Sheila Santisi Travessa

USO DE REDES NEURAIS ARTIFICIAIS COMO METAMODELO NA OTIMIZAÇÃO POR ALGORITMO PSO ('PARTICLE SWARM OPTIMIZATION') EM PROBLEMAS DE MAPEAMENTO ELETROMAGNÉTICO DE AMBIENTES

Tese submetida ao Programa de Pós Graduação em Engenharia Elétrica da Universidade Federal de Santa Catarina para a obtenção do Grau de Doutor em Engenharia Elétrica
Orientador: Prof. Dr. Walter Pereira Carpes Jr.

Florianópolis
2017

Ficha de identificação da obra elaborada pelo autor
através do Programa de Geração Automática da Biblioteca Universitária
da UFSC.

A ficha de identificação é elaborada pelo próprio autor
Maiores informações em:
http://portalbu.ufsc.br/ficha

Sheila Santisi Travessa

USO DE REDES NEURAIS ARTIFICIAIS COMO METAMODELO NA OTIMIZAÇÃO POR ALGORITMO PSO ('PARTICLE SWARM OPTIMIZATION') EM PROBLEMAS DE MAPEAMENTO ELETROMAGNÉTICO DE AMBIENTES

Esta Tese foi julgada adequada para obtenção do Título de "Doutor em Engenharia" e aprovada em sua forma final pelo Programa de Pós Graduação em Engenharia Elétrica

Florianópolis, 19 de junho de 2017.

_____
Prof. Marcelo Lobo Heldwein, Dr.
Coordenador do Curso

Banca Examinadora:

_____
Prof. Walter Pereira Carpes Jr, Dr.
Orientador
Universidade Federal de Santa Catarina

_____
Prof.ª Fernanda Isabel Marques Argoud, Dr.ª
Instituto Federal de Santa Catarina

_____
Prof. Jony L. Silveira, Dr.
Instituto Federal de Santa Catarina

_____
Prof. João Pedro Assumpção Bastos, Dr.
Universidade Federal de Santa Catarina

_____
Prof. Patrick Kuo Peng, Dr.
Universidade Federal de Santa Catarina

Essa tese é dedicada especialmente e totalmente aos meus Pais Osmar Roriz Travessa (em memória) e Filomena Santisi Travessa, os verdadeiros responsáveis por toda a orientação, suporte e base que me permitiram e permitem realizar tudo aquilo que me proponho, pois sem a presença deles em minha vida jamais teria chegado a lugar algum e nem realizado nada, pois a partir da base que me foi dada, ficou para mim o trabalho mais fácil seguir em frente.

AGRADECIMENTOS

Gostaria de agradecer a todos que contribuíram de maneira direta ou indireta nessa tese e na conclusão da mesma.

Agradecer também a todos os docentes que ministraram as disciplinas necessárias na formação do conhecimento para o desenvolvimento desse trabalho.

Em especial agradecer ao meu Orientador, Walter Pereira Carpes Jr., pela orientação, paciência e compreensão nos momentos que mais precisei, em função de todos os contratempos e compromissos vividos durante essa empreitada.

Agradecer também aos funcionários do Programa de Pós-graduação em Engenharia Elétrica – PGEEL: Marcelo Siqueira e Wilson da Silva Costa (Chefe de expediente), pelas orientações constantes no que se referia aos procedimentos ligados ao bom andamento do Doutorado. Além da extrema cortesia e boa vontade, em informar e ajudar, constantes.

Por fim e não menos importante, agradecer ao GRUCAD, pela oportunidade concedida em desenvolver meu trabalho em tão renomado laboratório.

“Quando abro a porta de uma nova descoberta já encontro Deus lá dentro” (Albert Einstein).

RESUMO

Este trabalho se propõe a fazer uma análise de ferramentas de otimização e custo computacional através de um estudo de caso proposto por Grubisic (2012), que trata da otimização do posicionamento de antenas em sistemas de comunicação sem fio para ambientes interiores (*indoor*) por meio de meta-heurísticas populacionais associadas à Técnica de Traçado de Raios, em que algoritmos Genéticos (GA) e Otimizadores por Enxames de Partículas (PSO) foram as duas modalidades de meta-heurísticas utilizadas como ferramentas de otimização. A proposta desta tese baseou-se na utilização da técnica de traçado de raios quase 3D (RTQ3D) para produzir o valor dos campos eletromagnéticos iniciais e calcular a função de mérito (*fitness*) para 160 receptores de acordo com os possíveis posicionamentos de duas antenas a serem distribuídas no ambiente em questão. As variáveis do problema são compostas pelos valores dos campos magnéticos para os 160 receptores em função das posições das antenas das estações radiobase, que servem como dados de entrada para o algoritmo da Rede Neural Artificial, Perceptron multicamadas, com algoritmo de aprendizado *backpropagation* Real. Os valores dos campos magnéticos associados às posições das antenas por sua vez entram como valores a serem aprendidos pela rede, ou seja, o professor da RMLP. Após o aprendizado da Rede Neural Artificial, que é o metamodelo utilizado com o objetivo de realizar eficientemente os cálculos do otimizador, entra o otimizador por enxame de partículas (PSO) para efetuar o posicionamento ótimo das antenas com uma redução significativa no custo computacional. Por fim, um dos exemplos propostos por Grubisic (2012) foi implementado como estudo de caso desta pesquisa, utilizando essa nova estrutura de análise, PSO com RMLP, como metamodelo. Essa estrutura é bem recomendada para projetos eletromagnéticos, entretanto ainda não foi aplicada para esse tipo de análise. O objetivo principal seria a diminuição do custo computacional, que no caso em questão é bem significativo. Portanto, essa tese tem um caráter inédito em relação às ferramentas usadas e ao objetivo principal (redução do custo computacional).

Palavras-chave: Otimização por enxame de partículas *(PSO)*. Rede Neural Artificial. Meta-heurísticas. Custo computacional.

ABSTRACT

This research has proposed to do an analysis of optimization tools and computational cost using a case study proposed by Grubisic (2012), which addressed optimization of the antennas positioning in wireless communication systems for indoor environments through meta-population heuristics associated with ray tracing technique, in which algorithms Genetic (GA) and Optimizers for Swarms of particles (PSO) were the two types of meta-heuristics used as optimization tools. The purpose of this thesis was based on the use of almost 3D ray tracing technique (RTQ3D) to produce the value of the initial electromagnetic fields and calculating the merit function (fitness) to 160 receivers according to the possible placements of two antennas which are distributed in the environment in a matter. The problem variables consist of the values of the magnetic fields to the 160 receivers depending on the positions of the antennas of the access points, which serve as input data for the algorithm of Artificial Neural Network, multilayer perceptron with Real backpropagation learning algorithm. The problem variables consist of the values of 160 magnetic fields to 160 receivers on the basis of the positions of the antennas of the access points, which serve as input data for the algorithm of Artificial Neural Network, multilayer perceptron with backpropagation real learning algorithm. The values of the magnetic fields associated with the positions of the antennas in turn to input values to be learned by the network, or the teacher RMLP. After learning of Artificial Neural Network, which is the metamodel used in order to enable the calculation of the optimizer, with a lower computational cost, the optimizer particle swarm enters (PSO) to make the optimum positioning of the antennas with a significant reduction the computational cost. Finally, one of the examples proposed by Grubisic (2012) is implemented as a case study of this research using this new analysis structure, PSO using RMLP as metamodel. This structure is well recommended for electromagnetic designs, but has not been applied to this type of analysis. The main objective would be to reduce the computational cost, which in this case is significant. Therefore, this thesis has a unique character in relation to the tools used and the main objective (reducing the computational cost).

Keywords: Particle Swarm Optimization (PSO). Artificial Neural Network. Meta-heuristics. Computational cost.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

PSO - Otimização por explosão de partículas (*Particle Swarm Optimization*).

RNA - Redes Neurais Artificiais.

MLP - *Perceptron* multicamadas.

RTQ3D - Traçado de Raios “Quase” 3D

EMC - Compatibilidade Eletromagnética.

LSM - Método dos Mínimos Quadrados *Least Square Method*.

RMLP - Rede *Perceptron* Multicamadas com algoritmo *Backpropagation* Real.

SUMÁRIO

1 INTRODUÇÃO

Este capítulo pretende traçar os objetivos gerais da tese a ser desenvolvida a partir desse trabalho de Doutorado, bem como fazer uma análise de métodos de otimização multiobjetivo, a partir da utilização de algoritmos evolucionários, mais especificamente o algoritmo PSO (*Particle Swarm Optimization*) assistido pela RNA, visando a redução do custo computacional.

1.1 OBJETIVOS GERAIS

Este trabalho de doutorado teve como objetivo principal desenvolver metamodelos baseados em RNAs (Redes Neurais Artificiais), com foco na redução do custo computacional, para a otimização multiobjetivo de problemas eletromagnéticos usando o algoritmo PSO (*Particle Swarm Optimization*). A rede usada como metamodelo foi a Rede Neural Artificial MLP (*Perceptron* multicamadas) com algoritmo de aprendizado *backpropagation* real.

Foram tratadas, como um dos focos principais, a importância e a relevância da utilização de metamodelos com objetivo de diminuir o custo computacional, o qual é bastante elevado na avaliação das funções objetivo durante o processo de seleção da população ótima de indivíduos.

1.1.1 Objetivos específicos

- Implementar o algoritmo evolucionário PSO;
- Aplicar o método de traçado de raios RTQ3D na produção dos dados iniciais que foram utilizados como treinamento, durante o aprendizado da RNA, MLP (*Perceptron* multicamadas) com algoritmo de aprendizado *backpropagation*;
- Fazer um estudo de alguns tipos de metamodelos, tais como: Redes Neurais Artificiais, Kriging e Planos de experiência;
- Realizar simulações da Rede Neural com o objetivo de encontrar a melhor configuração da rede no sentido de obter o melhor aprendizado com o menor custo computacional;
- Fazer simulações da PSO assistida pela RNA visando a otimização da função objetivo com foco na redução do custo computacional.

1.2 MOTIVAÇÃO

Problemas de otimização são caracterizados por situações em que se deseja maximizar ou minimizar uma função numérica de uma ou mais variáveis, num contexto em que podem existir restrições.

Tanto as funções acima mencionadas como as restrições dependem dos valores assumidos pelas variáveis de projeto ao longo do procedimento de otimização. Pode-se aplicar otimização em várias áreas, como por exemplo no projeto de sistemas ou componentes, planejamento e análise de operações, problemas de otimização de estruturas, otimização de forma, controle de sistemas dinâmicos, etc. A otimização tem como vantagens diminuir o tempo dedicado ao projeto, possibilitar o tratamento simultâneo de uma grande quantidade de variáveis e restrições de difícil visualização gráfica e/ou tabular, possibilitar a obtenção de algo melhor além de permitir a obtenção de soluções não tradicionais, com menor custo. As técnicas clássicas de otimização são conhecidas há bem mais de um século, sendo utilizadas na Física e na Geometria, servindo-se de ferramentas associadas às equações diferenciais e ao Cálculo Variacional. A sofisticação dos recursos computacionais, desenvolvidos nos últimos anos, tem motivado um grande avanço nas técnicas de otimização, aliado ao fato de que os problemas tornam-se cada vez mais complexos. Técnicas clássicas de otimização são confiáveis e possuem aplicações nos mais diferentes campos de engenharia e de outras ciências. Porém, essas técnicas podem apresentar algumas dificuldades numéricas e problemas de robustez relacionados com: a falta de continuidade das funções a serem otimizadas ou de suas restrições, funções não convexas, multimodalidade, existência de ruídos nas funções, necessidade de se trabalhar com valores discretos para as variáveis, existência de mínimos ou máximos locais, etc. Assim, os estudos de métodos heurísticos, com busca randômica controlada por critérios probabilísticos, reaparecem como uma forte tendência nos últimos anos, principalmente devido ao avanço dos recursos computacionais, pois um fator limitante destes métodos é a necessidade de um número elevado de avaliações da função objetivo (Schwefel e Taylor, 1994). Métodos clássicos possuem como grande vantagem o baixo número de avaliações da função objetivo, o que faz com que tenham convergência rápida. Contudo, esses métodos têm uma dificuldade para trabalhar com mínimos locais. Como estes métodos utilizam um único ponto do espaço de busca e informações sobre os gradientes, ao se depararem com mínimos locais, esses

métodos não conseguem avançar na busca, convergindo prematuramente sem encontrar o mínimo global.

Nos métodos de otimização natural, a função objetivo é avaliada várias vezes, sendo possível trabalhar com vários pontos ao mesmo tempo (população) em uma iteração. Isto eleva o custo computacional destes métodos. Entretanto, este fato é compensado pela menor chance que estes métodos têm de serem "presos" em mínimos locais. Há claramente uma relação de compromisso estabelecida.

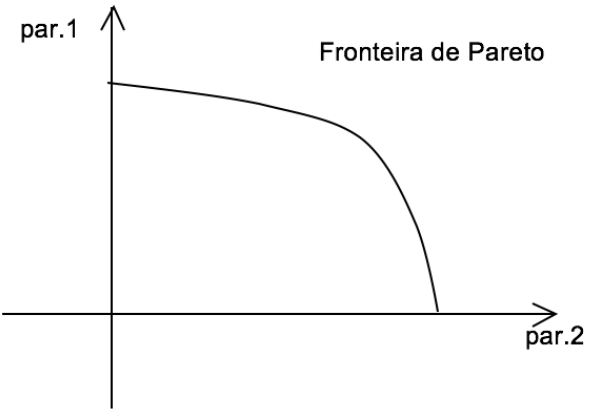
De forma geral, os métodos de otimização natural requerem maior esforço computacional quando comparados aos métodos clássicos, mas apresentam vantagens tais como: fácil implementação, robustez e dispensa dos requisitos de continuidade na definição do problema (Venter e Sobieszczanski-Sobieski, 2002). Como exemplo desta classe de métodos, podem-se citar os Algoritmos Genéticos, os quais trabalham com técnicas de computação evolutiva que modelam a evolução das espécies proposta por Darwin e operando sobre uma população de candidatos (possíveis soluções). A ideia é que a evolução da população faça com que a formação dos cromossomos dos indivíduos caminhe para o ponto ótimo, à medida que aumenta sua função de adaptação (*fitness*).

No algoritmo conhecido como Enxame de Partículas (*Particle Swarm Optimization*), um método baseado no comportamento social de aves, a busca por alimento ou pelo ninho e a interação entre os pássaros ao longo do vôo são modelados como um mecanismo de otimização. Fazendo uma analogia, a área sobrevoada é equivalente ao espaço de busca e encontrar o local com comida ou o ninho corresponde a encontrar o ponto ótimo. O algoritmo é baseado em um modelo simplificado da teoria de enxames (*swarm theory*), através da qual os pássaros ou partículas fazem uso de suas experiências e da experiência do próprio bando para encontrarem o ninho ou alimento. As aplicações presentes na literatura evidenciam a capacidade do algoritmo na solução de diferentes problemas, bem como salientam a habilidade de trabalhar com variáveis discretas e contínuas simultaneamente (Prado e Saramago, 2005).

A otimização pode ter foco num só objetivo, nesse caso conhecida como mono-objetivo, ou em mais de um objetivo ou parâmetro, em que teremos a otimização multiobjetivo. Quando trabalhamos com a otimização multiobjetivo, o resultado é dado a partir de uma fronteira sobre a qual se tem várias soluções possíveis. Essa fronteira é conhecida como "Fronteira de Pareto", conforme Figura 1. O conjunto de soluções que está fora da fronteira é desconsiderado. O

conjunto que se encontra sobre a fronteira vai ser considerado como solução, e a escolha da solução final vai ser baseada em outros aspectos que independem da seleção do melhor conjunto de soluções ou da população ótima de indivíduos.

Figura 1 - representação simbólica da "Fronteira de Pareto", em que estão exemplificados dois parâmetros de decisão (par.1 – parâmetro 1, par.2 – parâmetro 2).



Fonte: o autor.

Observa-se que grande parte dos problemas do mundo real envolve a otimização de múltiplos objetivos. Porém, a maioria dos métodos para resolução desses problemas evita as complexidades que um problema de otimização multiobjetivo envolve. Com isso, surgiram muitos métodos para converter problemas multiobjetivo em problemas com um único objetivo (DEB, 2001).

Dispondo-se de vários métodos de conversão, esqueceu-se que, na verdade, o problema de otimização com um único objetivo é uma simplificação de problemas com mais de um objetivo, e que existem diferenças fundamentais entre eles. A principal delas está na solução do problema. Por tratar de objetivos conflitantes, na otimização multiobjetivo cada objetivo corresponde a uma solução ótima. Isso faz com que esses problemas apresentem várias soluções ótimas, enquanto que algoritmos que solucionam problemas de otimização com um único objetivo normalmente geram apenas uma solução ótima. Mesmo levando em conta essa diferença fundamental entre problemas com um ou vários objetivos, sabemos que, independentemente do tipo de

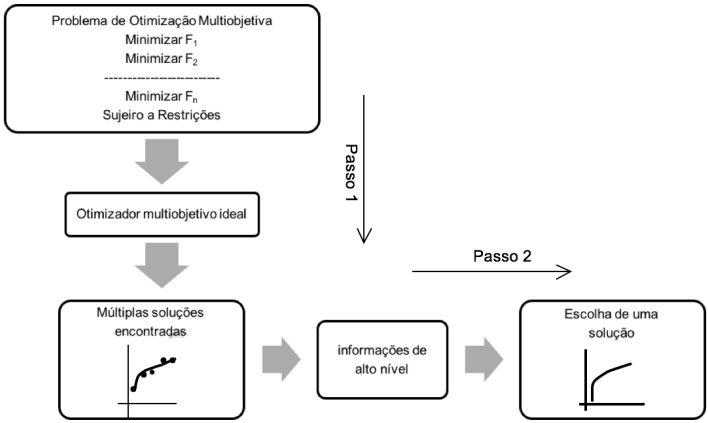
problema a ser resolvido, no mundo real, sob o ponto de vista prático, necessitamos de apenas uma solução.

Assim, para problemas de otimização multiobjetivo, cabe ao usuário, utilizando informações não técnicas e qualitativas, optar por uma das soluções apresentadas como sendo a solução ótima para o problema.

A Figura 2 mostra um esquema de um procedimento de otimização multiobjetivo ideal. No Passo 1, múltiplas soluções são encontradas, enquanto que no passo 2 uma delas é selecionada de acordo com as necessidades do usuário.

Cada solução encontrada como sendo ótima para um problema multiobjetivo corresponde a uma ordem específica de importância dos objetivos. Se uma preferência em relação aos objetivos é conhecida, não é necessário aplicar o esquema apresentado na Figura 2.

Figura 2 - Otimização multiobjetivo ideal.



Fonte: Reproduzido de Deb(2001).

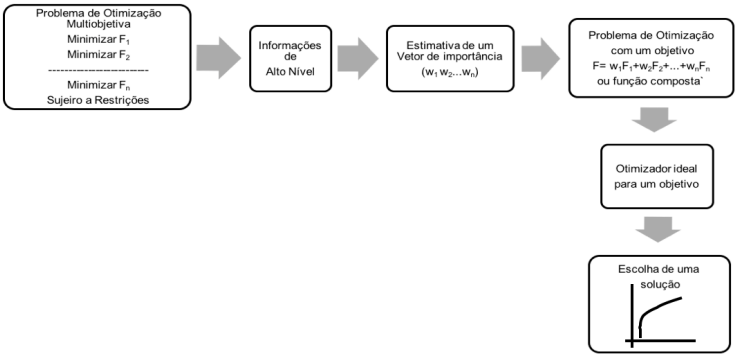
Um método simples pode ser utilizado para criar uma função composta objetiva, definida como a soma dos objetivos com seus respectivos pesos, em que o peso é proporcional ao fator de preferência de um objetivo em particular.

O método citado acima é denominado otimização multiobjetivo baseada em preferência, e é apenas um dos métodos utilizados para conversão de problemas multiobjetivo em problemas mono-objetivo e, apesar de muito simples, é mais subjetivo que o procedimento ideal

mostrado na Figura 2. A Figura 3 mostra um esquema do método baseado em preferência.

A diferença essencial nesses dois esquemas de otimização é que, no esquema ideal, a informação do problema não é utilizada para buscar por uma nova solução, e sim para escolher uma solução dentre um conjunto de soluções ótimas já escolhidas. Na otimização baseada em preferência, a informação deve ser fornecida antes da busca ser iniciada, sem nenhum conhecimento das possíveis consequências. Podemos então concluir que a abordagem multiobjetivo ideal é mais prática, mais metódica e menos subjetiva que a baseada em preferência. Porém, se um vetor de preferências para o problema já é conhecido, não há razões para não utilizá-lo (DEB, 2001).

Figura 3 - Otimização multiobjetivo baseada em preferência.



Fonte: Reproduzido de Deb, (2001).

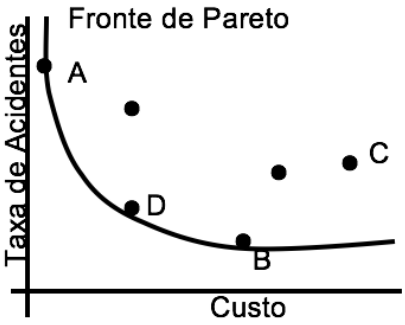
Se existem diferenças suficientes entre soluções ótimas correspondentes a diferentes objetivos, as funções objetivo são ditas conflitantes entre si. Quando isso ocorre, um conjunto de soluções ótimas surge. A essas soluções ótimas chamamos soluções ótimas de Pareto.

Consideremos um problema com dois objetivos a serem minimizados: o número de acidentes e o custo de fabricação de um certo produto. A Figura 4 mostra um conjunto de possíveis soluções para o problema. Observemos que, por exemplo, a solução A está próxima do custo mínimo, mas possui uma alta taxa de acidentes. Por outro lado, o ponto B apresenta uma solução custosa, mas que não tem tendência a acidentes. Se os dois objetivos são importantes para a solução do

problema, não podemos dizer que a solução A é melhor que a solução B, e vice-versa. Uma solução é melhor que a outra em um objetivo, mas pior em outro. A solução D, que também pertence ao conjunto de soluções ótimas, também não pode ser considerada melhor ou pior que A e B.

Observando ainda a Figura 4, concluímos que existem também algumas soluções não ótimas de Pareto, como C. Se compararmos a solução C com A, nós novamente não podemos dizer que A ou C sejam melhores nos dois objetivos. Porém, C não faz parte do fronte de Pareto porque existe uma solução D que é melhor que C nos dois objetivos. Por esse motivo, a solução C é conhecida como uma solução dominada. Nesse exemplo, a solução C também é dominada pela solução B.

Figura 4 – Conceito de soluções ótimas de Pareto.



Fonte: Reproduzido de Deb (1999).

Dessa forma, pode-se concluir que para um problema com mais de uma função objetivo, uma solução x_1 domina uma solução x_2 se duas condições são satisfeitas:

1. A solução x_1 não é pior que x_2 em nenhum dos objetivos.
2. A solução x_1 é estritamente melhor que a solução x_2 em pelo menos um objetivo.

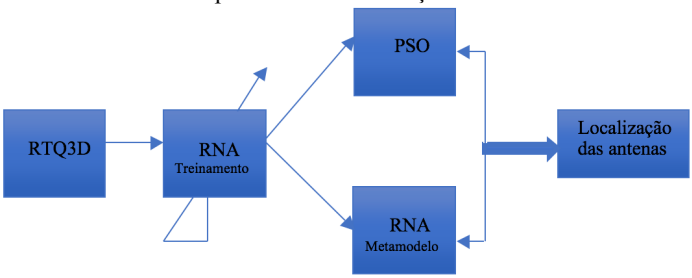
Assim, as soluções que não são dominadas por nenhuma outra solução são consideradas soluções ótimas de Pareto.

1.3 FLUXO APLICADO AO PROCESSO DE OTIMIZAÇÃO

De acordo com a figura 5, em primeiro lugar, os valores dos campos foram gerados pelo traçado de raios e usados como treinamento no processo de aprendizagem realizado pela RNA. Os valores dos pesos da rede, depois de ajustados, foram salvos em arquivo.

Após o treinamento, a RNA atua como metamodelo no processo de otimização realizado pelo algoritmo evolucionário PSO, na procura do melhor posicionamento das antenas. O critério para o posicionamento foi minimizar o número de pontos em que o campo está abaixo de um limiar mínimo de recepção pré-estabelecido (-110 dBm). O uso da RNA permitiu uma grande redução no tempo computacional.

Figura 5 – Fluxo de trabalho do processo de otimização.



Fonte: o autor.

1.4 ESTRUTURA DO TRABALHO

A estrutura desse trabalho está dividida em 6 capítulos, quais sejam:

- Capítulo 1, em que são abordados os aspectos básicos sobre o tema escolhido, assim como, objetivos, estado da arte e a estrutura do trabalho.
- Capítulo 2, em que é feita uma análise sobre a otimização multiobjetivo a partir do algoritmo evolucionário PSO, tratando-se primeiramente do problema geral da otimização, em seguida abordando todos os aspectos relativos e necessários à implementação e à análise do algoritmo evolucionário em questão.
- Capítulo 3, em que são feitas considerações do porquê da utilização dos metamodelos para assistir os algoritmos

evolucionários. Em seguida, os principais metamodelos são apresentados com suas formulações, visando fazer parte da construção da tese.

- Capítulo 4, em que é dado ênfase às RNA *feedforward Perceptron* multicamadas com algoritmo de aprendizado *backpropagation* real, que foi o metamodelo utilizado neste trabalho. A razão de utilizar a RNA em vez dos demais métodos também foi abordada.
- Capítulo 5, em que são apresentados os resultados obtidos no trabalho desenvolvido num estudo de caso real utilizando as ferramentas propostas neste trabalho.
- Capítulo 6, em que são abordados os aspectos relevantes da construção desse trabalho, dos resultados obtidos tendo em vista os objetivos traçados e a delimitação do problema. São também feitas sugestões para trabalhos futuros, com o intuito de ampliar a análise proposta ou usá-la como ponto de partida para linhas diferentes de estudo.

