para um determinado composto for próxima, ou menor que o valor médio experimental, então

pode se inferir que ocorreu uma boa aproximação da estrutura nativa. Um exemplo muito conhecido de algoritmo de *docking* que usa uma função essencialmente empírica é o AutoDock [MORRIS *et al.*, 1998; MORRIS *et al.*, 2009].

Funções baseadas em conhecimento buscam extrair informações estruturais de uma base de conformações tridimensionais (*e.g.*, PDB¹) de complexos conhecidos para construir funções de *score*. Estes tipos de funções se diferenciam das funções empíricas no sentido de que não dependem de medidas experimentais, mas somente extraem informações estatísticas de dados estruturais das moléculas. O DrugScore [VELEC *et al.*, 2005] é um modelo de função de *score* baseada em conhecimento.

Existem ainda implementações de funções que buscam combinar as características de diferentes tipos buscando melhorar a capacidade de discriminação ou mesmo diminuir o tempo de computação do *score*. Um exemplo, relativamente recente, de algoritmo que utiliza uma combinação de modelos, no caso, utilizando uma função com características empíricas e baseada em conhecimento, é o AutoDock Vina [TROTT & OLSON, 2010].

Os problemas de *docking* com funções baseadas em energia são problemas de minimização, uma vez que o complexo, em seu estado nativo, é considerado que está em seu estado de energia livre mínima. Neste trabalho, que investiga uma função baseada em complementaridade de superfícies, o problema é de maximização, dado que o objetivo é encontrar a conformação em que as superfícies de contato entre as moléculas são máximas.

4.3. COMPUTAÇÃO EVOLUTIVA APLICADA NO DOCKING MOLECULAR

O AutoDock [MORRIS *et al.*, 1998; MORRIS *et al.*, 2009], citado anteriormente no contexto de *docking*, utiliza uma AG Lamarckiano (AGL) que, em conjunto com os operadores tradicionais, inspirados nas teorias de Charles Darwin, utiliza um operador adaptativo inspirado na teoria evolutiva de Lamarck. Esta considera que os indivíduos, em suas interações com o ambiente, são capazes de se adaptar através de suas experiências. Estas adaptações são obtidas durante a vida do indivíduo e, de acordo com

¹ O PDB, do inglês *Protein Data Bank*, é uma base de pública de sequências e estruturas de proteínas e complexos proteicos disponibilizado em <http://www.pdb.org/> (visitado em 25/10/2011).

a teoria da herança dos caracteres adquiridos, são passados para seus filhos. Esta teoria foi a longo refutada, mas serviu de inspiração para o AGL utilizado pelo AutoDock.

Em [MORRIS *et al.*, 1998] o AGL utilizado codifica os indivíduos em função de três coordenadas cartesianas, representando a posição da molécula ligante no espaço, quatro ângulos para definir a orientação, evitando o *gimbal lock* nos ângulos de Euler², e um conjunto de ângulos definindo os ângulos de torção.

O algoritmo realiza uma etapa de mapeamento, onde através das informações dos genes (genótipo), gera a conformação do complexo (fenótipo). Só assim pode ser feito o cálculo da energia do complexo resultante.

O AGL realiza uma seleção proporcional de forma que o indivíduo com menor energia livre, possui maior probabilidade de ser selecionado e, portanto, de deixar maior número de descendentes.

A geração de novos indivíduos é realizada por um operador de *crossover* de dois pontos, e um operador de mutação que adiciona um valor aleatório com distribuição Cauchy. Esta possui, como na distribuição normal, maior probabilidade de gerar valores próximos do valor médio, gerando modificações pequenas no indivíduo. Porém, por ter caudas mais longas, existe uma probabilidade maior de gerar valores distantes da média quando comparada à distribuição normal [TINÓS & YANG, 2010].

Em geral operadores de mutação geram pequenas modificações nos indivíduos, realizando uma busca local. No AGL, mutações mais severas são possibilitadas, através da distribuição Cauchy. Isto ocorre devido a presença de um operador de busca local em uma porção da população ao final de cada geração [MORRIS *et al.* 1998].

A função de *score* da versão mais recente do AutoDock [MORRIS *et al.*, 2009] utiliza uma função semi-empírica separada em dois termos: um de energia intramolecular, que considera as mudanças energéticas das moléculas devido à flexibilidade, calculada considerando as moléculas como se estivessem desligadas; e uma intermolecular, que é a energia de ligação entre as moléculas.

Em [BOISSON *et al.*, 2008] são utilizados AGs no problema de *docking* molecular. A codificação utilizada é uma representação das duas moléculas, ligante e

² Neste caso são utilizados quatro ângulos para evitar o problema de *gimbal lock* nos ângulos de Euler, em que, quando dois eixos estão em conformação paralela devido a uma rotação, o sistema perde um grau de liberdade, limitando o sistema a rotações em duas dimensões.

receptora (*i.e.*, dois cromossomos). Desta forma, cada solução candidata é um par de vetores com as coordenadas (x , y e z) de cada átomo de ambas moléculas (ligante e receptor).

O algoritmo usa os operadores padrão (*i.e.*, cruzamento e mutação) adaptados ao problema. No caso do cruzamento, é realizada a troca dos vetores de coordenadas representando os ligantes entre as soluções candidatas. Usando este método é possível gerar conformações inválidas, como aquelas com colisões entre as moléculas. Porém, tais conformações são penalizadas por um termo de van der Waals na função de *score*, que causa um rápido aumento da energia com a aproximação dos átomos.

Três tipos de mutação são utilizados: um de orientação, um de posição e um de rotação de torção. As mutações de posição e orientação realizam alterações de caráter rígido no complexo, isto é, a molécula não muda sua conformação. Por outro lado, a rotação de torção provê flexibilidade às moléculas. O algoritmo pode utilizar uma política semi-flexível, que dá flexibilidade a somente uma das moléculas (em geral a ligante) ou completamente flexível, permitindo rotações dos ângulos de torção de ambas moléculas.

Uma característica interessante do algoritmo descrito em [BOISSON *et al.*, 2008] é a utilização de uma função de *score* multi-objetivo, com um componente energético, calculado como um somatório das interações de van der Waals, eletrostáticas e um termo de dessolvatação; e um componente de superfície, que busca minimizar a superfície acessível ao solvente.

No trabalho de Janson *et al.* [2007] é utilizado um algoritmo de otimização inspirado na organização de populações através de interações sociais (*e.g.*, revoadas de pássaros buscando alimento), chamado Otimização por Enxame de Partículas (em inglês *Particle Swarm Optimization* – PSO). Este algoritmo, como os AGs e AISs, são MhPs, porém cada solução candidata é uma partícula que faz um “passeio” pelo espaço de busca. A busca é orientada pelo histórico da própria partícula (melhor solução por ela obtida) e da partículas vizinhas (melhor posição, ou média, das partículas mais próximas).

Os autores de [JANSON *et al.*, 2007] utilizaram em seu trabalho uma função de *score* baseada naquela utilizada no AutoDock [MORRIS *et al.*, 1998], e descrevem a função como um somatório ponderado dos termos intra e intermolecular.

Autores em [GARDINER *et al.*, 2001] utilizaram um AG para modelar o *docking* molecular, aplicado especificamente a proteínas, utilizando uma função de *score* baseada em complementaridade de superfícies. Este artigo foi utilizado como referência para o estágio inicial de *docking*, abordado neste trabalho, e é descrito detalhadamente no sub-capítulo 5.4.3.

5. METODOLOGIA PROPOSTA

5.1. ALGORITMO PROPOSTO: SOCOPT-AINET

A proposta do SOCOpt-aiNet é a utilização de um método auto-organizável, inspirado no modelo de interação de espécies [BAK, 1997], em substituição à etapa de supressão e adição de soluções aleatórias utilizados no opt-aiNet. O restante do algoritmo (os processos de seleção clonal e maturação de afinidade) é mantido inalterado.

O algoritmo de supressão do opt-aiNet original assume que cada célula interage com todas as outras em determinado instante. Pode-se argumentar que a técnica proposta faz mais sentido biológico, uma vez que considera que as células do sistema imunológico estão distribuídas pelo espaço se movimentando de forma casual, o que deve permitir que elas se encontrem e interajam com qualquer outra de forma aleatória.

Quadro 5: Pseudo código do algoritmo de supressão do SOCOpt-aiNet.

```

1  Início supressão (célula  $c$ )
2    seleccione aleatoriamente duas células
3    para cada célula  $c_s$  selecionada faça
4      se  $f(c)$  significativamente próximo de  $f(c_s)$  então
5        seleccione célula  $c_s$  para supressão
6         $ext\_size \leftarrow ext\_size + 1$ 
7        supressão ( $c_s$ ) //chamada recursiva
8      fim se
9    fim para
10   suprima célula  $c$ 
11 Fim supressão

```

No Quadro 5 é apresentado em forma de pseudo-código o algoritmo de supressão proposto. Considerando-se a população de células dispostas aleatoriamente no espaço, de forma que pudessem se encontrar ao acaso, quando a etapa de supressão ocorre, uma célula de índice i (c_i) é selecionada aleatoriamente na população para iniciar a supressão. Em seguida, duas outras células são selecionadas aleatoriamente, analogamente ao fato de duas células do organismo se encontrarem ao acaso (linha 2). Uma verificação é feita no *fitness* de cada uma das células encontradas observando se os valores não são significativamente diferentes do *fitness* da célula c_i (linha 4). A verificação é realizada de forma semelhante à equação 5, substituindo $\bar{f}$ pelo *fitness* de uma das duas células encontradas, e $\bar{f}_0$ pelo da célula de índice i . Caso o valor resultante seja menor que um limiar de supressão σ_s então, diz-se que a célula verificada é

selecionada para supressão (linha 5). Cada célula selecionada para supressão pode interagir recursivamente com duas outras células até que não se encontrem outras com valores próximos de *fitness*, de acordo com o critério descrito. Nota-se que este processo também tem semelhança com uma avalanche do modelo de pilhas de areia [BAK, 1997]. Por fim, todas as células selecionadas para supressão e a de índice *i* são suprimidas (linha 10). Para que a melhor solução seja preservada, a solução com melhor *fitness* na população corrente é sempre mantida, isto é, ele não participa da etapa de supressão.

No opt-aiNet original, a verificação para supressão é feita através da distância euclidiana entre as células, eliminando soluções semelhantes (próximas) no espaço de busca. Os algoritmos propostos fazem uma simples comparação dos valores de *fitness* como uma medida de similaridade. Um problema da abordagem utilizada é que existe a possibilidade de supressão de soluções que não convergiram. Porém, este critério pode ser utilizado, pois é pouco comum, de uma forma geral, o encontro de soluções com *fitness* semelhantes em regiões diferentes do espaço.

As funções de avaliação são, em geral, imprecisas quando se tratam de simulações de problemas do mundo real. Estas imprecisões podem até mesmo ser propositais, para diminuir o custo computacional da avaliação. Considerando esta característica, o critério utilizado pelo opt-aiNet é interessante, pois armazena soluções distantes no espaço, mesmo que tenham *fitness* semelhantes na simulação, o que pode apresentar desempenhos diferentes no mundo real. Os algoritmos propostos independem de localização espacial para a supressão, portanto, soluções com *fitness* semelhante são suprimidas sempre que encontradas.

Diferente do algoritmo original, a etapa de supressão descrita é aplicada a cada intervalo g_s de gerações sem a prévia verificação de convergência da população, que somente ocorre na etapa de adição de soluções aleatórias. Note que no algoritmo de supressão apresentado no Quadro 5 é computado o tamanho *ext_size* da extinção (linha 6), ou seja, o número de células suprimidas no processo. Este valor é utilizado para determinar o número de imigrantes aleatórios a serem inseridos na população. Caso o *fitness* médio da população tenha convergido, de acordo com o critério da equação (5), são adicionados $r=(ext_size*10)\%$ de soluções aleatórias na população, após a supressão.

Note que, diferente do algoritmo original, a proporção de indivíduos adicionados não é fixo, e é baseada no tamanho da extinção. Porém, se o número a ser adicionado for menor que o tamanho da extinção, então são adicionadas soluções aleatórias suficientes para que o tamanho da população se mantenha constante. A estratégia utilizada nesta etapa busca permitir o controle do tamanho da população para auxiliar a busca de novas regiões promissoras.

Pode-se observar que o caráter recursivo da etapa de supressão descrita possui grande semelhança com a reação em cadeia que ocorre na técnica utilizada no SORIGA [TINÓS & YANG, 2007]. Portanto, espera-se que com a incorporação deste algoritmo de supressão e adição de soluções aleatórias, ocorra um controle da diversidade da população de forma auto-organizada, o que deve permitir uma melhor exploração do espaço de busca, em especial em funções multimodais, e possivelmente uma diminuição da complexidade do algoritmo. Considerando que o desempenho decepcionante de MhPs muitas vezes pode ser justificado pela escolha inadequada dos parâmetros, a presença da auto-organização no processo de otimização deve ainda diminuir a sensibilidade à escolha de parâmetros, uma vez que esta ocorre de acordo com o desenvolvimento da população e da diversidade desta no problema.

5.2. ALGORITMO PROPOSTO: SOCAIN

Em experimentos iniciais, foi notado que, devido ao fato do SOCopt-aiNet sempre eliminar uma solução na etapa de supressão, a população apresentou uma grande dificuldade de convergir, o que impossibilitou o controle do tamanho da população.

O SOCAIN, uma versão modificada do SOCopt-aiNet, possui uma alteração na etapa de supressão, buscando permitir que a população de células de memória (soluções candidatas) seja passada para a geração seguinte, sem que ocorra substituição alguma. Isto deve prover mais tempo ao algoritmo para realizar buscas localmente, diminuindo a supressão precoce de soluções.

O algoritmo de supressão do SOCAIN é apresentado no Quadro 6 em forma de pseudo-código. No SOCopt-aiNet, sempre que uma célula é selecionada para a supressão, esta é em seguida suprimida. Uma diferença sutil existe no algoritmo modificado, onde uma célula selecionada somente é suprimida caso o critério de

similaridade seja satisfeito (linha 4). Tal mudança permite que ocorram extinções de tamanho $ext_size=0$.

Quadro 6: Pseudo código do algoritmo de supressão do SOCAIN.

```
1  Início supressão (célula c)
2      selecione aleatoriamente duas células
3      para cada célula cs selecionada faça
4          se  $f(\mathbf{c})$  significativamente próximo de  $f(\mathbf{c}_s)$  então
5               $ext\_size \leftarrow ext\_size + 1$ 
6              supressão (cs) //chamada recursiva
7              suprima célula c
8          fim se
9      se c foi suprimido
10          $ext\_size \leftarrow ext\_size + 1$ 
11     fim para
12 Fim supressão
```

5.2.1. Aplicação de mutação com distribuição q-gaussiana

Tradicionalmente, o operador de mutação utilizado para a otimização de problemas de codificação real é com distribuição normal. Porém estudos têm demonstrado grande interesse na utilização de distribuições com caudas mais longas, como em [YAO *et al.*, 1999] que é utilizado um operador de mutação com distribuição Cauchy e em [TINÓS & YANG, 2010] que é utilizado um operador com distribuição q-gaussiana com parâmetro q auto adaptável. Neste trabalho foi aplicado um operador de mutação q-Gaussiano baseado no trabalho [TINÓS & YANG, 2010].

Os autores de [TINÓS & YANG, 2010] descrevem um estudo envolvendo a aplicação de um operador de mutação com distribuição q-gaussiana em AE com codificação real. Uma das características mais interessantes da distribuição q-gaussiana, para sua aplicação em AE, é a existência de um parâmetro q que rege a forma da curva. Alterações no valor de q permitem distribuições com maior ou menor probabilidade nas caudas da distribuição, como pode ser visto na Figura 4. Desta forma, controlar o parâmetro q deve permitir que ocorram valores mais ou menos distantes da média quando for mais interessante. Por exemplo, quando as soluções estão presas em um ótimo local pode ser mais interessante uma distribuição com caudas mais longas, e quando as soluções estão em convergência, uma distribuição mais local, de caráter mais exploratório, deve ser mais interessante.

Figura 4: Formas da distribuição q-gaussiana para diferentes valores de q [TINÓS & YANG, 2010]. Com $q=1$ temos a distribuição normal padrão, e com $q=2$, a distribuição Cauchy é reproduzida.

Nota-se na Figura 4 que com o aumento do valor de q , com q em $(-\infty, 3)$, há um aumento das caudas das distribuições (*i.e.*, frequência de valores mais distantes da média). Considerando isso, Tinós & Yang [2010] desenvolveram um operador de mutação, utilizando uma distribuição q-gaussiana, capaz de controlar a forma da distribuição das mutações, associando a cada solução candidata (indivíduo) $i = 1, 2, \dots, N$ do AE, um valor q_i auto-adaptável. Assim, o parâmetro q_i também é passível de mutação, que é realizada de acordo com:

$$q_i = q_{i_0} e^{\tau_q \text{rand}_g(0,1)} \quad (6)$$

sendo q_{i_0} o valor atual, $\text{rand}_g(0,1)$ um valor aleatório com distribuição normal, e $\tau_q = 1/\sqrt{2N_D}$, sendo que N_D é o número de dimensões do problema. Para a geração de valores com distribuição q-gaussiana foi utilizado o gerador descrito em [THISTLETON *et al.*, 2007], que recebe como parâmetro um valor q e retorna um número aleatório com tal distribuição.

A equação (6), no algoritmo em [TINÓS & YANG, 2011], é aplicada no q_i associado a cada indivíduo $i = 1, 2, \dots, N$, e ocorre com probabilidade r_q na geração dos novos indivíduos por mutação. O operador de mutação q-gaussiana aplicado nos AISs atualiza o q_i de cada célula de memória, e o novo valor em uma determinada célula é então utilizado para gerar números com distribuição q-gaussiana, que são utilizados

como na equação (3), em substituição aos valores com distribuição normal ($Rand_g(0,1)$), para mutar a célula clone.

Na inicialização do AIS o valor é $q_i = 1$ (*i.e.*, valor que reproduz a distribuição normal), e atualizações nos valores estão limitadas ao intervalo (0,8; 0,8e), como utilizado em [TINÓS & YANG, 2011]. Caso a saída da equação (6) seja um valor fora do intervalo, então o valor é passado inalterado para os clones, que serão gerados com o q_i da célula mãe. Embora o valor inicial de q seja um parâmetro do algoritmo, é importante salientar que o sistema é capaz, de forma auto-organizável, de controlar o valor de q , sem influência direta do usuário. Experimentos utilizando a mutação q -gaussiana foram aplicados nos AIS estudados neste trabalho e foram chamados (q)opt-aiNet, (q)SOCopt-aiNet e (q)SOCAIN.

A aplicação de mutação q -gaussiana em AISs é uma idéia original, ou seja, ainda não foi proposto o uso de tal distribuição neste tipo de algoritmo. Adicionalmente, no AutoDock descrito no subcapítulo 4.3 utiliza a distribuição Cauchy para a mutação, uma distribuição com caldas longas. Isto pode ser interessante em um contexto de busca global pelo algoritmo, porém uma mutação com caldas mais curtas é mais interessante em geral para exploração local. A utilização de um algoritmo com q adaptável deve ser capaz de controlar a distribuição para ambas situações.

5.3. AVALIAÇÃO DE ALGORITMOS EVOLUTIVOS

Para a avaliação dos algoritmos desenvolvidos neste trabalho, foram utilizadas as funções definidas em [SUGANTHAN *et al.*, 2005]. Trata-se de um relatório técnico referente a uma competição do Congresso em Computação Evolutiva (CEC), definindo uma série de parâmetros e funções para a avaliação de AEs. Tais parâmetros foram desenvolvidos devido ao surgimento de um grande número de algoritmos de otimização e conseqüentemente da necessidade de parametrizar a comparação das diferentes abordagens de uma forma sistemática e genérica. Isto foi atingido definindo uma série de funções uni e multimodais cujos ótimos são conhecidos. Foram definidos também critérios de parada como número máximo de avaliações por problema e erro mínimo.

São definidas 25 funções de avaliação, todas escaláveis, de minimização e com codificação real, divididas em funções unimodais ($f1$ a $f5$), multimodais básicas ($f6$ a $f12$), funções expandidas ($f13$ e $f14$) e composição de funções ($f15$ a $f25$). A Tabela 2 apresenta uma descrição das funções de *benchmark*.

Tabela 2: Descrição das funções de *benchmark* [SUGANTHAN *et al.*, 2005; TINÓS & YANG, 2007].

Função	Nome	$f(x^*)$	Intervalo	Propriedades
f1	Shifted sphere function	-450	$[-100, 100]$	U, S
f2	Shifted Schwefel's Problem 1.2	-450	$[-100, 100]$	U, NS
f3	Shifted rotated high conditioned elliptic function	-450	$[-100, 100]$	U, NS, RO
f4	Shifted Schwefel's Problem 1.2 with noise in fitness	-450	$[-100, 100]$	U, NS, RU
f5	Schwefel's Problem 2.6 with global optimum on bounds	-310	$[-100, 100]$	U, NS
f6	Shifted Rosenbrock's function	390	$[-100, 100]$	M, NS
f7	Shifted rotated Griewank's function without bounds	-180	$[-\infty, \infty]$	M, NS, RO
f8	Shifted rotated Ackley's function with global optimum on bounds	-140	$[-32, 32]$	M, NS, RO
f9	Shifted Rastrigin's function	-330	$[-5, 5]$	M, S
f10	Shifted rotated Rastrigin's function	-330	$[-5, 5]$	M, NS, RO
f11	Shifted rotated Weierstrass' function	90	$[-0,5; 0,5]$	M, NS, RO
f12	Schwefel's Prob. 2.13	-460	$[-\pi, \pi]$	M, NS
f13	Shifted expanded Griewank's plus Rosenbrock's function	-130	$[-3, 1]$	M, NS
f14	Shifted rotated expanded Schaffer's F6 function	-300	$[-100, 100]$	M, NS
f15	Hybrid composition function	120	$[-5, 5]$	M, NS
f16	Rotated version of hybrid composition function f15	120	$[-5, 5]$	M, NS, RO
f17	Function f16 with noise	120	$[-5, 5]$	M, NS, RO, RU
f18	Rotated hybrid composition function 1	10	$[-5, 5]$	M, NS, RO
f19	Rotated hybrid composition function 1 with narrow basin global optimum	10	$[-5, 5]$	M, NS
f20	Rotated hybrid composition function 1 with global optimum on bounds	10	$[-5, 5]$	M, NS
f21	Rotated hybrid composition function 2	360	$[-5, 5]$	M, NS, RO
f22	Rotated hybrid composition function 2 with high condition number matrix	360	$[-5, 5]$	M, NS
f23	Non-continuous rotated hybrid composition function 2	360	$[-5, 5]$	M, NS
f24	Rotated hybrid composition function 3	260	$[-5, 5]$	M, NS, RO
f25	Rotated hybrid composition function 3 without bounds	260	$[-\infty, \infty]$	M, NS, RO

U unimodal, M multimodal, S separável, NS não separável, RO rotacionada,, RU com ruído, x^* ótimo global

Aqui, os mesmos parâmetros de avaliação sugeridos no relatório em [SUGANTHAN *et al.*, 2005] são utilizados para testar e comparar o SOCOpt-aiNet e o opt-aiNet. Implementações das funções de avaliação são disponibilizadas nas linguagens C, Java e Matlab³. Neste trabalho foi realizada uma adaptação do código em C para C++.

Para verificar a significância estatística dos resultados, um teste não-paramétrico de Wilcoxon (*signed-rank Wilcoxon test*) [WILCOXON, 1945], com nível de significância de 5%, foi realizado para a comparação dos algoritmos. As comparações foram realizadas para experimentos utilizando dimensões $N_D = 2, 10, 30$ e 50.

³ As implementações podem ser encontradas em <http://www.ntu.edu.sg/home/EPNSugan/> (visitado em 25/10/2011).

5.4. CODIFICAÇÃO DO PROBLEMA DE *DOCKING*

Neste trabalho foi realizado o *bound docking* de proteínas utilizando uma função de *score* baseada em complementaridade de superfícies, como descrito em [GARDINER *et al.*, 2001]. Desta forma, para a implementação da função de *score*, foi necessário computar a superfície excluída ao solvente (na Figura 2 é representada pela região com transparência na proteína alvo ou gradeada na proteína ligante), uma superfície baseada nos raios de van der Waals que definem a vizinhança da molécula que não é acessível ao solvente.

5.4.1. Codificação da superfície excluída ao solvente

A computação da superfície excluída ao solvente (SES) foi feita através do algoritmo MSMS de [SANNER *et al.*, 1996] que implementa a técnica descrita por [CONNOLLY, 1983; CONNOLLY, 1993].

A SES é determinada através da “rolagem” de uma sonda esférica, representando uma molécula de solvente, por entre os átomos da molécula. Esta, quando entra em contato com dois ou mais átomos, delimita uma superfície, onde o solvente não é capaz de alcançar. Tal superfície é a chamada SES. No caso em que a sonda está em contato com somente um átomo, então, a SES é o próprio raio de van der Waals do átomo. Na Figura 5, é apresentada uma sonda (*probe*) rolando por entre quatro átomos ilustrando graficamente o funcionamento do algoritmo. Os pontos p_i , i em $[1, 6]$, representam pontos de contato dos átomos com a sonda esférica, e delimitam a SES.

A SES pode gerar três tipos de superfícies, que são computadas pelo algoritmo: côncava (ou reentrante); convexa (ou de contato); e sela. Na Figura 5 os conjuntos de pontos $[p_1, p_2, p_3]$ e $[p_4, p_5, p_6]$ delimitam superfícies côncavas e o conjunto $[p_2, p_3, p_4, p_5]$ uma superfície em sela. Regiões onde a sonda entra em contato com somente um átomo são superfícies convexas.

O programa MSMS tem como entrada um arquivo no formato PDB, que tem definida a molécula e sua estrutura; o nome do arquivo de saída, onde serão salvas as informações da superfície; o raio da sonda esférica, e a densidade de pontos de superfície. O arquivo de saída possui um conjunto de pontos representando a superfície da molécula. Cada ponto de superfície tem informações de aminoácido, seu número de sequência, tipo do átomo, coordenadas do ponto, tipo de superfície (côncava, convexa ou sela) e vetor unitário normal à superfície.

Figura 5: Representação da construção da superfície excluída ao solvente (adaptado de [SANNER *et al.* , 1996]).

5.4.2. Função de score

Considerando a realização de um *bound docking* as proteínas do complexo, que já são conhecidas a princípio, são armazenadas em arquivos separados e centralizadas na origem do espaço (0, 0, 0). Suas superfícies são, então, geradas pelo programa MSMS com raio da sonda de 1,5Å e densidade de pontos igual a 1 ponto/Å².

Autores em [GARDINER *et al.*, 2001] ainda realizaram uma discretização dos pontos das superfícies geradas pelo MSMS. Esta é feita colocando os pontos em uma grade cúbica de 2Å de aresta. Desta forma pontos dentro de um mesmo espaço da grade possuem a mesma coordenada (*e.g.* pontos (0,2; 1,8; 2,7) e (1,9; 0,7; 3,3) possuem mesma coordenada (0, 0, 2) na grade). Esta simplificação fornece uma tolerância no processo de ligação e, adicionalmente, deve levar a uma suavização do espaço de busca da função de *fitness*.

A função de *fitness* (ou *score*) avalia se a posição da molécula ligante referente à molécula receptora forma um complexo molecular fortemente ligado, buscando maximizar a superfície de contato minimizando colisões. Desta forma, possui como entrada dois tipos de parâmetros, um de coordenada e um de rotação, para cada um dos

eixos do espaço, totalizando seis valores reais utilizados para a transformação espacial da molécula ligante, enquanto a molécula receptora é mantida fixa na origem.

As entradas de coordenada são utilizadas para posicionar a molécula ligante em uma determinada região do espaço. Os limites do espaço de busca das coordenadas foram computados através dos raios máximos das moléculas receptora e ligante, de forma que o centro da proteína esteja posicionado no intervalo $[-r_r-r_l, r_r+r_l]$, sendo r_r e r_l os raios máximos (*i.e.*, distância euclidiana do centro até o ponto mais distante da superfície) das moléculas receptora e ligante, respectivamente. As entradas de rotação são no intervalo $[-\pi, \pi)$ e são utilizadas para gerar uma orientação candidata $\mathbf{D}_{Lc}$ da molécula ligante. A rotação é realizada através de matrizes de rotação nos três eixos, um a um [WOLFRAM MATHWORLD, 2012].

Nota-se que, embora ocorra a discretização das superfícies, o problema pode ser visto como uma otimização com codificação real, uma vez que as entradas de coordenadas e rotações podem possuir valores contínuos.

A função de *fitness* atribui um *score* parcial a cada par de pontos, do receptor e do ligante, que se localizam em uma mesma coordenada na grade. A função de *score* de cada ponto conta com quatro parâmetros que são avaliados sequencialmente. Desta forma, dois pontos de superfícies das moléculas receptora e ligante são considerados complementares se:

1. possuem mesma coordenada na grade;
2. ângulos das normais são aproximadamente opostos, considerando uma margem de tolerância;
3. há complementaridade dos tipos de superfícies e;
4. o potencial de ligações de hidrogênio foi satisfeito.

A tolerância utilizada na condição 2 é dada como parâmetro da função de avaliação. Na condição 3, a complementaridade ocorre quando os pontos têm tipos diferentes ou ambos são de sela, como descrito na Tabela 3. A condição 4 é satisfeita quando suas características são diferentes ou ambos são doador/receptor, exceto no caso de não ligante a H, que complementa somente não ligante a H (*i.e.* doador se liga a receptor ou doador/receptor, doador/receptor se liga a todos, exceto não ligante a H, e este se liga somente a não ligante a H), como apresentado na Tabela 4.

Tabela 3: Tabela de complementaridades para tipo de superfície.

	Côncavo	Convexo	Sela
Côncavo		X	X
Convexo	X		X
Sela	X	X	X

As características de ponte de hidrogênio, por simplicidade, são atribuídas na forma: átomos de oxigênio são classificados como aceptores, com exceção do OH da tirosina, do OG da serina e do OG1 da treonina, que são marcados como doadores/aceptores; átomos de nitrogênio são caracterizados como doadores; os demais átomos são caracterizados como não ligantes a H.

Tabela 4: Tabela de complementaridades para potencial de ligações de hidrogênio

	Doador	Receptor	Doador/Receptor	Não ligante a H
Doador		X	X	
Receptor	X		X	
Doador/Receptor	X	X	X	
Não ligante a H				X

Na implementação utilizada neste trabalho, baseada no modelo apresentado em [GARDINER *et al.*, 2001], todas as condições têm de ser satisfeitas para que um determinado par de pontos seja considerado complementar.

Adicionalmente, é aplicada uma penalidade no caso de colisão, isto é, quando uma molécula encontra-se no interior da outra. Para isso, em [GARDINER *et al.*, 2001] é definida uma região interna na molécula receptora através de uma grade interna à superfície. Desta forma, pontos do ligante que se encontram na grade interna são considerados pontos de colisão e é dada uma penalidade de w_c .

O *score* de um par de pontos ($\mathbf{d}_i$, $\mathbf{d}_j$), com i em $[1, n_R]$ e j em $[1, n_L]$, sendo n_R e n_L o número de pontos das superfícies $\mathbf{D}_R$ da molécula receptora, e $\mathbf{D}_L$ da ligante respectivamente, é dado pela equação

$$f_{dot}(\mathbf{d}_i, \mathbf{d}_j) = \begin{cases} 1, & \text{se } \mathbf{d}_i \text{ e } \mathbf{d}_j \text{ são complementares;} \\ -w_c, & \text{caso contrário.} \end{cases} \quad (7)$$

O *fitness* final de uma solução candidata $\mathbf{x}$, que gera uma orientação candidata $\mathbf{D}_{Lc}$, é dado por

$$f_{dock}(\mathbf{D}_R, \mathbf{D}_{Lc}) = \sum_{i=1}^{n_R} \sum_{j=1}^{n_L} f_{dot}(\mathbf{d}_i, \mathbf{d}_j) \quad (8)$$

A visualização dos complexos resultantes (como visto na Figura 2) foi realizada através do software Chimera, que contém uma série de ferramentas para a visualização e análise de estruturas moleculares [PETTERSEN, 2004].

5.4.3. Algoritmo genético de [GARDINER *et al.*, 2001]

Neste trabalho foi realizada também a implementação do AG descrito em [GARDINER *et al.*, 2001] e seus resultados comparados com os dos algoritmos aqui propostos na otimização da função de *score* descrita.

A implementação descrita em [GARDINER *et al.*, 2001] é um AG de estado fixo (*steady-state*), isto é, a população possui tamanho invariável, onde uma porção fixa dos indivíduos é substituída por novos gerados pelos operadores de reprodução. Neste caso, são sempre 5% dos indivíduos de pior *fitness*. Foram utilizados operadores de mutação e cruzamento (*crossover*) e os indivíduos são selecionados pelo método da roleta.

Prévia à geração de um indivíduo para a geração seguinte, é feita a escolha de um dos operadores, mutação ou cruzamento, com igual probabilidade. Mutação seleciona somente um indivíduo e gera um indivíduo para a geração seguinte, e pode ocorrer de duas formas: mutação padrão, que seleciona uma posição aleatória do cromossomo do indivíduo e altera para um valor aleatório; ou *small-creep* que seleciona uma posição da mesma forma que a padrão, porém a mutação ocorre adicionando um valor pequeno (no intervalo de $[-2\text{\AA}, 2\text{\AA}]$ no caso de coordenada e $[-5^\circ, 5^\circ]$ no caso de rotação). Ambas as formas de mutação ocorrem com igual probabilidade.

O operador de cruzamento, seleciona dois indivíduos e gera outros dois, e pode ocorrer de duas formas que são selecionadas com igual probabilidade: *crossover* simples de um ponto, onde um índice do cromossomo é selecionado aleatoriamente, e todo o trecho a partir do ponto selecionado é trocado entre os dois indivíduos selecionados; e *crossover* de troca (*exchange crossover*), onde um índice é selecionado da mesma forma, porém somente aquele selecionado é trocado entre os indivíduos.

Adicionalmente, no AG de [GARDINER *et al.*, 2001] é aplicada uma técnica de restrição de nicho, onde, após um número fixo de gerações, considera-se que a população convergiu, e então o melhor indivíduo é armazenado e a população é então reinicializada aleatoriamente, com a diferença que nenhuma nova solução pode ser posicionada em uma região em torno do centro da melhor solução encontrada. Esta região é chamada nicho e, neste caso, foi definido como uma esfera de raio $2\text{\AA}$. Tal

técnica busca restringir o espaço de busca, eliminando regiões já conhecidas, forçando o algoritmo a buscar soluções em regiões não exploradas anteriormente. Diversas restrições (nichos) podem ser criadas ao longo de uma execução do AG. Nos testes de [GARDINER *et al.*, 2001], foram criadas cinco. A fim de comparação, o operador de restrição de nicho foi ainda experimentado nos AISs propostos.

6. RESULTADOS E DISCUSSÃO

Neste capítulo são apresentados os resultados dos experimentos realizados neste trabalho. O capítulo foi dividido da seguinte forma: no subcapítulo 6.1 são apresentados os experimentos da otimização das funções contínuas de *benchmark* como proposto em [SUGANTHAN *et al.*, 2005], e; no subcapítulo 6.2 são apresentados os resultados do *docking* molecular.

6.1. EXPERIMENTOS COM OTIMIZAÇÃO DE FUNÇÕES CONTÍNUAS

Neste subcapítulo são apresentados os resultados dos experimentos envolvendo a otimização das funções descritas em [SUGANTHAN *et al.*, 2005].

6.1.1. Experimentos de *benchmark* do SOCOpt-aiNet

Nesta etapa, o desempenho do SOCOpt-aiNet foi comparado com o do opt-aiNet. Os algoritmos foram testados na otimização das funções descritas em [SUGANTHAN *et al.*, 2005].

Experimentos iniciais, descritos no apêndice A.1, foram realizados para a determinação dos parâmetros do SOCOpt-aiNet. Como a única modificação do algoritmo SOCOpt-aiNet, em comparação ao opt-aiNet, foi no algoritmo de supressão, os dois possuem parâmetros em comum que, para testes iniciais, foram utilizados como definidos em [CASTRO & TIMMIS, ou seja:

- Número inicial de células (soluções candidatas): $N=20$;
- Número de clones gerados por cada célula: $N_c=10$;
- Pressão da maturação de afinidade: $\beta=1$.

Os parâmetros específicos do opt-aiNet são:

- Limiar de convergência: $l_c=0,0001$;
- Limiar de supressão: $\sigma_s=0,2$;
- Porcentagem de soluções aleatórias adicionadas: $r=40\%$;
- Intervalo de supressão: $g_s=5$.

