Mônica Aparecida Cruvinel Valadão

Proposta de Um Algoritmo Evolutivo Assistido por um Modelo de Aproximação Kriging para Problemas de Otimização de Alto Custo Computacional

Belo Horizonte - MG

Dezembro de 2020

Mônica Aparecida Cruvinel Valadão

Proposta de Um Algoritmo Evolutivo Assistido por um Modelo de Aproximação Kriging para Problemas de Otimização de Alto Custo Computacional

Tese de Doutorado submetida à Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Escola de Engenharia da Universidade Federal de Minas Gerais, como requisito para obtenção do Título de Doutor em Engenharia Elétrica.

Universidade Federal de Minas Gerais – UFMG

Escola de Engenharia

Programa de Pós-Graduação em Engenharia Elétrica

Orientador: Lucas de Souza Batista

Belo Horizonte - MG

Dezembro de 2020

Valadão, Mônica Aparecida Cruvinel.
V136p Proposta de um algoritmo evolutivo assistido por um modelo de aproximação kriging para problemas de otimização de alto custo computacional [recurso eletrônico] / Mônica Aparecida Cruvinel Valadão. - 2020.
1 recurso online (116 f. : il., color.) : pdf.

Orientador: Lucas de Souza Batista.

Tese (doutorado) - Universidade Federal de Minas Gerais, Escola de Engenharia.

Apêndices: f. 109-116.
Bibliografia: f.103-108.

Exigências do sistema: Adobe Acrobat Reader.

1. Engenharia elétrica - Teses. 2. Algoritmos – Teses. 3. Otimização – Teses. I. Batista, Lucas de Souza. II. Universidade Federal de Minas Gerais. Escola de Engenharia. III. Título.

CDU: 621.3(043)

Ficha catalográfica: Biblioteca Prof. Mário Werneck, Escola de Engenharia da UFMG

UNIVERSIDADE FEDERAL DE MINAS GERAIS
Escola de Engenharia
Programa de Pós-Graduação em Engenharia Elétrica

"Proposta de Um Algoritmo Evolutivo Assistido Por Um Modelo de Aproximação Kriging Para Problemas de Otimização de Alto Custo Computacional"

Mônica Aparecida Cruvinel Valadão

Tese de Doutorado submetida à Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Escola de Engenharia da Universidade Federal de Minas Gerais, como requisito para obtenção do grau de Doutor em Engenharia Elétrica.

Aprovada em 14 de dezembro de 2020.

Por:

Prof. Dr. Lucas de Souza Batista
DEE (UFMG) - Orientador

Prof. Dr. Felipe Campelo França Pinto
DEE (UFMG)

Prof. Dr. Michel Bessani
(UFMG)

Prof. Dr. Rodrigo Tomás Nogueira Cardoso
DM (CEFET-MG)

Prof. Dr. Elizabeth Fialho Wanner
DECOM (CEFET-MG)

Aos meus pais, irmãs e sobrinhos.

Agradecimentos

Agradeço ao professor Lucas de Souza Batista, pela oportunidade de trabalhar sob sua orientação, por todo o apoio e profissionalismo com que me conduziu desde o início do doutorado. À Universidade Federal dos Vales do Jequitinhonha e Mucuri (UFVJM), em especial ao Instituto de Ciência e Tecnologia (ICT), pelo período de afastamento concedido para qualificação. Aos amigos que conheci no ORCS, dos quais sempre me lembrarei com muito carinho. E a Deus por essa benção.

“Here is that rainbow I’ve been praying for."
(Johnny Nash)

Resumo

Problemas de otimização que envolvem a avaliação de funções com alto custo computacional são frequentemente tratados na literatura através de estratégias baseadas em metamodelos. Um exemplo desse tipo de estratégia são os Algoritmos Evolutivos Assistidos por Metamodelo (SAEAs, *Surrogate Model Assisted Evolutionary Algorithms*), usualmente utilizados para a otimização de problemas computacionalmente caros de avaliar e que requerem muitas avaliações de função, como aqueles com elevado número de variáveis (atualmente, SAEAs têm sido aplicados a problemas com até 100 variáveis). Nesses métodos, o metamodelo é usado com a finalidade de conduzir o algoritmo evolutivo para regiões promissoras do espaço de busca e reduzir o número de avaliações exigidas por esses métodos. Entretanto, nesse tipo de estratégia, os custos associados à construção e atualização do metamodelo não podem ser proibitivos. Este trabalho propõe um SAEA autoadaptativo, denominado SAEAa, o qual acopla em um mesmo *framework* uma autoadaptação de parâmetro e um mecanismo detalhado que permite a escolha entre diferentes operadores de mutação. Especificamente incorpora-se operadores de mutação com estruturas distintas, o que permite agregar ao SAEAa uma manutenção de diversidade e maior pressão seletiva no processo evolutivo. Outra característica do SAEAa é o uso de um metamodelo *Ordinary Kriging* unidimensional, o que resulta em um menor custo computacional de construção/atualização de metamodelo em comparação com o uso de um modelo *Kriging* na sua forma usual. A descrição da estratégia proposta aborda também aspectos que influenciam diretamente na qualidade do metamodelo e diversidade da população, os quais não são tratados em SAEAs existentes na literatura. Aplicou-se o método proposto na otimização de um conjunto de funções analíticas e a análise de resultados mostrou que o método proposto apresentou um melhor desempenho, em termos de qualidade de solução e custo computacional, em comparação com estratégias recentes da literatura. Considerou-se também a aplicação do SAEAa na otimização de um projeto de antena para radar de subsolo (GPR, *Ground Penetrating Radar*), cujo o objetivo é encontrar uma configuração adequada de parâmetros que a antena deve apresentar para se adaptar bem a aplicação. Validou-se a solução retornada pelo SAEAa, a qual se mostrou adequada para aplicação em GPR. Além disso, foi possível evidenciar uma considerável redução de recurso computacional (tempo) a partir do emprego da abordagem proposta.

Palavras-chave: Modelos de Aproximação *Kriging*, Algoritmos Evolutivos Assistidos por Metamodelos, Problemas de Otimização de Alto Custo Computacional.

Abstract

Optimization problems that require the evaluation of functions with high computational cost are frequently solved through metamodel-based strategies. Examples of strategies based on metamodels are the Surrogate Model Assisted Evolutionary Algorithms (SAEAs) that are usually employed to solve optimization problems that are computationally expensive to be evaluated and require several function evaluations, such as the ones with a large number of variables. Currently, SAEAs have been applied in problems involving up to 100 variables. In such methods, the metamodel is used to guide the evolutionary algorithm towards promising regions of the search space and to reduce the number of function evaluations required. However, the cost associated with the update of the metamodel cannot be prohibitive. This work proposes a self-adaptive SAEA, named SAEAa, which couples in the same framework a parameter self-adaptation and a mechanism that allows the choice between different mutation operators. More precisely, it couples mutation operators with distinct features, adding in the SAEAa maintenance of population diversity and selective pressure in the evolutive process. Another feature of SAEAa is that it employs a unidimensional Ordinary Kriging metamodel. Thus, it reduces the computational cost of training this kind of metamodel compared to the standard form Kriging. The description of the proposed strategy also addresses aspects that directly influence the quality of metamodel and population diversity, which are not treated in SAEAs existing in the literature. The proposed approach was employed to solve a set of analytical functions of single-objective optimization problems. The results obtained suggest that the SAEAa presents a better performance, in terms of solution quality and computational cost, when compared to recent strategies. Besides, the proposed approach was employed on the solution of a ground-penetrating radar (GPR) antenna design. The solution returned by SAEAa was validated, and it showed to be suitable for application in GPR. Furthermore, it was possible to show a considerable reduction in computational resources (time) from using the proposed approach.

Keywords: Kriging Model, Surrogate Model Assisted Evolutionary Algorithms, Computationally Expensive Optimization Problem.

Lista de ilustrações

Figura 1 – Construção e refinamento de um metamodelo. 38
Figura 2 – Curvas de nível da função y em (a) e curvas de nível da função $\hat{y}$ em (b), com pontos da amostra identificados por + e novos pontos da amostra ∘ obtidos via maximização da função EI. 40
Figura 3 – Curvas de convergência dos métodos SAEAnoDE, SAEAnoUP e SAEAa e TASEA. 74
Figura 4 – Média da melhora em relação a solução inicial. 82
Figura 5 – Comparação da diferença média da melhora em relação a melhor solução inicial. Calculou-se os intervalos de confiaça de 95% usando bootstrap para diferenças pareadas considerando todos os problemas. 83
Figura 6 – Curvas de convergência da média de melhora em relação a melhor solução da população inicial em função do número de avaliações de função objetivo. Os plots são discretizados por função (vertical) e número de variáveis (horizontal). 84
Figura 7 – Média do tempo de execução (em segundos) por iteração associado a cada estratégia. 85
Figura 8 – Comparação da diferença média do tempo de execução (em segundos) por iteração. Calculou-se os intervalos de confiaça de 95% usando bootstrap para diferenças pareadas considerando todos os problemas. . 86
Figura 9 – Crossbar comparando a média dos indicadores de desempenho T_{sec} (eixo x) e $\Delta_{\%}^{(\star)}$ (eixo y) das diferentes estratégias. Marcadores quadrados representam a média e as linhas horizontal/vertical representam média ± erro padrão. 86
Figura 10 – Curvas de convergência da média de melhora em relação a melhor solução da população inicial em função do número de avaliações de função objetivo. Os plots são discretizados por função (vertical) e número de variáveis (horizontal). 88
Figura 11 – Crossbar comparando a média dos indicadores de desempenho T_{sec} (eixo x) e $\Delta_{\%}^{(\star)}$ (eixo y) das diferentes estratégias. Marcadores quadrados representam a média e as linhas horizontal/vertical representam média ± erro padrão. 89
Figura 12 – Curvas de convergência valor de normainf para cada teste. 91
Figura 13 – Validação da solução apresentada na Tabela 7. 92
Figura 13 – Validação da solução apresentada na Tabela 7. 93
Figura 14 – Curvas da solução apresentada em (AFRICANO et al., 2020). 94
Figura 14 – Curvas da solução apresentada em (AFRICANO et al., 2020). 95

Figura 15 – Curvas de ganho da solução obtida pelo SAEAa e da solução apresentada em (AFRICANO et al., 2020). 95
Figura 16 – Curvas de convergência da média de melhora em relação a melhor solução da população inicial em função do número de avaliações de função objetivo. Os plots são discretizados por função (vertical) e número de variáveis (horizontal). 116

Lista de tabelas

Tabela 1 – Problemas analíticos empregados nos experimentos computacionais. . . 79
Tabela 2 – Indicadores de desempenho adotados nos experimentos computacionais. 80
Tabela 3 – Parâmetros usados para cada problema (por número de variáveis). . . . 81
Tabela 4 – Parâmetros usados para cada problema (por número de variáveis). . . . 81
Tabela 5 – Média e desvio padrão do valor da melhor solução retornada pelas estratégias GEPEME, SA-COSO, SHPSO e SAEAa para alguns problemas e número de variáveis com $neval = 1000$. 87
Tabela 6 – Parâmetros da banda de frequência adotados na definição do problema. 90
Tabela 7 – Solução estimada para o problema da antena. 91
Tabela 8 – Média e desvio padrão do valor da melhor solução retornada por cada estratégia para todos os problemas e número de variáveis. 112
Tabela 9 – Média e desvio padrão do valor da melhor solução retornada por cada estratégia para todos os problemas e número de variáveis. 113
Tabela 10 – Média e desvio padrão do valor da melhor solução retornada por cada estratégia para todos os problemas e número de variáveis. 114
Tabela 11 – Média e desvio padrão do valor da melhor solução retornada por cada estratégia para todos os problemas e número de variáveis. 115

Sumário

1 INTRODUÇÃO 21
1.1 Apresentação 21
1.2 Objetivos da Tese 24
1.3 Contribuições da Tese 25
1.4 Estrutura do Trabalho 26

2 METAMODELOS KRIGING E RBF 29
2.1 Introdução 29
2.2 Conceitos Relacionados 29
2.2.1 Estimativa dos Parâmetros do Modelo de Regressão 30
2.3 Modelo Processo Gaussiano 31
2.4 Modelo Kriging 32
2.5 Funções de Base Radial 35
2.6 Construção de um Metamodelo 36
2.6.1 Expected Improvement (EI) 39
2.6.2 EGO 39
2.7 Conclusão 41

3 SURROGATE MODEL ASSISTED EVOLUTIONARY ALGORITHMS 43
3.1 Introdução 43
3.2 Algoritmos Evolutivos Assistidos por Metamodelo 43
3.2.1 SAEAs baseados em Controle Evolutivo 45
3.2.1.1 Soluções Controladas 45
3.2.1.2 Geração Controlada 46
3.2.2 SAEAs baseados apenas no Metamodelo 46
3.2.3 Construção de SAEAs baseados em Controle Evolutivo 47
3.2.3.1 Método de Redução Dimensional 58
3.3 Conclusão 59

4 PROPOSTA DE UM SAEA MONO-OBJETIVO 61
4.1 Introdução 61
4.2 SAEA autoadaptativo (SAEAa) 61
4.2.1 Operador de Mutação SL-PSO 62
4.2.2 Operadores de Mutação DE 63
4.2.3 Adaptação do parâmetro de cruzamento C 65
4.3 Método Proposto (SAEAa) 67

4.3.1 Considerações sobre a escolha e construção de metamodelo 70
4.3.1.1 Escolha do metamodelo 71
4.3.1.2 Escolha da amostra e atualização de DB 72
4.3.2 Efeito dos operadores de mutação e atualização de C 73
4.4 Conclusão **75**

5 PLANEJAMENTO EXPERIMENTAL E RESULTADOS 77
5.1 Implementação e configuração da máquina **77**
5.2 Problemas analíticos **78**
5.3 Indicadores de desempenho **80**
5.4 Planejamento Experimental **80**
5.5 Análise dos Resultados **82**
5.5.1 Qualidade da solução final 82
5.5.2 Capacidade de convergência 83
5.5.3 Custo computacional 84
5.5.4 Comparação com outros métodos 85
5.5.5 Considerações sobre limitações para realização de testes com $n > 200$ . . . 88
5.6 Aplicação em Problema Prático **89**
5.7 Conclusão **96**

6 CONCLUSÕES 99

REFERÊNCIAS 103

APÊNDICES 109

APÊNDICE A – TABELAS E FIGURAS 111

1 Introdução

1.1 Apresentação

O conceito de *otimização* pode ser entendido como o procedimento de minimizar (ou maximizar) uma ou mais funções mérito, usualmente chamadas de objetivos, definidas em um determinado domínio de projeto. De maneira geral, busca-se determinar a configuração ótima das variáveis de um projeto, tal que se obtenha um menor custo ou maior desempenho possível. Esses problemas, nos quais se deseja encontrar o menor custo da manufatura de um produto, ou a melhor eficiência de um processo, ou a maior confiabilidade de um produto etc., são abordados como Problemas de Otimização Mono-objetivo (SOOPs, *Single Objective Optimization Problems*), i.e., apenas uma função mérito é considerada. Esses problemas têm sido amplamente tratados na literatura via diferentes estratégias que não dependem de características específicas do problema de otimização como diferenciabilidade, convexidade e modalidade, e.g., *Evolutionary Algorithms* (EAs), *Simulated Annealing* (SA), *Ant Colony Optimization* (ACO), *Particle Swarm Optimization* (PSO), dentre outros (RAO, 2009; COELLO; LAMONT; van VELDHUIZEN, 2007; DRÉO et al., 2006).

Em muitos problemas de engenharia é comum que a avaliação das funções, que representam o problema, seja realizada por meio de modelos de simulação, como é o caso de muitos problemas que fazem o uso de métodos de elementos finitos. A otimização de um problema nessa situação torna-se um processo caro, devido ao alto custo computacional frequentemente associado a modelos de simulação, principalmente se são empregados métodos baseados em populações como estratégias de resolução. Problemas dessa natureza têm sido tratados na literatura com estratégias que fazem uso de metamodelo no processo de otimização, em que a predição de um dado não avaliado é realizada utilizando modelos, de aproximação do modelo de simulação, construídos a partir de dados observados no modelo de simulação que representa o problema (SCHONLAU, 1997; MENDES et al., 2013; SACKS et al., 1989). O *framework* EGO (*Efficient Global Optimization*) é um exemplo conhecido na literatura desse tipo de estratégia, proposto inicialmente por (JONES; SCHONLAU; WELCH, 1998) para a otimização global de funções caixa-preta de custo elevado. Outra forma de abordar problemas de otimização que envolvem alto custo computacional é resolver tais problemas via Algoritmos Evolutivos Assistidos por Metamodelo (SAEAs, *Surrogate Model Assisted Evolutionary Algorithms*) (EMMERICH; GIANNAKOGLOU; NAUJOKS, 2006; LIU; ZHANG; GIELEN, 2014; SUN et al., 2017). Diferente do EGO e suas variações, os SAEAs são estratégias que usam metamodelos para obter uma estimativa das avaliações da função original em momentos específicos do processo iterativo de um EA, e.g., com o intuito de estimar a avaliação de toda ou parte das soluções candidatas

em um metamodelo em cada iteração do processo de otimização. Essas duas metodologias de otimização são exploradas em capítulos seguintes deste trabalho.

A otimização de problemas práticos não é uma tarefa trivial e dependendo das especificidades envolvidas, a resolução do problema em questão torna-se um processo complexo. Características como alto número de variáveis e modelos de simulação com elevado custo computacional de avaliação, por exemplo, podem comprometer o orçamento computacional relacionado a avaliação de funções e, nesse sentido, o uso de metamodelos é apresentado na literatura como uma alternativa para contornar essa dificuldade. O uso de metamodelos no contexto de otimização é motivado, de forma geral, pela redução do número de avaliação na função original que pode ser alcançada ao aplicar estratégias de otimização baseadas em metamodelos. Neste trabalho investiga-se a integração de modelos de aproximação e computação evolutiva no contexto de SOOPs. Especificamente, considera-se aqui SAEAs aplicados em problemas de otimização mono-objetivo da forma (1.1) a seguir,

$$\begin{aligned} &\min \mathrm{y}(\mathbf{x}) \\ &\text{sujeito a: } \mathbf{x} \in \Omega \end{aligned} \tag{1.1}$$

onde $\Omega = \{\mathbf{x} \in \mathbb{R}^n | a_i \leq x_i \leq b_i, [a_i, b_i] \subset \mathbb{R}\, \forall i = 1, \ldots, n\}$ é a região factível das variáveis de decisão representadas no vetor $\mathbf{x} = [x_1, \ldots, x_n]$ e $\mathrm{y} : \Omega \longrightarrow \mathbb{R}$ é a função objetivo. A solução ótima desse SOOP é

$$\mathbf{x}^\star \in \Omega, \text{ tal que, } \mathrm{y}(\mathbf{x}^\star) \leq \mathrm{y}(\tilde{\mathbf{x}})\, \forall \tilde{\mathbf{x}} \in \Omega. \tag{1.2}$$

No contexto deste trabalho, assume-se que $\mathrm{y}(\cdot)$ em (1.1) é um modelo de simulação de alto custo computacional que representa um determinado problema. Nesse sentido, um metamodelo é um modelo de aproximação construído a partir de uma amostra de dados observados em $\mathrm{y}(\cdot)$, com o qual deseja-se fazer uma predição acurada de um dado ainda não observado. Adota-se neste trabalho as seguintes identificações: função $\mathrm{y}(\cdot)$ em (1.1) como a função mérito (função objetivo ou função original) que representa um modelo de simulação; função aproximada (ou função de aproximação) $\hat{\mathrm{y}}(\cdot)$ referindo-se ao metamodelo que representa um modelo de aproximação da função mérito; função *prescreening* (discutida a seguir) usada no processo de identificação de soluções que serão avaliadas em $\mathrm{y}(\cdot)$.

Em SAEAs frequentemente adota-se um critério para escolher quais soluções devem ser avaliadas em $\mathrm{y}(\cdot)$ em uma determinada iteração. Usualmente, avalia-se em $\mathrm{y}(\cdot)$ uma quantidade reduzida de soluções com melhor valor de função *prescreening*, a qual pode ser definida por exemplo como a função aproximada $\hat{\mathrm{y}}(\cdot)$. A função *prescreening* adotada está relacionada também com o tipo de metamodelo empregado no SAEA. Em SAEAs cujo metamodelo é do tipo *Kriging* usualmente empregam-se como função *prescreening* a função *Expected Improvement* (EI) ou *Lower Condition Bound* (LCB), as quais são

baseadas no erro associado a predição que tais metamodelos retornam (EMMERICH; GIANNAKOGLOU; NAUJOKS, 2006; LIU; ZHANG; GIELEN, 2014; YANG et al., 2019). De um modo geral, a distinção entre SAEAs e EAs é que no primeiro adota-se um critério para escolher quais soluções devem ser avaliadas em $y(\cdot)$ em uma determinada iteração, enquanto que nos EAs, em cada iteração, avalia-se na função mérito todas as soluções da população corrente. Esta etapa dos SAEAs tem como finalidade direcionar o uso do orçamento disponível de avaliação em $y(\cdot)$ somente para soluções que podem representar alguma melhora no processo evolutivo.

Em SAEAs é necessário considerar aspectos que influenciam a capacidade do metamodelo em conduzir o EA para regiões promissoras. Essa capacidade pode ser comprometida, por exemplo, ao se construir/atualizar o metamodelo com base em soluções que estão muito distantes da região que se deseja aproximar. Nesse sentido, nos SAEAs usualmente a amostra do metamodelo acompanha a evolução da população de soluções candidatas do EA. De modo geral, estruturas de SAEAs exigem a manutenção de um conjunto de armazenamento de soluções avaliadas na função mérito do qual seleciona-se a população corrente e amostras do metamodelo (ZHOU et al., 2007; LIU; ZHANG; GIELEN, 2014; SUN et al., 2017; CAI; LI, 2019). Embora esse conjunto influencie diretamente na qualidade do metamodelo e diversidade da população, muitos trabalhos na literatura não tratam em detalhes o processo de atualização desse conjunto. Observa-se também que, em geral, as condições que devem ser satisfeitas pela amostra empregada na construção de um metamodelo não são abordadas ou discutidas no contexto de SAEAs.

Em algoritmos do tipo SAEA, a convergência é influenciada também pelos operadores do EA usado como base. É um desafio manter a diversidade neste tipo de estratégia considerando que SAEAs são estruturados, de forma geral, na ideia de avaliar poucas soluções em $y(\cdot)$ por iteração e selecionar a população a partir de um conjunto de armazenamento. Uma alternativa observada na literatura para tratar esta questão consiste no emprego de diferentes operadores de variação de forma adaptativa (SUN et al., 2017; YANG et al., 2019). Este tipo de abordagem exige estabelecer critérios para definir qual operador empregar em cada iteração, considerando que o uso de um mesmo operador por muitas iterações sucessivas pode não contribuir para o processo evolutivo. Por exemplo, observa-se na literatura um SAEA estruturado a partir do emprego de diferentes variações de um mesmo operardor, o qual representa melhorias em relação a estratégias desenvolvidas com um único operador (YANG et al., 2019). Nesse sentido, é conveniente investigar também o efeito de incorporar em um mesmo SAEA operadores de variação de naturezas distintas.

O SAEA autoadaptativo proposto neste trabalho, identificado como SAEAa, emprega em sua estrutura diferentes operadores de variação visando evitar uma convergência prematura e melhorar a capacidade do algoritmo de escapar de uma região de estagnação.

Especificamente, incorpora-se no SAEAa uma combinação de operadores de mutação de naturezas distintas, resultando em uma manutenção de diversidade e maior pressão seletiva. O método considera também uma autoadaptação de parâmetros, associado à operadores de mutação, e define a escolha entre os diferentes operadores de acordo com informações sobre o aprendizado do algoritmo. Em relação ao metamodelo, o SAEAa adota um *Ordinary Kriging* (OK) unidimensional relatado na literatura no contexto de SAEAs, o que resulta em um menor custo de construção/atualização de metamodelo em comparação com o emprego de um modelo *Kriging* na sua forma usual. A descrição do método proposto aborda também aspectos que influenciam a capacidade do metamodelo em conduzir o EA para regiões promissoras, os quais observa-se que não são tratados nos SAEAs relatados neste trabalho. Especificamente, apresenta-se de forma detalhada o processo de atualização de um conjunto que armazena soluções avaliadas na função mérito. Além disso, aborda-se também as condições que devem ser satisfeitas pela amostra adotada para a construção de metamodelo. Usualmente em SAEAs define-se a amostra de metamodelo e a população corrente a partir de soluções armazenadas nesse conjunto, o que estabelece uma influência direta na qualidade do metamodelo e diversidade da população. Nesse sentido, a apresentação da abordagem proposta neste trabalho pode auxiliar também a compreender melhor o funcionamento de outros SAEAs existentes na literatura.

Para a validação do método proposto considera-se um conjunto de problemas analíticos envolvendo entre 10 e 200 variáveis. Em diferentes trabalhos relatados neste texto pode-se observar a validação de SAEAs em problemas analíticos com número de variáveis apenas entre 30 e 100. A análise dos resultados mostra que o SAEAa apresenta um melhor desempenho, em termos de qualidade de solução e custo computacional, em comparação com SAEAs recentes da literatura. O método proposto é aplicado também na otimização de um projeto de antena para radar de subsolo (GPR, *Ground Penetrating Radar*), no qual deseja-se obter os parâmetros apropriados que a antena deve apresentar para se adaptar bem à aplicação. A validação da solução retornada pelo SAEAa mostra que a antena é adequada para aplicações em GPR. Além disso, observa-se neste problema uma considerável redução de recurso computacional a partir do emprego do SAEA proposto neste trabalho.

1.2 Objetivos da Tese

Neste trabalho tem-se como principal objetivo propor um método SAEA, o qual emprega um metamodelo *Kriging* e explora o uso de diferentes operadores de mutação incorporado com um mecanismo de autoadaptação de parâmetro. Em mais detalhes, tem-se como objetivos específicos:

- Apresentar uma descrição detalhada de aspectos relacionados a construção/atualização de metamodelos que influenciam a busca por regiões promissoras no contexto de SAEAs que, em geral, não são tratados na literatura.

- Explorar o uso de um operador de mutação híbrido que combine operadores de mutação de estruturas distintas no contexto de SAEAs, com o intuito de evitar uma convergência prematura e melhorar a capacidade do algoritmo escapar de uma região de estagnação.

- Desenvolver e validar uma metodologia baseada em EAs e no modelo *Ordinary Kriging* (OK) que represente melhorias, em termos de custo computacional e qualidade de solução, em relação aos SAEAs relatados na literatura. Essa ferramenta deve se mostrar promissora para aplicação no contexto de problemas mono-objetivo, que possuem elevado custo associado a avaliação de funções.

- Investigar o desempenho do método proposto adaptado para adotar um metamodelo *Radial Basis Function* (RBF), considerando que este tipo de metamodelo também é frequentemente empregado no contexto de SAEAs.

- Apresentar uma análise qualitativa, relacionada a construção/atualização de metamodelo e capacidade de convergência, que justifique as melhorias apresentadas pela estratégia proposta.

1.3 Contribuições da Tese

A seguir são apresentadas as contribuições relacionadas ao desenvolvimento desta tese.

- Proposição de uma estratégia de otimização mono-objetivo aplicada ao contexto de problemas com elevado custo computacional. O método proposto, identificado como SAEAa, incorpora operadores de mutação de naturezas distintas visando agregar ao processo evolutivo uma manutenção de diversidade e maior pressão seletiva. A estrutura do SAEAa explora a informação sobre o aprendizado do algoritmo, a qual é empregada para direcionar a escolha entre os diferentes operadores de mutação adotados. O método considera também um mecanismo de autoadaptação de parâmetros, o qual é relacionado à operadores de mutação e se baseia em um histórico de melhora na função objetivo. O metamodelo empregado é um *Ordinary Kriging* (OK) unidimensional relatado na literatura no contexto de SAEAs, o que proporciona ao método proposto um menor custo de construção/atualização de metamodelo.

- Investigação e análise detalhada sobre o processo de atualização de um conjunto de armazenamento de soluções avaliadas na função mérito. Em diferentes SAEAs define-se a população corrente e amostra de metamodelo a partir das soluções deste conjunto, o que estabelece uma influência direta na qualidade do metamodelo e diversidade da população. Entretanto, nenhum dos trabalhos investigados da literatura abordaram em detalhes este processo.

- Investigação e análise detalhada das condições que devem ser satisfeitas pela amostra empregada na construção/atualização de metamodelo. Observa-se que, em geral, essa questão não é tratada no contexto de SAEAs e deve ser considerada no processo de atualização do conjunto de armazenamento já mencionado anteriormente.

As contribuições desta tese permitiram a publicação do artigo indicado a seguir, o qual apresenta um estudo investigativo sobre a influência de metamodelos do tipo *Kriging* e RBF no desempenho de um SAEA. Os resultados obtidos no artigo sugerem que o *Ordinary Kriging* (OK) apresenta desempenho superior em comparação com os outros metamodelos considerados.

- Valadão, M. A. C.; Batista, L.S.. A comparative study on surrogate models for SA-EAs. Optimization Letters, v. 14, p. 2595-2614, 2020. https://doi.org/10.1007/s11590-020-01575-2.

1.4 Estrutura do Trabalho

Capítulo 1 – **Introdução:** Este capítulo apresentou uma visão geral do contexto de estratégias de otimização que empregam metamodelos em sua estrutura. Especificou-se neste capítulo o interesse deste trabalho, que é investigar estratégias que integram modelos de aproximação e computação evolutiva no contexto de otimização mono-objetivo. Foram apresentados também conceitos considerados relevantes para o desenvolvimento deste trabalho.

Capítulo 2 – **Modelos Kriging e RBF:** Este capítulo discorre sobre a construção de um modelo de aproximação a partir de observações realizadas em um modelo de simulação, com o qual deseja-se fazer uma predição acurada de um dado ainda não observado. Especificamente o capítulo em questão aborda em detalhes os conceitos essenciais à compreensão e construção dos modelos *Kriging* e RBF; o estudo desse capítulo é fundamental para compreender as estratégias de otimização baseadas em metamodelos abordadas nos capítulos seguintes.

Capítulo 3 – **Surrogate Model Assisted Evolutionary Algorithms:** Este capítulo apresenta uma visão geral sobre Algoritmos Evolutivos Assistidos por Metamodelo

(SAEAs, *Surrogate Model Assisted Evolutionary Algorithms*), caracterizando tais métodos pelo tipo de metamodelo adotado e pela forma como o metamodelo é acoplado ao EA. O foco do capítulo é direcionado para a apresentação de SAEAs baseados em controle evolutivo, considerando especificamente a otimização de problemas mono-objetivo.

Capítulo 4 – **Proposta de um SAEA Mono-objetivo:** Neste capítulo propõe-se um SAEA autoadaptativo, identificado como SAEAa, no qual a escolha do operador de variação é baseada em informações sobre o aprendizado do algoritmo. O método proposto emprega um operador de mutação híbrido que envolve operadores de naturezas distintas, com intuito de agregar uma manutenção de diversidade e maior pressão seletiva ao processo evolutivo. Além disso, o método emprega também uma autoadaptação de parâmetros, associado à operadores de mutação. O metamodelo incorporado ao SAEAa é um OK unidimensional relatado na literatura no contexto de SAEA, o que proporciona ao método proposto um menor custo de construção/atualização de metamodelo em comparação com o emprego de um modelo *Kriging* na sua forma usual. O capítulo aborda também aspectos que influenciam a capacidade do metamodelo em conduzir o EA para regiões promissoras, o que pode auxiliar também a compreender melhor o funcionamento de outros SAEAs existentes na literatura.

Capítulo 5 – **Planejamento Experimental e Resultados:** Este capítulo apresenta o planejamento experimental adotado para a validação do SAEA proposto no Capítulo 4. O método em questão é aplicado em um conjunto de funções analíticas, envolvendo entre 10 e 200 variáveis. Os resultados obtidos mostram que o método proposto apresenta um melhor desempenho em relação aos SAEAs avaliadados. O SAEAa é aplicado também na resolução de um problema prático para obter uma configuração adequada de um projeto de antena para radar de subsolo (GPR, *Ground Penetrating Radar*). A validação da solução retornada pelo SAEAa mostra que a antena é adequada para aplicações em GPR. Além disso, observa-se neste problema uma considerável redução de recurso computacional (tempo) a partir do emprego do SAEA proposto neste trabalho.

Capítulo 6 – **Conclusões:** Este capítulo apresenta considerações finais sobre o desenvolvimento deste trabalho. Destaca também sugestões de estudos para trabalhos futuros.

2 Metamodelos Kriging e RBF

2.1 Introdução

O presente capítulo apresenta uma visão geral sobre a construção de metamodelos, considerando como interesse a construção de um metamodelo acurado com o menor número possível de observações no modelo de simulação. Diferentes variações de metamodelos são tratadas na literatura, sendo abordados aqui os modelos Processo Gaussiano, *Kriging* e Funções de Base Radial. O restante deste capítulo é organizado conforme a descrição a seguir. Na Seção 2.2 são apresentados conceitos relacionados a representação de dados por um modelo de regressão linear. A Seção 2.3 apresenta o modelo Processo Gaussiano. Os metamodelos *Kriging* e RBF são descritos, respectivamente, nas Seções 2.4 e 2.5. A Seção 2.6 descreve de forma geral o processo de construção de um metamodelo; essa seção apresenta também o método *Efficient Global Optimization* (EGO), que é base para o desenvolvimento de estratégias baseadas no conceito de adaptação de amostra; de acordo com esse conceito, novas soluções são inseridas na amostra a cada iteração. As considerações finais sobre o desenvolvimento do capítulo são apresentadas na Seção 2.7.

2.2 Conceitos Relacionados

É importante descrever primeiro as características dos dados da amostra, sobre os quais os modelos de aproximação tratados aqui são construídos. Nesse sentido, são apresentados inicialmente os conceitos comuns a esses metamodelos, como a representação dos dados por um modelo de regressão. As especificidades de cada modelo, como por exemplo a expressão para a predição, são tratadas em tópicos distintos.

Os metamodelos abordados neste trabalho assumem que, no modelo de simulação, as observações $\mathbf{y} = [\mathrm{y}(\mathbf{x}^1), \ldots, \mathrm{y}(\mathbf{x}^N)]^T$, $\mathrm{y} : \mathbb{X} \subset \mathbb{R}^n \longrightarrow \mathbb{R}$, de uma amostra com N pontos $\mathbf{X} = [\mathbf{x}^1, \ldots, \mathbf{x}^N]^T$ são determinísticas e, com exceção do metamodelo Funções de Base Radial, cada $\mathrm{y}(\mathbf{x}^i)$ é tratada como uma realização de uma variável aleatória (ou processo estocástico) sendo definida por um modelo de regressão (SACKS et al., 1989; SCHONLAU, 1997; JONES; SCHONLAU; WELCH, 1998)

$$\mathrm{y}(\mathbf{x}^i) = \sum_{k=0}^{d-1} \beta_k f_k(\mathbf{x}^i) + \epsilon(\mathbf{x}^i) = \mathbf{f}^T\boldsymbol{\beta} + \epsilon(\mathbf{x}^i), \quad i = 1, \ldots, N \tag{2.1}$$

onde $\mathbf{f} = [\mathrm{f}_0, \ldots, \mathrm{f}_{d-1}]^T$, $d \leq N$, são as funções de regressão e $\boldsymbol{\beta} = [\beta_0, \ldots, \beta_{d-1}]^T$ são os respectivos coeficientes a serem estimados; $\epsilon(\cdot)$ é um erro aleatório com média zero e

variância σ^2. É conveniente adotar a notação matricial de (2.1)

$$\mathbf{y} = \mathbf{F}\boldsymbol{\beta} + \boldsymbol{\epsilon} \tag{2.2}$$

e reescrever a predição dos dados da amostra da forma a seguir

$$\hat{\mathbf{y}} = \mathbf{F}\hat{\boldsymbol{\beta}} \tag{2.3}$$

onde $\hat{\boldsymbol{\beta}}$ é uma estimativa dos coeficientes de regressão em $\boldsymbol{\beta}$ e $\mathbf{F} = [\mathrm{f}_k(\mathbf{x}^i)]_{N\times d}$, $k = 0, \ldots, d-1$, $d \leq N$ e $i = 1, \ldots, N$ é a matriz com as avaliações das funções de regressão em cada dado da amostra. É conveniente descrever as expressões que fornecem as estimativas de $\boldsymbol{\beta}$ e de $\boldsymbol{\sigma}^2$ (caso $\boldsymbol{\sigma}^2$ seja desconhecida) em um tópico separado, uma vez que o cálculo de tais estimativas depende de premissas adicionais sobre $\epsilon(\cdot)$. Mais detalhes são apresentados no tópico a seguir.

2.2.1 Estimativa dos Parâmetros do Modelo de Regressão

Para descrever de forma geral os estimadores $\boldsymbol{\beta}$ e $\boldsymbol{\sigma}^2$, é exposto neste tópico de forma resumida uma adaptação da teoria apresentada em (MONTGOMERY; PECK; VINING, 2006) sobre estimadores de parâmetros em modelos de regressão nos casos de erros correlacionados e não correlacionados. Considere o modelo de regressão definido em (2.1), com $N > d$, $E(\boldsymbol{\epsilon}) = \mathbf{0}$, $Var(\boldsymbol{\epsilon}) = \sigma^2\mathbf{I}$ (σ^2 desconhecida) e que $(\mathbf{F}^T\mathbf{F})^{-1}$ exista, então pelo Método de Mínimos Quadrados, o Estimador de Mínimos Quadrados (LSE, *Least Square Estimator*) de $\boldsymbol{\beta}$ é

$$\hat{\boldsymbol{\beta}} = (\mathbf{F}^T\mathbf{F})^{-1}\mathbf{F}^T\mathbf{y} \tag{2.4}$$

e $\hat{\boldsymbol{\beta}}$ assim definido é um estimador não enviesado de $\boldsymbol{\beta}$. Se os erros são independentes e $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$, então $\hat{\boldsymbol{\beta}}$ é o Estimador de Máxima Verossimilhança (MLE, *Maximum Likelihood Estimator*), onde a função *likelihood* tem a forma

$$L(\mathbf{y}, \mathbf{F}, \boldsymbol{\beta}, \sigma^2) = \frac{1}{\sqrt{(2\pi\sigma^2)^N}} \exp\left(\frac{-1}{2\sigma^2}(\mathbf{y} - \mathbf{F}\boldsymbol{\beta})^T(\mathbf{y} - \mathbf{F}\boldsymbol{\beta})\right). \tag{2.5}$$

É usual aplicar o logaritmo natural em (2.5) e maximizar *log-likelihood*

$$\ln L(\mathbf{y}, \mathbf{F}, \boldsymbol{\beta}, \sigma^2) = -\frac{N}{2}\ln(2\pi) - N\ln(\sigma) - \frac{1}{2\sigma^2}(\mathbf{y} - \mathbf{F}\boldsymbol{\beta})^T(\mathbf{y} - \mathbf{F}\boldsymbol{\beta}) \tag{2.6}$$

e como $\hat{\boldsymbol{\beta}}$ em (2.4) é o MLE de $\boldsymbol{\beta}$ segue que o MLE de σ^2 é

$$\hat{\sigma}^2 = \frac{1}{N}(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}})^T(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}). \tag{2.7}$$

Por outro lado, se os erros são correlacionados, isto é, $Var(\boldsymbol{\epsilon}) = \sigma^2\mathbf{R}$, com $\mathbf{R} \neq \mathbf{I}$, então recorre-se ao Método de Mínimos Quadrados Generalizado, e o estimador não enviesado reescrito na forma

$$\hat{\boldsymbol{\beta}} = (\mathbf{F}^T\mathbf{R}^{-1}\mathbf{F})^{-1}\mathbf{F}^T\mathbf{R}^{-1}\mathbf{y} \tag{2.8}$$

é o Estimador de Mínimos Quadrados Generalizado (GLSE, *Generalized Least Square Estimator*) de $\boldsymbol{\beta}$, e

$$\hat{\sigma}^2 = \frac{1}{N}(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}})^T\mathbf{R}^{-1}(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}) \tag{2.9}$$

é o GLSE de σ^2, onde (2.5) e (2.6) assumem, respectivamente, as formas

$$L(\mathbf{y}, \mathbf{F}, \boldsymbol{\beta}, \sigma^2) = \frac{1}{\sqrt{(2\pi\sigma^2)^N|\mathbf{R}|}} \exp\left(\frac{-1}{2\sigma^2}(\mathbf{y} - \mathbf{F}\boldsymbol{\beta})^T\mathbf{R}^{-1}(\mathbf{y} - \mathbf{F}\boldsymbol{\beta})\right) \tag{2.10}$$

$$\ln L(\mathbf{y}, \mathbf{F}, \boldsymbol{\beta}, \sigma^2) = -\frac{N}{2}\ln(2\pi) - N\ln(\sigma) - \frac{1}{2}\ln(|\mathbf{R}|) - \frac{1}{2\sigma^2}(\mathbf{y} - \mathbf{F}\boldsymbol{\beta})^T\mathbf{R}^{-1}(\mathbf{y} - \mathbf{F}\boldsymbol{\beta}) \tag{2.11}$$

e dependendo da escolha da função de correlação, a função L passa a ser definida em função dos parâmetros envolvidos na construção da matriz $\mathbf{R}$. Esses detalhes são explorados na descrição do modelo Processo Gaussiano e do modelo *Kriging*.