1.5 CONSIDERAÇÕES FINAIS DO CAPÍTULO

Observou-se que os métodos tradicionais de busca, que trabalham com apenas uma solução, são aplicados para resolver problemas com um objetivo. Já a otimização multiobjetivo exige que o método de busca designado para resolver o problema encontre um conjunto de soluções ideais, e não uma solução global ideal. Dos métodos de busca não convencionais conhecidos hoje, os algoritmos evolucionários destacam-se por utilizar uma população de soluções a cada iteração, e retornar um conjunto de soluções.

De acordo com Deb (2001), os algoritmos evolucionários apresentam uma habilidade ímpar para encontrar múltiplas soluções ótimas em uma simulação, o que os torna únicos na resolução de problemas de otimização multiobjetivo.

No capítulo 2 será feita uma análise detalhada do método de otimização evolucionário escolhido para esse trabalho, o algoritmo PSO.

2 OTIMIZAÇÃO POR ENXAME DE PARTÍCULAS (PSO – *PARTICLE SWARM OPTIMIZATION*)

Neste capítulo, pretende-se analisar detalhadamente a otimização multiobjetivo a partir do algoritmo evolucionário PSO, tratando-se primeiramente do problema geral da otimização, em seguida abordando todos os aspectos relativos e necessários à implementação e à análise do algoritmo evolucionário em questão.

2.1 PROBLEMA GERAL DE OTIMIZAÇÃO

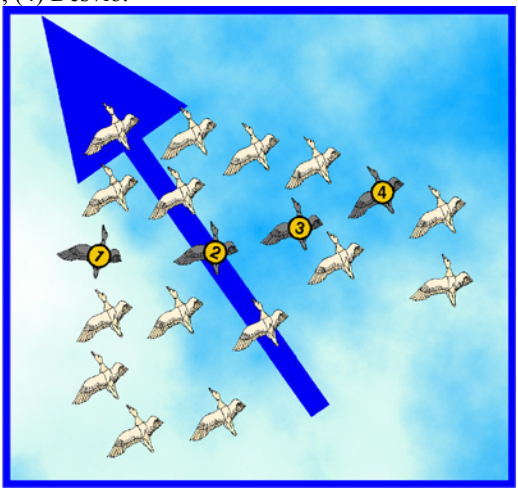
O problema geral de otimização consiste em minimizar uma função objetivo, sujeita, ou não, a restrições de igualdade, desigualdade e restrições laterais.

A função objetivo e as funções de restrições podem ser funções lineares ou não lineares em relação às variáveis de projeto, implícitas ou explícitas, calculadas por técnicas analíticas ou numéricas.

Baseado em observações, C. W. Reynolds percebeu que a movimentação de bandos de pássaros e cardumes de peixes eram sincronizados sem existir um controle central. Ele então criou um modelo do movimento de bandos de pássaros composto apenas por quatro regras (Figura 6):

- Separação: para evitar que cada pássaro colidisse com um outro;
- Alinhamento: para fazer com que cada pássaro seguisse a mesma direção de seus vizinhos;
- Coesão: para que cada pássaro seguisse a mesma posição de seus vizinhos;
- Desvio: para que cada pássaro desviasse de obstáculos à frente.

Figura 6 - Representação das regras: (1) Separação; (2) Alinhamento; (3) Coesão; (4) Desvio.



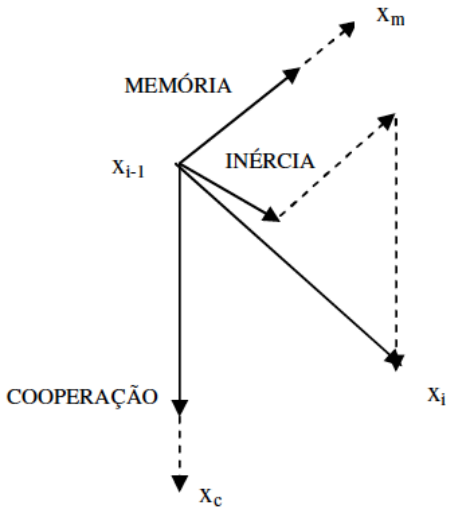
Introduzido por Kenedy e Eberhart em 1995, o método PSO (*Particle Swarm Optimization*) surgiu de experiências com algoritmos modelados a partir da observação do comportamento social de determinados tipos de pássaros. As partículas consideradas pelo algoritmo se comportam como os pássaros à procura de alimento ou do local de seus ninhos, utilizando o aprendizado próprio e o aprendizado do bando (ou enxame). O PSO é composto de partículas representadas por vetores que definem a velocidade atual de cada partícula, e de vetores de localização, atualizados segundo sua velocidade atual, seu aprendizado pessoal e o aprendizado adquirido pelo bando. O algoritmo de PSO engloba conceituações simples e pode ser implementado em poucas linhas de programação, requerendo apenas operadores matemáticos simples.

Apesar de classificado como algoritmo evolucionário, o PSO não apresenta a característica de sobrevivência do mais apto ou a utilização de operadores genéticos, como o cruzamento e a mutação. Em 1995, James Kennedy and Russell Eberhart seguiram o modelo de Reynolds para aplicar em problemas de otimização contínua. Para isso, eles utilizaram as regras de alinhamento e coesão e criaram a técnica de otimização de particulas denominada Particle Swarm Optimization (PSO).

No modelo matemático adotado para o PSO, um “enxame de partículas” é gerado aleatoriamente em um determinado espaço de

busca. Dentro desse espaço, cada partícula corresponde a uma possível solução, representada por sua posição no espaço de busca para um dado problema. As partículas têm associado um valor de velocidade, e também realizam um deslocamento sob a ação de três vetores que se somam. Essas influências são inércia, memória e comparação. O primeiro vetor impele a partícula em uma direção idêntica à que ela vinha seguindo. A memória atrai a partícula na direção da melhor posição até o momento, ocupado pela partícula dentro da sua vida. O último vetor atrai a partícula na direção do melhor ponto do espaço até o momento, descoberto pelo "enxame" (EBERHART e SHI, 2001). A figura 7 demonstra um exemplo de deslocamento de uma partícula, que se encontrava na posição x_{i-1}, para a posição x_i fazendo uso dos vetores mencionados.

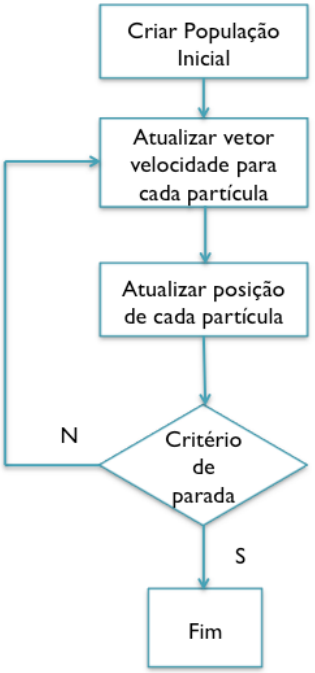
Figura 7 – Movimento de uma partícula.



Fonte: Borges (2006).

Na implementação do algoritmo, o primeiro passo é gerar as *N* partículas que formarão o "enxame", com suas respectivas posições. Pode-se também neste momento, arbitrar velocidades iniciais para cada partícula. O algoritmo se manterá ativo, atualizando os vetores de velocidade e posição ciclicamente até que seja atingido algum critério de parada (número máximo de iterações atingido, partícula com aptidão desejada, etc.), conforme mostra o fluxograma da Figura 8.

Figura 8 – Fluxograma do PSO (*Particle Swarm Optimization*).



Fonte: Menezes (2007).

O vetor velocidade de cada partícula deve ser atualizado conforme equação 1:

$$v_{k+1}^{i} = w \cdot v_{k}^{i} + c1 \cdot r1 \cdot \left(p^{i} - x_{k}^{i}\right) + c2 \cdot r2 \cdot \left(p_{k}^{s} - x_{k}^{i}\right) \quad (1)$$

onde v_{k}^{i} é a velocidade atual da partícula; p^{i} é a melhor posição encontrada pela partícula *i*; p_{k}^{s} é a melhor posição dentre todas as partículas na iteração *k*; e *w* é um parâmetro que representa a inércia da partícula e controla sua capacidade de exploração do espaço de soluções. Um valor alto de *w* determina uma busca global enquanto um valor baixo determina uma busca local. Usualmente estes valores oscilam entre 0,4 e 1,4. Já *c1 e c2* são os chamados parâmetros de confiança e definem o quanto uma partícula confia em si (*c1*) ou no

bando (*c2*). Usualmente escolhe-se *c1* = *c2* = 2 (MENEZES, 2007). Por fim, *r1 e r2* são números aleatórios compreendidos entre zero e um.

Para o cálculo da posição futura de cada partícula no algoritmo, é utilizada a equação 2:

$$x_{k+1}^{i} = x_{k}^{i} + v_{k+1}^{i} \tag{2}$$

Onde:

x_{k+1}^{i} é a posição de cada partícula *i* na iteração *k*+1;

v_{k+1}^{i} é o vetor velocidade da partícula (considera-se que já esteja multiplicado por um intervalo de tempo unitário).

2.1.1 O enxame inicial

A inicialização da população do enxame normalmente é obtida com as partículas dispostas aleatoriamente sobre o espaço de busca, e cada uma possui um vetor de velocidade aleatório inicial. As equações 3 e 4 mostram como são obtidos a posição e o vetor de velocidades iniciais (PRADO e SARAMAGO, 2005).

$$x_{0}^{i} = x_{\min} + r_1\left(x_{\max} - x_{\min}\right) \tag{3}$$

$$v_{0}^{i} = \frac{x_{\min} + r_2\left(x_{\max} - x_{\min}\right)}{\Delta t} \tag{4}$$

Onde:

r_1 e r_2 são números aleatórios entre 0 e 1;

$x_{\min}$ é o valor mínimo permitido para as variáveis de projeto;

$x_{\max}$ é o valor máximo permitido para as variáveis de projeto.

2.2 OTIMIZAÇÃO SEM RESTRIÇÕES

Os algarismos evolutivos e PSO, por se tratarem de algoritmos naturais, não trabalham diretamente com restrições. Uma estratégia para fazer com que estes algoritmos manipulem restrições é a utilização de funções de penalidade quadrática estendida (PRADO e SARAMAGO, 2005).

Assim, define-se uma função pseudo-objetivo, equação 5, $\psi(x)$:

$$\psi(x) = f(x) + rp\sum_{i=1}^{m} \max\left[0, g_i(x)\right]^2 \tag{5}$$

Sendo,

f(x) a função objetivo original;

rp um fator de penalidade (de ordem variável segundo o tipo de problema);

gi(x) o conjunto de todas as restrições (com violações para *gi(x)>0*).

Quando há restrições nos problemas de otimização, as partículas que desrespeitam alguma restrição enquadram-se em um grupo que merece um tratamento especial. Esse tratamento se inicia pelo cálculo do novo vetor de velocidade, dado pela equação 6:

$$v_i^{k+1} = c_1 r_1 \cdot \frac{\left(p^i - x_k^i\right)}{\Delta t} + c_2 r_2 \cdot \frac{\left(p_k^s - x_k^i\right)}{\Delta t} \tag{6}$$

Observe que a equação 6 não leva em consideração a informação do vetor de velocidade na iteração anterior para o novo cálculo do vetor de velocidade. Isso se deve ao fato da partícula estar “divergindo” em direção a uma violação. Na maioria das vezes, este novo vetor de velocidades se destinará a uma região viável, e a partícula sai da restrição.

2.3 VARIÁVEIS DE PROJETOS DISCRETAS/INTEIRAS

O PSO é um algoritmo contínuo, contrapondo-se aos Algoritmos Genéticos que, em sua concepção original, eram destinados a variáveis discretas. Porém, o PSO pode ser muito eficiente na resolução de problemas com variáveis discretas, desde que sejam feitas algumas modificações no algoritmo. Por exemplo, a posição de cada partícula é arredondada para o valor inteiro mais próximo logo após a aplicação da equação 2 ou da equação 3 (PRADO e SARAMAGO, 2005).

2.4 CONSIDERAÇÕES FINAIS DO CAPÍTULO

Através da análise do problema geral da otimização, verificou-se que otimizar consiste em minimizar uma função objetivo, sujeita ou não a restrições de igualdade, desigualdade e restrições laterais (faixa de valores permitidos).

Na sequência, todos os passos necessários à implementação do algoritmo foram analisados: desde o primeiro, que é criar a população inicial, até o fim do processo, passando pela atualização dos vetores velocidade e posição e pelo critério de parada, detalhados no fluxograma da Figura 8. Todas as equações necessárias à implementação do algoritmo também foram apresentadas, tornando esse capítulo um tutorial para a construção do algoritmo PSO, que será utilizado nas implementações desse trabalho.

No capítulo 3, serão apresentados os principais metamodelos que podem ser utilizados para assistir o algoritmo PSO, dando ênfase ao metamodelo construído a partir da RNA, que foi a opção para esse trabalho.

3 METAMODELOS

Nesse capítulo será feita uma discussão do porquê da utilização dos metamodelos para assistir os algoritmos evolucionários. Em seguida, os principais metamodelos serão apresentados, com suas devidas formulações.

3.1 JUSTIFICATIVAS

Vários modelos podem ser utilizados para obtenção de uma aproximação com boa precisão. Os mais populares são: o modelo polinomial, o modelo Kriging, as redes neurais *feedforward*, incluindo as *perceptrons* multicamadas, as redes de função de base radial e as máquinas de suporte vetorial.

Os algoritmos evolucionários que utilizam explosão de partículas e os algoritmos genéticos são frequentemente usados para resolver vários problemas de optimização. Entretanto, a convergência desses algoritmos geralmente exige um grande número de avaliações da função objetivo (função de mérito ou de *fitness*), o que implica a utilização de requisitos computacionais mais elevados para obter soluções de qualidade.

Em Mohammed, Park, Üler e Zigiang, (1992), um novo método de otimizar projetos de dispositivos eletromagnéticos é apresentado. O método utiliza redes neurais artificiais no ambiente do projeto, o qual engloba computação numérica e dados de entrega dinâmicos para gerar uma variabilidade de dados de treinamento da rede neural artificial.

O projeto otimizado de limites geométricos em dispositivos eletromagnéticos está recebendo especial atenção e interesse por parte de muitos pesquisadores. Nos últimos anos, redes neurais artificiais vêm sendo aplicadas com sucesso em problemas de engenharia, operações associativas, operações de busca, classificação e regressão. Também, as redes neurais artificiais podem ser utilizadas para providenciar uma alternativa de computação paralela e alta velocidade para a variabilidade dos problemas em eletromagnetismo, (MOHAMMED; PARK; ÜLER; ZIGIANG, 1992). Portanto, encontram aplicações para produzir boas soluções para problemas complexos de otimização.

Uma das grandes aplicações dos metamodelos é na otimização de problemas eletromagnéticos. De fato, como a modelagem de tais problemas pode ter um custo computacional elevado, a otimização de

dispositivos eletromagnéticos frequentemente demanda alguns procedimentos de substituição das funções objetivo. A função substituta tem um custo computacional menor, mas possibilitando obter resultados com boa precisão. Um desses métodos é chamado Kriging, que é uma ferramenta inicialmente desenvolvida para geoestatística (LEBENSZTJAN; MARRETTO; COSTA; COULOMB, 2004).

Segundo Emmerich e Varcol (2005), as estratégias de evolução por metamodelos assistidos são propostas como uma nova aproximação para otimização por simulação de base, no domínio do projeto de compatibilidade eletromagnética (EMC). No domínio desta aplicação, funções de estimativas tendem a consumir muito tempo, e técnicas como a metamodelagem são aplicadas para acelerar as estratégias de otimização evolucionária.

Segundo Jin (2005), devido à falta de dados e de um espaço de entrada de alta dimensão, é muito difícil obter uma aproximação funcional perfeita da função objetivo original. Para combater este problema, duas medidas primordiais devem ser tomadas. Em primeiro lugar, o modelo de aproximação precisa ser utilizado juntamente com a função objetivo original. Em muitos casos, a função objetivo original está disponível, entretanto com um custo computacional elevado para sua avaliação. Em segundo lugar, a qualidade do modelo de aproximação precisa ser provada tanto quanto possível para um número limitado de dados. Vários aspectos são importantes para demonstrar a qualidade do modelo, tais como: a seleção do modelo, o uso do dado ativo amostrando e atribuindo um peso a eles, a seleção do método de treinamento e a seleção da medida de erro a ser utilizada.

Os algoritmos evolutivos têm se mostrado ferramentas de otimização global poderosas. Normalmente, os algoritmos evolutivos têm um desempenho superior ao dos algoritmos convencionais de otimização para problemas descontínuos, multimodais, ruidosos e que não são muito bem definidos, tais como: projetos de arte, composições musicais e projetos experimentais. Independentemente do grande sucesso encontrado em aplicações do mundo real, os algoritmos evolutivos têm encontrado alguns desafios. Para a maior parte dos algoritmos evolutivos, um grande número de cálculos precisos é necessário antes de uma solução aceitável ser encontrada. Há várias situações em que os cálculos de precisão se tornam difíceis e, com isso, aproximações computacionais eficientes da função de aptidão (*fitness*) precisam ser adotadas (JIN, 2005).

Sun *et alii* (2013) atestam que a otimização por algoritmos evolucionários, nesse caso particularmente o PSO, que é uma meta-

heurística global, provadamente muito poderosa para otimizar uma ampla gama de problemas, exige um grande número de avaliações da função *fitness* para encontrar soluções aceitáveis. Foi verificado que se uma única avaliação da função objetivo através do algoritmo PSO tem um custo computacional elevado, o custo computacional para toda a execução da otimização se tornaria proibitivo. A solução proposta foi uma estratégia de estimação da função *fitness* com mínima perda de precisão e ganho significativo no que se refere ao custo computacional.

Em Regis (2014), foi desenvolvida a estrutura OPUS (Otimização por enxame de partículas usando substitutos) *framework* para o custo elevado na otimização de uma caixa preta. Os resultados numéricos sugeriram que algoritmos assistidos apresentam resultados mais satisfatórios no que diz respeito ao custo computacional.

Em Roy *et alii* (2014), um metamodelo baseado em RNA foi desenvolvido para correlacionar a função objetivo com as variáveis de controle escolhidas. O PSO assistido pelo metamodelo formou o algoritmo de otimização devido à sua simplicidade e eficiência inerentes ao obter uma convergência garantida da função objetivo, com um custo computacional significativamente menor em função da assistência dada pela estrutura de análise e otimização montada nesse estudo.

Em Zhao e Guo (2014), foi analisada a necessidade de uma previsão do consumo de energia para fornecer orientação confiável para planejadores e formuladores de políticas de energia que também possam reconhecer as tendências do desenvolvimento econômico e industrial de um país. Nesse artigo, foi proposto um modelo híbrido PSOCA, (combinação de algoritmo de otimização de enxame de partículas e algoritmo cultural), GRNN (rede neural de regressão generalizada) para a previsão anual de consumo de energia. Levando-se em consideração o consumo anual de energia da China como o exemplo empírico, a eficácia deste do modelo proposto PSOCA-GRNN foi satisfatório.

Segundo Parka e Kim (2017), os requisitos computacionais necessários ao processo de otimização tornam-se mais desafiadores quando problemas de otimização são associados a análises computacionais ou a tarefas de simulação. Para abordar essa questão, a metamodelagem mostrou resultados bem sucedidos na melhoria da eficiência da aproximação das funções de aptidão ou restrição de problemas de otimização complexos.

Parka e Kim (2017) utilizaram uma rede neural de regressão generalizada no trabalho desenvolvido para construir o metamodelo e para aproximar a função de aptidão na otimização por enxame de partículas. Para avaliar o desempenho e a qualidade dessas soluções, a

abordagem de metamodelagem proposta foi testada em dez funções de *benchmark*. Os resultados foram promissores em termos de qualidade da solução e eficiência computacional, especialmente quando comparados com os resultados da otimização por enxames de partículas sem metamodelagem e vários outros métodos de metamodelagem publicados anteriormente na literatura.

3.2 MODELOS POLINOMIAIS

O modelo de aproximação polinomial mais utilizado é o de segunda ordem, que apresenta a seguinte forma:

$$\hat{y} = \beta_o + \sum_{1 \le i \le n} \beta_i x_i + \sum_{1 \le i \le j \le n} \beta_{n-1+i+j} x_i x_j \tag{7}$$

onde β_0 e β_i são os coeficientes a serem estimados. O número de termos no modelo quadrático é $n_t = (n+1)(n+2)/2$ no total, onde n é o número de variáveis de entrada.

Para estimar os coeficientes desconhecidos do modelo polinomial, os métodos do mínimos quadrados (LSM – *Least Square Method*) e o método do gradiente podem ser utilizados (JIN, 2005).

3.2.1 Método dos mínimos quadrados

Para obter uma única estimação dos coeficientes utilizando o LSM, é necessário que o número de amostras (N) produzidas a partir da função original seja igual ou maior que o número de coeficientes n_t. O processo de estimação é descrito pelas equações 8, 9, 10 e 11.

$$\mathrm{y} = \left[y^{(1)}, y^{(2)}, \ldots, y^{(N)} \right]^T \tag{8}$$

e

$$X = \begin{bmatrix} 1 & x_1^{(1)} & x_2^{(1)} \cdots & \left(x_n^{(1)}\right)^2 \\ 1 & x_1^{(2)} & x_2^{(2)} \cdots & \left(x_n^{(2)}\right)^2 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_1^{(N)} & x_2^{(N)} \cdots & \left(x_n^{(N)}\right)^2 \end{bmatrix} \tag{9}$$

A equação que segue correlaciona y e x:

$$y = X \cdot \Theta \tag{10}$$

O algoritmo LSM funciona da seguinte maneira:

$$\hat{\Theta} = (X^T \cdot X)^{-1} X^T \cdot y \tag{11}$$

onde $\hat{\Theta}$ representa uma estimativa de Θ. Assume-se que as linhas de X são linearmente independentes.

3.2.2 Método do gradiente

O principal inconveniente do método dos mínimos quadrados é que o custo computacional se torna inaceitável com o aumento da dimensão. Para evitar este problema, o método do gradiente pode ser usado. Define-se a função do erro quadrado para a *k-ésima* amostra, equação 12:

$$E^{(k)} = \frac{1}{2}\left(y - y^{(k)}\right)^2 \tag{12}$$

onde y, definido na equação (7), leva à troca da regra a ser utilizada no cálculo dos coeficientes desconhecidos, conforme as equações 13 à 15:

$$\Delta\beta_0 = -\xi \cdot \left(y - y^{(k)}\right) \tag{13}$$

$$\Delta\beta_i = -\xi \cdot \left(y - y^{(k)}\right) \cdot x_i^{(k)} \tag{14}$$

$$\Delta\beta_{n-1+i+j} = -\xi \cdot \left(y - y^{(k)}\right) \cdot x_i^{(k)} x_j^{(k)}, \qquad (15)$$

$$1 \le i \le j \le n.$$

$$\xi \ge 0$$

3.3 MODELOS KRIGING

Segundo Lebensztjan *et ali* (2004), o Kriging explora a correlação espacial dos dados em vez de construir interpolações. Assim, a função de correlação escolhida afeta bastante a qualidade da aproximação. Kriging é um termo geral usado para uma família de métodos para estimar o mínimo erro de variância.

O modelo Kriging pode ser visto como a combinação de um modelo global acrescido de um "desvio" localizado, equação 16:

$$y(x) = g(x) + z(x) \qquad (16)$$

onde *g(x)* é uma função conhecida de *x* como um modelo global da função original, e *z(x)* é uma função gaussiana randômica, com média zero e covariância diferente de zero, que representa o desvio localizado de um modelo global. Normalmente *g(x)* é um polinômio e em alguns casos é reduzido a uma constante β.

A covariância de *z(x)* é expressa na equação 17:

$$Cov[z(x^{(j)}), z(x^{(k)})] = \sigma^2 \mathrm{R}[R(x^{(j)}), R(x^{(k)})], \qquad (17)$$

$$j,k = 1, \ldots, \mathrm{N}$$

onde *R* é a função de correlação entre duas amostras quaisquer das N consideradas e R é a matriz de correlação simétrica de dimensão N × N com a diagonal unitária. A forma da matriz de correlação pode ser escolhida pelo usuário. A forma a seguir tem sido bastante utilizada, como descrito na equação (18), (BROOKER et ali, 1998; GIUNTA e WATSON, 1998; SIMPSON et ali, 1998 *apud* JIN, 2005):

$$R(\mathrm{x}^{(j)}, \mathrm{x}^{(k)}) = \exp\left[-\sum_{i=1}^{n} \theta_i \left| x_i^{(j)} - x_i^{(k)} \right|^2 \right] \qquad (18)$$

onde θ_i são os parâmetros de correlação desconhecidos e $x_i^{(j)}$ e $x_i^{(k)}$ são os *i*-ésimos componentes dos pontos amostrados $\mathrm{x}^{(j)}$ e $\mathrm{x}^{(k)}$.

Então, a estimativa de $y(\mathrm{x})$, equação 19, é uma função de parâmetros desconhecidos β e θ_i, $i = 1, 2, \ldots, \mathrm{n}$:

$$\hat{y} = \hat{\beta} + \mathrm{r}^T(\mathrm{x})\mathrm{R}^{-1}(\mathrm{y} - \beta\,\mathrm{I}), \tag{19}$$

onde $\hat{y}$ é o valor estimado de *y* dadas as N amostras e a entrada considerada x , $\hat{\beta}$ é o valor estimado de β, *y* é o vetor de tamanho N como definido na equação (3.2), I é um vetor unitário de tamanho N, e r é o vetor de correlação de tamanho N, entre a entrada x e as amostras $\left\{\mathrm{x}^{(1)}, \ldots, \mathrm{x}^{(N)}\right\}$, equação (20):

$$\mathrm{r}^T(\mathrm{x}) = \left[R(\mathrm{x}, \mathrm{x}^{(1)}), R(\mathrm{x}, \mathrm{x}^{(2)}), \ldots, R(\mathrm{x}, \mathrm{x}^{(N)})\right]^T \tag{20}$$

A estimação dos parâmetros pode ser produzida usando o método da máxima verossimilhança.