Por fim, os parâmetros do algoritmo proposto selecionados após os experimentos apresentados no apêndice A.1:

- Limiar de convergência: $l_c=0,001$;
- Limiar de supressão: $\sigma_s=0,06$;
- Porcentagem de soluções aleatórias adicionadas: $r=(ext_size*10)\%$;
- Intervalo de supressão: $g_s=75$.

O critério de parada utilizado nos testes em [CASTRO & TIMMIS, 2002] não foi utilizado, pois viola o critério definido em [SUGANTHAN *et al.*, 2005] para avaliação de AEs. Desta forma, o critério de parada é $N_A=N_D \times 10^4$ avaliações, onde N_D é o número de dimensões do problema. Em [CASTRO & TIMMIS, 2002], o teste do opt-aiNet usa como critério de parada um determinado número de gerações. Na comparação de diferentes AEs, este pode não ser justo, considerando que o tamanho da população pode ser alterado, justificando a utilização de um número limitado de avaliações.

Cada função é executada para 25 sementes aleatórias diferentes para garantir a qualidade estatística do resultado, e é executada para 2, 10, 30 e 50 dimensões. De todas as execuções são armazenados o melhor *fitness* de cada, e então apresentados a média, o desvio padrão, a mediana e o mínimo dos erros das funções, além do p-valor do teste de Wilcoxon com nível de significância de 5%. Neste são ainda indicados se o SOCOpt-aiNet superou o opt-aiNet (sinal +) ou se foi superado (sinal -) e se a diferença possui significância estatística (marcado com S). Dado que as funções são conhecidas, o erro obtido em cada execução é computado pela simples diferença do *fitness* obtido com o *fitness* no ponto ótimo ($|f(\mathbf{x})-f(\mathbf{x}^*)|$). Quanto menor o valor do erro, melhor o desempenho do algoritmo. Tabelas com os resultados dos experimentos são apresentados no apêndice A.2.

Nos resultados obtidos para a otimização das funções com duas dimensões (Tabela A.11) o algoritmo proposto superou o opt-aiNet original em 9 dos casos, dos quais 6 apresentaram relevância estatística. Dos demais casos, onde o algoritmo original apresentou melhor desempenho na comparação das medianas, 8 apresentaram relevância estatística.

Nos resultados para os problemas de 10 dimensões apresentados na Tabela A.12, o algoritmo proposto apresentou desempenho melhor em 8 das funções, das quais 5 apresentaram diferença significativa estatisticamente. Dos demais casos, onde o opt-

aiNet apresentou melhores resultados, sendo que em 13 das funções houve relevância estatística.

Nos testes com $N_D=2$, das funções em que o SOCOpt-aiNet apresentou melhor desempenho, somente 4 delas são multimodais, das quais 3 com significância estatística. Nos experimentos com funções com $N_D=10$, o SOCOpt-aiNet apresentou melhores resultados em um número maior de funções multimodais, contando com 7 no total das quais 3 apresentaram relevância estatística. Já o opt-aiNet apresentou melhor desempenho em 12 das funções multimodais das quais 10 apresentaram significância no teste de Wilcoxon. Resultados para $N_D=2$ e 10 podem ser vistos na Figura 6.

Figura 6: Comparativo da frequências em que o SOCOpt-aiNet superou e foi superado pelo opt-aiNet para $N_D = 2$ (a esquerda) e 10 (a direita). Sinal + indica a frequência em que o algoritmo superou e S+ a frequência em que ocorre com significância estatística.

Nas funções com $N_D=30$ (Tabela A.13) o algoritmo proposto apresentou melhor desempenho em 7 funções, das quais 6 apresentaram diferença significativa de acordo com o teste estatístico. Das funções que o SOCOpt-aiNet apresentou melhores resultados 6 são multimodais. O opt-aiNet apresentou melhor desempenho com relevância estatística em 16 das funções das quais 13 são multimodais.

Os resultados referentes aos experimentos com funções de 50 dimensões são apresentados na Tabela A.14. O SOCOpt-aiNet apresentou menores erros em 6 funções, das quais 5 apresentaram significância estatística. O opt-aiNet original apresentou relevância estatística em 15 dos casos de teste, de onde 11 são funções multimodais. O

algoritmo proposto apresentou melhor resultado em apenas 6 funções multimodais e 3 com significância pelo teste de Wilcoxon.

Em dois casos, nas funções $f8$ e $f14$, os testes apresentaram mesmo valor de mediana e as médias não são significativamente diferentes, portanto foi considerado empate.

Em uma análise mais profunda, pode-se verificar que o SOCopt-aiNet apresentou desempenho melhor com significância estatística nas funções com ruído ($f4$ e $f17$) com $N_D=30$ e 50. A função Rastrigin ($f9$), muito utilizada no teste de AEs, é uma função altamente multimodal (grande quantidade de ótimos locais). Observa-se que para esta função com $N_D=30$ e 50, o SOCopt-aiNet apresentou resultados melhores com significância estatística.

Figura 7: Comparativo da frequências em que o SOCopt-aiNet superou e foi superado pelo opt-aiNet para $N_D = 30$ (a esquerda) e 50 (a direita).

Os resultados das comparações são apresentados na Figura 6 (para $N_D = 2$ e 10) e na Figura 7 (para $N_D = 30$ e 50).

6.1.2. Experimentos de *benchmark* do SOCAIN

O SOCAIN foi testado com os mesmos parâmetros selecionados no segundo experimento (Apêndice A.1) e seus resultados foram comparados com o SOCopt-aiNet (com novos parâmetros) e com o opt-aiNet original. Resultados da comparação são indicados da mesma forma, como nos outros experimentos, com sinal (+) em casos que o SOCAIN supera o algoritmo comparado, (-) caso contrário e (S) caso a diferença seja estatisticamente significativa.

As tabelas: Tabela A.15, Tabela A.16, Tabela A.17 e Tabela A.18; contêm os comparativos dos resultados do SOCOpt-aiNet com o novo algoritmo, SOCAIN. Nos testes com $N_D=2$, dentre as funções unimodais, o SOCAIN superou o SOCOpt-aiNet somente na função $f3$. Nos demais casos o SOCAIN foi superado, e em somente um houve relevância estatística (função $f4$). Nas funções multimodais básicas o SOCAIN obteve melhor desempenho nas funções $f8$, $f11$ e $f12$, sem relevância estatística. Nas funções onde o SOCAIN foi superado, houve diferença significativa em $f7$ e $f9$. Em 7 das 11 composições de funções, o SOCAIN superou o SOCOpt-Net. Nas demais funções, este obteve melhor desempenho somente na função $f24$. Nas funções $f16$, $f21$ e $f23$, ambos obtiveram mesmo valor de mediana. Os resultados são apresentados na Figura 8 e Figura 9.

Testes com $N_D=10$ o SOCAIN apresentou melhor desempenho em 15 dos casos de teste, das quais 4 são unimodais, 1 é multimodal básica, 1 é função expandida e 9 são em composições de funções. Em nenhum destes houve significância estatística. O SOCOpt-aiNet apresentou melhores resultados em 9 dos casos, 4 com diferença estatística, nas funções $f4$, $f7$, $f9$ e $f12$.

Figura 8: Comparativo da frequências em que o SOCAIN superou e foi superado pelo SOCOpt-aiNet para $N_D = 2$ (a esquerda) e 10 (a direita).

Com $N_D=30$, o SOCAIN apresentou melhor resultado em 4 das funções unimodais, e apresentou significância estatística na $f5$. Dentre a funções multimodais básicas o SOCAIN foi superado em 4 dos 7 casos, porém, nas funções que superou o SOCOpt-aiNet, todos apresentaram significância estatística. O SOCAIN ainda apresentou melhor desempenho em todas as funções estendidas e composições de funções. Destes 6 apresentaram diferença estatística.

Nos experimentos com $N_D=50$, o SOCOpt-aiNet apresentou melhor resultado somente em 4 funções: $f1$, $f4$, $f7$ e $f9$, das quais $f4$ e $f9$ com significância estatística. Em todos os demais casos de testes, o SOCAIN superou o SOCOpt-aiNet. Das 21 funções em que isto ocorreu, 16 foram relevantes estatisticamente.

Figura 9: Comparativo da frequências em que o SOCAIN superou e foi superado pelo SOCOpt-aiNet para $N_D = 30$ (a esquerda) e 50 (a direita).

Os resultados apresentados na Figura 8 são indicativos de que o SOCOpt-aiNet apresentam melhor resultado em problemas mais simples com menor dimensão. Isto pode ser verificado em comparação com a Figura 9, onde nota-se que o desempenho do SOCAIN é significativamente melhor em um maior número de funções nos casos de $N_D = 30$ e 50.

Nos experimentos com as diferentes dimensões, o SOCAIN apresentou melhores resultados em grande parte dos casos de composições de funções, comparado ao SOCOpt-aiNet, especialmente nos casos com maiores dimensões, onde ocorreu relevância estatística, como pode ser visto na Figura 9. Isto pode ser decorrente da complexidade destas funções, que podem exigir mais tempo para a exploração do espaço localmente. A característica do SOCAIN de suprimir soluções somente no encontro de uma solução semelhante (valor próximo de *fitness*) permite que não ocorra supressão de nenhum indivíduo, evitando a supressão precoce de soluções. Isto deve permitir uma melhor exploração de soluções potencialmente promissoras. Esta característica ainda pode justificar a melhora do desempenho do SOCAIN nas funções multimodais básicas e expandidas, nos experimentos com $N_D=30$ e 50, que têm maior complexidade devido ao número de dimensões.

A Figura 10 permite verificar o comportamento da evolução do erro médio do SOCAIN com o SOCOpt-aiNet em uma execução. Como citado anteriormente, pontos onde ocorrem aumento repentino do erro médio são gerações onde ocorrem substituições. Nota-se que, na curva representando o SOCAIN as supressões não ocorrem em intervalos regulares, como no SOCOpt-aiNet, indicando a ocorrência de um controle do intervalo de supressão. Isto também indica que o algoritmo é capaz de controlar as supressões na população, permitindo uma melhor busca de soluções promissoras em regiões próximas.

Figura 10: Evolução do erro médio da função f_{25} em uma das execuções. Comparativo SOCOpt-aiNet com os SOCAIN.

Os testes comparando o SOCAIN com o SOCOpt-aiNet mostraram a superioridade do primeiro na otimização de funções de maior complexidade, devido ao seu controle do intervalo de supressão. Em um último experimento na otimização das funções de *benchmark*, foi realizada a comparação do SOCAIN com opt-aiNet. Os resultados desta são descritos nas tabelas: Tabela A.19, Tabela A.20, Tabela A.21 e Tabela A.22.

No caso de $N_D=2$, nas funções unimodais, o SOCAIN apresentou desempenho melhor em 4 de 5 funções. Dentre as funções multimodais básicas e expandidas, o SOCAIN superou o opt-aiNet em 5 dos casos, sendo 4 deles com relevância estatística. Comparativos das composições de funções apontaram a superioridade do opt-aiNet em todos os casos, exceto a função f_{22} , em que ambos apresentam mesmo valor de mediana. Destas, houve diferença estatística em 8 das funções.

Testes das funções com $N_D=10$, apresentadas na Tabela A.20, indicaram, novamente, que o opt-aiNet tem melhores resultados nas composições de funções. Nestes casos, o SOCAIN foi superado em 9 casos dos quais 8 apresentaram relevância estatística. Dentre as funções multimodais básicas, o SOCAIN apresentou melhor desempenho em somente 3 casos, e também 3, das funções unimodais.

Dentre os experimentos com $N_D=30$ (Tabela A.21), o SOCAIN foi superior em 8 das funções, das quais: 2 são unimodais, ambas com significância estatística; 3 são multimodais básicas, com 2 casos com diferença estatística; 1 função expandida; e 2 composições de funções, ambas estatisticamente relevantes. Nas demais funções, o opt-aiNet superou o SOCAIN. Das composições de funções, 8 possuem significância pelo teste de Wilcoxon.

Com $N_D=50$, apresentado na Tabela A.22, nas composições de funções, o SOCAIN superou o opt-aiNet nas funções *f16* e *f17*, como ocorreu nos casos com $N_D=30$. Nos outros casos o SOCAIN foi superado, todos com significância estatística. Em comparação com os casos de $N_D=30$, uma melhora ocorreu no desempenho do SOCAIN nas funções multimodais básicas, onde este apresentou melhor desempenho em 4 das funções e, destas, 3 são estatisticamente relevantes.

A Figura 11 e Figura 12 ilustram um comparativo dos desempenhos obtidos pelo SOCAIN e pelo opt-aiNet.

Figura 11: Comparativo da frequências em que o SOCAIN superou e foi superado pelo opt-aiNet para $N_D = 2$ (a esquerda) e 10 (a direita).

Em um paralelo com os resultados apresentados nos experimentos iniciais, para $N_D=50$ (Tabela A.14), o SOCAIN apresentou algumas melhoras, visto nas funções *f8*,

$f10$, e $f14$ que no comparativo com o SOCOpt-aiNet possuía pior desempenho. No caso da função $f10$ foi ainda obtido melhor resultado com significância estatística.

Com $N_D=30$, o SOCAIN apresentou melhora nas função $f10$, $f14$, e $f22$, visto na Tabela A.13, embora na $f10$ e $f22$ o resultado ainda é pior que do opt-aiNet.

Figura 12: Comparativo da frequências em que o SOCAIN superou e foi superado pelo opt-aiNet para $N_D = 30$ (a esquerda) e 50 (a direita).

O SOCAIN apresentou melhores resultados que o SOCOpt-aiNet, visto nos resultados apresentados anteriormente. Um problema visto neste, é que sempre, na execução do algoritmo de supressão, ao menos uma solução é suprimida, mesmo que não seja encontrado um par com *fitness* similar. Porém, o SOCAIN ainda apresenta pior resultado quando comparado com o opt-aiNet. Isto pode ser explicado pela política de adição de soluções aleatórias adotada, que pode ser muito conservadora (*i.e.*, adiciona poucas soluções). Este problema é melhor investigado ao final do subcapítulo 6.1.4, que apresenta um paralelo entre evolução do *fitness* médio com o tamanho das extinções em cada geração.

6.1.3. Experimentos com mutação q-gaussiana

A seguir são apresentados os resultados dos algoritmos com mutação q-gaussiana. Foram comparados os algoritmos opt-aiNet (Tabela A.23, Tabela A.24, Tabela A.25 e Tabela A.26) e SOCAIN (Tabela A.27, Tabela A.28, Tabela A.29 e Tabela A.30) com suas respectivas versões utilizando a mutação q-gaussiana (indicados com um “q” antes dos nomes) e, por fim, foram comparadas as duas versões alteradas (Tabela A.31, Tabela A.32, Tabela A.33 e Tabela A.34).

Figura 13: Comparativo da frequências em que o (q)opt-aiNet superou e foi superado pelo opt-aiNet para $N_D = 2$ (a esquerda) e 10 (a direita).

Figura 14: Comparativo da frequências em que o (q)opt-aiNet superou e foi superado pelo opt-aiNet para $N_D = 30$ (a esquerda) e 50 (a direita).

A análise da Tabela A.25, apresentando os resultados referentes aos experimentos comparando o opt-aiNet com o (q)opt-aiNet para $N_D = 30$, indica que este, que utiliza a mutação q-gaussiana, superou o algoritmo original na maior parte dos casos: em 21 das funções das quais, 7 apresentaram significância estatística. Somente em 4 funções, o opt-aiNet superou o (q)opt-aiNet nestes experimentos, como pode ser visto na Figura 14 (esquerda). Nota-se uma grande melhora, principalmente nas funções unimodais e multimodais básicas, onde ocorreram 5 dos casos em que a melhora obteve significância estatística pelo teste de Wilcoxon.

No caso de $N_D = 50$, nota-se um comportamento semelhante ao obtido no caso com $N_D = 30$, em que o (q)opt-aiNet supera o opt-aiNet na maior parte dos casos, em especial nas funções unimodais e multimodais básicas que ocorre um maior número de significâncias estatísticas, como pode ser visto na análise da Tabela A.26.

Nos casos de 2 e 10 dimensões (Figura 13), nota-se que o algoritmo com mutação q-gaussiana supera o original em maior parte dos casos, mas com um número menor de funções com relevância estatística pelo teste de Wilcoxon.

A utilização da mutação com distribuição q-gaussiana no opt-aiNet, permitiu uma melhora interessante nas funções $f9$ e $f10$. Utilizando o operador de mutação com distribuição normal, o SOCAIN, supera o opt-aiNet nestes casos. A incorporação do operador de mutação q-gaussiana permitiu uma melhora substancial do *fitness* nestas funções. Desta forma, pode-se argumentar que o operador de mutação permitiu ao opt-aiNet uma melhor exploração do espaço de busca, possibilitando a fuga de ótimos locais, durante o processo de otimização, através do aumento do parâmetro q , o que permite a ocorrência de saltos mais longos no espaço de busca.

Nos experimentos comparando o SOCAIN com sua versão com mutação q-gaussiana, nota-se uma perda de desempenho em 14 das funções de *benchmark* com $N_D = 30$, muito embora em nenhum destes casos a perda ocorra com significância estatística. O (q)SOCAIN, por outro lado, supera o SOCAIN em 10 das funções, das quais 3 apresentaram relevância estatística, como pode ser visto na Tabela A.29.

Olhando agora para os casos com $N_D = 50$ (Tabela A.30), é verificado que a utilização da mutação q-gaussiana introduziu uma melhora em grande número de funções, totalizando 16, das quais 6 apresentaram melhores resultados com relevância estatística.

Uma melhora bastante consistente ocorre na função $f7$, que no SOCAIN, sem mutação q-gaussiana apresentou erro na escala de milhares. Porém, com a introdução da nova mutação o erro diminuiu para valores próximos de zero, tanto no caso de 30 como no de 50 dimensões.

Figura 15: Comparativo da frequências em que o (q)SOCAIN superou e foi superado pelo SOCAIN para $N_D = 2$ (a esquerda) e 10 (a direita).

Figura 16: Comparativo da frequências em que o (q)SOCAIN superou e foi superado pelo SOCAIN para $N_D = 30$ (a esquerda) e 50 (a direita).

Nos casos de 2 e 10 dimensões, a introdução da mutação q-gaussiana no SOCAIN apresentou melhoras mais significativas que para 30 e 50, notado pelo maior número de casos em que o novo algoritmo supera o primeiro, como pode ser visto na Figura 15 e Figura 16. Nas casos de 2 e 10 dimensões, ocorre uma perda com significância estatística apenas na função f_{25} .

Nota-se que nas funções rotacionadas ($f_3, f_8, f_{10}, f_{11}, f_{16}$ e f_{17}), com exceção da f_7 , não ocorreram melhoras significativas com a introdução da mutação q-gaussiana. Isto pode ser explicado pelo caráter anisotrópico do operador de mutação, que apresenta uma tendência maior de gerar soluções próximas aos eixos das coordenadas. Devido a

isso, em problemas rotacionados, é esperado que a busca utilizando tal operador apresente maiores dificuldades. Esta dificuldade pode ser superada através do uso da mutação q-gaussiana isotrópica, também apresentada em [TINÓS & YANG, 2011].

Figura 17: Comparativo da frequências em que o (q)SOCAIN superou e foi superado pelo (q)opt-aiNet para $N_D = 2$ (a esquerda) e 10 (a direita).

Figura 18: Comparativo da frequências em que o (q)SOCAIN superou e foi superado pelo (q)opt-aiNet para $N_D = 30$ (a esquerda) e 50 (a direita).

Por fim, na comparação entre o (q)opt-aiNet com o (q)SOCAIN, os resultados descritos na Tabela A.31, Tabela A.32, Tabela A.33 e Tabela A.34, indicam que, embora ambos tenham apresentado melhoras interessantes, a introdução da mutação q-gaussiana apresentou uma melhora mais significativa no primeiro que no segundo. A Figura 17 e Figura 18 mostram tal característica, onde o (q)SOCAIN apresenta um número menor de funções em que supera o (q)opt-aiNet, em comparação com o número de vezes que o SOCAIN supera o opt-aiNet (Figura 11 e Figura 12).

Em uma análise geral dos resultados, tanto do opt-aiNet quanto do SOCAIN, a introdução da mutação q-gaussiana apresentou melhora dos resultados principalmente nas funções unimodais e multimodais básicas. Tal fato indica que a mutação q-gaussiana possibilita uma melhor exploração do espaço. Porém, em alguns casos de funções rotacionadas, o operador de mutação q-gaussiana utilizado pode apresentar dificuldades devido a sua característica anisotrópica.

Nas composições de funções, a utilização da mutação q-gaussiana não apresentou melhora significativa na grande maioria dos casos. Isto se deve à grande complexidade destas funções, e ao número limitado de avaliações imposto como critério de parada.

6.1.4. Verificação de auto-organização da população

Para estudar o funcionamento do algoritmo de supressão do SOCAIN e (q)SOCAIN foram gerados a Figura 19 e Figura 20 com paralelos entre a evolução do *fitness* médio e o número de soluções eliminadas nos eventos de supressão. Nota-se no caso da função *f1*, uma função unimodal simples, que os eventos de supressão (que eliminam pelo menos uma solução) são mais frequentes que no caso da função *f9*, que é altamente multimodal.

Nota-se na Figura 19 que extinções maiores tendem a ocorrer na estagnação do *fitness* médio, indicando que o algoritmo está controlando a diversidade populacional neste caso. Analisando o resultado apresentado pela Figura 20 nota-se que não ocorrem grandes extinções neste caso. Isto ocorre devido a forma do espaço de busca, como pode ser visto na Figura 22. Desta forma, é provável que a população do algoritmo tenha alcançado diferentes ótimos com diferentes valores de *fitness*, tornando difícil o “encontro” de células com *fitness* suficientemente próximos para realizar uma supressão.

A dificuldade do algoritmo apresentar bons resultados em casos como da função *f9* é, em parte, devido ao tempo de execução que é curto para que ocorra a auto-organização, pois, como pode ser notado, o algoritmo não foi capaz de convergir no período. A população, por conter soluções espalhadas por diferentes regiões, tem dificuldade de realizar grandes supressões e, portanto, sofre poucas substituições. Isto não ocorre no caso da função *f1*, como pode ser visto na Figura 21, em que a forma do

espaço de busca, leva todas as soluções a convergirem mais rapidamente, permitindo a ocorrência de extinções maiores.

Figura 19: Comparativo do *fitness* médio e ocorrência de extinções do (q)SOCAIN para uma execução da função *f1* com $N_D = 50$.

Figura 20: Comparativo do *fitness* médio e ocorrência de extinções do (q)SOCAIN para uma execução da função *f9* com $N_D = 50$.

Figura 21: Superfície da função Esfera ($f1$) com $N_D=2$. As diferentes cores no gráfico são os valores de *fitness* como definidos na barra a direita.

Figura 22: Superfície da função Rastrigin ($f9$) com $N_D = 2$.

6.1.5. Avaliação do tempo de execução dos AISs

Os autores em [SUGANTHAN *et al.*, 2005] ainda descreveram um protocolo para análise de complexidade dos algoritmos em função do tempo de execução. A complexidade é calculada como:

$$T = (t_2 - t_1)/t_0 \quad (9)$$

onde t_0 é calculado como o tempo de execução do pseudo código apresentado no Quadro 7; t_1 é o tempo de 2×10^5 avaliações da função $f3$ e; t_2 o tempo médio de 5 execuções do algoritmo avaliado com número máximo de 2×10^5 avaliações otimizando a função $f3$. O índice de complexidade é calculado somente para a função $f3$ e somente para 10, 30 e 50 dimensões. t_0 , t_1 e t_2 , são calculados em milissegundos.

A Tabela 5, que apresenta o comparativo das complexidades entre o opt-aiNet e o SOCAIN, indica que este possui um tempo de execução menor que o primeiro, em geral. Isto pode ser visto pela diferença na complexidade que aumenta nos experimentos com maior dimensão. A Figura 23 apresenta uma visualização diferente dos resultados da análise de complexidade. Neste é notado que com o aumento do número de dimensões da função a diferença entre a complexidade do opt-aiNet e do SOCAIN é maior.

Quadro 7: Pseudo código do cálculo de t_0 .

```

1  Início
2      para i em 1:1000000 faça
3          x <- 5,55;
4          x <- x + x;
5          x <- x / 2;
6          x <- sqrt(x);
7          x <- ln(x);
8          x <- exp(x);
9          y <- x/x;
10     fim para

```

Tabela 5: Análise de complexidade do opt-aiNet e SOCAIN. t_0 , t_1 e t_2 em milissegundos.

	N_D	t_0	t_1	t_2	T
opt-aiNet	10	2,81E+02	1,09E+03	2,49E+03	4,96E+00
SOCAIN		2,63E+02	1,14E+03	2,44E+03	4,95E+00
opt-aiNet	30	2,64E+02	7,10E+03	1,01E+04	1,14E+01
SOCAIN		2,54E+02	6,96E+03	9,70E+03	1,08E+01
opt-aiNet	50	2,53E+02	1,82E+04	2,33E+04	2,02E+01
SOCAIN		2,81E+02	1,80E+04	2,23E+04	1,54E+01

Figura 23: Representação gráfica da análise de complexidade do opt-aiNet e SOCAIN.

O SOCAIN apresenta menor complexidade, pois, as comparações para a realização da supressão não ocorrem em todos os possíveis pares de soluções. Adicionalmente a comparação é baseada somente em uma comparação dos valores de *fitness* obtidos, o que é menos computacionalmente custoso que o calculo da distância euclidiana.

Por outro lado, o SOCAIN apresenta um aumento de complexidade dependendo do intervalo de supressão g_s . Se o intervalo for pequeno, ocorreria um número maior de eventos de supressão, aumentando assim a complexidade. Porém, a utilização de pequenos intervalos de supressão tem tendência de afetar negativamente o desempenho do algoritmo pois teria menos tempo para ocorrer a convergência aumentando a probabilidade de supressões prematuras.

6.1.6. Comparação com outros algoritmos

Finalmente, os AISs propostos foram comparados com os resultados dos algoritmos de otimização com codificação real apresentados na competição proposta em [SUGANTHAN *et al.*, 2005]. Os algoritmos comparados são: BLX-GL50 [GARCIA-MARÍNEZ & LOZANO, 2005], BLX-MA [MOLINA *et al.*, 2005], CoEVO [POŠÍK *et al.*, 2005], DE [RÖNKKÖNEN *et al.*, 2005], DMS-L-PSO [LIANG & SUGANTHAN, 2005], EDA [YUAN & GALLAGHER, 2005], G-CMA-ES [AUGER & HANSEN, 2005], K-PCX [SINHA *et al.*, 2005], L-CMA-ES [AUGER & HANSEN, 2005], L-SaDE [QIN & SUGANTHAN, 2005] e SPC-PNX [BALLESTER *et al.* 2005].

Figura 24: Comparativo do erro médio dos algoritmos de otimização para $N_D=10$.

Nota-se na Figura 24, que apresenta os erros médios obtidos no *benchmark* das funções com $N_D=10$, que tanto o (q)opt-aiNet quanto o (q)SOCAIN apresentaram resultados notavelmente piores que os demais algoritmos em geral nas funções unimodais e multimodais básicas (*f1-f12*). Nas composições de funções, o (q)opt-aiNet se mostrou melhor que os demais algoritmos, embora tenha apresentados valores altos de erro.

Nos casos com $N_D=30$ (Figura 25), ocorreu um comportamento semelhante ao caso de $N_D=10$ nas funções unimodais. Ambos, o (q)SOCAIN e (q)opt-aiNet, apresentaram resultados piores que os demais algoritmos, com exceção do EDA. Nas demais funções nenhum dos algoritmos propostos apresentou comportamento interessante.

Figura 25: Comparativo do erro médio dos algoritmos de otimização para $N_D=30$.

Figura 26: Comparativo do erro médio dos algoritmos de otimização para $N_D=50$.

A Figura 26, com os resultados das funções com $N_D=50$ o (q)SOCAIN e (q) opt-aiNet apresentaram piores resultados que o G-CMA-ES e L-CMA-ES nas funções unimodais com exceção da função $f4$. Dentre as multimodais básicas nota-se que os AISs apresentaram erros médios significativamente maiores nas funções $f5$ - $f7$. Nas composições de funções, uma diferença aparente na função $f16$ em que os AISs apresentaram pior desempenho, e nas funções $f23$ e $f24$ em que o (q)opt-aiNet apresentou desempenho melhor que os demais algoritmos. Resultados com $N_D=50$ para os demais algoritmos não estão disponíveis nas publicações citadas.

Nota-se que nenhum dos algoritmos propostos na competição de [SUGANTHAN *et al.*, 2005] apresentam bons resultados nas composições de funções em geral. Isto se deve, como citado anteriormente, a grande complexidade apresentada pela superfície de busca destas funções e o baixo número de avaliações. Nos AISs em especial, o limite no número de avaliações gera uma dificuldade na otimização devido ao grande número de clones gerados, resultando em muitas avaliações. Adicionalmente, no caso do SOCAIN, os efeitos benéficos da auto-organização são geralmente notados para um número maior de gerações em problemas de maior complexidade.

6.2. EXPERIMENTOS COM DOCKING

Os algoritmos SOCAIN e (q)SOCAIN foram comparados com uma implementação do AG apresentado em [GARDINER *et al.*, 2001] no problema de *docking* descrito no subcapítulo 5.4. As implementações do AG, e da função de *fitness* foram testadas no *docking* do complexo alfa-quimotripsina A (de *Bos taurus*) – inibidor ovomucóide (de *Meleagris gallopavo*) , de código pdb 1CHO (Figura 2), que foi utilizado em um dos experimentos descritos em [GARDINER *et al.*, 2001]. Em referências futuras este complexo será designado por seu código PDB.

Os parâmetros utilizados no AG são os mesmos definidos nos experimentos em [GARDINER *et al.*, 2001] e estes foram utilizados para todos os experimentos:

- Tamanho da população: 400 indivíduos;
- Porcentagem de indivíduos substituídos: 5%;
- Diâmetro do nicho: 2Å;
- Número de nichos: 5.

Os parâmetros da função de *fitness* foram:

- Tamanho da grade: 2Å de aresta;
- Tolerância entre normais: $\pm\pi/6$;
- Peso da penalidade por colisão w_c : 3.

Para que a comparação seja justa entre os algoritmos, o critério de parada utilizado nos testes, foi aquele descrito em [SUGANTHAN *et al.*, 2005] ao invés do número de gerações. Desta forma, os algoritmos foram executados por 5×10^5 execuções.

Testes iniciais foram realizados por um número pequeno de avaliações, para validação do AG e do modelo de *docking* como descrito em [GARDINER *et al.*, 2001]. Estes testes são descritos no Apêndice A.3. Embora nos testes iniciais tenha sido aplicado o SOCOpt-aiNet, no restante dos experimentos envolvendo *docking* foi utilizado o SOCAIN, que apresentou, em geral, melhor desempenho nos experimentos apresentados anteriormente (subcapítulo 6.1).

Como o AG foi comparado com o SOCAIN, o critério de parada utilizado foi baseado no número de avaliações da função de *score* (*fitness*). Desta forma, o intervalo em que ocorrem os reinícios pelo algoritmo de restrição de nicho foi definido como:

$$N_I = \frac{N_A}{\overline{N_A}(N_N+1)} \quad (10)$$

na qual N_A é o número máximo de avaliações, $\overline{N_A}$ é o número médio de avaliações por geração e N_N é o número de nichos. No caso do AG, o número médio de avaliações utilizado foi simplesmente 20 (5% de 400 indivíduos), e no caso do SOCAIN foi utilizado 200 (10 clones de cada uma das 20 células de memória).

Após a validação dos algoritmos, foram realizados testes por um número maior de avaliações buscando verificar o desempenho dos algoritmos. Os testes foram realizados comparando os algoritmos com e sem a aplicação do operador de restrição de nicho. Testes também foram realizados no SOCAIN com mutação q-gaussiana. Por fim os algoritmos que apresentaram melhor desempenho foram testados na otimização de um outro complexo proteico.

6.2.1. Estudo do efeito da restrição de nicho no SOCAIN

O operador de restrição de nicho utilizado pelo AG pode oferecer um benefício interessante para o algoritmo proposto, limitando o espaço de busca. Considerando tal possibilidade, foram construídas versões do AG e do SOCAIN, com e sem a aplicação do algoritmo de restrição de nicho.

O operador de restrição de nicho no SOCAIN foi aplicado como no AG: a cada início de geração é feita uma verificação se deve ou não ocorrer o reinício da população. Em caso positivo, todas as soluções correntes são eliminadas e então a população é reiniciada.

Tabela 6: Comparativo dos resultados de *fitness* obtidos de cinco execuções do AG e SOCAIN com e sem restrição de nicho.

Algoritmo	Média	Desvio padrão	Mediana	Máximo	p-value
GA	4,336E+03	6,088E+02	4,088E+03	5,300E+03	1,250E-01
(n)GA	5,227E+03	4,654E+02	5,307E+03	5,874E+03	
SOCAIN	5,646E+03	8,073E+02	5,289E+03	6,673E+03	8,125E-01
(n)SOCAIN	5,493E+03	1,924E+02	5,377E+03	5,797E+03	

A Tabela 6 apresenta os resultados comparativos de cinco execuções de cada um dos algoritmos, isto é, do AG e do SOCAIN com (identificados pela letra ‘n’) e sem o algoritmo de restrição de nicho. Observando os resultados, nota-se que as versões utilizando restrição de nicho apresentaram mediana melhor, porém o teste estatístico de Wilcoxon, indica que não houve diferença estatística. Por outro lado, no comparativo do SOCAIN padrão com sua versão com restrição de nicho, o primeiro apresentou maior média, e maior valor máximo das cinco execuções.

Na Figura 27 (cima), é possível verificar que o AG tem uma forte tendência de convergir a população para um único ótimo local, identificado pelos crescimentos abruptos do *fitness* médio. Isto justifica o melhor desempenho apresentado no experimento com utilização da restrição de nicho que, substituindo todos os indivíduos por imigrantes aleatórios, aumenta a diversidade da população, permitindo uma melhor exploração do espaço de busca. Adicionalmente, o procedimento de restrição de nicho remove uma região do espaço de busca em volta da melhor solução, impedindo a população de convergir para o mesmo ótimo.

Figura 27: Comparativo das melhores execuções do AG de [GARDINER et al. 2001] sem (cima) e com restrição de nicho (baixo).

Figura 28: Comparativo das melhores execuções do SOCAIN sem (acima) e com restrição de nicho (abaixo).

Por outro lado, na Figura 28 do SOCAIN, não nota-se uma grande vantagem na utilização da restrição de nicho, pelo tempo especificado. No experimento utilizando restrição de nicho (baixo), nota-se que o *fitness* médio ainda está em crescimento, o que indica que a população ainda está em desenvolvimento.

Pode-se argumentar que no caso do SOCAIN é menos vantajosa a utilização da restrição de nicho, uma vez que a natureza do algoritmo leva à exploração de múltiplas regiões paralelamente, levando a um tempo maior para que ocorra a convergência do *fitness* médio. Não ocorrendo a convergência até o reinício, o operador de restrição de nicho pode eliminar do espaço de busca uma região potencialmente promissora. Desta forma, a utilização da restrição de nicho pode ser interessante com um intervalo maior para a ocorrência do reinício da população.

Por outro lado, o SOCAIN utiliza o operador de supressão que busca controlar a diversidade da população sem a necessidade de uma intervenção explícita, como o reinício da população, para a geração aumentar a diversidade.

6.2.2. Aplicação do (q)SOCAIN no *docking*

Em seguida, os resultados do AG padrão (com restrição de nicho) e do SOCAIN padrão no *docking* do complexo 1CHO, foram comparados com os obtidos através do (q)SOCAIN. Os resultados são apresentados na Tabela 7.

Tabela 7: Comparativo dos resultados obtidos de cinco execuções do (q)SOCAIN com o (n)GA e SOCAIN na otimização do complexo 1CHO.

Algoritmo	Média	Desvio padrão	Mediana	Máximo	p-valor
(q)SOCAIN	6,019E+03	5,030E+02	6,027E+03	6,697E+03	6,250E-02
(n)GA	5,227E+03	4,654E+02	5,307E+03	5,874E+03	
(q)SOCAIN	6,019E+03	5,030E+02	6,027E+03	6,697E+03	6,250E-01
SOCAIN	5,646E+03	8,073E+02	5,289E+03	6,673E+03	

A Tabela 7 demonstra que o (q)SOCAIN é superior a ambos, o AG e o SOCAIN padrão, indicando que, com o operador de mutação q-Gaussiano, o SOCAIN apresentou um melhor compromisso entre aumento da diversidade e exploração local do espaço. Isto é devido ao operador de mutação q-Gaussiano que, como visto nos experimentos de *benchmark*, apresentam em geral melhor capacidade de exploração do espaço de busca, em comparação ao operador de mutação padrão.