2.3 Modelo Processo Gaussiano

A descrição do modelo Processo Gaussiano (GP, *Gaussian Process*) apresentada a seguir tem como referência base (LIU; ZHANG; GIELEN, 2014). Nesse modelo, dada uma amostra $\mathbf{x}^1, \ldots, \mathbf{x}^N$, cada observação $\mathrm{y}(\mathbf{x}^i)$ é considerada uma variável aleatória Gaussiana com distribuição $\mathcal{N}(\mu, \sigma^2)$ e a Equação (2.1) é reescrita como

$$\mathrm{y}(\mathbf{x}^i) = \mu + \epsilon(\mathbf{x}^i), \quad i = 1, \ldots, N \tag{2.12}$$

onde μ e σ^2 são constantes desconhecidas e independentes da amostra $\mathbf{x}^i$; $\epsilon(\cdot) \sim \mathcal{N}(0, \sigma^2)$, $\epsilon(\mathbf{x}^i)$ e $\epsilon(\mathbf{x}^j)$ correlacionados e a covariância definida da seguinte forma

$$Cov(\epsilon(\mathbf{x}^i), \epsilon(\mathbf{x}^j)) = \sigma^2\mathrm{R}(\mathbf{x}^i, \mathbf{x}^j), \tag{2.13}$$

em que o termo $\mathrm{R}(\mathbf{x}^i, \mathbf{x}^j)$ é a entrada (i, j) da matriz de correlação $\mathbf{R}$ e depende da distância entre $\mathbf{x}^i$ e $\mathbf{x}^j$. Frequentemente, $\mathrm{R}(\cdot)$ é considerada como

$$\mathrm{R}(\mathbf{x}, \tilde{\mathbf{x}}) = \prod_{r=1}^{n} \exp(-\theta_r|x_r - \tilde{x}_r|^{p_r}) \tag{2.14}$$

com $\theta_r > 0$ e $1 \leq p_r \leq 2$, $r = 1, \ldots, n$, no entanto outras formas de definir $\mathrm{R}(\cdot)$ podem ser encontradas em (MACKAY, 1998). Uma interpretação a respeito dos parâmetros $(\boldsymbol{\theta}, \mathbf{p})$ (usualmente chamados de hiperparâmetros) é descrita na Seção 2.4.

No contexto do modelo Processo Gaussiano, a função *log-likelihood* assume a forma

$$\ln L(\mathbf{y}, \mu, \sigma^2) = -\frac{N}{2}\ln(2\pi) - N\ln(\sigma) - \frac{1}{2}\ln(|\mathbf{R}|) - \frac{1}{2\sigma^2}(\mathbf{y} - \mu\mathbf{1})^T\mathbf{R}^{-1}(\mathbf{y} - \mu\mathbf{1})) \tag{2.15}$$

e para $(\boldsymbol{\theta}, \mathbf{p})$ fixos as estimativas ótimas (MLEs) de μ e de σ^2, adaptadas das Equações (2.8) e (2.9), são respectivamente

$$\hat{\mu} = \frac{\mathbf{1}^T \mathbf{R}^{-1} \mathbf{y}}{\mathbf{1}^T \mathbf{R}^{-1} \mathbf{1}} \tag{2.16}$$

$$\hat{\sigma}^2 = \frac{1}{N}(\mathbf{y} - \hat{\mu}\mathbf{1})^T \mathbf{R}^{-1}(\mathbf{y} - \hat{\mu}\mathbf{1}). \tag{2.17}$$

onde $\mathbf{1}_{j,1} = 1$, com $j = 1, \ldots, N$. Substituindo $\hat{\mu}$ e $\hat{\sigma}^2$ em (2.15), os parâmetros desconhecidos μ e σ^2 são eliminados e a função $\ln L$ passa a depender apenas de $(\boldsymbol{\theta}, \mathbf{p})$, que são obtidos ao resolver

$$\max_{(\boldsymbol{\theta},\mathbf{p}) \in \mathcal{H}} \ln L(\boldsymbol{\theta}, \mathbf{p}) \tag{2.18}$$

onde

$$\ln L(\boldsymbol{\theta}, \mathbf{p}) = -\frac{N}{2}\ln(2\pi) - N\ln(\hat{\sigma}) - \ln(|\mathbf{R}|) - \frac{1}{2} \tag{2.19}$$

e $\mathcal{H} = \{[\theta_1, \ldots, \theta_n, p_1, \cdots, p_n] \,|\, \theta_r > 0 \,\mathrm{e}\, 1 < p_r \leq 2, \forall r = 1, \ldots, n\}$; note que como a amostra é fixa, cada entrada da matriz $\mathbf{R}$ depende apenas de $(\boldsymbol{\theta}, \mathbf{p})$. Uma vez determinados $\boldsymbol{\theta}$ e $\mathbf{p}$, estes são substituídos em (2.16) e (2.17) para retornar $\hat{\mu}$ e $\hat{\sigma}^2$ e, assim, obter a predição em um ponto $\mathbf{x}$ não avaliado e o Erro Quadrado Médio (MSE, *Mean Squared Error*) associado em (2.20) e (2.21), respectivamente,

$$\hat{\mathrm{y}}(\mathbf{x}) = \hat{\mu} + \mathbf{r}^T(\mathbf{x})\mathbf{R}^{-1}(\mathbf{y} - \mathbf{1}\hat{\mu}) \tag{2.20}$$

$$\hat{s}^2(\mathbf{x}) = \hat{\sigma}^2 \left[1 - \mathbf{r}^T(\mathbf{x})\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}) + \frac{(1 - \mathbf{1}^T\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}))^2}{\mathbf{1}^T\mathbf{R}^{-1}\mathbf{1}}\right] \tag{2.21}$$

onde $\mathbf{y}$ é o vetor com avaliações da amostra e $\mathbf{r}^T(\mathbf{x}) = [\mathrm{R}(\mathbf{x}^1, \mathbf{x}), \ldots, \mathrm{R}(\mathbf{x}^N, \mathbf{x})]$ é o vetor de correlação, sendo que $\mathrm{R}(\mathbf{x}^i, \mathbf{x})$ representa a correlação entre $\epsilon(\cdot)$ em $\mathbf{x}^i$ da amostra e $\epsilon(\cdot)$ no dado não observado $\mathbf{x}$.

2.4 Modelo Kriging

Esta seção descreve os conceitos essenciais à compreensão do modelo *Kriging* e se baseia principalmente nas referências (SCHONLAU, 1997; SACKS et al., 1989; FORRESTER; SÓBESTER; KEANE, 2008; ZHAO; CHOI; LEE, 2011; HAN; ZHANG, 2012; ROUSTANT; GINSBOURGER; DEVILLE, 2012). Embora parte dos conceitos descritos neste tópico sejam comuns ao metamodelo Processo Gaussiano, o modelo *Kriging* possui características distintas que o diferencia desse metamodelo.

Considerando N pontos de uma amostra $\mathbf{X} = [\mathbf{x}^1, \ldots, \mathbf{x}^N]^T$ e respectivas observações $\mathbf{y} = [\mathrm{y}(\mathbf{x}^1), \ldots, \mathrm{y}(\mathbf{x}^N)]^T$, $\mathrm{y} : \mathbb{X} \subset \mathbb{R}^n \longrightarrow \mathbb{R}$, o modelo *Kriging* trata cada observação como

um modelo de regressão com as mesmas considerações acerca da Equação (2.1), reescrita a seguir,

$$\mathrm{y}(\mathbf{x}^i) = \mathbf{f}(\mathbf{x}^i)^T\boldsymbol{\beta} + Z(\mathbf{x}^i) \tag{2.22}$$

onde $Z(\mathbf{x}^i)$ no contexto deste trabalho é uma variável aleatória Gaussiana. É comum as funções de regressão serem identificadas como funções base e, no escopo deste trabalho, o conjunto com funções base candidatas (ZHAO; CHOI; LEE, 2011) é o conjunto $\mathcal{F} = \{\mathrm{f}_k(\mathbf{x}) | \mathrm{f}_k(\mathbf{x}) = x_1^{q_1} \dots x_n^{q_n}, k = 0, \dots, d-1, d \leq N-1, q_r \in [0, Q], \sum_{r=1}^{n} q_r \leq Q\}$, onde Q é o maior inteiro tal que

$$\frac{(n+Q)!}{(Q!n!)} \leq N - 1. \tag{2.23}$$

Em (2.22), $Z(\cdot)$ é assumido ter média zero, variância σ^2 e covariância

$$Cov(Z(\mathbf{x}^i), Z(\mathbf{x}^j)) = \sigma^2 \mathrm{R}(\mathbf{x}^i, \mathbf{x}^j) \tag{2.24}$$

sendo $\mathrm{R}(\cdot)$ uma função de correlação Gaussiana considerada aqui como

$$\mathrm{R}(\mathbf{x}, \tilde{\mathbf{x}}) = \prod_{r=1}^{n} \exp(-\theta_r |x_r - \tilde{x}_r|^{p_r}) \tag{2.25}$$

com $(\boldsymbol{\theta}, \mathbf{p}) \in \mathcal{H}$, $\mathcal{H} = \{[\theta_1, \dots, \theta_n, p_1, \dots, p_n] \,|\, \theta_r > 0 \,\mathrm{e}\, 0 < p_r \leq 2, \forall r = 1, \dots, n\}$. Os hiperparâmetros $(\boldsymbol{\theta}, \mathbf{p})$ são parâmetros que estão associados, respectivamente, com a influência de cada ponto da amostra em uma predição e com a suavidade da superfície de resposta do modelo de simulação. Um valor alto de p_r representa suavidade de $\mathrm{y}(\mathbf{x})$ em relação a variável x_r, enquanto valor muito pequeno de p_r significa que não exite correlação imediata entre $\mathbf{x}$ e $\tilde{\mathbf{x}}$. Em relação a $\boldsymbol{\theta}$, valores altos de θ_r significam que somente pontos da amostra em uma vizinhança de um ponto (sobre o qual deseja-se fazer uma predição) influenciam na predição; no caso de valores baixos de θ_r, todos os pontos da amostra influenciam nessa predição. Em (FORRESTER; SÓBESTER; KEANE, 2008) é apresentada ainda a possibilidade de usar o modelo *Kriging* para definir uma ordem de importância entre as variáveis, através desses hiperparâmetros, quando o problema em questão envolve muitas variáveis, no entanto neste trabalho essa abordagem não é considerada.

Da mesma forma que no metamodelo Processo Gaussiano, a função *log-likelihood* depende dos hiperparâmetros $(\boldsymbol{\theta}, \mathbf{p})$ e, no caso desse metamodelo,

$$\ln L(\mathbf{y}, \mathbf{F}, \boldsymbol{\beta}, \sigma^2) = -\frac{N}{2}\ln(2\pi) - N\ln(\sigma) - \frac{1}{2}\ln(|\mathbf{R}|) - \frac{1}{2\sigma^2}(\mathbf{y} - \mathbf{F}\boldsymbol{\beta})^T\mathbf{R}^{-1}(\mathbf{y} - \mathbf{F}\boldsymbol{\beta}) \tag{2.26}$$

e os MLEs de $\boldsymbol{\beta}$ e σ^2 para $(\boldsymbol{\theta}, \mathbf{p})$ fixos são dados, respectivamente, pelas Equações (2.8) e (2.9) reescritas a seguir

$$\hat{\boldsymbol{\beta}} = (\mathbf{F}^T\mathbf{R}^{-1}\mathbf{F})^{-1}\mathbf{F}^T\mathbf{R}^{-1}\mathbf{y} \tag{2.27}$$

$$\hat{\sigma}^2 = \frac{1}{N}(\mathbf{y} - \mathbf{F}\boldsymbol{\beta})^T\mathbf{R}^{-1}(\mathbf{y} - \mathbf{F}\boldsymbol{\beta}). \tag{2.28}$$

Substituindo $\hat{\boldsymbol{\beta}}$ e $\hat{\sigma}^2$ em (2.26), a determinação desses hiperparâmetros se resume ao problema

$$\max_{(\boldsymbol{\theta},\mathbf{p})\in\mathcal{H}} \ln L(\boldsymbol{\theta},\mathbf{p}) \tag{2.29}$$

onde

$$\ln L(\boldsymbol{\theta},\mathbf{p}) = -\frac{N}{2}\ln(2\pi) - N\ln(\hat{\sigma}) - \ln(|\mathbf{R}|) - \frac{1}{2} \tag{2.30}$$

e $\mathcal{H} = \{[\theta_1,\ldots,\theta_n,p_1,\ldots,p_n]\,|\,\theta_r \geq 0\,\text{e}\,0 < p_r \leq 2, \forall r = 1,\ldots,n\}$ (especificamente neste trabalho, $\theta_r \in [10^{-3},10^2]$ (FORRESTER; SÓBESTER; KEANE, 2008) e $p = 2$ (SCHONLAU, 1997)). A solução ótima desse problema em (2.25) fornece a matriz $\mathbf{R}$ e posteriormente os valores de $\hat{\boldsymbol{\beta}}$ e $\hat{\sigma}^2$ em (2.27) e (2.28). Assim, a predição da avaliação de um ponto $\mathbf{x}$ arbitrário que não pertence a amostra é definida por

$$\hat{\mathrm{y}}(\mathbf{x}) = \mathbf{f}^T\hat{\boldsymbol{\beta}} + \mathbf{r}^T\mathbf{R}^{-1}(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}) \tag{2.31}$$

e a Equação (2.21) que define o MSE, adaptada a esse contexto, é da forma

$$\hat{s}^2(\mathbf{x}) = \hat{\sigma}^2\left[1 - (\mathbf{f}^T(\mathbf{x}) + \mathbf{r}^T(\mathbf{x}))\begin{pmatrix}\mathbf{0} & \mathbf{F}^T\\ \mathbf{F} & \mathbf{R}\end{pmatrix}^{-1}\begin{pmatrix}\mathbf{f}(\mathbf{x})\\ \mathbf{r}(\mathbf{x})\end{pmatrix}\right] \tag{2.32}$$

onde $\mathbf{r}^T(\mathbf{x}) = [\mathrm{R}(\mathbf{x}^1,\mathbf{x}),\ldots,\mathrm{R}(\mathbf{x}^N,\mathbf{x})]$ é o vetor de correlação, sendo que $\mathrm{R}(\mathbf{x}^i,\mathbf{x})$ representa a correlação entre $\epsilon(\cdot)$ em $\mathbf{x}^i$ da amostra e $\epsilon(\cdot)$ no dado não observado $\mathbf{x}$. Além disso, o modelo *Kriging* assim definido é um modelo de interpolação; a Equação (2.31) avaliada em um ponto $\tilde{\mathbf{x}}$ que pertence a amostra coincide com $\mathrm{y}(\tilde{\mathbf{x}})$ uma vez que $\mathbf{r}^T(\tilde{\mathbf{x}})$ corresponde a uma linha da matriz $\mathbf{R}$.

É importante mencionar que em (2.27) e (2.28) não é calculada a inversa de $\mathbf{R}$, ao invés disso é aplicada a decomposição *Cholesky* e em seguida a fatoração SVD (*Singular Value Decomposition*) (FORD, 2014), uma vez que a inversão de matrizes pode ser inviável dependendo das dimensões da matriz em questão. O mesmo procedimento é realizado em (2.32) que envolve o cálculo de inversa de uma matriz particionada. Para uma revisão desse conceito é sugerido (HARVILLE, 2008; SANTNER; WILLIANS; NOTZ, 2003). Além disso, o mal condicionamento da matriz de correlação $\mathbf{R}$ é tratado em (FORRESTER; SÓBESTER; KEANE, 2008) com a correção $\mathbf{R} = \mathbf{R} + \epsilon\cdot\mathbf{I}$, onde ϵ é a precisão da máquina e $\mathbf{I}$ é a matriz identidade.

Diferentes variações do modelo *Kriging* são encontradas na literatura, sendo a forma geral descrita anteriormente conhecida como UK (*Universal Kriging*) e as diferentes formas se distinguem pela definição da média de cada observação $\mathrm{y}(\mathbf{x}^i)$ em (2.22) que, quando considerada conhecida e com as devidas adequações nas expressões envolvidas, fornece o *Simple Kriging* (SK); mais detalhes são encontrados em (ROUSTANT; GINSBOURGER; DEVILLE, 2012; GINSBOURGER; RICHE; CARRARO, 2010). Quando a média de $\mathrm{y}(\mathbf{x}^i)$ é desconhecida e, nesse caso, é da forma $\mu = \mathbf{f}^T(\mathbf{x}^i)\boldsymbol{\beta}$, as variações do UK são obtidas de

acordo com a escolha das funções base. O caso $k = 0$ ($\mathrm{f}_0(\cdot) = 1$ e $Q = 0$ em (2.23)) torna-se o *Ordinary Kriging* (OK) e as expressões envolvidas nesse caso são as descritas no caso do modelo Processo Gaussiano. O modelo OK com parâmetro fixo $p_r = 2$, $\forall i = 1, \ldots, n$ em (2.25) é amplamente conhecido como DACE (Design and Analysis of Computer Experiment) (JONES; SCHONLAU; WELCH, 1998; MA et al., 2015; HAO; SHAOPING; TOMOVIC, 2010; GINSBOURGER; RICHE; CARRARO, 2010). É importante destacar que em (LOPHANEV; NIELSEN; SØNDERGAARD, 2002) o DACE é definido de forma mais abrangente, permitindo que as funções base sejam polinômios de ordem $0, 1$ e 2. Para $Q = 1, 2, 3$ em (2.23) tem-se, respectivamente, as variações *First-Order Kriging* (First-Order UK), *Second-Order Kriging* (Second-Order UK) e *Third-Order Kriging* (Third-Order UK), com as devidas adequações no conjunto das funções base $\mathcal{F}$ de acordo com a escolha de Q. O modelo *Blind Kriging*, proposto em (JOSEPH; HUNG; SUDJIANTO, 2008), também assume $\mu = \mathbf{f}^T(\mathbf{x}^i)\boldsymbol{\alpha}$ desconhecida. A diferença em relação às variações do UK é que o BK assume que as funções base são desconhecidas, sendo escolhidas de um conjunto de funções candidatas previamente definido. Embora não seja explorado aqui, é proveitoso mencionar outra variação conhecida como modelo *Taylor Kriging*, que de modo geral se distingue por definir as funções base como $\mathrm{f}_k(\mathbf{x}) = (\mathbf{x} - \mathbf{x}_0)^k$ e coeficientes $\boldsymbol{\beta}$ envolvendo série de Taylor (XIA et al., 2015).

2.5 Funções de Base Radial

Para ilustrar a estrutura do metamodelo Funções de Base Radial (RBF, *Radial Basis Function*) é abordado aqui um caso particular desse método, sendo a teoria apresentada a seguir adaptada de (FORRESTER; SÓBESTER; KEANE, 2008; JIN; CHEN; SUDJIANTO, 2002). Considerando uma amostra com N pontos $\mathbf{X} = [\mathbf{x}^1, \cdots, \mathbf{x}^N]^T$ e respectivas observações $\mathbf{y} = [\mathrm{y}(\mathbf{x}^1), \cdots, \mathrm{y}(\mathbf{x}^N)]^T$, $\mathrm{y} : \mathbb{X} \subset \mathbb{R}^n \longrightarrow \mathbb{R}$ livre de ruído, a versão descrita aqui trata-se de um método de interpolação em que o valor $\mathrm{y}(\mathbf{x})$ de um $\mathbf{x}$ arbitrário ainda não avaliado é aproximado por

$$\hat{\mathrm{y}}(\mathbf{x}) = \omega_0 + \sum_{i=1}^{N} \omega_i \varphi_i(\mathbf{x}) \tag{2.33}$$

onde $\omega_0 = \dfrac{1}{N}\sum_{i=1}^{N} \mathrm{y}(\mathbf{x}^i)$, ω_i são os parâmetros desconhecidos e cada $\varphi_i(\cdot)$ (usualmente chamada de função de base radial) é uma função que depende da distância Euclidiana $||\cdot||$ entre $\mathbf{x}$ e um ponto $\mathbf{x}^i$ da amostra, conforme a expressão a seguir

$$\varphi_i(\mathbf{x}) = \phi(||\mathbf{x} - \mathbf{x}^i||). \tag{2.34}$$

A função ϕ definida aqui é uma função Gaussiana, $\phi(r) = e^{\frac{-r^2}{2\sigma^2}}$ com $\sigma = 1$. A condição $\mathrm{y}(\mathbf{x}^i) = \hat{\mathrm{y}}(\mathbf{x}^i)$, $i = 1, \ldots, N$, permite obter os coeficientes ω_i da seguinte forma

$$\mathbf{w} = \boldsymbol{\Psi}^{-1}(\mathbf{y} - \mathbf{1}\omega_0), \tag{2.35}$$

onde $\mathbf{w}_{i,1} = \omega_i$; $\mathbf{\Psi}_{i,j} = \varphi_j(\mathbf{x}^i)$ é a matriz de interpolação; $\mathbf{1}_{i,1} = 1$; $i, j = 1, \ldots, N$. Assim, (2.33) pode ser reescrita como

$$\hat{\mathrm{y}}(\mathbf{x}) = \omega_0 + \mathbf{m}\mathbf{\Psi}^{-1}(\mathbf{y} - \mathbf{1}\omega_0) \tag{2.36}$$

onde $\mathbf{m} = [\varphi_i(\mathbf{x})]_{1\times N}$. Em relação à função $\phi(\cdot)$, existem diferentes opções de escolha, e.g, $\phi(r) = r^\alpha$ com $1 \leq \alpha \leq 3$ (*power function*), $\phi(r) = r^2 \ln(r)$ (*thin plate spline*) e $\phi(r) = \sqrt{\sigma + r^2}$ (*multiquadric*). Algumas funções de base radial, diferente da Gaussiana, exigem a substituição de ω_0 em (2.33) por um termo polinomial e, assim, garantem que a matriz de interpolação associada possua inversa. Essa correção em (2.33) resulta em parâmetros adicionais ao modelo. Um outro parâmetro que pode ser acrescentado ao modelo é o termo σ em (2.34). Nesse caso, estima-se o valor desse parâmetro a partir da amostra ao invés de defini-lo como uma constante. Uma generalização do método RBF pode ser encontrada em (JIN; CHEN; SUDJIANTO, 2002; ROCHA, 2009). O modelo RBF, ao contrário do modelo *Kriging*, não fornece uma medida para o erro associado a predição em (2.33).

2.6 Construção de um Metamodelo

Na seção corrente são apresentadas inicialmente as ideias principais para a construção dos metamodelos RBF, *Ordinary Kriging* e *First-, Second-, Third-Order Kriging*. Esses modelos, ilustrados nos Algoritmos 2.1, 2.2 e 2.3, consideram uma amostra $\mathbf{X}$ de tamanho N gerada usando o Hipercubo Latino e respectivas avaliações em $\mathrm{y}(\cdot)$. O Algoritmo 2.1 ilustra um modelo de interpolação RBF com parâmetro $\sigma = 1$. O Algoritmo 2.2 refere-se

Algoritmo 2.1: Funções de Base Radial (RBF)

Entrada: $\Omega, \mathrm{y}(\cdot), N$
1 $\sigma \leftarrow 1$;
2 Gera amostra inicial $\mathbf{X} = [\mathbf{x}^1, \cdots, \mathbf{x}^N]^T$ em Ω;
3 Avalia $\mathbf{X}$: $\mathbf{y} = [\mathrm{y}(\mathbf{x}^1), \cdots, \mathrm{y}(\mathbf{x}^N)]^T$;
4 $\omega_0 \leftarrow \dfrac{1}{N}\displaystyle\sum_{i=1}^{N} \mathrm{y}(\mathbf{x}^i)$;
5 $\mathbf{\Psi} \leftarrow$ matriz de interpolação (2.34);
6 $\mathbf{w} \leftarrow$ vetor com os coeficientes (2.35);
7 $\hat{\mathrm{y}}(\cdot) \leftarrow$ função de predição (2.36);
Saída: $\hat{\mathrm{y}}(\cdot)$, $\mathbf{X}$, $\mathbf{y}$

ao metamodelo OK; o passo seguinte à geração da amostra define a função *log-likelihood* em função dos hiperparâmetros $(\tilde{\boldsymbol{\theta}}, \tilde{\mathbf{p}})$ conforme (2.19); essa etapa envolve o uso dos MLEs da média μ e variância do processo σ^2 em (2.16) e (2.17), uma vez que cada observação é considerada uma variável aleatória em (2.12). A maximização de $\ln L(\tilde{\boldsymbol{\theta}}, \tilde{\mathbf{p}})$ permite então calcular $\hat{\mu}, \hat{\sigma}^2$, a matriz de covariância $\mathbf{R}$ em (2.13) e definir também a função de predição $\hat{\mathrm{y}}(\cdot)$ e a expressão $s^2(\cdot)$ para o erro associado a predição. No Algoritmo 2.3 é adotado o

Algoritmo 2.2: Ordinary Kriging (OK)

Entrada: $\Omega, \mathrm{y}, N, \mathcal{H}$
1 Gera amostra inicial $\mathbf{X} = [\mathbf{x}^1, \cdots, \mathbf{x}^N]^T$ em Ω;
2 Avalia $\mathbf{X}$: $\mathbf{y} = [\mathrm{y}(\mathbf{x}^1), \cdots, \mathrm{y}(\mathbf{x}^N)]^T$;
3 Define $\ln L(\tilde{\boldsymbol{\theta}}, \tilde{\mathbf{p}})$, $(\tilde{\boldsymbol{\theta}}, \tilde{\mathbf{p}}) \in \mathcal{H}$ (2.19);
4 $(\boldsymbol{\theta}, \mathbf{p}) \leftarrow arg \max_{(\tilde{\boldsymbol{\theta}}, \tilde{\mathbf{p}}) \in \mathcal{H}} \ln L(\tilde{\boldsymbol{\theta}}, \tilde{\mathbf{p}})$;
5 $\mathbf{R} \leftarrow$ matriz de correlação definida em (2.13);
6 $\hat{\mu} \leftarrow$ MLE de μ (2.16);
7 $\hat{\sigma}^2 \leftarrow$ MLE de σ^2 (2.17);
8 $\hat{\mathrm{y}}(\cdot) \leftarrow$ função de predição (2.20);
9 $\hat{s}^2(\cdot) \leftarrow$ função do MSE (2.21);
Saída: $\hat{\mathrm{y}}(\cdot), \hat{\sigma}^2, \hat{s}^2(\cdot), \mathbf{X}, \mathbf{y}$

Algoritmo 2.3: Q-Order Kriging (First-, Second-, Third-Order Kriging)

Entrada: $\Omega, \mathrm{y}, N, \mathcal{H}, Q$
1 Gera amostra inicial $\mathbf{X} = [\mathbf{x}^1, \cdots, \mathbf{x}^N]^T$ em Ω;
2 Avalia $\mathbf{X}$: $\mathbf{y} = [\mathrm{y}(\mathbf{x}^1), \cdots, \mathrm{y}(\mathbf{x}^N)]^T$;
3 $n \leftarrow$ número de variáveis;
4 $d \leftarrow$ número de funções base possíveis dados Q e n;
5 $\mathbf{f} \leftarrow$ vetor com as d funções base possíveis;
6 $\mathbf{F} \leftarrow [\mathbf{f}(\mathbf{x}^i)]_{N \times d}$;
7 Define $\ln L(\tilde{\boldsymbol{\theta}}, \tilde{\mathbf{p}})$, $(\tilde{\boldsymbol{\theta}}, \tilde{\mathbf{p}}) \in \mathcal{H}$ (2.30);
8 $(\boldsymbol{\theta}, \mathbf{p}) \leftarrow arg \max_{(\tilde{\boldsymbol{\theta}}, \tilde{\mathbf{p}}) \in \mathcal{H}} \ln L(\tilde{\boldsymbol{\theta}}, \tilde{\mathbf{p}})$;
9 $\mathbf{R} \leftarrow$ matriz de correlação definida (2.25);
10 $\hat{\boldsymbol{\beta}} \leftarrow$ MLE de $\boldsymbol{\beta}$ (2.27);
11 $\hat{\sigma}^2 \leftarrow$ MLE de σ^2 (2.28);
12 $\hat{\mathrm{y}}(\cdot) \leftarrow$ função de predição (2.31);
13 $\hat{s}^2(\cdot) \leftarrow$ função do MSE (2.32);
Saída: $\hat{\mathrm{y}}(\cdot), \hat{\sigma}^2, \hat{s}^2(\cdot), \mathbf{X}, \mathbf{y}$

termo *Q-Order Kriging*, com $Q = 1, 2, 3$ para representar, respectivamente, *First-, Second-, Third-Order Kriging*. As diferenças em relação ao Algoritmo 2.2 são apenas nos passos 3-6; os demais passos são os mesmos do algoritmo anterior com as devidas adequações apresentadas na Seção 2.4. Nesse algoritmo, Q representa o maior grau permitido para cada função base do conjunto $\mathcal{F}$ e é necessário que o tamanho da amostra N satisfaça $\frac{(n+Q)!}{(Q!n!)} \leq N - 1$, onde n é o número de variáveis e $d = \frac{(n+Q)!}{(Q!n!)}$ é a cardinalidade de $\mathcal{F}$. No passo 5 do Algoritmo 2.3, $\mathbf{f}$ é um vetor com elementos de $\mathcal{F}$ e no passo 6, $\mathbf{F}$ é a matriz que contém a avaliação da amostra em cada função base.

Um ponto crucial para a eficiência de um metamodelo é a sua acurácia de predição, principalmente se este é usado em alguma etapa do processo de otimização de um dado problema. De modo geral, a construção e refinamento de um metamodelo é estruturada nas seguintes etapas principais: (i) a partir de uma amostra de dados observados gera-se um metamodelo inicial; (ii) avalia-se a acurácia do metamodelo e caso esta não satisfaça um critério de acurácia desejado, então insere-se um novo ponto na amostra e atualiza-se o metamodelo. A Figura 1 ilustra essa ideia e nesse contexto é pertinente questionar como identificar se o metamodelo alcançou o refinamento desejado e, se necessário, como encontrar um novo ponto a ser inserido na amostra. A métrica escolhida para avaliar a

acurácia e a estratégia para mapear um novo ponto a ser inserido na amostra depende do tipo de metamodelo e do propósito associado.

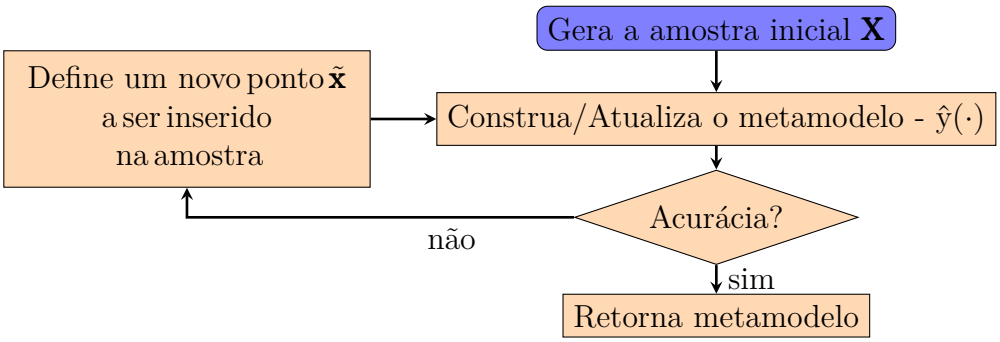


Figura 1 – Construção e refinamento de um metamodelo.

A forma como a amostra é atualizada, também denotada como *infill criteria*, é representada por meio de funções específicas que são definidas de modo a expressar as seguintes características: exploração global (*exploration*), exploração local (*exploitation*) ou exploração equilibrada entre *exploration* e *exploitation*. Diferentes *infill criteria* podem ser encontrados em (FORRESTER; SÓBESTER; KEANE, 2008), que apresenta de forma didática uma extensa revisão e orientações de como construir um metamodelo, incluindo as formas de realizar a atualização da amostra. A função *Expected Improvement* (EI) é um tipo de *infill criteria*, definido somente para modelos *Kriging*, muito explorado na literatura. Essa função é baseada no conceito de melhora esperada, conforme detalhado na Seção 2.6.1.

Em relação ao processo de construção e refinamento de metamodelo, ilustrado na Figura 1, pode-se considerar como critério de parada um número máximo de pontos a serem inseridos sequencialmente na amostra. Entretanto, é interessante atualizar a amostra com uma nova solução se esta representar uma melhora para o metamodelo. Uma forma de considerar essa questão é adotar como critério de parada o não atendimento a qualquer uma das desigualdades em (2.37) que estabelecem, respectivamente, uma tolerância $tolUp$ para o erro e uma tolerância $tolEI$ para o valor de função EI,

$$\frac{|\mathrm{y}(\tilde{\mathbf{x}}) - \hat{\mathrm{y}}(\tilde{\mathbf{x}})|}{maxerro} \leq tolUp \quad \text{ou} \quad \mathrm{EI}(\tilde{\mathbf{x}}) \leq tolEI \tag{2.37}$$

onde $maxerro$ é o maior valor $|\mathrm{y}(\cdot) - \hat{\mathrm{y}}(\cdot)|$ observado entre os pontos atualizados na amostra; $\tilde{\mathbf{x}}$ é a solução avaliada em $\mathrm{y}(\cdot)$ a cada iteração; $tolUp$ e $tolEI$ são estabelecidos com valores suficientemente pequenos, e.g, 10^{-2} e 10^{-5}, respectivamente. Esses critérios em (2.37) são fundamentados nas discussões relacionadas a diferentes formas de mapear um novo ponto para amostra e convergência do método EGO abordadas, respectivamente, em (FORRESTER; SÓBESTER; KEANE, 2008) e (JONES; SCHONLAU; WELCH, 1998). Detalhes do método *Efficient Global Optimization* (EGO) são apresentados na Seção 2.6.2

2.6.1 Expected Improvement (EI)

O critério *Expected Improvement* (EI) é baseado na ideia de que qualquer avaliação futura $\mathrm{y}(\cdot)$ em um novo ponto $\mathbf{x}$ constitui uma potencial melhora em relação ao mínimo y_{min} corrente (SCHONLAU, 1997). Considerando as definições de $\hat{\mathrm{y}}(\mathbf{x})$ e $\hat{s}^2(\mathbf{x})$ na Seção 2.4 e adaptando de forma resumida o exposto em (SCHONLAU, 1997; FORRESTER; SÓBESTER; KEANE, 2008), $\mathrm{y}(\mathbf{x})$ tem distribuição $\mathcal{N}(\hat{\mathrm{y}}(\mathbf{x}), \hat{s}^2(\mathbf{x}))$ e a função de melhora é definida por

$$\mathrm{I}(\mathbf{x}) = \begin{cases} (\mathrm{y}_{min} - \mathrm{y}(\mathbf{x})), & \text{se } \mathrm{y}(\mathbf{x}) < \mathrm{y}_{min} \\ 0, & \text{caso contrário} \end{cases} \tag{2.38}$$

O valor esperado de $\mathrm{I}(\mathbf{x})$ corresponde a função *Expected Improvement*

$$\mathrm{EI}(\mathbf{x}) = \begin{cases} (\mathrm{y}_{min} - \hat{\mathrm{y}}(\mathbf{x}))\Phi\left(\frac{(\mathrm{y}_{min} - \hat{\mathrm{y}}(\mathbf{x}))}{\hat{s}(\mathbf{x})}\right) + \hat{s}(\mathbf{x})\phi\left(\frac{(\mathrm{y}_{min} - \hat{\mathrm{y}}(\mathbf{x}))}{\hat{s}(\mathbf{x})}\right), & \text{se } \hat{s}(\mathbf{x}) > 0 \\ 0, & \text{caso contrário} \end{cases} \tag{2.39}$$

onde $\Phi(\cdot)$ e $\phi(\cdot)$ são, respectivamente, a função cumulativa e a função densidade de probabilidade da distribuição normal padrão. Em pontos da amostra com a qual o metamodelo associado é construído, a função EI é nula. Por outro lado, EI assume valores altos quando $\hat{\mathrm{y}}(\mathbf{x}) \leq \mathrm{y}_{min}$ ou em situações em que há muita incerteza em relação a predição; esse último caso pode ocorrer em pontos distantes da amostra. No caso $\hat{s}(\mathbf{x}) > 0$, $\mathrm{EI}(\mathbf{x})$ é avaliada pela expressão

$$\begin{aligned} \mathrm{EI}(\mathbf{x}) = (\mathrm{y}_{min} - \hat{\mathrm{y}}(\mathbf{x}))\left[\frac{1}{2} + \frac{1}{2}\mathrm{erf}\left(\frac{\mathrm{y}_{min} - \hat{\mathrm{y}}(\mathbf{x})}{\hat{s}(\mathbf{x})\sqrt{2}}\right)\right] \\ + \hat{s}(\mathbf{x})\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{(\mathrm{y}_{min} - \hat{\mathrm{y}}(\mathbf{x}))^2}{2\hat{s}^2(\mathbf{x})}\right), \end{aligned} \tag{2.40}$$

onde erf, representada a seguir, é uma implementação da função erro (JEFFREYS, 2003; ACTON, 1990)

$$\mathrm{erf}(z) = 2 \cdot \frac{1}{\sqrt{\pi}} \int_0^z e^{-t^2} \cdot dt. \tag{2.41}$$

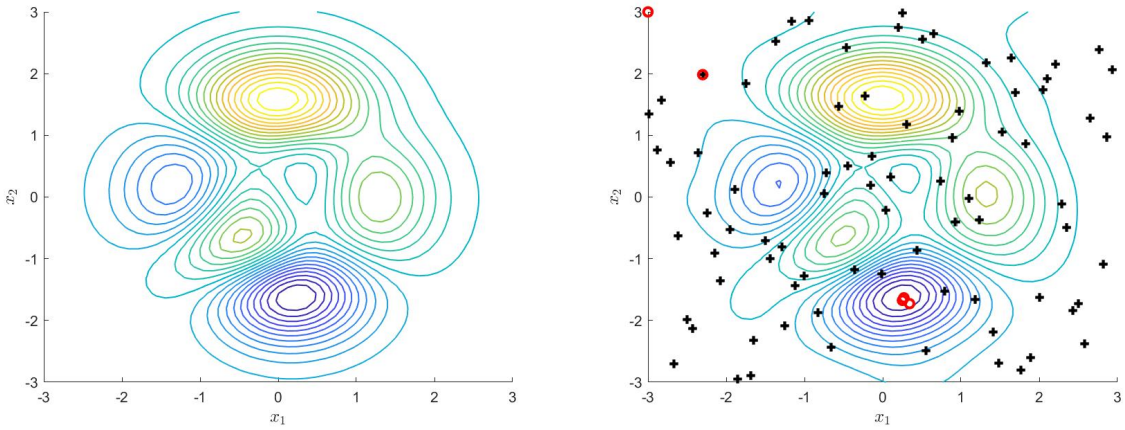
2.6.2 EGO

Efficient Global Optimization (EGO) também referido como *Sequential Kriging-based Optimization* (SKO) é um algoritmo desenvolvido para otimização de funções "caixa-preta", que combina um metamodelo DACE com sucessivas adaptações da amostra por meio da função EI. Proposto inicialmente por (JONES; SCHONLAU; WELCH, 1998), o EGO popularizou o uso da função EI no processo de refinamento de um metamodelo (ROUSTANT; GINSBOURGER; DEVILLE, 2012). Um pseudocódigo do EGO, adaptado de (ROUSTANT; GINSBOURGER; DEVILLE, 2012), é apresentado no Algoritmo 2.4.