Uma vantagem de se utilizar os modelos Kriging é que o intervalo de confiança da estimação dos valores pode ser obtido sem um aumento considerável no custo computacional. Porém, vale observar que é necessário realizar inversões de matrizes para estimar a saída no modelo Kriging, o que incrementa o custo computacional significativamente quando em altas dimensões (JIN, 2005).

3.4 CONSIDERAÇÕES FINAIS SOBRE O CAPÍTULO

Este capítulo possibilitou a análise de alguns dos principais metamodelos que podem assistir algoritmos evolucionários, mais especificamente o PSO, que é ferramenta deste trabalho.

No capítulo 4 será descrito com mais detalhes o metamodelo RNA *feedforward Perceptron* multicamadas com algoritmo de aprendizado *backpropagation* real, que será o metamodelo utilizado por este projeto e o porque da utilização desse metamodelo especificamente. Nesse capítulo também será descrito um estudo de caso, com o objetivo de comparar a performance de dois dos principais metamodelos estudados, o modelo Kriging e a RNA. Por fim a escolha da RNA como metamodelo será justificada.

4 REDES NEURAIS ARTIFICIAIS E RTQ3D

Neste capítulo será dado ênfase às RNA's *feedforward Perceptron* multicamadas com algoritmo de aprendizado *backpropagation* real, que será o metamodelo utilizado por esse projeto. Também será feita uma pequena introdução sobre o método RTQ3D, que produz o conjunto de treinamento da RNA em questão. Considerações serão feitas, sobre o porquê de utilizar a RNA em vez dos demais métodos abordados e um estudo de caso será apresentado, com o objetivo de comparar o desempenho de dois dos principais metamodelos estudados, o modelo Kriging e a RNA e a justificativa da escolha da RNA.

4.1 INTRODUÇÃO

Segundo Pereira (2003), a busca de métodos para automatizar tarefas repetitivas e extensas, ou cujo contexto aborde muitos detalhes a serem considerados em investigações, é um interesse comum em quase todas as áreas de estudos e pesquisas. Uma Rede Neural Artificial (RNA) é a base para muitos destes métodos de automatização. Contudo, a opção de se utilizar uma rede para desenvolver determinada tarefa leva à necessidade de se definir uma rede, dentre as várias redes existentes, que se adéque à aplicação desejada.

Redes neurais artificiais são motivadas pelo estudo do sistema nervoso biológico. Computadores modernos e algoritmos computacionais são bons para tarefas bem definidas, sequenciais, mas têm dificuldade em realizar tarefas como reconhecimento de padrões, por exemplo. As RNA's estão sendo estudadas por mais de 30 anos e seus modelos estão providenciando novas aproximações na resolução de problemas. Também podem ser simuladas com propósitos especiais: aceleradores de hardware neural, bem como máquinas convencionais (TAGLIARINI, 1991).

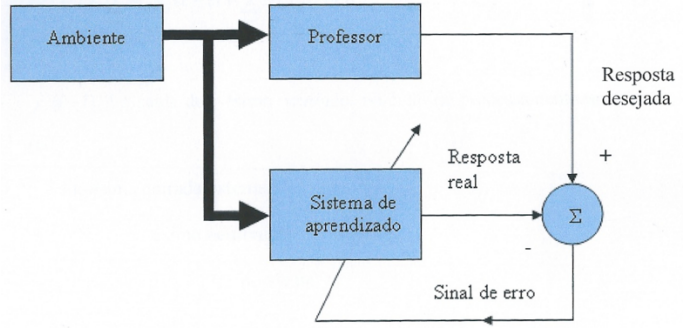
Segundo Haykin (1999), uma rede neural é um processador maciçamente paralelo distribuído constituído de unidades de processamento simples, que têm a propensão natural para armazenar conhecimento experimental e torná-lo disponível para o uso. Ela se assemelha ao cérebro em dois aspectos:

- O conhecimento é adquirido pela rede a partir de seu ambiente através de um processo de aprendizagem.

- Forças de conexão entre neurônios, conhecidos como pesos sinápticos, são utilizadas para armazenar o conhecimento adquirido.

O procedimento utilizado para realizar o processo de aprendizagem é chamado de algoritmo de aprendizagem, cuja função é modificar os pesos sinápticos da rede de uma forma ordenada para alcançar um objetivo de projeto desejado (HAYKIN, 1999). O tipo de aprendizado abordado neste trabalho foi o aprendizado supervisionado, cujo princípio pode ser ilustrado pelo diagrama em blocos da Figura 9.

Figura 9 - Diagrama em blocos que ilustra o aprendizado supervisionado.



Fonte: Travessa (2006).

Em termos conceituais pode-se considerar o professor, isto é o supervisor, como aquele que tem o conhecimento sobre o ambiente, com este conhecimento sendo representado por um conjunto de exemplos de entrada e saída. Quando o supervisor e a rede neural são expostos a um vetor de treinamento, retirado do ambiente, em virtude de seu conhecimento anterior, o supervisor é capaz de fornecer à rede neural uma resposta desejada para aquele vetor de treinamento. Cada parâmetro da rede é ajustado através da influência combinada do vetor de treinamento e do sinal de erro. O sinal de erro corresponde à diferença entre a resposta desejada e a resposta real da rede. Este ajuste através do erro é realizado passo a passo, iterativamente, com o objetivo de fazer a rede emular o professor. Pode-se considerar que a emulação seja ótima em um sentido estatístico. Com isso, o conhecimento do ambiente disponível ao supervisor é transferido para rede através de treinamento, da forma mais completa possível (HAYKIN, 1999).

No modelo geral de neurônios, as entradas w_{nIn} são combinadas usando uma função fc(*net*) = $\phi(.)$, para produzir um estado de ativação dos neurônios que, através da função de saída *S*, vai produzir a saída do neurônio Y, (RODRIGUES, 1999).

Cada entrada o neurônio possui um conjunto de pesos associados, que pode ser entendido como a 'força' da conexão sináptica quando comparado ao neurônio biológico. As entradas de um neurônio podem ser as saídas de outros neurônios, *y(t)*, entradas externas *I*, um *bias* θ ou qualquer combinação destes elementos. O somatório de todas essas entradas possibilita a origem do chamado *net* de um neurônio, que pode ser representado pelo somatório das entradas proveniente de outros neurônios e pelo somatório das entradas externas, como mostra a equação (21), (PEREIRA, 2003) (TRAVESSA, 2006) (SOVIERSOSKI, 2009):

$$net_i(t) = \sum_{j=1}^{p} w_{ji} y_j(t-1) + \sum_{l=1}^{m} w_{ki} I_k(t) + \theta_i \tag{21}$$

onde:

- $y_j(t\text{-}1)$ é a saída do j-ésimo neurônio, no ciclo de processamento anterior (t-1);

- I_k é a k-ésima entrada externa;

- θ_i é o bias do i-ésimo neurônio;

- *w* refere-se ao peso correspondente.

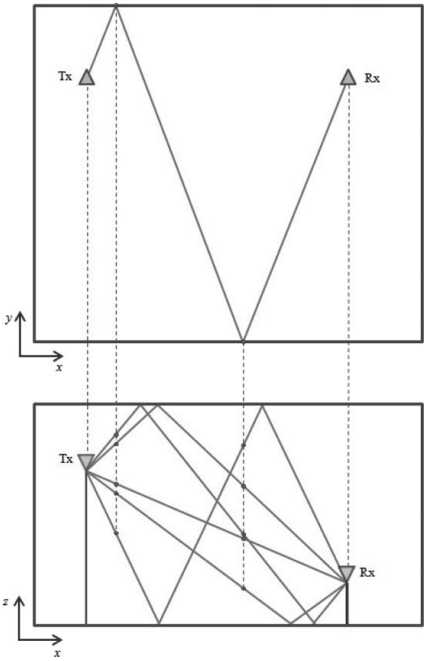
4.2 TÉCNICA DE TRAÇADO DE RAIOS "*Quasi*-3D" EM AMBIENTES INTERIORES

Para a proposta apresentada nesse trabalho, o algoritmo RTQ3D-*indoor* é utilizado para o cálculo de campo e avaliação de cobertura em ambientes interiores. O modelo RTQ3D-*indoor*, Grubisic (2012), usa o algoritmo traçado de raios 2D, com base na Teoria das Imagens para construir uma "árvore de imagens" e encontrar os diferentes raios refletidos para um determinado cenário. O banco de dados relacionado ao cenário de entrada é descrito no plano horizontal, mas adicionando a

altura do teto, do piso e as estações de transmissão (estações base) e de antenas de recepção (terminais de usuário) (GRUBISIC, 2013).

De acordo com o exemplo da Figura 10, o algoritmo de traçado de raios aplicado em ambientes interiores quase tridimensionais (RTQ3D-*indoor*) converte um determinado caminho inicialmente obtido a partir do algoritmo RT 2D em cinco raios: i) o caminho direto (do ponto de vista de um plano vertical); ii) o caminho refletido pelo solo; iii) o caminho refletido pelo teto; iv) o caminho refletido no chão e no teto e; V) o caminho refletido primeiro no teto e depois no chão (GRUBISIC, 2013).

Figura 10 – Passagem de um caminho 2D para cinco caminhos quasi-3D usando o algoritmo RTQ3D-*indoor*: a) plano horizontal; B) plano vertical.

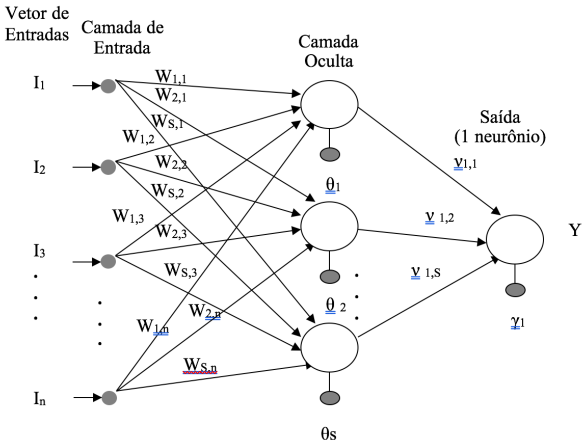


Fonte: Grubisic (2012) e (2013).

4.3 A RNA *PERCEPTRON* MULTICAMADAS

A Figura 11 ilustra o grafo das redes usadas aqui, que são as redes MLP (*Multilayer Perceptron – Perceptron* Multicamadas) *feedforward*. Estas redes se distinguem pela presença de uma ou mais camadas ocultas, cujos nós computacionais são chamados correspondentemente de neurônios ocultos ou unidades ocultas. A função dos neurônios ocultos é intervir entre a entrada externa e a saída de uma maneira útil (HAYKIN, 1999).

Figura 11 – Grafo ilustrativo da rede proposta.



Fonte: Pereira (2003).

Dois tipos de sinais são identificados na rede MLP (Haykin *apud* Parker, 1987):

- Sinais funcionais – É um estímulo que incide no terminal de entrada da rede, propaga-se para frente, neurônio por neurônio, através da rede e emerge no terminal de saída como um sinal de saída. Este sinal é tido como um sinal funcional, por duas razões: 1. Presume-se que ele realize uma função útil na saída da rede; 2. Em cada neurônio da rede através do qual um sinal funcional passa, o sinal é calculado como uma função de suas entradas e pesos associados, aplicados àquele neurônio. O sinal funcional é também conhecido como sinal de entrada;
- Sinais de erro – O sinal de erro se origina em um neurônio de saída da rede e se propaga para trás, camada por camada,

através da rede. Chama-se sinal de erro porque sua computação por cada neurônio da rede envolve uma função dependente do erro.

Cada neurônio oculto ou de saída de um *perceptron* multicamadas é projetado para realizar dois cálculos (Haykin, 1999):

- O cálculo do sinal funcional que aparece na saída de um neurônio, que é expresso como uma função não linear do sinal de entrada e pelos pesos sinápticos associados com aquele neurônio;
- O cálculo de uma estimativa do vetor gradiente (gradientes da superfície de erro em relação aos pesos conectados a entrada de um neurônio), que é necessário para retropropagação através da rede.

Após a determinação do valor de $net_i(t)$, equação 21, o estado do neurônio, ou ativação, é atualizado através da função de ativação $\phi(.)$. Analiticamente, a ativação do neurônio é obtida pela equação (22).

$$x_{t+h}=\phi[x_t,net_t,t,h] \tag{22}$$

Após a determinação do estado do neurônio $x(t)$, o sinal de saída do neurônio é calculado pela função de saída $\lambda(.)$, utilizando a equação (23).

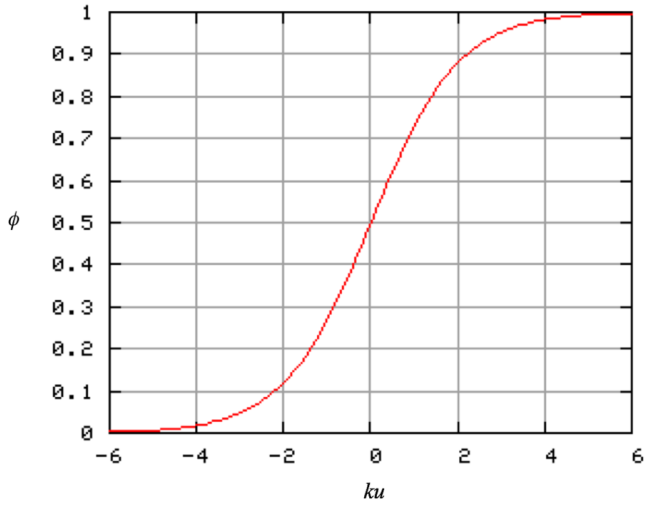
$$y(t) = \lambda[x(t),t] \tag{23}$$

A função de ativação para as duas camadas adotada neste estudo foi a função sigmoidal, figura 12 a qual é definida pela equação (24), (HAYKIN, 1999) (FREEMAN,1993).

$$\phi(ku) = \frac{1}{1+\exp(-ku)} \tag{24}$$

Onde k é um escalar positivo.

Figura 12 – Gráfico ilustrativo da função sigmoide.



4.4 ALGORITMO DE APRENDIZADO BACKPROPAGATION

Uma rede neural aprende através de um processo interativo de ajustes aplicados a seus pesos simpáticos e níveis de *bias* (limiar).

O algoritmo de aprendizado é um conjunto pré-estabelecido de regras bem definidas. Na fase de aprendizado de uma RNA, uma regra é usada para alterar os elementos da matriz de pesos e outros parâmetros modificáveis da rede usando a informação do meio externo disponibilizada pelo supervisor do aprendizado. No caso da rede utilizada neste trabalho, o supervisor fornece para a rede um conjunto de treinamento, ou seja, um conjunto de entradas e suas respectivas saídas desejadas. Deseja-se que a rede ajuste os seus pesos de forma a produzir, para cada entrada do conjunto de treinamento, a saída desejada fornecida pelo supervisor. Para cada entrada do conjunto de treinamento, a saída produzida pela rede neural é comparada com a saída desejada fornecida pelo supervisor e os pesos são alterados de forma a diminuir esta diferença (NASCIMENTO & YONEYAMA, 2000).

Para o algoritmo supervisionado por uma meta "*L"*, as alterações da matriz de pesos e do vetor *bias*, durante o processo de treinamento, são relacionadas através de uma taxa de aprendizado "ε", conforme as equações (25) e (26). Observa-se que a taxa de aprendizado "ε" é um valor maior que zero (0 < "ε" < 1).

$$\Delta W_{ik} = \varepsilon(L_i - y_i)I_k \tag{25}$$

$$\Delta bias_i = \varepsilon(L_i - y_i) \tag{26}$$

O erro (*e*) para a unidade de saída resultante de cada ajuste de pesos é relacionado pela meta *L* e a saída *y* em cada interação do treinamento, conforme equação 27.

$$e_k(i) = L_k(i) - y_k(i) \tag{27}$$

O erro quadrático calculado sobre as unidades de saída quando o padrão I_k é aplicado na entrada da rede é definido pela equação 28.

$$E_k = \frac{1}{2}\sum_{i=1}\left(L_k(i) - y_k(i)\right)^2 \tag{28}$$

Para as redes *feedforward* com uma camada oculta, devem-se considerar as matrizes de pesos da unidade oculta *j* e unidade de entrada $I(W_k(j,I))$ e da unidade de saída *i* e unidade oculta $j(\nu_k(i,j))$. E, a alteração destes pesos é definida pelas equações 29 e 30, respectivamente.

$$\Delta w_k(i,j) = -\varepsilon e_k(i)\phi_k(i)I_k(j) \tag{29}$$

$$\Delta \nu_k(i,j) = \varepsilon e_k(i)\lambda_k(i)y_k(j) \tag{30}$$

O algoritmo *backpropagation* minimiza a soma dos quadrados dos erros de saída, considerando-se sua média sobre o conjunto de treinamento, usando uma busca baseada na direção do gradiente.

Os algoritmos utilizados para treinar uma rede *feedforward* podem convergir para a solução de mínimo global rapidamente ou lentamente. Os algoritmos que convergem rapidamente podem ficar presos em pontos de mínimo local. Por outro lado, a taxa de aprendizagem é relacionada aos pesos sinápticos da seguinte forma: quanto menor for a taxa de aprendizagem, "ε", menores são as variações dos pesos sinápticos da rede, e mais suave é a trajetória no espaço de pesos; neste caso, porém, o processo de aprendizagem é lento. Aumentando-se a taxa de aprendizagem, "ε", modificações significativas nos pesos sinápticos podem tornar a rede instável. Entretanto, a inclusão de um termo de momento, isto é, constante de momento α, que é utilizada como um multiplicador na expressão de

atualização dos pesos sinápticos da camada de entrada, Equação (29), pode vir a compensar a instabilidade.

Segundo Pereira (2003) e (Soviersoski, 2009), geralmente não é possível demonstrar que o algoritmo de retropropagação irá convergir, e não existem critérios bem definidos para encerrar sua operação. Um dos métodos de parada usa a taxa de variação do erro médio quadrático, a qual deverá ser a menor possível. Isto, contudo, pode resultar em um encerramento prematuro do processo de aprendizagem. Outra possibilidade seria estabelecer um número de épocas razoável para um treinamento eficaz.

4.5 ALGORITMOS EVOLUCIONÁRIOS ASSISTIDOS POR METAMODELOS – ESTUDO DE CASO

O uso de metamodelos no contexto de algoritmos baseados em estratégias evolucionárias provou oferecer redução no tempo computacional. Tal redução resulta da pequena quantidade de cálculos exatos que esse método exige ou requer. Em particular, a função de estimação contribui grandemente para reduzir o tempo computacional enquanto a estimação do erro ajuda a incrementar a confiabilidade da convergência global em problemas multimodais complexos (EMMERICH, GIOTIS e OZDEMIR, 2002).

Nesta seção, a análise vai ser feita a partir dos metamodelos representados pelo método Kriging e pela rede neural *feedforward*.

Em Willmes, Bäck, Jin e Sendhoff (2003), um estudo de caso foi feito em que as redes neurais e o método Kriging são comparados na construção de modelos de aproximação adequados em algoritmos de otimização evolutivos.

Os dois modelos são aplicados em uma estrutura idêntica, para otimização de funções teste bem conhecidas.

Rede Neural – Modelos de redes neurais *feedforward* têm sido largamente utilizadas para aproximação de funções. Na maior parte dos casos, a rede é utilizada com o algoritmo *backpropagation* como método de aprendizado. Entretanto, esta estrutura de rede neural com o algoritmo *backpropagation* (BP) tem se mostrado frequentemente ineficiente. Por essa razão, com o objetivo de melhorar o desempenho da rede neural *feedforward,* é muito comum adotar uma variação mais rápida do algoritmo de aprendizado BP e otimizar a estrutura da rede neural para o problema dado (REED e MARKS, 1999 *apud* WILLMES, BÄCK, JIN e SENDHOFF, 2003).

Uma estrutura lamarquiana de algoritmos evolutivos é empregada para gerar o modelo da rede neural, Hiisken e Sendhoff (2000) *apud* Willmes, Bäck, Jin e Sendhoff (2003), em que ambos, a estrutura e os parâmetros, são otimizados. O dado disponível é partilhado entre o conjunto de dados de treinamento e um conjunto de dados de teste, e a função de adequação para o treinamento da rede neural é dada pela equação 31:

$$J = 0{,}75\, E_{TR} + 0{,}25\, E_{TT} \tag{31}$$

em que E_{TR} e E_{TT} são os erros aproximados nos dados de treinamento e nos dados de teste, respectivamente.

Um máximo de 10 neurônios na camada oculta é especificado e o tamanho da população para o treinamento da rede neural é 32. São rodadas 75 gerações. Três rodadas diferentes são conduzidas para cada conjunto de dados, e o melhor modelo de rede neural é selecionado para aproximação de adequação.

Kriging – O método utilizado para este estudo é o conhecido como Kriging padrão com estrutura de covariância Gaussiana, tendo sido descrito de forma detalhada na seção 3.3 do capítulo 3.

Problemas testados – Funções de teste bem conhecidas são usadas para comparar o desempenho dos algoritmos. São elas: a *função-Ackley*, *função-Rosenbrock* e a *função-Keane.* Todas as funções são para serem minimizadas e o problema tem dimensão *n*.

Estruturação do experimento – Foram conduzidas simulações das funções teste com dimensões de 10 e 50. Para cada função teste, foram rodadas 20 otimizações com a ajuda de um metamodelo já executado. Na estrutura de uso do metamodelo, o ciclo de controle e a frequência precisaram ser fixados. A fixação da frequência teve o objetivo de remover uma possível interferência entre fatores provenientes do metamodelo utilizado e aqueles oriundos da adaptação da frequência. Isto porque a proposta principal do estudo é investigar diferentes técnicas de metamodelagem.

Os experimentos foram conduzidos *offline* e *online*:

Para o experimento *offline*, todos os dados de aprendizado foram retirados de uma otimização prévia com estratégias evolutivas padrão.

No experimento *online,* novos dados de treinamento tornam-se disponíveis durante a otimização e estes dados devem ser utilizados na atualização dos metamodelos. A estrutura da rede neural é determinada com base no treinamento dos dados *offline* e não muda durante a

otimização. Há algumas diferenças na adaptação *online* do modelo Kriging. Durante o aprendizado *online*, um modelo Kriging separado é construído para cada nova descendência de indivíduos, cuja adaptação é para ser estimada.

Resultados para o aprendizado offline – A tabela 1 resume os resultados, em que o melhor desempenho foi marcado com o número "1", o pior com "-1" e é marcado com "0" o caso que não obteve nem a melhor nem o pior desempenho. Um resumo dos resultados obtidos com o modelo assistido e a estratégia evolutiva direta (não assitida) foi representado na tabela (WILLMES, BÄCK, JIN e SENDHOFF, 2003).

Tabela 1- Resumo dos resultados para o aprendizado *off-line*.

	Kriging	Rede Neural	**Modelo evolutivo direto**
Ackley dim 10	-1	0	**1**
Ackley dim 50	-1	0	**1**
Rosenbrock dim 10	1	-1	**1**
Rosenbrock dim 50	0	-1	**1**
Keane dim 10	0	1	**-1**
Keane dim 50	0	0	**0**
soma	**-1**	**-1**	**3**

Fonte: Willmes, Bäck, Jin e Sendhoff (2003).

Comparando a soma dos valores marcados neste experimento, observou-se que a estratégia direta conseguiu os maiores pontos.

Resultados para o aprendizado online – A tabela 2 resume os resultados, em que o melhor desempenho foi marcado com o número "1", o pior com "-1" e é marcado com "0" o caso que não obteve nem o melhor nem o pior desempenho. Um resumo dos resultados obtidos com o modelo assistido e a estratégia evolutiva direta (não assitida) foi representado na tabela (WILLMES, BÄCK, JIN e SENDHOFF, 2003).

Tabela 2 - Resumo dos resultados para o aprendizado *online*.

	Kriging	Rede Neural	**Modelo evolutivo direto**
Ackley dim 10	-1	1	**0**
Ackley dim 50	0	-1	**1**
Rosenbrock dim 10	1	-1	**0**
Rosenbrock dim 50	0	-1	**1**
Keane dim 10	-1	1	**0**
Keane dim 50	1	0	**-1**
soma	**-0**	**-1**	**1**

Fonte: Willmes, Bäck, Jin e Sendhoff (2003).

Comparando a soma dos valores marcados neste experimento, observou-se que o modelo evolutivo direto pode ser considerado, no todo, como sendo a melhor estratégia.

Conclusão – Neste estudo, três funções de teste diferentes, nomeadamente função *Ackley*, função *Rosenbrock* e função *Keane*, foram utilizadas para comparar a desempenho da otimização através de estratégias evolutivas assistidas por metamodelos. Como metamodelos, foram utilizados o método Kriging e a Rede Neural.

Para as duas técnicas, dois diferentes modos de construção foram considerados. No modo de aprendizado *offline,* dados de uma otimização feita previamente são usados para construir um modelo antes de ele ser aplicado na otimização. No modo de aprendizado *online*, o metamodelo é atualizado repetidamente com novos dados gerados da otimização. Em todos os casos, o modo de aprendizado *online* apresentou resultados significativamente melhores do que o modo *offline*.

Nem o modelo Kriging nem a rede neural utilizados como metamodelos puderam demostrar um desempenho vantajoso em relação a uma otimização evolutiva sem utilização dos metamodelos, como pudemos observar através das Tabelas 1 e 2.