Os resultados do (q)SOCAIN e SOCAIN são comparados na Figura 29. Nesta é possível estudar o comportamento dos algoritmos no *docking* do complexo 1CHO.

Verifica-se que o (q)SOCAIN converge a um *score* médio mais alto. Mais um indicativo de que a mutação q-gaussiana oferece uma vantagem na exploração do espaço de busca.

Figura 29: Comparativo das melhores execuções do (q)SOCAIN (acima) e SOCAIN (abaixo) na otimização do complexo 1CHO.

A Figura 30, Figura 31 e Figura 32 apresentam os complexos proteicos formados na execução com maior *score* do AG, SOCAIN e (q)SOCAIN respectivamente, com um comparativo miniaturizado do complexo nativo na mesma orientação. No caso do AG (Figura 32) existe uma porção considerável da proteína ligante permeando a superfície da receptora, o que penaliza o *score* final justificando o valor baixo obtido comparado ao SOCAIN (Figura 31) e (q)SOCAIN (Figura 30). Estes, que obtiveram valor de *fitness* (*score*) maiores, possuem menor área de superfície de colisão com o ligante.

Embora o (q)SOCAIN tenha apresentado melhores resultados neste complexo o *fitness* ainda é baixo comparado com o complexo nativo (*score* de 8729). O sítio de ligação da proteína receptora tem forma de um *pocket* (uma reentrância na molécula). Tal característica torna a interação extremamente específica. A partir da conformação nativa, uma pequena alteração na posição ou orientação da molécula ligante, levaria a um grande aumento na superfície de colisão.

Figura 30: Melhor resultado do (q)SOCAIN para o complexo 1CHO. Acima uma miniatura do complexo nativo com mesma orientação do receptor.

Figura 31: Melhor resultado do SOCAIN para o complexo 1CHO.

Figura 32: Melhor resultado do AG de [GARDINER *et al.*, 2001] para o complexo 1CHO.

6.2.3. Experimentos com o complexo proteico 1AVX

Os experimentos envolvendo o *docking* de proteínas descritos até então foram referentes somente ao complexo 1CHO. Por fim, o AG, o SOCAIN e o (q)SOCAIN foram utilizados no *docking* do complexo tripsina (de *Sus scrofa*) – inibidor de tripsina (de *Glycine max*), de código PDB 1AVX (Figura 33). Este complexo foi selecionado de [MINTSERIS, *et al.*, 2005; HWANG, *et al.*, 2008] que apresentam uma lista de complexos proteicos para o *benchmark* de algoritmos de *docking*.

O complexo 1AVX é consideravelmente maior que o 1CHO. Enquanto este possui 297 aminoácidos aquele contém um total de 400, o que leva a um tempo de execução consideravelmente maior. A Tabela 8 apresenta os tempos médios de execução de uma avaliação dos complexos 1CHO e 1AVX.

Figura 33: Complexo (em azul) tripsina A (de *Sus scrofa*) – (em verde) inibidor de tripsina (de *Glycine max*), código pdb 1AVX.

O tempo médio de avaliação do *score* é calculado com base em 25 avaliações da função, e o tempo total é calculado com base na média obtida. Deste modo, é importante ressaltar que o tempo total de 5×10^5 avaliações, apresentado na Tabela 8, não reflete o tempo total de execução dos algoritmos de otimização (SOCAIN e AG), pois não considera o tempo da aplicação dos operadores de seleção e reprodução.

Tabela 8: Tempo médio de avaliação dos complexos 1CHO e 1AVX.

Complexo	Média*	Desvio padrão	Tempo total**
1CHO	1,188E+00	9,125E-02	6,88
1AVX	4,431E+00	1,645E-01	25,64

*Tempo médio em segundos; **Tempo total médio de 5×10^5 avaliações em dias;

Uma descrição geral dos resultados do *docking* do complexo 1AVX são apresentadas na Tabela 9, com um comparativo entre o SOCAIN e o (q)SOCAIN. Nota-se que este apresentou um *score* médio maior que o primeiro, como visto nos experimentos envolvendo o complexo 1CHO. O maior *score* obtido foi, também, pelo (q)SOCAIN, com um valor de 8099, enquanto o SOCAIN obteve 7803. A fins de comparação foi computado o *score* do complexo 1AVX em seu estado nativo, sendo obtido um valor de 10090.

Tabela 9: Comparativo dos resultados obtidos de cinco execuções do (q)SOCAIN e SOCAIN na otimização do complexo 1AVX.

Algoritmo	Média	Desvio padrão	Mediana	Máximo	p-valor
(q)SOCAIN	7,237E+03	6,224E+02	6,941E+03	8,099E+03	6,250E-02
SOCAIN	6,792E+03	5,746E+02	6,592E+03	7,803E+03	

Uma comparação de duas execuções, uma do (q)SOCAIN e uma do SOCAIN, é feita na Figura 34. Neste caso, é possível verificar um comportamento similar aquele descrito na Figura 29, em que o (q)SOCAIN converge em um valor maior de *fitness* médio.

Figura 34: Comparativo das melhores execuções do (q)SOCAIN (acima) e SOCAIN (abaixo) na otimização do complexo 1AVX.

A Figura 35 e Figura 36 apresentam os complexos de maior *score* formados nos experimentos utilizando o SOCAIN e (q)SOCAIN, respectivamente. Nota-se, no caso deste (Figura 35), que a melhor solução obtida gera um complexo com as moléculas próximas aos sítios de ligação, porém as moléculas não estão corretamente orientadas. Mas ainda assim, a conformação obteve um valor de *score* relativamente alto. A Figura 36, com o complexo gerado pelo SOCAIN, mostra que, embora o algoritmo tenha obtido um resultado com moléculas distantes do sítio ativo, ele foi capaz de minimizar colisões entre as moléculas.

Figura 35: Melhor resultado do (q)SOCAIN para o complexo 1AVX.

Figura 36: Melhor resultado do SOCAIN para o complexo 1AVX.

Para melhor visualizar a proximidade do complexo gerado pelo (q)SOCAIN foi gerada a Figura 37. Nela, a molécula em vermelho representa o complexo com orientação nativa, e em verde com a orientação obtida pela otimização através do (q)SOCAIN. Pode-se verificar na figura que a posição da molécula é aproximadamente rotacionada, visto pela posição das estruturas circuladas – as mesmas estruturas, uma do complexo nativo e outra do complexo obtido pelo (q)SOCAIN.

Figura 37: Comparação da conformação nativa (em vermelho) com o de maior *score* do (q)SOCAIN (em verde).

Embora o melhor complexo encontrado tenha obtido uma conformação próxima aos sítios de ligação, os demais complexos obtidos não apresentaram resultados com tanta proximidade. Uma visão geral dos complexos obtidos nos experimentos do (q)SOCAIN podem ser vistos na Figura 38 que contém ilustrações dos melhores complexos obtidos em cada execução. A obtenção de diferentes complexos com ligantes em diferentes orientações indica que o algoritmo é capaz de gerar soluções nas diferentes posições e orientações, possibilitando explorar diferentes regiões do espaço de busca.

Figura 38: Conformações obtidas pelos experimentos do *docking* do complexo 1AVX através do (q)SOCAIN.

Nota-se com os experimentos que o *docking* é um problema de otimização de grande complexidade. Na Figura 39 é possível ver a interação das superfícies das moléculas ligante e receptora dos complexos 1CHO e 1AVX em seu estado nativo. Nota-se que o sítio de ligação do 1AVX, como o do 1CHO, também tem a forma de um *pocket* conferindo uma grande complementaridade entre as superfícies maximizando as superfícies de contato.

Figura 39: Detalhe dos sítios de ligação do complexo nativo de 1CHO (esquerda) e 1AVX (direita). O ligante está em pontilhado verde e em vermelho o receptor.

7. CONCLUSÕES

Os experimentos realizados com as funções de *benchmark* apresentadas em [SUGANTHAN *et al.*, 2005] permitiram comparar as implementações do opt-aiNet e dos algoritmos propostos.

Os resultados obtidos dos experimentos com os algoritmos propostos apresentaram melhor exploração do espaço de busca (*exploration*), encontrando soluções intermediárias. Porém, o fraco desempenho nas funções unimodais, é um indicativo de que o algoritmo apresenta dificuldades em realizar uma busca local próximo a soluções já existentes (*exploitation*).

Considerando que os operadores de seleção clonal e maturação de afinidade são aplicados sem alterações em ambos algoritmos, pode-se argumentar que a dificuldade de realizar buscas locais vem do método de supressão proposto, que deve estar suprimindo soluções prematuramente. Para estudar este problema, foram realizados testes com diferentes parâmetros, que indicaram que um aumento do intervalo em que ocorrem as supressões pode produzir melhores resultados. Isto foi verificado nos experimentos de *benchmark*, que apresentou melhores resultados nas funções multimodais, especialmente nas de maior complexidade (com 30 e 50 dimensões).

Embora melhoras tenham ocorrido após a seleção de parâmetros, o desempenho do SOCopt-aiNet continuou apresentando problemas de supressão precoce, que pode ser visto na Figura 10. Pensando nisto, testes utilizando um novo algoritmo foram realizados buscando sanar algumas das dificuldades verificadas nos resultados do SOCopt-aiNet.

O SOCAIN foi desenvolvido para controlar a diversidade de forma auto-organizada: extinções maiores tendem a ocorrer quando a população apresenta baixa diversidade. Esta característica deve permitir que o algoritmo seja capaz de manter a diversidade populacional sem uma intervenção explícita, causando grandes alterações na população em casos de baixa diversidade, o que pode ser visto na Figura 19.

O algoritmo de supressão proposto substitui soluções aleatórias no caso de encontro de um par de células com *fitness* semelhante, e enquanto não ocorrem supressões, não há adição de soluções aleatórias. Desta forma, em um caso especial estudado, o da função *f9*, o algoritmo de supressão proposto apresentou dificuldade em

controlar a diversidade no caso de estagnação do *fitness* populacional. Isto ocorre devido ao grande número de ótimos locais com diferentes valores de *fitness*.

Pode se dizer, portanto, que o algoritmo de adição de soluções aleatórias e controle do tamanho populacional do SOCAIN, é conservador em casos altamente multimodais. Como a otimização dos AISs não contam com um operador de cruzamento, a população tende a convergir para diferentes ótimos, com diferentes valores de *fitness*, o que dificulta a ocorrência de grandes supressões mesmo com a estabilização do *fitness* médio da população. Considerando tais observações, em trabalhos futuros deve ser estudada uma política menos conservadora, isto é, que permita a adição de soluções aleatórias mesmo não ocorrendo grandes supressões.

O SOCAIN, embora tenha sido superado pelo opt-aiNet original na maior parte dos casos de teste, apresentou um desempenho interessante em alguns casos específicos como, por exemplo, na própria função *f9* e as funções com ruído.

Foi proposta ainda a utilização de um operador de mutação com distribuição q-gaussiana [TINÓS & YANG, 2010] em AISs. Os resultados obtidos indicaram melhoras nos resultados em grande parte das funções tanto para o opt-aiNet como para o SOCAIN. Porém, o opt-aiNet obteve melhoras mais significativas com a utilização do novo operador, quando comparado com os benefícios obtidos pelo SOCAIN. Estas melhoras indicam que a utilização da distribuição q-gaussiana permitiu que a população escapasse de ótimos locais e explorasse melhor o espaço de busca.

Um problema foi notado na utilização do SOCAIN em que nas funções multimodais rotacionadas, ocorreu uma perda de desempenho com a incorporação da mutação q-gaussiana. Isto se deve à natureza anisotrópica do operador que possui um vício para geração de valores nos eixos das coordenadas. Uma solução para este problema seria a utilização de uma distribuição isotrópica, como proposto em [TINÓS & YANG, 2011].

Na avaliação de complexidade dos algoritmos SOCAIN e opt-aiNet, foi verificado que o primeiro apresenta menor tempo de execução médio, de acordo com os critérios fornecidos em [SUGANTHAN *et al.*, 2005]. O que sugere que o algoritmo de supressão auto-organizável proposto tem menor complexidade que a técnica aplicada pelo opt-aiNet.

É importante notar que as técnicas de supressão desenvolvidas neste trabalho podem ser aplicadas a outros modelos de AISs pois independe da codificação. A mutação q-gaussiana, por outro lado, pode somente ser utilizada em problemas de otimização real.

A aplicação do SOCAIN no *docking* apresentou resultados interessantes quando comparados com o AG. Embora divergências possam existir em comparação com a implementação de [GARDINER *et al.*, 2001], os resultados indicaram que a implementação do AG e da função de *fitness* são válidos na utilização para a realização e avaliação de *docking* de complexos moleculares. Isto pôde ser verificado pelas conformações obtidas na Figura 32 e no experimento inicial descrito no Apêndice A.3, Figura A.3, que demonstram que os algoritmos foram capazes de encontrar soluções, mesmo que locais, capazes de evitar as penalidades por colisão.

Para o complexo 1CHO, nota-se que o SOCAIN apresentou melhores resultados, embora sem relevância estatística (possivelmente devido ao número pequeno de execuções), em comparação à implementação do AG, visto na Tabela 6, que apresenta o comparativo de cinco execuções de cada algoritmo. Adicionalmente foram realizados testes utilizando o (q)SOCAIN que utiliza o operador de mutação q-Gaussiano, que apresentou resultados mais promissores, especialmente no *docking* do complexo 1AVX, onde foi obtido um complexo com uma conformação em que as moléculas estão ligadas próximas de seus sítios de ligação.

O problema de *docking* por complementaridade de superfícies, descrito neste trabalho, pode ser considerado um problema de otimização complexo. Nos experimentos realizados pôde-se verificar que existe uma exigência muito grande de complementaridade para maximizar a superfície de contato e obter um *score* próximo daquele obtido pela conformação nativa. Na Figura 39 pode-se visualizar a forma do sítio de ligação da molécula receptora, que em ambos os casos se localizam em um *pocket*. Desta forma, uma pequena alteração na orientação da molécula ligante leva a uma diminuição drástica na superfície de contato e um provável aumento da região de colisão. Tal característica, no espaço de busca, leva a uma região de pico no ótimo global com um aumento repentino de um valor baixo de *score* para um valor muito alto.

Continuando os estudos no *docking* devem ser implementadas novas funções de avaliação, mais realistas, envolvendo características energéticas das moléculas, e enfim adicionando flexibilidade. Neste trabalho foi realizada uma abordagem *ab initio*, sem

conhecimento prévio em relação às moléculas em estudos. Outra possibilidade seria o estudo de metodologias para filtrar e diminuir o espaço de busca através da adição de conhecimento.

Com o desenvolvimento de uma tecnologia suficientemente sofisticada para o *docking* molecular, um problema desafiador para avaliar o algoritmo é a submissão aos testes apresentados pelo CAPRI (*Critical Assessment of PRediction of Interactions*) [JANIN *et al.*, 2003], que têm o objetivo de avaliar algoritmos de *docking* através do oferecimento em turnos de estruturas preditas por métodos laboratoriais, porém nunca antes publicadas (*i.e.*, as estruturas não são conhecidas publicamente). Desta forma, aos algoritmos que serão avaliados são passadas somente as estruturas das moléculas separadas, e no final do experimento, a estrutura do complexo é publicada e os resultados são, então, avaliados. Tais características impossibilita que se obtenha conhecimento prévio das estruturas da proteínas exigindo que os algoritmos façam o *dockgin ab initio*.

8. REFERÊNCIAS BIBLIOGRÁFICAS

- AUGER, A.; KERN, S. & HANSEN, N. (2005), “A Restart CMA Evolution Strategy with Increasing Population Size”, *Special Session on Real-Parameter Optimization at CEC-05*, Edinburgh, UK.
- AUGER, A.; KERN, S. & HANSEN, N. (2005), “Performance Evaluation of an Advanced Local Search Evolutionary Algorithm”, *Special Session on Real-Parameter Optimization at CEC-05*, Edinburgh, UK.
- AZEVEDO JR., W. F. & DIAS, R. (2008), “Computational methods for calculation of ligand-binding affinity”, *Current Drug Targets*, Vol. 9, p. 1031-1039.
- BAK, P. (1997), “*How nature works: the science of self-organized criticality*”, Oxford University Press.
- BALLESTER, P. J.; STEPHENSON, J.; CARTER, J. N. & GALLAGHER, K. (2005), “Real-Parameter Optimization Performance Study on the CEC-2005 benchmark with SPC-PNX”, *Special Session on Real-Parameter Optimization at CEC-05*, Edinburgh, UK.
- BÖHM, H. J. & STAHL, M. (2002), “The use of scoring functions in drug discovery applications”, *Reviews in computational chemistry*, Vol. 18, p. 41-87.
- BOISSON, J.-C.; JOURDAN, L.; TALBI, E.-G. & HORVATH, D. (2008), “Parallel multi-objective algorithm for the molecular docking problem”, *Symposium on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB'08)*, Vol. 15, p. 187-194.
- BROOKS, B. R.; BRUCCOLERI, R. E.; OLAFSON, B. D.; STATES, D. J.; SWAMINATHAN, S. & KARPLUS, M. (1983), “CHARMM – a program for macromolecular energy, minimization, and dynamics calculations”, *Journal of Computational Chemistry*, Vol. 4(2), p. 187-217.
- BROOKS, B.R.; BROOKS III, C. L.; MACKERELL, A. D.; NILSSON, L.; PETRELLA, R. J.; ROUX, B.; WON, Y.; ARCHONTIS, C.; BARTELS, C.; BORESCH, C.; CAFLISCH, A.; CAVES, L.; CUI, Q.; DINNER, A. R.; FEIG, M.; FISCHER, S.; GAO, J.; GODOSCEK, M.; IM, W.; KUCZERA, K.; LAZARIDIS, T.; MA, J.; OVCHINNIKOV, V.; PACI, E.; PASTOR, R. W.;

- POST, C. B.; PU, J. Z.; SCHAEFER, M.; TIDOR, B.; VENABLE R. M.; WOODCOCK, H. L.; WU, X.; YANG, W.; YORK, D. M. & KARPLUS, M. (2009), "CHARMM: The Biomolecular simulation Program". *Journal of Computational Chemistry*, Vol. 30, p. 1545-1615
- CASE, D. A.; CHEATHAM III, T. E.; DARDEN, H.; GOHLKE, H.; LUO, R.; MERZ JR., K. M.; ONUFRIEV, A.; SIMMERLING, C.; WANG, B. & WOODS, R. (2005), "The Amber biomolecular simulation programs". *Journal of Computatoinal Chemistry*, Vol. 26, p. 1668-1688.
- CASTRO, L. N. & TIMMIS, J. I. (2002), "An artificial immune network for multimodal function optimization", *Evolutionary Computation*, Vol. 12, p. 699-704.
- CASTRO, L. N. & TIMMIS, J. I. (2003), "Artificial immune systems as a novel soft computing paradigm", *Soft Computing*, Vol. 7, p. 526-544.
- CASTRO, L. N. & VON ZUBEN, F. J. (2002), "Learning and optimization using the clonal selection principle", *Evolutionary Computation*, Vol. 6(3), p. 239-251.
- CHEN, R.; MINTESRIS, J.; JANIN, J. & WENG, Z. (2003), "Protein-protein docking benchmark", *PROTEINS: Structure, Function, and Genetics* Vol. 52(1), p.88-91.
- COBB, H. G. & GREFENSTETTE, J. J. (1993), "Genetic algorithms for tracking changing environments". In: *Proceedings of the 5th Int. Conf. on Genetic Algorithms*. p. 523–530.
- CONNOLLY, M. L. (1983), "Solvent-accessible surfaces of proteins and nucleic acids", *Science*, Vol. 221, p. 709-713.
- CONNOLLY, M. L. (1993), "The molecular surface package", *J. Mol. Graphics*, Vol. 11, p. 139-141.
- EIBEN, A. E.; HINTERDING, R. & MICHALEWICZ, Z. (1999), "Parameter control in evolutionary Algorithms", *Evolutionary Computation*, Vol. 3(2), p. 124–141.
- FERRARA, P.; GOHLKE, H.; PRICE, D. J.; KLEBE, G. & BROOKS III, C. L. (2004), "Assessing scoring functions for protein-ligand interactions", *J. Med. Chem.*, Vol. 47, p. 3032-3047.

- FERNANDES, C. M.; MERELO, J. J.; RAMOS, V.; ROSA, A. C. (2008), “A self-organized criticality mutation operator for dynamic optimization problems”, *Proceedings of 10th Genetic and Evolutionary Computation*, Atlanta, Georgia, USA, p. 937-944.
- FOGEL, G. B. & CORNE, D. W. (2003), “*Evolutionary computation in bioinformatics*”, Elsevier Science, ISBN 1-55860-797-8.
- GARCIA-MARTÍNEZ, C. & LOZANO, M. (2005), “Híbrido real-coded genetic algorithm with female and male differentiation”, *Special Session on Real-Parameter Optimization at CEC-05*, Edinburgh, UK.
- GARDINER, E. J.; WILLETT, P. & ARTYMIUK, P. J. (2001), “Protein docking using a genetic algorithm”, *PROTEINS: structure, function and genetics*, Vol. 44, p. 44-56.
- GOLDBERG, D. E. (1989), “*Genetic algorithms in search, optimization and machine learning*”, Addison-Wesley Longman Publishing. ISBN 0201157675.
- HEBERLÉ, G. & AZEVEDO JR., W. F. (2011), “Bio-inspired algorithms applied to molecular docking simulations”, *Current medical chemistry*, Vol. 18, p. 1339-1352.
- HOLLAND, J. H. (1975), “Adaptation in natural and artificial systems”, University of Michigan Press.
- HWANG, H.; PIERCE, B.; MINTSERIS, J.; JANIN, J. & WENG, Z. (2008), “Protein-protein docking benchmark version 3.0”, *PROTEINS: Structure, Function, and Genetics* Vol. 73(3), p.705-709.
- JANIN, J. ; HENRICK, K.; MOULT, j.; EYCK, L. T.; STERNBERG, M. J. E.; VAJDA, S. VAKSER, I. & WODAK, S. J. (2003), “CAPRI: A Critical Assessment of PRedicted Interactions”. *PROTEINS: structure, function and genetics*, Vol. 52, p. 2-9.
- JANSON, S.; MERKLE, D. & MIDDENDORF, M. (2007), “Molecular docking with multi-objective particle swarm optimization”, *Applied Soft Computing*, Vol. 8, p. 666-675.

- KITCHEN, D. B.; DECORNEZ, H.; FURR, J. R. & BAJORATH, J. (2004), “Docking and scoring in virtual screening for drug discovery: methods and applications” *Nature Reviews: Drug Discovery*, Vol. 3, p. 935-949.
- KRINK, T., RICKERS, P., THOMSEN, R. (2000) “Applying self-organized criticality to evolutionary algorithms”, *Proceedings of the 6th International Conference on Parallel Problem Solving*, London, UK, 375-384.
- KRINK, T. & THOMSEN, R. (2001), “Self-organized criticality and mass extinction in evolutionary algorithms”, *Proceedings of the 2001 Congress on Evolutionary Computation*, Seoul, South Korea, Vol. 2, p. 1155-1161.
- LIANG, J. J. & SUGANTHAN, P. N. (2005), “Dynamic Multi-Swarm Particle Swarm Optimizer with Local Search”, *Special Session on Real-Parameter Optimization at CEC-05*, Edinburgh, UK.
- MINTSERIS, J.; WIEHE, K.; PIERCE, B.; ANDERSON, R.; CHEN, R.; JANIN, J.; & WENG, Z. (2005), “Protein-protein *docking* benchmark 2.0: an update”, *PROTEINS: Structure, Function, and Genetics* Vol. 60(2), p.214-216.
- MITCHELL, M. (1996), “*An introduction to genetic algorithms*”, MIT Press.
- MOLINA, D.; HERRERA, F. & LOZANO (2005), “Adaptive Local Search Parameters for Real-Coded Memetic Algorithms”, *Special Session on Real-Parameter Optimization at CEC-05*, Edinburgh, UK.
- MORRIS, G. M.; GOODSELL, D. S.; HALLIDAY, R. S.; HUEY, R.; HART, W. E.; BELEW, R. K. & OLSON, A. J. (1998), “Automated docking using a lamarckian genetic algorithm and an empirical binding free energy function”, *Journal of computational Chemistry*, Vol. 19(14), p. 1639-1662.
- MORRIS, G. M.; HUEY, R.; LINDSTROM, W.; SANNER, M. F.; BELEW, R. K.; GOODSELL, D. S. & OLSON, A. J. (2009), “AutoDock4 and AutoDockTools4: Automated docking with selective receptor flexibility”. *Journal of Computational Chemistry*, Vol. 30, p. 2785-2791.
- PAL, S. K., BANDYOPADHYAY, S. & RAY, S. S. (2006), “Evolutionary computation in bioinformatics: a review”, *IEEE Trans on Systems, Man, and Cybernetics: Applications and Reviews*, Vol. 36(3), p. 601-615.

- PETTERSEN, E. F.; GODDARD, T. D.; HUGANG, C. C.; COUCH, G. S.; GREENBLATT, D. M.; MENG, E. C. & FERRIN, T. E. (2004), “UCSF Chimera – A visualization system for exploratory research and analysis” *J. Comput Chem.* Vol. 25(13), p. 1605-1612.
- POŠÍK, P. (2005), “Real Real Parameter Optimisation Using Mutation Step Co-evolution”, *Special Session on Real-Parameter Optimization at CEC-05*, Edinburgh, UK.
- QIN, A. K. & SUGANTHAN, P. N. (2005), “Self-adaptive Differential Evolution Algorithm for Numerical Optimization”, *Special Session on Real-Parameter Optimization at CEC-05*, Edinburgh, UK.
- RÖNKKÖNEN, J.; KUKKONEN, S. & PRICE, K. V. (2005), “Real-Parameter Optimization with Differential Evolution”, *Special Session on Real-Parameter Optimization at CEC-05*, Edinburgh, UK.
- SANNER, M. F.; OLSON, A. J. & SPEHNER, J. C. (1996), “Reduces surface: an efficient way to compute molecular surfaces”, *Biopolymers*, Vol. 38, p. 305-320.
- SINHA, A.; TIWARI, S. & DEB., K. (2005), “A Population-Based, Steady-State Procedure for Real-Parameter Optimization”, *Special Session on Real-Parameter Optimization at CEC-05*, Edinburgh, UK.
- SMITH, G. R. & STERNBERG, M. J. E. (2002), “Prediction of protein-protein interactions by docking methods”, *Current opinion on structural biology*, Vol. 12, p. 28-35.
- SOUSA, S. F.; FERNANDES, P. A. & RAMOS, M. J. (2006), “Protein-ligand docking: current status and future challenges”, *PROTEINS: Structure, Function and Bioinformatics*, Vol. 65, p. 15-26.
- SUGANTHAN, P. N.; HANSEN, N.; LIANG, J. J.; DEB, K.; CHEN, Y. P.; AUGER, A. & TIWARI, S., (2005). “Problem definitions and evaluation criteria for the CEC 2005 special session on real parameter optimization”. *Technical Report*, Nanyang Technological University.

- THISTLETON, W. J.; MARSH, J. A.; NELSON, K. & TSALLIS, C. (2007), “Generalized Box-Müller method for generating q-Gaussian random deviates”. *IEEE Trans on Information Theory*, Vol. 53(12), p. 4805-4810.
- TINÓS, R. & YANG, S. (2007), “Self-organizing genetic algorithms for dynamic optimization problems”, *Genetic Programming and Evolvable Machines*, Vol. 8(3), p. 255-286.
- TINÓS, R. & YANG, S. (2010), “Use of q-Gaussian mutation in evolutionary algorithms”, *Soft Computing*, Vol. 15, p. 1523-1549, Springer.
- TROTT, O. & OLSON, A. J. (2010) “AutoDock Vina: Improving the speed and accuracy of docking with a new scoring function, efficient optimization, and Multithreading” *Journal of Computational Chemistry*, Vol. 31, p. 455-461.
- VAVAK, F. & FOGARTY, T. C., (1996) “A comparative study of steady state and generational genetic algorithms for use in non-stationary environments” In: *Lecture Notes in Computer Science*. Vol. 1143. p. 297–304, Springer.
- VELEC, H. F.; GOHLKE, H.; KLEBE, G. (2005), “DrugScore^{CSD} – knowledge-based scoring function derived from small molecule crystal data with superior recognition rate of near-native ligand poses and better affinity prediction”. *Journal of Medicinal Chemistry*, Vol. 48, p. 6296-6303.
- WEINER, P. K. & KOLLMAN, P. A. (1981), “AMBER – Assisted Model Building with Energy Refinement – a general program for modeling molecules and their interactions”, *Journal of Computational Chemistry*, Vol. 2(3), p. 287-303.
- WILCOXON, F. (1945), “Individual comparison by ranking methods”, *Biometrics Bulletin*, Vol. 1(6), p. 80-83.
- WOLFRAM MATHWORLD, (2012), “Rotation matrix”, <http://mathworld.wolfram.com/RotationMatrix.html>, visitado em 13 de maio de 2012.
- YAO, X.; LIU, Y.; & LIN, G. (1999) “Evolutionary programming made faster”, *Evolutionary Computation*, Vol. 3(2), p. 82-102.

APÊNDICE

A.1. EXPERIMENTOS DE SELEÇÃO DE PARÂMETROS DO SOCOPT-AINET

Os parâmetros utilizados inicialmente para os testes do SOCOpt-aiNet foram:

- Número inicial de células (soluções candidatas): $N=20$;
- Número de clones gerados por cada célula: $N_c=10$;
- Pressão da maturação de afinidade: $\beta=1$.
- Limiar de convergência: $l_c=0,001$;
- Limiar de supressão: $\sigma_s=0,06$;
- Porcentagem de soluções aleatórias adicionadas: $r=(ext_size*10)\%$;
- Intervalo de supressão: $g_s=75$.

Figura A.1: Evolução do erro médio da função f_{25} de uma execução do SOCOpt-aiNet e opt-aiNet.

Pode se argumentar que o fraco desempenho do algoritmo na otimização de algumas das funções, ocorre devido à supressão precoce de algumas soluções, isto é, substituições ocorrem antes que a população tenha atingido um ótimo local, como pode ser visto na Figura A.1, que mostra a evolução do erro médio em uma das execuções da função f_{25} . No gráfico, gerações em que ocorrem substituições podem ser identificadas

por pontos onde ocorre um aumento repentino do erro, já que soluções geradas aleatoriamente tendem a apresentar *fitness* pior.

Nota-se na Figura A.1 que o opt-aiNet atinge um ótimo local antes que ocorra a supressão. Isto é garantido, dado que esta etapa ocorre somente quando verificada a estabilização do *fitness* médio populacional. No caso do SOCOpt-aiNet, nota-se que não ocorre a convergência devido ao curto intervalo de supressão, levando a grandes oscilações no *fitness*.

Experimentos foram realizados utilizando diferentes parâmetros do SOCOpt-aiNet, buscando possibilitar a exploração de ótimos locais. Para a escolha de novos parâmetros foram selecionadas as funções multimodais básicas *f7*, *f9*, *f10* e *f12* para testar o algoritmo com diferentes valores de intervalo de supressão (g_s), limiar de convergência (l_c) e limiar de supressão (σ_s). Testes foram realizados somente para $N_D=30$ e 50, e foram selecionados os parâmetros que apresentaram melhores resultados (*i.e.*, menor erro médio) em maior parte dos casos. Estes foram destacados nas tabelas a seguir, que descrevem os resultados.

A Tabela A.1 e Tabela A.2 apresentam os testes com diferentes parâmetros de intervalo de supressão. Neste caso o intervalo de supressão selecionado foi de 75 gerações que, nos testes, apresentou melhor desempenho em quatro dos oito casos de teste (nas funções *f9* e *f12* nos testes com $N_D=30$ e nas funções *f10* e *f12* com $N_D=50$).

Tabela A.1: Resultados da seleção de parâmetros do intervalo de supressão (g_s) com $N_D=30$.

		25	35	45	55	65	75
f7	Média	1,160E+03	1,190E+03	1,167E+03	1,201E+03	1,163E+03	1,161E+03
	Desvio padrão	3,733E+02	3,868E+02	3,449E+02	3,895E+02	3,613E+02	3,607E+02
	Mediana	1,156E+03	1,150E+03	1,155E+03	1,182E+03	1,163E+03	1,162E+03
	Mínimo	7,884E+02	6,985E+02	7,966E+02	7,522E+02	7,927E+02	7,226E+02
f9	Média	2,197E+02	2,167E+02	2,154E+02	2,200E+02	2,184E+02	2,140E+02
	Desvio padrão	3,420E+02	3,461E+02	3,446E+02	3,424E+02	3,418E+02	3,435E+02
	Mediana	2,195E+02	2,192E+02	2,178E+02	2,228E+02	2,199E+02	2,148E+02
	Mínimo	1,995E+02	1,705E+02	1,812E+02	1,873E+02	1,961E+02	1,959E+02
f10	Média	2,400E+02	2,416E+02	2,371E+02	2,359E+02	2,391E+02	2,375E+02
	Desvio padrão	3,439E+02	3,405E+02	3,407E+02	3,430E+02	2,391E+02	3,422E+02
	Mediana	2,413E+02	2,401E+02	2,372E+02	2,392E+02	2,377E+02	2,405E+02
	Mínimo	2,054E+02	2,202E+02	2,214E+02	1,907E+02	2,195E+02	2,163E+02
f12	Média	4,172E+05	4,276E+05	3,981E+05	3,866E+05	4,027E+05	3,810E+05
	Desvio padrão	5,743E+04	5,178E+04	4,915E+04	3,126E+04	4,705E+04	3,376E+04
	Mediana	4,109E+05	4,321E+05	4,116E+05	3,880E+05	3,970E+05	3,726E+05

Mínimo	2,703E+05	3,438E+05	2,955E+05	3,389E+05	2,988E+05	3,253E+05
---------------	-----------	-----------	-----------	-----------	-----------	-----------

Os resultados dos experimentos para seleção do parâmetro de limiar de convergência são apresentados na Tabela A.3 e Tabela A.4. Nestes testes os parâmetros 0,001, 0,002 e 0,003 apresentaram melhor resultado em quatro casos. Porém foi escolhido o parâmetro 0,001 pois apresentou melhor resultado em mais casos nos testes com $N_D=50$, que tem maior complexidade devido ao número de dimensões.

Finalmente, foi feita uma seleção do parâmetro de limiar de supressão (Tabela A.5 e Tabela A.6), que apresentou melhores resultados com o parâmetro 0,06, totalizando quatro casos, dos quais dois são dos testes realizados com $N_D=50$.

Tabela A.2: Resultados da seleção de parâmetros do intervalo de supressão (g_s) com $N_D=50$.

		25	35	45	55	65	75
f7	Média	5,757E+02	5,789E+02	5,753E+02	5,701E+02	5,730E+02	5,948E+02
	Desvio padrão	2,894E+02	3,086E+02	2,802E+02	2,905E+02	2,862E+02	2,962E+02
	Mediana	5,769E+02	6,070E+02	5,877E+02	5,825E+02	5,550E+02	5,980E+02
	Mínimo	4,087E+02	3,690E+02	3,732E+02	3,571E+02	3,865E+02	4,090E+02
f9	Média	4,592E+02	4,661E+02	4,555E+02	4,511E+02	4,561E+02	4,518E+02
	Desvio padrão	3,557E+02	3,541E+02	3,531E+02	3,570E+02	3,473E+02	3,541E+02
	Mediana	4,680E+02	4,667E+02	4,608E+02	4,431E+02	4,543E+02	4,479E+02
	Mínimo	3,920E+02	4,152E+02	3,860E+02	3,733E+02	4,262E+02	3,883E+02
f10	Média	5,030E+02	4,951E+02	5,382E+08	4,916E+02	4,936E+02	4,898E+02
	Desvio padrão	3,481E+02	3,469E+02	2,770E+08	3,578E+02	3,512E+02	3,508E+02
	Mediana	5,005E+02	4,982E+02	5,454E+08	4,969E+02	4,939E+02	4,917E+02
	Mínimo	4,717E+02	4,663E+02	1,052E+08	3,995E+02	4,496E+02	4,531E+02
f12	Média	2,372E+06	2,315E+06	2,183E+06	2,154E+06	2,118E+06	2,068E+06
	Desvio padrão	2,156E+05	1,967E+05	1,844E+05	2,019E+05	1,487E+05	1,801E+05
	Mediana	2,390E+06	2,379E+06	2,203E+06	2,154E+06	2,064E+06	2,086E+06
	Mínimo	1,649E+06	1,852E+06	1,612E+06	1,803E+06	1,873E+06	1,664E+06

Tabela A.3: Resultados da seleção de parâmetros do limiar convergência (I_c) com $N_D=30$.