Algoritmo 2.4: Efficient Global Optimization (EGO).

```
1 Gera amostra inicial X usando Hipercubo Latino;
2 Avalia X: y(X);
3 Constrói um metamodelo DACE;
4 enquanto critério de parada não é satisfeito faça
5 |   Maximiza EI e obtém x̃;
6 |   Avalia x̃: y(x̃);
7 |   Atualiza amostra: X ← X ∪ {x̃} e y ← y ∪ {y(x̃)};
8 |   Atualiza DACE
9 fim
```

Para ilustrar uma aplicação do método EGO, na Figura 2 são apresentados os gráficos com as curvas de nível na função *Peaks* (2.42) original y(·) e da função *Peaks*



(a) *Peaks* original (b) *Peaks* aproximada

Figura 2 – Curvas de nível da função y em (a) e curvas de nível da função ŷ em (b), com pontos da amostra identificados por + e novos pontos da amostra o obtidos via maximização da função EI.

aproximada ŷ(·). Considerou-se no Algoritmo 2.4 uma amostra inicial $\mathbf{X}$ com $N = 80$ soluções. Em seguida, atualizou-se sequencialmente o metamodelo com soluções obtidas via maximização da função EI. Definiu-se como critério de parada não satisfazer alguma das condições apresentadas em (2.37). Com esse critério, atualizou-se a amostra do metamodelo com 5 novas soluções, indentificadas com o símbolo o na Figura 2(b). A região do ótimo global dessa função corresponde a parte inferior do gráfico em (b) identificada com o símbolo o. Observe que no Algoritmo 2.4 a amostra do metamodelo aumenta a cada iteração.

$$\begin{aligned} \mathrm{y}(\mathbf{x}) \quad &= 3(1-x_1)^2 e^{[-x_1^2-(x_2+1)^2]} \\ &\quad -10\left(\tfrac{x_1}{5} - x_1^3 - x_2^5\right) e^{(-x_1^2-x_2^2)} - \tfrac{e^{[-(x_1+1)^2-x_2^2]}}{3}, \quad x_i \in [-3,3] \end{aligned} \tag{2.42}$$

2.7 Conclusão

O capítulo corrente apresentou uma visão geral da construção de aproximação global de funções via metamodelos *Kriging* e RBF, bem como o uso da função *Expected Improvement* para melhorar a acurácia de predição de metamodelos do tipo *Kriging*. Embora não tenham sido apresentados resultados associados ao uso de tais metamodelos no processo de otimização, o capítulo introduziu o algoritmo EGO que trata-se de uma estratégia desenvolvida para a otimização global de funções "caixa-preta" envolvendo poucas variáveis. Em (HAO; SHAOPING; TOMOVIC, 2010; ZHAO; CHOI; LEE, 2011; HAN; ZHANG, 2012) observou-se o uso do EGO e de variações desse método na otimização de problemas práticos com número de variáveis inferior a dez; essa mesma observação se faz com relação a construção de aproximação global via variações do modelo *Kriging* e integração com ferramentas de otimização em (ZHAO; CHOI; LEE, 2011; XIA et al., 2015; XIA; REN; KOH, 2015). De forma geral, os conceitos abordados no decorrer deste capítulo servem como ponto de partida para uma investigação e proposição de estratégias de otimização que empregam o uso de metamodelos em sua estrutura. Nesse contexto, os próximos capítulos abordam ferramentas integradas para otimização mono-objetivo.

3 Surrogate Model Assisted Evolutionary Algorithms

3.1 Introdução

No capítulo corrente são abordadas estratégias de otimização que integram EAs e metamodelos no contexto de problemas de otimização mono-objetivo da forma (1.1). Algoritmos Evolutivos representam uma alternativa para o tratamento de problemas de engenharia por não dependerem de características específicas do problema de otimização, e.g., diferenciabilidade e convexidade. No entanto, em situações em que existe alguma limitação associada ao custo de avaliação de funções, o uso desses algoritmos torna-se muito caro ou mesmo inviável. Visando contornar esse problema, diferentes estratégias na literatura integram metamodelos e EAs para reduzir o número de avaliações de função exigido por esses algoritmos. Este capítulo está organizado da seguinte forma: a Seção 3.2 é dedicada a uma revisão sobre Algoritmos Evolutivos Assistidos por Metamodelo, incluindo termos usuais nesse contexto e uma descrição de diferentes mecanismos adotados na literatura para inserir um metamodelo em EAs; finalmente na Seção 3.3 são apresentadas considerações sobre o desenvolvimento deste capítulo.

3.2 Algoritmos Evolutivos Assistidos por Metamodelo

Algoritmos Evolutivos Assistidos por Metamodelo (SAEAs, *Surrogate Model Assisted Evolutionary Algorithms*) são estratégias que usam metamodelos para substituir avaliações na função mérito de um dado problema (EMMERICH; GIANNAKOGLOU; NAUJOKS, 2006; LIU; ZHANG; GIELEN, 2014; SUN et al., 2017). Estes algoritmos se diferenciam, de modo geral, em relação ao mecanismo de incorporação e tipo do metamodelo escolhido. Para compreender melhor essas estratégias é apresentada inicialmente uma caracterização em relação a forma como um metamodelo é inserido em EAs, conforme abordado em (BÜCHE; SCHARAUDOLPH; KOUMOUNTSAKOS, 2005). Estes autores sugerem como principais estratégias de SAEAs as baseadas em *controle evolutivo* e as baseadas somente em *metamodelo*. Uma caracterização de SAEAs quanto ao tipo de metamodelo adotado é descrita na Seção 3.2.3, que apresenta uma visão geral de como esses métodos são estruturadas.

O uso de metamodelos em EAs envolve inicialmente a escolha do tipo de metamodelo a ser adotado que, uma vez definido, direciona a uma segunda questão relacionada a possibilidade de construção de uma aproximação global adequada para a função mérito.

Em muitos problemas práticos a realização de uma observação pode exigir elevado custo computacional, e.g., problemas envolvendo grande número de variáveis e uso de modelos de simulação; nesses casos é inviável obter a quantidade necessária de avaliações no modelo de simulação para construir um metamodelo que seja acurado em todo o espaço de busca. Para lidar com a segunda questão, (JIN, 2005) pontua duas medidas a serem adotadas, conforme descrição a seguir.

Primeiro, em um EA o metamodelo deve sempre ser usado em conjunto com o modelo de simulação para evitar o risco de convergência para um falso ótimo (JIN; OLHOFER; SENDHOFF, 2000). Além disso, considerando como motivação para o uso de metamodelo a redução no número de avaliações na função mérito, este recurso deve ser sempre usado de forma consciente. O termo controle evolutivo refere-se então a estabelecer quando as avaliações de funções envolvidas nos EAs serão realizadas na função mérito e quando serão realizadas no modelo de aproximação. A segunda medida é relacionada a acurácia do modelo que, mesmo com um limite de observações na função mérito, deve ser considerada. Uma forma de tratar essa questão é construir vários metamodelos durante o processo iterativo, ao invés de adotar um único modelo de aproximação em todo o processo (JIN; BRANKE, 2005; BÜCHE; SCHARAUDOLPH; KOUMOUNTSAKOS, 2005; EMMERICH; GIANNAKOGLOU; NAUJOKS, 2006; LIU; ZHANG; GIELEN, 2014). Nesse contexto, um metamodelo é usado para identificar soluções para serem avaliadas na função mérito, e.g., avalia-se no metamodelo as soluções geradas a partir da população corrente e escolhe-se a solução com melhor avaliação na função de aproximação. Entretanto, o metamodelo pode conduzir a uma escolha equivocada, principalmente se este é analisado em soluções muito distantes da região de aproximação. Nesse sentido, a cada geração um metamodelo é construído com uma amostra que, de certa forma, acompanha a evolução da população; por exemplo, pode-se escolher como amostra uma quantidade de soluções avaliadas mais recentemente na função mérito. A Seção 3.2.3, a qual descreve a estrutura de diferentes tipos de SAEAs, apresenta mais detalhes dessa abordagem.

Uma outra abordagem encontrada na literautra incorpora um método de redução de dimensão nas etapas de construção de metamodelo (LIU; ZHANG; GIELEN, 2014; YANG et al., 2019). Nessa abordagem, proposta em (LIU; ZHANG; GIELEN, 2014), a construção de metamodelo é realizada com uma amostra mapeada para um espaço de dimensão menor visando a redução do tempo computacional e a construção de uma aproximação mais acurada. Nos SAEAs propostos nos referidos trabalhos, emprega-se um metamodelo *Kriging*, cujo parâmetro é definido em um espaço de dimensão reduzida. Mais detalhes dessa estratégia são apresentados na Seção 3.2.3. Uma abordagem semelhate a redução dimensional é adotada em (CAI; LI, 2019), em que o SAEA proposto emprega um metamodelo *Kriging* com parâmetro unidimensional. O SAEA proposto no Capítulo 4 emprega essa abordagem.

3.2.1 SAEAs baseados em Controle Evolutivo

O conceito de *controle evolutivo* foi introduzido por (JIN; OLHOFER; SENDHOFF, 2000) ao investigar a convergência de uma Estratégia Evolutiva acoplada com um modelo de aproximação; no trabalho em questão, duas categorias são relatadas como formas de combinar o uso da função mérito e da função de aproximação. A primeira consiste em escolher uma quantidade de soluções de uma determinada geração, chamadas de *soluções controladas*, que são avaliadas na função mérito, enquanto as demais são avaliadas no metamodelo. Na segunda categoria, todas as soluções de uma dada geração são avaliadas na função mérito e, nesse caso, denomina-se *geração controlada*.

3.2.1.1 Soluções Controladas

Para realizar o controle de soluções é necessário definir como realizar a escolha dessas soluções, que pode ser, por exemplo, aleatória (MA et al., 2015) ou direcionada a partir de informações obtidas de um metamodelo (EMMERICH; GIANNAKOGLOU; NAUJOKS, 2006; LIU; ZHANG; GIELEN, 2014). Nos tópicos a seguir, adaptados de (JIN; BRANKE, 2005), são apresentadas as principais formas de identificar as soluções controladas.

- *Soluções aleatórias*: uma quantidade de soluções são escolhidas da população de forma aleatória.

- *Melhores soluções*: são escolhidas as melhores soluções considerando as avaliações no metamodelo.

- *Soluções incertas*: são selecionadas a partir do valor observado na medida de incerteza associada ao metamodelo, e.g., os modelos baseados em Processo Gaussiano permitem acessar o erro associado a predição de um dado ainda não observado. No caso de metamodelos que não retornam essa informação, uma medida para a incerteza é o cálculo da distância entre a alternativa analisada e o ponto mais próximo a esta que pertence à amostra com a qual o metamodelo é construído.

- *Método de pré-seleção*: nesse método $\lambda_{Pre} > \lambda$ novas soluções são geradas através de uma estratégia evolutiva (μ, λ) e avaliadas no metamodelo, enquanto que as λ soluções são avaliadas na função mérito.

- *Soluções representativas*: as soluções da população são agrupadas em um certo número de subconjuntos e as alternativas representativas de cada agrupamento são as que serão controladas.

3.2.1.2 Geração Controlada

Da mesma forma que no controle de soluções, o controle de geração é apresentado na literatura assumindo diferentes regras mas que, de modo geral, se distinguem em relação a forma de estabelecer a frequência do controle evolutivo.

- *Controle pela convergência*: a ideia nesse caso é controlar a primeira geração de um algoritmo evolutivo e usar um metamodelo para realizar as avaliações nas gerações seguintes; quando não houver melhora por sucessivas gerações o controle é aplicado novamente na próxima geração (RATLE, 1998).

- *Controle fixo*: define-se um parâmetro k para o qual as soluções são avaliadas na função mérito a cada k gerações, sendo k redefinido durante o processo evolutivo do algoritmo (JIN, 2005).

- *Controle adaptativo*: nessa abordagem, durante o processo iterativo de um EA, realiza-se o controle de η gerações em cada ciclo de λ gerações, sendo $1 \leq \eta \leq \lambda$ e η determinada a cada ciclo. Essa quantidade apropriada η, associada ao ciclo corrente, é estabelecida de forma dependente da acurácia do metamodelo construído no ciclo anterior. Para avaliar a acurácia do metamodelo é adotada uma medida de erro entre o valor de predição e o valor observado na função mérito associado às ηN soluções controladas no último ciclo, em que N é o tamanho da população (JIN; OLHOFER; SENDHOFF, 2001; JIN, 2005).

3.2.2 SAEAs baseados apenas no Metamodelo

A ideia geral dessas estratégias descrita em (BÜCHE; SCHARAUDOLPH; KOUMOUNTSAKOS, 2005; TORCZON; TROSSET, 1998) consiste em construir um metamodelo a partir de uma amostra inicial; os passos seguintes envolvem a otimização desse metamodelo, com o objetivo de melhorar a qualidade da aproximação. A solução ótima aproximada para o metamodelo é considerada uma solução candidata que pode favorecer a busca pela solução ótima do problema original. Assim, o ótimo aproximado do metamodelo é avaliado na função mérito, inserido na amostra inicial e usado para a reconstrução do metamodelo. Novamente, o metamodelo é otimizado e o processo iterativo se repete até um critério de convergência ser satisfeito.

Em (HAN; ZHANG, 2012) adota-se o termo Otimização Baseada em Metamodelos (SBO, *Surrogate-Based Optimization*) para referir-se as essas estratégias; no referido trabalho, os autores descrevem o processo de refinamento do metamodelo de uma forma mais abrangente, apresentando outras regras para mapear o ponto de atualização da amostra, como a *Expected Improvement* e *Lower Confidence Bound*. Tais regras são identificadas como *infill criteria* e, conforme abordagem anterior, outras formas de atualização da

amostra são descritas em (FORRESTER; SÓBESTER; KEANE, 2008). No contexto deste trabalho, identifica-se essas regras como função *prescreening*.

3.2.3 Construção de SAEAs baseados em Controle Evolutivo

Esta seção descreve uma visão geral da estrutura de SAEAs considerando o tipo de metamodelo envolvido. Nesse contexto, o uso do metamodelo Processo Gaussiano é frequente na literatura (BÜCHE; SCHARAUDOLPH; KOUMOUNTSAKOS, 2005; LIU; ZHANG; GIELEN, 2014; EMMERICH; GIANNAKOGLOU; NAUJOKS, 2006). Uma observação importante é que muitos trabalhos referem-se ao Processo Gaussiano como DACE ou *Kriging* (EMMERICH; GIANNAKOGLOU; NAUJOKS, 2006; HAO; SHAOPING; TOMOVIC, 2010; SUN et al., 2017; ZHOU et al., 2007) mas, como pode ser observado no Capítulo 2, esse último modelo foi definido de uma forma mais abrangente. Assim, ao descrever alguns SAEAs nos tópicos seguintes é mantido exatamente o mesmo termo usado nas respectivas referências para identificar o metamodelo adotado.

Em relação ao metamodelo é necessário considerar ainda se a função de aproximação é global ou local ou se ambos os tipos de aproximação são usados, sendo esse último caso identificado como *ensemble-surrogate* (SUN et al., 2017). Metamodelos globais são empregados com o objetivo de construir uma função de aproximação em todo o espaço de busca, no entanto, (SUN et al., 2017; ZHOU et al., 2007; BÜCHE; SCHARAUDOLPH; KOUMOUNTSAKOS, 2005) destacam que não é sempre possível construir globalmente uma aproximação confiável em problemas mais complicados (e.g., grande número de variáveis, multimodais, descontínuos) e, diante dessa questão, muitos trabalhos empregam modelos de aproximação local como uma alternativa para tratar essa situação. Embora metamodelos locais permitam realizar predições com uma acurácia satisfatória, a qualidade da aproximação é de certa forma restrita a região onde o metamodelo é construído. Outra questão relacionada ao metamodelo local abordada em (SUN et al., 2017) é que o uso em EAs pode conduzir o algoritmo a uma estagnação em uma região de ótimo local. Essas questões relacionadas a natureza da aproximação local inviabilizam usufruir de todo o potencial do metamodelo, e.g., modelos *Kriging* permitem adotar a função *Expected Improvement* como um indicativo de região a ser explorada. Entretanto, esse contexto pode comprometer a confiabilidade da EI quando analisada em pontos muitos distantes da região de aproximação. Nesse sentido, um valor alto de EI pode estar associado a elevada incerteza em relação a predição, o que não necessariamente representa uma contribuição para o EA.

Para aproveitar os benefícios que cada metamodelo global e local proporciona, a literatura apresenta como alternativa o desenvolvimento de SAEAs que empregam esses dois tipos de metamodelos. Diferentes trabalhos são citados em (SUN et al., 2017) e classificados em relação ao uso de metamodelo global, local e *ensemble-surrogate*. Na seção

corrente a descrição das estratégias apresentadas considera essas características. Antes de apresentar alguns exemplos de SAEAs encontrados na literatura é conveniente adotar uma notação comum às estratégias consideradas. Nesse contexto, considere $\mathrm{y}(\cdot)$ a função que representa o problema a ser otimizado, Ω a respectiva região factível; DB o conjunto de armazenamento de soluções avaliadas em $\mathrm{y}(\cdot)$, em geral DB tem cardinalidade limitada; e P a população corrente do algoritmo.

O primeiro exemplo de SAEA considerado aqui é o *Gaussian Process Optimization Procedure* (GPOP) ilustrado no Algoritmo 3.1. Proposto por (BÜCHE; SCHARAUDOLPH; KOUMOUNTSAKOS, 2005), esse método tem como ideia geral construir uma aproximação local na vizinhança da melhor solução corrente e mapear pontos a partir da otimização de uma função *prescreening*. Inicialmente gera-se uma amostra com N_c pontos em Ω e depois segue-se a atualização de DB. No processo iterativo, a amostra inicial para construir um metamodelo no passo 6 é formada pelos N_c pontos mais próximos da melhor solução corrente $\mathbf{x}^{best}$ e pelos N_R pontos recentemente avaliados, identificados respectivamente por X_c e X_R. Para mapear novos pontos a serem avaliados em $\mathrm{y}(\cdot)$ é considerada a função *Lower Confidence Bound* (LCB) (TORCZON; TROSSET, 1998; HAN; ZHANG, 2012), identificada também como *Statistical Lower Bound* em (FORRESTER; SÓBESTER; KEANE, 2008),

$$\mathrm{y}_M(\mathbf{x}) = \hat{\mathrm{y}}(\mathbf{x}) - \alpha\hat{s}(\mathbf{x}), \quad \mathbf{x} \in \mathcal{D} \tag{3.1}$$

onde

$$\mathcal{D} = \left\{ \mathbf{x} \in \Omega \,|\, \mathbf{x}^{best} - \mathbf{d} \leq \mathbf{x} \leq \mathbf{x}^{best} + \mathbf{d}, d_i = \frac{1}{2}\left(\max_{c\in\{1,\cdots,Nc\}} (x_i^c) - \min_{c\in\{1,\cdots,Nc\}} (x_i^c) \right), i = 1, \cdots, n \right\}.$$

No trabalho em questão considera-se que usar mais do que quatro variações de α não é benéfico para a convergência do processo de otimização. Dessa forma, para cada $\alpha = 0, 1, 2, 4$ a função em (3.1) é minimizada usando um algoritmo evolutivo no passo 9 e as soluções retornadas são avaliadas na função mérito do problema. Note que todas as soluções já avaliadas são armazenadas no conjunto DB, o qual é atualizado a cada iteração do algoritmo. Ao final do processo iterativo retorna-se a melhor solução $\mathbf{x}^\star$ observada em DB.

Um outro exemplo de SAEA, ilustrado no Algoritmo 3.2, incorpora um Processo Gaussiano identificado como *Gaussian Random Field Metamodels* (GRFM) em uma $(\mu + \nu < \lambda)-$ES (EMMERICH; GIANNAKOGLOU; NAUJOKS, 2006). Nesse método, usa-se metamodelos para pré-selecionar ν soluções consideradas mais promissoras para serem avaliadas na função mérito; esse processo de pré-seleção é denotado com o termo *prescreening*. O primeiro passo do algoritmo envolve a geração da população P com μ soluções, que é avaliada em $\mathrm{y}(\cdot)$ e posteriormente armazenada em DB. Em cada etapa do processo iterativo, seguindo (REGIS; SHOEMAKER, 2004), constrói-se um GRFM local para cada solução $\mathbf{x}^i$, $i \in \{1, \ldots, \lambda\}$, gerada a partir da população corrente. A amostra do

Algoritmo 3.1: Gaussian Process Optimization Procedure (GPOP)

Entrada: $\Omega, \mathrm{y}(\cdot), Nc, N_R$
Gera Nc pontos em Ω e avalia em $\mathrm{y}(\cdot)$;
Atualiza DB;
enquanto *critério de parada não é satisfeito* **faça**
  $\mathbf{x}^{best} \leftarrow$ melhor solução em DB;
  Identifica X_c e X_R;
  Constrói um metamodelo GP com a amostra $X \leftarrow X_c \cup X_R$;
  $A \leftarrow \emptyset$;
  para $\alpha = 0, 1, 2, 4$ **faça**
    $\mathbf{x}^{\alpha_i} \leftarrow arg\min_{\mathbf{x} \in \mathcal{D}} \mathrm{y}_M(\mathbf{x})$;
    $A \leftarrow A \cup \{\mathbf{x}^{\alpha_i}\}$;
  fim
  Elimina pontos de A que pertençam a DB, caso existam;
  Avalia A em $\mathrm{y}(\cdot)$;
  Atualiza DB com as novas avaliações;
fim
Saída: $\mathbf{x}^\star \leftarrow$ melhor solução em DB

metamodelo é definida como N_r soluções mais próximas a $\mathbf{x}^i$ entre as armazenadas em DB, sendo $N_r = 2n$ e n é o número de variáveis. Em seguida, seleciona-se as $1 \leq \nu \leq \mu$ soluções consideradas mais promissoras entre as avalidas no respectivo metamodelo. Nesse processo de *prescreening*, entre as opções apresentadas estão o uso de *Lower Confidence Bound* (LCB), *Expected Improvement* e *Probability of Improvement* (PoI). O conjunto DB é atualizado com as ν soluções avaliadas em $\mathrm{y}(\cdot)$ e, em seguida, μ soluções de $Q \cup P$ são selecionadas para a próxima geração. O último passo do algoritmo retorna a melhor solução $\mathbf{x}^\star$ no conjunto de armazenamento.

Algoritmo 3.2: $(\mu + \nu < \lambda)-$ES Assisted by GRFM

Entrada: $\Omega, \mathrm{y}(\cdot), N_r, (\mu + \nu < \lambda)$
Gera a população inicial P com μ soluções em Ω e avalia em $\mathrm{y}(\cdot)$;
Atualiza DB;
enquanto *critério de parada não é satisfeito* **faça**
  $G \leftarrow$ conjunto com λ soluções geradas a partir de P;
  para cada *solução* $\mathbf{x}^i$ *em* G **faça**
    $X_r \leftarrow N_r$ soluções mais próximas de $\mathbf{x}^i$ em DB;
    Constrói um GRFM com X_r e avalia $\mathbf{x}^i$ no metamodelo;
  fim
  $Q \leftarrow$ conjunto com as ν soluções mais promissoras de G;
  Avalia Q em $\mathrm{y}(\cdot)$;
  Atualiza DB;
  $P \leftarrow$ conjunto com μ melhores soluções de $Q \cup P$;
fim
Saída: $\mathbf{x}^\star \leftarrow$ melhor solução em DB

Como pode ser observado, os SAEAs abordados anteriormente não envolvem em um mesmo *framework* a construção local e global de metamodelos. Um exemplo de estratégia que combina o uso de aproximação global e local é o *Surrogate-Assisted Genetic Algorithm with Global and Local Search Strategy* (SAGA-GLS) proposto em (ZHOU et al., 2007). No Algoritmo 3.3 os primeiros passos são voltados para a execução de um EA durante uma quantidade específica de gerações, e.g., 3 gerações, sendo todas as avaliações realizadas na função $\mathrm{y}(\cdot)$ e respectivas soluções armazenadas em DB. A etapa

de busca global desse método emprega um *Data Parallel Gaussian Process* (DPGP) para construir uma aproximação global e identificar (via *Probability of Improvement*) η soluções promissoras na população corrente P do EA, as quais são armazenadas em S. Nessa estratégia define-se uma partição da amostra Q usada na construção de um metamodelo Processo Gaussiano usando um algoritmo de agrupamento proposto em (CHOUDHURY; NAIR; KEANE, 2002). Nesse contexto, a matriz de correlação em (2.14) é adequada em termos dos agrupamentos, sendo $\mathrm{R}(\mathbf{x}, \tilde{\mathbf{x}}) = \prod_{i=1}^{n} \exp(-\theta_i |x_i - \tilde{x}_i|^2)$ se $\mathbf{x}$ e $\tilde{\mathbf{x}}$ pertencem ao mesmo agrupamento e $\mathrm{R}(\mathbf{x}, \tilde{\mathbf{x}}) = 0$ caso contrário. Em cada solução $\mathbf{x}^i$ do conjunto

Algoritmo 3.3: Surrogate-Assisted Genetic Algorithm with Global and Local Search Strategy (SAGA-GLS)

Entrada: $\mathrm{y}(\cdot), q, \eta, m$
enquanto *critério de parada não é satisfeito* **faça**
  Executa um EA;
  Atualiza DB;
fim
$P \leftarrow$ população corrente do EA;
$Q \leftarrow$ conjunto com as q melhores soluções em DB;
Constrói um metamodelo DPGP com a amostra Q;
enquanto *critério de parada não é satisfeito* **faça**
  se *O conjunto Q é modificado* **então**
    Constrói um metamodelo DPGP com o conjunto Q atualizado;
  fim
  Avalia a população P com o metamodelo obtido no passo anterior;
  $S \leftarrow$ Conjunto com as η soluções promissoras em P via *Probability of Improvement*;
  para *cada solução* $\mathbf{x}^i$ *em* S **faça**
    $V \leftarrow$ conjunto com os m pontos de DB em uma vizinhança de $\mathbf{x}^i$;
    Constrói um metamodelo RBF usando o conjunto V e realiza uma busca local;
    Avalia em $\mathrm{y}(\cdot)$ a solução $\mathbf{x}_b$ retornada pela busca local;
    Substitua a solução $\mathbf{x}^i$ em P por $\mathbf{x}_b$ caso a busca local represente melhora;
    Atualiza DB;
    Atualiza Q;
  fim
  Evolua a população corrente P com operadores do EA;
fim
Saída: $\mathbf{x}^\star \leftarrow$ melhor solução em DB

S é aplicada uma busca local envolvendo a otimização de um metamodelo RBF, que é construído dinamicamente com m pontos de DB que pertencem a uma vizinhança do ponto de interesse; nessa etapa a solução $\mathbf{x}_b$ retornada na busca local é avaliada em $\mathrm{y}(\cdot)$ e, caso essa solução represente melhora em relação a $\mathbf{x}^i$, a alternativa $\mathbf{x}^i$ em P é substituída por $\mathbf{x}_b$ com a atualização da respectiva avaliação. O conjunto DB é atualizado sempre que uma nova solução é avaliada em $\mathrm{y}(\cdot)$ e, ao final do processo iterativo, a estimativa ótima para o problema é a melhor solução $\mathbf{x}^\star$ armazenada em DB.

O método *Generalized Surrogate Single-Objective Memetic Algorithm* (GS-SOMA) proposto em (LIM et al., 2009) é um outro exemplo de SAEA que emprega diferentes metamodelos no processo evolutivo, os quais estão associados às etapas de buscas locais que são realizadas para cada solução da população corrente. Os primeiros passos do GS-SOMA, ilustrado no Algoritmo 3.4, são voltados para a execução de um EA por um

determinado período de iterações, o qual pode ser executado até alcançar a quantidade mínima m de soluções necessárias para a construção dos metamaodelos nos passos 12 e 15. De uma forma diferente de vários SAEAs, o GS-SOMA define uma função de predição como uma soma ponderada envolvendo os metamodelos GP, RBF e Regressão Polinomial (PR, *Polynomial Regression*), os quais são identificados respectivamente por $\hat{\mathrm{q}}_1(\cdot), \hat{\mathrm{q}}_2(\cdot)$ e $\hat{\mathrm{q}}_3(\cdot)$. No passo 14, o qual identifica a função de predição $\hat{\mathrm{y}}_1(\cdot)$, os escalares c_1, c_2 e c_3 são tais que $\sum_{j=1}^{3} c_j = 1$. Para cada solução da população corrente o GS-SOMA realiza duas etapas separadas de busca local, as quais retornam as soluções que serão avaliadas em $\mathrm{y}(\cdot)$ a cada iteração e posteriormente inseridas no conjunto DB. Em relação a atualização da população corrente, a solução obtida no passo 19 é selecionada para a próxima iteração somente se representar melhora em relação a respectiva solução para a qual realizou-se as buscas locais. Ao final do processo evolutivo retorna-se a melhor solução $\mathbf{x}^\star$ armazenada em DB.

Algoritmo 3.4: Generalized Surrogate Single-Objective Memetic Algorithm (GS-SOMA)

Entrada: $\Omega, \mathrm{y}(\cdot), N, n, m, c_1, c_2, c_3$
1 Gera a população inicial P com N soluções em Ω e avalia em $\mathrm{y}(\cdot)$;
2 **enquanto** *critério de parada não é satisfeito* **faça**
3 Executa um EA;
4 Atualiza DB;
5 **fim**
6 $P \leftarrow$ população corrente do EA;
7 $G \leftarrow$ conjunto com N soluções geradas a partir de P;
8 **enquanto** *critério de parada não é satisfeito* **faça**
9 **para** *cada solução* $\mathbf{x}^i$ *em* P **faça**
10 $V \leftarrow$ conjunto com os m pontos de DB em uma vizinhança de $\mathbf{x}^i$;
11 **para** $j \in \{1, 2, 3\}$ **faça**
12 Constrói um metamodelo $\hat{\mathrm{q}}_j(\cdot)$ com a amostra V;
13 **fim**
14 $\hat{\mathrm{y}}_1(\cdot) \leftarrow \sum_{j=1}^{3} c_j \hat{\mathrm{q}}_j(\cdot)$;
15 Constrói um metamodelo $\hat{\mathrm{q}}_3(\cdot)$ com a amostra V;
16 Aplica uma busca local em $\hat{\mathrm{y}}_1(\cdot)$ e em $\hat{\mathrm{q}}_3(\cdot)$ e retorna $\mathbf{x}^1_{opt}$ e $\mathbf{x}^2_{opt}$, respectivamente;
17 Avalia as soluções $\mathbf{x}^1_{opt}$ e $\mathbf{x}^2_{opt}$ em $\mathrm{y}(\cdot)$ e atualiza DB;
18 $\mathbf{x}_{opt} \leftarrow$ solução com melhor valor de $\mathrm{y}(\cdot)$ escolhida em $\{\mathbf{x}^1_{opt}, \mathbf{x}^2_{opt}\}$;
19 Substitui $\mathbf{x}^i$ em P por $\mathbf{x}_{opt}$ caso esta nova solução represente melhora;
20 **fim**
21 **fim**
Saída: $\mathbf{x}^\star \leftarrow$ melhor solução em DB

Seguindo uma estrutura semelhante aos SAEAs ilustrados nos Algoritmos 3.1 e 3.2, em (LIU; ZHANG; GIELEN, 2014) é proposto o *Gaussian Process Surrogate Model Assisted Evolutionary Algorithm for Mediun-scale Computationally Expensive Optimization Problems* (GPEME) com as ideias principais ilustradas no Algoritmo 3.5. Inicialmente são geradas α soluções em Ω usando o Hipercubo Latino. Essas soluções são avaliadas na função mérito do problema e, em seguida, DB é atualizado com as mesmas. No processo iterativo são geradas λ novas soluções usando operadores modificados do algoritmo DE a

partir da população P formada pelas λ melhores soluções do conjunto de armazenamento DB. Um metamodelo GP é construído com uma amostra X constituída pelas τ soluções mais recentes de DB. O objetivo é usar esse metamodelo na identificação da solução $\tilde{\mathbf{x}}$ a ser avaliada na função mérito; a cada iteração DB é atualizado com essa nova solução. Especificamente, $\tilde{\mathbf{x}}$ corresponde a solução com o menor valor de avaliação de função entre as soluções avaliadas em *Lower Confidence Bound* (LCB). Uma característica interessante desse método que o distingue dos demais é o uso do método de redução dimensional *Sammon mapping* (SAMMON, 1969) apresentado na Seção 3.2.3.1. Nessa abordagem, o metamodelo construído no passo 8 utiliza uma amostra X_r, que é a imagem de $X \subset \mathbb{R}^n$ no espaço de dimensão reduzida $\mathbb{R}^l$, $l < n$. (LIU; ZHANG; GIELEN, 2014) propõe o uso dessa técnica como uma forma de reduzir o custo computacional e de melhorar a qualidade da função de aproximação no contexto de problemas com um grande número de variáveis ($n = 50$). Nesse sentido, constrói-se um metamodelo com uma amostra que, de certa forma, representa a região definida pela população corrente. E ainda, essa função de aproximação é construída em um espaço com dimensão menor que do espaço original. Ao final do processo iterativo, a solução $\mathbf{x}^\star$ retornada é a melhor solução do conjunto de

Algoritmo 3.5: Gaussian Process Surrogate Model Assisted Evolutionary Algorithm for Mediun-scale Computationally Expensive Optimization Problems (GPEME)

Entrada: $\Omega, \mathrm{y}(\cdot), \alpha, \tau, \lambda, l$
1 Gera α soluções em Ω e avalia em $\mathrm{y}(\cdot)$;
2 Atualiza DB;
3 **enquanto** *critério de parada não é satisfeito* **faça**
4 $P \leftarrow$ conjunto com λ melhores soluções de DB;
5 $\tilde{P} \leftarrow$ gera λ novas soluções a partir de P com operadores do DE;
6 $X \leftarrow$ conjunto com as τ soluções mais recentes de DB;
7 $X_r \cup \tilde{P}_r \leftarrow$ imagem de $X \cup \tilde{P}$ no espaço de dimensão reduzida $\mathbb{R}^l$;
8 Constrói um metamodelo GP a partir da amostra X_r;
9 $\tilde{\mathbf{x}}_r \leftarrow$ melhor solução ao avaliar $\tilde{P}_r$ com a função LCB;
10 $\tilde{\mathbf{x}} \leftarrow$ correspondente de $\tilde{\mathbf{x}}_r$ em $\tilde{P}$;
11 Avalia $\tilde{\mathbf{x}}$ em $\mathrm{y}(\cdot)$ e atualiza DB;
12 **fim**
Saída: $\mathbf{x}^\star \leftarrow$ melhor solução em DB

armazenamento DB. É importante mencionar que, para problemas com $n = 20, 30$, (LIU; ZHANG; GIELEN, 2014) aplica o GPEME sem o método de redução dimensional.

Nas abordagens ilustradas nos Algoritmos 3.1, 3.2 e 3.5, o controle de soluções envolve o uso de funções como *Lower Confidence Bound*, *Expected Iprovement* e *Probability of Improvement* ou até mesmo o valor assumido no metamodelo. De uma forma diferente, (MA et al., 2015) propõem uma estratégia de controle adaptado em que durante a geração corrente as soluções controladas são escolhidas de forma aleatória e usadas na predição da avaliação das alternativas não controladas, conforme os passos 6-12 do Algoritmo 3.6. Nos primeiros passos do algoritmo tem-se a geração de N_c soluções, com respectivas avaliações e atualização de DB. Os próximos passos são a construção de um metamodelo global

considerando como amostra inicial o conjunto DB e a identificação do conjunto P com as N melhores soluções em DB. No processo iterativo, $N_c > N$ novas soluções são geradas a partir de P de acordo com o EA adotado; em seguida, N destas soluções são escolhidas aleatoriamente e armazenadas em $\tilde{R}$. Nos passos 8-12, essas N soluções são avaliadas em $\mathrm{y}(\cdot)$ e usadas na atualização do DB. Uma vez que o metamodelo é reconstruído, o conjunto $\tilde{S}$ com as $N_c - N$ soluções não avaliadas é avaliado com o metamodelo. Na atualização da população para a próxima geração (passo 13) são escolhidas N alternativas a partir do conjunto formado pela união das soluções controladas (avaliadas na função mérito) e as não controladas (avaliadas apenas na função de aproximação), sendo o DACE o metamodelo adotado com as mesmas considerações da Seção 2.4. A melhor solução entre as avaliações realizadas em $\mathrm{y}(\cdot)$ é retornada em $\mathbf{x}^\star$. Um detalhe interessante desse SAEA é que (MA et al., 2015) investiga o desempenho do Algoritmo 3.6 usando diferentes EAs no passo 6. Observa-se um melhor desempenho para a configuração que emprega o SaDE (QIN; SUGANTHAN, 2005; DAS; SUGANTHAN, 2011).

Algoritmo 3.6: Updated-based Evolution Control

Entrada: $\Omega, \mathrm{y}(\cdot), N_c > N$
1 Gera Nc soluções em Ω e avalia em $\mathrm{y}(\cdot)$;
2 Atualiza DB;
3 Constói um metamodelo DACE usando DB;
4 $P \leftarrow$ conjunto com N melhores soluções de DB;
5 **enquanto** *critério de parada não é satisfeito* **faça**
6 $\quad \tilde{P} \leftarrow$ gera Nc novas soluções a partir de P usando um EA;
7 $\quad \tilde{R} \leftarrow$ conjunto com N soluções selecionadas aleatoriamente em $\tilde{P}$;
8 $\quad$ Avalia $\tilde{R}$ em $\mathrm{y}(\cdot)$;
9 $\quad$ Atualiza DB;
10 $\quad$ Atualiza o DACE com as novas avaliações;
11 $\quad \tilde{S} \leftarrow \tilde{P} \setminus \tilde{R}$;
12 $\quad$ Avalia $\tilde{S}$ usando o DACE;
13 $\quad P \leftarrow$ população com N soluções selecionadas a partir de $\tilde{R} \cup \tilde{S}$;
14 **fim**
Saída: $\mathbf{x}^\star \leftarrow$ melhor solução em DB

O método proposto em (GONG; ZHOU; CAI, 2015), o *Cheap Surrogate Model* (CSM) ilustrado no Algoritmo 3.7, é um outro exemplo de SAEA que apresenta uma abordagem distinta em relação a escolha das soluções a serem avaliadas na função mérito. Diferente de várias estratégias apresentadas no capítulo corrente, o CSM não utiliza metamodelos como *Kriging* e RBF que usualmente são observados no contexto de SAEAs. No CSM seleciona-se as soluções que serão avaliadas na função mérito de acordo com o valor estimado para a densidade de probabilidade das soluções candidatas. Além disso, esse método é estruturado também na ideia do uso de múltiplos operadores na geração novas soluções para melhorar a capacidade do EA em identificar soluções promissoras. Os operadores adotados no CSM correspondem a variações do DE e do PSO. Especificamente, para cada solução da população corrente gera-se λ novas soluções, cada uma associada a um operador diferente, e a solução com melhor valor estimado de densidade de probabilidade $\hat{\mathrm{F}}(\cdot)$ é escolhida para ser avaliada em $\mathrm{y}(\cdot)$. Para a atualização da população considera-se

substituir cada solução $\mathbf{x}^i$ em P pela respectiva solução $\tilde{\mathbf{x}}^\star$ somente se a nova solução representar melhora na função mérito. A solução $\mathbf{x}^\star$ retornada pelo CSM é a melhor solução entre as avaliações realizadas em $\mathrm{y}(\cdot)$.