4.6 CONSIDERAÇÕES FINAIS SOBRE A APLICAÇÃO DE METAMODELOS

Nos últimos anos, algoritmos genéticos, redes neurais e métodos de otimização evolutiva, como PSO, têm se tornado populares para a otimização de problemas em eletromagnetismo. Entretanto, esses métodos estão de forma geral associados a uma lenta convergência.

Portanto, a justificativa para utilização da metamodelagem teve sua base num aspecto principal que é a redução do custo computacional dos projetos que envolvem algoritmos evolutivos.

O uso de metamodelos permite tornar mais eficaz o processo de otimização, evitando cálculos de avaliação da função objetivo na avaliação da qualidade de uma solução. A ideia é usar uma função mais simples que permita avaliar a qualidade de uma solução sem ter que avaliar a função objetivo de maneira direta.

Segundo Jin (2005), a qualidade do modelo de aproximação precisa ser provada tanto quanto possível para um número limitado de dados. Vários aspectos são importantes para demonstrar a qualidade do modelo, tais como: a seleção do modelo, o uso do dado ativo

amostrando e atribuindo um peso a eles, a seleção do método de treinamento e a seleção da medida de erro a ser utilizada.

Em Mohammed, Park, Üler e Zigiang, (1992), um novo método de otimizar projetos de dispositivos eletromagnéticos é apresentado. O método utiliza redes neurais artificiais no ambiente do projeto, o qual engloba computação numérica e dados de entrega dinâmicos para gerar uma variabilidade de dados de treinamento da rede neural artificial.

Pôde-se concluir, a partir dos estudos analisados, que o modelo Kriging e a rede neural *feedforward* são robustos e oferecem economia no tempo computacional. Em particular, a função de estimação ajuda a acelerar a convergência global. Através dos estudos feitos, os dois metamodelos apresentam desempenho bastante similar. Por isso, optou-se pelo método que tem sido mais largamente utilizado em aplicações eletromagnéticas, que são as RNAs, não só pela facilidade de implementação, mas também pela possibilidade de trabalhar em paralelo com as RNAs complexas visando verificar suas características de robustez. A partir dos estudos analisados, verificou-se também que os algoritmos evolutivos assistidos por metamodelos, mais especificamente o modelo Kriging e a rede neural *feedforward*, são robustos e oferecem economia no tempo computacional. Esta economia resulta do fato dos algoritmos evolutivos assistidos requererem um menor número de cálculos exatos. Em particular, a função de estimação ajuda a acelerar a convergência global.

4.7 CONSIDERAÇÕES FINAIS DO CAPÍTULO

Esse capítulo possibilitou a análise de alguns dos principais metamodelos (a RNA *Perceptron* com algoritmo *backpropagation*), que podem assistir algoritmos evolucionários, mais especificamente o PSO, que é ferramenta deste trabalho.

O estudo de caso apresentado possibilitou a decisão pela melhor opção de metamodelo, em que foram pesados a facilidade de implementação e a eficiência na resposta. Ficou-se entre a RNA e o modelo Kriging. O que determinou a opção pela RNA, como metamodelo desse estudo, foi a experiência prévia da autora em trabalhar com redes neurais, o que facilitou a implementação e poupou tempo na construção dos códigos dos algoritmos necessários ao desenvolvimento dessa tese. Visto que os esses dois metamodelos apresentam respostas bastante similares quanto à eficiência.

No capítulo 5 mostram-se os resultados conseguidos a partir de uma implementação da ferramenta de otimização desenvolvida em

funções analíticas, tais como: Sinc, picos e rastringin. Por fim foi feito um estudo de caso Real proposto por Grubisic (2012), utilizando a ferramenta desenvolvida e outra implementação no campo dos projetos eletromagnéticos, (caso 6 – proposto por Grubisic (2012)).

5 RESULTADOS

Neste capítulo, vão ser apresentados os resultados conseguidos no projeto desenvolvido. Primeiramente, foram utilizadas duas funções teste cujo objetivo era verificar e validar os resultados através da aplicação do algoritmo evolutivo PSO (*Particle Swarm Optimization*) assistido pela RNA (Rede Neural Artificial). Dando sequência, foi feito um estudo de caso real utilizando as ferramentas propostas por esse trabalho, quais sejam: Algoritmo evolutivo (PSO) assistido por um metamodelo (RNA, *Multilayer Perceptron* com algoritmo de aprendizado *backpropagation real*), com o objetivo de diminuir o custo computacional.

5.1 FERRAMENTAS UTILIZADAS NA IMPLEMENTAÇÃO

Baseado nos estudos de Grubisic (2012 e 2013), optou-se pela utilização do algoritmo evolutivo PSO, desenvolvido por Kennedy & Eberhart (1995), (seção 2.1, figura 8).

Como metamodelo, optou-se em função dos testes e estudos realizados por Pereira (2003) e Soviersoski (2009), em termos da topologia, pela utilização de redes *feedforward* com uma camada oculta. Como padrão para implementação deste estudo, foram utilizadas as redes *feedforward* (seção 4.3, Figura 11), MLP (*Multilayer Perceptron*) com algoritmo de aprendizado *backpropagation* real. O número de neurônios da camada intermediária foi calculado a partir da média geométrica do número de neurônios da entrada da rede com o número de neurônios na saída da rede, equação 32, (PEREIRA, 2003).

$$Mg = \sqrt{Nne.NnS} \tag{32}$$

Mg – Média geométrica;
Nne – Número de neurônios na entrada;
NnS – Número de neurônios na Saída.

No que se refere aos parâmetros utilizados no treinamento da RNA escolhida: o momento que melhor se adequou ao tipo de dado foi 0,7. Os demais valores de momento 0,8 e 0,9, que em algumas aplicações são possíveis de serem utilizados, não permitiram que a rede finalizasse os treinamentos. Os mesmos foram finalizados com a mensagem de NAN (*Not a Number)*; com relação à constante de

aprendizado (ε), a que melhor se adequou foi a de valor 0,1. As demais tentativas, de 0,15 e 0,2, finalizaram os treinamentos com a mesma mensagem NAN.

Outro aspecto que foi observado antes da realização dos treinamentos foi que o valor numérico do campo magnético era da ordem de 10^{-8}, o que levou à necessidade de normalização dos valores. Isso foi feito calculando a raiz oitava dos valores obtidos, para facilitar os cálculos da rede e com isso torná-la mais eficiente e rápida.

5.2 OTIMIZAÇÃO DA FUNÇÃO TESTE

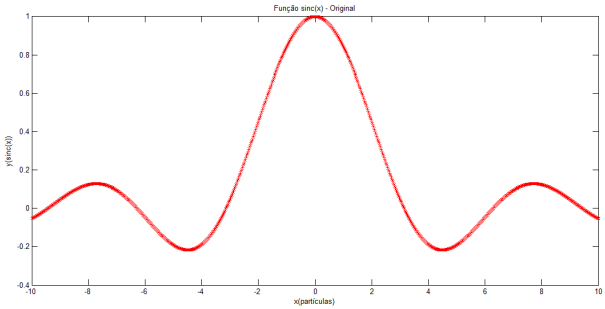
Foram realizados testes com a rede neural RMLP, com o objetivo de verificar a eficiência e eficácia do metamodelo escolhido e implementado. As funções escolhidas para os testes foram a função Sinc e a função picos.

5.2.1 Função Sinc

A função sinc(x) foi a primeira utilizada para verificar a eficácia do método proposto. Sua utilização como uma das funções teste deve-se ao fato de ela ter solução analítica. Especificamente, o objetivo foi encontrar o máximo global da função , dada na equação 33 e mostrada na Figura 13.

$$\sin c(x) = \frac{\sin(\pi x)}{\pi x} \tag{33}$$

Figura 13 - gráfico do traçado da função sinc(x).

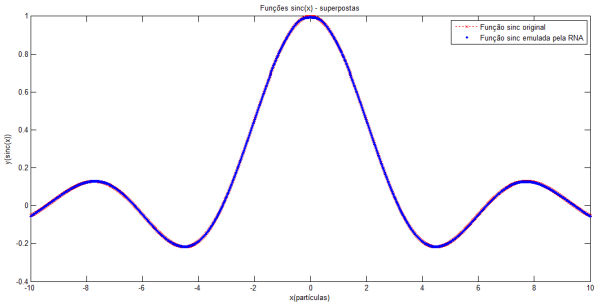


Fonte: o autor.

O objetivo desta implementação foi testar a eficácia do algoritmo evolutivo PSO usando a RNA como um metamodelo. A otimização foi feita utilizando a função dada na equação 33, com o metamodelo RNA para representá-la. Especificamente, o algoritmo PSO foi usado para obter a localização do máximo global da função sinc(x), nomeado por Pbest, que apresentou valor máximo de 0,994. Como o valor máximo da função sinc(x) é 1, o erro foi de 0,6%. O algoritmo foi inicializado com um conjunto aleatório de partículas e o processo teve lugar até que o ponto máximo foi localizado, o pico da onda que representa a função sinc(x).

A figura 14 mostra o gráfico de contorno dos resultados obtidos utilizando-se diretamente a equação 33, e os resultados obtidos usando a RNA para simular a função dada pela equação 33. O máximo global é o pico da curva da função sinc(x), o Pbest de 0,994.

Figura 14 - gráfico dos traçados das funções sinc(x), original e simulada pela RNA.



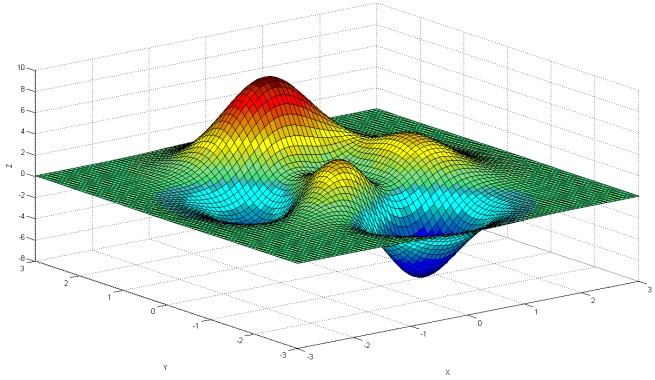
Fonte: o autor.

5.2.2 Função Picos

A segunda verificação da eficácia do método proposto foi feita aplicando-se o método na otimização de outra função de teste com solução analítica. Especificamente, o objetivo foi encontrar o máximo global da "função picos" de duas variáveis, dada na equação 34 e mostrada na Figura 15.

$$F(x,y) = 3(1-x)^2 e^{-x^2-(y+1)^2} - \left(2x - 10x^3 - 10y^5\right) e^{-x^2-y^2} - \frac{1}{3} e^{-(x+1)^2-y^2} \quad (34)$$

Figura 15 – Função analítica considerada (função picos).

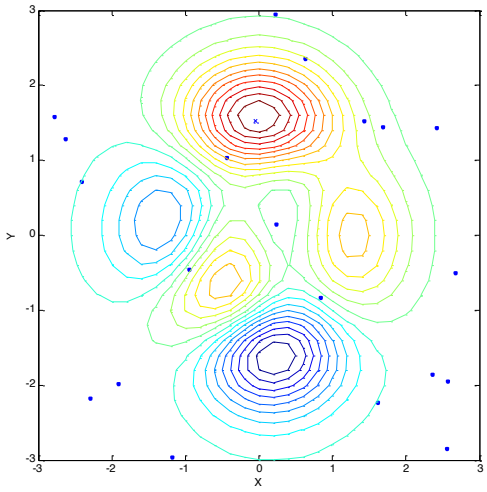


Fonte: o autor.

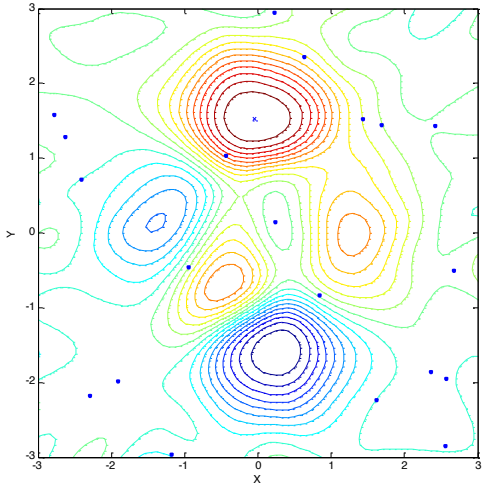
O objetivo desta implementação foi testar a eficácia do algoritmo evolutivo PSO usando a RNA como um metamodelo. A otimização foi feita utilizando a função dada na equação 34, com o metamodelo RNA para representá-la. Especificamente, o algoritmo PSO foi usado para obter a localização do máximo global da função picos. O algoritmo foi inicializado com um conjunto aleatório de partículas e o processo teve lugar até que o ponto máximo foi localizado, no centro dos contornos vermelhos (ver figura 16).

A figura 16 (a) mostra o gráfico de contorno dos resultados obtidos utilizando-se diretamente a equação 34, enquanto que a figura 16 (b) mostra os resultados obtidos usando a RNA para simular a função dada pela Equação 34. O máximo global está representado por "x" na figura 16 (a).

Figura 16 – (a). Gráfico de contorno da função real, (b) gráfico de contorno da função simulada, com o pico localizado.



(a)



(b)

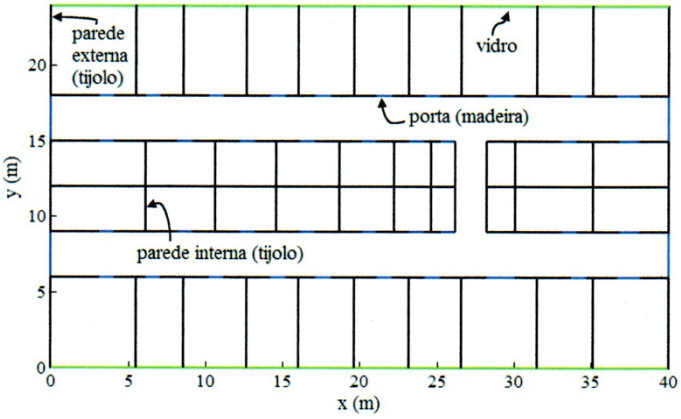
Fonte: o autor.

A RNA foi treinada até que um erro médio quadrático de 0,0004 fosse obtido. Para tal, foram necessárias 27.253 épocas. Uma época é um passo no processo de formação de uma RNA, isto é, cada vez que a rede é apresentada com um novo padrão de entrada. O erro médio quadrático do PSO assistido pela RNA foi igual a 0,00569. A partir dos resultados, podemos concluir que o uso do metamodelo proposto para substituir a função de *fitness* original sugere a obtenção de resultados satisfatórios, no que se refere à emulação da função *fitness* original e à localização dos pontos relativos aos picos da função, como se pode observar através da figura 16(a) e (b).

5.3 ESTUDO DE CASO

Foi utilizado um dos casos estudados por Grubisic (2012), correspondente a um ambiente *indoor*, de 40 m por 28 m, cuja planta baixa é mostrada na figura 17. Nesse cenário, a altura do teto em relação ao solo é de 3 m.

Figura 17 – Planta baixa do cenário avaliado, com detalhes dos materiais dos obstáculos.



Fonte: Grubisic (2012).

A utilização de um modelo de previsão eletromagnética representa um procedimento de projeto importante em sistemas sem fio. Aqui, foi usado um modelo de propagação conhecido como "traçado de raios quase 3D" (RT3D, *ray-tracing quasi*-3D) associado a um otimizador evolutivo (PSO) para encontrar o posicionamento ideal de duas antenas em um cenário fechado (GRUBISIC *et alli*, 2012).

Inicialmente, foi gerada uma população em que cada partícula corresponde às coordenadas (x, y) das antenas de transmissão posicionadas em um cenário de 40m × 28m. Definiu-se uma malha de 160 receptores distribuídos uniformemente dentro do ambiente. As antenas de todos os receptores foram consideradas a 1,5 m de altura do chão. O método foi aplicado para minimizar o número de pontos em que o campo está abaixo de um limiar mínimo de recepção pré-estabelecido (-110 dBm). Isso é equivalente a maximizar a área de cobertura. A variável do problema foi assumida como sendo um vetor contendo as coordenadas (x, y) das antenas de transmissão, permitindo diferentes localizações delas no plano horizontal do cenário. A altura dessas antenas de transmissão foi fixada (3 m), simulando antenas monopolo de quarto de onda verticalmente polarizadas fixadas ao teto. Duas antenas foram consideradas no problema. Portanto, a variável do problema é um vetor contendo quatro elementos, os quais representam as coordenadas de posição da antena de transmissão no plano horizontal. A frequência utilizada foi de 2,4 GHz, e o total da potência equivalente isotrópica radiada (EIRP) foi de 400 mW. Foi simulada uma WLAN com duas estações base que devem servir para atender o ambiente indoor em questão.

5.3.1 Desenvolvimento do estudo de caso

Em primeiro lugar, os valores dos campos foram gerados pelo traçado de raios e usados como treinamento no processo de aprendizagem realizado pela RNA, onde os valores dos pesos da rede, otimizados, foram salvos em arquivo.

Após o treinamento, a RNA atua como metamodelo no processo de otimização realizado pelo algoritmo evolucionário PSO, na procura do melhor posicionamento das antenas de acordo com o estabelecido, que foi minimizar o número de pontos em que o campo está abaixo de um limiar mínimo de recepção pré-estabelecido (-110 dBm), permitindo uma grande redução no tempo computacional, conforme pode-se verificar na figura 5, capítulo 1.

O PSO assistido pela RNA foi inicializado com um conjunto de 100 partículas aleatórias, visto que o aumento do número de partículas iniciais não fez diferença no que diz respeito à eficiência do otimizador, só impactando no tempo computacional. A otimização do PSO foi realizada com quatro números diferentes de épocas: 1.500; 3.000; 4.500 e 6.000. A Tabela 3 mostra os resultados para a PSO realizada com 3.000 épocas.

Tabela 3 - Resultado da otimização da posição das antenas através do PSO assistido pela RNA (3.000 épocas).

Épocas	Posição Antena 1	Posição Antena 2	Treinamento da RNA(10^{-8})
3.000	(X1,Y1)	(X2,Y2)	Erro, (1000 Neurônios de entrada), Primeiro conjunto de valores gerados pelo *RTQ3D*
1	(34,5258;15,5024)	(36,7256;22,5776)	100
2	(9,1315;17,6006)	(5,5487;6,5383)	10
3	**(20,1806; 7,5598)**	**(29,1253; 15,8245)**	**1**
4	(32,1313;8,5765)	(20,1839;8,4862)	0,1
			Erro, (1000 Neurônios de entrada), segundo conjunto de valores gerados pelo *RTQ3D*
5	(17,6887;13,1869)	(6,6639;14,2438)	100
6	(9,1315;17,6006)	(5,5487;6,5383)	10
7	(23,9788;19,3975)	(34,7084;11,0369)	1
8	(17,6887;13,1869)	(6,6639;14,2438)	0,1
			Erro, (1500 Neurônios de entrada), Primeiro conjunto de valores dados gerados pelo *RTQ3D*
9	(1,7395;6,7603)	(30,2211;6,3780)	100
10	(37,5801;23,0546)	(35,4780;19,8038)	10
11	(31,9132;11.0265)	(36,4969;8,0865)	1
12	(38,3427;6,3751)	(36,4537;19,0137)	0,1
			Erro, (1500 Neurônios de entrada), segundo conjunto de valores dados gerados pelo *RTQ3D*
13	(32,1066;22,6347)	(0,9928;4,9075)	100
14	(39,1181;1,9816)	(9,5419;3,1200)	10
15	(15,5100;19,4306)	(1,6609;3,8072)	1

Fonte: o autor

A Tabela 4 mostra os resultados obtidos para a PSO rodando com 4.500 épocas.

Tabela 4 - Resultado da otimização da posição das antenas através do PSO assistido pela RNA (4.500 épocas).

Épocas	Posição Antena 1	Posição Antena 2	Treinamento da RNA(10^{-8})
4,500	(X1,Y1)	(X2,Y2)	Erro, (1000 Neurônios de entrada), Primeiro conjunto de valores gerados pelo *RTQ3D*
1	(15,5595;4,1079)	(25,4087;4,8639)	100
2	(10,6277;19,6037)	(24,7960;2,6053)	10
3	(32,1159;20,5740)	(3,7432;20,4082)	1
4	(39,5645;5,5205)	(39,5444;11,4444)	0,1
			Erro, (1000 Neurônios de entrada), segundo conjunto de valores dados gerados pelo *RTQ3D*
5	(16,9551;20,5799)	(27,4131;8,1028)	100
6	(25,8862;7,1906)	(9,5124;8,4026)	10
7	(36,8088;3,3697)	(38,1096;1,7514)	1
8	(29,0186;9,7174)	(16,3957;0,6265)	0,1
			Erro, (1500 Neurônios de entrada), Primeiro conjunto de valores dados gerados pelo *RTQ3D*
9	(3,9611;15,7197)	(7,3387;1,6850)	100
10	(23,7305;21,5002)	(4,5864;11,4893)	10
11	(8,1750;4,5360)	(18,1765;20,6072)	1
12	(19,1449;17,1025)	(6,1786;1,9000)	0,1
			Erro, (1500 Neurônios de entrada), segundo conjunto de valores dados gerados pelo *RTQ3D*
13	(18,3303;1,7648)	(6,0661;19,8098)	100
14	(19,5186;12,8861)	(36,0429;22,1556)	10
15	(26,6234;12.5524)	(17,7279;10,9958)	1

Fonte: o autor

As tabelas 5 e 6 mostram os resultados obtidos para para a PSO rodando com 1.500 e 6.000 épocas.

Tabela 5 - Resultado da otimização da posição das antenas através do PSO assistido pela RNA (1.500 épocas).

Épocas	Posição Antena 1	Posição Antena 2	Treinamento da RNA(10^{-8})
1.500	(X1,Y1)	(X2,Y2)	Erro, (1000 Neurônios de entrada), Primeiro conjunto de valores gerados pelo *RTQ3D*
1	(11,9797;17,5184)	(20,9897;7,5143)	100
2	(28,9985;6,3813)	(31,4013;13,3475)	10
3	(36,0765;20,7300)	(23,6184;2,4970)	1
4	(12,2671;4,4928)	(3,4533;3,9909)	0,1

Fonte: o autor

Tabela 6 - Resultado da otimização da posição das antenas através do PSO assistido pela RNA (6.000 épocas).

Épocas	Posição Antena 1	Posição Antena 2	Treinamento da RNA(10^{-8})
6.000	(X1,Y1)	(X2,Y2)	Erro, (1000 Neurônios de entrada), Primeiro conjunto de valores gerados pelo *RTQ3D*
1	(36,4769;3,5231)	(11,0845;8,4839)	100
2	(18,2386;14,3416)	(4,6630;10,4786)	10
3	(21,2107;10,0983)	(0,9568;23,0432)	1
4	(38,2818;0,9824)	(38,9842;18,9649)	0,1

Fonte: o autor

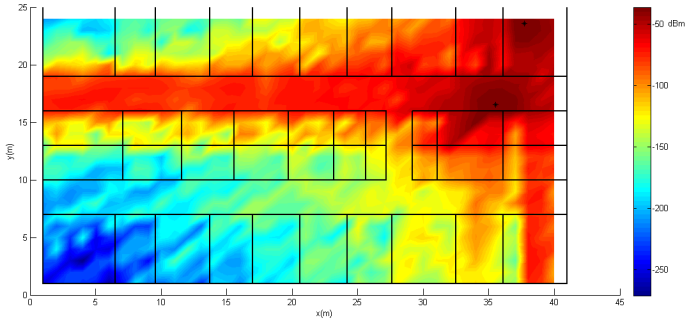
Os valores mostrados, em cada uma das tabelas de resultados, dizem respeito as localizações para cada uma das duas antenas, localizações essas obtidas pelo otimizador PSO assistido pela RNA, para os cinco casos analisados. A ordem de apresentação das tabelas foi estabelecida em função da melhor resposta para a localização das antenas.

As otimizações foram feitas com dois conjuntos de valores de campo magnético gerados pela *RTQ3D*, referentes ao conjunto de posições de duas antenas. Em seguida, através do treinamento da Rede neural artificial MLP *(Multi-layer Perceptron)*, esses valores de campo são aprendidos em relação às posições dos 160 receptores considerados nesse estudo e distribuídos uniformemente dentro do ambiente. Após o treinamento da rede a PSO com a RNA, determinam-se as posições ótimas. O método foi aplicado para minimizar o número de pontos em que o campo está abaixo de um limiar mínimo de recepção pré-

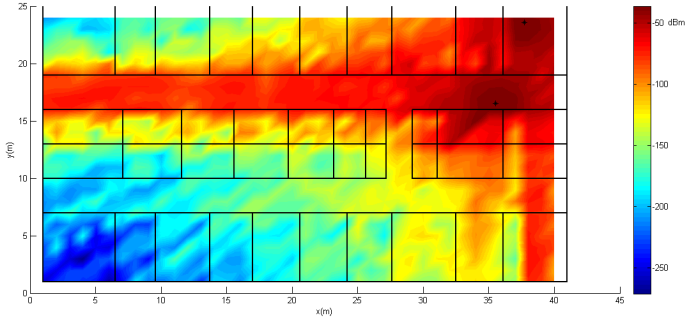
estabelecido (-110 dBm). A figura 5, no capítulo 1, faz uma ilustração do fluxo de trabalho seguido pela otimização.

As figuras 18, 19, 20, (a), (b), (c) e (d) e 21 (a), (b) e (c) mostram o mapeamento de potências recebidas, elencados na Tabela 3, para uma otimização através do algoritmo PSO assistido pela RNA com 3.000 épocas.