		0,001	0,002	0,003	0,004	0,005	0,006
f7	Média	1,161E+03	1,161E+03	2,902E+03	2,902E+03	2,902E+03	2,902E+03
	Desvio padrão	3,607E+02	3,607E+02	7,532E+02	7,532E+02	7,532E+02	7,532E+02
	Mediana	1,162E+03	1,162E+03	2,972E+03	2,972E+03	2,972E+03	2,972E+03
	Mínimo	7,226E+02	7,226E+02	1,639E+03	1,639E+03	1,639E+03	1,639E+03
f9	Média	2,140E+02	2,128E+02	2,128E+02	2,147E+02	2,147E+02	2,147E+02
	Desvio padrão	3,435E+02	3,435E+02	3,435E+02	3,436E+02	3,436E+02	3,436E+02
	Mediana	2,148E+02	2,114E+02	2,114E+02	2,148E+02	2,148E+02	2,148E+02
	Mínimo	1,959E+02	1,905E+02	1,905E+02	1,959E+02	1,959E+02	1,959E+02
f10	Média	2,375E+02	2,375E+02	2,375E+02	2,375E+02	2,375E+02	2,375E+02

	Desvio padrão	3,422E+02	3,422E+02	3,422E+02	3,422E+02	3,422E+02	3,422E+02
	Mediana	2,405E+02	2,405E+02	2,405E+02	2,405E+02	2,405E+02	2,405E+02
	Mínimo	2,163E+02	2,163E+02	2,163E+02	2,163E+02	2,163E+02	2,163E+02
f12	Média	3,810E+05	3,830E+05	3,762E+05	3,861E+05	3,834E+05	3,828E+05
	Desvio padrão	3,376E+04	3,434E+04	3,038E+04	4,564E+04	4,215E+04	4,716E+04
	Mediana	3,726E+05	3,736E+05	3,711E+05	3,782E+05	3,785E+05	3,711E+05
	Mínimo	3,253E+05	3,253E+05	3,253E+05	3,136E+05	3,136E+05	2,899E+05

Tabela A.4: Resultados da seleção de parâmetros do limiar de convergência (l_c) com $N_D=50$.

		0,001	0,002	0,003	0,004	0,005	0,006
f7	Média	5,948E+02	2,336E+03	4,504E+03	4,504E+03	4,504E+03	4,504E+03
	Desvio padrão	2,962E+02	1,835E+03	1,419E+03	1,419E+03	1,419E+03	1,419E+03
	Mediana	5,980E+02	1,670E+03	4,082E+03	4,082E+03	4,082E+03	4,082E+03
	Mínimo	4,090E+02	5,987E+02	3,109E+03	3,109E+03	3,109E+03	3,109E+03
f9	Média	4,518E+02	4,526E+02	4,543E+02	4,609E+02	4,587E+02	4,606E+02
	Desvio padrão	3,541E+02	3,490E+02	3,491E+02	3,519E+02	3,537E+02	3,513E+02
	Mediana	4,479E+02	4,508E+02	4,509E+02	4,568E+02	4,561E+02	4,611E+02
	Mínimo	3,883E+02	4,228E+02	4,232E+02	4,210E+02	4,210E+02	4,243E+02
f10	Média	4,898E+02	4,918E+02	4,867E+02	4,978E+02	4,937E+02	4,949E+02
	Desvio padrão	3,508E+02	3,504E+02	3,492E+02	3,501E+02	3,509E+02	3,478E+02
	Mediana	4,917E+02	4,979E+02	4,833E+02	4,997E+02	4,897E+02	4,989E+02
	Mínimo	4,531E+02	4,531E+02	4,541E+02	4,612E+02	4,612E+02	4,612E+02
f12	Média	2,068E+06	2,066E+06	2,077E+06	2,076E+06	2,140E+06	2,103E+06
	Desvio padrão	1,801E+05	1,809E+05	1,202E+05	1,691E+05	1,496E+05	1,185E+05
	Mediana	2,086E+06	2,086E+06	2,102E+06	2,062E+06	2,145E+06	2,102E+06
	Mínimo	1,664E+06	1,664E+06	1,799E+06	1,693E+06	1,877E+06	1,842E+06

Tabela A.5: Resultados da seleção de parâmetros do limiar de supressão (σ_s) com $N_D=30$.

		0,02	0,03	0,04	0,05	0,06	0,07
f7	Média	9,876E-01	9,823E-01	9,765E-01	1,161E+03	9,743E-01	9,759E-01
	Desvio padrão	1,800E+02	1,800E+02	1,800E+02	3,607E+02	1,800E+02	1,800E+02
	Mediana	9,869E-01	9,842E-01	9,791E-01	1,162E+03	9,818E-01	9,814E-01
	Mínimo	9,368E-01	9,419E-01	9,082E-01	7,226E+02	8,853E-01	8,853E-01
f9	Média	2,125E+02	2,121E+02	2,124E+02	2,140E+02	2,106E+02	2,111E+02
	Desvio padrão	3,417E+02	3,421E+02	3,429E+02	3,435E+02	3,441E+02	3,443E+02
	Mediana	2,132E+02	2,132E+02	2,132E+02	2,148E+02	2,098E+02	2,098E+02
	Mínimo	1,905E+02	1,905E+02	1,905E+02	1,959E+02	1,839E+02	1,839E+02
f10	Média	2,401E+02	2,420E+02	2,421E+02	2,375E+02	2,413E+02	2,417E+02
	Desvio padrão	3,390E+02	3,376E+02	3,377E+02	3,422E+02	3,375E+02	3,377E+02
	Mediana	2,394E+02	2,411E+02	2,423E+02	2,405E+02	2,405E+02	2,405E+02
	Mínimo	2,193E+02	2,205E+02	2,205E+02	2,163E+02	2,205E+02	2,242E+02
f12	Média	3,537E+05	3,718E+05	3,721E+05	3,810E+05	3,797E+05	3,872E+05
	Desvio padrão	3,544E+04	4,834E+04	4,553E+04	3,376E+04	3,702E+04	4,240E+04
	Mediana	3,616E+05	3,736E+05	3,748E+05	3,726E+05	3,814E+05	3,866E+05

Mínimo	2,658E+05	2,658E+05	2,812E+05	3,253E+05	3,137E+05	2,765E+05
--------	-----------	-----------	-----------	-----------	-----------	-----------

Tabela A.6: Resultados da seleção de parâmetros do limiar de supressão (σ_s) com $N_D=50$.

		0,02	0,03	0,04	0,05	0,06	0,07
f7	Média	1,049E+00	1,048E+00	1,049E+00	5,948E+02	1,047E+00	1,048E+00
	Desvio padrão	1,800E+02	1,800E+02	1,800E+02	2,962E+02	1,800E+02	1,800E+02
	Mediana	1,049E+00	1,048E+00	1,048E+00	5,980E+02	1,048E+00	1,048E+00
	Mínimo	1,038E+00	1,033E+00	1,038E+00	4,090E+02	1,021E+00	1,032E+00
f9	Média	4,504E+02	4,492E+02	4,418E+02	4,518E+02	4,400E+02	4,421E+02
	Desvio padrão	3,488E+02	3,513E+02	3,517E+02	3,541E+02	3,536E+02	3,534E+02
	Mediana	4,571E+02	4,540E+02	4,466E+02	4,479E+02	4,413E+02	4,491E+02
	Mínimo	3,962E+02	3,962E+02	3,830E+02	3,883E+02	3,884E+02	3,884E+02
f10	Média	4,936E+02	4,963E+02	4,968E+02	4,898E+02	4,986E+02	4,954E+02
	Desvio padrão	3,438E+02	3,483E+02	3,455E+02	3,508E+02	3,470E+02	3,516E+02
	Mediana	4,915E+02	4,958E+02	5,004E+02	4,917E+02	4,984E+02	4,957E+02
	Mínimo	4,719E+02	4,578E+02	4,634E+02	4,531E+02	4,477E+02	4,516E+02
f12	Média	2,084E+06	2,105E+06	2,041E+06	2,068E+06	2,118E+06	2,162E+06
	Desvio padrão	1,546E+05	1,171E+05	1,630E+05	1,801E+05	1,524E+05	1,278E+05
	Mediana	2,116E+06	2,104E+06	2,031E+06	2,086E+06	2,124E+06	2,185E+06
	Mínimo	1,730E+06	1,840E+06	1,783E+06	1,664E+06	1,891E+06	1,882E+06

Dos testes realizados, então, foram selecionados os seguintes parâmetros:

- Limiar de convergência: $l_c=0,001$;
- Limiar de supressão: $\sigma_s=0,06$;
- Porcentagem de soluções aleatórias adicionadas: $r=(ext_size*10)\%$;
- Intervalo de supressão: $g_s=75$.

Definidos novos parâmetros do SOCOpt-aiNet, foram realizados então testes mais elaborados, desta vez, envolvendo todas as funções de *benchmark*. Nas tabelas contendo os resultados é indicado na coluna “Parâmetros” por “1” os resultados com os parâmetros antigos, e “2” os resultados com os parâmetros selecionados. Resultados em que o novo conjunto de parâmetros supera o antigo são indicados com um sinal (+), (-) caso contrário e (S) caso haja significância estatística, semelhante aos resultados apresentados no início da seção.

Tabela A.7: Resultados dos erros $|f(x)-f(x^*)|$ referentes às execuções com $N_D = 2$. Comparação dos parâmetros antigos com os selecionados.

Função	Parâmetros	Média	Desvio padrão	Mediana	Mínimo	p-valor
f1	1	3,593E-04	3,067E-04	3,678E-04	1,270E-06	1,569E-03 (S+)
	2	1,395E-04	9,758E-05	1,319E-04	4,780E-06	

f2	1	7,632E-04	8,621E-04	5,424E-04	5,500E-05	6,531E-02 (+)
	2	4,696E-04	4,650E-04	2,396E-04	7,448E-05	
f3	1	1,253E+02	1,671E+02	8,953E+01	7,710E-01	9,772E-01 (+)
	2	1,417E+02	1,989E+02	6,647E+01	2,777E+00	
f4	1	1,376E-03	1,506E-03	8,854E-04	9,074E-05	7,423E-03 (S+)
	2	4,796E-04	4,061E-04	4,463E-04	9,300E-06	
f5	1	5,110E-01	4,617E-01	4,070E-01	1,982E-02	4,926E-01 (-)
	2	6,191E-01	6,188E-01	4,192E-01	8,911E-02	
f6	1	7,286E-01	3,551E+00	1,651E-02	1,183E-03	2,641E-01 (+)
	2	3,867E-02	1,240E-01	7,144E-03	8,183E-04	
f7	1	4,714E+01	1,533E+01	4,033E+01	2,903E+01	1,229E-05 (S+)
	2	3,317E-01	4,193E-01	1,786E-01	1,068E-02	
f8	1	1,077E+01	7,936E+00	1,311E+01	5,634E-01	2,109E-01 (+)
	2	1,160E+01	7,916E+00	1,242E+01	1,428E+00	
f9	1	5,118E-02	6,924E-02	2,704E-02	1,095E-03	1,186E-03 (S+)
	2	8,637E-03	1,118E-02	5,408E-03	2,214E-04	
f10	1	7,149E-02	9,185E-02	3,213E-02	1,185E-03	1,281E-02 (S+)
	2	3,168E-02	4,066E-02	1,641E-02	4,934E-04	
f11	1	1,778E-01	6,944E-02	1,592E-01	7,523E-02	2,015E-01 (-)
	2	1,593E-01	6,676E-02	1,618E-01	5,986E-03	
f12	1	1,238E-01	1,208E-01	1,161E-01	1,412E-03	8,191E-01 (+)
	2	1,382E-01	1,665E-01	8,953E-02	3,315E-03	
f13	1	1,709E-04	2,553E-04	7,927E-05	2,800E-07	4,471E-02 (S-)
	2	1,405E-04	2,782E-04	3,663E-05	2,000E-08	
f14	1	8,742E-02	1,476E-01	1,970E-02	5,115E-03	1,018E-02 (S-)
	2	1,892E-01	2,017E-01	1,564E-01	3,148E-03	
f15	1	2,539E+00	3,105E+00	1,333E+00	9,879E-02	6,148E-02 (-)
	2	2,990E+00	2,646E+00	2,335E+00	1,350E-01	
f16	1	2,820E+00	2,778E+00	1,688E+00	6,571E-02	7,317E-01 (+)
	2	3,299E+00	4,554E+00	1,416E+00	2,330E-02	
f17	1	4,842E+00	8,050E+00	2,648E+00	5,559E-02	8,329E-01 (-)
	2	4,118E+00	4,068E+00	2,681E+00	3,076E-02	
f18	1	7,526E+01	5,452E+01	6,389E+01	2,421E+00	4,405E-01 (+)
	2	7,161E+01	6,047E+01	5,160E+01	2,421E+00	
f19	1	2,006E+02	3,880E+01	2,058E+02	1,083E+02	6,542E-01 (-)
	2	1,950E+02	3,926E+01	2,082E+02	1,041E+02	
f20	1	1,011E+02	7,946E+01	8,350E+01	1,721E+00	3,002E-01 (+)
	2	1,113E+02	8,760E+01	6,860E+01	7,336E+00	
f21	1	1,049E+02	7,388E+01	1,027E+02	4,952E+00	5,663E-01 (-)
	2	1,129E+02	7,559E+01	1,040E+02	4,952E+00	
f22	1	1,829E+02	3,465E+01	2,001E+02	4,976E+01	9,757E-01 (-)
	2	1,694E+02	5,498E+01	2,004E+02	3,041E+01	
f23	1	1,315E+02	1,052E+02	1,093E+02	3,806E+00	4,263E-01 (-)
	2	1,503E+02	1,046E+02	1,459E+02	3,806E+00	
f24	1	1,905E+02	2,121E+01	2,002E+02	1,226E+02	7,151E-01 (+)
	2	1,883E+02	2,502E+01	2,001E+02	1,013E+02	

f25	1	1,576E+02	1,810E+01	1,574E+02	1,132E+02	5,481E-01 (-)
	2	1,629E+02	2,447E+01	1,578E+02	1,182E+02	

Dentre os casos de teste, o novo conjunto de parâmetros apresentou melhor resultado em 4 das funções unimodais (*f1* a *f5*), 6 das funções multimodais básicas (*f6* a *f12*) e 4 das composições de funções (*f15* a *f25*) com $N_D=2$. Das funções multimodais básicas, 3 apresentaram significância estatística e dentre as composições de funções, embora os parâmetros antigos apresentem melhor resultado na maioria dos casos, não houve relevância estatística em nenhum. Das funções expandidas (*f13* e *f14*), testes com os parâmetros antigos apresentaram melhores resultados com significância estatística em todos os casos. Os resultados deste experimento são apresentados detalhadamente na Tabela A.7.

Tabela A.8: Resultados dos erros $|f(x)-f(x^*)|$ referentes às execuções com $N_D = 10$. Comparação dos parâmetros antigos com os selecionados.

Função	Parâmetros	Média	Desvio padrão	Mediana	Mínimo	p-valor
f1	1	7,702E-01	2,509E-01	8,348E-01	3,303E-01	8,401E-01 (+)
	2	7,857E-01	1,972E-01	8,279E-01	2,448E-01	
f2	1	1,254E+00	4,063E-01	1,163E+00	5,909E-01	7,570E-01 (-)
	2	1,288E+00	3,596E-01	1,262E+00	6,674E-01	
f3	1	1,231E+05	4,922E+04	1,192E+05	5,617E+04	1,742E-01 (+)
	2	1,115E+05	6,898E+04	9,277E+04	2,669E+04	
f4	1	5,257E+03	4,677E+03	5,102E+03	4,555E+00	1,390E-05 (S+)
	2	6,658E+01	2,433E+02	5,880E+00	2,540E+00	
f5	1	7,797E+01	1,524E+01	7,603E+01	5,287E+01	6,766E-01 (-)
	2	7,875E+01	1,763E+01	7,708E+01	3,833E+01	
f6	1	8,464E+01	2,483E+01	8,164E+01	3,693E+01	9,464E-01 (-)
	2	8,495E+01	2,532E+01	8,797E+01	4,023E+01	
f7	1	8,138E+02	2,393E+02	8,472E+02	3,193E+02	1,229E-05 (S+)
	2	7,367E-01	1,539E-01	7,500E-01	4,829E-01	
f8	1	2,033E+01	7,805E-02	2,034E+01	2,012E+01	2,209E-01 (-)
	2	2,036E+01	6,622E-02	2,036E+01	2,022E+01	
f9	1	3,328E+01	4,981E+00	3,328E+01	2,418E+01	1,303E-03 (S+)
	2	2,836E+01	4,550E+00	2,953E+01	1,965E+01	
f10	1	3,842E+01	5,371E+00	3,975E+01	2,506E+01	7,982E-01 (+)
	2	3,802E+01	5,070E+00	3,838E+01	2,852E+01	
f11	1	7,251E+00	1,039E+00	7,520E+00	4,898E+00	1,500E-01 (+)
	2	6,878E+00	1,080E+00	6,972E+00	4,031E+00	
f12	1	7,705E+03	1,842E+03	7,630E+03	4,809E+03	1,941E-04 (S+)
	2	5,389E+03	2,019E+03	5,914E+03	1,234E+03	
f13	1	4,208E+00	7,280E-01	4,143E+00	2,662E+00	1,489E-02 (S+)
	2	3,707E+00	5,770E-01	3,840E+00	2,303E+00	
f14	1	3,765E+00	2,177E-01	3,812E+00	3,276E+00	1,079E-03 (S-)

	2	3,897E+00	2,291E-01	3,995E+00	3,297E+00	
f15	1	3,313E+02	3,956E+01	3,368E+02	2,524E+02	7,800E-02 (-)
	2	3,423E+02	4,407E+01	3,506E+02	2,175E+02	
f16	1	1,864E+02	1,039E+01	1,848E+02	1,643E+02	7,164E-01 (-)
	2	1,875E+02	1,212E+01	1,899E+02	1,583E+02	
f17	1	2,025E+02	1,071E+01	2,019E+02	1,792E+02	9,893E-01 (-)
	2	2,020E+02	1,673E+01	2,040E+02	1,553E+02	
f18	1	5,809E+02	3,375E+01	5,815E+02	5,036E+02	7,775E-01 (-)
	2	5,873E+02	3,508E+01	5,831E+02	5,241E+02	
f19	1	5,736E+02	3,541E+01	5,759E+02	5,030E+02	9,797E-02 (-)
	2	5,855E+02	3,297E+01	5,861E+02	5,261E+02	
f20	1	5,709E+02	2,858E+01	5,743E+02	4,995E+02	9,464E-01 (+)
	2	5,732E+02	3,697E+01	5,711E+02	4,952E+02	
f21	1	6,740E+02	4,535E+01	6,683E+02	6,003E+02	3,819E-01 (-)
	2	6,871E+02	4,670E+01	6,899E+02	6,295E+02	
f22	1	7,990E+02	1,758E+01	8,049E+02	7,248E+02	2,641E-01 (-)
	2	8,043E+02	8,175E+00	8,059E+02	7,869E+02	
f23	1	6,806E+02	3,884E+01	6,870E+02	6,120E+02	8,191E-01 (-)
	2	6,864E+02	6,096E+01	6,913E+02	5,616E+02	
f24	1	3,847E+02	4,052E+01	3,863E+02	3,248E+02	7,775E-01 (+)
	2	3,822E+02	4,446E+01	3,743E+02	2,884E+02	
f25	1	3,975E+02	3,784E+01	4,119E+02	2,761E+02	3,819E-01 (-)
	2	4,031E+02	1,964E+01	4,121E+02	3,388E+02	

A Tabela A.8 contém os resultados dos experimentos com $N_D=10$. Nestes casos, testes com novos parâmetros apresentaram melhor desempenho em 5 das funções multimodais básicas das quais 3 apresentaram relevância estatística. Das composições de funções, somente em 2 casos os novos parâmetros apresentaram melhor desempenho. Porém, mais uma vez, não houve diferença estatisticamente relevante pelo teste de Wilcoxon. Dos testes com os parâmetros antigos, somente na função *f14* foi apresentado melhor desempenho com relevância estatística.

Tabela A.9: Resultados dos erros $|f(x)-f(x^*)|$ referentes às execuções com $N_D = 30$. Comparação dos parâmetros antigos com os selecionados.

Função	Parâmetros	Média	Desvio padrão	Mediana	Mínimo	p-valor
f1	1	1,358E+01	1,884E+00	1,354E+01	1,009E+01	6,186E-01 (+)
	2	1,345E+01	1,209E+00	1,333E+01	1,087E+01	
f2	1	5,984E+01	1,176E+01	6,145E+01	3,890E+01	3,031E-02 (S+)
	2	5,296E+01	1,028E+01	5,224E+01	3,248E+01	
f3	1	2,707E+06	6,795E+05	2,594E+06	1,268E+06	3,967E-01 (-)
	2	2,865E+06	6,402E+05	2,994E+06	1,644E+06	
f4	1	3,478E+04	1,033E+04	3,355E+04	1,800E+04	1,229E-05 (S+)
	2	1,629E+04	3,261E+03	1,633E+04	1,124E+04	
f5	1	4,433E+03	6,990E+02	4,444E+03	2,649E+03	6,848E-03 (S+)

	2	4,015E+03	5,745E+02	4,119E+03	2,747E+03	
f6	1	3,578E+03	7,970E+02	3,621E+03	2,251E+03	2,142E-02 (S+)
	2	3,117E+03	6,699E+02	3,216E+03	1,768E+03	
f7	1	1,160E+03	1,933E+02	1,156E+03	7,884E+02	1,229E-05 (S+)
	2	9,743E-01	3,026E-02	9,818E-01	8,853E-01	
f8	1	2,095E+01	4,038E-02	2,096E+01	2,086E+01	4,118E-01 (+)
	2	2,094E+01	5,176E-02	2,095E+01	2,078E+01	
f9	1	2,197E+02	1,201E+01	2,195E+02	1,995E+02	2,299E-02 (S+)
	2	2,106E+02	1,406E+01	2,098E+02	1,839E+02	
f10	1	2,400E+02	1,389E+01	2,413E+02	2,054E+02	4,926E-01 (+)
	2	2,413E+02	7,536E+00	2,405E+02	2,205E+02	
f11	1	3,950E+01	1,152E+00	3,974E+01	3,638E+01	4,237E-01 (+)
	2	3,916E+01	1,434E+00	3,946E+01	3,595E+01	
f12	1	4,172E+05	5,697E+04	4,109E+05	2,703E+05	1,489E-02 (S+)
	2	3,797E+05	3,656E+04	3,814E+05	3,137E+05	
f13	1	3,885E+01	3,606E+00	3,825E+01	2,911E+01	9,042E-05 (S+)
	2	3,326E+01	2,150E+00	3,353E+01	2,900E+01	
f14	1	1,330E+01	2,167E-01	1,330E+01	1,271E+01	6,531E-02 (-)
	2	1,340E+01	2,379E-01	1,351E+01	1,282E+01	
f15	1	5,142E+02	2,997E+00	5,148E+02	5,077E+02	2,418E-01 (-)
	2	5,151E+02	4,143E+00	5,148E+02	5,062E+02	
f16	1	2,734E+02	1,377E+01	2,758E+02	2,467E+02	5,272E-01 (-)
	2	2,752E+02	1,488E+01	2,790E+02	2,405E+02	
f17	1	3,028E+02	1,593E+01	3,055E+02	2,564E+02	7,164E-01 (+)
	2	3,010E+02	1,799E+01	3,019E+02	2,683E+02	
f18	1	9,353E+02	3,772E+00	9,352E+02	9,270E+02	1,742E-01 (-)
	2	9,363E+02	3,143E+00	9,366E+02	9,290E+02	
f19	1	9,365E+02	3,507E+00	9,367E+02	9,301E+02	6,148E-02 (+)
	2	9,344E+02	3,905E+00	9,348E+02	9,258E+02	
f20	1	9,330E+02	4,726E+00	9,339E+02	9,240E+02	3,955E-02 (S-)
	2	9,361E+02	3,361E+00	9,361E+02	9,294E+02	
f21	1	8,683E+02	2,849E+01	8,751E+02	7,873E+02	8,705E-03 (S+)
	2	8,488E+02	2,780E+01	8,498E+02	7,891E+02	
f22	1	9,777E+02	1,173E+01	9,780E+02	9,542E+02	3,130E-01 (-)
	2	9,806E+02	1,129E+01	9,787E+02	9,541E+02	
f23	1	8,918E+02	2,439E+01	8,966E+02	8,386E+02	2,699E-03 (S+)
	2	8,670E+02	2,265E+01	8,653E+02	8,190E+02	
f24	1	7,638E+02	2,889E+01	7,684E+02	7,026E+02	3,955E-02 (S+)
	2	7,448E+02	2,781E+01	7,417E+02	6,848E+02	
f25	1	2,170E+02	1,521E+00	2,171E+02	2,146E+02	1,919E-01 (+)
	2	2,164E+02	1,526E+00	2,164E+02	2,140E+02	

Nos testes com $N_D=30$ (Tabela A.9) ocorreram resultados com significância estatística dentre as composições de funções. Os testes com os novos parâmetros apresentaram melhor desempenho em 6 dos casos, dos quais 3 possuem relevância estatística ($f24$, $f23$ e $f21$). O conjunto de parâmetros novo também apresentou melhor

desempenho em todas as funções multimodais básicas, sendo 4 com significância estatística.

Tabela A.10: Resultados dos erros $|f(x)-f(x^*)|$ referentes às execuções com $N_D = 50$. Comparação dos parâmetros antigos com os selecionados.

Função	Parâmetros	Média	Desvio padrão	Mediana	Mínimo	p-valor
f1	1	4,138E+01	3,638E+00	4,186E+01	3,204E+01	1,405E-04 (S+)
	2	3,692E+01	4,074E+00	3,801E+01	2,662E+01	
f2	1	5,415E+02	1,261E+02	5,142E+02	3,684E+02	1,721E-03 (S+)
	2	4,433E+02	4,373E+01	4,375E+02	3,451E+02	
f3	1	7,914E+06	1,991E+06	8,060E+06	4,240E+06	7,570E-01 (+)
	2	7,795E+06	1,461E+06	7,844E+06	4,952E+06	
f4	1	8,274E+04	1,598E+04	8,250E+04	5,836E+04	1,229E-05 (S+)
	2	5,093E+04	8,798E+03	5,368E+04	2,854E+04	
f5	1	1,046E+04	1,600E+03	1,047E+04	7,682E+03	4,530E-03 (S+)
	2	9,402E+03	1,168E+03	9,346E+03	6,338E+03	
f6	1	1,666E+04	3,752E+03	1,772E+04	8,319E+03	8,753E-02 (+)
	2	1,504E+04	2,933E+03	1,398E+04	8,252E+03	
f7	1	5,757E+02	1,094E+02	5,769E+02	4,087E+02	1,229E-05 (S+)
	2	1,047E+00	9,654E-03	1,048E+00	1,021E+00	
f8	1	2,113E+01	3,402E-02	2,113E+01	2,104E+01	5,159E-02 (-)
	2	2,114E+01	3,691E-02	2,114E+01	2,104E+01	
f9	1	4,592E+02	2,573E+01	4,680E+02	3,920E+02	1,281E-02 (S+)
	2	4,400E+02	2,363E+01	4,413E+02	3,884E+02	
f10	1	5,030E+02	1,809E+01	5,005E+02	4,717E+02	3,260E-01 (+)
	2	4,986E+02	1,704E+01	4,984E+02	4,477E+02	
f11	1	7,335E+01	8,799E-01	7,334E+01	7,183E+01	3,674E-01 (-)
	2	7,294E+01	1,603E+00	7,336E+01	6,849E+01	
f12	1	2,372E+06	2,152E+05	2,390E+06	1,649E+06	7,224E-05 (S+)
	2	2,118E+06	1,519E+05	2,124E+06	1,891E+06	
f13	1	1,404E+02	1,542E+01	1,403E+02	1,007E+02	1,229E-05 (S+)
	2	9,289E+01	1,156E+01	9,366E+01	7,538E+01	
f14	1	2,304E+01	2,483E-01	2,312E+01	2,225E+01	1,725E-02 (S-)
	2	2,319E+01	1,869E-01	2,321E+01	2,259E+01	
f15	1	5,126E+02	4,898E+00	5,147E+02	4,999E+02	4,593E-01 (+)
	2	5,143E+02	4,228E+00	5,144E+02	5,061E+02	
f16	1	3,553E+02	1,632E+01	3,590E+02	3,236E+02	2,312E-01 (+)
	2	3,495E+02	1,459E+01	3,526E+02	3,277E+02	
f17	1	3,854E+02	1,515E+01	3,872E+02	3,457E+02	1,036E-01 (-)
	2	3,909E+02	1,715E+01	3,935E+02	3,475E+02	
f18	1	9,848E+02	8,519E+00	9,851E+02	9,589E+02	2,312E-01 (-)
	2	9,880E+02	6,376E+00	9,882E+02	9,709E+02	
f19	1	9,864E+02	5,922E+00	9,868E+02	9,740E+02	5,272E-01 (-)
	2	9,876E+02	7,947E+00	9,882E+02	9,721E+02	
f20	1	9,879E+02	8,402E+00	9,886E+02	9,678E+02	6,766E-01 (+)

	2	9,878E+02	7,119E+00	9,877E+02	9,700E+02	
f21	1	1,046E+03	1,881E+01	1,047E+03	1,011E+03	
	2	1,040E+03	3,042E+01	1,046E+03	9,434E+02	7,775E-01 (+)
f22	1	1,046E+03	1,744E+01	1,046E+03	1,010E+03	
	2	1,044E+03	1,668E+01	1,047E+03	1,011E+03	7,982E-01 (-)
f23	1	1,041E+03	2,335E+01	1,038E+03	9,986E+02	
	2	1,040E+03	2,820E+01	1,035E+03	9,816E+02	8,401E-01 (+)
f24	1	1,016E+03	2,587E+01	1,021E+03	9,313E+02	
	2	1,009E+03	3,298E+01	1,013E+03	9,446E+02	1,500E-01 (+)
f25	1	2,506E+02	6,173E+00	2,512E+02	2,383E+02	
	2	2,522E+02	7,198E+00	2,501E+02	2,409E+02	2,879E-01 (+)

Com $N_D=50$ (Tabela A.10), o novo conjunto de parâmetros apresentou melhor desempenho em todas as funções unimodais das quais 4 apresentaram relevância estatística, 5 das multimodais básicas, onde 3 apresentaram diferença estatisticamente significativa, e 7 das funções compostas. Nestas, porém, mais uma vez, não houve relevância estatística em nenhum dos casos.

Nas funções, com $N_D=30$ e 50, o novo conjunto de parâmetros apresentou melhor desempenho na maior parte dos casos de testes. Considerando o interesse na utilização deste algoritmo para a resolução de problemas multimodais complexos, percebe-se que a seleção de parâmetros desempenhou uma melhora importante, visto que testes com o novo conjunto de parâmetros apresentou melhores resultados na grande maioria das composições de funções e nas funções multimodais básicas.

Para verificar como a mudança dos parâmetros afetou a evolução do erro médio, foi gerado um novo gráfico (Figura A.2) da mesma execução com os parâmetros antigos, apresentada na Figura A.1, comparando-a a uma execução com o novo conjunto de parâmetros.

Figura A.2: Evolução do erro médio da função f_{25} em uma das execuções. Comparativo dos parâmetros antigos (1), com os parâmetros novos (2).

Na Figura A.2, utilizando o novo conjunto de parâmetros, nota-se que o erro médio ficou mais estável em comparação com o resultado com os parâmetros antigos. Porém, ainda assim, entre uma supressão e outra, pode-se verificar que o *fitness* médio ainda está melhorando, o que indica que, mesmo com um intervalo maior, a supressão pode estar substituindo soluções em regiões que ainda não foram completamente exploradas, que podem ser potencialmente promissoras.

O algoritmo de supressão proposto no SOCopt-aiNet, buscando manter a variabilidade da população, sempre elimina no mínimo uma solução a cada intervalo de supressão. Desta forma, não possibilita a manutenção da população inalterada uma vez que a etapa de supressão sempre ocorre. O opt-aiNet controla este problema pela verificação da convergência antes da ocorrência da supressão. Para amenizar este problema foi proposto o SOCAIN.

A.2. TABELAS DOS RESULTADOS DOS EXPERIMENTOS DE *BENCHMARK*

Tabela A.11: Resultados dos erros $|f(x) - f(x^*)|$ referentes às execuções com $N_D = 2$. Comparação do opt-aiNet com SOCopt-aiNet.

Função	Algoritmo	Média	Desvio padrão	Mediana	Mínimo	p-value
f1	<i>opt-aiNet</i>	5,558E-04	1,671E-03	1,561E-04	2,019E-05	9,797E-02 (+)
	<i>SOCopt-aiNet</i>	1,395E-04	9,758E-05	1,319E-04	4,780E-06	
f2	<i>opt-aiNet</i>	3,452E-04	3,588E-04	1,434E-04	2,543E-05	3,395E-01 (-)
	<i>SOCopt-aiNet</i>	4,696E-04	4,650E-04	2,396E-04	7,448E-05	

f3	<i>opt-aiNet</i>	2,395E+02	3,374E+02	1,596E+02	8,726E+00	1,281E-02 (S+)
	<i>SOCopt-aiNet</i>	1,417E+02	1,989E+02	6,647E+01	2,777E+00	
f4	<i>opt-aiNet</i>	1,880E+02	5,039E+02	7,757E-03	4,115E-05	2,665E-04 (S+)
	<i>SOCopt-aiNet</i>	4,796E-04	4,061E-04	4,463E-04	9,300E-06	
f5	<i>opt-aiNet</i>	2,319E+02	4,923E+02	1,381E+00	9,790E-02	4,927E-03 (S+)
	<i>SOCopt-aiNet</i>	6,191E-01	6,188E-01	4,192E-01	8,911E-02	
f6	<i>opt-aiNet</i>	8,799E+00	1,900E+01	1,058E-02	1,681E-05	2,299E-02 (S+)
	<i>SOCopt-aiNet</i>	3,867E-02	1,240E-01	7,144E-03	8,183E-04	
f7	<i>opt-aiNet</i>	1,074E+02	2,919E+01	1,050E+02	6,592E+01	1,229E-05 (S+)
	<i>SOCopt-aiNet</i>	3,317E-01	4,193E-01	1,786E-01	1,068E-02	
f8	<i>opt-aiNet</i>	1,446E+01	6,040E+00	1,679E+01	1,377E+00	1,919E-01 (+)
	<i>SOCopt-aiNet</i>	1,160E+01	7,916E+00	1,242E+01	1,428E+00	
f9	<i>opt-aiNet</i>	8,237E-03	9,818E-03	5,622E-03	2,376E-04	7,366E-01 (+)
	<i>SOCopt-aiNet</i>	8,637E-03	1,118E-02	5,408E-03	2,214E-04	
f10	<i>opt-aiNet</i>	1,461E-02	1,459E-02	1,012E-02	7,025E-04	8,041E-03 (S-)
	<i>SOCopt-aiNet</i>	3,168E-02	4,066E-02	1,641E-02	4,934E-04	
f11	<i>opt-aiNet</i>	1,502E-01	4,393E-02	1,477E-01	8,075E-02	7,164E-01 (-)
	<i>SOCopt-aiNet</i>	1,593E-01	6,676E-02	1,618E-01	5,986E-03	
f12	<i>opt-aiNet</i>	7,164E-02	6,741E-02	4,573E-02	1,330E-02	1,036E-01 (-)
	<i>SOCopt-aiNet</i>	1,382E-01	1,665E-01	8,953E-02	3,315E-03	
f13	<i>opt-aiNet</i>	6,161E-05	7,425E-05	4,104E-05	2,000E-08	7,775E-01 (+)
	<i>SOCopt-aiNet</i>	1,405E-04	2,782E-04	3,663E-05	2,000E-08	
f14	<i>opt-aiNet</i>	2,982E-01	2,087E-01	2,608E-01	1,943E-02	1,603E-02 (S+)
	<i>SOCopt-aiNet</i>	1,892E-01	2,017E-01	1,564E-01	3,148E-03	
f15	<i>opt-aiNet</i>	1,038E+00	9,489E-01	7,617E-01	4,004E-02	3,507E-03 (S-)
	<i>SOCopt-aiNet</i>	2,990E+00	2,646E+00	2,335E+00	1,350E-01	
f16	<i>opt-aiNet</i>	2,111E+00	7,874E+00	3,272E-01	4,497E-02	1,382E-02 (S-)
	<i>SOCopt-aiNet</i>	3,299E+00	4,554E+00	1,416E+00	2,330E-02	
f17	<i>opt-aiNet</i>	1,297E+00	1,478E+00	6,374E-01	6,473E-02	1,569E-03 (S-)
	<i>SOCopt-aiNet</i>	4,118E+00	4,068E+00	2,681E+00	3,076E-02	
f18	<i>opt-aiNet</i>	3,746E+01	4,933E+01	1,869E+01	4,867E+00	8,041E-03 (S-)
	<i>SOCopt-aiNet</i>	7,161E+01	6,047E+01	5,160E+01	2,421E+00	
f19	<i>opt-aiNet</i>	1,632E+02	3,428E+01	1,653E+02	1,050E+02	3,507E-03 (S-)
	<i>SOCopt-aiNet</i>	1,950E+02	3,926E+01	2,082E+02	1,041E+02	
f20	<i>opt-aiNet</i>	4,784E+01	5,733E+01	3,404E+01	1,068E+00	7,224E-05 (S-)
	<i>SOCopt-aiNet</i>	1,113E+02	8,760E+01	6,860E+01	7,336E+00	
f21	<i>opt-aiNet</i>	7,844E+01	8,131E+01	2,956E+01	3,058E+00	1,094E-01 (S-)
	<i>SOCopt-aiNet</i>	1,129E+02	7,559E+01	1,040E+02	4,952E+00	
f22	<i>opt-aiNet</i>	1,762E+02	4,768E+01	2,002E+02	1,457E+01	6,766E-01 (-)
	<i>SOCopt-aiNet</i>	1,694E+02	5,498E+01	2,004E+02	3,041E+01	
f23	<i>opt-aiNet</i>	9,881E+01	1,036E+02	4,871E+01	1,187E-01	4,951E-02 (S-)
	<i>SOCopt-aiNet</i>	1,503E+02	1,046E+02	1,459E+02	3,806E+00	
f24	<i>opt-aiNet</i>	1,794E+02	3,020E+01	2,000E+02	1,002E+02	5,784E-02 (-)
	<i>SOCopt-aiNet</i>	1,883E+02	2,502E+01	2,001E+02	1,013E+02	
f25	<i>opt-aiNet</i>	1,554E+02	2,277E+01	1,498E+02	1,202E+02	2,758E-01 (-)
	<i>SOCopt-aiNet</i>	1,629E+02	2,447E+01	1,578E+02	1,182E+02	

Tabela A.12: Resultados dos erros $|f(x)-f(x^*)|$ referentes às execuções com $N_D = 10$. Comparação do opt-aiNet com SOCOpt-aiNet.