Algoritmo 3.7: Cheap Surrogate Model (CSM)

Entrada: $\Omega, \mathrm{y}(\cdot), \mu, \lambda$
1 Gera a população inicial P com μ soluções em Ω e avalia em $\mathrm{y}(\cdot)$;
2 **enquanto** *critério de parada não é satisfeito* **faça**
3 **para** *cada solução* $\mathbf{x}^i$ *em* P **faça**
4 $P_0 \leftarrow$ conjunto com λ soluções geradas a partir de P com λ operadores distintos;
5 Estima a densidade de probabilidade $\hat{\mathrm{F}}(\cdot)$ de cada solução $\tilde{\mathbf{x}}^i$ em P_0;
6 $\tilde{\mathbf{x}}^\star \leftarrow$ solução de P_0 com melhor valor de $\hat{\mathrm{F}}(\cdot)$;
7 Avalia $\tilde{\mathbf{x}}^\star$ em $\mathrm{y}(\cdot)$;
8 Substitui $\mathbf{x}^i$ em P por $\tilde{\mathbf{x}}^\star$ caso a nova solução represente melhora;
9 **fim**
10 **fim**
Saída: $\mathbf{x}^\star \leftarrow$ melhor solução entre as avaliações realizadas em $\mathrm{y}(\cdot)$

Um exemplo recente de um SAEA envolvendo metamodelo global e local é o *Surrogate-assisted Cooperative Swarm Optimization Algorithm* (SA-COSO) proposto em (SUN et al., 2017). No processo iterativo desse método, um algoritmo PSO assistido por um metamodelo RBF é usado na busca pela região em que se encontra o ótimo global, enquanto que um PSO assistido por uma *Fitness Estimation Strategy* (FES) é aplicado na busca local na região de interesse. Optou-se por não apresentar o SA-COSO devido a sua estrutura ser mais extensa que a dos outros métodos relacionados neste capítulo. Embora não sejam apresentados detalhes do referido método neste trabalho, é importante mencionar que o SAEA proposto no Capítulo 4 emprega a mesma variação do operador PSO adotada no SA-COSO.

Em diferentes SAEAs que empregam RBF pode-se observar este tipo de metamodelo aplicado na realização de uma busca local para cada solução da população corrente. Entretanto, um SAEA recente proposto em (YU et al., 2018), identificado como *Surrogate-Assisted Hierarchical Particle Swarm Optimization* (SHPSO), emprega este tipo de metamodelo como uma aproximação da região definida pela população corrente em cada iteração do processo evolutivo. Os primeiros passos do SHPSO, ilustrado no Algoritmo 3.8, envolvem a geração de uma amostra inicial, seguida da identificação da população inicial P e dos parâmetros de um PSO. Nesse método utiliza-se um metamodelo RBF em duas etapas para identificar as soluções que serão avaliadas na função mérito. Na primeira etapa (passos 6 a 13) utiliza-se um SL-PSO para otimizar um metamodelo RBF, a qual obtém uma solução $\mathbf{x}_r$ que representa a solução ótima estimada da região de busca definida pela população corrente. Com essa nova solução avaliada em $\mathrm{y}(\cdot)$ verifica-se a necessidade de reconstrução do metamodelo e realiza-se também a atualização de informações do PSO. Na segunda etapa (passos 14 a 17) tem-se a geração de uma nova população P do PSO, a qual é avaliada no metamodelo RBF corrente. Nesse processo avalia-se em $\mathrm{y}(\cdot)$ cada solução de

P que representa melhora em relação a melhor solução conhecida pela respectiva partícula. Ao final do processo evolutivo retorna-se a melhor solução $\mathbf{x}^\star$ armazenada em DB.

Algoritmo 3.8: Surrogate-Assisted Hierarchical Particle Swarm Optimization (SHPSO)

Entrada: $\Omega, \mathrm{y}(\cdot), N_s, N, q$
1 Gera amostra inicial com N_s soluções em Ω e avalia em $\mathrm{y}(\cdot)$;
2 Atualiza DB;
3 $P \leftarrow$ população inicial do PSO definida pelas N melhores soluções de DB;
4 Inicializa parâmetros do PSO;
5 **enquanto** *critério de parada não é satisfeito* **faça**
6 $Q \leftarrow$ conjunto com q melhores soluções não duplicadas de DB;
7 Constrói um metamodelo RBF usando Q e retorna $\hat{\mathrm{y}}(\cdot)$;
8 $\mathbf{x}_r \leftarrow$ solução obtida ao otimizar o metamodelo $\hat{\mathrm{y}}(\cdot)$;
9 Avalia $\mathbf{x}_r$ em $\hat{\mathrm{y}}(\cdot)$ e atualiza DB;
10 **se** *se o conjunto Q é modificado* **então**
11 Constrói um metamodelo RBF usando Q e retorna $\hat{\mathrm{y}}(\cdot)$;
12 **fim**
13 Atualiza parâmetros do PSO;
14 Gera nova população P para o PSO;
15 Avalia P em $\hat{\mathrm{y}}(\cdot)$;
16 $X \leftarrow$ conjunto de soluções de P que representam melhora em relação a melhor solução conhecida pela respectiva partícula;
17 Avalia X em $\mathrm{y}(\cdot)$ e atualiza DB;
18 **fim**
Saída: $\mathbf{x}^\star \leftarrow$ melhor solução armazenada em DB

Outro SAEA apresentado aqui é o *Two-layer surrogate-assisted evolutionary algorithm for high-dimensional computationally expensive problems* (TASEA) proprosto em (YANG et al., 2019), ilustrado no Algoritmo 3.9, o qual compartilha algumas características com o GPEME (LIU; ZHANG; GIELEN, 2014). O TASEA, da mesma forma que o GPEME, emprega o método de redução dimensional *Sammon mapping* (SAMMON, 1969) na etapa de construção de metamodelo GP. Além disso, a geração da amostra inicial e atualização do conjunto DB são realizadas como no GPEME. Uma característica interressante do TASEA é o uso dos parâmetros N_f, N_c e *good*, os quais selecionam de forma adaptativa os operadores do DE para a geração de novas soluções, conforme ilustra o Algorithmo 3.10. Tais parâmetros estabelecem também a natureza do metamodelo GP, se a função de aproximação é global ou local. Quando a avaliação da solução $\tilde{\mathbf{x}}$ representa melhora em relação a melhor solução corrente $\mathbf{x}^\star$, o algoritmo explora essa região de melhora com a construção de um metamodelo local; a amostra nesse caso é definida como as soluções mais próximas a $\tilde{\mathbf{x}}$. Note que quando há melhora, devido a forma de escolher a população P, a solução $\mathbf{x}^{best}$ na iteração seguinte coincide com $\tilde{\mathbf{x}}$. O tamanho da amostra do metamodelo local aumenta de forma gradativa obedecendo o valor do parâmetro γ_{max}. Com relação ao parâmetro k envolvido no processo de *prescreening* (passo 18), (YANG et al., 2019) observa que a solução com melhor valor de $\mathrm{EI}(\cdot)$ do passo 18 não necessariamente possui o melhor valor de $\mathrm{y}(\cdot)$. Por outro lado, a melhor solução pode estar entre as k melhores soluções avaliadas em $\mathrm{EI}(\cdot)$, o que justifica então a escolha aleatória no passo 19.

O último SAEA apresentado nesse capítulo trata-se do *Efficient Generalized*

Algoritmo 3.9: Two-layer surrogate-assisted evolutionary algorithm for high-dimensional computationally expensive problems (TASEA)

Entrada: $\Omega, \mathrm{y}(\cdot), \alpha, \beta, \gamma_{max}, \mu, N_c, l$
Gera β soluções em Ω e avalia em $\mathrm{y}(\cdot)$;
Atualiza DB;
$\mathbf{x}^\star \leftarrow$ melhor solução entre as avaliadas em $\mathrm{y}(\cdot)$;
$N_f \leftarrow 1$, $good \leftarrow 0$, $\gamma \leftarrow 0$;
enquanto *critério de parada não é satisfeito* **faça**
  $P \leftarrow$ conjunto com α melhores soluções de DB;
  $\mathbf{x}^{best}$ melhor solução em P;
  $\tilde{P} \leftarrow$ gera α novas soluções a partir de P com operadores do DE (Algoritmo 3.10);
  se $good = 0$ **então**
    $N \leftarrow \mu$;
    $X \leftarrow$ conjunto com as N soluções mais recentes em DB;
  senão
    $N \leftarrow \mu + \gamma$;
    $X \leftarrow$ conjunto com as N soluções de DB mais próximas a $\mathbf{x}^{best}$;
  fim
  $X_r \cup \tilde{P}_r \leftarrow$ imagem de $X \cup \tilde{P}$ no espaço de dimensão reduzida $\mathbb{R}^l$;
  Constrói um metamodelo GP a partir da amostra X_r;
  $\tilde{\mathrm{X}}_r \leftarrow$ conjunto com as k melhores soluções ao avaliar $\tilde{P}_r$ com a função $\mathrm{EI}(\cdot)$;
  $\tilde{\mathbf{x}}_r \leftarrow$ solução aleatória em $\tilde{\mathrm{X}}_r$;
  $\tilde{\mathbf{x}} \leftarrow$ correspondente de $\tilde{\mathbf{x}}_r$ em $\tilde{P}$;
  Avalia $\tilde{\mathbf{x}}$ em $\mathrm{y}(\cdot)$ e atualiza DB;
  se $\mathrm{y}(\tilde{\mathbf{x}}) < \mathrm{y}(\mathbf{x}^\star)$ **então**
    $N_f \leftarrow 1$, $good \leftarrow 1$;
    $\mathbf{x}^\star \leftarrow \tilde{\mathbf{x}}$;
  senão
    $N_f \leftarrow N_f + 1$, $good \leftarrow 0$;
  fim
  $\gamma \leftarrow \min\{(\gamma + 1), \gamma_{max}\}$;
fim
Saída: $\mathbf{x}^\star \leftarrow$ melhor solução entre as avaliações realizadas em $\mathrm{y}(\cdot)$

Surrogate-Assisted Evolutionary Algorithm for High-Dimensional Expensive Problems (GSGA) proposto em (CAI; LI, 2019) e ilustrado no Algoritmo 3.12, o qual otimiza um metamodelo RBF para encontrar a melhor solução na vizinhança de cada solução da população corrente. Essas soluções mapeadas via um metamodelo RBF e a população corrente, identificadas respectivamente por $\tilde{P}$ e P, são usadas na geração de novas soluções C_{child}, as quais são geradas com operadores GA. Para identificar quais soluções, entre essas novas soluções, serão avalidas em $\mathrm{y}(\cdot)$ emprega-se um metamodelo *Kriging* para identificar as N_{new} soluções com melhor avaliação em $\mathrm{EI}(\cdot)$. É importante mencionar que o GSGA emprega um metamodelo *First-Order Kriging* (UK1) com otimização unidimensional para o parâmetro $\boldsymbol{\theta}$. Essa variação de modelo *Kriging* é referida no GSGA como *Simplified Kriging*. Nessa variação, o custo computacional de construção é menor em comparação com a construção de um modelo *Kriging* com $\boldsymbol{\theta}$ na forma usual. Na construção de cada metamodelo local RBF, considera-se como amostra uma quantidade N_{RBF} de soluções maior que $5 \cdot n$ na vizinhança V de cada solução $\mathbf{x}^i$, $i = 1, \ldots, N$, definida a seguir.

$$V = \left[\mathbf{x}^i - \mathrm{r}^i \cdot \mathbf{1}, \mathbf{x}^i + \mathrm{r}^i \cdot \mathbf{1}\right] \cap \Omega \tag{3.2}$$

Algoritmo 3.10: Operadores de Mutação do DE

Entrada: $P, \mathbf{x}^{best}, \alpha, n, N_f, N_c, good$
$\tilde{P} \leftarrow \emptyset$;
se $N_f < N_c$ e $good = 0$ **então**
  $F \leftarrow 0.8$, $C \leftarrow 0.8$;
  para cada $i = 1, \cdots, \alpha$ **faça**
    j_1, j_2 índices aleatórios em $\{1, \cdots, \alpha\} \setminus \{i\}$ tais que $j_1 \neq j_2$;
    $\tilde{\mathbf{x}} \leftarrow \mathbf{x}^i + F \cdot (\mathbf{x}^{best} - \mathbf{x}^i) + F(\mathbf{x}^{j_1} - \mathbf{x}^{j_2})$;
    $\tilde{\mathbf{u}} \leftarrow$ executa o Algoritno 3.11;
    $\tilde{P} \leftarrow \tilde{P} \cup \{\mathbf{u}\}$;
  fim
senão se $N_f \geq N_c$ e $good = 0$ **então**
  $F \leftarrow 0.8$, $C \leftarrow 0.6$;
  $F_{rand} \leftarrow$ valor aleatório com distribuição uniforme em $[0, 1]$;
  para cada $i = 1, \cdots, \alpha$ **faça**
    j_1, j_2 índices aleatórios em $\{1, \cdots, \alpha\} \setminus \{i\}$ tais que $j_1 \neq j_2$;
    $\tilde{\mathbf{x}} \leftarrow \mathbf{x}^i + F_{rand} \cdot (\mathbf{x}^{best} - \mathbf{x}^i) + F(\mathbf{x}^{j_1} - \mathbf{x}^{j_2})$;
    $\tilde{\mathbf{u}} \leftarrow$ executa o Algoritno 3.11;
    $\tilde{P} \leftarrow \tilde{P} \cup \{\mathbf{u}\}$;
  fim
senão se $good = 1$ **então**
  $F \leftarrow 0.9$, $C \leftarrow 0.2$;
  para cada $i = 1, \cdots, \alpha$ **faça**
    j_1, j_2 índices aleatórios em $\{1, \cdots, \alpha\} \setminus \{i\}$ tais que $j_1 \neq j_2$;
    $\tilde{\mathbf{x}} \leftarrow \mathbf{x}^{best} + F(\mathbf{x}^{j_1} - \mathbf{x}^{j_2})$;
    $\tilde{\mathbf{u}} \leftarrow$ executa o Algoritno 3.11;
    $\tilde{P} \leftarrow \tilde{P} \cup \{\mathbf{u}\}$;
  fim
fim
Saída: $\tilde{P}$

Algoritmo 3.11: Crossover

Entrada: $\mathbf{x}^i, \tilde{\mathbf{x}}, n, C$
para cada $v = 1, \cdots, n$ **faça**
  $c \leftarrow$ valor aleatório em $[0, 1]$, $r \leftarrow$ índice aleatório em $\{1, \cdots, n\}$;
  $\mathrm{u}_v \leftarrow \begin{cases} \tilde{\mathrm{x}}_v, \text{ se } c \leq C \text{ ou } r = v \\ \mathrm{x}^i_v, \text{caso contrário} \end{cases}$
fim
Saída: $\tilde{\mathbf{u}}$

$$r^i = \frac{(0.5)d_{imax}}{\sqrt{n}\sqrt[n]{N-1}} \tag{3.3}$$

onde $d_{imax} = \max_{\mathbf{x}^j \in P}(dist(\mathbf{x}^i, \mathbf{x}^j))$, $dist(\cdot, \cdot)$ é a função distância Euclidiana e $\mathbf{1}$ é um vetor linha $1 \times n$ com elementos iguais a 1, sendo n o número de variáveis. No caso de V não possuir a quantidade necessária de soluções, completa-se a amostra com soluções avaliadas em $\mathrm{y}(\cdot)$ mais próximas a $\mathbf{x}^i$. É importante mencionar ainda que a inicialização do GSGA exige uma quantidade mínima de soluções previamente avaliadas em $\mathrm{y}(\cdot)$ necessárias para a construção dos metamodelos do tipo RBF. Além disso, observa-se que o critério de parada adotado no GSGA é o número de avaliações realizadas em $\mathrm{y}(\cdot)$ a partir do passo 1. Com relação ao passo 14, os detalhes da geração de soluções com operadores GA são apresentados em (CAI; LI, 2019).

Algoritmo 3.12: Efficient Generalized Surrogate-Assisted Evolutionary Algorithm for High-Dimensional Expensive Problems (GSGA)

Entrada: $\Omega, \mathrm{y}(\cdot), N, N_{RBF}, N_P, N_{new}$
$P \leftarrow$ conjunto com N geradas em Ω e avaliadas em $\mathrm{y}(\cdot)$;
$N_P \leftarrow N - N_{new}$;
Atualiza DB;
enquanto *critério de parada não é satisfeito* **faça**
  $P \leftarrow$ ordena P da melhor para a pior solução;
  $\mathbf{x}_{best} \leftarrow$ melhor solução em P;
  para i = 1,...,N **faça**
    $\hat{P} \leftarrow \emptyset$;
    $R \leftarrow$ conjunto com N_{RBF} soluções na vizinhança V de $\mathbf{x}^i$;
    Constrói um metamodelo RBF $\hat{\mathrm{y}}_{RBF}(\cdot)$ usando R;
    $\hat{\mathbf{x}} \leftarrow arg\min_{\mathbf{x} \in V} \hat{\mathrm{y}}_{RBF}(\mathbf{x})$;
    $\hat{P} \leftarrow P \cup \{\hat{\mathbf{x}}\}$;
  fim
  $C_{child} \leftarrow$ soluções geradas a partir de P e $\hat{P}$ com operadores GA;
  Constrói um metamodelo *Simplified Kriging* usando DB e retorna EI$(\cdot)$;
  Avalia C_{child} em EI$(\cdot)$;
  $C_{new} \leftarrow N_{new}$ melhores soluções de C_{child};
  Avalia C_{new} em $\mathbf{y}(\cdot)$;
  Atualiza DB;
  $\tilde{P} \leftarrow N_P$ melhores soluções em P;
  $P \leftarrow \emptyset$;
  $P \leftarrow \tilde{P} \cup \{C_{new}\}$;
fim
Saída: $\mathbf{x}^\star \leftarrow$ melhor solução entre as avaliadas realizadas em $\mathrm{y}(\cdot)$

3.2.3.1 Método de Redução Dimensional

Esta seção apresenta uma visão geral do método de redução dimensional *Sammon mapping* (SAMMON, 1969). São apresentados também alguns detalhes de como essa técnica é empregada em (LIU; ZHANG; GIELEN, 2014; YANG et al., 2019) e em uma variação do SAEA proposto no Capítulo 4. Dada uma amostra com N pontos $X = \{\mathbf{x}^1, \cdots, \mathbf{x}^N\} \subset \mathbb{R}^n$, a ideia no método *Sammon mapping* é mapear X no espaço $l-$dimensional $\mathbb{R}^l$, $l < n$, de forma que a estrutura da distância entre pontos da amostra seja preservada o máximo possível. Dessa forma, o objetivo em *Sammon mapping* é minimizar a seguinte função erro

$$E = \left(\frac{1}{\sum_{i<j} d^\star_{ij}} \right) \sum_{i<j} \frac{(d^\star_{ij} - d_{ij})^2}{d^\star_{ij}} \tag{3.4}$$

onde $d^\star_{ij}$ é a distância entre $\mathbf{x}^i, \mathbf{x}^j \in X$, d_{ij} é a distância entre $\bar{\mathbf{x}}^i, \bar{\mathbf{x}}^j \in X_r$ e $X_r \subset \mathbb{R}^l$ é o correspondente de X em $\mathbb{R}^l$.

No contexto de SAEAs, o método *Sammon mapping* é aplicado conforme ilustrado no Algoritmo 3.13, no qual $X \subset \mathbb{R}^n$ representa a amostra para a construção do metamodelo; P é o conjunto de soluções a serem avaliadas no metamodelo (conjunto de soluções geradas a partir da população corrente); $2 \leq l < n$ é a dimensão do espaço reduzido $\mathbb{R}^l$ e $\hat{\mathrm{f}}(\cdot)$ é a função *prescreening* (e.g., $\hat{\mathrm{y}}(\cdot)$, EI$(\cdot)$) usada na etapa de identificação da solução a ser avaliada na função mérito. Tal função *prescreening* e a quantidade de soluções a serem

avalidas na função mérito são definidos de acordo com a estrutura do SAEA.

Algoritmo 3.13: Construção de metamodelo usando Sammon mapping

Entrada: $X \subset \mathbb{R}^n$, P, l
1 $X_r \cup P_r \leftarrow$ correspondente de $X \cup P$ em $\mathbb{R}^l$;
2 Constrói um metamodelo $\hat{y}(\cdot)$ a partir da amostra X_r e defina a função *prescreening* $\hat{f}(\cdot)$ associada;
3 $\tilde{x}_r \leftarrow$ melhor solução ao avaliar P_r em $\hat{f}(\cdot)$;
4 $\tilde{x} \leftarrow$ correspondente de $\tilde{x}_r$ em P;
Saída: $\tilde{x} \leftarrow$ solução a ser avaliada na função mérito

3.3 Conclusão

O capítulo corrente apresentou uma revisão sobre SAEAs, que são algoritmos evolutivos que empregam em sua estrutura o uso de metamodelos, visando reduzir o número de avaliações na função mérito. Notou-se nas estratégias abordadas o uso dos metamodelos *Ordinary Kriging* (incluindo aqui as variações DACE e Processo Gaussiano) e RBF, com reconstrução de metamodelo a medida que a população do algoritmo evolui. Observou-se também que uma forma interessante de tratar a acurácia de predição é manter uma relação de proximidade entre amostra desse metamodelo e a população corrente, com objetivo de assegurar que predições sejam realizadas em pontos que não se distanciam muito da região de aproximação. Além disso, notou-se estratégias em que a construção do metamodelo é realizada em um espaço de dimensão reduzida como uma forma de melhorar a qualidade da função de aproximação.

Considerando-se o exposto, observa-se que o metamodelo desempenha um papel fundamental na eficiência de um SAEA, não apenas no sentido de conduzir o EA para regiões promissoras e de reduzir o número de avaliações, mas também porque o custo de sua construção/atualização não pode ser proibitivo. Além disso, mesmo que a amostra não seja muito grande, o desafio enfrentado na estruturação de um SAEA é equilibrar a realização de um menor número de avaliações na função mérito com a necessidade de reconstrução do metamodelo em sucessivas iterações. Estratégias que empregam redução dimensional enfrentam o desafio adicional de equilibrar o tamanho da amostra e da população do EA com o tempo computacional envolvido no uso dessa técnica.

Embora o contexto deste trabalho esteja relacionado a SAEAs envolvendo SOOPs, é importante ressaltar que existem SAEAs desenvolvidos também para tratar Problemas de Otimização Multiobjetivo (MOP, *Multiobjective Optimization Problems*) computacionalmente caros (KNOWLES, 2006; LIM et al., 2009; ZHANG et al., 2009; EMMERICH; DEUTZ; KLINKENBERG, 2011; Pilát; Neruda, 2012; TABATABAEI et al., 2015; ALLMENDINGER et al., 2017; YANG et al., 2020). Da mesma forma que no contexto de SOOPs, a motivação para o uso de metamodelos em estratégias envolvendo MOPs é a redução do número de avaliação de funções associadas ao problema que exigem um alto

custo computacional. Enquanto que em SOOPs o metamodelo usualmente representa uma aproximação da função mérito, no contexto de MOPs existem estratégias que empregam uma função de aproximação para cada função objetivo (CHAFEKAR et al., 2005; Pilát; Neruda, 2012; BROWNLEE; WRIGHT, 2015), bem como outras que consideram construir um metamodelo para aproximar uma função definida por uma soma ponderada dos objetivos (KNOWLES, 2006; LIM et al., 2009). Outras abordagens para escolha do tipo de função que deve ser aproximada pelo metamodelo, a qual pode ser relacionada a definição do MOP ou ao algoritmo evolutivo adotado, também são relatadas na literatura (PILÁT; NERUDA, 2013; YUN; NAKAYAMA, 2013). Entretanto, aponta-se aqui apenas um direcionamento que pode ser seguido para um estudo a respeito dessas estratégias, uma vez que a proposta deste trabalho não envolve MOPs. Para o início de uma investigação detalhada sugere-se os trabalhos de (TABATABAEI et al., 2015; ALLMENDINGER et al., 2017), os quais apresentam uma revisão da literatura sobre diferentes abordagens de SAEAs envolvendo MOPs. Além disso, considera-se que a revisão apresentada neste capítulo pode auxiliar a compreender a estrutura de diferentes SAEAs desenvolvidos no contexto de MOPs.

4 Proposta de um SAEA Mono-Objetivo

4.1 Introdução

O capítulo corrente propõe um SAEA para otimização mono-objetivo, que acopla em um mesmo *framework* uma autoadaptação de parâmetro e um mecanismo que permite a escolha entre diferentes operadores de mutação. Especificamente, emprega-se um operador de mutação híbrido SL-PSO/DE que envolve diferentes variações desses operadores como uma forma de evitar uma convergência prematura e melhorar a capacidade do algoritmo de escapar de uma região de estagnação. Em relação ao metamodelo adotado, o SAEAa emprega um metamodelo *Ordinary Kriging* unidimensional, o que resulta em um menor custo computacional de construção/atualização de metamodelo em comparação com o uso de um metamodelo *Kriging* na sua forma usual. Este capítulo está organizado da seguinte forma: a Seção 4.2 apresenta uma visão geral do método proposto; as Seções 4.2.1 e 4.2.2 apresentam os operadores de mutação adotados; a Seção 4.2.3 descreve o processo de autoadaptação do operador de cruzamento adotado; a Seção 4.3 propõe um SAEA autoadaptativo aplicado ao contexto de otimização mono-objetivo; finalmente, na Seção 4.4 são apresentadas as considerações finais sobre o desenvolvimento deste capítulo.

4.2 SAEA autoadaptativo (SAEAa)

De modo geral, o SAEA autoadaptitvo, proposto neste capítulo e identificado como SAEAa, possui as seguintes características:

- É dedicado à otimização de problemas de alto custo computacional.
- Emprega um operador de mutação híbrido SL-PSO/DE para gerar novas soluções a cada iteração.
- Explora a informação sobre o *status* do processo de otimização. Essa informação é usada para guiar a escolha entre os diferentes operadores de mutação adotados.
- Considera um mecanismo de autoadaptação do parâmetro de cruzamento do operador DE. Esse processo de autoadaptação está relacionado com um histórico de melhora na função objetivo.
- Estabelece um limite para o uso de um mesmo operador por sucessivas iterações sem melhora. O objetivo desse critério é evitar uma convergência prematura e melhorar a capacidade do algoritmo de escapar de uma região de estagnação.

- Emprega um metamodelo *Ordinary Kriging* unidimensional, o que resulta em um menor custo computacional de construção/atualização de metamodelo.

Considerando que o SAEA autoadaptativo envolve o uso de operadores SL-PSO e DE, optou-se por descrever primeiro como são geradas novas soluções a partir desses operadores. A autoadaptação do parâmetro de cruzamento também é apresentada em uma seção anterior a descrição do SAEAa proposto na Seção 4.3.

4.2.1 Operador de Mutação SL-PSO

Esta seção apresenta detalhes de como o operador SL-PSO (SL-PSO, *Social Learning Particle Swarm Optimization*) é usado neste trabalho. O algoritmo SL-PSO, proposto em (CHENG; JIN, 2015), é baseado na ideia de aprendizado social somente entre os elementos da população corrente. No contexto de SAEAs obsverva-se uma variação do PSO entre os diferentes métodos investigados em (MA et al., 2015). Em (SUN et al., 2017) emprega-se um PSO e um SL-PSO como partes da estrutura de um SAEA. De um modo geral, SAEAs são estruturados na ideia de aproveitar-se de avaliações realizadas na função objetivo $\mathrm{y}(\cdot)$ para construir metamodelos que serão empregados para mapear regiões promissoras. Nesse sentido, é coerente explorar também informações dessas soluções no contexto de aprendizado social para gerar soluções que serão avaliadas no metamodelo.

Considerou-se neste trabalho empregar o operador SL-PSO, ilustrado no Algoritmo 4.1 e com respectivos parâmetros adotados em (CHENG; JIN, 2015), para auxiliar o metamodelo na identificação de regiões promissoras. Além disso, uma análise preliminar realizada com algumas funções analíticas na Seção 4.3.2 sugere que a combinação do operador SL-PSO com variações do operador DE contribui para uma convergência mais efetiva do algoritmo. O uso do operador SL-PSO está associado a um histórico de melhora na função objetivo, conforme detalhes na Seção 4.3. No Algoritmo 4.1, P é a população corrente, V armazena o vetor *correção de comportamento* $\mathbf{v}^i$ associado a cada solução $\mathbf{x}^i$ em P, N é o número de soluções, n é o número de variáveis e $\mathrm{y}(\cdot)$ é a função objetivo. Considerando P e V ordenados de acordo com a ordem decrescente de $\mathrm{y}(P)$, as novas soluções geradas a partir de P a cada iteração são obtidas conforme a seguir,

$$\tilde{v}_j^i = r_1 \cdot v_j^i + r_2 \cdot (x_j^l - x_j^i) + r_3 \cdot \epsilon \cdot (\bar{x}_j - x_j^i) \tag{4.1}$$

$$\epsilon = \beta \cdot \frac{n}{M} \tag{4.2}$$

$$\bar{x}_j = \frac{1}{N} \cdot \sum_{i=1}^{N} x_j^i \tag{4.3}$$

$$\tilde{x}_j^i = \begin{cases} x_j^i + \tilde{v}_j^i, & \text{se } p_i \leq \mathcal{P}(i) \\ x_j^i, & \text{caso contrário} \end{cases} \tag{4.4}$$

onde $i \in \{1, \ldots, N-1\}$, $j \in \{1, \ldots, n\}$ e $l \in \{i+1, \ldots, N\}$; r_1, r_2, r_3 são valores aleatórios em $[0, 1]$ atualizados a cada aplicação de (4.1); ϵ é o *fator de influência social*, $\beta = 0.01$ e $M = N - \left\lfloor \frac{n}{10} \right\rfloor$; $\bar{x}_j$ é a *componente de influência social*; $\mathcal{P}(i)$ é a *probabilidade de aprendizado* e p_i é um valor aleatório em $[0, 1]$. Para cada componete x_j^i da solução $\mathbf{x}^i$ escolhe-se uma solução aleatória $\mathbf{x}^l$ tal que $\mathrm{y}(\mathbf{x}^l) \geq \mathrm{y}(\mathbf{x}^i)$. Observe que, devido ao ordenamento de $\mathrm{y}(P)$, a pior solução $\mathbf{x}^1$ não é usada na *correção de comportamento* de nenhuma outra solução e as Equações (4.1)-(4.4) não são aplicadas para a melhor solução $\mathbf{x}^N$. Para cada solução $\mathbf{x}^i$ define-se uma probabilidade de aprendizagem conforme equação a seguir,

$$\mathcal{P}(i) = \left(1 - \frac{i-1}{N}\right)^{\alpha \cdot \log(\lceil \frac{n}{M} \rceil)} \tag{4.5}$$

onde $\alpha = 0.5$. Assim, cada componete x_j^i de $\mathbf{x}^i$ é modificada somente se $p_i \leq \mathcal{P}(i)$.

Algoritmo 4.1: Operador SL-PSO

Entrada: $P, \mathrm{y}(P), V, N, n$
$\tilde{P} \leftarrow \emptyset$;
$M \leftarrow N - \left\lfloor \frac{n}{10} \right\rfloor$;
$\alpha \leftarrow 0.5$, $\beta \leftarrow 0.01$;
$\epsilon \leftarrow \beta \times \frac{n}{M}$;
para cada $i = 1, \ldots, N$ **faça**
  $\mathcal{P}(i) \leftarrow$ equação (4.5);
fim
$\mathrm{y}(P) \leftarrow$ ordena $\mathrm{y}(P)$ em ordem decrescente;
$P, V, \mathcal{P} \leftarrow$ ordena P, V e $\mathcal{P}$ de acordo com o ordenamento em $\mathrm{y}(P)$;
para cada $i = 1, \ldots, N-1$ **faça**
  $p_i \leftarrow$ valor aleatório em $[0, 1]$;
  para cada $j = 1, \ldots, n$ **faça**
    $\mathbf{x}^l \leftarrow$ solução aleatória em P tal que $l \in \{i+1, \ldots, N\}$;
    $r_1, r_2, r_3 \leftarrow$valores aleatórios em $[0, 1]$;
    $\bar{x}_j \leftarrow$ equação (4.3);
    $\tilde{v}_j^i \leftarrow$ equação (4.1);
    $\tilde{x}_j^i \leftarrow$ equação (4.4);
  fim
  $\tilde{P} \leftarrow \tilde{P} \cup \{\tilde{\mathbf{x}}\}$;
fim
$\tilde{P} \leftarrow \tilde{P} \cup \{\mathbf{x}^N\}$;
$\mathrm{y}(P) \leftarrow$ ordena $\mathrm{y}(P)$ em ordem crescente;
$\tilde{P}, V \leftarrow$ ordena $\tilde{P}$ e V de acordo com o ordenamento em $\mathrm{y}(P)$;
Saída: $\tilde{P}$

4.2.2 Operadores de Mutação DE

Operadores DE (DE, *Differential Evolution*) (PRICE; STORN, 2005) são incorporados em diferentes formas no contexto de SAEAs. Observa-se em (LIU; ZHANG; GIELEN, 2014) o uso de um único operador, o DE/best/1, durante o processo iterativo. Em (YANG et al., 2019), um critério baseado na convergência do algoritmo determina qual dos operadores DE/best/1, DE/current-to-best/1 e current-to-randbest/1 deve ser adotado em cada iteração de um SAEA. Em (MA et al., 2015), o desempenho do SaDE (QIN; SUGANTHAN, 2005; DAS; SUGANTHAN, 2011) se destaca entre os diferentes EAs

incorporados em um SAEA. O SaDE (QIN; SUGANTHAN, 2005; DAS; SUGANTHAN, 2011) emprega uma autoadaptação do critério de escolha entre os operadores DE/rand/1 e DE/current-to-best/1. Outra característica do SaDE é a autoadaptação do parâmetro de cruzamento. No contexto desse trabalho, considerou-se no SAEAa o uso de diferentes

Algoritmo 4.2: Operadores DE

Entrada: $P, \mathbf{x}^{best}, C, N, n, N_f, N_c, N_{good}$
$\tilde{P} \leftarrow \emptyset$;
$N_{mod} \leftarrow 5$;
se $N_{good} = 0$ **então**
  se $N_f \geq N_c$ e $N_f < 2N_c$ **então**
    se $\mathrm{mod}(N_f, N_{mod}) \neq 0$ **então**
      `/* DE/current-to-randbest/1 */`
      $F \leftarrow 0.8$;
      $F_{rand} \leftarrow$ valor aleatório com distribuição uniforme em $[0, 1]$;
      para cada $i = 1, \ldots, N$ **faça**
        j_1, j_2 índices aleatórios em $\{1, \ldots, \alpha\} \setminus \{i\}$ tais que $j_1 \neq j_2$;
        $\tilde{\mathbf{x}} \leftarrow \mathbf{x}^i + F_{rand} \cdot (\mathbf{x}^{best} - \mathbf{x}^i) + F(\mathbf{x}^{j_1} - \mathbf{x}^{j_2})$;
        $\tilde{\mathbf{u}} \leftarrow$ executa o Algoritno 3.11;
        $\tilde{P} \leftarrow \tilde{P} \cup \{\mathbf{u}\}$;
      fim
    senão
      `/* DE/current-to-best/2 modificado */`
      $F \leftarrow 0.8$;
      $F_1, F_2 \leftarrow$ valores aleatórios com distribuição uniforme em $[0.4, 0.9]$;
      para cada $i = 1, \ldots, N$ **faça**
        j_1, j_2, j_3, j_4 índices aleatórios distintos em $\{1, \ldots, \alpha\} \setminus \{i\}$;
        $\tilde{\mathbf{x}} \leftarrow F \cdot (\mathbf{x}^{best} - \mathbf{x}^i) + F_1(\mathbf{x}^{j_1} - \mathbf{x}^{j_2}) + F_2(\mathbf{x}^{j_3} - \mathbf{x}^{j_4})$;
        $\tilde{\mathbf{u}} \leftarrow$ executa o Algoritno 3.11;
        $\tilde{P} \leftarrow \tilde{P} \cup \{\mathbf{u}\}$;
      fim
    fim
  senão
    `/* DE/rand/1 */`
    $F \leftarrow$valor aleatório com distribuição uniforme em $[0.8, 1]$;
    para cada $i = 1, \ldots, N$ **faça**
      j_1, j_2 índices aleatórios em $\{1, \ldots, \alpha\} \setminus \{i\}$ tais que $j_1 \neq j_2$;
      $\tilde{\mathbf{x}} \leftarrow \mathbf{x}^i + F(\mathbf{x}^{j_1} - \mathbf{x}^{j_2})$;
      $\tilde{\mathbf{u}} \leftarrow$ executa o Algoritmo 3.11;
      $\tilde{P} \leftarrow \tilde{P} \cup \{\mathbf{u}\}$;
    fim
  fim
senão
  `/* DE/current-to-best/1 */`
  $F \leftarrow 0.8$;
  para cada $i = 1, \ldots, N$ **faça**
    j_1, j_2 índices aleatórios em $\{1, \ldots, \alpha\} \setminus \{i\}$ tais que $j_1 \neq j_2$;
    $\tilde{\mathbf{x}} \leftarrow \mathbf{x}^i + F \cdot (\mathbf{x}^{best} - \mathbf{x}^i) + F(\mathbf{x}^{j_1} - \mathbf{x}^{j_2})$;
    $\tilde{\mathbf{u}} \leftarrow$ executa o Algoritmo 3.11;
    $\tilde{P} \leftarrow \tilde{P} \cup \{\mathbf{u}\}$;
  fim
fim
Saída: $\tilde{P}$

operadores DE conforme ilustra o Algoritmo 4.2. O critério para a escolha do operador está associado a melhora na função objetivo. Considerando P a população corrente, identifica-se as seguintes informações associadas: melhor solução $\mathbf{x}^{best}$; vetor C com o valor do parâmetro de cruzamento de cada solução; N e n, o número de soluções e variáveis, respectivamente. As quantidades N_f, N_c e N_{good} representam, respectivamente, contador

do número de iterações consecutivas sem melhora, número máximo permitido de iterações consecutivas sem melhora e indicador de melhora no valor de função objetivo. Se não há melhora ($N_{good} = 0$) e $N_c \leq N_f < 2N_c$, a escolha é entre os operadores DE/curent-to-randbest/1 (YANG et al., 2019) e DE/current-to-best/2 modificado. O uso desses dois operadores de forma alternada tem como finalidade identificar regiões promissoras. Note que em DE/current-to-randbest/1, o vetor mutação recebe maior influência do vetor aleatório do que de $\mathbf{x}^{best}$, dependendo do valor de F_{rand}. Por outro lado, no caso de uma população com pouca diversidade, o vetor mutação recebe mais informação do vetor base $\mathbf{x}^i$. Nesse sentido, para contribuir com a busca realizada nessa etapa, a cada N_{mod} iterações emprega-se o operador DE/current-to-best/2 modificado. Essa variação permite afastar-se de $\mathbf{x}^{best}$ e depois seguir em uma direção aleatória. Quando esses operadores não resultam em melhora e se $N_f \geq 2N_c$, então aplica-se o DE/rand/1 considerando como prioridade buscar diversidade e não usar informações de $\mathbf{x}^{best}$. O operador DE/current-best/1 tem como finalidade direcionar a busca para a região indicada por $\mathbf{x}^{best}$ sem perder diversidade.