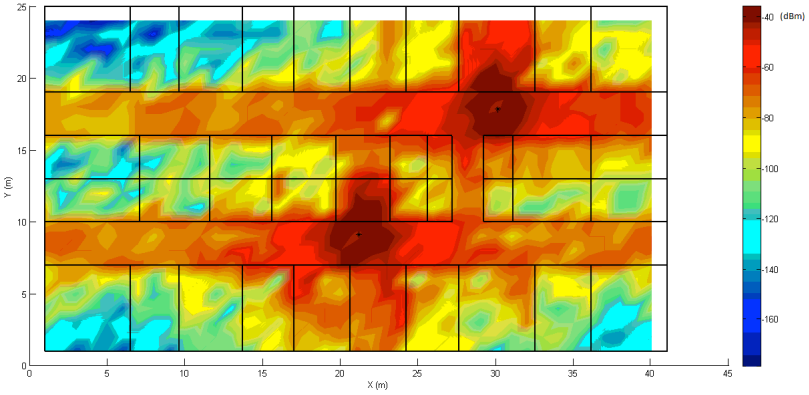
Figura 18 (a), (b), (c) e (d) - mostra os mapeamentos de potências recebidas, de acordo com a Tabela 3. Primeiro conjunto de valores gerados pela RTQ3D e 1.000 neurônios na entrada da RNA de treinamento.



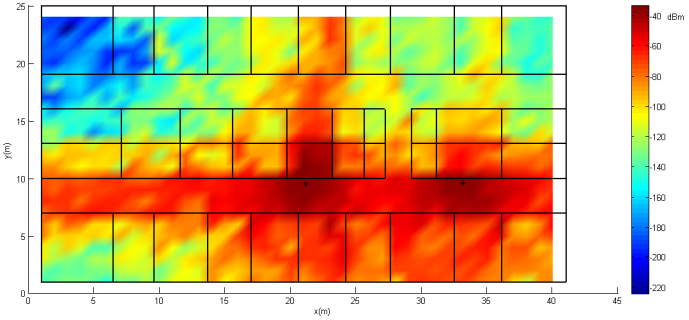
(a)



(b)

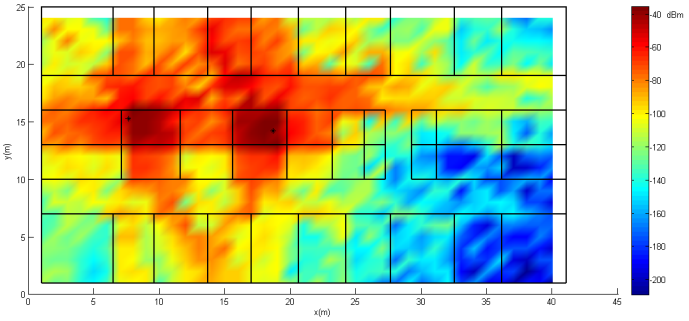


(c)

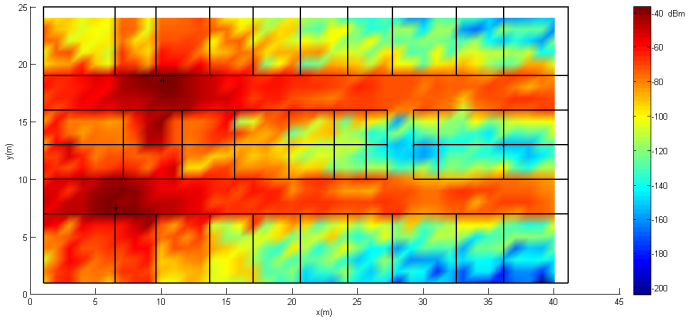


(d)

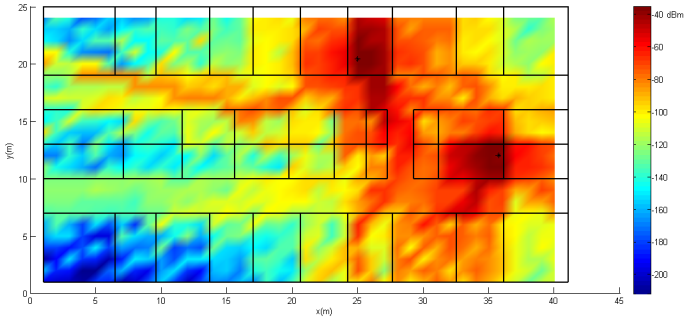
Figura 19 (a), (b), (c) e (d) - mostra os mapeamentos de potências recebidas, de acordo com a Tabela 3. Segundo conjunto de valores gerados pela RTQ3D e 1000 neurônios na entrada da RNA de treinamento.



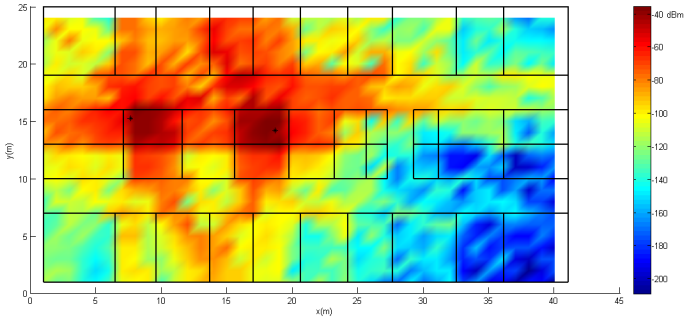
(a)



(b)

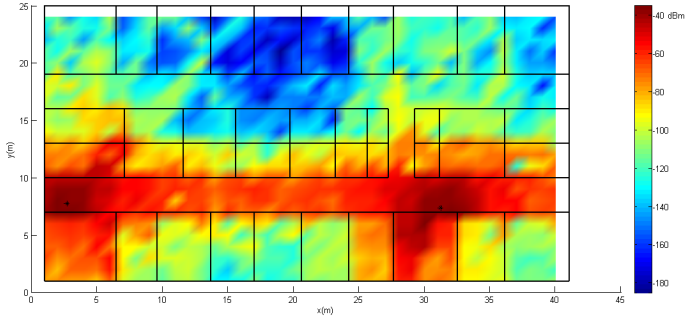


(c)

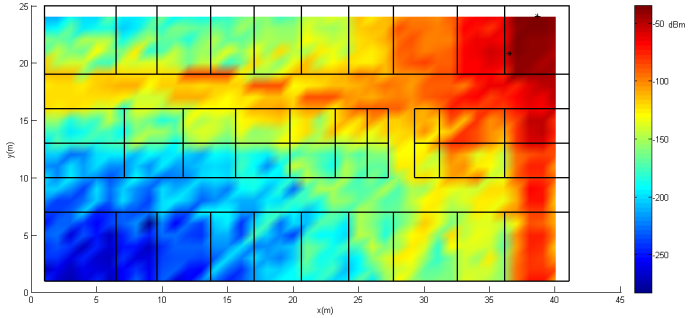


(d)

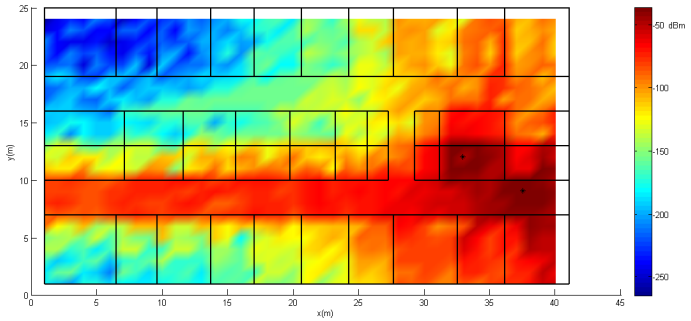
Figura 20 (a), (b), (c) e (d) - mostra os mapeamentos de potências recebidas, de acordo com a Tabela 3. Primeiro conjunto de valores gerados pela RTQ3D e 1.500 neurônios na entrada da RNA de treinamento.



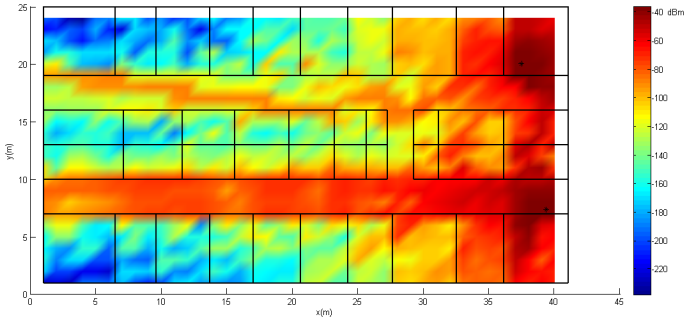
(a)



(b)

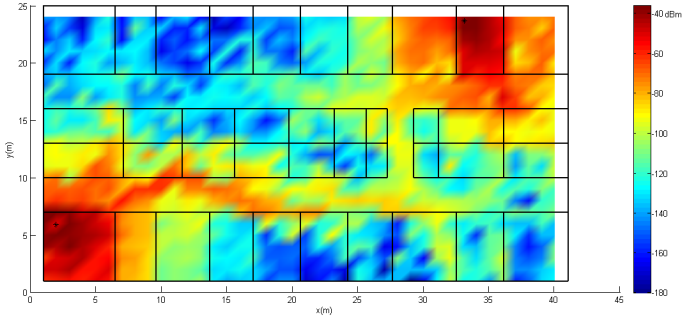


(c)

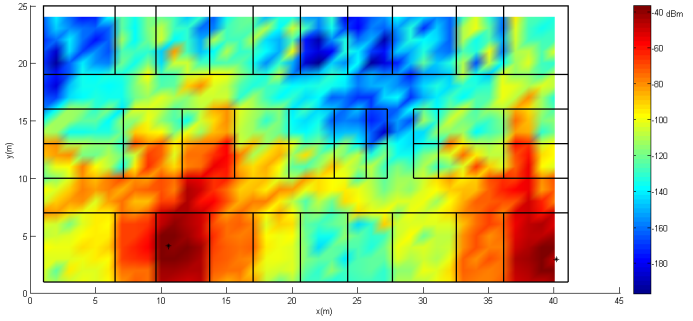


(d)

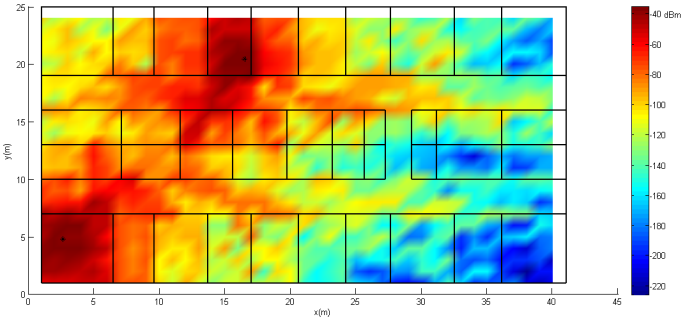
Figura 21 (a), (b) e (c) - mostra os mapeamentos de potências recebidas, de acordo com a Tabela 3. Segundo conjunto de valores gerados pela RTQ3D e 1.500 neurônios na entrada da RNA de treinamento.



(a)



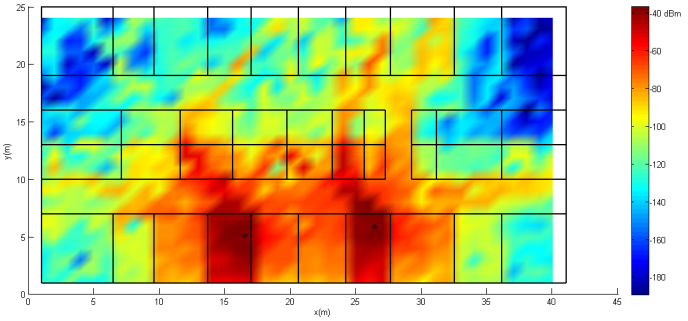
(b)



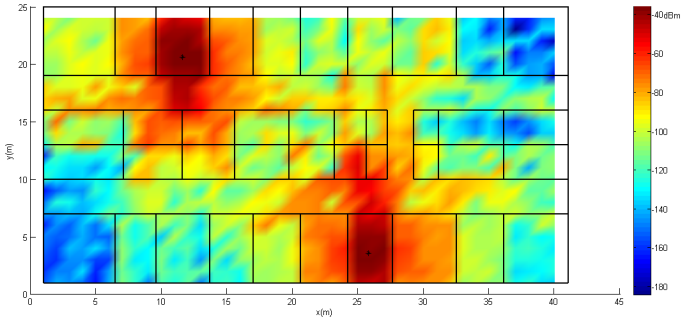
(c)

As figuras 22, 23, 24, (a), (b), (c) e (d) e 25 (a), (b) e (c) mostram o mapeamento de potências recebidas, elencados na Tabela 4, para uma otimização através do algoritmo PSO assistido pela RNA com 4.500 épocas.

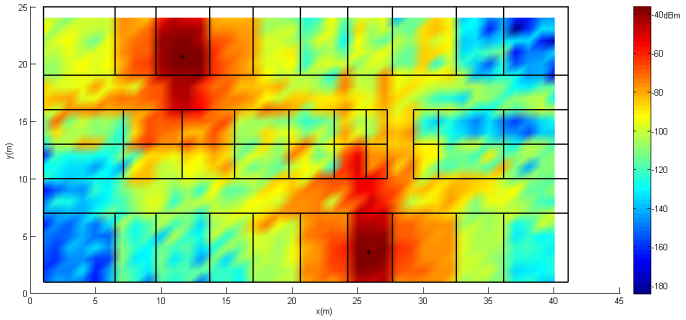
Figura 22 (a), (b), (c) e (d) - mostra os mapeamentos de potências recebidas, de acordo com a Tabela 4. Primeiro conjunto de valores gerados pela RTQ3D e 1.000 neurônios na entrada da RNA de treinamento.



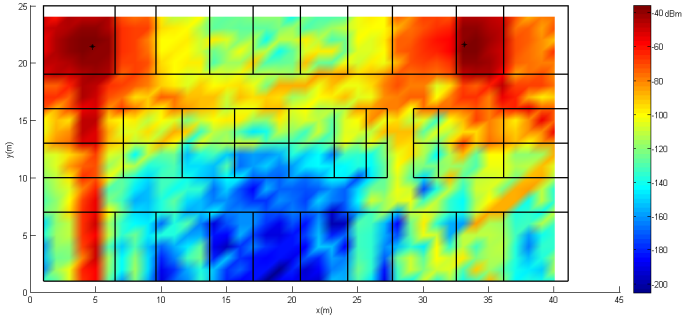
(a)


y(m)
x(m)
-40dBm
-60
-80
-100
-120
-140
-160
-180

(b)

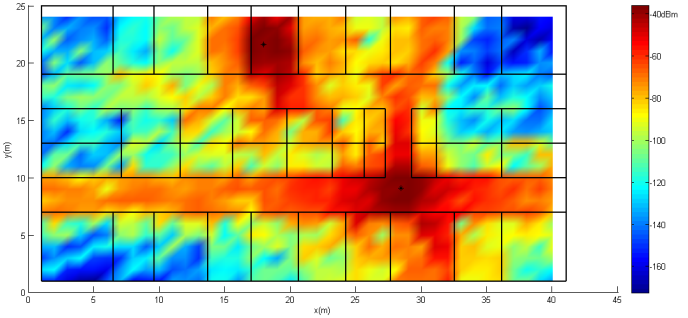

y(m)
x(m)
-40dBm
-60
-80
-100
-120
-140
-160
-180

(c)

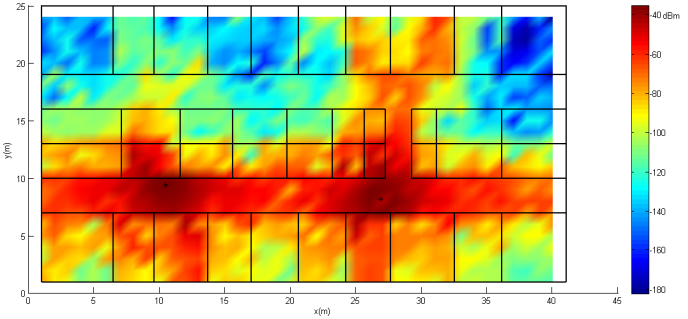

y(m)
x(m)
-40 dBm
-60
-80
-100
-120
-140
-160
-180
-200

(d)

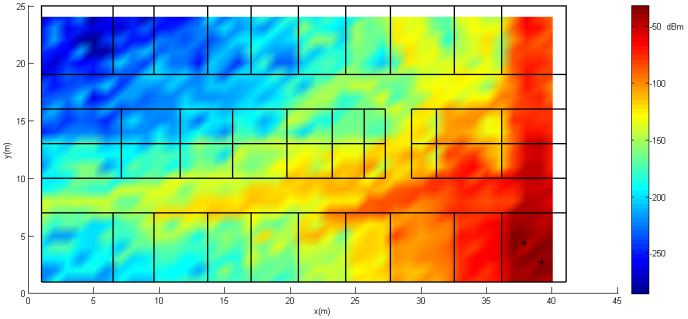
Figura 23 (a), (b), (c) e (d) - mostra os mapeamentos de potências recebidas, de acordo com a Tabela 4. Segundo conjunto de valores gerados pela RTQ3D e 1.000 neurônios na entrada da RNA de treinamento.



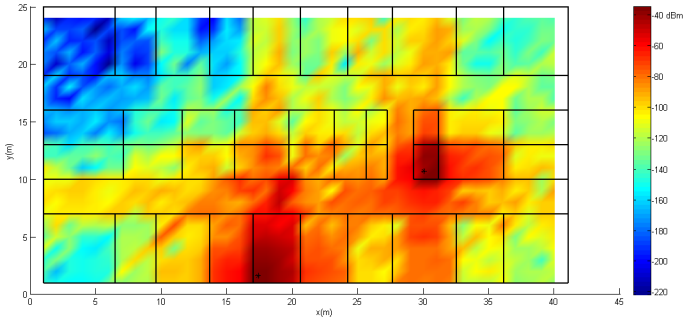
(a)



(b)

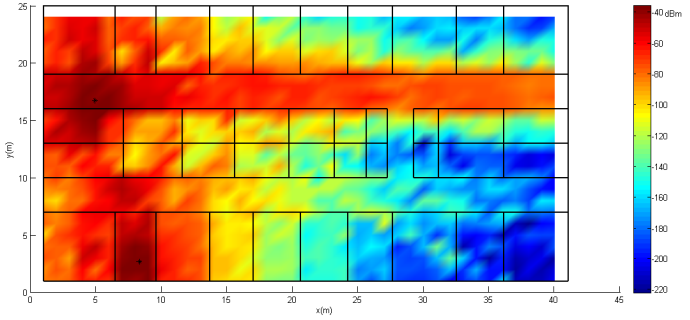


(c)

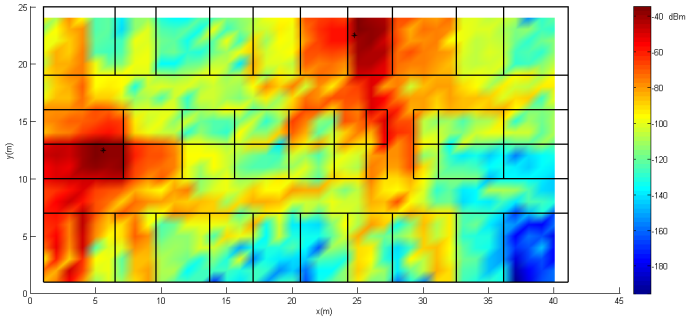


(d)

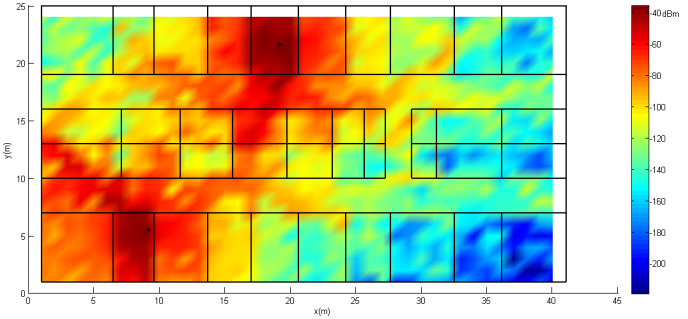
Figura 24 (a), (b), (c) e (d) - mostra os mapeamentos de potências recebidas, de acordo com a Tabela 4. Primeiro conjunto de valores gerados pela RTQ3D e 1.500 neurônios na entrada da RNA de treinamento.



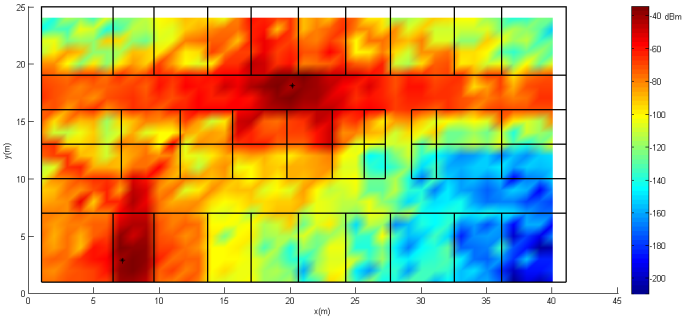
(a)



(b)

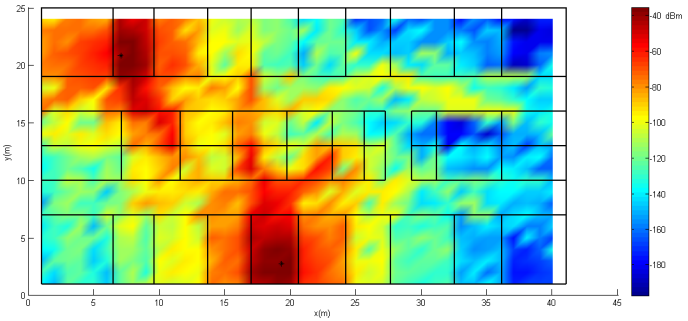


(c)

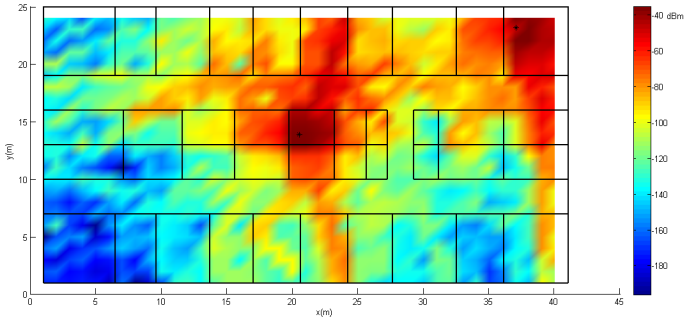


(d)

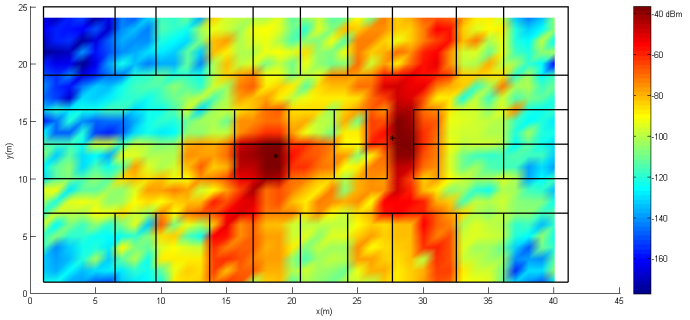
Figura 25 (a), (b) e (c) - mostra os mapeamentos de potências recebidas, de acordo com a Tabela 4. Segundo conjunto de valores gerados pela RTQ3D e 1.500 neurônios na entrada da RNA de treinamento.



(a)



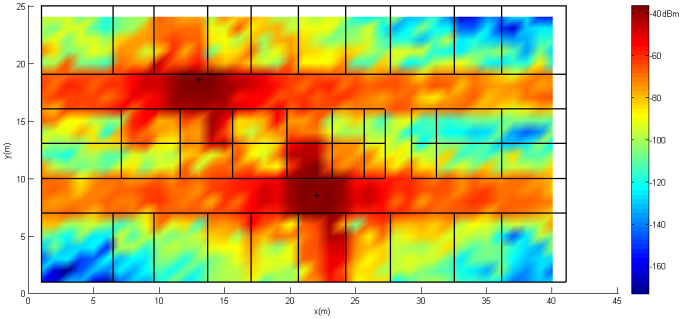
(b)



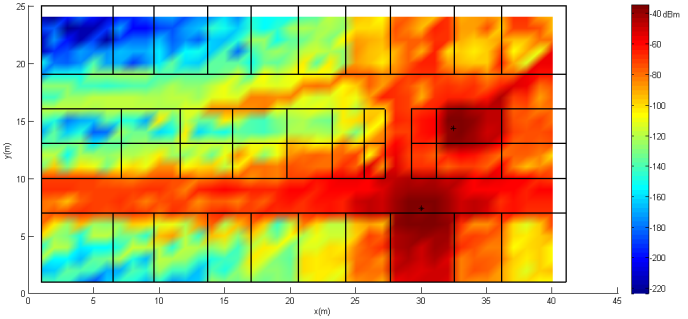
(c)

A figura 26 (a), (b), (c) e (d) mostram o mapeamento de potências recebidas, elencados na Tabela 5, para uma otimização através do algoritmo PSO assistido pela RNA com 1.500 épocas.

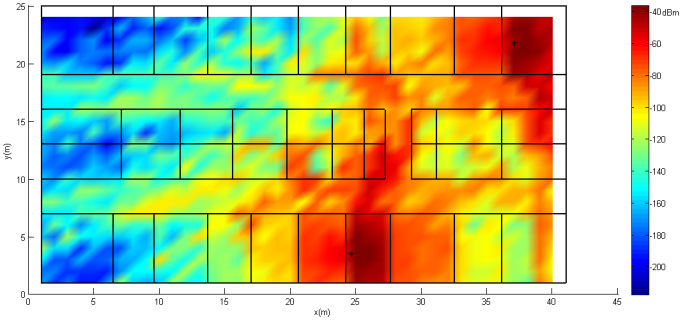
Figura 26 (a), (b), (c) e (d) - mostra os mapeamentos de potências recebidas, de acordo com a Tabela 5. Primeiro conjunto de valores gerados pela RTQ3D e 1.000 neurônios na entrada da RNA de treinamento.