Função	Algoritmo	Média	Desvio padrão	Mediana	Mínimo	p-value
f1	opt-aiNet	1,035E-01	2,889E-02	1,004E-01	4,990E-02	1,229E-05 (S-)
	SOCopt-aiNet	7,857E-01	1,972E-01	8,279E-01	2,448E-01	
f2	opt-aiNet	2,583E-01	8,776E-02	2,456E-01	1,200E-01	1,229E-05 (S-)
	SOCopt-aiNet	1,288E+00	3,596E-01	1,262E+00	6,674E-01	
f3	opt-aiNet	1,154E+05	5,837E+04	1,155E+05	1,369E+04	5,272E-01 (+)
	SOCopt-aiNet	1,115E+05	6,898E+04	9,277E+04	2,669E+04	
f4	opt-aiNet	1,889E+04	9,042E+03	1,695E+04	5,135E+03	1,229E-05 (S+)
	SOCopt-aiNet	6,658E+01	2,433E+02	5,880E+00	2,540E+00	
f5	opt-aiNet	1,469E+03	7,317E+02	1,341E+03	6,678E+01	1,390E-05 (S+)
	SOCopt-aiNet	7,875E+01	1,763E+01	7,708E+01	3,833E+01	
f6	opt-aiNet	1,945E+01	3,035E+00	1,985E+01	1,224E+01	1,229E-05 (S-)
	SOCopt-aiNet	8,495E+01	2,532E+01	8,797E+01	4,023E+01	
f7	opt-aiNet	2,309E+03	4,150E+02	2,455E+03	1,355E+03	1,229E-05 (S+)
	SOCopt-aiNet	7,367E-01	1,539E-01	7,500E-01	4,829E-01	
f8	opt-aiNet	2,034E+01	7,943E-02	2,035E+01	2,018E+01	2,312E-01 (-)
	SOCopt-aiNet	2,036E+01	6,622E-02	2,036E+01	2,022E+01	
f9	opt-aiNet	3,408E+01	4,010E+00	3,343E+01	2,510E+01	1,569E-03 (S+)
	SOCopt-aiNet	2,836E+01	4,550E+00	2,953E+01	1,965E+01	
f10	opt-aiNet	3,308E+01	4,447E+00	3,436E+01	2,341E+01	5,355E-03 (S-)
	SOCopt-aiNet	3,802E+01	5,070E+00	3,838E+01	2,852E+01	
f11	opt-aiNet	8,309E+00	5,481E-01	8,362E+00	6,723E+00	2,257E-05 (S+)
	SOCopt-aiNet	6,878E+00	1,080E+00	6,972E+00	4,031E+00	
f12	opt-aiNet	2,098E+03	8,490E+02	1,880E+03	8,914E+02	2,001E-05 (S-)
	SOCopt-aiNet	5,389E+03	2,019E+03	5,914E+03	1,234E+03	
f13	opt-aiNet	3,183E+00	5,004E-01	3,361E+00	1,865E+00	1,721E-03 (S-)
	SOCopt-aiNet	3,707E+00	5,770E-01	3,840E+00	2,303E+00	
f14	opt-aiNet	3,914E+00	2,110E-01	3,917E+00	3,336E+00	7,982E-01 (-)
	SOCopt-aiNet	3,897E+00	2,291E-01	3,995E+00	3,297E+00	
f15	opt-aiNet	3,400E+02	2,734E+01	3,447E+02	2,575E+02	6,766E-01 (-)
	SOCopt-aiNet	3,423E+02	4,407E+01	3,506E+02	2,175E+02	
f16	opt-aiNet	1,832E+02	1,312E+01	1,859E+02	1,523E+02	3,395E-01 (-)
	SOCopt-aiNet	1,875E+02	1,212E+01	1,899E+02	1,583E+02	
f17	opt-aiNet	1,995E+02	1,593E+01	1,999E+02	1,666E+02	6,571E-01 (-)
	SOCopt-aiNet	2,020E+02	1,673E+01	2,040E+02	1,553E+02	
f18	opt-aiNet	4,733E+02	3,985E+01	4,674E+02	4,170E+02	1,390E-05 (S-)
	SOCopt-aiNet	5,873E+02	3,508E+01	5,831E+02	5,241E+02	
f19	opt-aiNet	4,729E+02	3,594E+01	4,774E+02	4,142E+02	1,229E-05 (S-)
	SOCopt-aiNet	5,855E+02	3,297E+01	5,861E+02	5,261E+02	
f20	opt-aiNet	4,613E+02	2,559E+01	4,598E+02	3,910E+02	1,229E-05 (S-)
	SOCopt-aiNet	5,732E+02	3,697E+01	5,711E+02	4,952E+02	
f21	opt-aiNet	4,767E+02	7,794E+01	5,257E+02	3,276E+02	1,229E-05 (S-)
	SOCopt-aiNet	6,871E+02	4,670E+01	6,899E+02	6,295E+02	
f22	opt-aiNet	6,659E+02	1,504E+02	7,793E+02	4,061E+02	1,229E-05 (S-)

	<i>SO Copt-aiNet</i>	8,043E+02	8,175E+00	8,059E+02	7,869E+02	
f23	<i>opt-aiNet</i>	5,333E+02	5,721E+01	5,630E+02	4,252E+02	1,571E-05 (S-)
	<i>SO Copt-aiNet</i>	6,864E+02	6,096E+01	6,913E+02	5,616E+02	
f24	<i>opt-aiNet</i>	2,409E+02	8,977E+00	2,416E+02	2,243E+02	1,229E-05 (S-)
	<i>SO Copt-aiNet</i>	3,822E+02	4,446E+01	3,743E+02	2,884E+02	
f25	<i>opt-aiNet</i>	2,569E+02	3,440E+01	2,545E+02	2,282E+02	1,229E-05 (S-)
	<i>SO Copt-aiNet</i>	4,031E+02	1,964E+01	4,121E+02	3,388E+02	

Tabela A.13: Resultados dos erros $|f(x)-f(x^*)|$ referentes às execuções com $N_D = 30$. Comparação do *opt-aiNet* com *SO Copt-aiNet*.

Função	Algoritmo	Média	Desvio padrão	Mediana	Mínimo	p-value
f1	<i>opt-aiNet</i>	1,761E+00	1,419E-01	1,790E+00	1,409E+00	1,229E-05 (S-)
	<i>SO Copt-aiNet</i>	1,345E+01	1,209E+00	1,333E+01	1,087E+01	
f2	<i>opt-aiNet</i>	1,237E+01	3,397E+00	1,140E+01	7,307E+00	1,229E-05 (S-)
	<i>SO Copt-aiNet</i>	5,296E+01	1,028E+01	5,224E+01	3,248E+01	
f3	<i>opt-aiNet</i>	1,696E+06	3,837E+05	1,793E+06	9,158E+05	1,773E-05 (S-)
	<i>SO Copt-aiNet</i>	2,865E+06	6,402E+05	2,994E+06	1,644E+06	
f4	<i>opt-aiNet</i>	8,952E+04	1,892E+04	9,082E+04	5,420E+04	1,229E-05 (S+)
	<i>SO Copt-aiNet</i>	1,629E+04	3,261E+03	1,633E+04	1,124E+04	
f5	<i>opt-aiNet</i>	5,873E+03	5,126E+02	5,869E+03	4,889E+03	1,229E-05 (S+)
	<i>SO Copt-aiNet</i>	4,015E+03	5,745E+02	4,119E+03	2,747E+03	
f6	<i>opt-aiNet</i>	2,486E+02	3,411E+01	2,476E+02	1,812E+02	1,229E-05 (S-)
	<i>SO Copt-aiNet</i>	3,117E+03	6,699E+02	3,216E+03	1,768E+03	
f7	<i>opt-aiNet</i>	4,845E+03	2,770E+02	4,822E+03	4,342E+03	1,229E-05 (S+)
	<i>SO Copt-aiNet</i>	9,743E-01	3,026E-02	9,818E-01	8,853E-01	
f8	<i>opt-aiNet</i>	2,095E+01	5,462E-02	2,097E+01	2,082E+01	3,819E-01 (+)
	<i>SO Copt-aiNet</i>	2,094E+01	5,176E-02	2,095E+01	2,078E+01	
f9	<i>opt-aiNet</i>	2,396E+02	1,066E+01	2,393E+02	2,153E+02	1,229E-05 (S+)
	<i>SO Copt-aiNet</i>	2,106E+02	1,406E+01	2,098E+02	1,839E+02	
f10	<i>opt-aiNet</i>	2,288E+02	1,504E+01	2,310E+02	1,986E+02	4,162E-03 (S-)
	<i>SO Copt-aiNet</i>	2,413E+02	7,536E+00	2,405E+02	2,205E+02	
f11	<i>opt-aiNet</i>	3,938E+01	8,741E-01	3,936E+01	3,793E+01	6,766E-01 (-)
	<i>SO Copt-aiNet</i>	3,916E+01	1,434E+00	3,946E+01	3,595E+01	
f12	<i>opt-aiNet</i>	1,471E+05	1,963E+04	1,438E+05	1,115E+05	1,229E-05 (S-)
	<i>SO Copt-aiNet</i>	3,797E+05	3,656E+04	3,814E+05	3,137E+05	
f13	<i>opt-aiNet</i>	2,302E+01	1,378E+00	2,330E+01	1,948E+01	1,229E-05 (S-)
	<i>SO Copt-aiNet</i>	3,326E+01	2,150E+00	3,353E+01	2,900E+01	
f14	<i>opt-aiNet</i>	1,346E+01	1,738E-01	1,348E+01	1,313E+01	3,395E-01 (-)
	<i>SO Copt-aiNet</i>	1,340E+01	2,379E-01	1,351E+01	1,282E+01	
f15	<i>opt-aiNet</i>	4,832E+02	2,311E+00	4,837E+02	4,742E+02	1,229E-05 (S-)
	<i>SO Copt-aiNet</i>	5,151E+02	4,143E+00	5,148E+02	5,062E+02	
f16	<i>opt-aiNet</i>	2,855E+02	1,575E+01	2,860E+02	2,339E+02	1,725E-02 (S+)
	<i>SO Copt-aiNet</i>	2,752E+02	1,488E+01	2,790E+02	2,405E+02	
f17	<i>opt-aiNet</i>	3,200E+02	2,336E+01	3,194E+02	2,764E+02	8,705E-03 (S+)
	<i>SO Copt-aiNet</i>	3,010E+02	1,799E+01	3,019E+02	2,683E+02	
f18	<i>opt-aiNet</i>	9,009E+02	3,757E+01	9,177E+02	7,614E+02	1,229E-05 (S-)

	<i>SO Copt-aiNet</i>	9,363E+02	3,143E+00	9,366E+02	9,290E+02	
f19	<i>opt-aiNet</i>	9,057E+02	2,554E+01	9,174E+02	8,430E+02	1,229E-05 (S-)
	<i>SO Copt-aiNet</i>	9,344E+02	3,905E+00	9,348E+02	9,258E+02	
f20	<i>opt-aiNet</i>	8,959E+02	5,677E+01	9,173E+02	6,549E+02	1,229E-05 (S-)
	<i>SO Copt-aiNet</i>	9,361E+02	3,361E+00	9,361E+02	9,294E+02	
f21	<i>opt-aiNet</i>	6,077E+02	1,177E+01	6,132E+02	5,844E+02	1,229E-05 (S-)
	<i>SO Copt-aiNet</i>	8,488E+02	2,780E+01	8,498E+02	7,891E+02	
f22	<i>opt-aiNet</i>	9,676E+02	1,676E+01	9,710E+02	9,296E+02	3,216E-03 (S-)
	<i>SO Copt-aiNet</i>	9,806E+02	1,129E+01	9,787E+02	9,541E+02	
f23	<i>opt-aiNet</i>	6,276E+02	1,510E+01	6,308E+02	5,889E+02	1,229E-05 (S-)
	<i>SO Copt-aiNet</i>	8,670E+02	2,265E+01	8,653E+02	8,190E+02	
f24	<i>opt-aiNet</i>	3,886E+02	1,425E+01	3,881E+02	3,639E+02	1,229E-05 (S-)
	<i>SO Copt-aiNet</i>	7,448E+02	2,781E+01	7,417E+02	6,848E+02	
f25	<i>opt-aiNet</i>	2,129E+02	3,735E-01	2,128E+02	2,121E+02	1,229E-05 (S-)
	<i>SO Copt-aiNet</i>	2,164E+02	1,526E+00	2,164E+02	2,140E+02	

Tabela A.14: Resultados dos erros $|f(x)-f(x^*)|$ referentes às execuções com $N_D = 50$. Comparação do *opt-aiNet* com *SO Copt-aiNet*.

Função	Algoritmo	Média	Desvio padrão	Mediana	Mínimo	p-value
f1	<i>opt-aiNet</i>	5,248E+00	3,826E-01	5,293E+00	4,472E+00	1,229E-05 (S-)
	<i>SO Copt-aiNet</i>	3,692E+01	4,074E+00	3,801E+01	2,662E+01	
f2	<i>opt-aiNet</i>	1,142E+02	1,734E+01	1,130E+02	7,554E+01	1,229E-05 (S-)
	<i>SO Copt-aiNet</i>	4,433E+02	4,373E+01	4,375E+02	3,451E+02	
f3	<i>opt-aiNet</i>	3,946E+06	6,312E+05	3,821E+06	2,559E+06	1,229E-05 (S-)
	<i>SO Copt-aiNet</i>	7,795E+06	1,461E+06	7,844E+06	4,952E+06	
f4	<i>opt-aiNet</i>	1,844E+05	3,180E+04	1,769E+05	1,443E+05	1,229E-05 (S+)
	<i>SO Copt-aiNet</i>	5,093E+04	8,798E+03	5,368E+04	2,854E+04	
f5	<i>opt-aiNet</i>	1,272E+04	8,883E+02	1,272E+04	1,047E+04	1,571E-05 (S+)
	<i>SO Copt-aiNet</i>	9,402E+03	1,168E+03	9,346E+03	6,338E+03	
f6	<i>opt-aiNet</i>	8,843E+02	1,088E+02	8,640E+02	6,251E+02	1,229E-05 (S-)
	<i>SO Copt-aiNet</i>	1,504E+04	2,933E+03	1,398E+04	8,252E+03	
f7	<i>opt-aiNet</i>	4,083E+03	1,978E+02	4,116E+03	3,737E+03	1,229E-05 (S+)
	<i>SO Copt-aiNet</i>	1,047E+00	9,654E-03	1,048E+00	1,021E+00	
f8	<i>opt-aiNet</i>	2,113E+01	3,316E-02	2,113E+01	2,105E+01	2,012E-01 (-)
	<i>SO Copt-aiNet</i>	2,114E+01	3,691E-02	2,114E+01	2,104E+01	
f9	<i>opt-aiNet</i>	5,075E+02	2,418E+01	5,082E+02	4,518E+02	1,390E-05 (S+)
	<i>SO Copt-aiNet</i>	4,400E+02	2,363E+01	4,413E+02	3,884E+02	
f10	<i>opt-aiNet</i>	4,972E+02	1,910E+01	4,929E+02	4,687E+02	9,036E-01 (-)
	<i>SO Copt-aiNet</i>	4,986E+02	1,704E+01	4,984E+02	4,477E+02	
f11	<i>opt-aiNet</i>	7,266E+01	1,461E+00	7,294E+01	6,890E+01	3,260E-01 (-)
	<i>SO Copt-aiNet</i>	7,294E+01	1,603E+00	7,336E+01	6,849E+01	
f12	<i>opt-aiNet</i>	8,472E+05	6,926E+04	8,417E+05	7,101E+05	1,229E-05 (S-)
	<i>SO Copt-aiNet</i>	2,118E+06	1,519E+05	2,124E+06	1,891E+06	
f13	<i>opt-aiNet</i>	5,103E+01	2,566E+00	5,167E+01	4,560E+01	1,229E-05 (S-)
	<i>SO Copt-aiNet</i>	9,289E+01	1,156E+01	9,366E+01	7,538E+01	
f14	<i>opt-aiNet</i>	2,307E+01	2,376E-01	2,312E+01	2,239E+01	1,425E-01 (-)

	<i>SOCoapt-aiNet</i>	2,319E+01	1,869E-01	2,321E+01	2,259E+01	
f15	<i>opt-aiNet</i>	4,644E+02	1,302E+00	4,645E+02	4,626E+02	1,229E-05 (S-)
	<i>SOCoapt-aiNet</i>	5,143E+02	4,228E+00	5,144E+02	5,061E+02	
f16	<i>opt-aiNet</i>	3,635E+02	1,453E+01	3,632E+02	3,295E+02	3,507E-03 (S+)
	<i>SOCoapt-aiNet</i>	3,495E+02	1,459E+01	3,526E+02	3,277E+02	
f17	<i>opt-aiNet</i>	3,988E+02	2,039E+01	3,997E+02	3,588E+02	1,353E-01 (+)
	<i>SOCoapt-aiNet</i>	3,909E+02	1,715E+01	3,935E+02	3,475E+02	
f18	<i>opt-aiNet</i>	9,374E+02	2,369E+01	9,449E+02	8,656E+02	1,229E-05 (S-)
	<i>SOCoapt-aiNet</i>	9,880E+02	6,376E+00	9,882E+02	9,709E+02	
f19	<i>opt-aiNet</i>	9,315E+02	2,777E+01	9,443E+02	8,656E+02	1,229E-05 (S-)
	<i>SOCoapt-aiNet</i>	9,876E+02	7,947E+00	9,882E+02	9,721E+02	
f20	<i>opt-aiNet</i>	9,427E+02	1,475E+01	9,456E+02	8,728E+02	1,229E-05 (S-)
	<i>SOCoapt-aiNet</i>	9,878E+02	7,119E+00	9,877E+02	9,700E+02	
f21	<i>opt-aiNet</i>	6,833E+02	1,343E+01	6,845E+02	6,524E+02	1,229E-05 (S-)
	<i>SOCoapt-aiNet</i>	1,040E+03	3,042E+01	1,046E+03	9,434E+02	
f22	<i>opt-aiNet</i>	1,025E+03	1,186E+01	1,026E+03	9,951E+02	4,028E-04 (S-)
	<i>SOCoapt-aiNet</i>	1,044E+03	1,668E+01	1,047E+03	1,011E+03	
f23	<i>opt-aiNet</i>	7,146E+02	1,900E+01	7,130E+02	6,770E+02	1,229E-05 (S-)
	<i>SOCoapt-aiNet</i>	1,040E+03	2,820E+01	1,035E+03	9,816E+02	
f24	<i>opt-aiNet</i>	5,455E+02	2,678E+01	5,508E+02	4,696E+02	1,229E-05 (S-)
	<i>SOCoapt-aiNet</i>	1,009E+03	3,298E+01	1,013E+03	9,446E+02	
f25	<i>opt-aiNet</i>	2,276E+02	1,321E+00	2,276E+02	2,251E+02	1,229E-05 (S-)
	<i>SOCoapt-aiNet</i>	2,522E+02	7,198E+00	2,501E+02	2,409E+02	

Tabela A.15: Resultados dos erros $|f(x)-f(x^*)|$ referentes às execuções com $N_D = 2$. Comparação do SOCoapt-aiNet com o SOCAIN.

Função	Algoritmo	Média	Desvio padrão	Mediana	Mínimo	p-valor
f1	<i>SOCoapt-aiNet</i>	1,395E-04	9,758E-05	1,319E-04	4,780E-06	6,148E-02 (-)
	<i>SOCAIN</i>	3,281E-04	5,840E-04	1,523E-04	2,942E-05	
f2	<i>SOCoapt-aiNet</i>	4,696E-04	4,650E-04	2,396E-04	7,448E-05	1,477E-01 (-)
	<i>SOCAIN</i>	5,896E-04	6,790E-04	2,715E-04	6,063E-05	
f3	<i>SOCoapt-aiNet</i>	1,417E+02	1,989E+02	6,647E+01	2,777E+00	1,712E-01 (+)
	<i>SOCAIN</i>	1,501E+02	1,877E+02	9,629E+01	7,188E+00	
f4	<i>SOCoapt-aiNet</i>	4,796E-04	4,061E-04	4,463E-04	9,300E-06	1,011E-04 (S-)
	<i>SOCAIN</i>	1,357E-03	1,077E-03	9,623E-04	5,460E-05	
f5	<i>SOCoapt-aiNet</i>	6,191E-01	6,188E-01	4,192E-01	8,911E-02	5,057E-01 (-)
	<i>SOCAIN</i>	8,238E-01	1,099E+00	4,683E-01	1,216E-01	
f6	<i>SOCoapt-aiNet</i>	3,867E-02	1,240E-01	7,144E-03	8,183E-04	2,097E-01 (-)
	<i>SOCAIN</i>	1,073E-02	1,078E-02	7,170E-03	1,176E-03	
f7	<i>SOCoapt-aiNet</i>	3,317E-01	4,193E-01	1,786E-01	1,068E-02	1,229E-05 (S-)
	<i>SOCAIN</i>	4,577E+01	1,266E+01	4,360E+01	2,900E+01	
f8	<i>SOCoapt-aiNet</i>	1,160E+01	7,916E+00	1,242E+01	1,428E+00	3,804E-01 (+)
	<i>SOCAIN</i>	1,065E+01	7,654E+00	8,988E+00	7,665E-01	
f9	<i>SOCoapt-aiNet</i>	8,637E-03	1,118E-02	5,408E-03	2,214E-04	1,079E-03 (S-)
	<i>SOCAIN</i>	2,516E-02	2,830E-02	1,471E-02	5,021E-04	
f10	<i>SOCoapt-aiNet</i>	3,168E-02	4,066E-02	1,641E-02	4,934E-04	7,775E-01 (-)

	<i>SOCAIN</i>	4,092E-02	4,515E-02	2,312E-02	2,971E-04	
f11	<i>SOCopt-aiNet</i>	1,593E-01	6,676E-02	1,618E-01	5,986E-03	1,289E-01 (+)
	<i>SOCAIN</i>	1,455E-01	7,086E-02	1,272E-01	5,986E-03	
f12	<i>SOCopt-aiNet</i>	1,382E-01	1,665E-01	8,953E-02	3,315E-03	6,964E-01 (+)
	<i>SOCAIN</i>	1,311E-01	1,108E-01	8,665E-02	9,772E-03	
f13	<i>SOCopt-aiNet</i>	1,405E-04	2,782E-04	3,663E-05	2,000E-08	9,515E-01 (-)
	<i>SOCAIN</i>	1,520E-04	2,452E-04	5,896E-05	1,000E-08	
f14	<i>SOCopt-aiNet</i>	1,892E-01	2,017E-01	1,564E-01	3,148E-03	6,150E-02
	<i>SOCAIN</i>	2,036E-01	2,144E-01	1,564E-01	1,292E-02	
f15	<i>SOCopt-aiNet</i>	2,990E+00	2,646E+00	2,335E+00	1,350E-01	3,223E-01 (+)
	<i>SOCAIN</i>	2,570E+00	2,484E+00	1,765E+00	1,350E-01	
f16	<i>SOCopt-aiNet</i>	3,299E+00	4,554E+00	1,416E+00	2,330E-02	9,097E-01
	<i>SOCAIN</i>	2,815E+00	3,369E+00	1,416E+00	2,327E-02	
f17	<i>SOCopt-aiNet</i>	4,118E+00	4,068E+00	2,681E+00	3,076E-02	5,186E-01 (+)
	<i>SOCAIN</i>	3,828E+00	5,725E+00	2,037E+00	3,076E-02	
f18	<i>SOCopt-aiNet</i>	7,161E+01	6,047E+01	5,160E+01	2,421E+00	9,032E-01 (+)
	<i>SOCAIN</i>	7,350E+01	6,541E+01	5,066E+01	2,421E+00	
f19	<i>SOCopt-aiNet</i>	1,950E+02	3,926E+01	2,082E+02	1,041E+02	1,000E+00 (+)
	<i>SOCAIN</i>	1,932E+02	4,147E+01	2,075E+02	1,041E+02	
f20	<i>SOCopt-aiNet</i>	1,113E+02	8,760E+01	6,860E+01	7,336E+00	6,387E-01 (+)
	<i>SOCAIN</i>	1,070E+02	8,509E+01	6,428E+01	7,336E+00	
f21	<i>SOCopt-aiNet</i>	1,129E+02	7,559E+01	1,040E+02	4,952E+00	4,258E-01
	<i>SOCAIN</i>	1,103E+02	7,242E+01	1,040E+02	4,952E+00	
f22	<i>SOCopt-aiNet</i>	1,694E+02	5,498E+01	2,004E+02	3,041E+01	1,000E+00 (+)
	<i>SOCAIN</i>	1,772E+02	4,708E+01	2,002E+02	4,976E+01	
f23	<i>SOCopt-aiNet</i>	1,503E+02	1,046E+02	1,459E+02	3,806E+00	7,500E-01
	<i>SOCAIN</i>	1,484E+02	1,061E+02	1,459E+02	3,806E+00	
f24	<i>SOCopt-aiNet</i>	1,883E+02	2,502E+01	2,001E+02	1,013E+02	2,500E-01 (-)
	<i>SOCAIN</i>	1,926E+02	1,742E+01	2,002E+02	1,411E+02	
f25	<i>SOCopt-aiNet</i>	1,629E+02	2,447E+01	1,578E+02	1,182E+02	8,457E-01 (+)
	<i>SOCAIN</i>	1,618E+02	1,943E+01	1,609E+02	1,182E+02	

Tabela A.16: Resultados dos erros $|f(x)-f(x^*)|$ referentes às execuções com $N_D = 10$. Comparação do SOCopt-aiNet com o SOCAIN.

Função	Algoritmo	Média	Desvio padrão	Mediana	Mínimo	p-valor
f1	<i>SOCopt-aiNet</i>	7,857E-01	1,972E-01	8,279E-01	2,448E-01	6,571E-01 (+)
	<i>SOCAIN</i>	7,657E-01	2,252E-01	8,058E-01	2,430E-01	
f2	<i>SOCopt-aiNet</i>	1,288E+00	3,596E-01	1,262E+00	6,674E-01	2,209E-01 (+)
	<i>SOCAIN</i>	1,124E+00	3,516E-01	1,167E+00	3,830E-01	
f3	<i>SOCopt-aiNet</i>	1,115E+05	6,898E+04	9,277E+04	2,669E+04	3,130E-01 (+)
	<i>SOCAIN</i>	9,931E+04	5,340E+04	9,082E+04	1,691E+04	
f4	<i>SOCopt-aiNet</i>	6,658E+01	2,433E+02	5,880E+00	2,540E+00	1,229E-05 (S-)
	<i>SOCAIN</i>	5,204E+03	4,461E+03	4,705E+03	2,449E+01	
f5	<i>SOCopt-aiNet</i>	7,875E+01	1,763E+01	7,708E+01	3,833E+01	7,570E-01 (-)
	<i>SOCAIN</i>	7,785E+01	1,392E+01	8,048E+01	5,213E+01	
f6	<i>SOCopt-aiNet</i>	8,495E+01	2,532E+01	8,797E+01	4,023E+01	8,191E-01 (+)

	<i>SOCAIN</i>	8,242E+01	2,210E+01	8,171E+01	4,098E+01	
f7	<i>SOCopt-aiNet</i>	7,367E-01	1,539E-01	7,500E-01	4,829E-01	1,229E-05 (S-)
	<i>SOCAIN</i>	8,280E+02	2,346E+02	8,936E+02	2,872E+02	
f8	<i>SOCopt-aiNet</i>	2,036E+01	6,622E-02	2,036E+01	2,022E+01	9,544E-01 (-)
	<i>SOCAIN</i>	2,035E+01	6,798E-02	2,037E+01	2,018E+01	
f9	<i>SOCopt-aiNet</i>	2,836E+01	4,550E+00	2,953E+01	1,965E+01	1,079E-03 (S-)
	<i>SOCAIN</i>	3,326E+01	3,638E+00	3,257E+01	2,578E+01	
f10	<i>SOCopt-aiNet</i>	3,802E+01	5,070E+00	3,838E+01	2,852E+01	7,800E-02 (+)
	<i>SOCAIN</i>	3,581E+01	3,893E+00	3,628E+01	2,476E+01	
f11	<i>SOCopt-aiNet</i>	6,878E+00	1,080E+00	6,972E+00	4,031E+00	4,432E-01 (-)
	<i>SOCAIN</i>	7,127E+00	7,785E-01	7,102E+00	5,688E+00	
f12	<i>SOCopt-aiNet</i>	5,389E+03	2,019E+03	5,914E+03	1,234E+03	4,162E-03 (S-)
	<i>SOCAIN</i>	7,207E+03	1,953E+03	7,684E+03	1,835E+03	
f13	<i>SOCopt-aiNet</i>	3,707E+00	5,770E-01	3,840E+00	2,303E+00	6,571E-01 (+)
	<i>SOCAIN</i>	3,851E+00	5,137E-01	3,790E+00	3,017E+00	
f14	<i>SOCopt-aiNet</i>	3,897E+00	2,291E-01	3,995E+00	3,297E+00	7,570E-01
	<i>SOCAIN</i>	3,862E+00	2,394E-01	3,995E+00	3,383E+00	
f15	<i>SOCopt-aiNet</i>	3,423E+02	4,407E+01	3,506E+02	2,175E+02	6,571E-01 (+)
	<i>SOCAIN</i>	3,414E+02	2,215E+01	3,425E+02	3,062E+02	
f16	<i>SOCopt-aiNet</i>	1,875E+02	1,212E+01	1,899E+02	1,583E+02	3,011E-01 (+)
	<i>SOCAIN</i>	1,843E+02	1,522E+01	1,823E+02	1,504E+02	
f17	<i>SOCopt-aiNet</i>	2,020E+02	1,673E+01	2,040E+02	1,553E+02	4,751E-01 (+)
	<i>SOCAIN</i>	1,996E+02	1,653E+01	2,038E+02	1,632E+02	
f18	<i>SOCopt-aiNet</i>	5,873E+02	3,508E+01	5,831E+02	5,241E+02	4,926E-01 (+)
	<i>SOCAIN</i>	5,782E+02	3,262E+01	5,815E+02	5,058E+02	
f19	<i>SOCopt-aiNet</i>	5,855E+02	3,297E+01	5,861E+02	5,261E+02	2,528E-01 (+)
	<i>SOCAIN</i>	5,747E+02	3,489E+01	5,765E+02	4,734E+02	
f20	<i>SOCopt-aiNet</i>	5,732E+02	3,697E+01	5,711E+02	4,952E+02	1,531E-01 (-)
	<i>SOCAIN</i>	5,942E+02	6,399E+01	5,945E+02	4,735E+02	
f21	<i>SOCopt-aiNet</i>	6,871E+02	4,670E+01	6,899E+02	6,295E+02	3,458E-01 (+)
	<i>SOCAIN</i>	6,764E+02	6,816E+01	6,555E+02	5,991E+02	
f22	<i>SOCopt-aiNet</i>	8,043E+02	8,175E+00	8,059E+02	7,869E+02	1,742E-01 (+)
	<i>SOCAIN</i>	8,008E+02	1,034E+01	8,028E+02	7,645E+02	
f23	<i>SOCopt-aiNet</i>	6,864E+02	6,096E+01	6,913E+02	5,616E+02	3,002E-01 (-)
	<i>SOCAIN</i>	6,998E+02	5,692E+01	7,040E+02	5,984E+02	
f24	<i>SOCopt-aiNet</i>	3,822E+02	4,446E+01	3,743E+02	2,884E+02	1,793E-01 (+)
	<i>SOCAIN</i>	3,612E+02	4,429E+01	3,631E+02	2,745E+02	
f25	<i>SOCopt-aiNet</i>	4,031E+02	1,964E+01	4,121E+02	3,388E+02	3,674E-01 (+)
	<i>SOCAIN</i>	4,015E+02	2,429E+01	4,119E+02	3,173E+02	

Tabela A.17: Resultados dos erros $|f(x)-f(x^*)|$ referentes às execuções com $N_D = 30$. Comparação do SOCopt-aiNet com o SOCAIN.