4.2.3 Adaptação do parâmetro de cruzamento C

O mecanismo de autoadaptação do parâmetro de cruzamento C empregado neste trabalho é uma modificação da abordagem adotada para atualizar este parâmetro no algoritmo SaNSDE proposto em (YANG; TANG; YAO, 2008), no qual o processo de autoadaptação está relacionado com a melhora na função objetivo. Diferentes trabalhos exploram formas de autoadaptação de parâmetros do DE (QIN; SUGANTHAN, 2005; YANG; TANG; YAO, 2008; DICK, 2010), nos quais a ideia fundamental é aproveitar informações sobre o processo evolutivo para identificar uma configuração adequada dos parâmetros. De um modo geral, essas variações do DE permitem tornar o algoritmo mais adequado às características do problema.

De certa forma, SAEAs compartilham dessa ideia de aproveitar informações obtidas durante o processo evolutivo. Especificamente, nessas estratégias aproveita-se avaliações realizadas na função objetivo $\mathrm{y}(\cdot)$ para mapear regiões promissoras. Considerando que o método proposto emprega diferentes operadores de mutação do DE, é coerente então pensar em um mecanismo de autoadaptação de C para o contexto deste trabalho. Nesse sentido, cada solução da população corrente possui um valor do parâmetro de cruzamento associado, o qual é atualizado durante o processo iterativo. Para cada $i \in \{1, \ldots, N\}$, define-se $C(i)$ da seguinte forma:

- $C(i) = 0.8$ para cada solução $\mathbf{x}^i$ da população inicial, permanecendo esse valor até um determinado número de iterações N_g;
- $C(i)$ é um valor aleatório tal que

$$C(i) \sim \mathcal{N}(CR_m, 0.1) \tag{4.6}$$

para cada solução $\mathbf{x}^i$ das populações seguintes.

A cada iteração g coleta-se o seguinte percentual de melhora

$$\Delta_{\%}^{g} = \frac{\left|\mathrm{y}(\mathbf{x}^{\star}) - \min(\mathrm{y}(\mathbf{x}^{\star}), \mathrm{y}(\tilde{\mathbf{x}}))\right|}{|\mathrm{y}(\mathbf{x}^{\star})|} \cdot 100 \tag{4.7}$$

onde $\Delta_{\%}^{g}$ é definido no intervalo $[0, 100]$; $\mathrm{y}(\mathbf{x}^{\star})$ é o valor de $\mathrm{y}(\cdot)$ da melhor solução na iteração $(g-1)$ e $\mathrm{y}(\tilde{\mathbf{x}})$ é a nova solução avaliada na iteração g. Outra informação identificada é o valor C^g, que corresponde ao parâmetro de cruzamento associado a $\tilde{\mathbf{x}}$. Os valores $\Delta_{\%}^{g}$ e C^g são armazenados, respectivamente, em Δ_{rec} e CR_{rec}; note que Δ_{rec} e CR_{rec} possuem a mesma cardinalidade N_{rec}. Essas informações são utilizadas para atualizar a média CR_m, conforme a seguir

$$CR_m = \sum_{k=1}^{N_{rec}} \omega(k) CR_{rec}(k) \tag{4.8}$$

$$\omega(k) = \frac{\Delta_{rec}(k)}{\sum_{k=1}^{N_{rec}} \Delta_{rec}(k)} \tag{4.9}$$

Inicialmente define-se $CR_m = 0.5$, em seguida esse parâmetro é atualizado a cada iteração de acordo com (4.8), usando as informações armazenadas em Δ_{rec} e CR_{rec} durante um período de N_{mod1} iterações. Esses conjuntos são definidos como vazio antes de armazenar as informações das próximas N_{mod1} iterações. A principal distinção entre o mecanismo de autoadaptação proposto aqui (4.7)-(4.9) e o método em (YANG; TANG; YAO, 2008) está na forma de medir a melhora na função objetivo. Optou-se por adotar aqui o percentual de melhora (4.7), o qual permite eliminar a dependência da autoadaptação em relação a magnitude do valor de função objetivo observada no método em (YANG; TANG; YAO, 2008). O Algoritmo 4.3 ilustra o mecanismo de autoadaptação do parâmetro de cruzamento. Nesse algoritmo, C mantém-se inalterado por um período de N_g iterações, passando a ser atualizado a cada iteração de acordo com (4.6). A atualização de CR_m considera a contribuição obtida em (4.7) e armazenada em Δ_{rec}. O parâmetro N_{mod1} define o período de armazenamento de informações em Δ_{rec} e CR_{rec}. Note que, durante um ciclo N_{mod1}, pode ocorrer de (4.7) ser igual a zero em cada iteração g nesse ciclo, resultando em (4.9) indefinida. Nesse caso, define-se a média em (4.6) como o CR_m corrente. É importante mencionar que não é investigado neste trabalho o efeito de diferentes valores para os parâmetros associados ao Algoritmo 4.3 em função do número de variávies. Para estabelecer os valores de tais parâmetros é considerado como referência a estrutura do SAEAa, ilustrado no Algoritmo 4.4, e valores adotados nos trabalhos (YANG; TANG; YAO, 2008; YANG et al., 2019). Nesse sentido, a escolha do parâmetro $C = 0.8$ fixo nas N_g primeiras iterações prioriza a busca na região onde se encontra a melhor solução da população corrente quando o operador DE/current-to-best/1 for utilizado; no TASEA esse operador é aplicado com esse mesmo valor do parâmetro C. No Algoritmo 4.3, os valores

estabelecidos para N_g nas linhas 2 e 6 considera que em (YANG; TANG; YAO, 2008) o parâmetro C permanece fixo nas 5 primeiras iterações para problemas com número de variáveis $n = 30$ e que para o SAEAa considera-se a validação do método em problemas analíticos envolvendo de 10 a 200 variáveis. O critério $N_{mod1} = 40$ iterações para atualizar CR_m considera que (YANG; TANG; YAO, 2008) define $CR_m = 0.5$ e realiza a atualização desse parâmetro a cada 25 iterações. Como no SAEAa não é em toda iteração que se aplica um operador DE, considerou-se aumentar esse intervalo N_{mod1} de 25 para 40 iterações para ter mais informações de quando os operadores DE são usados.

Algoritmo 4.3: Autoadaptação do Parâmetro de Cruzamento

Entrada: $C, CR_m, \Delta_{rec}, CR_{rec}, N, n, g$
1 **se** $n \leq 100$ **então**
2 | $N_g \leftarrow 5$;
3 **senão**
4 | $N_g \leftarrow 10$;
5 **fim**
6 $N_{mod1} \leftarrow 40$;
7 **se** $g \leq N_g$ **então**
8 | $C \leftarrow C$;
9 **senão**
10 | $\omega \leftarrow$ equação (4.9);
11 | $CR_m \leftarrow$ equação (4.8);
12 | **para cada** $i = 1, \ldots, N$ **faça**
13 | | $C(i) \leftarrow$ valor aleatório tal que $C(i) \sim \mathcal{N}(CR_m, 0.1)$;
14 | **fim**
15 | **se** $\mathrm{mod}(g, N_{mod1}) = 0$ **então**
16 | | $\Delta_{rec} \leftarrow \emptyset$;
17 | | $CR_{rec} \leftarrow \emptyset$;
18 | **fim**
19 **fim**
Saída: $C, CR_m, \Delta_{rec}, CR_{rec}$

4.3 Método Proposto (SAEAa)

Nesta seção é proposto um SAEA para otimização mono-objetivo, identificado como SAEAa, que acopla em um mesmo *framework* uma autoadaptação de parâmetro e um mecanismo que permite a escolha entre diferentes operadores de mutação. O método proposto, ilustrado no Algoritmo 4.4, emprega um metamodelo *Ordinary Kriging* (OK) na avaliação de soluções geradas a partir de operadores de mutação SL-PSO e DE. As Seções 4.2.1 e 4.2.2 apresentam detalhes de como esses operadores são usados neste trabalho. No Algortimo 4.4, a função $\mathrm{y}(\cdot)$ é a função objetivo associada a um problema mono-objetivo (1.1) com região factível Ω; as quantidades N_s, N, N_m e n representam, respectivamente, o número de observações iniciais realizadas em $\mathrm{y}(\cdot)$, número de soluções da população a ser evoluída, número de soluções da amostra usada para a construção do metamodelo e número de variáveis. Os parâmetros N_c e k representam, respectivamente, o número máximo de iterações consecutivas sem melhora na função objetivo e o número de soluções mapeadas pela função *prescreening* $\mathrm{EI}(\cdot)$. Essa quantidade N_c estabelece quando alternar entre o operador SL-PSO e variações do DE. A população corrente e o conjunto

de armazenamento são representados, respectivamente, por P e DB. Nesses conjuntos, cada solução está associada ao seu valor de avaliação em $\mathrm{y}(\cdot)$. Além disso, cada solução de P está associada também a um vetor correção de comportamento e a um valor de parâmetro de cruzamento. Nesse sentido, V e C armazenam, respectivamente, o vetor correção de comportamento $\mathbf{x}_{bc}^{i}$ e o valor de parâmetro de cruzamento $C(i)$ de cada solução $\mathbf{x}^{i}$ da população corrente P.

Algoritmo 4.4: SAEA autoadaptativo - SAEAa

Entrada: $\Omega, \mathrm{y}(\cdot), N_s, N, N_m, n, N_c, k$
Gera N_s soluções em Ω e avalia em $\mathrm{y}(\cdot)$;
Atualiza DB;
$P \leftarrow N$ melhores soluções em DB;
$V \leftarrow$ matriz nula $N \times n$;
$C \leftarrow$ matriz $N \times 1$ com elementos 0.8;
$\mathbf{x}^{\star} \leftarrow$ melhor solução entre as avaliações realizadas em $\mathrm{y}(\cdot)$;
$CR_m \leftarrow 0.5$, $\Delta_{rec} \leftarrow$ equação (4.10), $CR_{rec} \leftarrow C$;
$N_f \leftarrow 1$, $N_{good} \leftarrow 0$, $g \leftarrow 1$;
enquanto *critério de parada não é satisfeito* **faça**
    $\mathbf{x}^{best}$ melhor solução em P;
    se $N_{good} = 0$ **então**
        se $N_f < N_c$ **então**
            $\tilde{P}, \tilde{V} \leftarrow$ executa o Algoritmo 4.1; `/* SL-PSO */`
        senão
            $\tilde{P} \leftarrow$ executa o Algoritmo 4.2; `/* DE/current-to-ranbest/1 */`
            $\tilde{V} \leftarrow V$; `/* ou */`
            `/* DE/current-to-best/2 modificado */`
        fim
    senão
        $\tilde{P} \leftarrow$ executa o Algoritmo 4.2; `/* DE/current-to-best/1 */`
        $\tilde{V} \leftarrow V$;
    fim
    $X \leftarrow$ conjunto com as N_m soluções mais recentes em DB;
    Construa um metamodelo OK usando X e retorna $\mathrm{EI}(\cdot)$;
    $\tilde{\mathbf{x}}, \tilde{c} \leftarrow$ executa o Algoritmo 4.5;
    $CR_{rec} \leftarrow CR_{rec} \cup \{\tilde{c}\}$;
    Avalia $\tilde{\mathbf{x}}$ em $\mathrm{y}(\cdot)$;
    Atualiza DB;
    se $\mathrm{y}(\tilde{\mathbf{x}}) < \mathrm{y}(\mathbf{x}^{\star})$ **então**
        $N_f \leftarrow 1$, $good \leftarrow 1$;
        $\mathbf{x}^{\star} \leftarrow \tilde{\mathbf{x}}$;
    senão
        $N_f \leftarrow N_f + 1$, $good \leftarrow 0$;
    fim
    $\Delta_{\%}^{g} \leftarrow$ equação (4.7);
    $\Delta_{rec} \leftarrow \Delta_{rec} \cup \{\Delta_{\%}^{g}\}$;
    $C, CR_m, \Delta_{rec}, CR_{rec} \leftarrow$ executa o Algoritmo 4.3;
    $P \leftarrow$ conjunto com N melhores soluções de DB;
    $V \leftarrow \tilde{V}$;
    $g \leftarrow g + 1$;
fim
Saída: $\mathbf{x}^{\star} \leftarrow$ melhor solução entre as avaliações realizadas em $\mathrm{y}(\cdot)$

Algoritmo 4.5: Seleciona Solução

Entrada: $\mathrm{EI}(\cdot), \tilde{P}, C, k$
$\tilde{\mathrm{X}} \leftarrow$ conjunto com as k melhores soluções ao avaliar $\tilde{P}$ com a função $\mathrm{EI}(\cdot)$;
$\tilde{\mathbf{x}} \leftarrow$ solução aleatória em $\tilde{\mathrm{X}}$;
$\tilde{c} \leftarrow$ correspondente de $\tilde{\mathbf{x}}$ em C;
Saída: $\tilde{\mathbf{x}}, \tilde{c}$

Em algoritmos do tipo SAEA, a convergência é influenciada pelos operadores do EA usado como base e pela capacidade do metamodelo em mapear soluções promissoras; a Seção 4.3.1 aborda essa última questão. Com relação aos operadores, o SAEA autoadaptativo explora os benefícios apresentados em (YANG et al., 2019) com o uso de diferentes operadores DE. A distinção aqui é que o método proposto emprega um operador híbrido SL-PSO/DE. Outra ideia adotada do referido artigo é usufruir de informações sobre a convegergência do algoritmo. Nesse sentido, os parâmetros N_c e N_{good} estabelecem quando gerar novas soluções com o operador SL-PSO. Se há melhora na função objetivo, indicada por N_{good}, emprega-se o operador DE/current-to-best/1. A ideia é direcionar a busca na região onde se encontra a melhor solução da população sem perder diversidade. Quando não há melhora na função objetivo, a escolha entre os operadores SL-PSO e DE no passo 13 é deterninada por N_c, o qual representa o número máximo permitido de iterações consecutivas sem melhora. Assim, usa-se o operador SL-PSO quando $N_f < N_c$, onde N_f representa um ciclo de iterações sem melhora atualizado conforme os passos 29 e 32. O algoritmo SL-PSO é baseado na ideia de aprendizado social somente entre os elementos da população corrente. Considerando que no SAEAa a população é definida como as N melhores soluções, faz sentido então aplicar essa ideia de aprendizado social no contexto deste trabalho. Além disso, observou-se o uso do SL-PSO no SAEA proposto em (SUN et al., 2017). O caso em que $N_f \geq N_c$ sugere que o operador SL-PSO não contribui com a exploração da região de convergência. Por outro lado, o uso de operadores DE baseado na melhor solução $\mathbf{x}^{best}$ para explorar essa região pode resultar em resultados promissores. Mais detalhes dessa etapa do Algoritmo 4.4 são abordados a seguir.

Como pode ser observado, o método proposto estabelece um limite para o uso de um mesmo operador por sucessivas iterações sem melhora. A ideia é evitar uma convergência prematura e melhorar a capacidade do algoritmo de escapar de uma região de estagnação. Nesse sentido, para os operadores DE aplicados nos passos 14-17, considerou-se como critério $N_c \leq N_f < 2N_c$, conforme ilustra o Algoritmo 4.2. Durante esse período são usadas variações do operador DE/current-to-best para identificar regiões promissoras. No Algoritmo 4.2, o parâmetro N_{mod} estabelece quando aplicar o operador DE/current-to-randbest/1 proposto em (YANG et al., 2019) ou o operador DE/current-to-best/2 modificado. Quando esses operadores não resultam em melhora, isto é, $N_f \geq 2N_c$ aplica-se o operador DE/rand/1 como uma tentativa de reestabelecer a diversidade e retoma-se a busca com o operador SL-PSO.

Conforme abordado na Seção 4.2.3, o SAEA autoadaptativo também usa informações sobre o processo de convergênica para atualizar o valor do parâmetro de cruzamento. Nesse sentido, a primeira atualização de Δ_{rec} armazena o valor $\Delta^1_{\%}(i)$ associado a cada solução $\mathbf{x}^i$ da população inicial, conforme a adaptação de (4.7) a seguir,

$$\Delta^1_{\%}(i) = \frac{\left|\mathrm{y}(\mathbf{x}^\star) - \min(\mathrm{y}(\mathbf{x}^\star), \mathrm{y}(\mathbf{x}^i))\right|}{|\mathrm{y}(\mathbf{x}^\star)|} \cdot 100 \tag{4.10}$$

onde $\mathbf{x}^\star$ é a melhor solução entre as avaliações realizadas em y$(\cdot)$ na primeira iteração. O conjunto CR_{rec} é atualizado inicialmente como $CR_{rec} = C$, com C definido no passo 5. Nas iterações seguintes, Δ_{rec} e CR_{rec} armazenam, respectivamente, o valor da medida de melhora $\Delta^g_{\%}$ e o valor do parâmetro de cruzamento $\tilde{c}$ associados a solução $\tilde{\mathbf{x}}$ avaliada em y$(\cdot)$ na iteração g. A atualização de C e CR_m no passo 36 considera, conforme ilustra o Algoritmo 4.3, somente informações armazenas em Δ_{rec} e CR_{rec} durante um ciclo de $N_{mod1} = 40$ iterações.

Com relação a construção de metamodelo no passo 23, o SAEAa emprega um metamodelo OK considerando a igualdade das compomentes do parâmetro $\boldsymbol{\theta}$ em (2.25), seguindo a mesma ideia adotada em (CAI; LI, 2019). Nesse sentido, o problema em (2.29) se reduz a uma otimização unidimensional do parâmetro $\boldsymbol{\theta}$, uma vez que, no contexto deste trabalho, define-se o OK com $p_r = 2$, $\forall r = 1, \ldots, n.$. Da mesma forma que (CAI; LI, 2019), o SAEAa emprega o algoritmo DIRECT (FINKEL, 2003) na otimização do parâmetro $\boldsymbol{\theta}$. Note que o custo computacional de um modelo *Kriging* com essa abordagem é menor em comparação com a construção de um modelo *Kriging* com $\boldsymbol{\theta}$ $n-$dimensional.

No SAEA autoadaptativo avalia-se em y$(\cdot)$ apenas uma solução $\tilde{\mathbf{x}}$ por iteração, a qual corresponde a uma solução aleatória entre as k melhores soluções avaliadas em EI$(\cdot)$. Devido ao erro associado a predição, pode ocorrer de a melhor solução mapeada via EI$(\cdot)$ não necessariamente corresponder a solução com melhor valor de y$(\cdot)$. Por outro lado, a melhor solução avaliada em y$(\cdot)$ pode estar entre as k melhores soluções identificadas por EI$(\cdot)$. Nesse sentido, é interessante considerar uma quantidade k de soluções com melhor valor EI$(\cdot)$ como opções para essa escolha aleatória. É importante mencionar que (YANG et al., 2019) investigou, em algumas funções analíticas, a convergência de um SAEA considerando diferentes valores para k. Dessa forma, o valor considerado adequado em (YANG et al., 2019) e adotado aqui é $k = 3$. Entretanto, observa-se que há a necessidade de uma análise detalhada relacionada a escolha de k para o SAEAa mas isto não é considerado neste trabalho. Uma vez identificada $\tilde{\mathbf{x}}$, essa solução é avaliada em y$(\cdot)$ no passo 26 do Algoritmo 4.4 e as etapas seguintes correspondem às atualizações exigidas dos Algortimos 4.1, 4.2 e 4.3. Note que a melhora de função objetivo que se espera alcançar com essa avaliação realizada no passo 26 está relacionada com a capacidade de EI$(\cdot)$ em mapear soluções promissoras. Nesse sentido, a Seção 4.3.1 aborda questões relacionadas a construção de metamodelo.

4.3.1 Considerações sobre a escolha e construção de metamodelo

Esta seção aborda aspectos que influenciam a capacidade do metamodelo em conduzir um EA para regiões promissoras. Essa capacidade pode ser comprometida ao analisar o metamodelo em soluções que estão muito distantes da região de aproximação. Nesse sentido, nos SAEAs de um modo geral, a amostra do metamodelo acompanha a

evolução da população. No caso do SAEAa, a forma de definir a população corrente e de escolher a amostra caracteriza o metamodelo como uma aproximação local na região da população corrente. Isso permite estabelecer uma relação de proximidade entre a amostra e as soluções avaliadas no metamodelo. Os tópicos a seguir abordam mais detalhes relacionados a escolha e construção de metamodelo, os quais são tratados também no trabalho de (VALADãO; BATISTA, 2020) elaborado em paralelo ao desenvolvimento deste texto.

4.3.1.1 Escolha do metamodelo

A escolha do *Ordinary Kriging* como metamodelo adotado no SAEAa fundamenta-se no trabalho de (VALADãO; BATISTA, 2020), o qual apresenta um estudo investigativo que compara o desempenho de variações do modelo *Kriging* e RBF quando incorporados em um SAEA *framework*. Observa-se que diferentes SAEAs relatados no Capítulo 3 não exploram variações do modelo *Kriging*. Usualmente tais estratégias empregam metamodelos OK ou RBF. Entretanto, em muitos SAEAs a escolha do metamodelo é baseada em estudos desenvolvidos no contexto do método EGO, no qual as etapas de construção/atualização de metamodelo priorizam também a qualidade da aproximação global. Além disso, nota-se também que não há um estudo comparando o desempenho de metamomdelos, em termos de qualidade de solução e custo computacional, no contexto de SAEAs. Nesse sentido, no trabalho em questão considera-se um SAEA com uma estrutura simples para permitir visualizar o desempenho dos metamodelos *Ordinary Kriging* (OK), *First-order Universal Kriging* (UK1), *Second-order Universal Kriging* (UK2), *Blind Kriging* (BK) e RBF.

A partir dos resultados reportados em (VALADãO; BATISTA, 2020) observa-se que o OK apresenta melhor desempenho em relação aos demais métodos analisados. Entretanto, nota-se que não existe diferença estatisticamente significativa do OK em relação aos metamodelos OK1, OK2 e BK quanto a qualidade de solução retornada. Em relação ao custo computacional, nota-se que o BK exige um maior tempo de construção/atualização de metamodelo. Por outro lado, observa-se que o RBF é destacado como uma alternativa interessante por apresentar o menor custo computacional entre demais metamodelos. Além disso, a diferença do RBF em relação aos outros quanto a qualidade de solução pode não ter um significado prático quando considerado em problemas de aplicação. Embora em (VALADãO; BATISTA, 2020) seja considerado somente a aplicação em problemas analíticos envolvendo entre 5 e 20 variáveis, o estudo em questão permite justificar a escolha dos metamodelos OK e RBF no contexto deste trabalho. Conforme pode ser obeservado no Capítulo 5, considera-se também investigar o desempenho do RBF na abordagem proposta e nos SAEAs adotados para a comparação com o SAEAa.

4.3.1.2 Escolha da amostra e atualização de DB

O processo de atualização de DB, ilustrado no Algortimo 4.6, tem como finalidade armazenar soluções que sejam relevantes para a construção de um metamodelo no passo 23 do Algoritmo 4.4. Além disso, as soluções da população corrente também são selecionadas desse conjunto. Dessa forma, a diversidade da população corrente também é influenciada pela forma de atualização desse conjunto. No SAEAa, a amostra de metamodelo corresponde a uma quantidade de soluções mais recentemente armazenadas em DB. A ideia é assegurar que, no passo 24 do Algoritmo 4.4, as avaliações realizadas em $\mathrm{EI}(\cdot)$ sejam realizadas em soluções que não estejam muito distantes da região de aproximação. Conforme observado na Seção 2.6.1, considerando $\mathbf{x} \notin X$, X amostra de metamodelo, $\mathrm{EI}(\mathbf{x})$ assumirá um valor alto se $\mathbf{x}$ estiver distante da amostra ou se $\hat{\mathrm{y}}(\mathbf{x}) \leq \mathrm{y}_{min}$, onde $\mathrm{y}_{min} = \min_{\tilde{\mathbf{x}} \in X} \mathrm{y}(\tilde{\mathbf{x}})$. No contexto do SAEAa é interessante conseguir mapear via $\mathrm{EI}(\cdot)$ uma solução que represente melhora em relação ao valor mínimo da amostra corrente.

Note que no SAEAa a amostra de metamodelo é definida como as soluções avaliadas mais recentemente em $\mathrm{y}(\cdot)$. Por outro lado, considerar somente essa condição na atualização

Algoritmo 4.6: Atualização de DB

Entrada: $DB, \mathrm{y}(DB), \tilde{\mathbf{x}}, \mathrm{y}(\tilde{\mathbf{x}}), N_{DB}$
1 $d_{tol} \leftarrow 10^{-6}$;
2 $N_{db} \leftarrow$ número de soluções em DB;
3 $\hat{\mathbf{x}} \leftarrow \underset{\mathbf{x} \in DB}{\mathrm{argmin}} ||\mathbf{x} - \tilde{\mathbf{x}}||$;
4 $d \leftarrow ||\hat{\mathbf{x}} - \tilde{\mathbf{x}}||$;
5 $\mathrm{y}(\hat{\mathbf{x}}) \leftarrow$ correspondente de $\hat{\mathbf{x}}$ em $\mathrm{y}(DB)$;
6 **se** $d \leq d_{tol}$ **então**
7 $\quad \hat{\mathbf{x}} \leftarrow \tilde{\mathbf{x}}$;
8 $\quad \mathrm{y}(\hat{\mathbf{x}}) \leftarrow \mathrm{y}(\tilde{\mathbf{x}})$;
9 **senão**
10 $\quad$ **se** $N_{db} < N_{DB}$ **então**
11 $\quad\quad DB \leftarrow DB \cup \{\tilde{\mathbf{x}}\}$;
12 $\quad\quad \mathrm{y}(DB) \leftarrow \mathrm{y}(DB) \cup \{\mathrm{y}(\tilde{\mathbf{x}})\}$;
13 $\quad$ **senão**
14 $\quad\quad \check{\mathbf{x}} \leftarrow$ solução mais antiga em DB;
15 $\quad\quad \mathrm{y}(\check{\mathbf{x}}) \leftarrow$ correspondente de $\check{\mathbf{x}}$ em $\mathrm{y}(DB)$;
16 $\quad\quad \check{\mathbf{x}} \leftarrow \tilde{\mathbf{x}}$;
17 $\quad\quad \mathrm{y}(\check{\mathbf{x}}) \leftarrow \mathrm{y}(\tilde{\mathbf{x}})$;
18 $\quad$ **fim**
19 **fim**
Saída: $DB, \mathrm{y}(DB)$

de DB pode resultar no armazenamento de soluções redundantes nesse conjunto. A consequência disso é a seleção de uma amostra que não é adequada para a construção de metamodelo. Assim, na atualização de DB considera-se que, a cada iteração, a construção de metamodelo é realizada com uma amostra normalizada de acordo com

$$\tilde{X} = \frac{X - \mu(X)}{\sigma(X)} \quad \text{e} \quad \mathrm{y}(\tilde{X}) = \frac{\mathrm{y}(X) - \mu(\mathrm{y}(X))}{\sigma(\mathrm{y}(X))} \tag{4.11}$$

onde $X \subset DB$, $\mu(\cdot)$ e $\sigma(\cdot)$ representam, respectivamente, a média e desvio padrão; cada direção coordenada em $\tilde{X}$ deve ter média 0 e variância 1. Essa atualização de DB deve

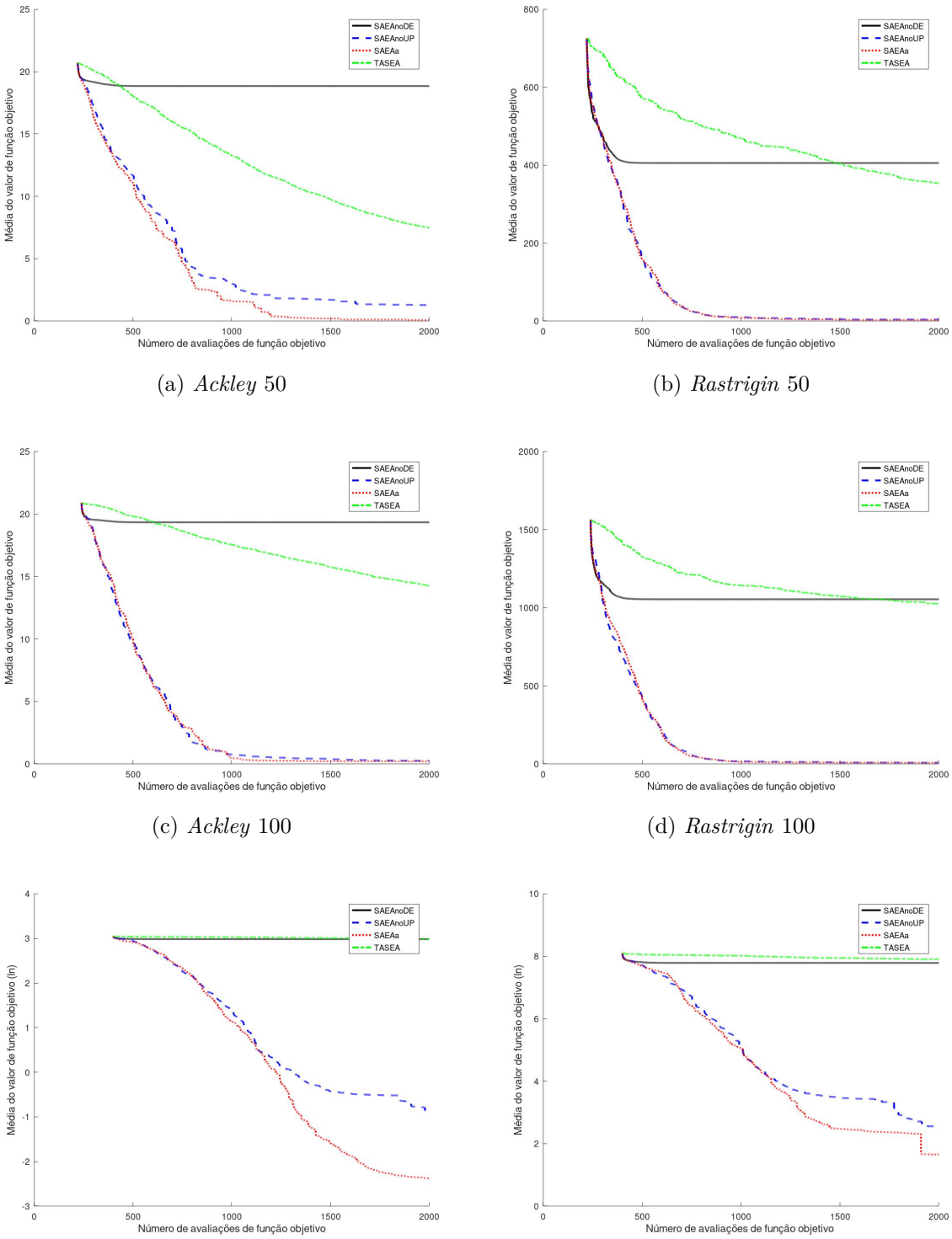
ser realizada de tal forma que as variâncias de X e $\mathrm{y}(X)$ não sejam nulas. De acordo com (COUCKUYT et al., 2012), a normalização em (4.11) melhora a qualidade do modelo *Kriging*.

Para atender as condições estabelecidas para a seleção da amostra, considera-se no Algoritmo 4.6 a distância Euclidiana entre $\tilde{\mathbf{x}}$ e cada solução $\mathbf{x}$ em DB. Se a distância d da solução em DB mais próxima a $\tilde{\mathbf{x}}$ é menor que 10^{-6}, então substitui-se essa solução mais próxima por $\tilde{\mathbf{x}}$. No caso em que $d \geq 10^{-6}$, o critério $N_{db} < N_{DB}$ determina se acrescenta $\tilde{\mathbf{x}}$ em DB ou se substitui a solução mais antiga armazenada em DB, onde N_{db} é o número de soluções em DB. Um detalhe importante é que a solução $\tilde{\mathbf{x}}$ é inserida em DB, de acordo com os passos 6-17, somente se (4.11) estiver definida para $DB \cup \{\tilde{\mathbf{x}}\}$ e $\mathrm{y}(DB) \cup \{\mathrm{y}(\tilde{\mathbf{x}})\}$. Especificamente, se $\sigma(DB \cup \{\tilde{\mathbf{x}}\}) \leq 10^{-6}$ ou $\sigma(\mathrm{y}(DB) \cup \{\mathrm{y}(\tilde{\mathbf{x}})\}) \leq 10^{-6}$, então $\tilde{\mathbf{x}}$ e $\mathrm{y}(\tilde{\mathbf{x}})$ não são inseridos em DB e em $\mathrm{y}(DB)$, respectivamente.

É importante observar que, embora DB satisfaça as condições em (4.11), pode ocorrer dessas condições não serem satisfeitas para qualquer subconjunto de DB. A consequência disso é a construção de metamodelo considerando como amostra todo o conjunto DB, se ocorrer de o conjunto com as N_m soluções mais recentes em DB não atender (4.11). Para tratar essa questão, a cada iteração armazena-se a amostra selecionada para o metamodelo. Se na iteração g observar-se que é necessário uma amostra com cardinalidade maior que N_m^{max}, $N_m < N_m^{max} \leq N_{DB}$, então a partir da iteração g define-se como amostra de metamodelo a amostra usada na iteração $(g-1)$. A medida que o SAEAa converge, tem-se soluções cada vez mais próximas em DB. Nesse sentido, na seleção da amostra de metamodelo deve-se considerar o condicionamente da matriz de correlação em (2.25), a qual é definida em função da distância Euclidiana entre soluções da amostra. Considerou-se nesse trabalho $N_m^{max} = 120$.

4.3.2 Efeito dos operadores de mutação e atualização de C

Esta seção apresenta uma análise gráfica, para algumas funções analíticas, sobre o efeito dos operadores de mutação e da atualização do parâmetro C na estratégia proposta. Para essa análise considerou-se as configurações SAEAa somente com o operador SL-PSO e SAEAa com SL-PSO e DE adotando parâmetro fixo C, identificadas como SAEAnoDE e SAEAnoUP, respectivamente. Na configuração SAEAnoUP considerou-se $C = 0.2$ para o DE/current-to-best/1, $C = 0.6$ para DE/current-to-randbest/1 e $C = 0.8$ para DE/rand/1. Considerou-se também nesta comparação o método TASEA (YANG et al., 2019), que representa um dos SAEAs mais recentes da literatura. Nesse sentido, a Figura 3 ilustra a convergência das configurações SAEAnoDE, SAEAnoUP e SAEAa em contraste com o método TASEA. Para essa análise empregou-se duas funções analíticas, *Ackley* e *Rastrigin* da Tabela 1, com a variação $n = 50, 100$ e 200, onde n é o número de variáveis. Para cada problema (funções *Ackley* e *Rastrigin* com $n \in \{50, 100, 200\}$) executou-se 20 testes com



(a) *Ackley* 50

(b) *Rastrigin* 50

(c) *Ackley* 100

(d) *Rastrigin* 100

(e) *Ackley* 200

(f) *Rastrigin* 200

Figura 3 – Curvas de convergência dos métodos SAEAnoDE, SAEAnoUP e SAEAa e TASEA.

cada uma das abordagens. Note que os métodos SAEAnoUP e SAEAa se destacam em relação ao SAEAnoDE e ao TASEA em todos os casos considerados. Observa-se que usar somente o operador SL-PSO não contribui para a convegência do algoritmo. As curvas de

convergência em (b), (c) e (d) sugerem um comportamento semelhante entre o SAEAnoUP e SAEAa. Por outro lado, observa-se um melhor desempenho para o SAEAa nas curvas de convergência em (a), (e) e (f). Observa-se que a combinação do operador SL-PSO com variações de operadores DE resultou em uma convergência mais efetiva do algoritmo. Nos casos com maior número de variáveis, a atualização do parâmetro C contribuiu para a qualidade da solução. A Figura 3 sugere que o SAEAa é um método promissor no contexto de SAEAs. Nesse sentido, o Capítulo 5 apresenta o planejamento experimental adotado para validar a estratégia proposta neste capítulo.

4.4 Conclusão

Este capítulo apresentou a proposta de um SAEA aplicado ao contexto de problemas de otimização mono-objetivo de alto custo computacional. Em sua estrutura, o SAEAa emprega diferentes operadores de mutação visando evitar uma convergência prematura e melhorar a capacidade do algoritmo de escapar de uma região de estagnação. Além disso, o método considera uma autoadaptação de parâmetro e define a escolha entre os diferentes operadores de acordo com o *status* de evolução do algoritmo. Realizou-se também uma análise gráfica preliminar, a qual sugere que o SAEAa é uma estratégia promissora no contexto de SAEAs.

De modo geral, estruturas de SAEAs exigem a manutenção de um conjunto de armazenamento de soluções avaliadas na função mérito. Embora esse conjunto influencie diretamente na qualidade do metamodelo e diversidade da população, muitos trabalhos na literatura não tratam em detalhes o processo de atualização desse conjunto. Nesse contexto, a descrição do método proposto abordou em detalhes questões que devem ser consideradas durante a atualização desse conjunto.

5 Planejamento Experimental e Resultados

O capítulo corrente apresenta o planejamento experimental adotado para a validação do SAEAa proposto no Capítulo 4. Nesse sentido, são escolhidos dois SAEAs recentes na literatura para comparar com o SAEAa, a saber, o TASEA(YANG et al., 2019) e o GSGA(CAI; LI, 2019) apresentados na Seção 3.2.3. Considerou-se pertinente explorar também variações do SAEAa e do TASEA, uma vez que ambos os métodos podem ser adaptados para empregar outros tipos de metamodelos. Nesse sentido, SAEAok representa a configuração do SAEAa que emprega o método de redução dimensional adotado em (YANG et al., 2019) na etapa de construção de um metamodelo OK, enquanto que SAEArbf identifica a configuração do SAEAa que também utiliza o método de redução dimensional na construção de um metamodelo RBF. Para o TASEA considerou-se a forma original deste método, a qual emprega um metamodelo OK com o método de redução dimensional. Considerou-se também a configuração do TASEA que emprega o método de redução dimensional na construção de um metamodelo RBF (TASEArbf). O restante deste capítulo está organizado conforme a descrição a seguir. A Seção 5.1 apresenta detalhes de implementação e configuração da máquina empregada na realização dos testes. A Seção 5.2 apresenta o conjunto de funções analíticas adotado para a validação do método proposto. Os indicadores de desempenho são apresentados na Seção 5.3. A configuração adotada para os experimentos é apresentada na Seção 5.4. A Seção 5.5 apresenta a análise dos resultados. A Seção 5.6 apresenta a aplicação do SAEAa em um projeto de antena para radar de subsolo (GPR, *Ground Penetrating Radar*). Finalmente, na Seção 5.7 são apresentadas as considerações finais sobre o desenvolvimento do capítulo.