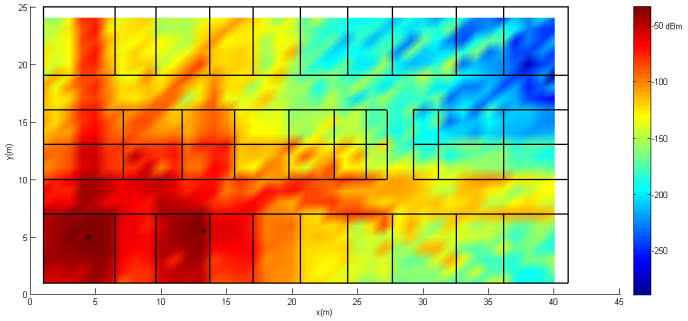
(a)



(b)



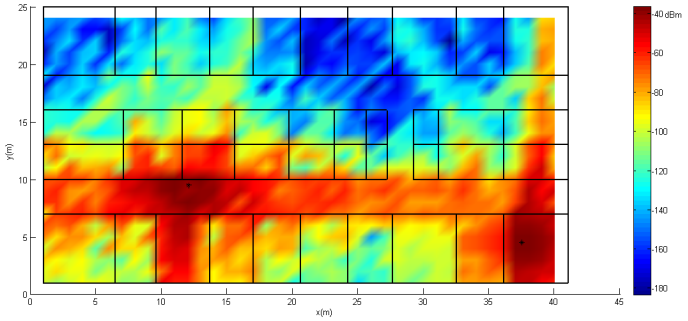
(c)



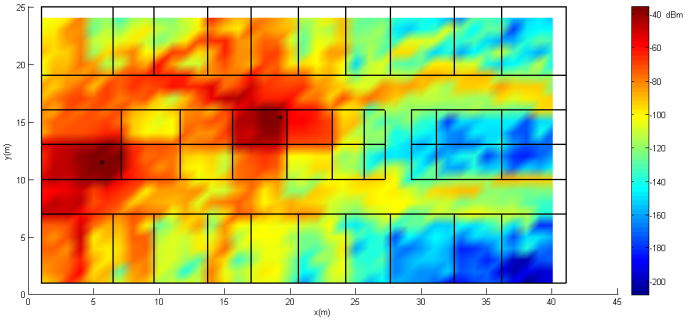
(d)

A figura 27 (a), (b), (c) e (d) mostram o mapeamento de potências recebidas, elencados na Tabela 6, para uma otimização através do algoritmo PSO assistido pela RNA com 6.000 épocas.

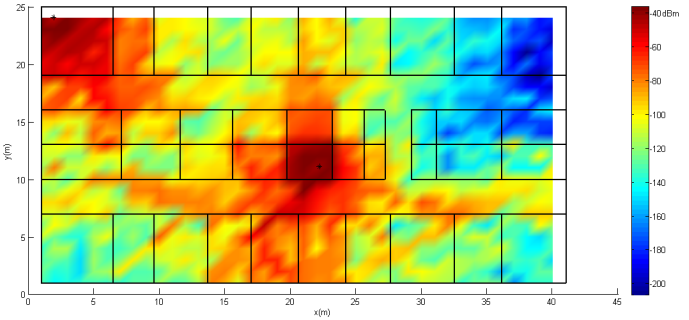
Figura 27 (a), (b), (c) e (d) - mostra os mapeamentos de potências recebidas, de acordo com a Tabela 6. Primeiro conjunto de valores gerados pela RTQ3D e 1.000 neurônios na entrada da RNA de treinamento.



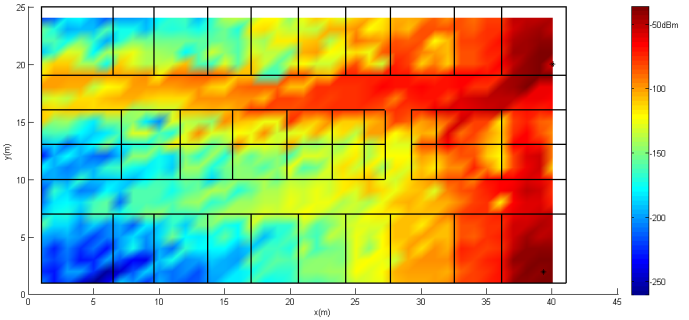
(a)



(b)



(c)



(d)

Outro aspecto importante que foi levado em consideração na construção do estudo de caso em questão foi o erro utilizado no treinamento da RNA. Foram utilizados quatro valores de erros, mostrados nas Tabelas 3, 4, 5 e 6. A utilização de valores de erros

diferentes teve como objetivo evitar a superespecialização da rede neural, o que causaria um aprendizado eficiente somente para os valores de treinamento. Observando-se os resultados do mapeamento das potências recebidas, que se encontram mostrados nas figuras 18 à 27, o resultado com 3.000 épocas e o erro de treinamento da ordem de 10^{-8} produzem o melhor par de posições das antenas encontradas, (figura 18(c)). Vemos que a distribuição do campo mostra uma eficiência satisfatória em função dos parâmetros de análise estabelecidos por esse estudo. Entretanto os resultados sugerem que melhores resultados, com campos mais uniformes, podem ser obtidos usando-se mais antenas de transmissão no cenário. Entretanto isso certamente impactaria no custo computacional, que foi foco deste estudo.

Quanto aos resultados com o algoritmo PSO assistido pela RNA, utilizando-se 4.500 épocas (Tabela 4, figuras 22 à 25), 1.500 (Tabela 5, figura 26) e 6.000 (Tabela 6, figura 27), pode-se observar que não houve resultado significativo no que se refere a uma melhora no mapeamento de potência distribuída e tampouco no tempo computacional, pois o custo computacional para 4.500 e 6.000 épocas aumentou e a eficiência não, e no caso das 1.500 épocas o custo computacional diminui juntamente com a eficiência do mapeamento.

5.4 CONSIDERAÇÕES FINAIS DO CAPÍTULO

O processo em que a *RTQ3D* foi inserida corresponde ao seguinte mecanismo: o traçado de raios é rodado uma vez para produzir o conjunto de treinamento, que vai ser utilizado no aprendizado da rede RMLP (Rede Neural Artificial *Perceptron* Multicamadas com algoritmo de aprendizado *backpropagation* Real). Em seguida, a rede neural treinada vai assistir o algoritmo evolutivo PSO, em que o foco dessa otimização multibjetivo foi determinar as posições ótimas das duas antenas, figura 17.

O traçado de raios quase 3D pode demorar até 72 horas para obter os valores de treinamento da rede neural. A rede neural demora em torno de 24 horas para treinar.

Vale ressaltar que esse tempo de treinamento da rede vai depender, também, da quantidade de partículas que vão fazer parte da camada de entrada e do número de neurônios da camada oculta. A partir da Equação (5.1), seção 5.1, temos um valor de referência para o número de neurônios da camada oculta, entretanto, segundo Travessa (2006) e Pereira (2003), esse valor pode ser alterado para que uma maior eficiência no treinamento possa ser alcançada. Tanto o *RTQ3D* quanto a

rede neural só precisam ser rodados uma só vez.

O tempo do algoritmo PSO assistido pelo metamodelo RNA no processo de otimização da posição das antenas chega a um máximo de 72 min.

Através das Tabelas 3, 4, 5 e 6, pode-se observar os resultados do posicionamento das antenas em função do número de épocas utilizadas. Para 1.500 épocas, o tempo de otimização, com 100 partículas iniciais, dura em torno de 20 min. Para 3.000 épocas, nas mesmas condições, o tempo médio é de 36 min. Para 4.500 épocas, o tempo médio é de 50 min. Para 6.000 épocas, o tempo é de aproximadamente 72 min. Logo, o tempo médio necessário para otimização com 3.000 épocas, que foi o melhor resultado obtido, fica em torno de 36 min.

Com esse capítulo, pode-se observar os resultados relativos à proposta de tese, sugerindo que a metodologia de otimização está de acordo com os objetivos pretendidos.

No capítulo 6, foram feitas as considerações finais relacionadas aos objetivos traçados pela tese e aos resultados obtidos.

6 DISCUSSÕES E CONCLUSÕES

Neste capítulo são abordados os aspectos relevantes da construção deste trabalho, dos resultados obtidos tendo em vista os objetivos traçados e a delimitação do problema. São também feitas sugestões para trabalhos futuros, com o intuito de ampliar a análise proposta ou usá-la como ponto de partida para linhas diferentes de estudo.

6.1 JUSTIFICATIVA PARA O DESENVOLVIMENTO DO TRABALHO

A principal justificativa para a proposição deste trabalho foi que os projetos envolvendo dispositivos eletromagnéticos precisam de processos cada vez mais voltados para produção de um resultado ótimo, ou seja, precisam atender à disponibilidade dos recursos técnicos e econômicos empregados.

Outro aspecto importante observado foi que o algoritmo PSO e os metamodelos, tais como as redes neurais, vêm sendo largamente utilizados por um número cada vez maior de pesquisadores, visando à otimização de projetos eletromagnéticos. Como dito anteriormente, o uso de metamodelos permite avaliações rápidas e precisas das funções objetivo, tornando eficiente o processo de otimização.

Segundo Prado e Saramago (2005), observou-se que a otimização PSO trabalha com um tamanho de população bastante reduzido, de modo que seu esforço computacional é menor do que o dos algoritmos genéticos. Esse fato incentiva pesquisas futuras de aplicação dessa técnica em problemas complexos que requerem muitas avaliações da função objetivo.

Segundo Rahmat-Samii e Jin (2007), o enorme interesse na técnica PSO é evidente devido ao grande leque de problemas em eletromagnetismo que podem ser resolvidos por esse algoritmo evolutivo. O comportamento do enxame se encaixa no processo de otimização e engloba diferentes tipos de otimizações com características real, binária e híbrida. O PSO pode ser facilmente aplicado aos diferentes problemas sem alterações significativas para o núcleo do otimizador.

Pôde-se concluir, a partir dos estudos analisados, que o algoritmo PSO além de ser simples do ponto de vista da implementação e compreensão dos seus mecanismos, sugere uma considerável robustez e

economia no tempo computacional, que foi o foco principal desse trabalho.

6.2 ANÁLISE DOS RESULTADOS

O objetivo principal desse trabalho de doutorado foi desenvolver seus estudos em cima da otimização multiobjetivo para problemas eletromagnéticos usando o algoritmo PSO (*Particle Swarm Optimization*) com a aplicação de metamodelos baseados em RNAs (Redes Neurais Artificiais), com seu foco na redução do custo computacional. A rede usada como metamodelo foi a Rede Neural Artificial MLP (*Perceptron* multicamadas) com algoritmo de aprendizado *backpropagation* real.

Foram tratadas, como um dos focos principais, a importância e a relevância da utilização de metamodelos com objetivo de diminuir o custo computacional, bastante elevado quando os cálculos são feitos durante o processo de seleção da população ótima de indivíduos.

Para isso, foi realizada uma revisão bibliográfica e feitos testes com as ferramentas propostas para o desenvolvimento desse trabalho, conforme relatado nos capítulos 1, 2, 3, 4 e 5.

O mecanismo que envolve a utilização do algoritmo evolucionário PSO assistido pela RNA com o objetivo de diminuir o custo computacional decorrente da utilização de um algoritmo evolutivo foi o seguinte: a técnica de Traçado de Raios *"quase 3D"*, desenvolvida por Grubisic (2012), produziu o conjunto de treinamento para a RNA, que após o aprendizado, foi utilizada para auxiliar o algoritmo evolucionário PSO. O uso da RNA como metamodelo teve como objetivo reduzir o elevado custo computacional existente nesse tipo de otimização, que envolve projetos eletromagnéticos. Esse caso específico envolveu o cálculo de campos eletromagnéticos e posicionamento de antenas num ambiente interno. A figura 17, capítulo 5, mostra o fluxo de trabalho do processo de otimização. No estudo, tem-se 160 receptores e 2 transmissores a serem posicionados em localização ideal, para produzir o resultado ótimo de posicionamento das antenas em função de um limiar mínimo de recepção (-110 dBm).

Na tarefa de melhorar o rendimento da RNA, foram feitas alterações nos parâmetros da rede, no sentido de evitar sua superespecialização. O primeiro parâmetro a ser alterado foi o momento (conforme seção 4.4, capítulo 4), e em seguida o número de neurônios da camada oculta, que foi calculado pela Equação 5.1 da seção 5.1. Com efeito, segundo Soviersoski (2009), o valor fornecido pela equação

admite alterações na busca do treinamento mais eficaz. Por fim, mexeu-se no coeficiente de aprendizado da rede. A cada alteração de parâmetro, novos treinamentos foram realizados e os resultados aplicados ao mesmo problema estudado.

A alteração dos parâmetros momento e coeficiente de aprendizado não teve impacto significativo no treinamento da rede. Na verdade, os valores utilizados foram os melhores e únicos que permitiram que a rede chegasse ao final do treinamento. Tais valores foram especificados na seção 5.1. Com os testes feitos com a quantidade de neurônios na camada oculta calculada a partir da equação 5.1, seção 5.1, concluiu-se que o valor calculado teve que ser alterado em função da complexidade do aprendizado realizado pela rede e o valor mais eficaz aplicado foi o de 500 neurônios na camada oculta.

Outro aspecto que melhorou o desempenho dos treinamentos foi a normalização dos valores dos campos eletromagnéticos, pois valores muito pequenos (da ordem de 10^{-8}) influenciam negativamente no treinamento da rede no que diz respeito à velocidade de treinamento, e com isso impacta no custo computacional do caso analisado.

Foram feitas análises em relação ao valor utilizado para o erro no treinamento da rede RMLP, pois o erro não deve levar a rede à superespecialização. Foram realizados diversos treinamentos, conforme Tabelas 3 a 6. Tais análises levaram a conclusão que os melhores resultados de treinamento da RNA, que consequentemente refletem na otimização do PSO assistido pela RNA, foram para os erros de treinamento da ordem de 10^{-8}.

Através dos gráficos dos mapeamentos de potências recebidas, mostrados nas figuras 18 a 21 (gerados após a otimização pelo algoritmo PSO assistido pela RNA, com 3.000 épocas, para os quarto erros discriminados nas Tabelas 3 e 4, com 1000 e 1500 neurônios de entrada da RNA e o primeiro e segundo conjunto de valores *fitness* gerados pelo *RTQ3D*), verificou-se que os melhores resultados para os mapeamentos ocorreram, independentemente do conjunto gerado pelo traçado de raios, quando o treinamento da rede foi realizado com um erro da ordem de 10^{-8}. O mesmo ocorreu para os experimentos realizados e sumarizados pelas Tabelas 5 e 6 e mostrados através dos gráficos das Figuras 22 a 25, com a diferença de a PSO estar rodando com 4.500 épocas.

Com relação às Figuras 26 e 27, que mostram os mapeamentos de potências recebidas (gerados após a otimização pelo algoritmo PSO assistido pela RNA para 1.500 e 6.000 épocas, para os quarto erros discriminados nas Tabelas 5 e 6, com 1000 neurônios de entrada na RNA e o primeiro conjunto de valores *fitness* gerados pelo *RTQ3D*),

optou-se por não demonstrar todos os testes, visto que o comportamento repetiu-se e apresentou-se menos eficaz que os anteriores.

O otimizador proposto, foi implementado usando o programa MATLAB versão 7.10.0.499. O processo de otimização foi realizado através de um computador MacBook Pro Retina com processador Intel Core i5 de 2,6 GHz, turbo Boost de 3,1GHz e 8 GB de memória RAM 1600 MHz DDR3, com 256 GB de armazenamento *flash*. Na otimização de um problema eletromagnético, o custo computacional pode ser muito grande, uma vez que é necessário realizar uma simulação numérica para cada avaliação da função de fitness. Nesse caso, o uso de uma RNA como metamodelo permite obter resultados satisfatórios com uma redução do custo computacional. O estudo desenvolvido neste trabalho baseia-se em Grubisic (2013), em que a associação de RTQ3D-indoor com otimizadores PSO foi aplicada em um problema típico de telecomunicações: um sistema de rede de área local sem fio (WLAN) em um cenário interno que deve ser coberto com dois *access points* (AP) 802.11b / g / n (2.4 GHz). No entanto, o estudo desenvolvido por Grubisic (2013) não leva em consideração o custo computacional, que é bastante alto quando um metamodelo não é usado. Como exemplo, o tempo médio necessário para otimização, levando-se em consideração o fluxo total do processo de otimização proposto, figura 5, capítulo 1, usando-se o a PSO assistida pelo metamodelo RNA com 3.000 épocas foi cerca de 24 horas e 36 min. Sem usar o metamodelo, o PSO leva cerca de 72 horas para produzir resultados relativamente melhores. Ou seja, tem-se em torno de 65,3% de redução no custo computacional.

Para avaliar a eficácia do método proposto, ele foi aplicado em uma função de teste com um máximo global. Além disso, foi também aplicado em um problema eletromagnético (baseado no melhor posicionamento de antenas em sistemas sem fio). A resposta do PSO assistido pela RNA para um problema eletromagnético real pôde ser considerada satisfatória. Todas as observações sugeriram que uma melhoria na RNA, treinamentos adicionais ou aumento no número de antenas (*access points*) levaria a melhores resultados. No entanto, isso aumentaria o tempo computacional.

Como conclusão, verificou-se que o melhor mapeamento de potências recebidas, com um menor custo computacional, também ocorreu para a PSO assistida para RNA rodando com 3.000 épocas e com um erro de treinamento da RNA de 10^{-8} (Figura 18(c)).

6.2.1 Publicações do trabalho

A proposta do trabalho preliminar que utiliza um otimizador evolucionário PSO assitido pela RNA, para auxiliar a técnica de traçado de raios *"quasi-3D"*, com objetivo de procurar o melhor posicionamento de antenas em sinais sem fio em ambiente fechado, com menor custo computacional, foi validada através da submissão e aceite para publicação na conferência COMPUMAG 2013.

A publicação COMPUMAG 2013 foi uma apresentação em pôster (Anexo A). Outra publicação foi a do Artigo estendido no MOMAG 2014, com apresentação oral (Anexo B).

Por fim, o essencial da tese foi publicada, em forma de artigo, na revista JMOe (*Journal of Microwaves, Optoelectronics and Electromagnetic Applications*), em dezembro de 2016 (Anexo C).

6.3 SUGESTÕES PARA TRABALHOS FUTUROS

Como uma das opções para trabalhos futuros está a possibilidade de experimentar as redes Neurais Artificiais complexas, com o algoritmo *backpropagation* complexo (Nitta, 1997), que está baseada em fatos tais como: (a) a média de aprendizado do perceptron multicamadas (MLP) usando o algoritmo *backpropagation* complexo é superior à velocidade média de aprendizado da rede MLP com algoritmo *backpropagation* real; (b) o número de pesos e limiares necessários para MLP com algoritmo *backpropagation* complexo fica em torno da metade daqueles necessários na MLP com *backpropagation* real; (c) alguns valores matemáticos reais são mais bem entendidos quando considerados no plano complexo, pois se considera um maior número de informações, as quais não seriam levadas em conta no plano real, (Hirose, 2003).

Outra proposição para trabalhos futuros seria a utilização de mais de duas antenas distribuídas pelo espaço considerado, o que em tese melhoraria o mapeamento, mas por outro lado deve levar ao aumento do custo computacional.

6.4 CONSIDERAÇÕES FINAIS DO CAPÍTULO

Com este capítulo, pudemos estabelecer a análise dos resultados obtidos e chegamos ao melhor resultado dentro das limitações da proposta, que teve seu foco principal na redução do custo computacional, que pode ser bem significativo em projetos que envolvem de otimização multiobjetivo feita por algoritmos evolucionários.

REFERÊNCIAS

BORGES, G. S. Desenvolvimento de Ambientes Computacionais para Análise de Dispositivos e Sistemas para Redes de Comunicações Ópticas e Móveis. Universidade Federal do Pará. Pará, p.54. 2006.

BULL, L. "On Model-Based Evolutionary Computation". *Soft Computing 3* , pp. 76-82 Springer-Verlag, 1999.

DEB, K. "Multi-Objective Evolutionary Algorithms: Introducing Bias Among Pareto-Optimal Solutions". Kanpur Genetic Algorithms Laboratory Report nº 99002, India,1999.

DEB, K. "Multi-Objective Optimization using Evolutionary Algorithms". John Wiley & Sons, Inglaterra, 2001.

EBERHART R. C.; SHI Y. "Particle Swarm Optimization: Developments, Applications and Resources". Congress on Evolutionary Computation, 2001, vol. 1, pp. 81-86, 2001.

EMMERICH M.; GIOTIS A.; OZDEMIR M. "Metamodel–Assisted Evolution Strategies". *In: Parallel Problem Solving from Nature - PPSN VII*, Springer-Verlag. 361-370, 2002.

EMMERICH, M. T. M.; VARCOL, C. M. "Metamodel–Assisted Evolution Strategies in Electromagnetic Compatibility Design". *EUROGEN*. 1-12, 2005.

FREEMAN, J. A.; SKAPURA, D. M. "Redes Neuronales Algoritmos, Aplicaciones y Técnicas de Programación". Editora: Addison-Wesley, 1993.

GRUBISIC S.; CARPES Jr W. P.; BASTOS J. P. A. "An Efficient indoor ray- tracing propagation model with a quasi-3D approach". IEEE CEFC 2012, November 2012, Oita/Japan.

GRUBISIC S.; CARPES Jr W. P.; BASTOS J. P. A.; SANTOS G. "Association of a PSO optimizer with a *quasi*-3D ray-tracing propagation model for mono and multi-criterion antenna positioning in indoor environments". IEEE CEFC 2012, November 2012, Oita/Japan.

GRUBISIC S.; CARPES Jr W. P.; BASTOS J. P. A.; SANTOS G. "Association of a PSO optimizer with a *quasi*-3D ray-tracing propagation model for mono and multi-criterion antenna positioning in indoor environments". IEEE Transactions on Magnetics, Vol. 49, No. 5, May 2013.

GRUBISIC, S. Técnica de Traçado de Raios Associada a Metaheurísticas para Otimização do Posicionamento de Antenas em Ambientes Interiores. 2012. Tese (Doutorado em Engenharia Elétrica) – GRUCAD, Centro Tecnológico, Universidade Federal de Santa Catarina, Florianópolis, 2012.

HAHMAT-SAMII, Y.; JIN, N. "Particle Swarm Optimization (PSO) in Engineering Electromagnetics: A Nature-Inspired Evolutionary Algorithm". IEEE, 2007. Inc, United States of America, pp. 87-88.

HAYKIN, S. Redes Neurais Princípios e Prática. [S.l.]: Bookman.1999.

HIROSE, A. "Complex-Valued Neural Networks – Theories and Aplications". Published by World Scientific Publishing Co. Pte Ltd, USA, Chapter 2, pp.7-28, 2003.

JIN, Y. "A Comprehensive Survey of Fitness Approximation in Evolutionary Computation". *Soft Computing Journal*, vol. 9, issue 1, pp. 3-12, 2005.

KENNEDY, J. and EBERHART, R. C. "Particle Swarm Optimization", Proceedings of the 1995 IEEE Internacional Conference on Nerual Networks, Perth, Australia, 1995, pp. 1942-1948.

LEBENSZTJAN, L.; MARRETTO, C. A. R.; COSTA, M. C.; COULOMB, J. L. "Kriging: a useful tool for eletromagnetic device optimization". IEEE TRANSACTIONS ON MAGNETICS, vol.40, N^{o}.2, p. 1196-99, March 2004.

MENEZES, N. G. Síntese Topológica Evolucionária e Otimização Numérica de Parâmetros em Estrutura CMOS. (Mestrado). Engenharia Elétrica, COPPE/UFRJ, Rio de Janeiro, 2007. 59 p.

MOHAMMED, O. A.; PARK, D. C.; ÜLER, F. G.; ZIGIANG, C. "Design optimization of electromgnetic devices using artifitial neural

networks". IEEE TRANSACTIONS ON MAGNETICS, vol.28, N°.5, p. 2805-07, september 1992.

NASCIMENTO, C. L.; YONEYAMA, T. Inteligência Artificial em Controle e Automação, FAPESP Editora Edgard Blücher, 2000.

NITTA, T. "An Extension of the Back-Propagation Algorithm to Complex Numbers". *Neural Networks*, vol 10, no 8, pp. 1391-1415, Elsevier Science Ltda, 1997.

PEREIRA, M. C. V. Avaliação de Técnicas de Pré-processamento de Sinais do EEG para Detecção de Eventos Epileptogênicos Utilizando Redes Neurais Artificiais. 2003. Tese (Doutorado em Engenharia Elétrica) – Engenharia Biomédica, Centro Tecnológico, Universidade Federal de Santa Catarina, Florianópolis, 2003.

POMEROY, P. "An Introduction to Particle Swarm Optimization", http://www.adaptiveview.com, [15 Setembro de 2003].

PARKA J.; KIM Kyoung-Yun. "Meta-modeling using generalized regression neural network and particle swarm optimization". ELSEVIER, APLIED SOFT COMPUTING, pp 354, 2017

PRADO, J. R. SARAMAGO, S. F. P. Otimização por Colônia de Partículas. Trabalho dsenvolvido na Faculdade de Matemática - FAMAT, UNIVERSIDADE FEDERAL DE UBERLÂNIA – UFU, 2005.

REGIS, R. G. "Particle swarm with radial basis function surrogates for expensive black-box optimization". ELSEVIER. JOURNAL OF COMPUTATIONAL SCIENCE. Pp. 12, 2014.

RODRIGUES, M. A. B. Sistemas para Detecção e Classificação Automática de Apnéias do Sono a partir de Registros Polissonográficos. Exame de Qualificação, IEB, UFSC, 1999.