Função	Algoritmo	Média	Desvio padrão	Mediana	Mínimo	p-valor
f1	<i>SOCopt-aiNet</i>	1,345E+01	1,209E+00	1,333E+01	1,087E+01	3,819E-01 (+)
	<i>SOCAIN</i>	1,311E+01	1,527E+00	1,309E+01	1,019E+01	
f2	<i>SOCopt-aiNet</i>	5,296E+01	1,028E+01	5,224E+01	3,248E+01	1,218E-01 (+)

	<i>SOCAIN</i>	4,836E+01	7,567E+00	4,798E+01	3,187E+01	
f3	<i>SOCopt-aiNet</i>	2,865E+06	6,402E+05	2,994E+06	1,644E+06	3,002E-01 (+)
	<i>SOCAIN</i>	2,712E+06	4,828E+05	2,812E+06	1,619E+06	
f4	<i>SOCopt-aiNet</i>	1,629E+04	3,261E+03	1,633E+04	1,124E+04	1,229E-05 (S-)
	<i>SOCAIN</i>	3,178E+04	7,564E+03	3,264E+04	1,895E+04	
f5	<i>SOCopt-aiNet</i>	4,015E+03	5,745E+02	4,119E+03	2,747E+03	1,281E-02 (S+)
	<i>SOCAIN</i>	3,688E+03	6,751E+02	3,726E+03	2,352E+03	
f6	<i>SOCopt-aiNet</i>	3,117E+03	6,699E+02	3,216E+03	1,768E+03	4,221E-02 (S+)
	<i>SOCAIN</i>	2,675E+03	6,284E+02	2,639E+03	1,572E+03	
f7	<i>SOCopt-aiNet</i>	9,743E-01	3,026E-02	9,818E-01	8,853E-01	1,229E-05 (S-)
	<i>SOCAIN</i>	1,184E+03	1,954E+02	1,198E+03	7,847E+02	
f8	<i>SOCopt-aiNet</i>	2,094E+01	5,176E-02	2,095E+01	2,078E+01	2,012E-01 (-)
	<i>SOCAIN</i>	2,096E+01	3,174E-02	2,096E+01	2,090E+01	
f9	<i>SOCopt-aiNet</i>	2,106E+02	1,406E+01	2,098E+02	1,839E+02	5,449E-01 (-)
	<i>SOCAIN</i>	2,130E+02	1,401E+01	2,158E+02	1,829E+02	
f10	<i>SOCopt-aiNet</i>	2,413E+02	7,536E+00	2,405E+02	2,205E+02	2,831E-02 (S+)
	<i>SOCAIN</i>	2,331E+02	1,462E+01	2,383E+02	2,015E+02	
f11	<i>SOCopt-aiNet</i>	3,916E+01	1,434E+00	3,946E+01	3,595E+01	6,892E-01 (-)
	<i>SOCAIN</i>	3,928E+01	1,165E+00	3,961E+01	3,638E+01	
f12	<i>SOCopt-aiNet</i>	3,797E+05	3,656E+04	3,814E+05	3,137E+05	4,028E-04 (S+)
	<i>SOCAIN</i>	3,342E+05	3,702E+04	3,305E+05	2,700E+05	
f13	<i>SOCopt-aiNet</i>	3,326E+01	2,150E+00	3,353E+01	2,900E+01	4,073E-05 (S+)
	<i>SOCAIN</i>	2,978E+01	1,939E+00	2,954E+01	2,446E+01	
f14	<i>SOCopt-aiNet</i>	1,340E+01	2,379E-01	1,351E+01	1,282E+01	6,571E-01 (+)
	<i>SOCAIN</i>	1,343E+01	2,340E-01	1,340E+01	1,293E+01	
f15	<i>SOCopt-aiNet</i>	5,151E+02	4,143E+00	5,148E+02	5,062E+02	2,528E-01 (+)
	<i>SOCAIN</i>	5,179E+02	2,213E+01	5,137E+02	5,025E+02	
f16	<i>SOCopt-aiNet</i>	2,752E+02	1,488E+01	2,790E+02	2,405E+02	7,366E-01 (+)
	<i>SOCAIN</i>	2,734E+02	1,376E+01	2,751E+02	2,417E+02	
f17	<i>SOCopt-aiNet</i>	3,010E+02	1,799E+01	3,019E+02	2,683E+02	4,593E-01 (+)
	<i>SOCAIN</i>	2,970E+02	1,359E+01	2,975E+02	2,757E+02	
f18	<i>SOCopt-aiNet</i>	9,363E+02	3,143E+00	9,366E+02	9,290E+02	1,018E-02 (S+)
	<i>SOCAIN</i>	9,333E+02	3,137E+00	9,340E+02	9,277E+02	
f19	<i>SOCopt-aiNet</i>	9,344E+02	3,905E+00	9,348E+02	9,258E+02	3,260E-01 (+)
	<i>SOCAIN</i>	9,339E+02	2,886E+00	9,341E+02	9,276E+02	
f20	<i>SOCopt-aiNet</i>	9,361E+02	3,361E+00	9,361E+02	9,294E+02	1,994E-02 (S+)
	<i>SOCAIN</i>	9,339E+02	3,367E+00	9,332E+02	9,272E+02	
f21	<i>SOCopt-aiNet</i>	8,488E+02	2,780E+01	8,498E+02	7,891E+02	4,797E-02 (S+)
	<i>SOCAIN</i>	8,354E+02	2,187E+01	8,325E+02	7,805E+02	
f22	<i>SOCopt-aiNet</i>	9,806E+02	1,129E+01	9,787E+02	9,541E+02	1,603E-02 (S+)
	<i>SOCAIN</i>	9,730E+02	8,367E+00	9,728E+02	9,575E+02	
f23	<i>SOCopt-aiNet</i>	8,670E+02	2,265E+01	8,653E+02	8,190E+02	1,885E-03 (S+)
	<i>SOCAIN</i>	8,414E+02	2,394E+01	8,406E+02	7,887E+02	
f24	<i>SOCopt-aiNet</i>	7,448E+02	2,781E+01	7,417E+02	6,848E+02	3,130E-01 (+)
	<i>SOCAIN</i>	7,333E+02	2,976E+01	7,373E+02	6,722E+02	
f25	<i>SOCopt-aiNet</i>	2,164E+02	1,526E+00	2,164E+02	2,140E+02	2,209E-01 (+)

<i>SOCAIN</i>	$2,160E+02$	$1,159E+00$	$2,157E+02$	$2,147E+02$
---------------	-------------	-------------	-------------	-------------

Tabela A.18: Resultados dos erros $|f(x)-f(x^*)|$ referentes às execuções com $N_D = 50$. Comparação do SOCOpt-aiNet com o SOCAIN.

Função	Algoritmo	Média	Desvio padrão	Mediana	Mínimo	p-valor
f1	<i>SOCopt-aiNet</i>	$3,692E+01$	$4,074E+00$	$3,801E+01$	$2,662E+01$	$7,800E-02$ (-)
	<i>SOCAIN</i>	$3,918E+01$	$2,813E+00$	$3,948E+01$	$3,286E+01$	
f2	<i>SOCopt-aiNet</i>	$4,433E+02$	$4,373E+01$	$4,375E+02$	$3,451E+02$	$1,603E-02$ (S+)
	<i>SOCAIN</i>	$4,067E+02$	$6,255E+01$	$4,144E+02$	$3,013E+02$	
f3	<i>SOCopt-aiNet</i>	$7,795E+06$	$1,461E+06$	$7,844E+06$	$4,952E+06$	$1,603E-02$ (S+)
	<i>SOCAIN</i>	$6,950E+06$	$8,080E+05$	$7,011E+06$	$5,456E+06$	
f4	<i>SOCopt-aiNet</i>	$5,093E+04$	$8,798E+03$	$5,368E+04$	$2,854E+04$	$1,229E-05$ (S-)
	<i>SOCAIN</i>	$6,943E+04$	$9,008E+03$	$6,679E+04$	$5,087E+04$	
f5	<i>SOCopt-aiNet</i>	$9,402E+03$	$1,168E+03$	$9,346E+03$	$6,338E+03$	$1,829E-01$ (+)
	<i>SOCAIN</i>	$9,026E+03$	$9,957E+02$	$9,276E+03$	$6,720E+03$	
f6	<i>SOCopt-aiNet</i>	$1,504E+04$	$2,933E+03$	$1,398E+04$	$8,252E+03$	$6,848E-03$ (S+)
	<i>SOCAIN</i>	$1,222E+04$	$3,308E+03$	$1,245E+04$	$7,566E+03$	
f7	<i>SOCopt-aiNet</i>	$1,047E+00$	$9,654E-03$	$1,048E+00$	$1,021E+00$	$1,229E-05$ (S-)
	<i>SOCAIN</i>	$5,736E+02$	$1,057E+02$	$5,688E+02$	$3,882E+02$	
f8	<i>SOCopt-aiNet</i>	$2,114E+01$	$3,691E-02$	$2,114E+01$	$2,104E+01$	$3,212E-02$ (S+)
	<i>SOCAIN</i>	$2,112E+01$	$3,912E-02$	$2,112E+01$	$2,100E+01$	
f9	<i>SOCopt-aiNet</i>	$4,400E+02$	$2,363E+01$	$4,413E+02$	$3,884E+02$	$4,926E-01$ (-)
	<i>SOCAIN</i>	$4,422E+02$	$2,519E+01$	$4,448E+02$	$3,504E+02$	
f10	<i>SOCopt-aiNet</i>	$4,986E+02$	$1,704E+01$	$4,984E+02$	$4,477E+02$	$3,243E-02$ (S+)
	<i>SOCAIN</i>	$4,868E+02$	$1,545E+01$	$4,846E+02$	$4,509E+02$	
f11	<i>SOCopt-aiNet</i>	$7,294E+01$	$1,603E+00$	$7,336E+01$	$6,849E+01$	$4,842E-01$ (+)
	<i>SOCAIN</i>	$7,261E+01$	$1,870E+00$	$7,303E+01$	$6,849E+01$	
f12	<i>SOCopt-aiNet</i>	$2,118E+06$	$1,519E+05$	$2,124E+06$	$1,891E+06$	$1,566E-04$ (S+)
	<i>SOCAIN</i>	$1,887E+06$	$1,488E+05$	$1,913E+06$	$1,620E+06$	
f13	<i>SOCopt-aiNet</i>	$9,289E+01$	$1,156E+01$	$9,366E+01$	$7,538E+01$	$1,390E-05$ (S+)
	<i>SOCAIN</i>	$6,943E+01$	$4,005E+00$	$6,934E+01$	$6,335E+01$	
f14	<i>SOCopt-aiNet</i>	$2,319E+01$	$1,869E-01$	$2,321E+01$	$2,259E+01$	$4,501E-02$ (S+)
	<i>SOCAIN</i>	$2,309E+01$	$2,319E-01$	$2,311E+01$	$2,235E+01$	
f15	<i>SOCopt-aiNet</i>	$5,143E+02$	$4,228E+00$	$5,144E+02$	$5,061E+02$	$2,831E-02$ (S+)
	<i>SOCAIN</i>	$5,114E+02$	$3,323E+00$	$5,116E+02$	$5,034E+02$	
f16	<i>SOCopt-aiNet</i>	$3,495E+02$	$1,459E+01$	$3,526E+02$	$3,277E+02$	$4,118E-01$ (+)
	<i>SOCAIN</i>	$3,459E+02$	$1,467E+01$	$3,428E+02$	$3,195E+02$	
f17	<i>SOCopt-aiNet</i>	$3,909E+02$	$1,715E+01$	$3,935E+02$	$3,475E+02$	$1,919E-01$ (+)
	<i>SOCAIN</i>	$3,866E+02$	$1,381E+01$	$3,861E+02$	$3,449E+02$	
f18	<i>SOCopt-aiNet</i>	$9,880E+02$	$6,376E+00$	$9,882E+02$	$9,709E+02$	$1,569E-03$ (S+)
	<i>SOCAIN</i>	$9,822E+02$	$5,249E+00$	$9,816E+02$	$9,718E+02$	
f19	<i>SOCopt-aiNet</i>	$9,876E+02$	$7,947E+00$	$9,882E+02$	$9,721E+02$	$3,031E-02$ (S+)
	<i>SOCAIN</i>	$9,826E+02$	$6,851E+00$	$9,832E+02$	$9,673E+02$	
f20	<i>SOCopt-aiNet</i>	$9,878E+02$	$7,119E+00$	$9,877E+02$	$9,700E+02$	$5,437E-02$ (+)
	<i>SOCAIN</i>	$9,849E+02$	$5,623E+00$	$9,844E+02$	$9,720E+02$	
f21	<i>SOCopt-aiNet</i>	$1,040E+03$	$3,042E+01$	$1,046E+03$	$9,434E+02$	$9,042E-05$ (S+)

	<i>SOCAIN</i>	9,895E+02	2,299E+01	9,916E+02	9,378E+02	
f22	<i>SOCopt-aiNet</i>	1,044E+03	1,668E+01	1,047E+03	1,011E+03	1,994E-02 (S+)
	<i>SOCAIN</i>	1,036E+03	1,186E+01	1,036E+03	1,015E+03	
f23	<i>SOCopt-aiNet</i>	1,040E+03	2,820E+01	1,035E+03	9,816E+02	8,085E-05 (S+)
	<i>SOCAIN</i>	9,993E+02	1,995E+01	1,002E+03	9,495E+02	
f24	<i>SOCopt-aiNet</i>	1,009E+03	3,298E+01	1,013E+03	9,446E+02	6,451E-05 (S+)
	<i>SOCAIN</i>	9,620E+02	3,047E+01	9,707E+02	8,773E+02	
f25	<i>SOCopt-aiNet</i>	2,522E+02	7,198E+00	2,501E+02	2,409E+02	1,721E-03 (S+)
	<i>SOCAIN</i>	2,472E+02	2,796E+00	2,469E+02	2,421E+02	

Tabela A.19: Resultados dos erros $|f(x)-f(x^*)|$ referentes às execuções com $N_D = 2$. Comparação do *opt-aiNet* com o *SOCAIN*.

Função	Algoritmo	Média	Desvio padrão	Mediana	Mínimo	p-valor
f1	<i>opt-aiNet</i>	5,558E-04	1,671E-03	1,561E-04	2,019E-05	6,766E-01 (+)
	<i>SOCAIN</i>	3,281E-04	5,840E-04	1,523E-04	2,942E-05	
f2	<i>opt-aiNet</i>	3,452E-04	3,588E-04	1,434E-04	2,543E-05	1,285E-01 (-)
	<i>SOCAIN</i>	5,896E-04	6,790E-04	2,715E-04	6,063E-05	
f3	<i>opt-aiNet</i>	2,395E+02	3,374E+02	1,596E+02	8,726E+00	9,263E-02 (+)
	<i>SOCAIN</i>	1,501E+02	1,877E+02	9,629E+01	7,188E+00	
f4	<i>opt-aiNet</i>	1,880E+02	5,039E+02	7,757E-03	4,115E-05	2,299E-02 (S+)
	<i>SOCAIN</i>	1,357E-03	1,077E-03	9,623E-04	5,460E-05	
f5	<i>opt-aiNet</i>	2,319E+02	4,923E+02	1,381E+00	9,790E-02	2,142E-02 (S+)
	<i>SOCAIN</i>	8,238E-01	1,099E+00	4,683E-01	1,216E-01	
f6	<i>opt-aiNet</i>	8,799E+00	1,900E+01	1,058E-02	1,681E-05	1,725E-02 (S+)
	<i>SOCAIN</i>	1,073E-02	1,078E-02	7,170E-03	1,176E-03	
f7	<i>opt-aiNet</i>	1,074E+02	2,919E+01	1,050E+02	6,592E+01	1,229E-05 (S+)
	<i>SOCAIN</i>	4,577E+01	1,266E+01	4,360E+01	2,900E+01	
f8	<i>opt-aiNet</i>	1,446E+01	6,040E+00	1,679E+01	1,377E+00	5,437E-02 (S+)
	<i>SOCAIN</i>	1,065E+01	7,654E+00	8,988E+00	7,665E-01	
f9	<i>opt-aiNet</i>	8,237E-03	9,818E-03	5,622E-03	2,376E-04	2,470E-03 (S-)
	<i>SOCAIN</i>	2,516E-02	2,830E-02	1,471E-02	5,021E-04	
f10	<i>opt-aiNet</i>	1,461E-02	1,459E-02	1,012E-02	7,025E-04	1,994E-02 (S-)
	<i>SOCAIN</i>	4,092E-02	4,515E-02	2,312E-02	2,971E-04	
f11	<i>opt-aiNet</i>	1,502E-01	4,393E-02	1,477E-01	8,075E-02	6,186E-01 (+)
	<i>SOCAIN</i>	1,455E-01	7,086E-02	1,272E-01	5,986E-03	
f12	<i>opt-aiNet</i>	7,164E-02	6,741E-02	4,573E-02	1,330E-02	1,100E-02 (S-)
	<i>SOCAIN</i>	1,311E-01	1,108E-01	8,665E-02	9,772E-03	
f13	<i>opt-aiNet</i>	6,161E-05	7,425E-05	4,104E-05	2,000E-08	2,758E-01 (-)
	<i>SOCAIN</i>	1,520E-04	2,452E-04	5,896E-05	1,000E-08	
f14	<i>opt-aiNet</i>	2,982E-01	2,087E-01	2,608E-01	1,943E-02	2,643E-02 (S+)
	<i>SOCAIN</i>	2,036E-01	2,144E-01	1,564E-01	1,292E-02	
f15	<i>opt-aiNet</i>	1,038E+00	9,489E-01	7,617E-01	4,004E-02	5,816E-03 (S-)
	<i>SOCAIN</i>	2,570E+00	2,484E+00	1,765E+00	1,350E-01	
f16	<i>opt-aiNet</i>	2,111E+00	7,874E+00	3,272E-01	4,497E-02	6,848E-03 (S-)
	<i>SOCAIN</i>	2,815E+00	3,369E+00	1,416E+00	2,327E-02	
f17	<i>opt-aiNet</i>	1,297E+00	1,478E+00	6,374E-01	6,473E-02	2,831E-02 (S-)

	<i>SOCAIN</i>	3,828E+00	5,725E+00	2,037E+00	3,076E-02	
f18	<i>opt-aiNet</i>	3,746E+01	4,933E+01	1,869E+01	4,867E+00	3,822E-03 (S-)
	<i>SOCAIN</i>	7,350E+01	6,541E+01	5,066E+01	2,421E+00	
f19	<i>opt-aiNet</i>	1,632E+02	3,428E+01	1,653E+02	1,050E+02	4,927E-03 (S-)
	<i>SOCAIN</i>	1,932E+02	4,147E+01	2,075E+02	1,041E+02	
f20	<i>opt-aiNet</i>	4,784E+01	5,733E+01	3,404E+01	1,068E+00	3,636E-04 (S-)
	<i>SOCAIN</i>	1,070E+02	8,509E+01	6,428E+01	7,336E+00	
f21	<i>opt-aiNet</i>	7,844E+01	8,131E+01	2,956E+01	3,058E+00	1,218E-01 (-)
	<i>SOCAIN</i>	1,103E+02	7,242E+01	1,040E+02	4,952E+00	
f22	<i>opt-aiNet</i>	1,762E+02	4,768E+01	2,002E+02	1,457E+01	4,118E-01
	<i>SOCAIN</i>	1,772E+02	4,708E+01	2,002E+02	4,976E+01	
f23	<i>opt-aiNet</i>	9,881E+01	1,036E+02	4,871E+01	1,187E-01	4,586E-02 (S-)
	<i>SOCAIN</i>	1,484E+02	1,061E+02	1,459E+02	3,806E+00	
f24	<i>opt-aiNet</i>	1,794E+02	3,020E+01	2,000E+02	1,002E+02	4,927E-03 (S-)
	<i>SOCAIN</i>	1,926E+02	1,742E+01	2,002E+02	1,411E+02	
f25	<i>opt-aiNet</i>	1,554E+02	2,277E+01	1,498E+02	1,202E+02	1,829E-01 (-)
	<i>SOCAIN</i>	1,618E+02	1,943E+01	1,609E+02	1,182E+02	

Tabela A.20: Resultados dos erros $|f(x)-f(x^*)|$ referentes às execuções com $N_D = 10$. Comparação do *opt-aiNet* com o *SOCAIN*.

Função	Algoritmo	Média	Desvio padrão	Mediana	Mínimo	p-valor
f1	<i>opt-aiNet</i>	1,035E-01	2,889E-02	1,004E-01	4,990E-02	1,229E-05 (S-)
	<i>SOCAIN</i>	7,657E-01	2,252E-01	8,058E-01	2,430E-01	
f2	<i>opt-aiNet</i>	2,583E-01	8,776E-02	2,456E-01	1,200E-01	1,229E-05 (S-)
	<i>SOCAIN</i>	1,124E+00	3,516E-01	1,167E+00	3,830E-01	
f3	<i>opt-aiNet</i>	1,154E+05	5,837E+04	1,155E+05	1,369E+04	1,218E-01 (+)
	<i>SOCAIN</i>	9,931E+04	5,340E+04	9,082E+04	1,691E+04	
f4	<i>opt-aiNet</i>	1,889E+04	9,042E+03	1,695E+04	5,135E+03	1,229E-05 (S+)
	<i>SOCAIN</i>	5,204E+03	4,461E+03	4,705E+03	2,449E+01	
f5	<i>opt-aiNet</i>	1,469E+03	7,317E+02	1,341E+03	6,678E+01	1,390E-05 (S+)
	<i>SOCAIN</i>	7,785E+01	1,392E+01	8,048E+01	5,213E+01	
f6	<i>opt-aiNet</i>	1,945E+01	3,035E+00	1,985E+01	1,224E+01	1,229E-05 (S-)
	<i>SOCAIN</i>	8,242E+01	2,210E+01	8,171E+01	4,098E+01	
f7	<i>opt-aiNet</i>	2,309E+03	4,150E+02	2,455E+03	1,355E+03	1,229E-05 (S+)
	<i>SOCAIN</i>	8,280E+02	2,346E+02	8,936E+02	2,872E+02	
f8	<i>opt-aiNet</i>	2,034E+01	7,943E-02	2,035E+01	2,018E+01	5,812E-01 (-)
	<i>SOCAIN</i>	2,035E+01	6,798E-02	2,037E+01	2,018E+01	
f9	<i>opt-aiNet</i>	3,408E+01	4,010E+00	3,343E+01	2,510E+01	5,098E-01 (+)
	<i>SOCAIN</i>	3,326E+01	3,638E+00	3,257E+01	2,578E+01	
f10	<i>opt-aiNet</i>	3,308E+01	4,447E+00	3,436E+01	2,341E+01	2,142E-02 (S-)
	<i>SOCAIN</i>	3,581E+01	3,893E+00	3,628E+01	2,476E+01	
f11	<i>opt-aiNet</i>	8,309E+00	5,481E-01	8,362E+00	6,723E+00	1,571E-05 (S+)
	<i>SOCAIN</i>	7,127E+00	7,785E-01	7,102E+00	5,688E+00	
f12	<i>opt-aiNet</i>	2,098E+03	8,490E+02	1,880E+03	8,914E+02	1,390E-05 (S-)
	<i>SOCAIN</i>	7,207E+03	1,953E+03	7,684E+03	1,835E+03	
f13	<i>opt-aiNet</i>	3,183E+00	5,004E-01	3,361E+00	1,865E+00	9,804E-04 (S-)

	SOCAIN	3,851E+00	5,137E-01	3,790E+00	3,017E+00	
f14	opt-aiNet	3,914E+00	2,110E-01	3,917E+00	3,336E+00	3,130E-01 (-)
	SOCAIN	3,862E+00	2,394E-01	3,995E+00	3,383E+00	
f15	opt-aiNet	3,400E+02	2,734E+01	3,447E+02	2,575E+02	8,824E-01 (+)
	SOCAIN	3,414E+02	2,215E+01	3,425E+02	3,062E+02	
f16	opt-aiNet	1,832E+02	1,312E+01	1,859E+02	1,523E+02	5,629E-01 (+)
	SOCAIN	1,843E+02	1,522E+01	1,823E+02	1,504E+02	
f17	opt-aiNet	1,995E+02	1,593E+01	1,999E+02	1,666E+02	9,893E-01 (-)
	SOCAIN	1,996E+02	1,653E+01	2,038E+02	1,632E+02	
f18	opt-aiNet	4,733E+02	3,985E+01	4,674E+02	4,170E+02	1,390E-05 (S-)
	SOCAIN	5,782E+02	3,262E+01	5,815E+02	5,058E+02	
f19	opt-aiNet	4,729E+02	3,594E+01	4,774E+02	4,142E+02	1,229E-05 (S-)
	SOCAIN	5,747E+02	3,489E+01	5,765E+02	4,734E+02	
f20	opt-aiNet	4,613E+02	2,559E+01	4,598E+02	3,910E+02	1,229E-05 (S-)
	SOCAIN	5,942E+02	6,399E+01	5,945E+02	4,735E+02	
f21	opt-aiNet	4,767E+02	7,794E+01	5,257E+02	3,276E+02	1,229E-05 (S-)
	SOCAIN	6,764E+02	6,816E+01	6,555E+02	5,991E+02	
f22	opt-aiNet	6,659E+02	1,504E+02	7,793E+02	4,061E+02	4,073E-05 (S-)
	SOCAIN	8,008E+02	1,034E+01	8,028E+02	7,645E+02	
f23	opt-aiNet	5,333E+02	5,721E+01	5,630E+02	4,252E+02	1,229E-05 (S-)
	SOCAIN	6,998E+02	5,692E+01	7,040E+02	5,984E+02	
f24	opt-aiNet	2,409E+02	8,977E+00	2,416E+02	2,243E+02	1,229E-05 (S-)
	SOCAIN	3,612E+02	4,429E+01	3,631E+02	2,745E+02	
f25	opt-aiNet	2,569E+02	3,440E+01	2,545E+02	2,282E+02	1,229E-05 (S-)
	SOCAIN	4,015E+02	2,429E+01	4,119E+02	3,173E+02	

Tabela A.21: Resultados dos erros $|f(x)-f(x^*)|$ referentes às execuções com $N_D = 30$. Comparação do opt-aiNet com o SOCAIN.

Função	Algoritmo	Média	Desvio padrão	Mediana	Mínimo	p-valor
f1	opt-aiNet	1,761E+00	1,419E-01	1,790E+00	1,409E+00	1,229E-05 (S-)
	SOCAIN	1,311E+01	1,527E+00	1,309E+01	1,019E+01	
f2	opt-aiNet	1,237E+01	3,397E+00	1,140E+01	7,307E+00	1,229E-05 (S-)
	SOCAIN	4,836E+01	7,567E+00	4,798E+01	3,187E+01	
f3	opt-aiNet	1,696E+06	3,837E+05	1,793E+06	9,158E+05	1,571E-05 (S-)
	SOCAIN	2,712E+06	4,828E+05	2,812E+06	1,619E+06	
f4	opt-aiNet	8,952E+04	1,892E+04	9,082E+04	5,420E+04	1,229E-05 (S+)
	SOCAIN	3,178E+04	7,564E+03	3,264E+04	1,895E+04	
f5	opt-aiNet	5,873E+03	5,126E+02	5,869E+03	4,889E+03	1,229E-05 (S+)
	SOCAIN	3,688E+03	6,751E+02	3,726E+03	2,352E+03	
f6	opt-aiNet	2,486E+02	3,411E+01	2,476E+02	1,812E+02	1,229E-05 (S-)
	SOCAIN	2,675E+03	6,284E+02	2,639E+03	1,572E+03	
f7	opt-aiNet	4,845E+03	2,770E+02	4,822E+03	4,342E+03	1,229E-05 (S+)
	SOCAIN	1,184E+03	1,954E+02	1,198E+03	7,847E+02	
f8	opt-aiNet	2,095E+01	5,462E-02	2,097E+01	2,082E+01	9,250E-01 (+)
	SOCAIN	2,096E+01	3,174E-02	2,096E+01	2,090E+01	
f9	opt-aiNet	2,396E+02	1,066E+01	2,393E+02	2,153E+02	1,571E-05 (S+)

	<i>SOCAIN</i>	$2,130E+02$	$1,401E+01$	$2,158E+02$	$1,829E+02$	
f10	<i>opt-aiNet</i>	$2,288E+02$	$1,504E+01$	$2,310E+02$	$1,986E+02$	$3,130E-01$ (-)
	<i>SOCAIN</i>	$2,331E+02$	$1,462E+01$	$2,383E+02$	$2,015E+02$	
f11	<i>opt-aiNet</i>	$3,938E+01$	$8,741E-01$	$3,936E+01$	$3,793E+01$	$5,998E-01$ (-)
	<i>SOCAIN</i>	$3,928E+01$	$1,165E+00$	$3,961E+01$	$3,638E+01$	
f12	<i>opt-aiNet</i>	$1,471E+05$	$1,963E+04$	$1,438E+05$	$1,115E+05$	$1,229E-05$ (S-)
	<i>SOCAIN</i>	$3,342E+05$	$3,702E+04$	$3,305E+05$	$2,700E+05$	
f13	<i>opt-aiNet</i>	$2,302E+01$	$1,378E+00$	$2,330E+01$	$1,948E+01$	$1,229E-05$ (S-)
	<i>SOCAIN</i>	$2,978E+01$	$1,939E+00$	$2,954E+01$	$2,446E+01$	
f14	<i>opt-aiNet</i>	$1,346E+01$	$1,738E-01$	$1,348E+01$	$1,313E+01$	$7,775E-01$ (+)
	<i>SOCAIN</i>	$1,343E+01$	$2,340E-01$	$1,340E+01$	$1,293E+01$	
f15	<i>opt-aiNet</i>	$4,832E+02$	$2,311E+00$	$4,837E+02$	$4,742E+02$	$1,229E-05$ (S-)
	<i>SOCAIN</i>	$5,179E+02$	$2,213E+01$	$5,137E+02$	$5,025E+02$	
f16	<i>opt-aiNet</i>	$2,855E+02$	$1,575E+01$	$2,860E+02$	$2,339E+02$	$4,530E-03$ (S+)
	<i>SOCAIN</i>	$2,734E+02$	$1,376E+01$	$2,751E+02$	$2,417E+02$	
f17	<i>opt-aiNet</i>	$3,200E+02$	$2,336E+01$	$3,194E+02$	$2,764E+02$	$2,665E-04$ (S+)
	<i>SOCAIN</i>	$2,970E+02$	$1,359E+01$	$2,975E+02$	$2,757E+02$	
f18	<i>opt-aiNet</i>	$9,009E+02$	$3,757E+01$	$9,177E+02$	$7,614E+02$	$1,229E-05$ (S-)
	<i>SOCAIN</i>	$9,333E+02$	$3,137E+00$	$9,340E+02$	$9,277E+02$	
f19	<i>opt-aiNet</i>	$9,057E+02$	$2,554E+01$	$9,174E+02$	$8,430E+02$	$1,229E-05$ (S-)
	<i>SOCAIN</i>	$9,339E+02$	$2,886E+00$	$9,341E+02$	$9,276E+02$	
f20	<i>opt-aiNet</i>	$8,959E+02$	$5,677E+01$	$9,173E+02$	$6,549E+02$	$1,229E-05$ (S-)
	<i>SOCAIN</i>	$9,339E+02$	$3,367E+00$	$9,332E+02$	$9,272E+02$	
f21	<i>opt-aiNet</i>	$6,077E+02$	$1,177E+01$	$6,132E+02$	$5,844E+02$	$1,229E-05$ (S-)
	<i>SOCAIN</i>	$8,354E+02$	$2,187E+01$	$8,325E+02$	$7,805E+02$	
f22	<i>opt-aiNet</i>	$9,676E+02$	$1,676E+01$	$9,710E+02$	$9,296E+02$	$2,012E-01$ (-)
	<i>SOCAIN</i>	$9,730E+02$	$8,367E+00$	$9,728E+02$	$9,575E+02$	
f23	<i>opt-aiNet</i>	$6,276E+02$	$1,510E+01$	$6,308E+02$	$5,889E+02$	$1,229E-05$ (S-)
	<i>SOCAIN</i>	$8,414E+02$	$2,394E+01$	$8,406E+02$	$7,887E+02$	
f24	<i>opt-aiNet</i>	$3,886E+02$	$1,425E+01$	$3,881E+02$	$3,639E+02$	$1,229E-05$ (S-)
	<i>SOCAIN</i>	$7,333E+02$	$2,976E+01$	$7,373E+02$	$6,722E+02$	
f25	<i>opt-aiNet</i>	$2,129E+02$	$3,735E-01$	$2,128E+02$	$2,121E+02$	$1,229E-05$ (S-)
	<i>SOCAIN</i>	$2,160E+02$	$1,159E+00$	$2,157E+02$	$2,147E+02$	

Tabela A.22: Resultados dos erros $|f(x)-f(x^*)|$ referentes às execuções com $N_D = 50$. Comparação do *opt-aiNet* com o *SOCAIN*.

Função	Algoritmo	Média	Desvio padrão	Mediana	Mínimo	p-valor
f1	<i>opt-aiNet</i>	$5,248E+00$	$3,826E-01$	$5,293E+00$	$4,472E+00$	$1,229E-05$ (S-)
	<i>SOCAIN</i>	$3,918E+01$	$2,813E+00$	$3,948E+01$	$3,286E+01$	
f2	<i>opt-aiNet</i>	$1,142E+02$	$1,734E+01$	$1,130E+02$	$7,554E+01$	$1,229E-05$ (S-)
	<i>SOCAIN</i>	$4,067E+02$	$6,255E+01$	$4,144E+02$	$3,013E+02$	
f3	<i>opt-aiNet</i>	$3,946E+06$	$6,312E+05$	$3,821E+06$	$2,559E+06$	$1,229E-05$ (S-)
	<i>SOCAIN</i>	$6,950E+06$	$8,080E+05$	$7,011E+06$	$5,456E+06$	
f4	<i>opt-aiNet</i>	$1,844E+05$	$3,180E+04$	$1,769E+05$	$1,443E+05$	$1,229E-05$ (S+)
	<i>SOCAIN</i>	$6,943E+04$	$9,008E+03$	$6,679E+04$	$5,087E+04$	
f5	<i>opt-aiNet</i>	$1,272E+04$	$8,883E+02$	$1,272E+04$	$1,047E+04$	$1,229E-05$ (S+)

	<i>SOCAIN</i>	9,026E+03	9,957E+02	9,276E+03	6,720E+03	
f6	<i>opt-aiNet</i>	8,843E+02	1,088E+02	8,640E+02	6,251E+02	1,229E-05 (S-)
	<i>SOCAIN</i>	1,222E+04	3,308E+03	1,245E+04	7,566E+03	
f7	<i>opt-aiNet</i>	4,083E+03	1,978E+02	4,116E+03	3,737E+03	1,229E-05 (S+)
	<i>SOCAIN</i>	5,736E+02	1,057E+02	5,688E+02	3,882E+02	
f8	<i>opt-aiNet</i>	2,113E+01	3,316E-02	2,113E+01	2,105E+01	6,377E-01 (+)
	<i>SOCAIN</i>	2,112E+01	3,912E-02	2,112E+01	2,100E+01	
f9	<i>opt-aiNet</i>	5,075E+02	2,418E+01	5,082E+02	4,518E+02	1,229E-05 (S+)
	<i>SOCAIN</i>	4,422E+02	2,519E+01	4,448E+02	3,504E+02	
f10	<i>opt-aiNet</i>	4,972E+02	1,910E+01	4,929E+02	4,687E+02	4,797E-02 (S+)
	<i>SOCAIN</i>	4,868E+02	1,545E+01	4,846E+02	4,509E+02	
f11	<i>opt-aiNet</i>	7,266E+01	1,461E+00	7,294E+01	6,890E+01	8,824E-01 (-)
	<i>SOCAIN</i>	7,261E+01	1,870E+00	7,303E+01	6,849E+01	
f12	<i>opt-aiNet</i>	8,472E+05	6,926E+04	8,417E+05	7,101E+05	1,229E-05 (S-)
	<i>SOCAIN</i>	1,887E+06	1,488E+05	1,913E+06	1,620E+06	
f13	<i>opt-aiNet</i>	5,103E+01	2,566E+00	5,167E+01	4,560E+01	1,229E-05 (S-)
	<i>SOCAIN</i>	6,943E+01	4,005E+00	6,934E+01	6,335E+01	
f14	<i>opt-aiNet</i>	2,307E+01	2,376E-01	2,312E+01	2,239E+01	8,824E-01 (+)
	<i>SOCAIN</i>	2,309E+01	2,319E-01	2,311E+01	2,235E+01	
f15	<i>opt-aiNet</i>	4,644E+02	1,302E+00	4,645E+02	4,626E+02	1,229E-05 (S-)
	<i>SOCAIN</i>	5,114E+02	3,323E+00	5,116E+02	5,034E+02	
f16	<i>opt-aiNet</i>	3,635E+02	1,453E+01	3,632E+02	3,295E+02	1,079E-03 (S+)
	<i>SOCAIN</i>	3,459E+02	1,467E+01	3,428E+02	3,195E+02	
f17	<i>opt-aiNet</i>	3,988E+02	2,039E+01	3,997E+02	3,588E+02	4,221E-02 (S+)
	<i>SOCAIN</i>	3,866E+02	1,381E+01	3,861E+02	3,449E+02	
f18	<i>opt-aiNet</i>	9,374E+02	2,369E+01	9,449E+02	8,656E+02	1,229E-05 (S-)
	<i>SOCAIN</i>	9,822E+02	5,249E+00	9,816E+02	9,718E+02	
f19	<i>opt-aiNet</i>	9,315E+02	2,777E+01	9,443E+02	8,656E+02	1,229E-05 (S-)
	<i>SOCAIN</i>	9,826E+02	6,851E+00	9,832E+02	9,673E+02	
f20	<i>opt-aiNet</i>	9,427E+02	1,475E+01	9,456E+02	8,728E+02	1,229E-05 (S-)
	<i>SOCAIN</i>	9,849E+02	5,623E+00	9,844E+02	9,720E+02	
f21	<i>opt-aiNet</i>	6,833E+02	1,343E+01	6,845E+02	6,524E+02	1,229E-05 (S-)
	<i>SOCAIN</i>	9,895E+02	2,299E+01	9,916E+02	9,378E+02	
f22	<i>opt-aiNet</i>	1,025E+03	1,186E+01	1,026E+03	9,951E+02	1,281E-02 (S-)
	<i>SOCAIN</i>	1,036E+03	1,186E+01	1,036E+03	1,015E+03	
f23	<i>opt-aiNet</i>	7,146E+02	1,900E+01	7,130E+02	6,770E+02	1,229E-05 (S-)
	<i>SOCAIN</i>	9,993E+02	1,995E+01	1,002E+03	9,495E+02	
f24	<i>opt-aiNet</i>	5,455E+02	2,678E+01	5,508E+02	4,696E+02	1,229E-05 (S-)
	<i>SOCAIN</i>	9,620E+02	3,047E+01	9,707E+02	8,773E+02	
f25	<i>opt-aiNet</i>	2,276E+02	1,321E+00	2,276E+02	2,251E+02	1,229E-05 (S-)
	<i>SOCAIN</i>	2,472E+02	2,796E+00	2,469E+02	2,421E+02	

Tabela A.23: Resultados dos erros $|f(x)-f(x^*)|$ referentes às execuções com $N_D = 2$. Comparação do *opt-aiNet* com o (q)*opt-aiNet*.