5.1 Implementação e configuração da máquina

Para investigar o desempenho do SAEAa e comparar com os métodos TASEA e GSGA, implementou-se cada configuração desses métodos usando o MATLAB (versão 2017b). Na construção do metamodelo *Kriging* no SAEAa e GSGA empregou-se a SURROGATES toolbox (VIANA, 2010) devido a facilidade em adaptar essa toolbox para o uso do *DIRECT Algorithm* na otimização do parâmetro do metamodelo. Utilizou-se o código do *DIRECT Algorithm* disponível em <https://github.com/npinto/direct>; os parâmetros adotados são definidos em (FINKEL, 2003). Para construir o metamodelo *Kriging* em SAEAok e TASEA empregou-se a ooDACE *toolbox* (COUCKUYT; DHAENE; DEMEESTER, 2013). Adotou-se a SURROGATES toolbox (VIANA, 2010) para construir o metamodelo RBF em TASEArbf, SAEArbf e GSGA; a SURROGATES toolbox usa o software RBF (JēKABSONS, 2009). É importante mencionar que a normalização da

amostra em (4.11) é realizada pela ooDACE *toolbox* e SURROGATES toolbox. O processo de redução dimensional é realizada pela SOM toolbox (VESANTO et al., 2000), a qual é configurada com o critério de parada de 100 iterações. Para implemetar as configurações SAEAok e SAEArbf, utiliza-se o Algoritmo 3.13 na adapatação dos passos 22-24 do Algoritmo 4.4. No caso das configurações SAEArbf e TASEArbf, considera-se a função de aproximação $\hat{y}(\cdot)$ no lugar de EI$(\cdot)$ e LCB$(\cdot)$, respectivamente.

Um detalhe importante é que a versão original GSGA emprega um modelo RBF com função base do tipo *cubic* (*power function*). No entanto, a toolbox adotada neste trabalho para a construção de metamodelo do tipo RBF não apresenta esse tipo função base. Optou-se então por implementar o GSGA com metamodelo RBF e função base multiquadric; em testes preliminares esse tipo de RBF mostrou-se mais adequado para o GSGA entre as opções de RBF disponíveis na SURROGATES toolbox. Em relação as configurações SAEArbf e TASEArbf, adotou-se o metamodelo RBF com função base Gaussiana.

Devido ao elevado custo computacional de aplicação de SAEAs para a otimização de um conjunto considerável de problemas teste, não realizou-se um ajuste adequado dos parâmetros dos algoritmos, mas sim manteve-se os valores sugeridos em trabalhos anteriores (os quais também não evidenciam o emprego de técnicas específicas para este ajuste). Nesse contexto, assume-se que as comparações realizadas neste trabalho, embora limitadas, não privilegiam sabidamente um método específico, uma vez que nenhum dos algoritmos apresentou um ajuste específico de parâmetros.

Para a realização dos testes computacionais usou-se uma máquina virtual com 8 núcleos, 50 GB de memória RAM e sistema operacional Ubuntu 19.04, 64 bits. Implementou-se todas as configurações dos algoritmos em MATLAB 2017b e usou-se o R 3.5.2 para a análise estatística. O código fonte de cada algoritmo implementado e o roteiro das análises realizadas estão disponíveis para download em <https://github.com/monicavaladao/PhD-UFMG>.

5.2 Problemas analíticos

Aplicou-se cada configuração dos métodos SAEAa, TASEA e GSGA na minimização de um conjunto de funções analíticas listadas na Tabela 1. Essas funções possuem o valor 0 como mínimo global e mais detalhes são encontrados em (SURJANOVIC; BINGHAM, 2018; SUGANTHAN et al., 2005). Para cada função considerou-se o número de variáveis $n \in \{10, 20, 30, 50, 100, 200\}$, resultando em 72 problemas diferentes.

Tabela 1 – Problemas analíticos empregados nos experimentos computacionais.

Função	**Característica**	**Definição**
Sphere	*Unimodal*	$\mathrm{y}_1(\mathbf{x}) = \sum_{i=1}^{n} x_i^2$ $x_i \in [-5.12, 5.12]$
Different Powers	*Unimodal*	$\mathrm{y}_2(\mathbf{x}) = \sum_{i=1}^{n} i\lvert x_i\rvert^{i+1}$ $x_i \in [-1, 1]$
Ellipsoid	*Unimodal*	$\mathrm{y}_3(\mathbf{x}) = \sum_{i=1}^{n} i \cdot x_i^2$ $x_i \in [-5.12, 5.12]$
Rosenbrock	*Multimodal (narrow valley)*	$\mathrm{y}_4(\mathbf{x}) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (1 - x_i^2)^2]$ $x_i \in [-2.048, 2.048]$
Ackley	*Multimodal*	$\mathrm{y}_5(\mathbf{x}) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) +20 + \exp(1)$ $x_i \in [-32.768, 32.768]$
Griewank	*Multimodal*	$\mathrm{y}_6(\mathbf{x}) = 1 + \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right)$ $x_i \in [-600, 600]$
Rastrigin	*Multimodal*	$\mathrm{y}_7(\mathbf{x}) = 10n + \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i)]$ $x_i \in [-5.12, 5.12]$
Shifted Rotated Rastrigin	*Multimodal*	$\mathrm{y}_8(\mathbf{x}) = \sum_{i=1}^{n} [z_i^2 - 10\cos(2\pi z_i) + 10] - 330$ $\mathbf{z} = (\mathbf{x} - \mathbf{o}) \cdot \mathrm{M}$, $x_i \in [-5, 5]$ $\mathbf{o}$ e M definidos em (SUGANTHAN et al., 2005)
Rotated Version of Hybrid Composition	*Multimodal*	$\mathrm{y}_9(\mathbf{x}) = F(\mathbf{x})$ $x_i \in [-5, 5]$ $F(\cdot)$ definida em (SUGANTHAN et al., 2005)
Rotated Hybrid Composition with narrow	*Multimodal*	$\mathrm{y}_{10}(\mathbf{x}) = F(\mathbf{x})$, $x_i \in [-5, 5]$ $F(\cdot)$ definida em (SUGANTHAN et al., 2005)
Shifted Sphere	*Unimodal*	$\mathrm{y}_{11}(\mathbf{x}) = \sum_{i=1}^{n} z_i^2 - 450$ $\mathbf{z} = (\mathbf{x} - \mathbf{o})$, $x_i \in [-100, 100]$ $\mathbf{o}$ definido em (SUGANTHAN et al., 2005)
Shifted Rosenbrock	*Multimodal*	$\mathrm{y}_{12}(\mathbf{x}) = \sum_{i=1}^{n-1} [100(z_i^2 - z_{i+1} - 1)^2 + (z_i^2)^2] + 390$ $\mathbf{z} = (\mathbf{x} - \mathbf{o})$, $x_i \in [-100, 100]$ $\mathbf{o}$ definido em (SUGANTHAN et al., 2005)

5.3 Indicadores de desempenho

Nesta seção são apresentados os indicadores adotados para avaliar o desempenho das configurações SAEAa, SAEArbf, SAEAok, TASEA, TASEArbf e GSGA. Embora todas as funções da Tabela 1 possuam o valor 0 como mínimo global, a magnitude dos valores de função objetivo pode variar muito de um problema para outro dependendo do número de variávies. Nesse sentido, além do valor de função objetivo $\mathrm{y}(\cdot)$ como indicador de convergência, considera-se também a normalização a seguir (VALADãO; BATISTA, 2020)

$$\Delta_{\%}^{(i)} = \frac{\mathrm{y}^{(0)} - \mathrm{y}^{(i)}}{\mathrm{y}^{(0)}} \tag{5.1}$$

onde $\Delta_{\%}^{(i)}$ é definido no intervalo $[0, 1]$; $\mathrm{y}^{(0)}$ e $\mathrm{y}^{(i)}$ representam, respectivamente, o valor de $\mathrm{y}(\cdot)$ na melhor solução da população inicial e o valor de $\mathrm{y}(\cdot)$ na melhor solução da população corrente da iteração i. Assim, $\Delta_{\%}^{(i)}$ representa o percentual de melhora do valor de função objetivo associado a melhor solução da iteração i em relação a melhor solução da população inicial.

Para comparar as estratégias em relação a qualidade da melhor solução encontrada adota-se o indicador $\Delta_{\%}^{(\star)}$, o qual representa o valor obtido em (5.1) na última iteração do algoritmo. Considera-se também um indicador para comparar o tempo de execução associado a cada estratégia. Assim, para cada abordagem, T_{sec} representa tempo médio (em segundos) por iteração do algoritmo. A Tabela 2 apresenta os indicadores de desempenho descritos nessa seção.

Tabela 2 – Indicadores de desempenho adotados nos experimentos computacionais.

Indicator	**Descrição**
$\mathrm{y}(\cdot)$	Valor de função objetivo.
$\Delta_{\%}^{(*)}$	Qualidade da melhor solução representada pela melhora relativa em relação a melhor solução da população inicial.
T_{sec}	Tempo médio (em segundos) de execução em cada iteração do algoritmo.

5.4 Planejamento Experimental

Esta seção apresenta a definção dos parâmetros associados as abordagens SAEAa, SAEArbf, SAEAok, TASEA, TASEArbf e GSGA. A Tabela 3 apresenta os parâmetros adotados que são comuns aos problemas mencionados anteriormente. Por outro lado, existem parâmetros que são específicos de cada configuração, os quais são apresentados na Tabela 4. O critério de parada adotado em todas as abordagens é o número máximo de avaliações *neval* permitido em $\mathrm{y}(\cdot)$. Nessas tabelas, n é o número de variáveis; N_s é o número de soluções da amostra inicial; N é o número de soluções da população; N_{DB} é

o número máximo de soluções permitido em DB; N_m é número de soluções da amostra do metamodelo; N_c é número máximo de iterações sem melhora na função objetivo; k é o número de soluções mapeadas pela função *prescreening*; N_{RBF} é o número de soluções da amostra do metamodelo RBF e N_{new} é o número de soluções mapeadas pela função *prescreening*. Para as configurações TASEA,TASEArbf, SAEAok, SAEArbr, a dimensão do espaço reduzido é $l = 4$, i.e., o mesmo valor adotado em (YANG et al., 2019). No GSGA utiliza-se todas as soluções armazenadas em DB para construir o modelo *Kriging*. Em todas as abordagens, a atualização de DB é realizada conforme o Algoritmo 4.6 e detalhes relacionados na Seção 4.3.1. A escolha das soluções que constituirão a amostra do metamodelo em cada abordagem considera também o exposto na Seção 4.3.1. Com relação ao parâmetro N_m, o valor adotado considera como referência os trabalhos (LIU; ZHANG; GIELEN, 2014; YANG et al., 2019). Com relação ao parâmtro N_s optou-se por definir a amostra inicial com uma quantidade a mais de soluções além do que é necessário para a construção de metamodelo. Para definir o valor do parâmetro N considerou-se o mesmo critério adotado no SL-PSO proposto em (CHENG; JIN, 2015) pelo fato de ter sido aplicado em problemas analíticos com número de variáveis $n \geq 100$. Com relação ao parâmetro N_{DB} observa-se que a literatura não trata em detalhes o processo de atualização e cardinalidade do conjunto de armazenamento. Entretanto, entende-se que é necessário definir para N_{DB} um valor maior que N e N_m, uma vez que a população corrente e amostra de metamodelo são definidas a partir de soluções do conjunto DB.

Tabela 3 – Parâmetros usados para cada problema (por número de variáveis).

n	N_s	N	N_{DB}	$neval$
10	$4 \cdot N$	$50 + \left\lfloor \frac{n}{10} \right\rfloor$	400	2000
20	$4 \cdot N$	$50 + \left\lfloor \frac{n}{10} \right\rfloor$	400	2000
30	$4 \cdot N$	$50 + \left\lfloor \frac{n}{10} \right\rfloor$	400	2000
50	$4 \cdot N$	$50 + \left\lfloor \frac{n}{10} \right\rfloor$	400	2000
100	$4 \cdot N$	$50 + \left\lfloor \frac{n}{10} \right\rfloor$	400	2000
200	$4 \cdot N$	$80 + \left\lfloor \frac{n}{10} \right\rfloor$	400	2000

Tabela 4 – Parâmetros usados para cada problema (por número de variáveis).

	SAEAa SAEAok SAEArbf			TASEA TASEArbf			GSGA	
n	N_m	N_c	k	N_m	N_c	k	N_{RBF}	N_{new}
10	100	10	3	100	20	3	$5 \cdot n + 10$	10
20	100	10	3	100	20	3	$5 \cdot n + 10$	10
30	100	10	3	100	20	3	$5 \cdot n + 10$	10
50	100	10	3	100	20	3	$5 \cdot n + 10$	10
100	100	10	3	100	20	3	$5 \cdot n + 10$	10
200	100	20	3	100	20	3	$5 \cdot n + 10$	10

Para cada problema (cada função da Tabela 1 com $n \in \{10, 20, 50, 100, 200\}$)

executou-se 20 testes com cada uma das abordagens. Ao final de cada execução, de cada configuração com cada problema, calculou-se os indicadores $\Delta_{\%}^{(*)}$e T_{sec}. Considerando que cada problema representa uma replicação no experimento (BIRATTARI, 2004; MONTGOMERY; RUNGER, 2013), considerou-se cada indicador de desempenho sumarizado como média das replicações para cada problema em cada configuração. Assim, para cada configuração, a amostra associada a cada um dos indicadores possui cardinalidade 72. Para cada indicador de desempenho, calculou-se o intervalo de confiança 95% usando bootstrap (CRAWLEY, 2007) para diferenças pareadas. Usou-se a correção de Bonferroni (MONTGOMERY; RUNGER, 2013) para ajustar o nível de significância de cada intervalo.

5.5 Análise dos Resultados

5.5.1 Qualidade da solução final

A Figura 4 mostra a média da melhor solução representada pelo indicador de melhora relativa em relação a melhor solução inicial ($\Delta_{\%}^{(\star)}$). Observa-se nessa figura um desempenho semelhante entre as estratégias SAEAok, SAEAa e GSGA. Ainda com relação ao indicador $\Delta_{\%}^{(\star)}$, a Figura 5 apresenta comparações múltiplas entre as estratégias na qual observa-se que não existe diferença estatisticamente significativa da estratégia SAEAa quando comparada com SAEAok e GSGA. Entre as variações do método proposto neste trabalho, observa-se também que não existe diferença estatisticamente significativa entre as estratégias SAEArbf e TASEA.

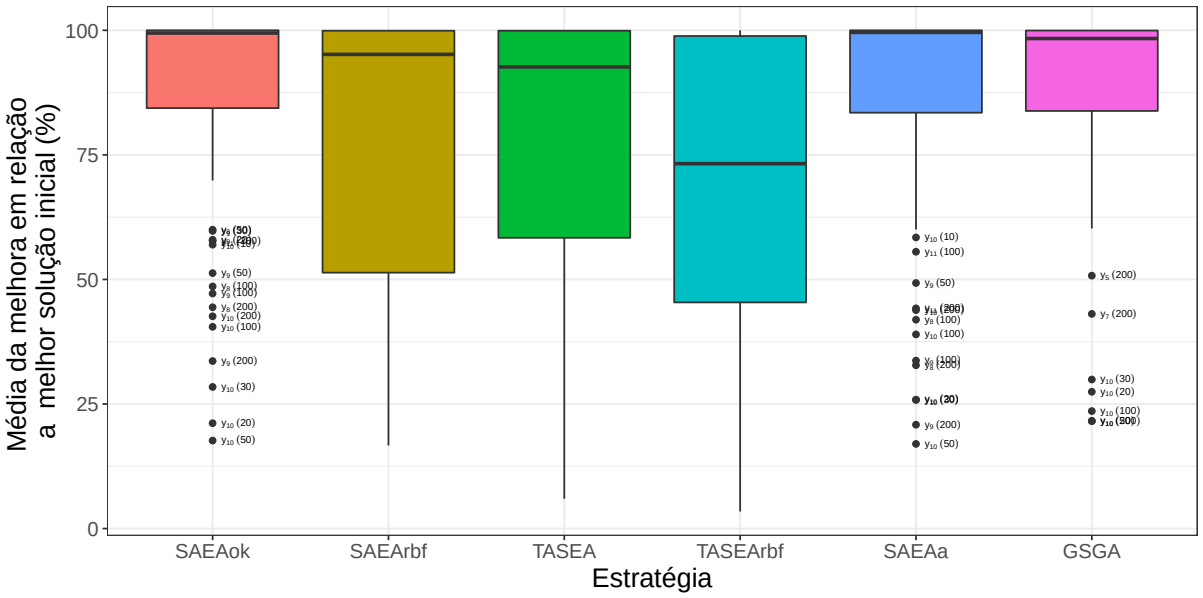


Figura 4 – Média da melhora em relação a solução inicial.

É importante apresentar também informações sobre o valor de função objetivo da melhor solução retornada por cada método. Devido a quantidade de problemas analíticos

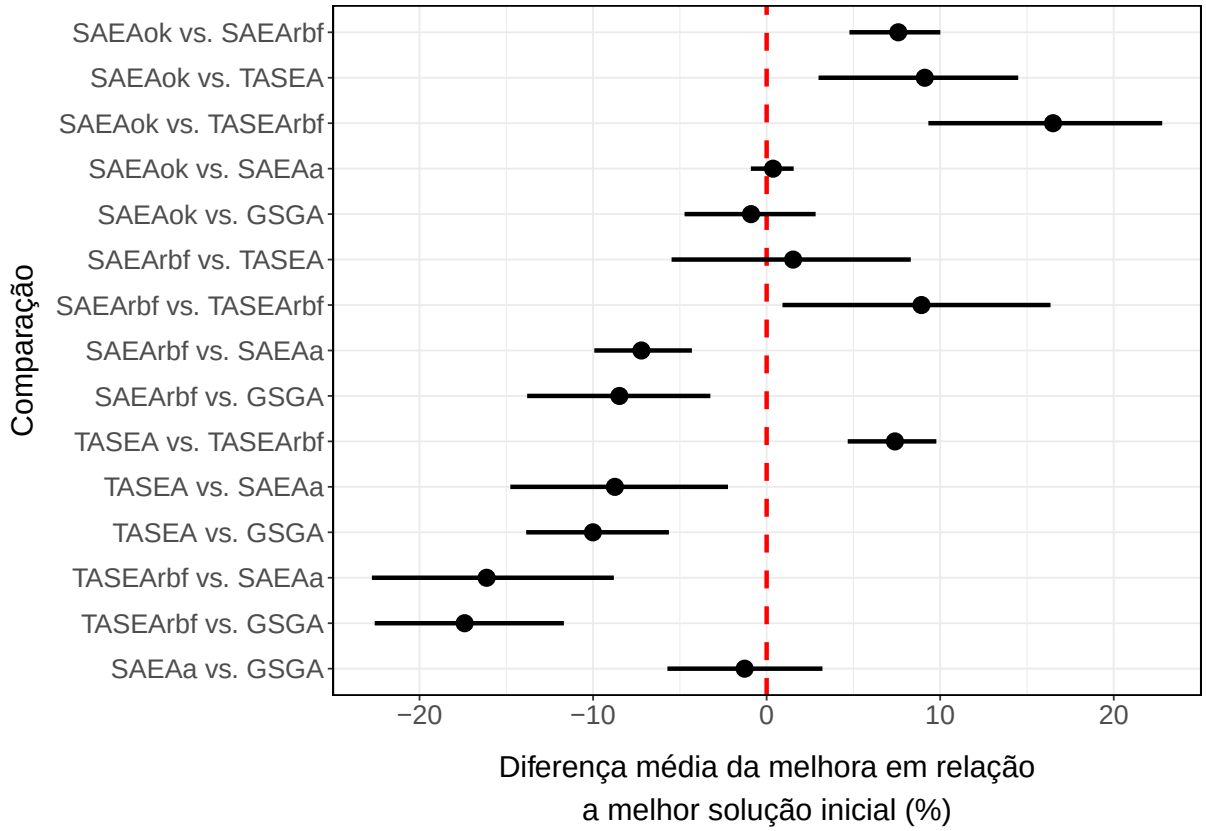


Figura 5 – Comparação da diferença média da melhora em relação a melhor solução inicial. Calculou-se os intervalos de confiaça de 95% usando bootstrap para diferenças pareadas considerando todos os problemas.

considerados, optou-se por mostrar estas informações no Apêndice A. Nesse sentido, as Tabelas 8-11 apresentam a média e o desvio padrão do valor de função objetivo da melhor solução retornada por cada estratégia. Os valores apresentados nessas tabelas correspondem às funções analíticas listadas na Tabela 1 para cada número de variáveis n. Para facilitar a leitura, optou-se por apresentar os resultados organizando em uma mesma tabela as estratégias que apresentam melhor desempenho. Além disso, as Tabelas 10-11 apresentam os resultados da estratégia SAEAa (método proposto neste trabalho) com as estratégias TASEA e GSGA (métodos recentes da literatura).

5.5.2 Capacidade de convergência

A Figura 6 apresenta, para algumas funções da Tabela 1 e $n = 200$, a curva de convergência de cada estratégia, considerando a média de melhora em relação a melhor solução da população inicial ($\Delta_{\%}^{(\star)}$) em função do número de avaliações de função objetivo. Em relação aos métodos da literatura adotados para a comparação neste trabalho, observa-se que o TASEA apresenta uma convergência lenta em relação ao GSGA e o SAEAa. Observa-se também uma melhor convergência do GSGA em relação ao SAEAa para y_8 e

comportamento de convergência semelhante para as demais funções desta figura. Por outro lado, conforme ilustrado na Figura 5, não existe diferença estatisticamente significativa entre o SAEAa e GSGA considerando o indicador de qualidade de solução. As curvas de convergência de todos os problemas analíticos considerados neste trabalho são apresentadas na Figura 16 do Apêndice A.

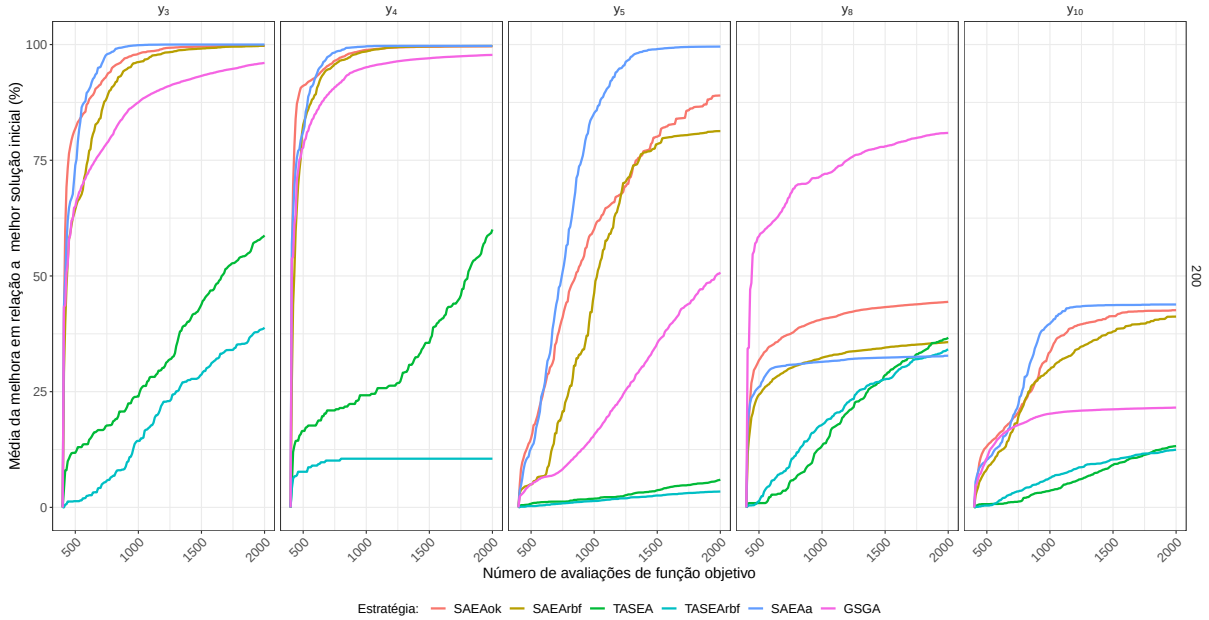


Figura 6 – Curvas de convergência da média de melhora em relação a melhor solução da população inicial em função do número de avaliações de função objetivo. Os plots são discretizados por função (vertical) e número de variáveis (horizontal).

5.5.3 Custo computacional

Esta análise compara o tempo de execução por iteração (T_{sec}) associado a cada estratégia. As informações apresentadas nas Figuras 7 e 8 referem-se a média do tempo de execução (em segundos) para os métodos SAEAok, SAEArbf, TASEA, TASEArbf e SAEAa. Optou-se por não apresentar o GSGA nestas figuras para facilitar a visualização das informações, uma vez que este método exigiu o maior tempo computacional de execução por iteração, conforme pode ser observado na Figura 9. Para o GSGA, um maior número de variáveis representa um aumento considerável no custo computacional devido a construção e minimização de um metamodelo RBF para cada solução da população corrente a cada iteração. Os resultados apresentados desta forma permitem observar na Figura 7 a menor média de custo computacional por iteração associado ao método SAEAa. Em relação a Figura 8, é interessante observar que existe diferença estatisticamente significativa da média do tempo de execução por iteração do TASEA em comparação com a estratégia SAEArbf. Essa informação associada a Figura 5, na qual observa-se que não existe diferença estatisticamente significativa entre as estratégias SAEArbf e

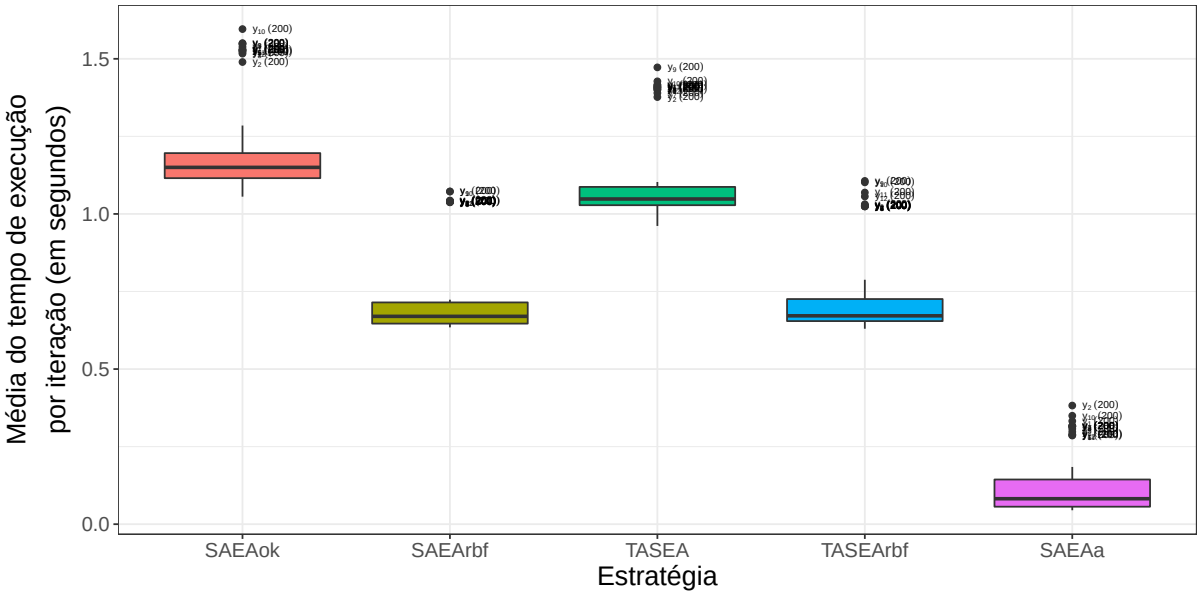


Figura 7 – Média do tempo de execução (em segundos) por iteração associado a cada estratégia.

TASEA em relação ao inidicador de qualidade de solução, sugere o SAEArbf como uma alternativa melhor que o método da literatura TASEA. A Figura 9 permite visualizar simultaneamente os indicadores $\Delta_{\%}^{(\star)}$ e T_{sec} das estratégias analisadas neste capítulo. Essa figura permite observar ainda que o tempo médio de execução por iteração do GSGA é maior que das outras estratégias. Assim, o SAEAa é uma alternativa melhor que o TASEA, considerando que este se destaca do TASEA em relação ao indicador $\Delta_{\%}^{(\star)}$ e custo computacional. Embora não exista diferença estatisticamente significativa entre o SAEAa e GSGA em relação ao indicador $\Delta_{\%}^{(\star)}$, o SAEAa representa uma melhor alternativa que o GSGA devido ao menor custo computacional conforme mostra a Figura 9.

5.5.4 Comparação com outros métodos

Considerou-se pertinente apresentar também uma comparação qualitativa do SAEAa com outros três SAEAs da literatura. Entretanto, realizou-se somente este tipo de comparação porque os algoritmos em questão não são replicáveis a partir da descrição dos trabalhos relacionados e não foi possível obter os códigos fonte junto aos autores. Nesse sentido, a Tabela 5 apresenta a média e o desvio padrão do valor final da melhor solução retornada pelo SAEAa em comparação com as estratégias GEPEME (LIU; ZHANG; GIELEN, 2014), SA-COSO (SUN et al., 2017) e SHPSO (YU et al., 2018). Os resultados do GEPEME, SA-COSO e SHPSO apresentados na tabela são obtidos a partir do que é reportado nos respectivos trabalhos[1]. Observou-se o uso de algumas funções em comum e realização de testes com critério de parada dos métodos $neval = 1000$

[1] Optou-se por manter a função *Shifted Rotated Rastrigin* com o valor mínimo -330 reportado em (LIU; ZHANG; GIELEN, 2014; SUN et al., 2017; YU et al., 2018) e adequar o valor do SAEAa.

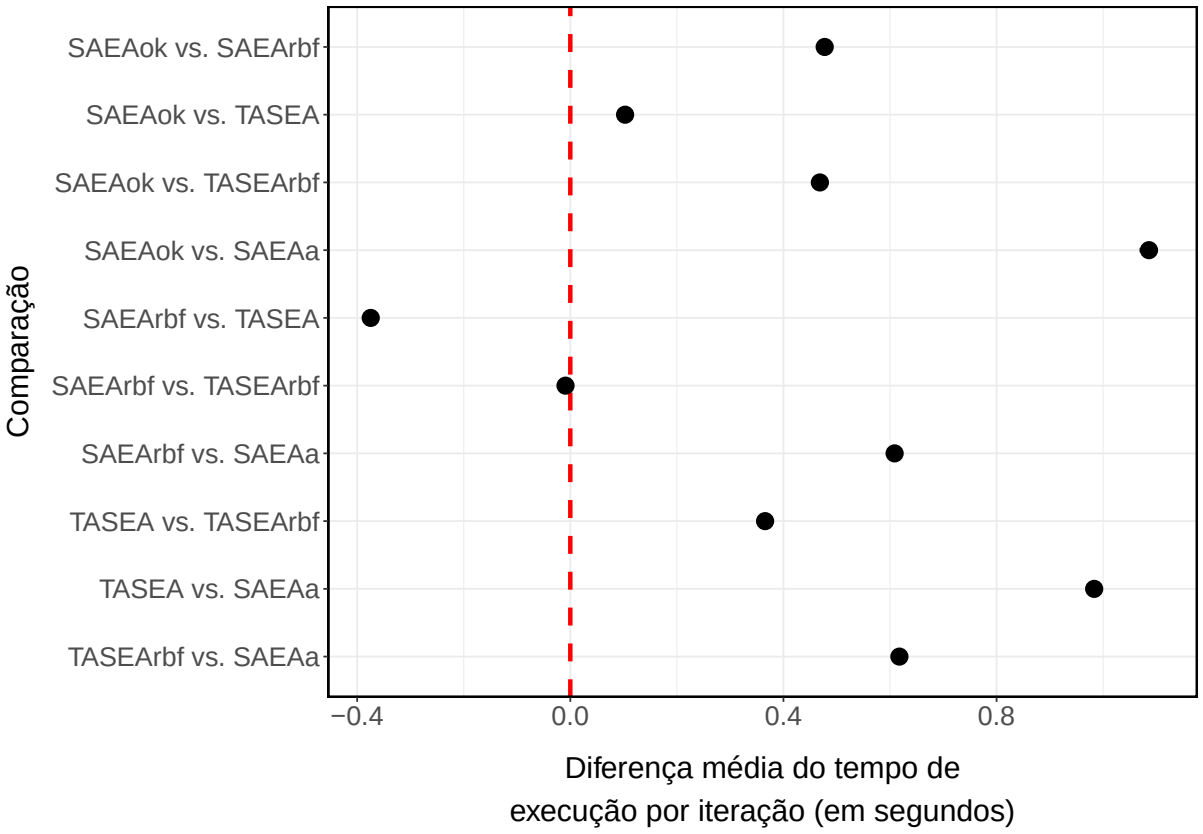


Figura 8 – Comparação da diferença média do tempo de execução (em segundos) por iteração. Calculou-se os intervalos de confiaça de 95% usando bootstrap para diferenças pareadas considerando todos os problemas.

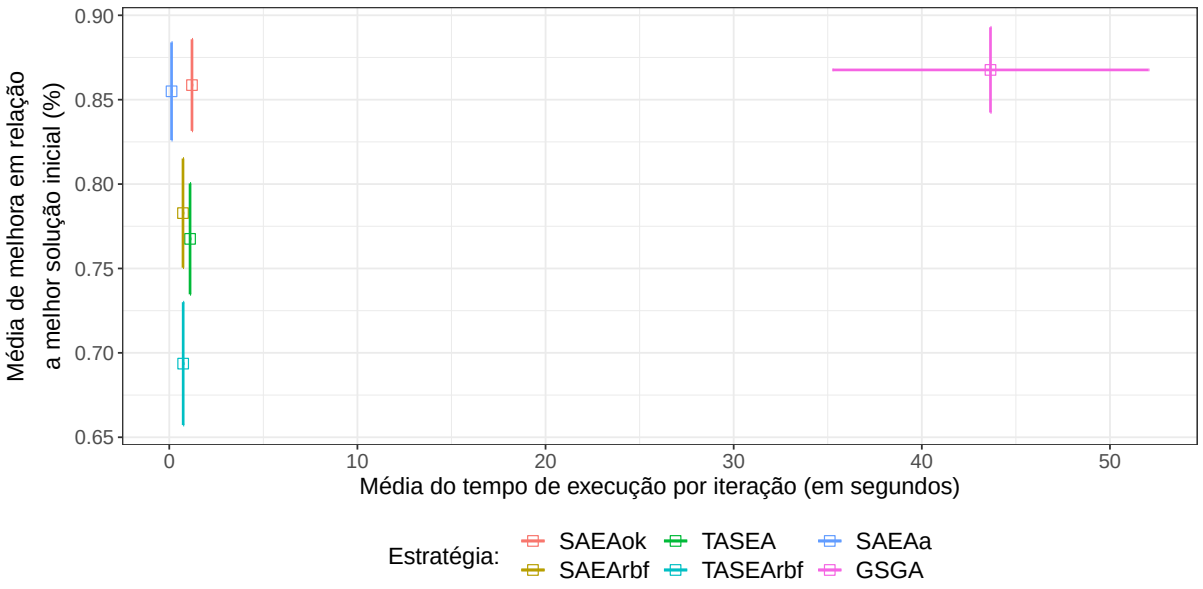


Figura 9 – Crossbar comparando a média dos indicadores de desempenho T_{sec} (eixo x) e $\Delta^{(\star)}_{\%}$ (eixo y) das diferentes estratégias. Marcadores quadrados representam a média e as linhas horizontal/vertical representam média $\pm$ erro padrão.

Tabela 5 – Média e desvio padrão do valor da melhor solução retornada pelas estratégias GEPEME, SA-COSO, SHPSO e SAEAa para alguns problemas e número de variáveis com $neval = 1000$.

	GEPEME		SA-COSO		SHPSO		SAEAa	
n	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.
Ellipsoid								
20	**1.3e-5**	2.18e-5	-	-	-	-	0.0160	0.0304
30	**0.0762**	0.0401	-	-	0.2120	0.1523	2.4461	4.0067
50	221.0774	81.6123	51.475	16.246	**4.0281**	2.0599	55.5797	76.1902
100	-	-	1033.2	317.18	76.106	21.447	**61.4230**	113.5187
Rosenbrok								
20	22.4287	18.7946	-	-	-	-	**18.8104**	1.5168
30	46.1773	25.5199	-	-	**28.566**	0.4044	40.0218	17.6094
50	258.2787	80.1877	252.58	40.744	**50.800**	3.0305	62.2836	23.6856
100	-	-	2714.2	117.02	165.59	26.366	**126.9652**	52.3717
Ackley								
20	**0.1990**	0.5771	-	-	-	-	3.7121	3.6759
30	3.0105	0.9250	-	-	**1.4418**	0.7740	2.7088	3.3257
50	13.2327	1.5846	8.9318	1.0668	1.8389	0.5637	**1.6277**	2.3116
100	-	-	15.756e	0.50245	4.1134	0.5925	**0.4866**	0.4956
Griewank								
20	**0.0307**	0.0682	-	-	-	-	0.1914	0.2504
30	0.9969	0.1080	-	-	**0.9205**	0.0881	0.9534	1.2674
50	36.6459	13.1755	6.0062	1.1043	**0.9452**	0.0614	3.6417	5.1726
100	-	-	63.353	19.021	**1.0704**	0.0205	7.3762	13.0509
Shifted Rotated Rastrigin								
30	-21.8610	36.4492	-	-	-92.830	22.544	**-169.2839**	32.3888
50	-	-	197.16	30.599	**134.42**	32.256	313.2128	82.9705
100	-	-	1273.1	117.19	**801.73**	72.252	1822.4	100.0834

avaliações de função objetivo. Observa-se que o método proposto sugere um melhor desempenho para a maioria das funções consideradas em comparação com o GEPEME e SA-COSO. O GEPEME emprega um operador DE/best/1, o que pode explicar um melhor desempenho deste método na função unimodal *Ellipsoid* para $n = 20, 30$. Com relação ao custo computacional é importante ressaltar que o GEPEME emprega um metamodelo OK com redução dimensional somente para $n = 50$, enquanto que no SAEAa emprega-se um metamodelo OK unidimensional em todos os problemas adotados para a validação deste método. Devido a esta característica, pode-se afirmar que o GEPEME representa um maior custo computacional em relação ao SAEAa. Por outro lado, para o SA-COSO não é possível fazer uma afirmativa assertiva sobre seu custo computacional em relação SAEAa, considerando somente uma análise qualitativa, porque estes métodos empregam metamodelos de naturezas distintas. Entretanto, observa-se que no SA-COSO são construídos vários metamodelos RBF a cada iteração, o que sugere que este método envolve um custo computacional que pode ser considerável conforme aumenta o número de variáveis do problema. Para os métodos SAEAa e SHPSO não é possível identificar qual estratégia sugere um melhor desempenho, uma vez que para uma mesma função estes métodos se diferenciam um do outro conforme o número de variáveis. Também não é possível inferir entre o SAEAa e SHPSO qual método é mais vantajoso em termos de

custo computacional. Embora o SHPSO empregue um metamodelo RBF, o qual é um metamodelo de baixo custo computacional, pode ocorrer a construção de dois metamodelos RBF em uma mesma iteração. Assim, para uma comparação justa, em termos de custo computacional e qualidade de solução, considera-se que é necessário realizar uma investigação detalhada seguindo o mesmo protocolo experimental adotado neste capítulo.