ROY, S; DAS, K. A.; BOSE, P. K.; BANERGEE R. "ANN metamodel assisted Particle Swarm Optimization of the performance-emission trade-off characteristics of a single cylinder CRDI engine under CNG dual-fuel operation". ELSEVIER. Journal of Natural Gas Science and Engineering. Volume 21, pp. 1156, 2014.

SCHWEFEL H.P.; TAYLOR E. L. “Evolution and Optimum Seeking”. John Wiley & Sons. Inc, United States of America, pp. 87-88, 1994.

SOVIERSOSKI, M. A. Avaliação de Descritores Morfológicos na Identificação de Eventos Epileptiformes. 2009. Tese (Doutorado em Engenharia Elétrica) – Engenharia Biomédica, Centro Tecnológico, Universidade Federal de Santa Catarina, Florianópolis, 2009.

SUN C., ZENG J., PAN J., XUE S., and JIN Y. “A new fitness estimation strategy for particle swarm optimization”. Information Sciences, vol. 221, pp. 355–370, 2013.

TAGLIARINI, G. A. “Optimization using neural networks. IEEE TRANSACTIONS ON COMPUTERS”. vol.40, N^{o}.12, p. 1347-58, December 1991.

THONGVIGGITMANEE, T.; MAY, G. S. “Optimization of nanocomposite integral capacitor fabrication using neural networks and genetic algorithms”. SEMI Technology symposium: International electronics manufacturing technology (IEMT) symposium. 2002.

TRAVESSA, S. S. Análise das Redes Neurais Complexas na Detecção de Espículas e Piscadas em Sinais de EEG, 2006. Dissertação (Mestrado em Engenharia Elétrica) – Engenharia Biomédica, Centro Tecnológico, Universidade Federal de Santa Catarina, Florianópolis, 2006.

VENTER, G.; SOBIESZCZANSKI-SOBIESKI, J. “Particle Swarm Optimization”. Proceedings of the 43rd AIAA/ASME/ASCE/AHS/ASC Strutures, Structural Dynamics, and Materials Conference, Denver, CO, Vol. AIAA-2002-1235, April 22-25 2002.

WILLMES, L.; BÄCK, T.; JIN, Y.; SENDHOFF, B. “Comparing neural network and kriging for fitness approximation in evolutionary optimization”. IEEE, pp. 670, 2003.

ZHAO, H., GUO, S. “Annual Energy Consumption Forecasting Based on PSOCA-GRNN Model”. Hindawi Publishing Corporation, Abstract and Applied Analysis, Volume 2014, Article ID 217630, 11 pages http://dx.doi.org/10.1155/2014/217630, 2014.

ANEXO A – Artigo resumido COMPUMAG - 2013

Use of an Artificial Neural Network-based metamodel in the optimization by Particle Swarm Optimization method

S. S. Travessa; W. P. Carpes Jr; M. A. Nunes Filho
GRUCAD, Universidade Federal de Santa Catarina
C. P. 476, 88040-900, Florianópolis, SC, Brazil
sheila@grucad.ufsc.br, carpes@grucad.ufsc.br, marcelo@grucad.ufsc.br

***Abstract*— This study aims to develop efficient techniques for optimization of electromagnetic problems. We use the PSO algorithm (Particle Swarm Optimization) associated with a metamodel based on an ANN (Artificial Neural Network). Specifically, we use the MLP (Multilayer Perceptron) with the backpropagation algorithm. The ANN will be used to assist the technique of "*quasi* 3D" ray-tracing in order to reduce the high computational cost of this technique in PSO optimization.**

***Index Terms*— Artificial Neural Networks, Multilayer Perceptron, electromagnetic fields, Particle Swarm Optimization, metamodeling.**

I. Introduction

An important application of metamodels is in the optimization of electromagnetic problems. In fact, given that the modeling of such problems generally have a high computational cost, the optimization of electromagnetic devices often require certain procedures for the replacement of objective functions, in order to evaluate them in an efficient way. The replacement function allows obtaining results with good precision, but with a much lower computational cost [1].

Several models can be used to obtain approximations of objective functions with good accuracy. The kriging model [1] and the feed-forward neural network, for example, are robust and offer savings in computational time. Studies have shown that these two metamodels have similar performance [2]. Therefore, we chose to use the Artificial Neural Networks (ANN), which has been widely used in several applications due to ease of implementation and effectiveness.

It is important to stress that the evolutionary algorithms associated with metamodels are becoming extensively used by a growing number of researchers in design optimization [3]. Here, we show the validity of the PSO (Particle Swarm Optimization) algorithm in conjunction with an ANN-based metamodel to be used in the optimization of electromagnetic problems.

II. Use of the PSO and ANN in Optimization

In order to verify the effectiveness of the PSO algorithm assisted by an ANN, this approach was initially used in the optimization of a test function. Specifically, the goal is to find the global maximum of the function Peaks, given by the equation (1):

$$F(x,y) = 3(1-x)^2 e^{-x^2-(y+1)^2} - \left(2x - 10x^3 - 10y^5\right)e^{-x^2-y^2} - \frac{1}{3}e^{-(x+1)^2-y^2} \quad (1)$$

It also implemented a case study presented in [4], corresponding to the propagation of an electromagnetic wave in an indoor environment, of 40 m × 28 m, whose ceiling height from the ground is 3 m. The goal is to find the best placement of two antennas in order to have the best environment coverage. A relevant aspect is that this case study refers to a problem with continuous search space, which is permitted placement of antennas in any position within the environment and not just at specific points. It was defined a grid of 160 reception points uniformly distributed within the environment. The antennas at all the reception points were considered to be 1.5 m high from the ground.

The goal of this implementation is to test the effectiveness of the PSO optimizer using an ANN (network MLP with backpropagation algorithm) as a metamodel.

Another important aspect in defining the topology of neural network is the number of neurons in the hidden layer, due to its influence on both the speed and the effectiveness of the learning network, which was defined by the geometric mean between the input and output neurons [5].

The neural network was used in order to replace the calculations made by the quasi 3D technique, [4], [6].

The neural network through supervised learning characteristic of Multilayer Perceptron with backpropagation algorithm optimizes weight values that are saved in a file. With these weigh values, the ANN calculates the field at each reception point. These field values are used by the PSO optimizer, which will obtain the optimal positions of the antennas and the corresponding received power mapping.

III. Results

The optimization was done both directly using the function given in equation (1) and using an ANN-based metamodel to represent it. In both cases, the PSO was used to obtain the global maximum location, represented by particle "×" in Fig. 1. The algorithm was initialized with a random set of particles and the process took place until the maximum point was located, at the center of the red contours.

In the first case, we consider an optimization problem with analytical solution. Figure 1 shows the function that was simulated using an ANN-based metamodel in the optimization by PSO method. Figure 2(a) shows the contour plot of the

results obtained using directly equation (1) while figure 2(b) shows the results obtained using the ANN simulating the function given in (1).

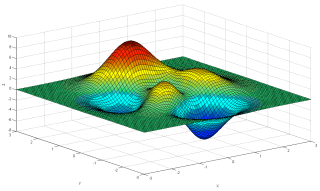
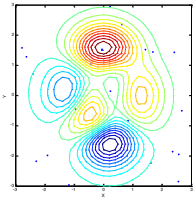


Fig. 1. The analytical function considered.



(a)

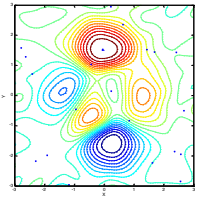


Fig. 2(a). The contour plot of the actual function; (b) The contour plot of the simulated function, with the located peak.

The ANN was trained until a root mean square error of 0.0004 was obtained, requiring 27,253 epochs to get this error value. The mean square error solution of PSO assisted by the ANN was equal to 0.00569. Analyzing the results, we can conclude that the use of a metamodel to replace the original fitness function allows obtaining good results.

In the second example, we analyzed a real problem: the optimization of two antennas placement in an indoor environment. Figure 3 shows the result of PSO assisted by ANN, aiming to replace the ray tracing "*quasi*-3D" simulation, which has high computational coast. The graph shows the placement of two antennas in order to supply the needs of 160 receivers as described in section II. The PSO assisted by RNA was initialized with a set of 50 random particles. The optimization was set to happen in 3000 epochs, under the conditions presented.

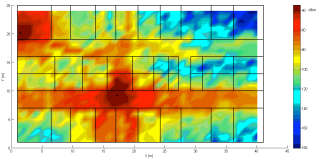


Fig. 3. Received power mapping in the indoor environment.

IV. CONCLUSIONS

In the optimization of an electromagnetic problem, the computational cost can be very large, since it is necessary to carry out a numerical simulation for each evaluation of the fitness function. In this case, the use of an ANN as metamodel permits to obtain satisfactory results with a significant reduction of the computational cost

In this paper, we proposed a PSO optimizer assisted by an ANN. In order to assess its effectiveness, it was applied in a test function with a global maximum. Also, it was applied this approach in an electromagnetic problem (namely, the best antenna positioning in wireless systems). The response of the ANN-assisted PSO for a real electromagnetic problem can be considered very good, but not necessarily the best, since with more training, time can get closer to the best response. The proposition for the extended paper is to reach the optimal time training to achieve a satisfactory response. New results as well as more details and discussions will also be presented in the full paper.

REFERENCES

[1] L. Lebensztjan, C. A. R. Marretto, M. C. Costa, J. L. Coulomb, "Kriging: a useful tool for eletromagnetic device optimization," vol.40, N°.2. IEEE Transactions on Magnetics, march 2004, p. 1196-99.

[2] O. A. Mohammed, D. C. Park, F. G. Üle; C. Zigiang, "Design optimization of electromgnetic devices using artifitial neural networks," vol.28, N°.5 IEEE Transactions on Magnetics, September 1992, p. 2805-07.

[3] Rahmat-samii Y., N. JIN, "Particle Swarm Optimization (PSO) in Engineering Electromagnetics: A Nature-Inspired Evolutionary Algorithm," IEEE, 2007. Ine, United States of America, 1994 pp. 87-88.

[4] S. Grubisic, W. P. Carpes Jr, J. P. A. Bastos, G. Santos, "Association of a PSO optimizer with a *quasi*-3D ray-tracing propagation model for mono and multi-criterion antenna positioning in indoor environments," IEEE CEFC 2012, November 2012, Oita/Japan (accepted).

[5] J. M. Barreto, "Neural Networks: Mathematical and Computational Aspects," *Annals of the Brazilian Society of Applied and Computational Mathematics (BSACM)*, 1996.

[6] S. Grubisic, W. P. Carpes Jr, J. P. A. Bastos, "An Efficient indoor ray-tracing propagation model with a quasi-3D approach ," IEEE CEFC 2012, November 2012, Oita/Japan (accepted).

ANEXO B – Artigo MOMAG – 2014

Use of an Artificial Neural Network-based metamodel in the optimization by Particle Swarm Optimization method

Sheila Santisi Travessa, Walter Pereira Carpes Jr.
Federal University of Santa Catarina, Electric Engineering Department
Grupo de Concepção e Análise de Dispositivos Eletromagnéticos (GRUCAD)
Florianópolis, Brazil
sheila.santisi@gmail.com, carpes@grucad.ufsc.br, walterpcjr@gmail.com

***Abstract*— This study aims to develop efficient techniques for optimization of electromagnetic problems. We use the PSO algorithm (Particle Swarm Optimization) associated with a metamodel based on an ANN (Artificial Neural Network). Specifically, we use the MLP (Multilayer Perceptron) with the backpropagation algorithm in order to evaluate objective functions in an efficient way. The ANN will be used to assist the technique of "quasi 3D" ray-tracing in order to reduce the high computational cost of this technique in PSO optimization.**

Keywords—Artificial Neural Networks, Multilayer Perceptron, electromagnetic fields, Particle Swarm Optimization, metamodeling.

I. INTRODUCTION

Optimization problems are characterized by situations in which one needs to maximize or minimize a numerical function of several variables, in a context where there may be restrictions.

The sophistication of computer resources developed in recent years has motivated a breakthrough in optimization techniques. In fact, optimization problems have become increasingly complex, increasing the associated computational cost. Classical optimization techniques are reliable and have applications in many different fields of engineering and other sciences. However, these techniques can present some numerical difficulties and problems related to robustness: the lack of continuity of the functions to be optimized or its constraints, nonconvex functions, multimodality, the existence of noise functions, the need to work with discrete values for variables, the existence of local minimum or maximum, etc. Thus, studies of heuristic methods with search-controlled randomized probabilistic criteria reappear as a strong trend in recent years, mainly due to the advancement of computational resources. However, a limiting factor of these methods is the need for a large number of function evaluations goal [1].

In the methods of natural optimization, the objective function is evaluated several times, since we deal with a great number of points (population) at a given iteration. This increases the computational cost associated with these methods. However, this high computational cost is the price to pay for preventing solutions from becoming trapped in local minima.

In general, the methods of natural optimization require more computational effort when compared to classical methods, but have some advantages such as ease of implementation, robustness and do not require continuity of the objective function associated with the problem [2].

An important application of metamodels is in the optimization of electromagnetic problems. In fact, given that the modeling of such problems generally has a high computational cost, the optimization of electromagnetic devices often requires certain procedures for the replacement of objective functions, in order to evaluate them in an efficient way. The replacement function allows obtaining results with good precision, but with a much lower computational cost [3].

Several models can be used to obtain approximations of objective functions with good accuracy. The kriging model [3] and the feed-forward neural network, for instance, are robust and offer savings in computational time. Studies have shown that these two metamodels have similar performance [4]. Therefore, we chose to use the Artificial Neural Networks (ANN), which have been widely used in several applications due to ease of implementation and effectiveness.

It is important to stress that the evolutionary algorithms associated with metamodels are becoming extensively used by a growing number of researchers in design optimization [5]. Here, we show the validity of the PSO (Particle Swarm Optimization) algorithm in conjunction with an ANN-based metamodel to be used in the optimization of electromagnetic problems.

II. PSO AND THE NNA-BASED METAMODEL

A. PSO – Particle Swarm Optimization

Introduced by Kennedy and Eberhart in 1995, the particle swarm optimization has emerged from experiments with algorithms modeled from the observation of social behavior of certain types of birds. The particles considered by the algorithm behave like birds looking for food or a place to make their nests, each one using both its own knowledge and the knowledge of the flock (or swarm) concerned to the best location for that. PSO is composed of particles represented by vectors defining the current speed of each particle and its location. The location of each particle is updated according to its current speed, its own knowledge and the knowledge

acquired by the whole flock. The PSO algorithm encompasses simple concepts and can be implemented in a few lines of code, requiring only simple mathematical operators.

The algorithm starts by generating a population of N particles that form the swarm and establish (randomly) their respective positions in the search space. It also sets the initial speed for each particle. Subsequently, at each iteration k, the algorithm updates the position $x^i(k)$ and the velocity $v^i(k)$ of each particle i. After the first evaluation of the fitness associated to a given particle, the algorithm keeps track of the coordinates p^i in the search space associated with its best solution. The overall best value (considering the whole population) and its location $p^s(k)$ are also stored during the procedure. With a simple scheme, these stored data are iteratively updated using random parameters and constants, changing the particles velocities in order to intelligently explore the search domain at each iteration [6].

The velocity vector of each particle is updated according to:

$$v^i(k+1) = w\, v^i(k) + c_1\, r_1 \left[p^i - x^i(k)\right] + c_2\, r_2 \left[p^s(k) - x^i(k)\right] \quad (1)$$

where w is the inertia weight; c_1 and c_2 are, respectively, the cognition and social coefficient; and r_1 and r_2 are random numbers uniformly distributed in [0,1]. Inertia weight w usually is chosen between 0.4 and 1.4: lower values speed up convergence while higher values encourage exploring the search space. Cognition coefficient c_1 limits the size of the step that the particle takes toward its individual best while social coefficient c_2 limits the size of the step that the particle takes toward the global best. Usually both assume values close to 2.

The position of each particle is updated according to:

$$x^i(k+1) = x^i(k) + v^i(k+1) \quad (2)$$

The iterations proceed until some stopping criterion is reached (e.g., maximum number of iterations or desired particle fitness).

B. Metamodel: ANN (Artificial Neural Network) with backpropagation algorithm

According to [7], a neural network is a massively parallel distributed processor consisting of simple processing units, which have the natural tendency for storing experiential knowledge and making it available for use. It resembles the brain in two respects:

- knowledge is acquired by the network from its environment through a learning process;

- connection strengths between neurons, known as synaptic weights, are used to store the acquired knowledge.

The procedure used to perform the learning process is called a learning algorithm, whose function is to modify the synaptic weights of the network in a systematic manner to achieve a desired design goal. We consider the multilayer perceptron (MLP), which consists in a network of simple neurons called perceptrons. The MLP is a hierarchical structure of several perceptrons with some advantages compared to single-layer networks and it is capable of learning a rich variety of nonlinear decision surfaces [7-8].

The so-called neuron's net input can be represented by the weighted sum of the inputs from other neurons and external inputs, according to (3).

$$net_i(t) = \sum_{j=1}^{p} w_{ji}\, y_j(t-1) + \sum_{l=1}^{m} w_{ki}\, I_k(t) + \theta_i \quad (3)$$

where $y_j(t-1)$ is the output of the j-th neuron in the previous processing cycle (t-1); I_k is the k-th external input; θ_i is the bias of the i-th neuron and w refers to the corresponding weight, shown in Fig. 1[9].

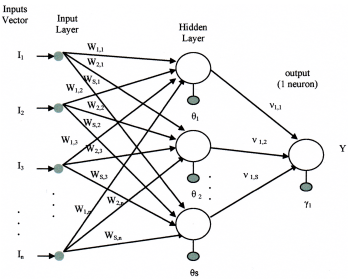


Fig. 1. Illustrative *"Grafo"* of The proposed network [9].

In order to model an electromagnetic problem, data from the electromagnetic simulation are used to train the ANN, i.e., to compute the weights of (3). Here, we chose to use the backpropagation (BP) algorithm for this training (supervised learning). Backpropagation is an abbreviation for "backward propagation of errors". In the BP algorithm, the network begins with a random set of weights. Input data is normalized to a range of ±1 and input vectors are presented to the network. Thus, the corresponding output is calculated using the initial weight matrix. Next, the ANN output is compared to the output of the electromagnetic simulation (using the same input data). The squared difference between the two output vectors determines the system error. The accumulated error for all of the input-output pairs is defined as the Euclidean distance in the weight space which the network attempts to minimize. Minimization is accomplished via a gradient descent approach, in which the weights are adjusted in the direction of decreasing error. When the error has been decreased below a pre-defined tolerance, training ceases. It is worth remarking the nonlinear functions can be represented by MLP with units that use nonlinear activation functions. Since the BP algorithm requires an activation function to be differentiable, we used a sigmoid (logistic function) [7]. Another important aspect in defining the topology of neural network is the number of neurons in the hidden layer, due to its influence on both the speed and the effectiveness of the learning network, which was defined by the geometric mean between the input and output neurons [8].

APÊNDICE A – Descrição

Textos elaborados pelo autor, a fim de completar a sua argumentação.

III. VALIDATION

In order to verify the effectiveness of proposed method, it was initially used in the optimization of a test function with analytical solution. Specifically, the goal is to find the global maximum of the Peaks function, given by (4) and shown in Figure 2.

$$F(x,y) = 3(1-x)^2 e^{-x^2-(y+1)^2} - \left(2x - 10x^3 - 10y^5\right) e^{-x^2-y^2} - \frac{1}{3} e^{-(x+1)^2-y^2} \quad (4)$$

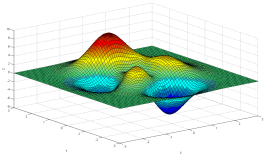
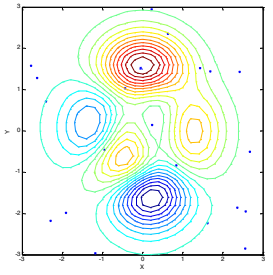


Fig. 2. The analytical function considered.

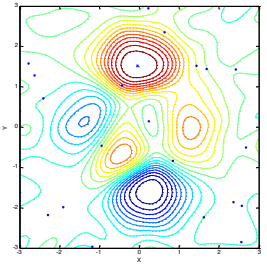
The goal of this implementation is to test the effectiveness of the PSO optimizer using an ANN (network MLP with backpropagation algorithm) as a metamodel.

The optimization was done both directly using the function given by (4) and using an ANN-based metamodel to represent it. Specifically, the PSO was used to obtain the global maximum location of Peaks function. The algorithm was initialized with a random set of particles and the process took place until the maximum point was located, at the center of the red contours.

Fig. 3(a) shows the contour plot of the results obtained using directly (4) while fig. 3(b) shows the results obtained using the ANN simulating the function given by (4). The global maximum is represented by "×" in fig. 3(a)



(a)



(b)

Fig. 3. (a). Contour plot of the actual function; (b) Contour plot of the simulated function, with the located peak.

The ANN was trained until a root mean square error of 0.0004 was obtained, requiring 27,253 epochs to get this error value. An epoch is a step in the training process of an ANN, i.e., each time the network is presented with a new input pattern. The mean square error solution of PSO assisted by the ANN was equal to 0.00569. From the results, we can conclude that the use of the proposed metamodel to replace the original fitness function allows obtaining good results.

IV. ELECTROMAGNETIC PROBLEM

The use of an electromagnetic prediction model represents an important design procedure in wireless systems. Here, we use a quasi-3D ray-tracing propagation model associated with a PSO optimizer to find optimal antenna placement in an indoor scenario [10-11].

Initially, a population is generated where each particle corresponds to the coordinates (x, y) of the transmitting antennas placements in a 40 m × 28 m scenario. It was defined a mesh of 160 receptors uniformly distributed within the environment. The antennas of all receivers were considered to be 1.5 m high from the ground. The method was applied to minimize the number of points where the field is below a minimum reception threshold (-110 dBm). This is equivalent to maximize the coverage area. The variable of the problem was assumed to be a vector containing the coordinates (x, y) of the transmitting antennas, allowing different locations of them on the horizontal plane of the scenario. The height of these transmitting antennas was assumed to be fixed (3 m), simulating vertically polarized quarter-wave monopole antennas fixed to the ceiling. Two antennas were considered in the problem. Therefore, the variable of the problem is a vector containing four elements, which represent coordinates of transmitting antenna positions on the horizontal plane. The frequency used was 2.4 GHz, and the total equivalent isotropically radiated power (EIRP) was 400 mW.

Firstly, the field values generated by ray-tracing are used as a guide in the learning process performed by the network, optimizing weight values, which are saved in a file. After training, the ANN calculates the field values at all reception points, which are used by the PSO optimizer, allowing a great reduction in computation time.

The PSO assisted by ANN was initialized with a set of 100 random particles. The PSO assisted by ANN optimizing was performed for three different numbers of iterations: 2,000, 3,000 and 6,000. Table 1 shows the best antenna locations found by the PSO for these three cases. We observe that the results are very sensitive to the number of iterations, which indicates that the ANN needs to be further improved. Fig. 4 shows the mapping of received powers considering the best pair of positions of the antennas found for 3,000 iterations. We see that the distribution of field is reasonable. Best results, with more uniform fields, can be obtained using more transmitting antennas in the scenario.

TABLE I.

Iterations	Results of the Optimization ANN-Assisted PSO	
	Position Antenna 1 (x_1(m), y_1(m))	***Position Antenna 2 (x_2(m), y_2(m))***
2,000	(25.6126,9.5614)	(34.0432,17.5768)
3,000	(20.1806,7.5598)	(29.1253,15.8245)
6,000	(34.4975,4.9075)	(32.0039,6.6523)

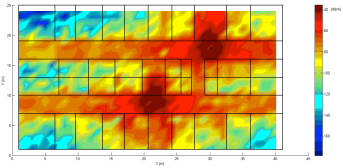


Fig. 4. Received power mapping in the indoor environment (3,000 iteractions).

V. CONCLUSIONS

In this paper, we proposed a PSO optimizer assisted by an ANN, The optimizer was implemented using the MATLAB program version 7.10.0.499. The optimization process was carried out through a Mac Mini computer With 1.8 GHz intel core i5 processor and 8GB of RAM memory 1600MHz DDR3. In the optimization of an electromagnetic problem, the computational cost can be very large, since it is necessary to carry out a numerical simulation for each evaluation of the fitness function. In this case, the use of an ANN as a metamodel permits to obtain satisfactory results with a reduction of the computational cost. In order to assess the effectiveness of the proposed method, it was applied in a test function with a global maximum. Also, it was applied in an electromagnetic problem (namely, the best antenna positioning in wireless systems). The response of the ANN-assisted PSO for a real electromagnetic problem can be considered satisfactory. All observations suggest that an improvement in the ANN and additional training will lead to the best results. However, this would increase the computational time.

REFERENCES

[1] H.P. Schwefel, E. L. Taylor, "Evolution and Optimum Seeking, *" John Wiley & Sons. Inc*, United States of America, pp. 87-88, 1994.

[2] G. Venter, and J. Sobieszczanski-sobieski, "Particle Swarm Optimization," *Proceedings of the 43rd AIAA/ASME/ASCE/AHS/ASC Srrutures, Structural Dynamics, and Materials Conference*, Denver, CO, Vol. AIAA-2002-1235, April 22-25 2002.

[3] L. Lebensztjan, C. A. R. Marretto, M. C. Costa, J. L. Coulomb, "Kriging: a useful tool for electromagnetic device optimization," vol.40, N°.2. IEEE Transactions on Magnetics, March 2004, p. 1196-99.

[4] O. A. Mohammed, D. C. Park, F. G. Üle; C. Zigiang, "Design optimization of electromagnetic devices using artificial neural networks," vol.28, N°.5 IEEE Transactions on Magnetics, September 1992, p. 2805-07.

[5] Rahmat-Samii Y., N. JIN, "Particle Swarm Optimization (PSO) in Engineering Electromagnetics: A Nature-Inspired Evolutionary Algorithm," IEEE, 2007. Inc, United States of America, 1994 pp. 87-88.