Função	Algoritmo	Média	Desvio padrão	Mediana	Mínimo	p-valor
f1	<i>opt-aiNet</i>	5,558E-04	1,671E-03	1,561E-04	2,019E-05	1,011E-04 (S+)

	(q) <i>opt-aiNet</i>	3,045E-05	2,799E-05	3,086E-05	2,440E-06	
f2	<i>opt-aiNet</i>	3,452E-04	3,588E-04	1,434E-04	2,543E-05	1,229E-05 (S+)
	(q) <i>opt-aiNet</i>	3,298E-05	3,969E-05	1,939E-05	1,200E-07	
f3	<i>opt-aiNet</i>	2,395E+02	3,374E+02	1,596E+02	8,726E+00	1,603E-02 (S+)
	(q) <i>opt-aiNet</i>	8,897E+01	9,697E+01	5,077E+01	9,320E-01	
f4	<i>opt-aiNet</i>	1,880E+02	5,039E+02	7,757E-03	4,115E-05	2,257E-05 (S+)
	(q) <i>opt-aiNet</i>	2,132E-04	2,488E-04	9,464E-05	1,867E-05	
f5	<i>opt-aiNet</i>	2,319E+02	4,923E+02	1,381E+00	9,790E-02	1,773E-05 (S+)
	(q) <i>opt-aiNet</i>	9,323E-02	4,762E-02	9,161E-02	2,601E-02	
f6	<i>opt-aiNet</i>	8,799E+00	1,900E+01	1,058E-02	1,681E-05	1,885E-03 (S+)
	(q) <i>opt-aiNet</i>	6,123E-03	1,091E-02	2,198E-03	1,635E-04	
f7	<i>opt-aiNet</i>	1,074E+02	2,919E+01	1,050E+02	6,592E+01	1,229E-05 (S+)
	(q) <i>opt-aiNet</i>	2,085E+00	1,922E+00	1,662E+00	6,073E-02	
f8	<i>opt-aiNet</i>	1,446E+01	6,040E+00	1,679E+01	1,377E+00	3,467E-02 (S+)
	(q) <i>opt-aiNet</i>	1,102E+01	7,615E+00	1,005E+01	4,840E-02	
f9	<i>opt-aiNet</i>	8,237E-03	9,818E-03	5,622E-03	2,376E-04	3,967E-01 (-)
	(q) <i>opt-aiNet</i>	1,067E-02	1,399E-02	6,441E-03	9,099E-05	
f10	<i>opt-aiNet</i>	1,461E-02	1,459E-02	1,012E-02	7,025E-04	6,186E-01 (-)
	(q) <i>opt-aiNet</i>	1,521E-02	1,264E-02	1,235E-02	1,277E-03	
f11	<i>opt-aiNet</i>	1,502E-01	4,393E-02	1,477E-01	8,075E-02	7,366E-01 (+)
	(q) <i>opt-aiNet</i>	1,445E-01	5,329E-02	1,473E-01	4,064E-02	
f12	<i>opt-aiNet</i>	7,164E-02	6,741E-02	4,573E-02	1,330E-02	1,658E-01 (+)
	(q) <i>opt-aiNet</i>	4,527E-02	5,280E-02	2,309E-02	6,581E-05	
f13	<i>opt-aiNet</i>	6,161E-05	7,425E-05	4,104E-05	2,000E-08	5,272E-01 (+)
	(q) <i>opt-aiNet</i>	6,360E-05	1,025E-04	2,381E-05	0,000E+00	
f14	<i>opt-aiNet</i>	2,982E-01	2,087E-01	2,608E-01	1,943E-02	6,022E-04 (S+)
	(q) <i>opt-aiNet</i>	7,821E-02	7,802E-02	7,445E-02	1,943E-02	
f15	<i>opt-aiNet</i>	1,038E+00	9,489E-01	7,617E-01	4,004E-02	1,855E-02 (S+)
	(q) <i>opt-aiNet</i>	4,706E-01	6,200E-01	3,451E-01	5,794E-03	
f16	<i>opt-aiNet</i>	2,111E+00	7,874E+00	3,272E-01	4,497E-02	6,766E-01 (-)
	(q) <i>opt-aiNet</i>	6,143E-01	8,666E-01	3,525E-01	1,649E-02	
f17	<i>opt-aiNet</i>	1,297E+00	1,478E+00	6,374E-01	6,473E-02	2,831E-02 (S+)
	(q) <i>opt-aiNet</i>	4,679E-01	7,498E-01	2,810E-01	1,653E-02	
f18	<i>opt-aiNet</i>	3,746E+01	4,933E+01	1,869E+01	4,867E+00	1,018E-02 (S+)
	(q) <i>opt-aiNet</i>	1,492E+01	1,078E+01	1,223E+01	2,725E+00	
f19	<i>opt-aiNet</i>	1,632E+02	3,428E+01	1,653E+02	1,050E+02	2,312E-01 (-)
	(q) <i>opt-aiNet</i>	1,740E+02	3,514E+01	2,002E+02	1,035E+02	
f20	<i>opt-aiNet</i>	4,784E+01	5,733E+01	3,404E+01	1,068E+00	1,285E-01 (+)
	(q) <i>opt-aiNet</i>	6,163E+01	7,866E+01	2,524E+01	2,562E+00	
f21	<i>opt-aiNet</i>	7,844E+01	8,131E+01	2,956E+01	3,058E+00	7,164E-01 (+)
	(q) <i>opt-aiNet</i>	8,507E+01	8,808E+01	2,872E+01	3,148E-01	
f22	<i>opt-aiNet</i>	1,762E+02	4,768E+01	2,002E+02	1,457E+01	4,758E-01
	(q) <i>opt-aiNet</i>	1,726E+02	5,523E+01	2,002E+02	2,203E+01	
f23	<i>opt-aiNet</i>	9,881E+01	1,036E+02	4,871E+01	1,187E-01	4,237E-01 (+)
	(q) <i>opt-aiNet</i>	6,666E+01	8,500E+01	2,640E+01	9,789E-01	
f24	<i>opt-aiNet</i>	1,794E+02	3,020E+01	2,000E+02	1,002E+02	4,432E-01

	(q) opt-aiNet	1,866E+02	2,381E+01	2,000E+02	1,067E+02	
f25	opt-aiNet	1,554E+02	2,277E+01	1,498E+02	1,202E+02	2,699E-03 (S-)
	(q) opt-aiNet	1,805E+02	2,200E+01	1,885E+02	1,411E+02	

Tabela A.24: Resultados dos erros $|f(x)-f(x^*)|$ referentes às execuções com $N_D = 10$. Comparação do opt-aiNet com o (q)opt-aiNet.

Função	Algoritmo	Média	Desvio padrão	Mediana	Mínimo	p-valor
f1	opt-aiNet	1,035E-01	2,889E-02	1,004E-01	4,990E-02	2,879E-01 (S+)
	(q) opt-aiNet	9,369E-02	2,767E-02	9,678E-02	5,449E-02	
f2	opt-aiNet	2,583E-01	8,776E-02	2,456E-01	1,200E-01	1,773E-05 (S+)
	(q) opt-aiNet	1,659E-01	4,013E-02	1,684E-01	8,776E-02	
f3	opt-aiNet	1,154E+05	5,837E+04	1,155E+05	1,369E+04	1,079E-03 (S+)
	(q) opt-aiNet	7,139E+04	3,645E+04	6,837E+04	1,463E+04	
f4	opt-aiNet	1,889E+04	9,042E+03	1,695E+04	5,135E+03	1,229E-05 (S+)
	(q) opt-aiNet	4,299E+03	2,446E+03	4,621E+03	1,015E+02	
f5	opt-aiNet	1,469E+03	7,317E+02	1,341E+03	6,678E+01	1,229E-05 (S+)
	(q) opt-aiNet	2,918E+01	5,612E+00	2,701E+01	2,193E+01	
f6	opt-aiNet	1,945E+01	3,035E+00	1,985E+01	1,224E+01	3,031E-02 (S+)
	(q) opt-aiNet	1,749E+01	3,387E+00	1,709E+01	9,539E+00	
f7	opt-aiNet	2,309E+03	4,150E+02	2,455E+03	1,355E+03	1,229E-05 (S+)
	(q) opt-aiNet	6,365E-01	3,550E-01	6,007E-01	1,888E-01	
f8	opt-aiNet	2,034E+01	7,943E-02	2,035E+01	2,018E+01	9,036E-01 (+)
	(q) opt-aiNet	2,034E+01	7,534E-02	2,034E+01	2,011E+01	
f9	opt-aiNet	3,408E+01	4,010E+00	3,343E+01	2,510E+01	4,932E-04 (S+)
	(q) opt-aiNet	2,745E+01	4,266E+00	2,784E+01	1,641E+01	
f10	opt-aiNet	3,308E+01	4,447E+00	3,436E+01	2,341E+01	4,501E-02 (S+)
	(q) opt-aiNet	2,984E+01	5,815E+00	3,045E+01	1,783E+01	
f11	opt-aiNet	8,309E+00	5,481E-01	8,362E+00	6,723E+00	2,947E-03 (S+)
	(q) opt-aiNet	7,674E+00	6,714E-01	7,916E+00	6,219E+00	
f12	opt-aiNet	2,098E+03	8,490E+02	1,880E+03	8,914E+02	9,250E-01 (-)
	(q) opt-aiNet	2,073E+03	5,146E+02	2,033E+03	1,170E+03	
f13	opt-aiNet	3,183E+00	5,004E-01	3,361E+00	1,865E+00	6,148E-02 (-)
	(q) opt-aiNet	3,451E+00	5,065E-01	3,572E+00	1,851E+00	
f14	opt-aiNet	3,914E+00	2,110E-01	3,917E+00	3,336E+00	2,641E-01 (+)
	(q) opt-aiNet	3,843E+00	1,638E-01	3,867E+00	3,473E+00	
f15	opt-aiNet	3,400E+02	2,734E+01	3,447E+02	2,575E+02	7,333E-04 (S+)
	(q) opt-aiNet	3,026E+02	3,606E+01	3,001E+02	2,298E+02	
f16	opt-aiNet	1,832E+02	1,312E+01	1,859E+02	1,523E+02	8,401E-01 (+)
	(q) opt-aiNet	1,824E+02	1,086E+01	1,818E+02	1,648E+02	
f17	opt-aiNet	1,995E+02	1,593E+01	1,999E+02	1,666E+02	3,819E-01 (+)
	(q) opt-aiNet	1,953E+02	1,686E+01	1,954E+02	1,634E+02	
f18	opt-aiNet	4,733E+02	3,985E+01	4,674E+02	4,170E+02	9,464E-01 (-)
	(q) opt-aiNet	4,680E+02	2,990E+01	4,703E+02	4,145E+02	
f19	opt-aiNet	4,729E+02	3,594E+01	4,774E+02	4,142E+02	5,272E-01 (+)
	(q) opt-aiNet	4,807E+02	2,981E+01	4,754E+02	4,095E+02	
f20	opt-aiNet	4,613E+02	2,559E+01	4,598E+02	3,910E+02	8,612E-01 (-)

	(q) opt-aiNet	4,643E+02	2,821E+01	4,706E+02	4,078E+02	
f21	opt-aiNet	4,767E+02	7,794E+01	5,257E+02	3,276E+02	6,148E-02 (-)
	(q) opt-aiNet	5,072E+02	6,303E+01	5,359E+02	3,392E+02	
f22	opt-aiNet	6,659E+02	1,504E+02	7,793E+02	4,061E+02	6,148E-02 (-)
	(q) opt-aiNet	7,516E+02	9,385E+01	7,847E+02	4,994E+02	
f23	opt-aiNet	5,333E+02	5,721E+01	5,630E+02	4,252E+02	5,485E-01 (+)
	(q) opt-aiNet	5,394E+02	4,625E+01	5,616E+02	4,252E+02	
f24	opt-aiNet	2,409E+02	8,977E+00	2,416E+02	2,243E+02	7,366E-01 (-)
	(q) opt-aiNet	2,414E+02	9,012E+00	2,432E+02	2,267E+02	
f25	opt-aiNet	2,569E+02	3,440E+01	2,545E+02	2,282E+02	9,042E-05 (S-)
	(q) opt-aiNet	3,297E+02	6,897E+01	2,828E+02	2,559E+02	

Tabela A.25: Resultados dos erros $|f(x)-f(x^*)|$ referentes às execuções com $N_D = 30$. Comparação do opt-aiNet com o (q)opt-aiNet.

Função	Algoritmo	Média	Desvio padrão	Mediana	Mínimo	p-valor
f1	opt-aiNet	1,761E+00	1,419E-01	1,790E+00	1,409E+00	5,451E-04 (S+)
	(q) opt-aiNet	1,570E+00	1,752E-01	1,588E+00	1,191E+00	
f2	opt-aiNet	1,237E+01	3,397E+00	1,140E+01	7,307E+00	1,500E-01 (+)
	(q) opt-aiNet	1,112E+01	1,555E+00	1,095E+01	9,022E+00	
f3	opt-aiNet	1,696E+06	3,837E+05	1,793E+06	9,158E+05	4,926E-01 (+)
	(q) opt-aiNet	1,633E+06	3,599E+05	1,615E+06	1,015E+06	
f4	opt-aiNet	8,952E+04	1,892E+04	9,082E+04	5,420E+04	1,229E-05 (S+)
	(q) opt-aiNet	3,442E+04	6,932E+03	3,584E+04	2,041E+04	
f5	opt-aiNet	5,873E+03	5,126E+02	5,869E+03	4,889E+03	5,756E-05 (S+)
	(q) opt-aiNet	5,013E+03	5,565E+02	4,969E+03	4,125E+03	
f6	opt-aiNet	2,486E+02	3,411E+01	2,476E+02	1,812E+02	6,571E-01 (+)
	(q) opt-aiNet	2,582E+02	7,428E+01	2,358E+02	1,822E+02	
f7	opt-aiNet	4,845E+03	2,770E+02	4,822E+03	4,342E+03	1,229E-05 (S+)
	(q) opt-aiNet	4,160E-01	4,387E-02	4,271E-01	3,444E-01	
f8	opt-aiNet	2,095E+01	5,462E-02	2,097E+01	2,082E+01	7,775E-01 (+)
	(q) opt-aiNet	2,094E+01	5,064E-02	2,096E+01	2,082E+01	
f9	opt-aiNet	2,396E+02	1,066E+01	2,393E+02	2,153E+02	2,864E-05 (S+)
	(q) opt-aiNet	2,076E+02	1,325E+01	2,064E+02	1,770E+02	
f10	opt-aiNet	2,288E+02	1,504E+01	2,310E+02	1,986E+02	5,998E-01 (S+)
	(q) opt-aiNet	2,264E+02	1,971E+01	2,282E+02	1,817E+02	
f11	opt-aiNet	3,938E+01	8,741E-01	3,936E+01	3,793E+01	4,926E-01 (-)
	(q) opt-aiNet	3,903E+01	1,809E+00	3,949E+01	3,296E+01	
f12	opt-aiNet	1,471E+05	1,963E+04	1,438E+05	1,115E+05	5,272E-01 (+)
	(q) opt-aiNet	1,487E+05	1,797E+04	1,422E+05	1,067E+05	
f13	opt-aiNet	2,302E+01	1,378E+00	2,330E+01	1,948E+01	9,464E-01 (+)
	(q) opt-aiNet	2,297E+01	1,768E+00	2,294E+01	1,747E+01	
f14	opt-aiNet	1,346E+01	1,738E-01	1,348E+01	1,313E+01	6,964E-01 (-)
	(q) opt-aiNet	1,348E+01	1,609E-01	1,349E+01	1,310E+01	
f15	opt-aiNet	4,832E+02	2,311E+00	4,837E+02	4,742E+02	1,829E-01 (+)
	(q) opt-aiNet	4,825E+02	3,768E+00	4,834E+02	4,652E+02	
f16	opt-aiNet	2,855E+02	1,575E+01	2,860E+02	2,339E+02	2,699E-03 (S+)

	(q) opt-aiNet	2,696E+02	1,523E+01	2,676E+02	2,438E+02	
f17	opt-aiNet	3,200E+02	2,336E+01	3,194E+02	2,764E+02	1,188E-02 (S+)
	(q) opt-aiNet	3,052E+02	1,511E+01	3,098E+02	2,651E+02	
f18	opt-aiNet	9,009E+02	3,757E+01	9,177E+02	7,614E+02	2,209E-01 (+)
	(q) opt-aiNet	8,832E+02	5,617E+01	9,173E+02	6,696E+02	
f19	opt-aiNet	9,057E+02	2,554E+01	9,174E+02	8,430E+02	6,377E-01 (-)
	(q) opt-aiNet	8,935E+02	5,277E+01	9,177E+02	6,825E+02	
f20	opt-aiNet	8,959E+02	5,677E+01	9,173E+02	6,549E+02	6,571E-01 (+)
	(q) opt-aiNet	8,917E+02	3,379E+01	9,169E+02	8,436E+02	
f21	opt-aiNet	6,077E+02	1,177E+01	6,132E+02	5,844E+02	2,312E-01 (+)
	(q) opt-aiNet	6,032E+02	1,216E+01	6,045E+02	5,781E+02	
f22	opt-aiNet	9,676E+02	1,676E+01	9,710E+02	9,296E+02	1,919E-01 (+)
	(q) opt-aiNet	9,619E+02	1,475E+01	9,632E+02	9,277E+02	
f23	opt-aiNet	6,276E+02	1,510E+01	6,308E+02	5,889E+02	2,528E-01 (+)
	(q) opt-aiNet	6,259E+02	1,141E+01	6,253E+02	6,057E+02	
f24	opt-aiNet	3,886E+02	1,425E+01	3,881E+02	3,639E+02	8,265E-02 (+)
	(q) opt-aiNet	3,802E+02	2,049E+01	3,812E+02	3,321E+02	
f25	opt-aiNet	2,129E+02	3,735E-01	2,128E+02	2,121E+02	2,142E-02 (S-)
	(q) opt-aiNet	2,131E+02	3,216E-01	2,131E+02	2,125E+02	

Tabela A.26: Resultados dos erros $|f(x)-f(x^*)|$ referentes às execuções com $N_D = 50$. Comparação do opt-aiNet com o (q)opt-aiNet.

Função	Algoritmo	Média	Desvio padrão	Mediana	Mínimo	p-valor
f1	opt-aiNet	5,248E+00	3,826E-01	5,293E+00	4,472E+00	1,155E-01 (+)
	(q) opt-aiNet	5,112E+00	2,440E-01	5,162E+00	4,596E+00	
f2	opt-aiNet	1,142E+02	1,734E+01	1,130E+02	7,554E+01	3,243E-02 (S+)
	(q) opt-aiNet	1,041E+02	1,470E+01	1,042E+02	6,685E+01	
f3	opt-aiNet	3,946E+06	6,312E+05	3,821E+06	2,559E+06	2,758E-01 (+)
	(q) opt-aiNet	3,712E+06	6,911E+05	3,578E+06	2,621E+06	
f4	opt-aiNet	1,844E+05	3,180E+04	1,769E+05	1,443E+05	1,229E-05 (S+)
	(q) opt-aiNet	8,904E+04	1,256E+04	9,013E+04	5,914E+04	
f5	opt-aiNet	1,272E+04	8,883E+02	1,272E+04	1,047E+04	9,464E-01 (-)
	(q) opt-aiNet	1,262E+04	9,656E+02	1,278E+04	1,077E+04	
f6	opt-aiNet	8,843E+02	1,088E+02	8,640E+02	6,251E+02	4,797E-02 (S+)
	(q) opt-aiNet	8,173E+02	1,480E+02	8,222E+02	5,759E+02	
f7	opt-aiNet	4,083E+03	1,978E+02	4,116E+03	3,737E+03	1,229E-05 (S+)
	(q) opt-aiNet	6,475E-01	3,384E-02	6,433E-01	5,738E-01	
f8	opt-aiNet	2,113E+01	3,316E-02	2,113E+01	2,105E+01	4,118E-01 (-)
	(q) opt-aiNet	2,114E+01	3,771E-02	2,114E+01	2,107E+01	
f9	opt-aiNet	5,075E+02	2,418E+01	5,082E+02	4,518E+02	1,773E-05 (S+)
	(q) opt-aiNet	4,523E+02	2,036E+01	4,491E+02	4,101E+02	
f10	opt-aiNet	4,972E+02	1,910E+01	4,929E+02	4,687E+02	2,259E-03 (S+)
	(q) opt-aiNet	4,766E+02	2,751E+01	4,829E+02	3,950E+02	
f11	opt-aiNet	7,266E+01	1,461E+00	7,294E+01	6,890E+01	8,824E-01 (-)
	(q) opt-aiNet	7,276E+01	1,550E+00	7,309E+01	6,933E+01	
f12	opt-aiNet	8,472E+05	6,926E+04	8,417E+05	7,101E+05	9,250E-01 (-)

	(q) <i>opt-aiNet</i>	8,480E+05	7,387E+04	8,484E+05	7,339E+05	
f13	<i>opt-aiNet</i>	5,103E+01	2,566E+00	5,167E+01	4,560E+01	1,285E-01 (-)
	(q) <i>opt-aiNet</i>	5,216E+01	2,266E+00	5,242E+01	4,764E+01	
f14	<i>opt-aiNet</i>	2,307E+01	2,376E-01	2,312E+01	2,239E+01	2,641E-01 (-)
	(q) <i>opt-aiNet</i>	2,314E+01	1,782E-01	2,317E+01	2,276E+01	
f15	<i>opt-aiNet</i>	4,644E+02	1,302E+00	4,645E+02	4,626E+02	9,678E-01 (+)
	(q) <i>opt-aiNet</i>	4,640E+02	2,055E+00	4,642E+02	4,587E+02	
f16	<i>opt-aiNet</i>	3,635E+02	1,453E+01	3,632E+02	3,295E+02	3,507E-03 (S+)
	(q) <i>opt-aiNet</i>	3,492E+02	1,959E+01	3,542E+02	2,956E+02	
f17	<i>opt-aiNet</i>	3,988E+02	2,039E+01	3,997E+02	3,588E+02	6,531E-02 (+)
	(q) <i>opt-aiNet</i>	3,888E+02	1,848E+01	3,875E+02	3,521E+02	
f18	<i>opt-aiNet</i>	9,374E+02	2,369E+01	9,449E+02	8,656E+02	5,098E-01 (-)
	(q) <i>opt-aiNet</i>	9,449E+02	2,047E+00	9,452E+02	9,411E+02	
f19	<i>opt-aiNet</i>	9,315E+02	2,777E+01	9,443E+02	8,656E+02	2,312E-01 (-)
	(q) <i>opt-aiNet</i>	9,405E+02	1,590E+01	9,449E+02	8,799E+02	
f20	<i>opt-aiNet</i>	9,427E+02	1,475E+01	9,456E+02	8,728E+02	2,312E-01 (+)
	(q) <i>opt-aiNet</i>	9,384E+02	1,953E+01	9,455E+02	8,701E+02	
f21	<i>opt-aiNet</i>	6,833E+02	1,343E+01	6,845E+02	6,524E+02	2,528E-01 (+)
	(q) <i>opt-aiNet</i>	6,804E+02	1,320E+01	6,813E+02	6,496E+02	
f22	<i>opt-aiNet</i>	1,025E+03	1,186E+01	1,026E+03	9,951E+02	2,641E-01 (+)
	(q) <i>opt-aiNet</i>	1,022E+03	9,979E+00	1,021E+03	1,004E+03	
f23	<i>opt-aiNet</i>	7,146E+02	1,900E+01	7,130E+02	6,770E+02	6,186E-01 (+)
	(q) <i>opt-aiNet</i>	7,115E+02	1,811E+01	7,111E+02	6,684E+02	
f24	<i>opt-aiNet</i>	5,455E+02	2,678E+01	5,508E+02	4,696E+02	1,578E-01 (+)
	(q) <i>opt-aiNet</i>	5,381E+02	1,851E+01	5,411E+02	4,938E+02	
f25	<i>opt-aiNet</i>	2,276E+02	1,321E+00	2,276E+02	2,251E+02	6,964E-01
	(q) <i>opt-aiNet</i>	2,278E+02	1,400E+00	2,276E+02	2,256E+02	

Tabela A.27: Resultados dos erros $|f(x)-f(x^*)|$ referentes às execuções com $N_D = 2$. Comparação do SOCAIN com o (q)SOCAIN.

Função	Algoritmo	Média	Desvio padrão	Mediana	Mínimo	p-valor
f1	SOCAIN	3,281E-04	5,840E-04	1,523E-04	2,942E-05	9,263E-02 (+)
	(q) SOCAIN	1,523E-04	1,311E-04	1,319E-04	4,780E-06	
f2	SOCAIN	5,896E-04	6,790E-04	2,715E-04	6,063E-05	4,221E-02 (S+)
	(q) SOCAIN	2,693E-04	2,177E-04	2,051E-04	1,994E-05	
f3	SOCAIN	1,501E+02	1,877E+02	9,629E+01	7,188E+00	4,927E-03 (S+)
	(q) SOCAIN	4,938E+01	6,639E+01	3,562E+01	1,130E-01	
f4	SOCAIN	1,357E-03	1,077E-03	9,623E-04	5,460E-05	1,571E-05 (S+)
	(q) SOCAIN	4,712E-04	5,007E-04	4,274E-04	9,300E-06	
f5	SOCAIN	8,238E-01	1,099E+00	4,683E-01	1,216E-01	6,848E-03 (S+)
	(q) SOCAIN	2,502E-01	1,991E-01	1,995E-01	1,658E-02	
f6	SOCAIN	1,073E-02	1,078E-02	7,170E-03	1,176E-03	1,094E-01 (+)
	(q) SOCAIN	6,212E-03	5,871E-03	5,062E-03	1,082E-04	
f7	SOCAIN	4,577E+01	1,266E+01	4,360E+01	2,900E+01	1,229E-05 (S+)

	(q) SOCAIN	2,855E-01	3,346E-01	1,659E-01	1,352E-02	
f8	SOCAIN	1,065E+01	7,654E+00	8,988E+00	7,665E-01	3,819E-01 (+)
	(q) SOCAIN	8,949E+00	6,874E+00	7,287E+00	5,460E-01	
f9	SOCAIN	2,516E-02	2,830E-02	1,471E-02	5,021E-04	5,451E-04 (S+)
	(q) SOCAIN	8,351E-03	1,100E-02	4,382E-03	2,398E-04	
f10	SOCAIN	4,092E-02	4,515E-02	2,312E-02	2,971E-04	5,098E-01 (+)
	(q) SOCAIN	2,591E-02	3,024E-02	1,309E-02	4,934E-04	
f11	SOCAIN	1,455E-01	7,086E-02	1,272E-01	5,986E-03	4,118E-01 (-)
	(q) SOCAIN	1,667E-01	6,245E-02	1,711E-01	5,434E-02	
f12	SOCAIN	1,311E-01	1,108E-01	8,665E-02	9,772E-03	8,824E-01 (-)
	(q) SOCAIN	1,546E-01	1,764E-01	8,863E-02	3,315E-03	
f13	SOCAIN	1,520E-04	2,452E-04	5,896E-05	1,000E-08	8,191E-01 (+)
	(q) SOCAIN	1,740E-04	3,886E-04	3,391E-05	2,000E-08	
f14	SOCAIN	2,036E-01	2,144E-01	1,564E-01	1,292E-02	2,543E-05 (S+)
	(q) SOCAIN	2,268E-02	1,659E-02	1,943E-02	5,175E-04	
f15	SOCAIN	2,570E+00	2,484E+00	1,765E+00	1,350E-01	1,186E-03 (S+)
	(q) SOCAIN	7,056E-01	1,160E+00	2,902E-01	3,030E-02	
f16	SOCAIN	2,815E+00	3,369E+00	1,416E+00	2,327E-02	2,831E-02 (S+)
	(q) SOCAIN	1,061E+00	1,081E+00	6,033E-01	3,702E-02	
f17	SOCAIN	3,828E+00	5,725E+00	2,037E+00	3,076E-02	1,919E-01 (+)
	(q) SOCAIN	1,517E+00	1,630E+00	7,023E-01	5,539E-02	
f18	SOCAIN	7,350E+01	6,541E+01	5,066E+01	2,421E+00	3,822E-03 (S+)
	(q) SOCAIN	3,869E+01	4,714E+01	2,343E+01	3,708E+00	
f19	SOCAIN	1,932E+02	4,147E+01	2,075E+02	1,041E+02	1,425E-01 (+)
	(q) SOCAIN	1,826E+02	4,321E+01	1,937E+02	1,084E+02	
f20	SOCAIN	1,070E+02	8,509E+01	6,428E+01	7,336E+00	8,753E-02 (+)
	(q) SOCAIN	7,978E+01	7,066E+01	6,345E+01	1,500E+00	
f21	SOCAIN	1,103E+02	7,242E+01	1,040E+02	4,952E+00	6,377E-01 (-)
	(q) SOCAIN	1,013E+02	7,618E+01	1,077E+02	1,332E+00	
f22	SOCAIN	1,772E+02	4,708E+01	2,002E+02	4,976E+01	7,982E-01 (+)
	(q) SOCAIN	1,688E+02	5,694E+01	2,001E+02	3,367E+01	
f23	SOCAIN	1,484E+02	1,061E+02	1,459E+02	3,806E+00	3,896E-01 (+)
	(q) SOCAIN	1,218E+02	1,028E+02	9,157E+01	4,996E-01	
f24	SOCAIN	1,926E+02	1,742E+01	2,002E+02	1,411E+02	1,829E-01 (+)
	(q) SOCAIN	1,962E+02	1,130E+01	2,001E+02	1,540E+02	
f25	SOCAIN	1,618E+02	1,943E+01	1,609E+02	1,182E+02	3,704E-02 (S-)
	(q) SOCAIN	1,764E+02	2,602E+01	1,791E+02	1,121E+02	

Tabela A.28: Resultados dos erros $|f(x)-f(x^*)|$ referentes às execuções com $N_D = 10$. Comparação do SOCAIN com o (q)SOCAIN.

Função	Algoritmo	Média	Desvio padrão	Mediana	Mínimo	p-valor
f1	SOCAIN	7,657E-01	2,252E-01	8,058E-01	2,430E-01	3,822E-03 (S+)
	(q) SOCAIN	5,543E-01	1,773E-01	5,985E-01	1,970E-01	
f2	SOCAIN	1,124E+00	3,516E-01	1,167E+00	3,830E-01	5,812E-01 (+)
	(q) SOCAIN	1,174E+00	2,891E-01	1,087E+00	7,157E-01	
f3	SOCAIN	9,931E+04	5,340E+04	9,082E+04	1,691E+04	5,998E-01 (+)

	(q) SOCAIN	8,856E+04	3,797E+04	8,173E+04	2,789E+04	
f4	SOCAIN	5,204E+03	4,461E+03	4,705E+03	2,449E+01	1,229E-05 (S+)
	(q) SOCAIN	6,800E+01	1,429E+02	1,026E+01	2,389E+00	
f5	SOCAIN	7,785E+01	1,392E+01	8,048E+01	5,213E+01	4,028E-04 (S+)
	(q) SOCAIN	6,165E+01	8,668E+00	6,041E+01	4,922E+01	
f6	SOCAIN	8,242E+01	2,210E+01	8,171E+01	4,098E+01	6,377E-01 (+)
	(q) SOCAIN	7,979E+01	2,205E+01	7,626E+01	4,566E+01	
f7	SOCAIN	8,280E+02	2,346E+02	8,936E+02	2,872E+02	1,229E-05 (S+)
	(q) SOCAIN	7,005E-01	1,182E-01	7,437E-01	4,930E-01	
f8	SOCAIN	2,035E+01	6,798E-02	2,037E+01	2,018E+01	1,919E-01 (-)
	(q) SOCAIN	2,038E+01	8,641E-02	2,040E+01	2,017E+01	
f9	SOCAIN	3,326E+01	3,638E+00	3,257E+01	2,578E+01	4,932E-04 (S+)
	(q) SOCAIN	2,852E+01	3,756E+00	2,774E+01	2,218E+01	
f10	SOCAIN	3,581E+01	3,893E+00	3,628E+01	2,476E+01	8,824E-01 (+)
	(q) SOCAIN	3,591E+01	4,909E+00	3,622E+01	2,661E+01	
f11	SOCAIN	7,127E+00	7,785E-01	7,102E+00	5,688E+00	1,725E-02 (S+)
	(q) SOCAIN	6,523E+00	6,699E-01	6,545E+00	5,213E+00	
f12	SOCAIN	7,207E+03	1,953E+03	7,684E+03	1,835E+03	2,699E-03 (S+)
	(q) SOCAIN	5,504E+03	1,989E+03	5,600E+03	2,176E+03	
f13	SOCAIN	3,851E+00	5,137E-01	3,790E+00	3,017E+00	8,753E-02 (+)
	(q) SOCAIN	3,574E+00	6,230E-01	3,690E+00	2,336E+00	
f14	SOCAIN	3,862E+00	2,394E-01	3,995E+00	3,383E+00	9,804E-04 (S+)
	(q) SOCAIN	3,641E+00	2,761E-01	3,577E+00	3,027E+00	
f15	SOCAIN	3,414E+02	2,215E+01	3,425E+02	3,062E+02	1,430E-03 (S+)
	(q) SOCAIN	3,069E+02	4,064E+01	3,033E+02	2,316E+02	
f16	SOCAIN	1,843E+02	1,522E+01	1,823E+02	1,504E+02	4,593E-01 (-)
	(q) SOCAIN	1,806E+02	1,593E+01	1,832E+02	1,506E+02	
f17	SOCAIN	1,996E+02	1,653E+01	2,038E+02	1,632E+02	2,312E-01 (-)
	(q) SOCAIN	2,045E+02	1,184E+01	2,069E+02	1,827E+02	
f18	SOCAIN	5,782E+02	3,262E+01	5,815E+02	5,058E+02	6,377E-01 (+)
	(q) SOCAIN	5,835E+02	3,844E+01	5,756E+02	5,204E+02	
f19	SOCAIN	5,747E+02	3,489E+01	5,765E+02	4,734E+02	8,191E-01 (-)
	(q) SOCAIN	5,728E+02	3,271E+01	5,782E+02	4,968E+02	
f20	SOCAIN	5,942E+02	6,399E+01	5,945E+02	4,735E+02	6,186E-01 (+)
	(q) SOCAIN	5,841E+02	2,872E+01	5,855E+02	5,287E+02	
f21	SOCAIN	6,764E+02	6,816E+01	6,555E+02	5,991E+02	8,401E-01 (-)
	(q) SOCAIN	6,689E+02	5,004E+01	6,725E+02	5,839E+02	
f22	SOCAIN	8,008E+02	1,034E+01	8,028E+02	7,645E+02	8,265E-02 (-)
	(q) SOCAIN	8,055E+02	8,611E+00	8,055E+02	7,813E+02	
f23	SOCAIN	6,998E+02	5,692E+01	7,040E+02	5,984E+02	2,699E-03 (S+)
	(q) SOCAIN	6,527E+02	5,063E+01	6,445E+02	5,806E+02	
f24	SOCAIN	3,612E+02	4,429E+01	3,631E+02	2,745E+02	4,926E-01 (-)
	(q) SOCAIN	3,648E+02	3,885E+01	3,736E+02	2,859E+02	
f25	SOCAIN	4,015E+02	2,429E+01	4,119E+02	3,173E+02	1,281E-02 (S-)
	(q) SOCAIN	4,115E+02	3,074E+00	4,121E+02	3,974E+02	

Tabela A.29: Resultados dos erros $|f(x)-f(x^*)|$ referentes às execuções com $N_D = 30$. Comparação do SOCAIN com o (q)SOCAIN.