5.5.5 Considerações sobre limitações para realização de testes com $n > 200$

Em um SAEA é importante observar os requisitos necessários para a construção do metamodelo envolvido. Além dos aspectos relacionados a construção de metamodelo, abordados na Seção 4.3.1, deve-se considerar também a quantidade mínima de soluções que o metamodelo exige na amostra. Neste sentido, nota-se que o método GSGA emprega um metamodelo UK1, o qual exige uma amostra com no mínimo $(n + 1)$ soluções, onde n é o número de variáveis. Na Seção 2.6 são apresentados mais detalhes relacionados a cardinalidade mínima da amostra em variações do modelo *Kriging*. Para problemas com um maior número de variáveis, o GSGA exige inicialmente uma quantidade considerável do orçamento de avaliação de função apenas para definir a amostra do metamodelo. Por outro lado, estratégias que empregam o metamodelo OK não apresentam essa limitação quando considerado o contexto de problemas com elevado número de variávies.

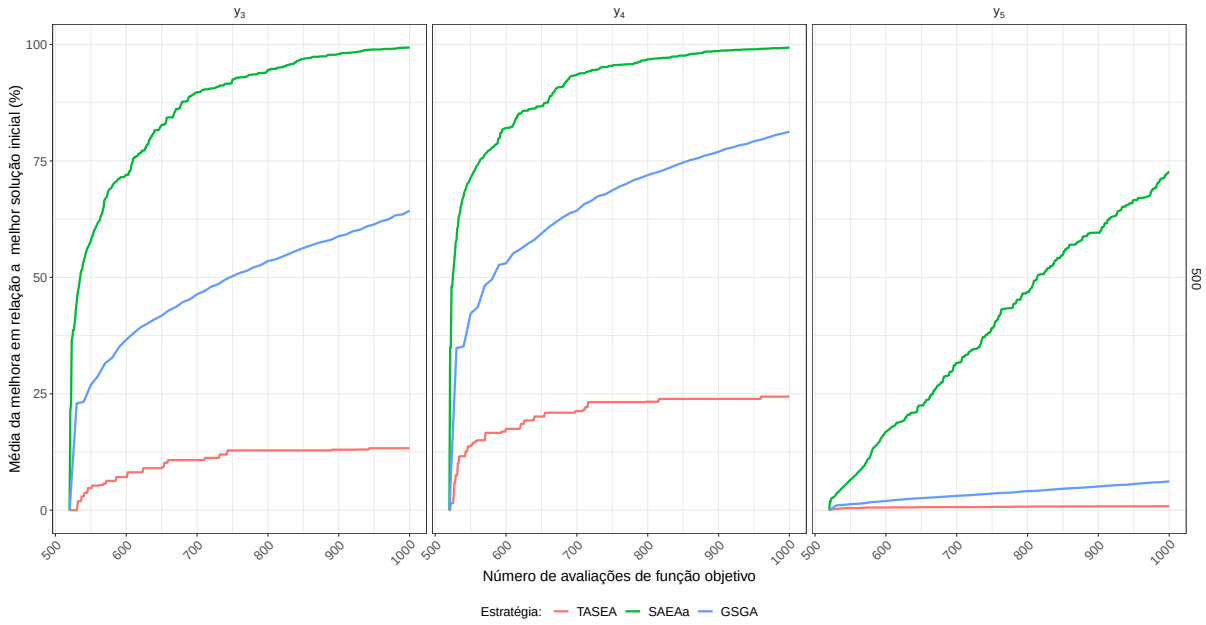


Figura 10 – Curvas de convergência da média de melhora em relação a melhor solução da população inicial em função do número de avaliações de função objetivo. Os plots são discretizados por função (vertical) e número de variáveis (horizontal).

Para ilustrar essa questão, considerou-se a aplicação das estratégias SAEAa, TASEA e GSGA em três funções da Tabela 1 com $n = 500$ variáveis. Manteve-se também os mesmos parâmetros destes métodos definidos na Seção 5.4 e com critério de parada de 1000

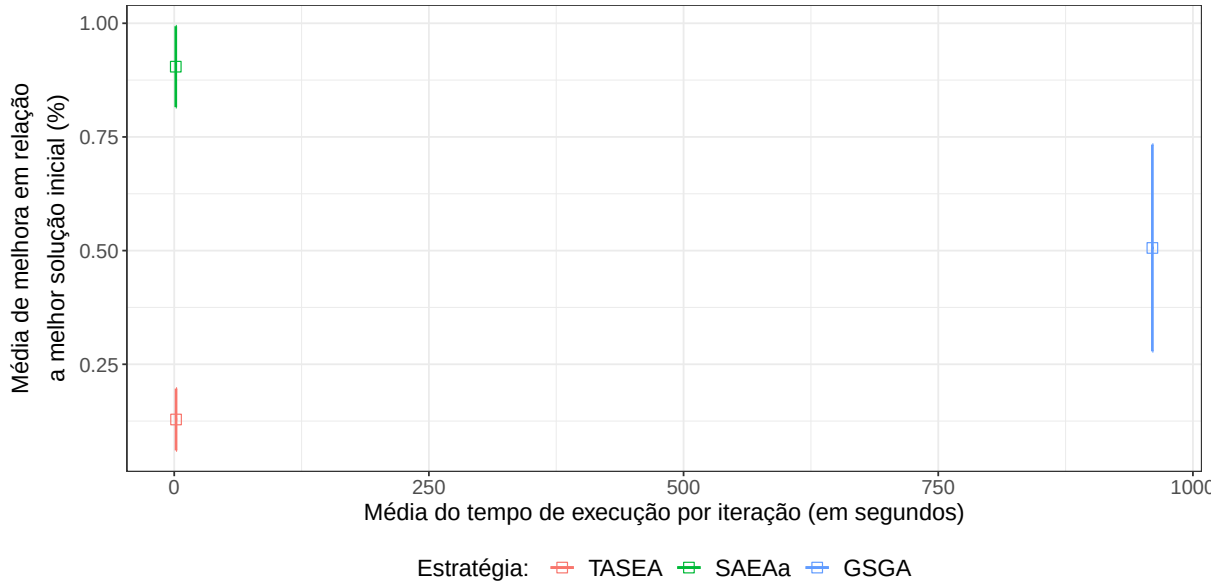


Figura 11 – Crossbar comparando a média dos indicadores de desempenho T_{sec} (eixo x) e $\Delta_{\%}^{(\star)}$ (eixo y) das diferentes estratégias. Marcadores quadrados representam a média e as linhas horizontal/vertical representam média ± erro padrão.

avaliações da função mérito. Na Figura 10 são apresentadas as curvas de convergência, de cada método aplicado nos problemas mencionados anteriormente, considerando a média de melhora em relação a melhor solução da população inicial ($\Delta_{\%}^{\star}$) em função do número de avaliações de função objetivo. Nota-se que o desempenho do SAEAa se destaca em relação ao TASEA e o GSGA. O custo computacional de aplicação das estratégias para $n = 500$ é apresentado na Figura 11, na qual observa-se um custo considerável para o GSGA.

Um detalhe importante é que o GSGA pode ser adaptado para empregar um metamodelo OK, da mesma forma que o TASEA e SAEAa, mantendo ainda o uso de parâmetro unidimensional. Esta modificação permitirá realizar uma comparação mais equilibibrada entre esses métodos, quando aplicado em problemas com elevado número de variáveis. O uso de um metamodelo OK exigirá uma menor quantidade de soluções na amostra do metamodelo em comparação com um metamodelo UK1. Assim, ao reduzir o número de avaliações iniciais será possível realizar mais iterações do GSGA com um mesmo orçamento de avaliação da função mérito. Entretanto, tal modificação não foi considerada no contexto deste trabalho.

5.6 Aplicação em Problema Prático

O projeto de antenas utilizando ferramentas de otimização tem sido muito explorado em diversas aplicações caracterizando-se sempre por um constante *trade-off* entre os seus objetivos a serem atingidos. O custo computacional da otimização no projeto de antenas depende principalmente do método numérico utilizado para calcular as características da

antena, seja no domínio do tempo ou da frequência. Particularmente, para uma antena apresentar um desempenho adequado em um sistema de radar de subsolo (GPR) precisa-se de: baixo custo (estrutura microstrip), baixo perfil (volume reduzido) e um desempenho eletromagnético alto (ganho constante e adequado, diretividade, banda de frequências ampla, bom casamento de impedância, baixa dispersão) (TRAVASSOS et al., 2018).

Especificamente, o projeto de antena considerado neste trabalho para aplicação do SAEAa é abordado em (AFRICANO et al., 2020), o qual estebelece uma metodologia para a obtenção das características eletromagnéticas das antenas com aplicação em GPR (*Ground Penetrating Radar*). O trabalho em questão propõe um problema de pavimento para o qual determina-se os parâmetros apropriados em tempo e frequência que a antena deve apresentar para se adaptar bem à aplicação. Os parâmetros da banda de frequências apresentados na Tabela 6 são adotados na definição do problema de otimização em (5.2).

Tabela 6 – Parâmetros da banda de frequência adotados na definição do problema.

Parâmetro	**Valor**
Frequência central da banda	2.93 GHz
Frequência mínima da banda (f_{min})	1.33 GHz
Frequência máxima da banda (f_{max})	4.53 GHz
Largura de banda	3.2 GHz
Largura de pulso máxima	3.46 ns

Para definir o problema de otimização são escolhidas duas características electromagnéticas da antena que são tradicionalmente selecionadas neste tipo de projeto: o parâmetro s11, relacionado ao casamento da antena com a linha de transmissão, e a diretividade da antena na direção de máxima radiação Dmax. Assim, o problema em questão é definido como segue,

$$\begin{aligned} &\min -||s11(\mathbf{x})||_{\infty} \\ &\text{sujeito a: } \mathrm{Dmax} > 0.5\mathrm{Dref} \end{aligned} \tag{5.2}$$

onde Dref $= 9.41$ é um valor de referência de diretividade e $\mathbf{x}$ é o vetor das variáveis de decisão do problema. As especificações das variáveis de decisão, bem como o número de variáveis (o qual é $n = 15$), a geometria da antena e o material adotado são apresentados em (AFRICANO et al., 2020). Em relação aos parâmetros da Tabela 6 considerou-se a alteração $f_{min} = 1.5$ GHz e $f_{max} = 4.5$ GHz.

Para aplicar o método SAEAa no projeto de antena transformou-se (5.2) no problema irrestrito a seguir,

$$\min \Big\{ \; \mathrm{y}(\mathbf{x}) = -||s11(\mathbf{x})||_{\infty} + \alpha \Big[\max(-\mathrm{Dmax} + 0.5\mathrm{Dref}, 0) \Big]^2, \mathbf{x} \in \Omega \tag{5.3}$$

com $\alpha = 1$ e Ω a região factível do problema.

Para o projeto de antena (5.3) executou-se 5 testes[2] com o método SAEAa. Considerou-se $N_s = 2N$ e critério de parada $neval = 1000$; os demais parâmetros são os mesmos das Tabelas 3 e 4 para $n = 10$. Em cada teste, para cada iteração, armazenou-se o valor normainf $= -||s11(\mathbf{x})||_\infty$, a diretividade na direção de máxima radiação, a respectiva solução $\mathbf{x}$ e a quantidade *neval* associada. Emprega-se o método Diferenças Finitas no Domínio do Tempo (FDTD, *Finite Difference Time Domain*) (ELSHERBENI; DEMIR, 2009) para obter o valor de função objetivo em (5.3). Para ilustrar o desempenho do SAEAa no problema em (5.3) são apresentadas na Figura 12 as curvas de convergência do valor de normainf para cada teste em função do número de avaliações realizadas na função objetivo. Observa-se que o SAEAa não apresenta um comportamento de convergência semelhante em todos testes realizados. Entretanto, espera-se que com um critério de parada com maior número de avaliações de função todas as execuções retornem soluções finais parecidas. A melhor solução encontrada pelo SAEAa (Teste 4) é apresentada na Tabela 7.

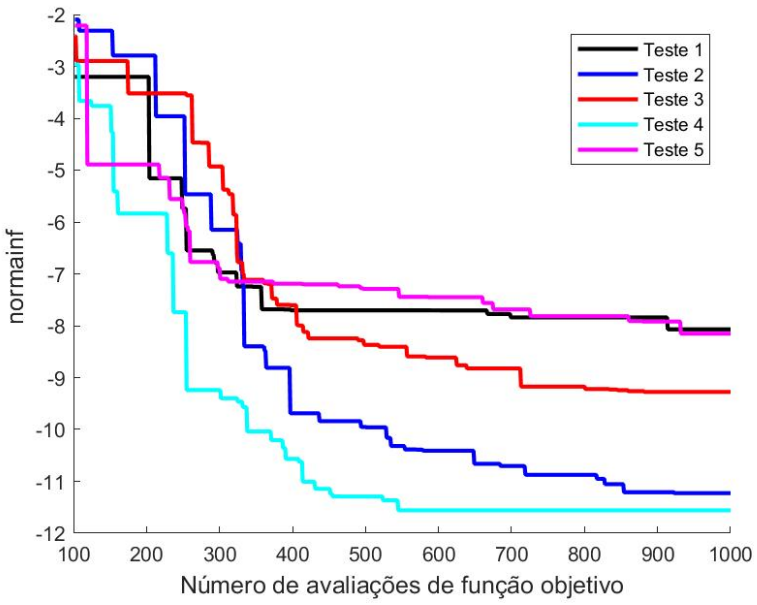


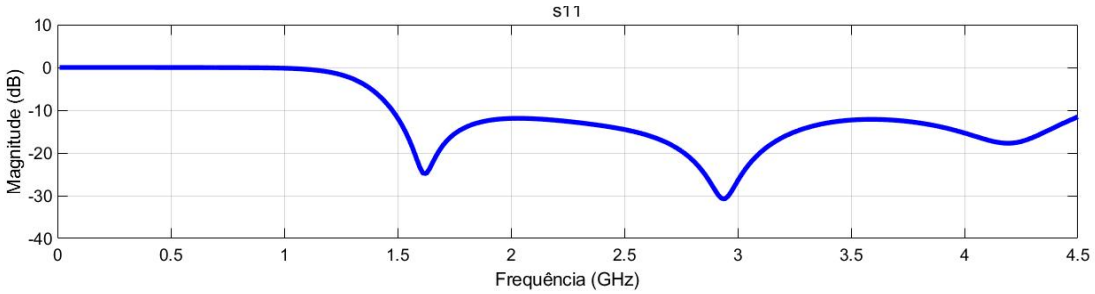
Figura 12 – Curvas de convergência valor de normainf para cada teste.

Tabela 7 – Solução estimada para o problema da antena.

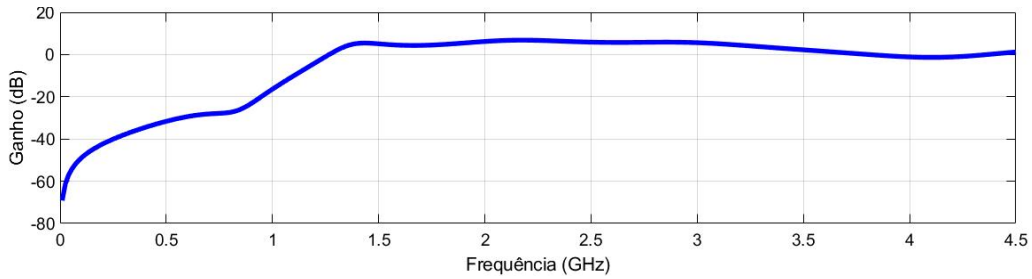
Variável	W_d	L_d	W_f	L_f	W_p	L_p	W_g	L_g	x_a	y_a	W_{xa}	L_{xa}	W_R	L_R	D_R
Valor (mm)	60	74	3	35	21	25	50	33	6	72	15	1	123.8	118.5	37.6

A Figura 13 apresenta a validação da antena, no domínio da frequência e do tempo, obtida pelo algoritmo de otimização utilizando o método FDTD. Conforme apresentado

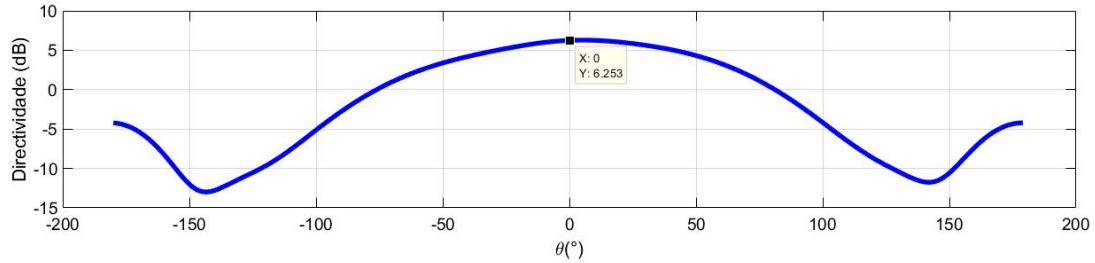
[2] Especificações da máquina: máquina virtual com 8 núcleos, 50 GB de memória RAM e sistema operacional Ubuntu 19.04, 64 bits. Implementação dos códigos em MATLAB 2017b.



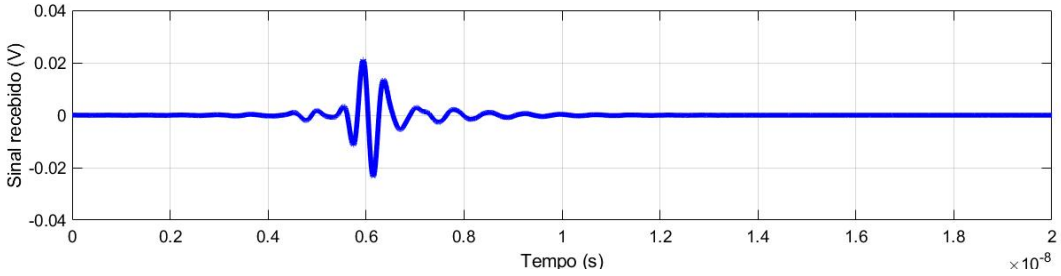
(a) s11



(b) Ganho



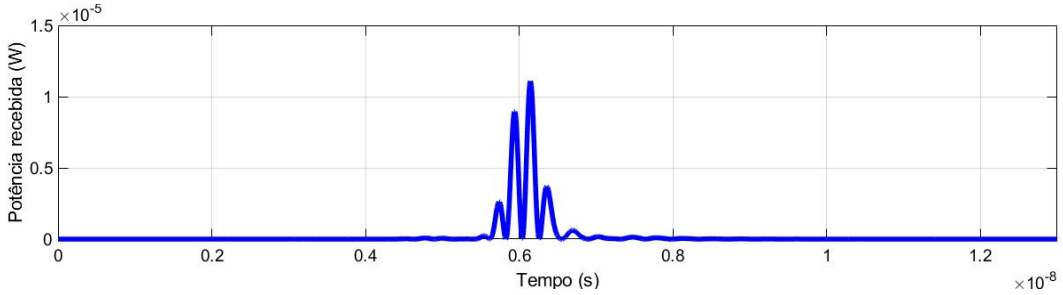
(c) Diretividade



(d) Sinal recebido $1m$

Figura 13 – Validação da solução apresentada na Tabela 7.

na Figura 13a, o parâmetro s11 mostra que a antena possui um casamento de impedâncias adequado na faixa de frequências de interesse (1.5 GHz até 4.5 GHz). Por outro lado, as Figuras 13b e 13c mostram que a antena apresenta um ganho adequado em quase toda a faixa de interesse e que a antena é diretiva, ou seja, concentra a energia irradiada na direção desejada ($\theta = 0$). Além disso, seguindo o mesmo processo de validação da antena no domínio do tempo utilizado em (AFRICANO et al., 2020), o sinal recebido na antena receptora é apresentado na Figura 13d e a largura do pulso (t_p) é obtida utilizando o gráfico de potência mostrado na Figura 13e. A largura do sinal recebido é $0.97\,ns$, o que



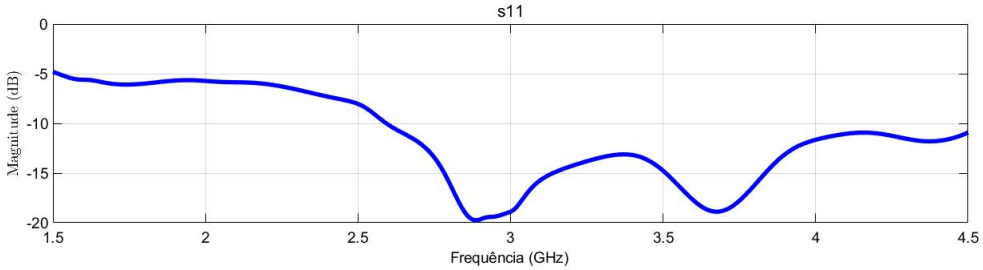
(e) Potência recebida

Figura 13 – Validação da solução apresentada na Tabela 7.

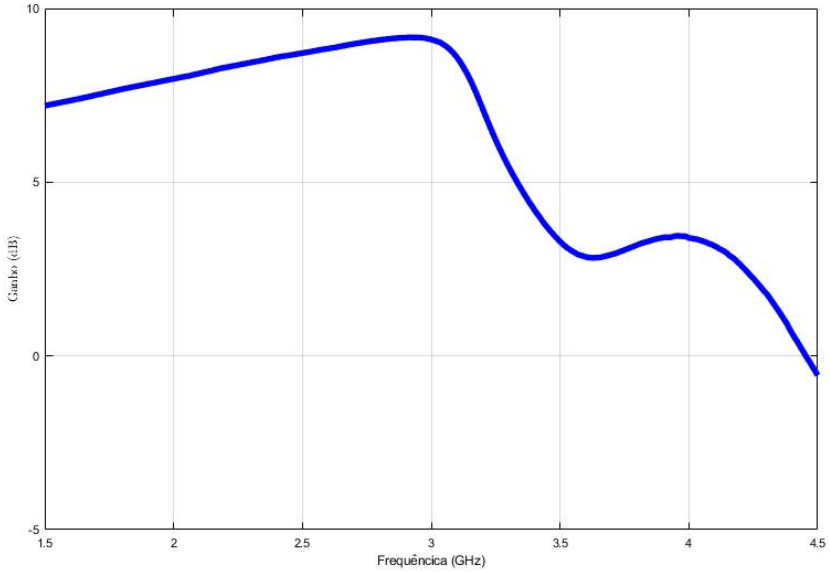
mostra que a antena é adequada para aplicações GPR pois não distorce significativamente o sinal de entrada.

Embora o problema de antena definido em (5.2) seja abordado em (AFRICANO et al., 2020), não é possível realizar uma comparação detalhada entre a solução apresentada na Tabela 7 e a solução de referência obtida em (AFRICANO et al., 2020), uma vez que o referido artigo trata este problema com uma abordagem paramétrica. Outro detalhe é que o método numérico empregado em (AFRICANO et al., 2020) para estimar as características da antenna é o HFSS (*High-Frequency Structure Simulator*). Entretanto, consegue-se ressaltar as distinções entre as soluções a partir das curvas associadas a solução de referência apresentadas na Figura 14. Em relação ao parâmetro $s11$, a antena identificada na Tabela 7 melhora os resultados da Figura 14a na faixa de 1.5 GHz até 2.6 GHz, conforme observado na Figura 13a, demostrando que a antena obtida pelo SAEAa apresenta um melhor casamento de impedância em toda a faixa de interesse. Por outro lado, antena de referência apresenta um ganho superior ao da antena na Tabela 7 em quase toda a faixa de interesse. Para visualizar essa informação a Figura 15 apresenta as curvas de ganho associadas a cada antena. Os gráficos de sinal recebido e potência do sinal são utilizados aqui apenas para comparações de largura do sinal, uma vez que os métodos numéricos empregados são diferentes e resulta em diferenças de amplitude desses sinais. Especificamente, a largura do sinal obtida na referência é $0.8\,ns$, enquanto que a largura de sinal da antena na Tabela 7 é $0.97\,ns$. Isto indica que a antena retornada pelo SAEAa é um pouco mais dispersiva do que a de referência. Entretanto, o valor obtido encontra-se dentro da largura de pulso máxima permissível de $3.46\,ns$ apresentada na Tabela 6. Dessa forma, pode-se concluir que a antena obtida pelo SAEAa também representa uma boa alternativa para aplicações GPR.

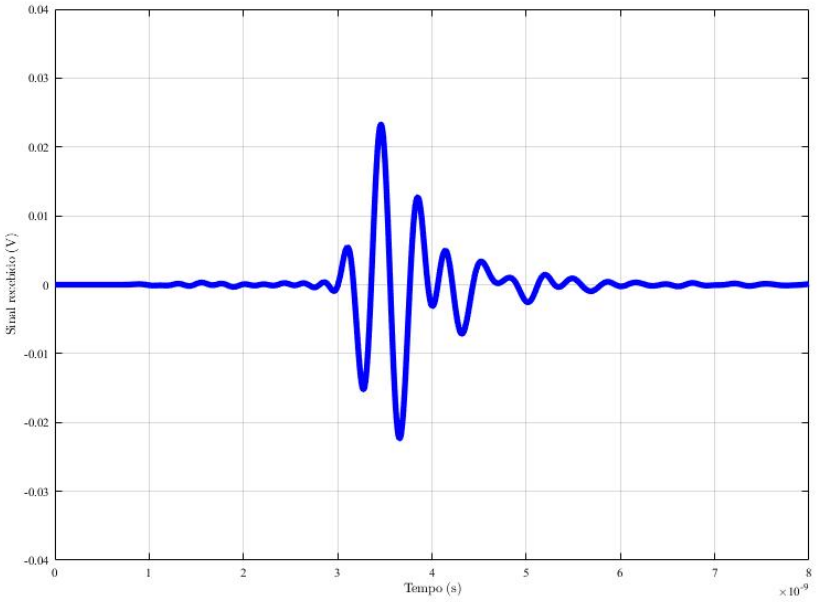
É importante apresentar também informações do custo computacional da aplicação do SAEAa no problema em (5.3). Observou-se neste problema que a média de tempo exigido na avalição de uma solução em $\mathrm{y}(\cdot)$ é $2153.8\,s$. Considerando também que a média de tempo de construção/atualização de metamodelo neste problema é $0.2500s$, é possível estabelecer uma comparação do custo computacional de avaliação de função a cada iteração



(a) s11

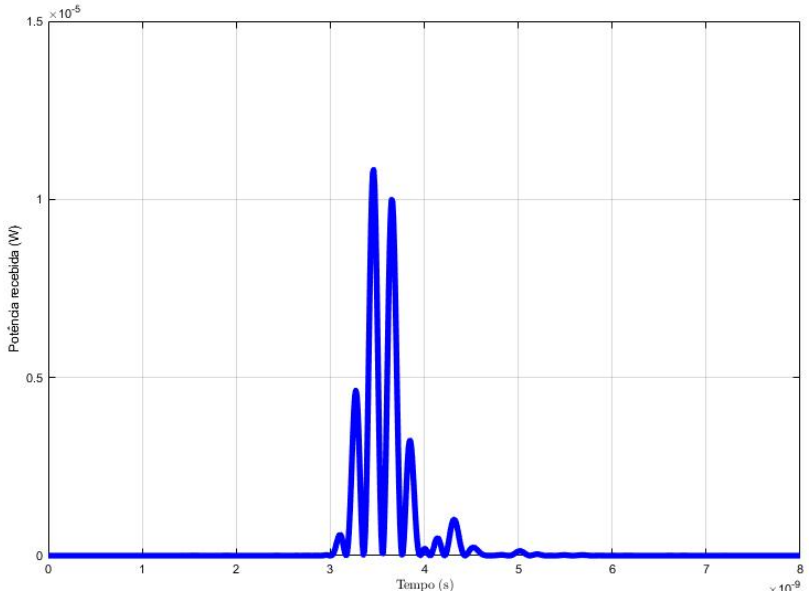


(b) Ganho



(c) Sinal Recebido $1m$

Figura 14 – Curvas da solução apresentada em (AFRICANO et al., 2020).



(d) Potência recebida

Figura 14 – Curvas da solução apresentada em (AFRICANO et al., 2020).

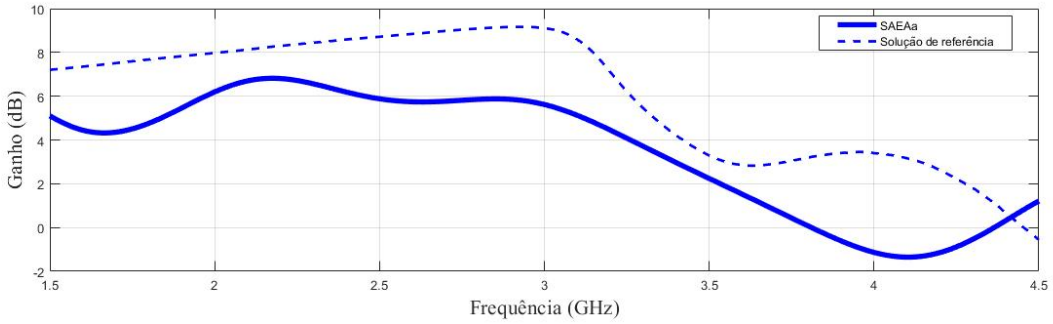


Figura 15 – Curvas de ganho da solução obtida pelo SAEAa e da solução apresentada em (AFRICANO et al., 2020).

do SAEAa com e sem o uso de metamodelo. Para o SAEAa sem metamodelo, a média de tempo exigido para avaliar *npop* soluções em y(·) por iteração é $(npop \times 2153.8)s$, enquanto que para o SAEAa na forma original o custo de avaliação de função por iteração é $(0.2500 + 2153.8)\, s$, uma vez que apenas uma solução é avaliada em y(·) por iteração (todas as demais são avaliadas no metamodelo construído). Assim, observou-se neste problema uma considerável redução de recurso computacional a partir do emprego do SAEA proposto neste trabalho.

5.7 Conclusão

Neste capítulo apresentou-se a validação do método SAEAa proposto neste trabalho, o qual mostrou um melhor desempenho, em termos de qualidade de solução e custo computacional, em comparação com as estratégias da literatura TASEA e GSGA. Os resultados sugerem que o método SAEAa representa uma alternativa promissora a ser aplicada no contexto de problemas envolvendo alto custo computacional associado a avaliação de função objetivo. Realizou-se também uma comparação qualitativa com os métodos GEPEME, SA-COSO e SHPSO, na qual observou-se uma indicativa de melhores resultados do SAEAa em relação ao GEPEME e SA-COSO. Por outro lado, para o SAEAa e SHPSO observou-se que não é possível inferir qual método se destaca em relação ao outro a partir da comparação qualitativa realizada. Considerou-se também a aplicação do SAEAa na otimização de um projeto de antena para radar de subsolo (GPR), no qual deseja-se obter os parâmetros apropriados que a antena deve apresentar para se adaptar bem à aplicação. A validação da solução retornada pelo SAEAa mostrou que a antena é adequada para aplicações em GPR. Além disso, foi possível evidenciar uma considerável redução de recurso computacional (tempo) a partir do emprego de um metamodelo.

Considerou-se pertinente analisar também o comportamento de variações do método proposto e do método TASEA. Especificamente para o método TASEA, considerou-se a variação TASEArbf que resulta de uma adaptação na estrutura do TASEA para empregar um metamodelo RBF. Para o SAEAa considerou-se as variações SAEAok e SAEArbf, as quais empregam o método de redução dimensional adotado no TASEA na etapa de construção de metamodelos OK e RBF, respectivamente. Nessa análise observou-se que a configuração SAEArbf representa uma alternativa melhor que o método da TASEA, uma vez que não existe diferença estatisticamente significativa entre esses métodos com relação ao indicador de qualidade de solução. Além disso, o SAEArbf apresentou um menor custo computacional que o TASEA.

No contexto de SAEAs, o custo computacional desse tipo de estratégia é um fator relevante a ser considerado, uma vez que a ideia é aplicar tais métodos em problemas que envolvem alto custo computacional associado a avaliação de função objetivo. Em um SAEA, o custo computacional é influenciado principalmente pelo tipo de metamodelo empregado. Dependendo da quantidade de parâmetros do metamodelo, no caso do *Kriging* é usual definir o número de parâmetros igual ao número de variáveis, o custo computacional do metamodelo resulta em um aumento considerável do tempo de execução do SAEA. Isso explica estratégias da literatura que empregam metamodelo do tipo *Kriging* com número de parâmetros menor que o número de variáveis do problema a ser otimizado. São exemplos estratégias que empregam o método de reduçaõ dimensional ou *Direct Method*. Por outro lado, a forma em que se emprega o metamodelo também influência nesse aspecto, como é o caso do GSGA que emprega metamodelos de baixo custo computacional. Entretanto, a

construção de vários metamodelos RBF por iteração resultou em um maior tempo médio de execução por iteração dessa estratégia.

É relevante mencionar ainda que abordou-se aspectos e parâmetros relacionados às toolbox adotadas, de forma a permitir a replicação das estratégias implementadas neste trabalho. Nesse sentido, buscou-se apresentar também em detalhes todos os parâmetros associadas às estratégias investigadas nesse capítulo.

6 Conclusões

O presente trabalho apresentou uma visão geral sobre a aplicação de estratégias de otimização que empregam metamodelos *Kriging* e RBF em sua estrutura na resolução de SOOPs com restrição de caixa, que possuem alguma limitação de tempo ou orçamento computacional associados a avaliação de função objetivo. Nesse sentido, o Capítulo 1 contextualizou de forma geral o uso dessas estratégias, que frequentemente são aplicadas para tratar problemas práticos envolvendo grande número de variáveis e uso de um modelo de simulação quando não existe uma forma analítica de avaliação da função objetivo. Nesse contexto, o uso de metamodelo no processo de otimização visa reduzir o número de avaliações realizadas na função que representa o problema.

O Capítulo 2 apresentou inicialmente os conceitos necessários para construir aproximações globais via os metamodelos *Kriging* e RBF, considerando também o mecanismo adotado para melhorar a acurácia de predição. Embora não tenha sido tratada a resolução de SOOPs no capítulo em questão, a seção dedicada a descrição do algoritmo EGO direciona um caminho para explorar estratégias de otimização baseadas no conceito de aproximação global e adaptação sequencial de amostra, via função *Expected Improvement* (ou outras funções *prescreening*), relatadas na literatura.

Diferente do Capítulo 2, onde o foco é a construção de um metamodelo global com uma certa acurácia, os SAEAs apresentados no Capítulo 3 trabalham com metamodelos que não representam necessariamente uma aproximação adequada em todo o espaço de busca, mas que permitem obter informações sobre soluções que sejam interessantes para serem avaliadas na função mérito do problema. Nessas estratégias abordadas, a região de aproximação é redefinida a medida que a população do algoritmo evolui, de forma a garantir que quando necessário realizar predições com o metamodelo, estas ocorram nas proximidades da amostra com a qual o metamodelo foi construído. Observou-se que os SAEAs são descritos na literatura como alternativa para reduzir o número de avaliações na função objetivo exigidas pelos EAs, considerando a aplicação em problemas computacionalmente caros. A construção de uma aproximação global de problemas com essa característica naturalmente exige uma amostra com muitos pontos, o que representa um consumo considerável de tempo.

A partir dessa visão geral sobre SAEAs apresentada no Capítulo 3, desenvolveu-se no Capítulo 4 o método proposto neste trabalho. Em sua estrutura, o SAEAa emprega diferentes variações de um operador de mutação híbrido que combina operadores de naturezas distintas. Observou-se na literatura que o uso de mais de um operador tem como finalidade melhorar a diversidade no processo evolutivo. No caso do operador adotado, as

variações do SL-PSO/DE agregam ao SAEAa uma manutenção de diversidade e maior pressão seletiva. O método considera também uma autoadaptação de parâmetro, associado à operadores de mutação, e define a escolha entre os diferentes operadores de acordo com o *status* de convergência do algoritmo. Nesse sentido, o armazenamento de informações sobre o aprendizado do algoritmo tem um papel fundamental para definir o momento de alterações entre operadores de variação, porque permite estimar informações sobre a diversidade da população. Em relação ao metamodelo, o SAEAa emprega um OK definido com parâmetro unidimensional, o que resulta em um menor custo de construção/atualização de metamodelo.

Apresentou-se no Capítulo 5 a validação do SAEAa, o qual foi aplicado na otimização de um conjunto de funções analíticas. A análise de resultados mostrou que o método proposto apresentou um melhor desempenho, em termos de qualidade de solução e custo computacional, em comparação com as estratégias da literatura TASEA e GSGA. Os resultados sugerem que o método SAEAa representa uma alternativa promissora a ser aplicada no contexto de problemas envolvendo alto custo computacional associado a avaliação de função objetivo. Realizou-se também uma comparação qualitativa com os métodos GEPEME, SA-COSO e SHPSO, na qual observou-se uma indicativa de melhores resultados do SAEAa em relação ao GEPEME e SA-COSO. Por outro lado, para o SAEAa e SHPSO observou-se que não é possível inferir qual método se destaca em relação ao outro a partir da comparação qualitativa realizada. Aplicou-se também a estratégia proposta na otimização de um projeto de antena para radar de subsolo (GPR). A validação da solução retornada pelo SAEAa mostra que a antena se adapta bem a aplicações em GPR. Além disso, foi possível constatar uma considerável redução de recurso computacional (tempo) a partir do emprego do SAEA proposto.

É relevante mencionar que buscou-se abordar também nos Capítulos 4 e 5 questões que, se não forem devidamente esclarecidas, inviabilizam a replicação do SAEAa e demais estratégias comparadas com este método. Um exemplo que ilustra esta situação é a necessidade de manutenção de um conjunto de armazenamento de soluções avaliadas na função mérito, o qual é observado em diferentes estruturas de SAEAs. Embora esse conjunto influencie diretamente na qualidade do metamodelo e diversidade da população, muitos trabalhos na literatura não tratam em detalhes o processo de atualização desse conjunto. Outra questão que frequentemente não é tratada em detalhes é relacionada ao uso de toolboxs e respectivos parâmetros adotados. Nesse sentido, a descrição do SAEAa considerando tais aspectos pode auxiliar a compreender melhor o funcionamento de outros SAEAs existentes na literatura.

Outra característica relevante neste trabalho é o detalhamento de como incorporar operadores de mutação com estruturas distintas, como é o caso do SL-PSO e DE. Em relação ao uso de mais de um operador de mutação em SAEAs, observou-se na literatura

o emprego de diferentes variações de operador do tipo DE. Convém observar também que o SAEAa possui vários parâmetros para os quais definiu-se valores sem investigar o efeito em função do número de varáveis. Nesse contexto, um ajuste adequado de parâmetros do SAEAa pode melhorar o desempenho do método para uma determinada classe de problemas. Entretanto, é importante perceber que mesmo na falta desse ajuste, o método proposto dominou as demais abordagens avaliadas. É importante ressaltar que considerou-se a validação do SAEAa em problemas analíticos envolvendo de 10 a 200 variáveis, no entanto, o projeto de antena considerado como aplicação prática neste trabalho possui $n = 15$ variáveis. Nesse sentido, reconhece-se a necessidade de considerar a aplicação do SAEAa em um problema prático envolvendo um maior número de variáveis.

Finalmente, o desenvolvimento deste trabalho permitiu identificar outros aspectos que podem ser abordados a partir do método proposto. Nesse sentido, os tópicos a seguir apresentam propostas de trabalhos futuros:

- Explorar o desempenho de SAEAs na resolução de problemas práticos com maior dimensão. Embora diferentes trabalhos apresentem a validação de SAEAs em problemas analíticos com número de variáveis entre 30 e 100 variáveis, observou-se que em geral a aplicação em problemas práticos envolve um número de variáveis no máximo igual a 20. É importante destacar também a necessidade de desenvolver SAEAs para otimização de problemas de maior dimensão, i.e., $n > 200$.

- Investigar o desempenho de SAEAs considerando diferentes funções *prescreening*, tais como *Lower Confidence Bond* e *Probability of Improvement*.