[6] R. C. Eberhart and Y. Shi, "Particle Swarm Optimization: Developments, Applications and Resources", *Congress on Evolutionary Computation, 2001*, vol. 1, pp. 81-86, 2001.

[7] Haykin, S., *Neural Networks: A Comprehensive Foundation*, Prentice Hall, 1999.

[8] J. M. Barreto, "Neural Networks: Mathematical and Computational Aspects," *Annals of the Brazilian Society of Applied and Computational Mathematics (BSACM)*, 1996.

[9] M. C. V. PEREIRA, "Evaluation of Techniques of Preprocessing EEG signals to epileptogenic Events Detection Using Artificial Neural Networks," 2003. Thesis (Ph.D. in Electrical Engineering) – Biomedical Engineering, Technology Center, Federal University of Santa Catarina, Florianópolis, 2003.

[10] S. Grubisic, W. P. Carpes Jr, J. P. A. Bastos, "An Efficient indoor ray-tracing propagation model with a quasi-3D approach ," IEEE CEFC 2012, November 2012, Oita/Japan.

[11] S. Grubisic, W. P. Carpes Jr, J. P. A. Bastos, G. Santos, "Association of a PSO optimizer with a *quasi*-3D ray-tracing propagation model for mono and multi-criterion antenna positioning in indoor environments," IEEE CEFC 2012, November 2012, Oita/Japan.

ANEXO C – Artgo publicado JMOe – 2016

Journal of Microwaves, Optoelectronics and Electromagnetic Applications, Vol. 15, No. 4, December 2016 418
DOI: http://dx.doi.org/10.1590/2179-10742016v15i4816

Use of an Artificial Neural Network-based Metamodel to Reduce the Computational Cost in a Ray-tracing Prediction Model

Sheila S. Travessa, Walter P. Carpes Jr.,
Federal University of Santa Catarina, Electric Engineering Department
Grupo de Concepção e Análise de Dispositivos Eletromagnéticos (GRUCAD)
Florianópolis, Brazil
sheila.santisi@gmail.com, walterpcjr@gmail.com

***Abstract*— The purpose of this article is based on analyzing the use of RTQ3D ("quasi-3D" ray tracing technique) to produce the value of the initial electromagnetic fields or fitness for a hundred and sixty receivers according to the possible positions of two antennas to be distributed in a closed environment. The problem variables consist of the values of the magnetic fields for one hundred and sixty receptors depending on the positions of the antennas to the base stations, which serve as input data for the algorithm to the RMLP (Artificial Neural Network, multilayer perceptron with Real backpropagation learning algorithm). The values of the magnetic fields associated with the positions of the antennas are the values to be learned by the network, the teacher of RMLP. This study aims to develop efficient techniques for optimization of electromagnetic problems. We use the PSO (Particle Swarm Optimization) algorithm associated with a metamodel based on an ANN (Artificial Neural Network). Specifically, we use the MLP (Multilayer Perceptron) with the backpropagation algorithm in order to evaluate objective functions in an efficient way. The ANN will be used to assist the technique of "quasi 3D" ray-tracing in order to reduce the high computational cost of this technique in PSO optimization.**

***Index Terms*—** Artificial Neural Networks, Multilayer Perceptron, electromagnetic fields, Particle Swarm Optimization, metamodeling.

I. INTRODUCTION

Optimization problems are characterized by situations in which one needs to maximize or minimize a numerical function of several variables, in a context where there may be restrictions.

The sophistication of computer resources developed in recent years has motivated a breakthrough in optimization techniques. In fact, optimization problems have become increasingly complex, increasing the associated computational cost. Classical optimization techniques are reliable and have applications in many different fields of engineering and other sciences. However, deterministic methods can present some numerical difficulties and problems related to robustness: the lack of continuity of the functions to be optimized or its constraints, nonconvex functions, multimodality, the existence of noise functions, the need to work with discrete values for variables, the existence of local minimum or maximum, etc. Thus, studies of heuristic methods with search-controlled randomized

Brazilian Microwave and Optoelectronics Society-SBMO *received 21 Apr 2016; for review 28 Apr 2016; accepted 20 Oct 2016*
Brazilian Society of Electromagnetism-SBMag *ISSN 2179-1074*

Journal of Microwaves, Optoelectronics and Electromagnetic Applications, Vol. 15, No. 4, December 2016 419
DOI: http://dx.doi.org/10.1590/2179-10742016v15i4816

probabilistic criteria reappear as a strong trend in recent years, mainly due to their robustness and to the advancement of computational resources. One of the main motivations for using stochastic optimization methods, such as PSO, is their ability to converge to the global optimum point. Indeed, most of the electromagnetics problems are multimodal, and using a deterministic method could lead to a local optimum. However, a limiting factor of stochastic methods is the need for a large number of objective function evaluations [1].

In the methods of natural optimization, the objective function is evaluated several times, since we deal with a great number of points (population) at a given iteration. This increases the computational cost associated with these methods. However, this high computational cost is the price to pay for preventing solutions from becoming trapped in local minima.

In general, the methods of natural optimization require more computational effort when compared to classical methods, but have some advantages such as ease of implementation, robustness and do not require continuity of the objective function associated with the problem [2].

An important application of metamodels is in the optimization of electromagnetic problems. In fact, given that the modeling of such problems generally has a high computational cost, the optimization of electromagnetic devices often requires certain procedures for the replacement of objective functions, in order to evaluate them in an efficient way. The replacement function allows obtaining results with good precision, but with a much lower computational cost [3].

Several models can be used to obtain approximations of objective functions with good accuracy. The most popular are the polynomial model, Kriging model, feedforward neural networks, including multilayer perceptrons, radial basis function networks and support vector machines. According to [4], the development of strategies for assisted metamodels are proposed as a new approach to basic simulation optimization in the field of electromagnetic project. In this application field, the estimation of objective functions are likely to have a significant computational cost, and techniques such as metamodeling are applied to accelerate the evolutionary optimization strategies. Some experiments done with the main metamodels mentioned in this study showed that better results were obtained with the Kriging model and artificial neural network.

The Kriging model [3] and the feed-forward neural network are robust and offer savings in computational time. Studies have shown that these two metamodels have similar performance [5]. Therefore, we chose to use the Artificial Neural Networks (ANN), which have been widely used in several applications due to ease of implementation and effectiveness.

It is important to stress that the evolutionary algorithms associated with metamodels are becoming extensively used by a growing number of researchers in design optimization [6]. Here, we show the validity of the PSO (Particle Swarm Optimization) algorithm in conjunction with an ANN-based metamodel to be used in the optimization of electromagnetic problems.

Brazilian Microwave and Optoelectronics Society-SBMO *received 21 Apr 2016; for review 28 Apr 2016; accepted 20 Oct 2016*
Brazilian Society of Electromagnetism-SBMag *ISSN 2179-1074*

Journal of Microwaves, Optoelectronics and Electromagnetic Applications, Vol. 15, No. 4, December 2016 420
DOI: http://dx.doi.org/10.1590/2179-10742016v15i4816

II. PSO, RTQ3D AND THE ANN-BASED METAMODEL

A. PSO – Particle Swarm Optimization

Introduced by Kennedy and Eberhart in 1995, the particle swarm optimization has emerged from experiments with algorithms modeled from the observation of social behavior of certain types of birds. The particles considered by the algorithm behave like birds looking for food or a place to make their nests, each one using both its own knowledge and the knowledge of the flock (or swarm) concerned to the best location for that. PSO is composed of particles represented by vectors defining the current speed of each particle and its location. The location of each particle is updated according to its current speed, its own knowledge and the knowledge acquired by the whole flock. The PSO algorithm encompasses simple concepts and can be implemented in a few lines of code, requiring only simple mathematical operators.

The algorithm starts by generating a population of N particles that form the swarm and establish (randomly) their respective positions in the search space. It also sets the initial speed for each particle. Subsequently, at each iteration k, the algorithm updates the position $x^i(k)$ and the velocity $v^i(k)$ of each particle i. After the first evaluation of the fitness associated to a given particle, the algorithm keeps track of the coordinates p^i in the search space associated with its best solution. The overall best value (considering the whole population) and its location $p^s(k)$ are also stored during the procedure. With a simple scheme, these stored data are iteratively updated using random parameters and constants, changing the particles velocities in order to intelligently explore the search domain at each iteration [7].

The velocity vector of each particle is updated according to (1):

$$v^i(k+1) = w\, v^i(k) + c_1\, r_1 \left[p^i - x^i(k)\right] + c_2\, r_2 \left[p^s(k) - x^i(k)\right] \tag{1}$$

Where w is the inertia weight; c_1 and c_2 are, respectively, the cognition and social coefficient; and r_1 and r_2 are random numbers uniformly distributed in [0,1]. Inertia weight w usually is chosen between 0.4 and 1.4: lower values speed up convergence while higher values encourage exploring the search space. The value used for w in this study was 0.9, which allows obtaining very good results for the analyzed problem. Cognition coefficient c_1 limits the size of the step that the particle takes toward its individual best while social coefficient c_2 limits the size of the step that the particle takes toward the global best. Usually both assume values close to 2. It would be possible to use different values of c_1, c_2 and w, including the idea of using variable parameters in time (e.g., decreasing w with the advancement of iterations). This could have been done to improve the convergence. However, we use typical fixed values for these parameters, according to the literature, which also yields satisfactory results.

The position of each particle is updated according to (2):

$$x^i(k+1) = x^i(k) + v^i(k+1) \tag{2}$$

The iterations proceed until some stopping criterion is reached (e.g., maximum number of iterations or desired particle fitness).

Brazilian Microwave and Optoelectronics Society-SBMO *received 21 Apr 2016; for review 28 Apr 2016; accepted 20 Oct 2016*
Brazilian Society of Electromagnetism-SBMag *ISSN 2179-1074*

Journal of Microwaves, Optoelectronics and Electromagnetic Applications, Vol. 15, No. 4, December 2016 421
DOI: http://dx.doi.org/10.1590/2179-10742016v15i4816

B. An RTQ3D-indoor

For the purpose presented in this paper, the RTQ3D-indoor algorithm is used for field computation and coverage evaluation, since the applications are in indoor environments. The RTQ3D-indoor [8] model uses 2D ray-tracing algorithm based on the Theory of Images to build a "tree of images" and to find the different reflected rays for a given scenario. The input scenario database is described in the horizontal plane, but adding the ceiling height from the floor and the transmitting (base stations) and receiving (user terminals) antennas heights [9].

According to the example of Fig. 1, the ray-tracing algorithm applied on quasi three-dimensional indoor environments (RTQ3D-indoor) converts a given path initially obtained from the RT 2D algorithm in five rays: i) the direct path (from the viewpoint of a vertical plane); ii) the ground reflected path; iii) the ceiling reflected path; iv) the path reflected on the ground and on the ceiling and; v) the path reflected firstly on the ceiling and then on the ground [9].

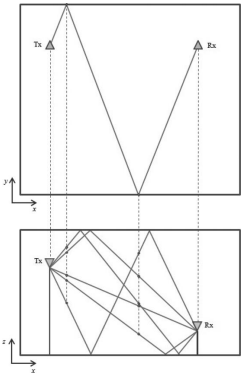


Fig. 1. Passage of a 2D path to five quasi-3D paths using the RTQ3D-indoor algorithm [8]: a) horizontal plane; b) vertical plane [9].

C. Metamodel: ANN (Artificial Neural Network) with backpropagation algorithm

According to [10], a neural network is a massively parallel distributed processor consisting of simple processing units, which have the natural tendency for storing experiential knowledge and making it available for use. It resembles the brain in two respects:

- knowledge is acquired by the network from its environment through a learning process;
- connection strengths between neurons, known as synaptic weights, are used to store the acquired knowledge.

The procedure used to perform the learning process is called a learning algorithm, whose function is

Brazilian Microwave and Optoelectronics Society-SBMO *received 21 Apr 2016; for review 28 Apr 2016; accepted 20 Oct 2016*
Brazilian Society of Electromagnetism-SBMag *ISSN 2179-1074*

Journal of Microwaves, Optoelectronics and Electromagnetic Applications, Vol. 15, No. 4, December 2016 422
DOI: http://dx.doi.org/10.1590/2179-10742016v15i4816

to modify the synaptic weights of the network in a systematic manner to achieve a desired design goal. We consider the multilayer perceptron (MLP), which consists in a network of simple neurons called perceptrons. The MLP is a hierarchical structure of several perceptrons with some advantages compared to single-layer networks and it is capable of learning a rich variety of nonlinear decision surfaces [10-11].

The so-called neuron's net input can be represented by the weighted sum of the inputs from other neurons and external inputs, according to (3).

$$net_i(t) = \sum_{j=1}^{p} w_{ji} y_j(t-1) + \sum_{l=1}^{m} w_{ki} I_k(t) + \theta_i \tag{3}$$

Where $y_j(t-1)$ is the output of the j-th neuron in the previous processing cycle (t-1); I_k is the k-th external input; θ_i is the bias of the i-th neuron and w refers to the corresponding weight, shown in Fig.2[10].

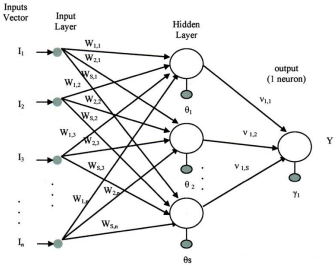


Fig.2. Illustrative *"Graph"* of The proposed network [10].

In order to model an electromagnetic problem, data from the electromagnetic simulation are used to train the ANN, i.e., to compute the weights of (3). Here, we chose to use the backpropagation (BP) algorithm for this training (supervised learning). Backpropagation is an abbreviation for "backward propagation of errors". In the BP algorithm, the network begins with a random set of weights. Input data is normalized to a range of ±1 and input vectors are presented to the network. Thus, the corresponding output is calculated using the initial weight matrix. Next, the ANN output is compared to the output of the electromagnetic simulation (using the same input data). The squared difference between the two output vectors determines the system error. The accumulated error for all of the input-output pairs is defined as the Euclidean distance in the weight space which the network attempts to minimize. Minimization is accomplished via a gradient descent approach, in which the weights are adjusted in the direction of decreasing error. When the error has been decreased below a pre-defined tolerance, training ceases. It is worth remarking the nonlinear functions can be represented by MLP

Brazilian Microwave and Optoelectronics Society-SBMO *received 21 Apr 2016; for review 28 Apr 2016; accepted 20 Oct 2016*
Brazilian Society of Electromagnetism-SBMag *ISSN 2179-1074*

Journal of Microwaves, Optoelectronics and Electromagnetic Applications, Vol. 15, No. 4, December 2016 423
DOI: http://dx.doi.org/10.1590/2179-10742016v15i4816

with units that use nonlinear activation functions. Since the BP algorithm requires an activation function to be differentiable, we used a sigmoid (logistic function) [10]. Another important aspect in defining the topology of neural network is the number of neurons in the hidden layer, due to its influence on both the speed and the effectiveness of the learning network, which was defined by the geometric mean between the input and output neurons [11].

III. VALIDATION

In order to verify the effectiveness of proposed method, it was initially used in the optimization of a test function with analytical solution. Specifically, the goal is to find the global maximum of the Peaks function, given by (4) and shown in Fig. 3. It is important to mention that the Rastrigin and Sinc functions were also tested during the validation process of this study, with satisfactory and similar results for all of them. The results obtained through the functions Peaks, Rastring and Sinc showed the efficiency and effectiveness necessary to validate the proposed PSO optimizer assisted by an ANN.

$$F(x,y) = 3(1-x)^2 e^{-x^2-(y+1)^2} - \left(2x - 10x^3 - 10y^5\right) e^{-x^2-y^2} - \frac{1}{3} e^{-(x+1)^2-y^2} \tag{4}$$

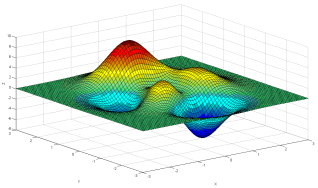


Fig.3. The analytical function considered.

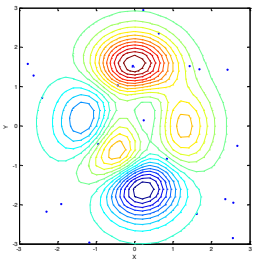
The goal of this implementation is to test the effectiveness of the PSO optimizer using an ANN (network MLP with backpropagation algorithm) as a metamodel.

The optimization was done both directly using the function given by (4) and using an ANN-based metamodel to represent it. Specifically, the PSO was used to obtain the global maximum location of Peaks function. The algorithm was initialized with a random set of particles and the process took place until the maximum point was located, at the center of the red contours.

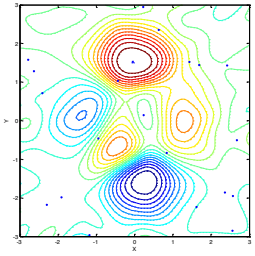
Fig. 4(a) shows the contour plot of the results obtained using directly (4) while fig. 4(b) shows the results obtained using the ANN simulating the function given by (4). The global maximum is represented by "×" in fig. 4(a).

Brazilian Microwave and Optoelectronics Society-SBMO *received 21 Apr 2016; for review 28 Apr 2016; accepted 20 Oct 2016*
Brazilian Society of Electromagnetism-SBMag *ISSN 2179-1074*

Journal of Microwaves, Optoelectronics and Electromagnetic Applications, Vol. 15, No. 4, December 2016 424
DOI: http://dx.doi.org/10.1590/2179-10742016v15i4816



(a)



(b)

Fig.4. (a). Contour plot of the actual function; (b) Contour plot of the simulated function, with the located peak.

The ANN was trained until a root mean square error of 0.0004 was obtained, requiring 27,253 epochs to get this error value. An epoch is a step in the training process of an ANN, i.e., each time the network is presented with a new input pattern. The root mean square error solution of PSO assisted by the ANN was equal to 0.00569. From the results, we can conclude that the use of the proposed metamodel to replace the original fitness function allows obtaining good results.

IV. ELECTROMAGNETIC PROBLEM

The use of an electromagnetic prediction model represents an important design procedure in wireless systems. Here, we use a quasi-3D ray-tracing propagation model associated with a PSO optimizer to find optimal antenna placement in an indoor scenario [9-11].

Initially, a population is generated where each particle corresponds to the coordinates (x, y) of the

Brazilian Microwave and Optoelectronics Society-SBMO *received 21 Apr 2016; for review 28 Apr 2016; accepted 20 Oct 2016*
Brazilian Society of Electromagnetism-SBMag *ISSN 2179-1074*

Journal of Microwaves, Optoelectronics and Electromagnetic Applications, Vol. 15, No. 4, December 2016 425
DOI: http://dx.doi.org/10.1590/2179-10742016v15i4816

transmitting antennas placements in a 40 m × 28 m scenario. It was defined a mesh of 160 receptors uniformly distributed within the environment. The antennas of all receivers were considered to be 1.5 m high from the ground. The method was applied to minimize the number of points where the field is below a minimum reception threshold (-110 dBm). This is equivalent to maximize the coverage area. The variable of the problem was assumed to be a vector containing the coordinates (x, y) of the transmitting antennas, allowing different locations of them on the horizontal plane of the scenario. The height of these transmitting antennas was assumed to be fixed (3 m), simulating vertically polarized quarter-wave monopole antennas fixed to the ceiling. Two antennas were considered in the problem. Therefore, the variable of the problem is a vector containing four elements, which represent coordinates of transmitting antenna positions on the horizontal plane. The frequency used was 2.4 GHz, and the total equivalent isotropically radiated power (EIRP) was 400 mW.

Firstly, the field values generated by ray-tracing are used as a guide in the learning process performed by the network, optimizing weight values, which are saved in a file. After training, the ANN calculates the field values at all reception points, which are used by the PSO optimizer, allowing a great reduction in computation time.

The PSO assisted by ANN was initialized with a set of 100 random particles. The PSO assisted by ANN optimizing was performed for three different numbers of iterations: 2,000, 3,000 and 6,000. Table I shows the best antenna locations found by the PSO for these three cases. We observe that the results are very sensitive to the number of iterations, which indicates that the ANN needs to be further improved. Fig. 5 shows the mapping of received powers considering the best pair of positions of the antennas found for 3,000 iterations. We see that the distribution of field is reasonable. Best results, with more uniform fields, can be obtained using more transmitting antennas in the scenario.

TABLE I. RESULTS OF OPTIMIZATION ANN-ASSISTED PSO

Iterations	**Positions Antenna 1 (x_1(m), y_1(m))**	**Positions Antenna 2 (x_2(m), y_2(m))**
2,000	(25.61, 9.56)	(34.04, 17.58)
3,000	(20.18, 7.56)	(29.12, 15.82)
6,000	(34.50, 4.91)	(32.00, 6.65)

Brazilian Microwave and Optoelectronics Society-SBMO *received 21 Apr 2016; for review 28 Apr 2016; accepted 20 Oct 2016*
Brazilian Society of Electromagnetism-SBMag *ISSN 2179-1074*

Journal of Microwaves, Optoelectronics and Electromagnetic Applications, Vol. 15, No. 4, December 2016 426
DOI: http://dx.doi.org/10.1590/2179-10742016v15i4816

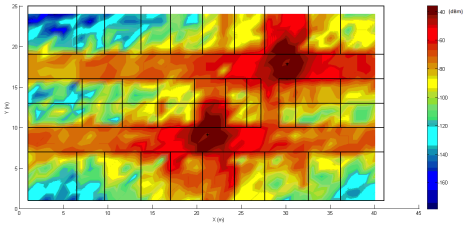


Fig.5. Received power mapping in the indoor environment (3,000 iteractions).

V. CONCLUSIONS

In this paper, we proposed a PSO optimizer assisted by an ANN, the optimizer was implemented using the MATLAB program version 7.10.0.499. The optimization process was carried out through a MacBook Pro Retina Computer with 2,6 GHz intel Core i5 processor and 8GB of RAM memory 1600MHz DDR3, with 256GB of flash storage. In the optimization of an electromagnetic problem, the computational cost can be very large, since it is necessary to carry out a numerical simulation for each evaluation of the fitness function. In this case, the use of an ANN as a metamodel permits to obtain satisfactory results with a reduction of the computational cost. The study developed in this work is based on [9], where the association of RTQ3D-indoor with PSO optimizers was applied in a typical telecommunication problem: a wireless local area network (WLAN) system in an indoor scenario that must be covered with two 802.11b/g/n (2.4 GHz) access points (AP). However, the study developed in [9] disregards the computational cost, which is quite high when a metamodel is not used. As an example, the average time required for optimization using the ANN metamodel with 3,000 epochs is around 36 min. Without using the metamodel, the PSO takes about 72 hours to produce slightly better results.

In order to assess the effectiveness of the proposed method, it was applied in a test function with a global maximum. Also, it was applied in an electromagnetic problem (namely, the best antenna positioning in wireless systems). The response of the ANN-assisted PSO for a real electromagnetic problem can be considered satisfactory. All observations suggest that an improvement in the ANN and additional training will lead to the best results. However, this would increase the computational time.

Brazilian Microwave and Optoelectronics Society-SBMO *received 21 Apr 2016; for review 28 Apr 2016; accepted 20 Oct 2016*
Brazilian Society of Electromagnetism-SBMag *ISSN 2179-1074*

DOI: http://dx.doi.org/10.1590/2179-10742016v15i4816

REFERENCES

[1] H.P. Schwefel, E. L. Taylor, "Evolution and Optimum Seeking," *John Wiley & Sons. Inc*, United States of America, pp. 87-88, 1994.
[2] G. Venter, and J. Sobieszczanski-sobieski, "Particle Swarm Optimization," *Proceedings of the 43rd AIAA/ASME/ASCE/AHS/ASC Strutures, Structural Dynamics, and Materials Conference*, Denver, CO, Vol. AIAA-2002-1235, April 22-25 2002.
[3] L. Lebensztjan, C. A. R. Marretto, M. C. Costa, J. L. Coulomb, "Kriging: a useful tool for electromagnetic device optimization," vol.40, N°.2. IEEE Transactions on Magnetics, March 2004, p. 1196-99.
[4] M. T. M. Emmerich, C. M. Varcol, "Metamodel–Assisted Evolution Strategies in Electromagnetic Compatibility Design," *EUROGEN*. 1-12, 2005.
[5] L. Willmes, T. Bäck, Y. Jin, "Sendhoff, B. Comparing neural network and kriging for fitness approximation in evolutionary optimization," IEEE, 2003 pp. 670.
[6] Y. Rahmat-Samii, N. Jin, "Particle Swarm Optimization (PSO) in Engineering Electromagnetics: A Nature-Inspired Evolutionary Algorithm," IEEE, 2007. Inc, United States of America, 1994 pp. 87-88.
[7] R. C. Eberhart and Y. Shi, "Particle Swarm Optimization: Developments, Applications and Resources," *Congress on Evolutionary Computation, 2001*, vol. 1, pp. 81-86, 2001.
[8] S. Grubisic, W. P. Carpes Jr, J. P. A. Bastos, "An Efficient indoor ray- tracing propagation model with a quasi-3D approach," IEEE CEFC 2012, November 2012, Oita/Japan.
[9] S. Grubisic, W. P. Carpes Jr, J. P. A. Bastos, G. Santos, "Association of a PSO optimizer with a *quasi*-3D ray-tracing propagation model for mono and multi-criterion antenna positioning in indoor environments," IEEE Transactions on Magnetics, Vol. 49, No. 5, May 2013.
[10] S. Haykin, "Neural Networks: A Comprehensive Foundation," Prentice Hall, 1999.
[11] J. M. Barreto, "Neural Networks: Mathematical and Computational Aspects," *Annals of the Brazilian Society of Applied and Computational Mathematics (BSACM)*, 1996.

Brazilian Microwave and Optoelectronics Society-SBMO *received 21 Apr 2016; for review 28 Apr 2016; accepted 20 Oct 2016*
Brazilian Society of Electromagnetism-SBMag *ISSN 2179-1074*