Função	Algoritmo	Média	Desvio padrão	Mediana	Mínimo	p-valor
f1	SOCAIN	1,311E+01	1,527E+00	1,309E+01	1,019E+01	6,647E-04 (S+)
	(q) SOCAIN	1,082E+01	2,761E+00	1,155E+01	3,973E+00	
f2	SOCAIN	4,836E+01	7,567E+00	4,798E+01	3,187E+01	8,191E-01 (-)
	(q) SOCAIN	4,890E+01	5,086E+00	4,900E+01	3,913E+01	
f3	SOCAIN	2,712E+06	4,828E+05	2,812E+06	1,619E+06	6,377E-01 (+)
	(q) SOCAIN	2,784E+06	6,424E+05	2,639E+06	1,796E+06	
f4	SOCAIN	3,178E+04	7,564E+03	3,264E+04	1,895E+04	1,229E-05 (S+)
	(q) SOCAIN	1,525E+04	4,837E+03	1,568E+04	4,199E+03	
f5	SOCAIN	3,688E+03	6,751E+02	3,726E+03	2,352E+03	4,593E-01 (-)
	(q) SOCAIN	3,804E+03	6,034E+02	3,883E+03	2,313E+03	
f6	SOCAIN	2,675E+03	6,284E+02	2,639E+03	1,572E+03	1,036E-01 (+)
	(q) SOCAIN	2,407E+03	5,862E+02	2,325E+03	1,031E+03	
f7	SOCAIN	1,184E+03	1,954E+02	1,198E+03	7,847E+02	1,229E-05 (S+)
	(q) SOCAIN	9,849E-01	2,605E-02	9,902E-01	9,123E-01	
f8	SOCAIN	2,096E+01	3,174E-02	2,096E+01	2,090E+01	8,612E-01
	(q) SOCAIN	2,095E+01	5,745E-02	2,096E+01	2,079E+01	
f9	SOCAIN	2,130E+02	1,401E+01	2,158E+02	1,829E+02	8,191E-01 (+)
	(q) SOCAIN	2,121E+02	1,289E+01	2,129E+02	1,791E+02	
f10	SOCAIN	2,331E+02	1,462E+01	2,383E+02	2,015E+02	1,155E-01 (-)
	(q) SOCAIN	2,404E+02	1,175E+01	2,411E+02	2,122E+02	
f11	SOCAIN	3,928E+01	1,165E+00	3,961E+01	3,638E+01	7,775E-01 (+)
	(q) SOCAIN	3,908E+01	1,696E+00	3,931E+01	3,304E+01	
f12	SOCAIN	3,342E+05	3,702E+04	3,305E+05	2,700E+05	1,500E-01 (+)
	(q) SOCAIN	3,168E+05	4,012E+04	3,160E+05	2,412E+05	
f13	SOCAIN	2,978E+01	1,939E+00	2,954E+01	2,446E+01	4,273E-01 (-)
	(q) SOCAIN	3,015E+01	2,805E+00	3,063E+01	2,347E+01	
f14	SOCAIN	1,343E+01	2,340E-01	1,340E+01	1,293E+01	9,263E-02 (+)
	(q) SOCAIN	1,328E+01	2,556E-01	1,330E+01	1,266E+01	
f15	SOCAIN	5,179E+02	2,213E+01	5,137E+02	5,025E+02	6,766E-01 (+)
	(q) SOCAIN	5,132E+02	4,643E+00	5,135E+02	5,026E+02	
f16	SOCAIN	2,734E+02	1,376E+01	2,751E+02	2,417E+02	5,098E-01 (-)
	(q) SOCAIN	2,762E+02	1,638E+01	2,783E+02	2,362E+02	
f17	SOCAIN	2,970E+02	1,359E+01	2,975E+02	2,757E+02	1,094E-01 (-)
	(q) SOCAIN	3,025E+02	1,610E+01	3,035E+02	2,594E+02	
f18	SOCAIN	9,333E+02	3,137E+00	9,340E+02	9,277E+02	3,260E-01 (-)
	(q) SOCAIN	9,337E+02	4,095E+00	9,345E+02	9,208E+02	
f19	SOCAIN	9,339E+02	2,886E+00	9,341E+02	9,276E+02	4,926E-01 (-)
	(q) SOCAIN	9,336E+02	2,906E+00	9,345E+02	9,274E+02	
f20	SOCAIN	9,339E+02	3,367E+00	9,332E+02	9,272E+02	7,366E-01 (+)
	(q) SOCAIN	9,329E+02	4,654E+00	9,330E+02	9,196E+02	
f21	SOCAIN	8,354E+02	2,187E+01	8,325E+02	7,805E+02	8,191E-01 (-)
	(q) SOCAIN	8,342E+02	2,909E+01	8,381E+02	7,751E+02	
f22	SOCAIN	9,730E+02	8,367E+00	9,728E+02	9,575E+02	7,164E-01 (-)

	(q) SOCAIN	9,733E+02	1,308E+01	9,749E+02	9,432E+02	
f23	SOCAIN	8,414E+02	2,394E+01	8,406E+02	7,887E+02	5,098E-01 (-)
	(q) SOCAIN	8,469E+02	2,317E+01	8,493E+02	8,073E+02	
f24	SOCAIN	7,333E+02	2,976E+01	7,373E+02	6,722E+02	9,036E-01 (-)
	(q) SOCAIN	7,324E+02	3,254E+01	7,405E+02	6,554E+02	
f25	SOCAIN	2,160E+02	1,159E+00	2,157E+02	2,147E+02	2,012E-01 (-)
	(q) SOCAIN	2,164E+02	1,305E+00	2,162E+02	2,140E+02	

Tabela A.30: Resultados dos erros $|f(x)-f(x^*)|$ referentes às execuções com $N_D = 50$. Comparação do SOCAIN com o (q)SOCAIN.

Função	Algoritmo	Média	Desvio padrão	Mediana	Mínimo	p-valor
f1	SOCAIN	3,918E+01	2,813E+00	3,948E+01	3,286E+01	3,223E-05 (S+)
	(q) SOCAIN	3,002E+01	5,149E+00	2,916E+01	2,264E+01	
f2	SOCAIN	4,067E+02	6,255E+01	4,144E+02	3,013E+02	6,531E-02 (+)
	(q) SOCAIN	3,710E+02	6,384E+01	3,876E+02	2,430E+02	
f3	SOCAIN	6,950E+06	8,080E+05	7,011E+06	5,456E+06	9,036E-01 (+)
	(q) SOCAIN	6,920E+06	1,102E+06	6,725E+06	4,626E+06	
f4	SOCAIN	6,943E+04	9,008E+03	6,679E+04	5,087E+04	1,229E-05 (S+)
	(q) SOCAIN	4,811E+04	7,462E+03	4,929E+04	3,271E+04	
f5	SOCAIN	9,026E+03	9,957E+02	9,276E+03	6,720E+03	9,464E-01 (+)
	(q) SOCAIN	8,939E+03	1,218E+03	9,028E+03	6,859E+03	
f6	SOCAIN	1,222E+04	3,308E+03	1,245E+04	7,566E+03	1,566E-04 (S+)
	(q) SOCAIN	8,353E+03	1,192E+03	8,469E+03	5,328E+03	
f7	SOCAIN	5,736E+02	1,057E+02	5,688E+02	3,882E+02	1,229E-05 (S+)
	(q) SOCAIN	1,049E+00	6,487E-03	1,050E+00	1,031E+00	
f8	SOCAIN	2,112E+01	3,912E-02	2,112E+01	2,100E+01	3,130E-01 (-)
	(q) SOCAIN	2,113E+01	3,186E-02	2,114E+01	2,104E+01	
f9	SOCAIN	4,422E+02	2,519E+01	4,448E+02	3,504E+02	8,824E-01
	(q) SOCAIN	4,424E+02	2,023E+01	4,448E+02	4,058E+02	
f10	SOCAIN	4,868E+02	1,545E+01	4,846E+02	4,509E+02	9,036E-01 (-)
	(q) SOCAIN	4,885E+02	1,658E+01	4,893E+02	4,552E+02	
f11	SOCAIN	7,261E+01	1,870E+00	7,303E+01	6,849E+01	5,449E-01 (+)
	(q) SOCAIN	7,272E+01	1,801E+00	7,297E+01	6,714E+01	
f12	SOCAIN	1,887E+06	1,488E+05	1,913E+06	1,620E+06	1,658E-01 (+)
	(q) SOCAIN	1,796E+06	1,604E+05	1,794E+06	1,465E+06	
f13	SOCAIN	6,943E+01	4,005E+00	6,934E+01	6,335E+01	9,797E-02 (-)
	(q) SOCAIN	7,188E+01	6,913E+00	7,069E+01	6,197E+01	
f14	SOCAIN	2,309E+01	2,319E-01	2,311E+01	2,235E+01	7,570E-01 (+)
	(q) SOCAIN	2,307E+01	2,730E-01	2,302E+01	2,231E+01	
f15	SOCAIN	5,114E+02	3,323E+00	5,116E+02	5,034E+02	3,281E-04 (S+)
	(q) SOCAIN	5,071E+02	4,015E+00	5,082E+02	4,997E+02	
f16	SOCAIN	3,459E+02	1,467E+01	3,428E+02	3,195E+02	6,148E-02 (-)
	(q) SOCAIN	3,530E+02	1,226E+01	3,536E+02	3,311E+02	
f17	SOCAIN	3,866E+02	1,381E+01	3,861E+02	3,449E+02	6,186E-01 (+)
	(q) SOCAIN	3,851E+02	1,253E+01	3,843E+02	3,674E+02	
f18	SOCAIN	9,822E+02	5,249E+00	9,816E+02	9,718E+02	2,209E-01 (-)

	(q) SOCAIN	9,836E+02	5,254E+00	9,831E+02	9,702E+02	
f19	SOCAIN	9,826E+02	6,851E+00	9,832E+02	9,673E+02	9,250E-01 (+)
	(q) SOCAIN	9,825E+02	6,969E+00	9,819E+02	9,696E+02	
f20	SOCAIN	9,849E+02	5,623E+00	9,844E+02	9,720E+02	2,643E-02 (S+)
	(q) SOCAIN	9,814E+02	6,706E+00	9,825E+02	9,632E+02	
f21	SOCAIN	9,895E+02	2,299E+01	9,916E+02	9,378E+02	3,674E-01 (+)
	(q) SOCAIN	9,827E+02	1,740E+01	9,856E+02	9,471E+02	
f22	SOCAIN	1,036E+03	1,186E+01	1,036E+03	1,015E+03	6,571E-01 (-)
	(q) SOCAIN	1,037E+03	1,566E+01	1,042E+03	9,945E+02	
f23	SOCAIN	9,993E+02	1,995E+01	1,002E+03	9,495E+02	3,533E-01 (+)
	(q) SOCAIN	9,965E+02	1,974E+01	9,966E+02	9,560E+02	
f24	SOCAIN	9,620E+02	3,047E+01	9,707E+02	8,773E+02	5,629E-01 (+)
	(q) SOCAIN	9,672E+02	2,225E+01	9,667E+02	9,220E+02	
f25	SOCAIN	2,472E+02	2,796E+00	2,469E+02	2,421E+02	8,824E-01 (+)
	(q) SOCAIN	2,471E+02	3,876E+00	2,460E+02	2,415E+02	

Tabela A.31: Resultados dos erros $|f(x)-f(x^*)|$ referentes às execuções com $N_D = 2$. Comparação do (q)opt-aiNet com o (q)SOCAIN.

Função	Algoritmo	Média	Desvio padrão	Mediana	Mínimo	p-valor
f1	(q) opt-aiNet	3,045E-05	2,799E-05	3,086E-05	2,440E-06	9,042E-05 (S-)
	(q) SOCAIN	1,523E-04	1,311E-04	1,319E-04	4,780E-06	
f2	(q) opt-aiNet	3,298E-05	3,969E-05	1,939E-05	1,200E-07	1,390E-05 (S-)
	(q) SOCAIN	2,693E-04	2,177E-04	2,051E-04	1,994E-05	
f3	(q) opt-aiNet	8,897E+01	9,697E+01	5,077E+01	9,320E-01	3,243E-02 (S+)
	(q) SOCAIN	4,938E+01	6,639E+01	3,562E+01	1,130E-01	
f4	(q) opt-aiNet	2,132E-04	2,488E-04	9,464E-05	1,867E-05	1,018E-02 (S-)
	(q) SOCAIN	4,712E-04	5,007E-04	4,274E-04	9,300E-06	
f5	(q) opt-aiNet	9,323E-02	4,762E-02	9,161E-02	2,601E-02	7,333E-04 (S-)
	(q) SOCAIN	2,502E-01	1,991E-01	1,995E-01	1,658E-02	
f6	(q) opt-aiNet	6,123E-03	1,091E-02	2,198E-03	1,635E-04	3,819E-01 (-)
	(q) SOCAIN	6,212E-03	5,871E-03	5,062E-03	1,082E-04	
f7	(q) opt-aiNet	2,085E+00	1,922E+00	1,662E+00	6,073E-02	2,159E-04 (S+)
	(q) SOCAIN	2,855E-01	3,346E-01	1,659E-01	1,352E-02	
f8	(q) opt-aiNet	1,102E+01	7,615E+00	1,005E+01	4,840E-02	3,260E-01 (+)
	(q) SOCAIN	8,949E+00	6,874E+00	7,287E+00	5,460E-01	
f9	(q) opt-aiNet	1,067E-02	1,399E-02	6,441E-03	9,099E-05	5,812E-01 (+)
	(q) SOCAIN	8,351E-03	1,100E-02	4,382E-03	2,398E-04	
f10	(q) opt-aiNet	2,251E-02	2,192E-02	1,632E-02	6,329E-04	8,401E-01 (+)
	(q) SOCAIN	2,591E-02	3,024E-02	1,309E-02	4,934E-04	
f11	(q) opt-aiNet	1,445E-01	5,329E-02	1,473E-01	4,064E-02	2,012E-01 (-)
	(q) SOCAIN	1,667E-01	6,245E-02	1,711E-01	5,434E-02	
f12	(q) opt-aiNet	4,527E-02	5,280E-02	2,309E-02	6,581E-05	1,885E-03 (S-)
	(q) SOCAIN	1,546E-01	1,764E-01	8,863E-02	3,315E-03	
f13	(q) opt-aiNet	6,360E-05	1,025E-04	2,381E-05	0,000E+00	2,758E-01 (-)
	(q) SOCAIN	1,740E-04	3,886E-04	3,391E-05	2,000E-08	
f14	(q) opt-aiNet	7,821E-02	7,802E-02	7,445E-02	1,943E-02	1,079E-03 (S+)

	(q) SOCAIN	2,268E-02	1,659E-02	1,943E-02	5,175E-04	
f15	(q) opt-aiNet	4,706E-01	6,200E-01	3,451E-01	5,794E-03	4,758E-01 (+)
	(q) SOCAIN	7,056E-01	1,160E+00	2,902E-01	3,030E-02	
f16	(q) opt-aiNet	6,143E-01	8,666E-01	3,525E-01	1,649E-02	7,800E-02 (-)
	(q) SOCAIN	1,061E+00	1,081E+00	6,033E-01	3,702E-02	
f17	(q) opt-aiNet	4,679E-01	7,498E-01	2,810E-01	1,653E-02	1,281E-02 (S-)
	(q) SOCAIN	1,517E+00	1,630E+00	7,023E-01	5,539E-02	
f18	(q) opt-aiNet	1,492E+01	1,078E+01	1,223E+01	2,725E+00	7,800E-02 (-)
	(q) SOCAIN	3,869E+01	4,714E+01	2,343E+01	3,708E+00	
f19	(q) opt-aiNet	1,740E+02	3,514E+01	2,002E+02	1,035E+02	2,641E-01 (+)
	(q) SOCAIN	1,826E+02	4,321E+01	1,937E+02	1,084E+02	
f20	(q) opt-aiNet	6,163E+01	7,866E+01	2,524E+01	2,562E+00	1,658E-01 (-)
	(q) SOCAIN	7,978E+01	7,066E+01	6,345E+01	1,500E+00	
f21	(q) opt-aiNet	8,507E+01	8,808E+01	2,872E+01	3,148E-01	5,449E-01 (-)
	(q) SOCAIN	1,013E+02	7,618E+01	1,077E+02	1,332E+00	
f22	(q) opt-aiNet	1,726E+02	5,523E+01	2,002E+02	2,203E+01	9,893E-01 (+)
	(q) SOCAIN	1,688E+02	5,694E+01	2,001E+02	3,367E+01	
f23	(q) opt-aiNet	6,666E+01	8,500E+01	2,640E+01	9,789E-01	6,746E-02 (-)
	(q) SOCAIN	1,218E+02	1,028E+02	9,157E+01	4,996E-01	
f24	(q) opt-aiNet	1,866E+02	2,381E+01	2,000E+02	1,067E+02	3,243E-02 (S-)
	(q) SOCAIN	1,962E+02	1,130E+01	2,001E+02	1,540E+02	
f25	(q) opt-aiNet	1,805E+02	2,200E+01	1,885E+02	1,411E+02	6,571E-01 (+)
	(q) SOCAIN	1,764E+02	2,602E+01	1,791E+02	1,121E+02	

Tabela A.32: Resultados dos erros $|f(x)-f(x^*)|$ referentes às execuções com $N_D = 10$. Comparação do (q)opt-aiNet com o (q)SOCAIN.

Função	Algoritmo	Média	Desvio padrão	Mediana	Mínimo	p-valor
f1	(q) opt-aiNet	9,369E-02	2,767E-02	9,678E-02	5,449E-02	1,229E-05 (S-)
	(q) SOCAIN	5,543E-01	1,773E-01	5,985E-01	1,970E-01	
f2	(q) opt-aiNet	1,659E-01	4,013E-02	1,684E-01	8,776E-02	1,229E-05 (S-)
	(q) SOCAIN	1,174E+00	2,891E-01	1,087E+00	7,157E-01	
f3	(q) opt-aiNet	7,139E+04	3,645E+04	6,837E+04	1,463E+04	1,155E-01 (-)
	(q) SOCAIN	8,856E+04	3,797E+04	8,173E+04	2,789E+04	
f4	(q) opt-aiNet	4,299E+03	2,446E+03	4,621E+03	1,015E+02	1,229E-05 (S+)
	(q) SOCAIN	6,800E+01	1,429E+02	1,026E+01	2,389E+00	
f5	(q) opt-aiNet	2,918E+01	5,612E+00	2,701E+01	2,193E+01	1,229E-05 (S-)
	(q) SOCAIN	6,165E+01	8,668E+00	6,041E+01	4,922E+01	
f6	(q) opt-aiNet	1,749E+01	3,387E+00	1,709E+01	9,539E+00	1,229E-05 (S-)
	(q) SOCAIN	7,979E+01	2,205E+01	7,626E+01	4,566E+01	
f7	(q) opt-aiNet	6,365E-01	3,550E-01	6,007E-01	1,888E-01	2,758E-01 (-)
	(q) SOCAIN	7,005E-01	1,182E-01	7,437E-01	4,930E-01	
f8	(q) opt-aiNet	2,034E+01	7,534E-02	2,034E+01	2,011E+01	3,955E-02 (S-)
	(q) SOCAIN	2,038E+01	8,641E-02	2,040E+01	2,017E+01	
f9	(q) opt-aiNet	2,745E+01	4,266E+00	2,784E+01	1,641E+01	2,528E-01 (+)
	(q) SOCAIN	2,852E+01	3,756E+00	2,774E+01	2,218E+01	
f10	(q) opt-aiNet	3,382E+01	5,169E+00	3,288E+01	2,347E+01	3,704E-02 (S-)

	(q) SOCAIN	3,591E+01	4,909E+00	3,622E+01	2,661E+01	
f11	(q) opt-aiNet	7,674E+00	6,714E-01	7,916E+00	6,219E+00	4,574E-05 (S+)
	(q) SOCAIN	6,523E+00	6,699E-01	6,545E+00	5,213E+00	
f12	(q) opt-aiNet	2,073E+03	5,146E+02	2,033E+03	1,170E+03	1,773E-05 (S-)
	(q) SOCAIN	5,504E+03	1,989E+03	5,600E+03	2,176E+03	
f13	(q) opt-aiNet	3,451E+00	5,065E-01	3,572E+00	1,851E+00	3,967E-01 (-)
	(q) SOCAIN	3,574E+00	6,230E-01	3,690E+00	2,336E+00	
f14	(q) opt-aiNet	3,843E+00	1,638E-01	3,867E+00	3,473E+00	8,905E-04 (S+)
	(q) SOCAIN	3,641E+00	2,761E-01	3,577E+00	3,027E+00	
f15	(q) opt-aiNet	3,026E+02	3,606E+01	3,001E+02	2,298E+02	7,775E-01 (-)
	(q) SOCAIN	3,069E+02	4,064E+01	3,033E+02	2,316E+02	
f16	(q) opt-aiNet	1,824E+02	1,086E+01	1,818E+02	1,648E+02	6,964E-01 (-)
	(q) SOCAIN	1,806E+02	1,593E+01	1,832E+02	1,506E+02	
f17	(q) opt-aiNet	1,953E+02	1,686E+01	1,954E+02	1,634E+02	1,500E-01 (-)
	(q) SOCAIN	2,045E+02	1,184E+01	2,069E+02	1,827E+02	
f18	(q) opt-aiNet	4,680E+02	2,990E+01	4,703E+02	4,145E+02	1,229E-05 (S-)
	(q) SOCAIN	5,835E+02	3,844E+01	5,756E+02	5,204E+02	
f19	(q) opt-aiNet	4,807E+02	2,981E+01	4,754E+02	4,095E+02	1,773E-05 (S-)
	(q) SOCAIN	5,728E+02	3,271E+01	5,782E+02	4,968E+02	
f20	(q) opt-aiNet	4,643E+02	2,821E+01	4,706E+02	4,078E+02	1,229E-05 (S-)
	(q) SOCAIN	5,841E+02	2,872E+01	5,855E+02	5,287E+02	
f21	(q) opt-aiNet	5,072E+02	6,303E+01	5,359E+02	3,392E+02	1,229E-05 (S-)
	(q) SOCAIN	6,689E+02	5,004E+01	6,725E+02	5,839E+02	
f22	(q) opt-aiNet	7,516E+02	9,385E+01	7,847E+02	4,994E+02	1,390E-05 (S-)
	(q) SOCAIN	8,055E+02	8,611E+00	8,055E+02	7,813E+02	
f23	(q) opt-aiNet	5,394E+02	4,625E+01	5,616E+02	4,252E+02	1,229E-05 (S-)
	(q) SOCAIN	6,527E+02	5,063E+01	6,445E+02	5,806E+02	
f24	(q) opt-aiNet	2,414E+02	9,012E+00	2,432E+02	2,267E+02	1,229E-05 (S-)
	(q) SOCAIN	3,648E+02	3,885E+01	3,736E+02	2,859E+02	
f25	(q) opt-aiNet	3,297E+02	6,897E+01	2,828E+02	2,559E+02	1,571E-05 (S-)
	(q) SOCAIN	4,115E+02	3,074E+00	4,121E+02	3,974E+02	

Tabela A.33: Resultados dos erros $|f(x)-f(x^*)|$ referentes às execuções com $N_D = 30$. Comparação do (q)opt-aiNet com o (q)SOCAIN.

Função	Algoritmo	Média	Desvio padrão	Mediana	Mínimo	p-valor
f1	(q) opt-aiNet	1,570E+00	1,752E-01	1,588E+00	1,191E+00	1,229E-05 (S-)
	(q) SOCAIN	1,082E+01	2,761E+00	1,155E+01	3,973E+00	
f2	(q) opt-aiNet	1,112E+01	1,555E+00	1,095E+01	9,022E+00	1,229E-05 (S-)
	(q) SOCAIN	4,890E+01	5,086E+00	4,900E+01	3,913E+01	
f3	(q) opt-aiNet	1,633E+06	3,599E+05	1,615E+06	1,015E+06	1,229E-05 (S-)
	(q) SOCAIN	2,784E+06	6,424E+05	2,639E+06	1,796E+06	
f4	(q) opt-aiNet	3,442E+04	6,932E+03	3,584E+04	2,041E+04	1,229E-05 (S+)
	(q) SOCAIN	1,525E+04	4,837E+03	1,568E+04	4,199E+03	
f5	(q) opt-aiNet	5,013E+03	5,565E+02	4,969E+03	4,125E+03	2,543E-05 (S+)
	(q) SOCAIN	3,804E+03	6,034E+02	3,883E+03	2,313E+03	
f6	(q) opt-aiNet	2,582E+02	7,428E+01	2,358E+02	1,822E+02	1,229E-05 (S-)

	(q) SOCAIN	2,407E+03	5,862E+02	2,325E+03	1,031E+03	
f7	(q) opt-aiNet	4,160E-01	4,387E-02	4,271E-01	3,444E-01	1,229E-05 (S-)
	(q) SOCAIN	9,849E-01	2,605E-02	9,902E-01	9,123E-01	
f8	(q) opt-aiNet	2,094E+01	5,064E-02	2,096E+01	2,082E+01	5,998E-01
	(q) SOCAIN	2,095E+01	5,745E-02	2,096E+01	2,079E+01	
f9	(q) opt-aiNet	2,076E+02	1,325E+01	2,064E+02	1,770E+02	2,879E-01 (-)
	(q) SOCAIN	2,121E+02	1,289E+01	2,129E+02	1,791E+02	
f10	(q) opt-aiNet	2,268E+02	1,645E+01	2,295E+02	1,840E+02	7,423E-03 (S-)
	(q) SOCAIN	2,404E+02	1,175E+01	2,411E+02	2,122E+02	
f11	(q) opt-aiNet	3,903E+01	1,809E+00	3,949E+01	3,296E+01	7,164E-01 (+)
	(q) SOCAIN	3,908E+01	1,696E+00	3,931E+01	3,304E+01	
f12	(q) opt-aiNet	1,487E+05	1,797E+04	1,422E+05	1,067E+05	1,229E-05 (S-)
	(q) SOCAIN	3,168E+05	4,012E+04	3,160E+05	2,412E+05	
f13	(q) opt-aiNet	2,297E+01	1,768E+00	2,294E+01	1,747E+01	1,229E-05 (S-)
	(q) SOCAIN	3,015E+01	2,805E+00	3,063E+01	2,347E+01	
f14	(q) opt-aiNet	1,348E+01	1,609E-01	1,349E+01	1,310E+01	2,699E-03 (S+)
	(q) SOCAIN	1,328E+01	2,556E-01	1,330E+01	1,266E+01	
f15	(q) opt-aiNet	4,825E+02	3,768E+00	4,834E+02	4,652E+02	1,229E-05 (S-)
	(q) SOCAIN	5,132E+02	4,643E+00	5,135E+02	5,026E+02	
f16	(q) opt-aiNet	2,696E+02	1,523E+01	2,676E+02	2,438E+02	7,800E-02 (-)
	(q) SOCAIN	2,762E+02	1,638E+01	2,783E+02	2,362E+02	
f17	(q) opt-aiNet	3,052E+02	1,511E+01	3,098E+02	2,651E+02	2,418E-01 (+)
	(q) SOCAIN	3,025E+02	1,610E+01	3,035E+02	2,594E+02	
f18	(q) opt-aiNet	8,832E+02	5,617E+01	9,173E+02	6,696E+02	1,229E-05 (S-)
	(q) SOCAIN	9,337E+02	4,095E+00	9,345E+02	9,208E+02	
f19	(q) opt-aiNet	8,935E+02	5,277E+01	9,177E+02	6,825E+02	1,229E-05 (S-)
	(q) SOCAIN	9,336E+02	2,906E+00	9,345E+02	9,274E+02	
f20	(q) opt-aiNet	8,917E+02	3,379E+01	9,169E+02	8,436E+02	1,229E-05 (S-)
	(q) SOCAIN	9,329E+02	4,654E+00	9,330E+02	9,196E+02	
f21	(q) opt-aiNet	6,032E+02	1,216E+01	6,045E+02	5,781E+02	1,229E-05 (S-)
	(q) SOCAIN	8,342E+02	2,909E+01	8,381E+02	7,751E+02	
f22	(q) opt-aiNet	9,619E+02	1,475E+01	9,632E+02	9,277E+02	8,905E-04 (S-)
	(q) SOCAIN	9,733E+02	1,308E+01	9,749E+02	9,432E+02	
f23	(q) opt-aiNet	6,259E+02	1,141E+01	6,253E+02	6,057E+02	1,229E-05 (S-)
	(q) SOCAIN	8,469E+02	2,317E+01	8,493E+02	8,073E+02	
f24	(q) opt-aiNet	3,802E+02	2,049E+01	3,812E+02	3,321E+02	1,229E-05 (S-)
	(q) SOCAIN	7,324E+02	3,254E+01	7,405E+02	6,554E+02	
f25	(q) opt-aiNet	2,131E+02	3,216E-01	2,131E+02	2,125E+02	1,229E-05 (S-)
	(q) SOCAIN	2,164E+02	1,305E+00	2,162E+02	2,140E+02	

Tabela A.34: Resultados dos erros $|f(x)-f(x^*)|$ referentes às execuções com $N_D = 50$. Comparação do (q)opt-aiNet com o (q)SOCAIN.

Função	Algoritmo	Média	Desvio padrão	Mediana	Mínimo	p-valor
f1	(q) opt-aiNet	5,112E+00	2,440E-01	5,162E+00	4,596E+00	1,229E-05 (S-)
	(q) SOCAIN	3,002E+01	5,149E+00	2,916E+01	2,264E+01	
f2	(q) opt-aiNet	1,041E+02	1,470E+01	1,042E+02	6,685E+01	1,229E-05 (S-)

	(q) SOCAIN	3,710E+02	6,384E+01	3,876E+02	2,430E+02	
f3	(q) opt-aiNet	3,712E+06	6,911E+05	3,578E+06	2,621E+06	1,229E-05 (S-)
	(q) SOCAIN	6,920E+06	1,102E+06	6,725E+06	4,626E+06	
f4	(q) opt-aiNet	8,904E+04	1,256E+04	9,013E+04	5,914E+04	1,229E-05 (S+)
	(q) SOCAIN	4,811E+04	7,462E+03	4,929E+04	3,271E+04	
f5	(q) opt-aiNet	1,262E+04	9,656E+02	1,278E+04	1,077E+04	1,229E-05 (S+)
	(q) SOCAIN	8,939E+03	1,218E+03	9,028E+03	6,859E+03	
f6	(q) opt-aiNet	8,173E+02	1,480E+02	8,222E+02	5,759E+02	1,229E-05 (S-)
	(q) SOCAIN	8,353E+03	1,192E+03	8,469E+03	5,328E+03	
f7	(q) opt-aiNet	6,475E-01	3,384E-02	6,433E-01	5,738E-01	1,229E-05 (S-)
	(q) SOCAIN	1,049E+00	6,487E-03	1,050E+00	1,031E+00	
f8	(q) opt-aiNet	2,114E+01	3,771E-02	2,114E+01	2,107E+01	8,191E-01
	(q) SOCAIN	2,113E+01	3,186E-02	2,114E+01	2,104E+01	
f9	(q) opt-aiNet	4,523E+02	2,036E+01	4,491E+02	4,101E+02	1,425E-01 (+)
	(q) SOCAIN	4,424E+02	2,023E+01	4,448E+02	4,058E+02	
f10	(q) opt-aiNet	4,818E+02	1,726E+01	4,816E+02	4,553E+02	3,395E-01 (-)
	(q) SOCAIN	4,885E+02	1,658E+01	4,893E+02	4,552E+02	
f11	(q) opt-aiNet	7,276E+01	1,550E+00	7,309E+01	6,933E+01	9,464E-01 (+)
	(q) SOCAIN	7,272E+01	1,801E+00	7,297E+01	6,714E+01	
f12	(q) opt-aiNet	8,480E+05	7,387E+04	8,484E+05	7,339E+05	1,229E-05 (S-)
	(q) SOCAIN	1,796E+06	1,604E+05	1,794E+06	1,465E+06	
f13	(q) opt-aiNet	5,216E+01	2,266E+00	5,242E+01	4,764E+01	1,229E-05 (S-)
	(q) SOCAIN	7,188E+01	6,913E+00	7,069E+01	6,197E+01	
f14	(q) opt-aiNet	2,314E+01	1,782E-01	2,317E+01	2,276E+01	2,879E-01 (+)
	(q) SOCAIN	2,307E+01	2,730E-01	2,302E+01	2,231E+01	
f15	(q) opt-aiNet	4,640E+02	2,055E+00	4,642E+02	4,587E+02	1,229E-05 (S-)
	(q) SOCAIN	5,071E+02	4,015E+00	5,082E+02	4,997E+02	
f16	(q) opt-aiNet	3,492E+02	1,959E+01	3,542E+02	2,956E+02	5,998E-01 (+)
	(q) SOCAIN	3,530E+02	1,226E+01	3,536E+02	3,311E+02	
f17	(q) opt-aiNet	3,888E+02	1,848E+01	3,875E+02	3,521E+02	3,819E-01 (+)
	(q) SOCAIN	3,851E+02	1,253E+01	3,843E+02	3,674E+02	
f18	(q) opt-aiNet	9,449E+02	2,047E+00	9,452E+02	9,411E+02	1,229E-05 (S-)
	(q) SOCAIN	9,836E+02	5,254E+00	9,831E+02	9,702E+02	
f19	(q) opt-aiNet	9,405E+02	1,590E+01	9,449E+02	8,799E+02	1,229E-05 (S-)
	(q) SOCAIN	9,825E+02	6,969E+00	9,819E+02	9,696E+02	
f20	(q) opt-aiNet	9,384E+02	1,953E+01	9,455E+02	8,701E+02	1,229E-05 (S-)
	(q) SOCAIN	9,814E+02	6,706E+00	9,825E+02	9,632E+02	
f21	(q) opt-aiNet	6,804E+02	1,320E+01	6,813E+02	6,496E+02	1,229E-05 (S-)
	(q) SOCAIN	9,827E+02	1,740E+01	9,856E+02	9,471E+02	
f22	(q) opt-aiNet	1,022E+03	9,979E+00	1,021E+03	1,004E+03	2,699E-03 (S-)
	(q) SOCAIN	1,037E+03	1,566E+01	1,042E+03	9,945E+02	
f23	(q) opt-aiNet	7,115E+02	1,811E+01	7,111E+02	6,684E+02	1,229E-05 (S-)
	(q) SOCAIN	9,965E+02	1,974E+01	9,966E+02	9,560E+02	
f24	(q) opt-aiNet	5,381E+02	1,851E+01	5,411E+02	4,938E+02	1,229E-05 (S-)
	(q) SOCAIN	9,672E+02	2,225E+01	9,667E+02	9,220E+02	
f25	(q) opt-aiNet	2,278E+02	1,400E+00	2,276E+02	2,256E+02	1,229E-05 (S-)

(q) SOCAIN	2,471E+02	3,876E+00	2,460E+02	2,415E+02
------------	-----------	-----------	-----------	-----------

A.3. EXPERIMENTOS INICIAIS DE DOCKING

Testes iniciais foram realizados no *docking* de proteínas, especificamente do complexo com código PDB 1CHO, utilizando o AG de [GARDINER *et al.* 2001] e o SOCOpt-aiNet. Estes experimentos foram executados por um número menor de avaliações, totalizando 6×10^4 .

Na Figura A.3 e na Figura A.4 são apresentados os complexos formados pelos melhores resultados obtidos das cinco execuções de cada um dos algoritmos: do AG e do SOCOpt-aiNet respectivamente. Verifica-se que ambos algoritmos foram capazes de evitar as penalidades provenientes de colisões, e conseguiram encontrar conformações que, mesmo que não sejam próximas a nativa (Figura 2), possuem uma complementaridade relativamente alta. Tal fato pode ser visto pelo *fitness* final obtido pelos algoritmos, de 4343 e 5609, pelo AG e pelo SOCOpt-aiNet respectivamente, como visto na Tabela A.35, enquanto a estrutura nativa apresenta *fitness* de 8729.

Tabela A.35: Melhores *fitness* obtidos de cinco execuções do AG e SOCOpt-aiNet.

	Média	Desvio padrão	Mediana	Máximo	p-valor
AG	3,840E+03	6,410E+02	4,220E+03	4,340E+03	4,375E-01
SOCOpt-aiNet	4,590E+03	6,520E+02	4,450E+03	5,610E+03

Figura A.3: Melhor resultado da implementação do AG de [GARDINER *et al.*, 2001] no experimento de validação.

Figura A.4: Melhor resultado do SOCOpt-aiNet no experimento de validação.

Figura A.5: Evolução do *fitness* do *docking* utilizando o SOCOpt-aiNet.

Figura A.6: Evolução do *fitness* do *docking* utilizando o AG de [GARDINER, et al., 2001].

Para verificar a adequação do tempo de otimização do algoritmo, foram gerados gráficos da evolução do *fitness* das melhores execuções do AG e do SOCOpt-aiNet (Figura A.5 e Figura A.6). Na Figura A.5, do SOCOpt-aiNet, verifica-se que o algoritmo ainda não convergiu ao final da execução. O mesmo ocorre no caso do AG (Figura A.6), que pode se verificar que o *fitness* médio ainda está em crescimento após cada

reinicialização. Portanto, em testes futuros, os algoritmos devem ser executados para um número maior de execuções.