- Investigar o efeito de definir como *prescreening* escolher a solução com melhor valor de função aproximado obtido via *Fitness Estimation Strategy* (FES). Especificamente, investigar o desempenho do SAEAa ao definir a função *prescreening* conforme o tipo de operador empregado em cada iteração; por exemplo, *prescreening* via *Fitness Estimation Strategy* para o operador SL-PSO e *prescreening* via *Expected Improvement* para operadores DE. Nesse sentido, realizar uma revisão detalhada sobre contribuições relacionadas a *Fitness Estimation Strtategy* no contexto de SAEAs.

- Investigar o efeito do operador SL-PSO/DE em outros SAEAs existentes na literatura, considerando incorporar também o metamodelo *Kriging* unidimensional nos SAEAs existentes que empregam este tipo de metamodelo.

Referências

ACTON, F. S. *Numerical Methods That Work*. [S.l.]: The Mathematical Association of America, 1990. Citado na página 39.

AFRICANO, M. et al. Ground-penetrating radar antenna design for homogeneous and low-loss dielectric multilayer media. *Journal of Microwaves, Optoelectronics and Electromagnetic Applications*, v. 19, n. 2, p. 137–151, 2020. Citado 7 vezes nas páginas 15, 16, 90, 92, 93, 94 e 95.

ALLMENDINGER, R. et al. Surrogate-assisted multicriteria optimization: Complexities, prospective solutions, and business case. *Journal of Multi-Criteria Decision Analysis*, Wiley Online Library, v. 24, n. 1-2, p. 5–24, 2017. Citado 2 vezes nas páginas 59 e 60.

BIRATTARI, M. *On the Estimation of the Expected Performance of a Metaheuristic on a Class of Instances: How many instances, how many runs?* [S.l.], 2004. Citado na página 82.

BROWNLEE, A. E.; WRIGHT, J. A. Constrained, mixed-integer and multi-objective optimisation of building designs by nsga-ii with fitness approximation. *Applied Soft Computing*, Elsevier, v. 33, p. 114–126, 2015. Citado na página 60.

BÜCHE, D.; SCHARAUDOLPH, N. N.; KOUMOUNTSAKOS, P. Accelerating evolutionary algorithms with Gaussian process fitness function models. *IEEE Transactions on Systems Man and Cybernetics, Part C (Applications and Reviews)*, v. 35, n. 2, p. 183–194, 2005. Citado 5 vezes nas páginas 43, 44, 46, 47 e 48.

CAI, L. G. X.; LI, X. Efficient generalized surrogate-assisted evolutionary algorithm for high-dimensional expensive problems. *IEEE Transactions on Evolutionary Computation*, IEEE, v. 24, n. 2, p. 365–379, 2019. Citado 6 vezes nas páginas 23, 44, 56, 57, 70 e 77.

CHAFEKAR, D. et al. Multiobjective ga optimization using reduced models. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, IEEE, v. 35, n. 2, p. 261–265, 2005. Citado na página 60.

CHENG, R.; JIN, Y. A social learning particle swarm optimization algorithm for scalable optimization. *Information Sciences*, Elsevier, v. 291, p. 43–60, 2015. Citado 2 vezes nas páginas 62 e 81.

CHOUDHURY, A.; NAIR, P. B.; KEANE, A. J. A data parallel approach for large-scale Gaussian process modeling. In: *SIAM International Conference on Data Mining*. [S.l.: s.n.], 2002. p. 95–111. Citado na página 50.

COELLO, C. C.; LAMONT, G. B.; van VELDHUIZEN, D. A. *Evolutionary Algorithms for Solving Multi-objective Problems*. 2. ed. [S.l.]: Springer, 2007. (Genetic and Evolutionary Computation). Citado na página 21.

COUCKUYT, I.; DHAENE, T.; DEMEESTER, P. *ooDACE toolbox A Matlab Kriging toolbox: Getting started*. 3rd june. ed. [S.l.], 2013. Disponível em: <http://sumo.intec.ugent.be/ooDACE>. Citado na página 77.

COUCKUYT, I. et al. Blind kriging: Implementation and performance analysis. *Advances in Engineering Software*, v. 49, p. 1–13, 2012. Citado na página 73.

CRAWLEY, M. J. *The R Book*. 1st. ed. [S.l.]: Wiley Publishing, 2007. Citado na página 82.

DAS, S.; SUGANTHAN, P. N. Differential evolution: A survey of the state-of-the-art. *IEEE Transactions on Evolutionary Computation*, v. 15, n. 1, p. 4–31, February 2011. Citado 3 vezes nas páginas 53, 63 e 64.

DICK, G. The utility of scale factor adaptation in differential evolution. In: IEEE. *Congress on Evolutionary Computation*. [S.l.], 2010. p. 1–8. Citado na página 65.

DRÉO, J. et al. *Methaeuristics for Hard Optimization: Methods and Case Studies*. 1. ed. [S.l.]: Springer, 2006. Citado na página 21.

ELSHERBENI, A.; DEMIR, V. The finite difference time domain method for electromagnetics: With matlab simulations. In: . [S.l.: s.n.], 2009. Citado na página 91.

EMMERICH, M. T.; DEUTZ, A. H.; KLINKENBERG, J. W. Hypervolume-based expected improvement: Monotonicity properties and exact computation. In: IEEE. *2011 IEEE Congress of Evolutionary Computation (CEC)*. [S.l.], 2011. p. 2147–2154. Citado na página 59.

EMMERICH, M. T. M.; GIANNAKOGLOU, K. C.; NAUJOKS, B. Single- and multiobjective evolutionary optimization assisted by Gaussian random field metamodels. *IEEE Transactions on Evolutionary Computation*, v. 10, n. 4, p. 421–439, 2006. Citado 7 vezes nas páginas 21, 23, 43, 44, 45, 47 e 48.

FINKEL, D. E. *DIRECT Optimization Algorithm User Guide*. [S.l.], 2003. Disponível em: <https://ctk.math.ncsu.edu/Finkel_Direct>. Citado 2 vezes nas páginas 70 e 77.

FORD, W. *Numerical Linear Algebra with Aplications Using MATLAB*. 1. ed. [S.l.]: Academic Press, 2014. Citado na página 34.

FORRESTER, A. I. J.; SÓBESTER, A.; KEANE, A. J. *Engineering Design via Surrogate Modelling: A Practical Guide*. [S.l.]: John Wiley & Sons, 2008. Citado 8 vezes nas páginas 32, 33, 34, 35, 38, 39, 47 e 48.

GINSBOURGER, D.; RICHE, R. L.; CARRARO, L. Kriging is well-suited to parallelize optimization. In: TENNE, Y.; GOH, C.-K. (Ed.). *Computational Intelligence in Expensive Optimization Problems*. [S.l.]: Springer, 2010, (Adaptation, Learning and Optimization, v. 2). cap. 6, p. 131–162. Citado 2 vezes nas páginas 34 e 35.

GONG, W.; ZHOU, A.; CAI, Z. A multioperator search strategy based on cheap surrogate models for evolutionary optimization. *IEEE transactions on Evolutionary Computation*, IEEE, v. 19, n. 5, p. 746–758, 2015. Citado na página 53.

HAN, Z.-H.; ZHANG, K.-S. Surrogate-based optimization. In: ROEVA, O. (Ed.). *Real-World Applications of Genetic Algorithms*. [S.l.]: InTech, 2012. cap. 17, p. 343–362. Citado 4 vezes nas páginas 32, 41, 46 e 48.

HAO, W.; SHAOPING, W.; TOMOVIC, M. M. Modified sequential kriging optimization for multidisciplinary complex product simulation. *Chinese Journal of Aeronautics*, v. 23, n. 5, p. 616–622, 2010. Citado 3 vezes nas páginas 35, 41 e 47.

HARVILLE, D. A. *Matrix Algebra From a Statistician's Perpective*. [S.l.]: Springer, 2008. Citado na página 34.

JEFFREYS, H. *Theory of Probability*. 3. ed. [S.l.]: Oxford University Press, 2003. (Oxford Classic Texts in The Physical Science). Citado na página 39.

JIN, R.; CHEN, W.; SUDJIANTO, A. On sequential sampling for global metamodeling in engineering design. In: *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. Montreal, Canada: American Society of Mechanical Engineers (ASME), 2002. p. 10. Citado 2 vezes nas páginas 35 e 36.

JIN, Y. A comprehensive survey of fitness approximation in evolutionary computation. *Soft Computing*, v. 9, n. 1, p. 3–12, 2005. Citado 2 vezes nas páginas 44 e 46.

JIN, Y.; BRANKE, J. Evolutionary optimization in uncertain environments - a survey. *IEEE Transactions on Evolutionary Computation*, v. 9, n. 3, p. 303–317, 2005. Citado 2 vezes nas páginas 44 e 45.

JIN, Y.; OLHOFER, M.; SENDHOFF, B. On evolutioary optimization with approximate fitness function. In: *Genetic and Evolutionary Computation Conference*. [S.l.: s.n.], 2000. p. 786–793. Citado 2 vezes nas páginas 44 e 45.

JIN, Y.; OLHOFER, M.; SENDHOFF, B. Managing approximate models in evolutionary aerodynamc design optimization. In: *Congress on Evolutionary Computation*. [S.l.: s.n.], 2001. p. 592–599. Citado na página 46.

JONES, D. R.; SCHONLAU, M.; WELCH, W. J. Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, v. 13, n. 4, p. 455–492, 1998. Citado 5 vezes nas páginas 21, 29, 35, 38 e 39.

JOSEPH, V. R.; HUNG, Y.; SUDJIANTO, A. Blind kriging: A new method for developing metamodels. *Jornal of Mechanical Design*, v. 130, p. 031102(1–7), 2008. Citado na página 35.

JēKABSONS, G. *RBF: Radial Basis Function interpolation for MATLAB/OCTAVE*. Version 1.1. [S.l.], 2009. Disponível em: <http://www.cs.rtu.lv/jekabsons/regression.html>. Citado na página 77.

KNOWLES, J. ParEGO: a hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation*, IEEE, v. 10, n. 1, p. 50–66, 2006. Citado 2 vezes nas páginas 59 e 60.

LIM, D. et al. Generalizing surrogate-assisted evolutionary computation. *IEEE Transactions on Evolutionary Computation*, IEEE, v. 14, n. 3, p. 329–355, 2009. Citado 3 vezes nas páginas 50, 59 e 60.

LIU, B.; ZHANG, Q.; GIELEN, G. G. E. A Gaussian process surrogate model assisted evolutionary algorithm for medium scale expensive optimization problems. *IEEE Transactions on Evolutionary Computation*, v. 18, n. 2, p. 180–192, 2014. Citado 14 vezes nas páginas 21, 23, 31, 43, 44, 45, 47, 51, 52, 55, 58, 63, 81 e 85.

LOPHANEV, S. N.; NIELSEN, H. B.; SØNDERGAARD, J. *DACE - A MATLAB Kriging Toolbox*. [S.l.], 2002. Citado na página 35.

MA, H. et al. Update-based evolution control: A new fitness approximation method for evolutionary algorithms. *Engineering Optimization*, v. 47, n. 9, p. 1177–1190, 2015. Citado 6 vezes nas páginas 35, 45, 52, 53, 62 e 63.

MACKAY, D. J. C. *Introduction to Gaussian Process*. 1998. Cambridge University. Disponível em: <http://www.inference.org.uk/mackay/gpB.pdf>. Citado na página 31.

MENDES, M. H. S. et al. A surrogate genetic programming based model to facilitate robust multi-objective optimization: A case study in magnetostatics. *IEEE Transactions on Magnetics*, v. 49, n. 5, p. 2065–2068, 2013. Citado na página 21.

MONTGOMERY, D. C.; PECK, E. A.; VINING, G. G. *Introduction to Regression to Linear Regression Analysis*. 4. ed. [S.l.]: John Wiley & Sons, Inc., 2006. Citado na página 30.

MONTGOMERY, D. C.; RUNGER, G. C. *Applied Statistics and Probability for Engineers*. [S.l.]: Wiley, 2013. Citado na página 82.

PILÁT, M.; NERUDA, R. Aggregate meta-models for evolutionary multiobjective and many-objective optimization. *Neurocomputing*, Elsevier, v. 116, p. 392–402, 2013. Citado na página 60.

Pilát, M.; Neruda, R. An evolutionary strategy for surrogate-based multiobjective optimization. In: *2012 IEEE Congress on Evolutionary Computation*. [S.l.: s.n.], 2012. p. 1–7. Citado 2 vezes nas páginas 59 e 60.

PRICE, K. V.; STORN, R. M. *Differential Evolution: A Practical Approach to Global Optimization*. Germany: Springer, 2005. Citado na página 63.

QIN, A. K.; SUGANTHAN, P. N. Self-adaptive differential evolution algorithm for numerical optimization. In: IEEE. *Congress on Evolutionary Computation*. [S.l.], 2005. v. 2, p. 1785–1791. Citado 4 vezes nas páginas 53, 63, 64 e 65.

RAO, S. S. *Engineering Optimization: Theory and Practice*. 4. ed. [S.l.]: John Wiley & Sons, 2009. Citado na página 21.

RATLE, A. Accelerating the convergence of evolutionary algorithms by fitness landscape approximation. In: *International Conference on Parallel Problem Solving from Nature*. [S.l.: s.n.], 1998. p. 87–96. Citado na página 46.

REGIS, R. G.; SHOEMAKER, C. A. Local function approximation in evolutionary algorithms for the optimization of costly functions. *IEEE Transactions on Evolutionary Computing*, v. 8, n. 5, p. 490–505, 2004. Citado na página 48.

ROCHA, H. On the selection of the most adequate radial basis function. *Applied mathematical Modelling*, v. 33, n. 3, p. 1573–1583, 2009. Citado na página 36.

ROUSTANT, O.; GINSBOURGER, D.; DEVILLE, Y. DiceKriging, DiceOptim: Two R packages for the analysis of computer experiments by kriging-based metamodeling and optimization. *Journal of Statistical Software*, v. 51, n. 1, p. 1–55, 2012. Citado 3 vezes nas páginas 32, 34 e 39.

SACKS, J. et al. Design and analysis of computer experiments. *Statistical Science*, v. 4, n. 4, p. 409–423, 1989. Citado 3 vezes nas páginas 21, 29 e 32.

SAMMON, J. J. W. A nonlinear mapping for data structure analysis. *IEEE Transactions on Computers*, IEEE, C-18, n. 5, p. 401–409, 1969. Citado 3 vezes nas páginas 52, 55 e 58.

SANTNER, T. J.; WILLIANS, B. J.; NOTZ, W. I. *The Design and Analysis of Computer Experiments*. 1. ed. [S.l.]: Springer, 2003. (Springer Series in Statistics). Citado na página 34.

SCHONLAU, M. *Computer Experiments and Global Optimization*. Tese (Doutorado) — University of Waterloo, 1997. Citado 5 vezes nas páginas 21, 29, 32, 34 e 39.

SUGANTHAN, P. N. et al. *Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization*. [S.l.], 2005. Citado 2 vezes nas páginas 78 e 79.

SUN, C. et al. Surrogate-assisted cooperative swarm optimization of high-dimensional expensive problems. *IEEE Transactions on Evolutionary Computation*, v. 21, n. 4, p. 644–660, 2017. Citado 8 vezes nas páginas 21, 23, 43, 47, 54, 62, 69 e 85.

SURJANOVIC, S.; BINGHAM, D. *Virtual Library of Simulation Experiments: Test Functions and Datasets*. 2018. Retrieved April 27, 2018, from <http://www.sfu.ca/~ssurjano>. Citado na página 78.

TABATABAEI, M. et al. A survey on handling computationally expensive multiobjective optimization problems using surrogates: non-nature inspired methods. *Structural and Multidisciplinary Optimization*, Springer, v. 52, n. 1, p. 1–25, 2015. Citado 2 vezes nas páginas 59 e 60.

TORCZON, V.; TROSSET, M. W. *Using Approximation to Accelerate Engineering Design Optimization*. [S.l.], 1998. Citado 2 vezes nas páginas 46 e 48.

TRAVASSOS, X. L. et al. A review of ground penetrating radar antenna design and optimization. *Journal of Microwaves, Optoelectronics and Electromagnetic Applications*, v. 17, n. 3, p. 385–402, 2018. Citado na página 90.

VALADãO, M. A. C.; BATISTA, L. S. A comparative study on surrogate models for SAEAs. *Optim. Lett.*, v. 14, p. 2595–2614, abr. 2020. Citado 2 vezes nas páginas 71 e 80.

VESANTO, J. et al. *SOM Toolbox for Matlab 5*. [S.l.], 2000. Citado na página 78.

VIANA, F. A. C. *SURROGATES Toolbox User's Guide*. Version 2.1. [S.l.], 2010. Disponível em: <http://sites.google.com/site/felipeacviana/surrogatestoolbox>. Citado na página 77.

XIA, B. et al. A numerically efficient muli-objective optimization algorithm: Combination of dynamic taylor kriging and differential evolution. *IEEE Transactions on Magnetics*, v. 51, n. 3, p. 1–4, March 2015. Citado 2 vezes nas páginas 35 e 41.

XIA, B.; REN, Z.; KOH, C. seop. Comparative study on kriging surrogate models for metaheuristic optimization of multidimensional eletromagnetic problems. *IEEE Transactions on Magnetics*, v. 51, n. 3, p. 1–4, March 2015. Citado na página 41.

YANG, C. et al. Offline data-driven multiobjective optimization: Knowledge transfer between surrogates and generation of final solutions. *IEEE Transactions on Evolutionary Computation*, IEEE, v. 24, n. 3, p. 409–423, 2020. Citado na página 59.

YANG, Z. et al. Two-layer adaptive surrogate-assisted evolutionary algorithm for high-dimensional computationally expensive problems. *Journal of Global Optimization*, Springer, v. 74, n. 2, p. 327–359, 2019. Citado 12 vezes nas páginas 23, 44, 55, 58, 63, 65, 66, 69, 70, 73, 77 e 81.

YANG, Z.; TANG, K.; YAO, X. Self-adaptive differential evolution with neighborhood search. In: IEEE. *Congress on Evolutionary Computation*. [S.l.], 2008. p. 1110–1116. Citado 3 vezes nas páginas 65, 66 e 67.

YU, H. et al. Surrogate-assisted hierarchical particle swarm optimization. *Information Sciences*, Elsevier, v. 454, p. 59–72, 2018. Citado 2 vezes nas páginas 54 e 85.

YUN, Y.; NAKAYAMA, H. Utilizing expected improvement and generalized data envelopment analysis in multi-objective genetic algorithms. *Journal of Global Optimization*, Springer, v. 57, n. 2, p. 367–384, 2013. Citado na página 60.

ZHANG, Q. et al. Expensive multiobjective optimization by MOEA/D with gaussian process model. *IEEE Transactions on Evolutionary Computation*, IEEE, v. 14, n. 3, p. 456–474, 2009. Citado na página 59.

ZHAO, L.; CHOI, K. K.; LEE, I. Metamodeling method using dynamic kriging for design optimization. *AIAA Journal*, v. 49, n. 9, p. 2034–2046, 2011. Citado 3 vezes nas páginas 32, 33 e 41.

ZHOU, Z. et al. Combining global and local surrogate models to accelerate evolutionary optimization. *IEEE Transactions on Systems Man and Cybernetics, Part C (Applications and Reviews)*, v. 37, n. 1, p. 66–76, 2007. Citado 3 vezes nas páginas 23, 47 e 49.

Apêndices

APÊNDICE A – Tabelas e Figuras

Tabela 8 – Média e desvio padrão do valor da melhor solução retornada por cada estratégia para todos os problemas e número de variáveis.

	SAEAok		SAEArbf		TASEArbf	
n	Mean	Std.	Mean	Std.	Mean	Std.
Sphere						
10	0e+00	0e+00	0e+00	0e+00	1.07e-04	7.26e-05
20	1.4e-03	5.94e-03	9e-07	1.55e-06	5.22e-02	2.85e-02
30	7.84e-05	2.39e-04	1.46e-05	1.29e-05	4.43e-01	2.23e-01
50	1.51e-01	3.39e-01	1.14e-04	9.59e-05	5.86e+00	2.07e+00
100	3.86e-01	7.99e-01	1.87e-03	1.88e-03	1.02e+02	3.6e+01
200	5.85e+00	5.77e+00	3.09e+00	5.68e+00	8.67e+02	1.23e+02
Differential Powers						
10	0e+00	0e+00	0e+00	0e+00	6.34e-05	3.79e-05
20	0e+00	0e+00	0e+00	0e+00	2.7e-04	2.39e-04
30	0e+00	0e+00	0e+00	0e+00	2.15e-04	1.35e-04
50	0e+00	0e+00	0e+00	0e+00	3.36e-04	3.78e-04
100	0e+00	0e+00	0e+00	0e+00	5.9e-03	1.59e-02
200	0e+00	0e+00	0e+00	0e+00	1.01e+00	2.78e-01
Ellipsoid						
10	0e+00	0e+00	0e+00	0e+00	7.75e-03	4.48e-03
20	2.22e-02	8.22e-02	5.52e-05	5.74e-05	1.16e+00	5.24e-01
30	4.69e-02	1.46e-01	6.94e-03	2.63e-02	1.47e+01	9.8e+00
50	2.36e+00	7.57e+00	6.59e-03	5.06e-03	1.86e+02	9.14e+01
100	6.96e+00	1.31e+01	1.18e-01	4.65e-02	4.2e+03	1.39e+03
200	3.8e+02	4.3e+02	3.66e+02	6.86e+02	8.5e+04	1.19e+04
Rosenbrock						
10	6.03e+00	8.52e-01	6.55e+00	9.73e-01	6.36e+00	6.2e-01
20	1.8e+01	4.51e-01	1.8e+01	4.73e-01	3.18e+01	1.97e+01
30	2.83e+01	5.8e-01	2.82e+01	4.21e-01	1.04e+02	3.91e+01
50	4.91e+01	2.51e+00	4.86e+01	2.18e-01	4.05e+02	9.69e+01
100	9.87e+01	1.11e-01	9.88e+01	9.36e-02	2.94e+03	6.44e+02
200	2.62e+02	8.77e+01	2.18e+02	6.19e+00	6.2e+04	7.57e+03
Ackley						
10	1.93e-05	7.54e-05	1.75e+00	4.31e+00	2.93e-01	3.84e-01
20	1.99e-01	5.07e-01	1.22e+00	3.73e+00	7.49e+00	2.96e+00
30	3.33e-01	1.36e+00	2.03e+00	4.19e+00	1.14e+01	2.31e+00
50	9.11e-01	2.29e+00	1.42e+00	2.93e+00	1.52e+01	1.34e+00
100	7.87e-01	1.3e+00	3.66e+00	2.85e+00	1.82e+01	6.4e-01
200	2.32e+00	2.28e+00	3.93e+00	1.36e+00	2.03e+01	2.17e-01
Griewank						
10	5.5e-07	2.46e-06	1.09e-01	2.22e-01	1.62e-01	8.92e-02
20	5.87e-03	1.84e-02	7.23e-01	1.47e+00	1.25e+00	3.04e-01
30	5.57e-02	1.25e-01	2.69e-01	6.11e-01	5.93e+00	4.09e+00
50	3.94e-02	7.48e-02	1.5e-01	3.03e-01	8.73e+01	5.33e+01
100	1.62e+00	4.85e+00	5.2e+00	7.51e+00	6.02e+02	1.17e+02
200	2.28e+01	1.89e+01	1.04e+02	1.06e+02	2.75e+03	2.71e+02
Rastrigin						
10	0e+00	0e+00	2.3e+00	2.9e+00	1.92e+01	5.47e+00
20	1.24e-04	5.45e-04	9.14e+00	1.04e+01	7.59e+01	2.4e+01
30	1.47e-01	4.73e-01	1.08e+01	1.68e+01	1.43e+02	3.23e+01
50	3.91e+00	7.56e+00	1.34e+01	1.49e+01	3.29e+02	5.65e+01
100	8.43e+00	1.47e+01	1.27e+01	1.52e+01	9.29e+02	1.37e+02
200	2.4e+01	2.05e+01	6.03e+01	3.22e+01	2.84e+03	1.39e+02
Shifted Rotated Rastrigin						
10	1.86e+01	1.03e+01	4.19e+01	1.53e+01	4.62e+01	7.62e+00
20	1.02e+02	3.36e+01	2.16e+02	2.94e+01	1.64e+02	2.78e+01
30	2.19e+02	4.2e+01	3.96e+02	3.46e+01	3.25e+02	6.09e+01
50	6.24e+02	9.06e+01	9.15e+02	9.91e+01	6.94e+02	1.19e+02
100	1.87e+03	1.96e+02	2.3e+03	2.16e+02	1.89e+03	2.29e+02
200	4.13e+03	2.51e+02	4.78e+03	2.08e+02	4.89e+03	3.41e+02

Tabela 9 – Média e desvio padrão do valor da melhor solução retornada por cada estratégia para todos os problemas e número de variáveis.

	SAEAok		SAEArbf		TASEArbf	
n	Mean	Std.	Mean	Std.	Mean	Std.
Rotated Version of Hybrid Composition						
10	9.26e+01	8.06e+01	1.51e+02	3.32e+01	1.82e+02	1e+02
20	3.75e+02	1.46e+02	4.27e+02	1.08e+02	3.93e+02	9.78e+01
30	4.31e+02	1.31e+02	5.96e+02	8.22e+01	4.3e+02	9.63e+01
50	5.71e+02	6.82e+01	7.36e+02	7.42e+01	5.6e+02	1.18e+02
100	7.63e+02	6.35e+01	9.98e+02	1.2e+02	7.82e+02	1.53e+02
200	1.12e+03	9.1e+01	1.3e+03	9.39e+01	9.4e+02	1.33e+02
Rotated Hybrid Composition with narrow						
10	4.63e+02	1.78e+02	7.7e+02	2.22e+02	5.84e+02	2.85e+02
20	1.06e+03	5.2e+01	1.12e+03	5.97e+01	9.18e+02	1.83e+02
30	1.07e+03	1.3e+02	1.13e+03	1.59e+02	1.1e+03	1.12e+02
50	1.19e+03	3.69e+01	1.2e+03	1.35e+02	1.2e+03	7.63e+01
100	9.66e+02	1.52e+02	9.96e+02	1.96e+02	1.34e+03	6.74e+01
200	9.23e+02	3.18e+01	9.45e+02	5.52e+01	1.41e+03	4.01e+01
Shifted Sphere						
10	8.42e-01	2.15e+00	3.24e+03	2.65e+03	1.62e+02	3.13e+02
20	1.52e+03	1.68e+03	2.87e+04	7.8e+03	1e+03	1.3e+03
30	1.01e+04	6.19e+03	5.93e+04	7.58e+03	3.53e+03	3.01e+03
50	2.99e+04	8.32e+03	1.14e+05	1.57e+04	2.2e+04	1.21e+04
100	1.5e+05	2.36e+04	2.86e+05	1.8e+04	1.41e+05	5.16e+04
200	4.49e+05	2.97e+04	6.45e+05	3.34e+04	5.24e+05	6.65e+04
Shifted Rosenbrock						
10	7.31e+03	2.12e+04	1.02e+08	1.19e+08	2.6e+03	3.78e+03
20	9.68e+07	1.55e+08	3.07e+09	2.07e+09	1.19e+08	3.69e+08
30	9.65e+08	7.92e+08	1.41e+10	5.4e+09	2.34e+08	2.71e+08
50	5.49e+09	1.95e+09	3.83e+10	8.08e+09	6.13e+09	4.74e+09
100	4.48e+10	9.71e+09	1.11e+11	1.19e+10	5.4e+10	2.47e+10
200	1.58e+11	1.79e+10	2.71e+11	2.56e+10	3.56e+11	7.03e+10

Tabela 10 – Média e desvio padrão do valor da melhor solução retornada por cada estratégia para todos os problemas e número de variáveis.

	TASEA		SAEAa		GSGA	
n	Mean	Std.	Mean	Std.	Mean	Std.
Sphere						
10	0e+00	0e+00	0e+00	0e+00	2.51e-02	7.66e-03
20	1.75e-04	8.43e-05	3.96e-04	1.16e-03	6.88e-03	8.3e-03
30	1.98e-02	8.52e-03	2.26e-01	6.81e-01	6.02e-04	5.79e-04
50	1.52e+00	6.61e-01	3.32e-01	8.95e-01	2.83e-02	1.41e-02
100	3.72e+01	7.58e+00	9.57e-01	3.01e+00	4.14e+00	1.11e+00
200	6.17e+02	1.57e+02	1.86e-02	6.89e-02	5.48e+01	1.26e+01
Differential Powers						
10	6.8e-06	7.46e-06	0e+00	0e+00	1.4e-06	1.35e-06
20	1.1e-05	5.52e-06	0e+00	0e+00	1.39e-05	1.52e-05
30	1.48e-05	1.06e-05	1e-07	3.08e-07	1.78e-05	1.22e-05
50	3.03e-05	2.56e-05	2.4e-06	1.07e-05	1.32e-05	1.53e-05
100	1.36e-03	1.43e-03	0e+00	0e+00	8.34e-05	8.73e-05
200	8.55e-01	3.39e-01	0e+00	0e+00	7.83e-02	7.41e-02
Ellipsoid						
10	0e+00	0e+00	0e+00	0e+00	1.12e-01	2.13e-02
20	1.32e-02	1.33e-02	9.6e-03	2.98e-02	3.65e-02	3.78e-02
30	1.71e+00	1.72e+00	1.87e-01	7.67e-01	1.14e-01	5.8e-02
50	4.64e+01	1.13e+01	2.51e+01	6.75e+01	4.02e+00	1.43e+00
100	1.76e+03	3.25e+02	5.33e+01	1.12e+02	2.54e+02	4.62e+01
200	5.75e+04	1.73e+04	1.59e-01	3.19e-01	5.52e+03	7.76e+02
Rosenbrock						
10	4.22e+00	7.47e-01	2.34e+00	1.67e+00	6.9e+00	4.4e-01
20	1.81e+01	1.14e+00	1.73e+01	1e+00	1.72e+01	8.11e-01
30	3.5e+01	1.4e+01	2.86e+01	3.96e-01	2.83e+01	6.42e-01
50	1.38e+02	4.53e+01	5.29e+01	1.77e+01	5.21e+01	1.24e+01
100	1.19e+03	3.02e+02	1.13e+02	4.22e+01	3.84e+02	4.45e+01
200	2.75e+04	1.29e+04	1.99e+02	6.56e-01	1.55e+03	1.47e+02
Ackley						
10	1.16e-01	3.56e-01	1.8e-01	4.21e-01	1.09e-01	3.4e-02
20	1.52e+00	8.51e-01	1.59e+00	2.68e+00	3.57e-01	1.81e-01
30	2.65e+00	7.99e-01	1.57e-01	5.17e-01	7.64e-01	3.06e-01
50	7.46e+00	1.57e+00	6.53e-02	1.74e-01	1.63e+00	2.51e-01
100	1.42e+01	1.17e+00	2.09e-01	3.36e-01	4.38e+00	6.17e-01
200	1.98e+01	6.47e-01	9.26e-02	1.5e-01	1.04e+01	6.77e-01
Griewank						
10	3.86e-02	3.18e-02	2.09e-03	5.43e-03	1.06e+00	1.63e-02
20	1.93e-01	9.24e-02	7.06e-03	1.38e-02	1e+00	1.01e-01
30	1.08e+00	6.13e-02	8.65e-02	2.35e-01	9.58e-01	5.06e-02
50	6.46e+00	2.49e+00	1.43e+00	3.89e+00	1.18e+00	4.68e-02
100	1.46e+02	2.94e+01	1.13e-01	4.75e-01	4.42e+01	9.27e+00
200	2.36e+03	5.65e+02	1.52e-02	4.36e-02	5.77e+02	7.9e+01
Rastrigin						
10	8.61e+00	4.27e+00	5.35e-02	2.24e-01	1.37e+01	6.3e+00
20	4.68e+01	2.56e+01	1.28e+00	2.36e+00	3.6e+01	1.46e+01
30	1.37e+02	5.45e+01	1.77e+00	3.89e+00	6.53e+01	1.85e+01
50	3.53e+02	1.06e+02	2.37e+00	5.42e+00	1.03e+02	3.16e+01
100	1.03e+03	6.91e+01	7.2e+00	1.26e+01	4.52e+02	7.11e+01
200	2.71e+03	2.13e+02	5.2e+00	1.49e+01	1.87e+03	2.28e+02
Shifted Rotated Rastrigin						
10	1.11e+01	4.82e+00	1.92e+01	7.82e+00	2.55e+01	1.05e+01
20	1.07e+02	4.81e+01	6.3e+01	2e+01	4.42e+01	1.52e+01
30	2.5e+02	3.83e+01	1.45e+02	3.18e+01	7.16e+01	1.68e+01
50	5.52e+02	4.21e+01	6.23e+02	8.03e+01	1.34e+02	3.2e+01
100	1.51e+03	9.87e+01	2.1e+03	9.1e+01	3.96e+02	6.51e+01
200	4.71e+03	3.6e+02	4.99e+03	2.02e+02	1.42e+03	2.03e+02

Tabela 11 – Média e desvio padrão do valor da melhor solução retornada por cada estratégia para todos os problemas e número de variáveis.

	TASEA		SAEAa		GSGA	
n	Mean	Std.	Mean	Std.	Mean	Std.
Rotated Version of Hybrid Composition						
10	7.27e+01	1.6e+01	8.98e+01	8.15e+01	1.08e+02	2.23e+01
20	2.98e+02	1.23e+02	2.43e+02	1.03e+02	2.73e+02	1.19e+02
30	3.88e+02	1.04e+02	3.59e+02	1.28e+02	3.46e+02	8.82e+01
50	4.89e+02	5.83e+01	5.95e+02	7.38e+01	4.56e+02	5.97e+01
100	5.72e+02	3.91e+01	9.6e+02	1.01e+02	5.12e+02	5.02e+01
200	7.9e+02	7.2e+01	1.34e+03	5.47e+01	5.76e+02	7.89e+01
Rotated Hybrid Composition with narrow						
10	4.66e+02	2.51e+02	4.45e+02	1.94e+02	4.27e+02	1.68e+02
20	9.1e+02	2e+02	9.97e+02	6.09e+01	9.76e+02	1.7e+02
30	1.03e+03	1.3e+02	1.1e+03	8.15e+01	1.04e+03	1.94e+02
50	1.11e+03	5.91e+01	1.2e+03	2.87e+01	1.13e+03	2.75e+01
100	1.24e+03	4.47e+01	9.91e+02	1.76e+02	1.24e+03	2.99e+01
200	1.4e+03	3.82e+01	9.04e+02	1.04e+01	1.26e+03	2.1e+01
Shifted Sphere						
10	1.8e-06	1.24e-06	4.5e-07	2.01e-06	6.23e-01	4.29e-01
20	1.8e-01	3.99e-01	3.97e+00	7.49e+00	6.36e-01	1.29e+00
30	2.03e+01	1.77e+01	9.79e+02	7.76e+02	0e+00	0e+00
50	1.51e+03	1.79e+03	3.31e+04	9.34e+03	6.9e-06	2.88e-05
100	4.36e+04	1.08e+04	2.2e+05	2.19e+04	7.55e+03	1.32e+03
200	4.72e+05	6.55e+04	5.92e+05	3.14e+04	5.9e+04	1.07e+04
Shifted Rosenbrock						
10	7.42e+02	2.08e+03	9.1e+03	1.49e+04	7.05e+02	1.19e+03
20	8.42e+03	1.93e+04	2.64e+06	2.58e+06	3.29e+03	2.67e+03
30	6.04e+06	2.56e+07	1.47e+08	9.92e+07	9.7e+02	1.51e+03
50	1.09e+08	1.61e+08	8.23e+09	3.83e+09	5.44e+03	7.76e+03
100	9.85e+09	3.7e+09	7.53e+10	1.15e+10	1.46e+09	7.37e+08
200	3.49e+11	1.07e+11	2.35e+11	1.1e+10	1.39e+10	3.89e+09

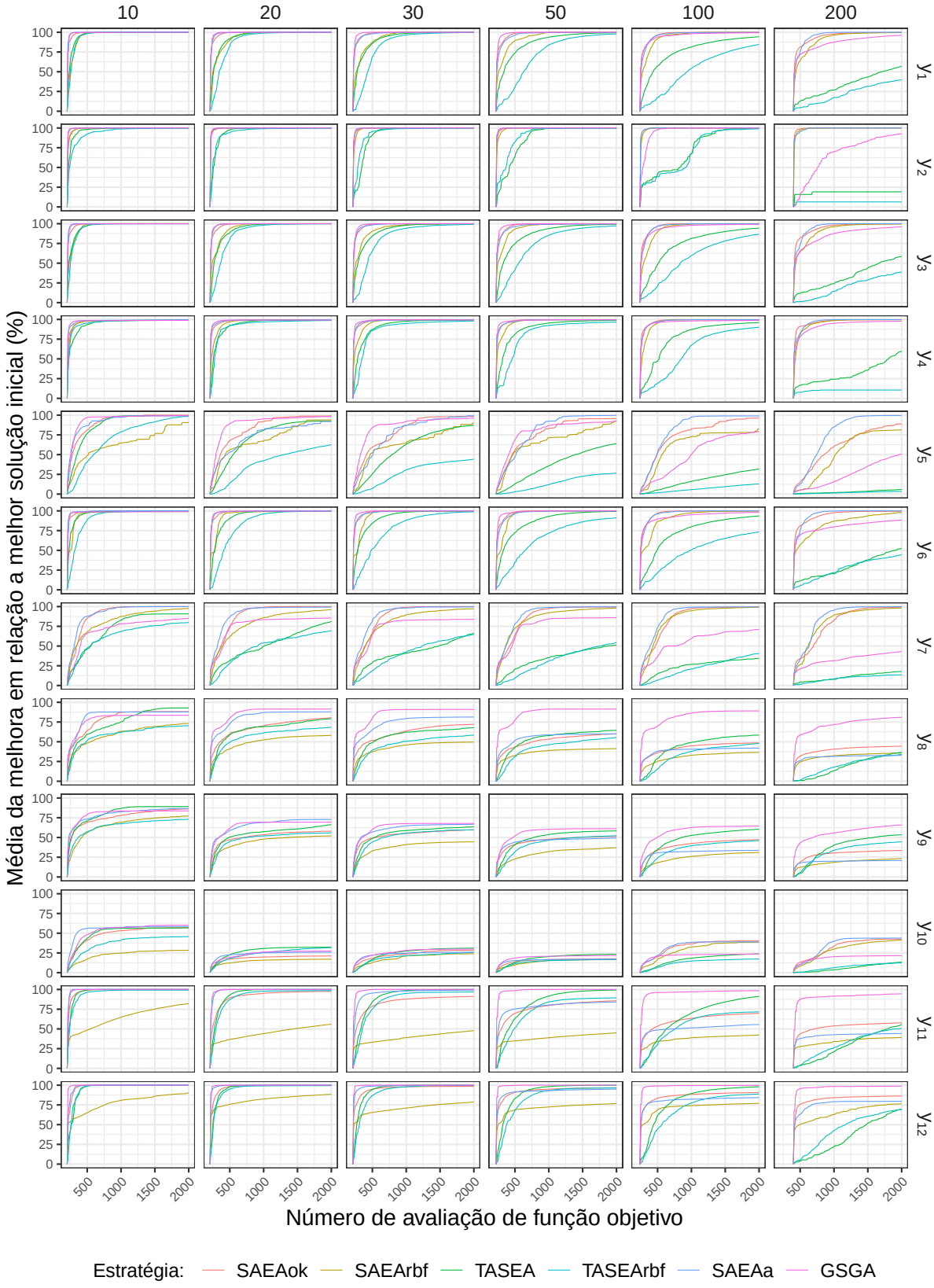


Figura 16 – Curvas de convergência da média de melhora em relação a melhor solução da população inicial em função do número de avaliações de função objetivo. Os plots são discretizados por função (vertical) e número de variáveis (horizontal).