De acordo com [1] o método para a solução de equações diferenciais ordinárias (EDOs) baseia-se na capacidade de aproximação de funções das redes neurais feedforward e resulta na construção de uma solução em forma analítica fechada e diferenciável. Esta forma emprega uma rede neural feedforward como elemento de aproximação básico, cujos parâmetros (pesos e vieses) são ajustados para minimizar uma função de erro apropriada. Técnicas de otimização são utilizadas para minimizar a quantidade de erro e treinamento da rede, o que por sua vez requer o cálculo do gradiente de erro em relação às entradas e parâmetros da rede.

$$F(\vec{x}, y(\vec{x}), \nabla y(\vec{x}), \nabla^2 y(\vec{x})) = 0, \quad \vec{x} \in D \tag{1}$$

definido em certas condições de contorno onde $\bar{x} = (x_1, x_2, \ldots, x_n) \in \mathbb{R}^n, D \subset \mathbb{R}^n$ denota o domínio de definição e $y(\vec{x})$ é a solução a ser computada. Os seguintes passos são necessários para o cálculo da equação diferencial acima Fig. 1.

**(a) Transformação**

Primeiro, discretizamos o domínio $D$ e sua fronteira $S$ em um conjunto de pontos discretos $\tilde{D}$ e $\tilde{S}$, respectivamente. O problema é então transformado em um sistema de equações

$$F(\vec{x}, y(\vec{x}), \nabla y(\vec{x}), \nabla^2 y(\vec{x})) = 0 \quad \forall \vec{x} \in \tilde{D} \tag{2}$$

sujeito às restrições impostas pelas condições de contorno. Se $y_t(\vec{x}, \vec{w})$ denota uma solução de teste com os parâmetros ajustáveis $\vec{w}$, o problema é transformado em um problema de otimização

$$\min_{\vec{p}} \sum_{\vec{x} \in \tilde{D}} F(\vec{x}, y_t(\vec{x}, \vec{w}), \nabla y_t(\vec{x}, \vec{w}), \nabla^2 y_t(\vec{x}, \vec{w}))^2 \tag{3}$$

sujeito às restrições impostas pelas condições de contorno.

**(b) Construção da Solução de Teste**

Para construir a solução de teste $y_t(\vec{x})$, assumimos que a função de teste satisfaz as condições de contorno dadas e é a soma de dois termos - um é independente dos parâmetros ajustáveis, e o outro é com parâmetros ajustáveis. Suponha que a função de teste seja

$$y_t(\vec{x}) = A(\vec{x}) + f(\vec{x}, N(\vec{x}, \vec{w})) \tag{4}$$

onde $A(\vec{x})$ não contém parâmetros ajustáveis que satisfaçam as condições iniciais de contorno e $N(\vec{x}, \vec{w})$ é uma rede neural de alimentação direta de saída única com parâmetros $\vec{w}$ e $n$ entradas com o vetor de entrada $\vec{x}$. O segundo termo $f$ é construído de uma forma tal que ele não contribui para a condição de contorno, já que $y_t(\vec{x})$ também deve satisfazê-la. Este termo representa uma rede neural cujos parâmetros são ajustados. Para resolver o problema de minimização, o problema foi reduzido para o problema de otimização sem restrições do problema original com restrições que é muito mais fácil de tratar devido à escolha da forma da solução de teste que satisfaz as condições de contorno por construção.

## 0.1 Cálculo do Gradiente

A minimização eficiente da equação (3) pode ser considerada como um procedimento de treinamento da rede neural onde o erro correspondente a cada vetor de entrada $\vec{x}_i$ é o valor $F(\vec{x}_i)$ que tem que se tornar zero. O cálculo deste valor de erro envolve não apenas a saída da rede (como é o caso no treinamento convencional), mas também as derivadas da saída em relação a todos os seus

inputs. Portanto, no cálculo do gradiente do erro em relação aos pesos da rede, precisamos calcular não apenas o gradiente da rede, mas também o gradiente das derivadas da rede em relação aos seus inputs. Considere um perceptron multicamadas com $n$ unidades de entrada, uma camada oculta com $H$ unidades sigmoides e uma saída linear. A extensão para o caso de mais de uma camada oculta pode ser obtida de forma semelhante. Para um dado vetor de entrada $\vec{x} = (x_1, ..., x_n)$, a saída da rede é $N = \sum_{k=1}^{H} v_i \varphi(s_k)$ onde $s_k = \sum_{j=1}^{n} u_j + b_j$, $u_j$ denota o peso da unidade de entrada $j$ para a unidade oculta $i$, $b_j$ denota o peso da unidade oculta $i$ para a saída, $v_i$ denota o viés da unidade oculta $i$ e $\varphi(x)$ é a função de transferência sigmoide. É direto mostrar que:

$$\frac{\partial^k N}{\partial x_j^k} = \sum_{i=1}^{H} v_i w_{ij}^k \varphi_i^{(k)} \tag{5}$$

onde $\varphi_i = \varphi(s_i)$ e $\varphi^{(k)}$ denota a $k$-ésima derivada da sigmoide. Além disso, é facilmente verificável que:

$$\frac{\partial^{\lambda_1}}{\partial x_1^{\lambda_1}} \frac{\partial^{\lambda_2}}{\partial x_2^{\lambda_2}} ... \frac{\partial^{\lambda_n}}{\partial x_2^{\lambda_n}} N = \sum_{i=1}^{n} v_i P_i \varphi_i^{(\Lambda)} \tag{6}$$

onde:

$$P_i = \prod_{k=1}^{n} w_{ik}^{\lambda_k} \tag{7}$$

e $\Lambda = \sum_{i=1}^{N} \lambda_i$. A equação (6) indica que a derivada da rede em relação a quaisquer de suas entradas é equivalente a uma rede neural direta $N_g(\vec{x})$ com uma camada oculta, tendo os mesmos valores para os pesos $w_{ij}$ e limiares $u_i$, e com cada peso $w_{ij}$ sendo substituído por $w_{ij} P_i$. Além disso, a função de transferência de cada unidade oculta é substituída pela $k$-ésima derivada da sigmoide. Portanto, o gradiente de $N_g$ em relação aos parâmetros da rede original pode ser facilmente obtido como:

$$\frac{\partial N_g}{\partial v_i} = v_i P_i \varphi_i^{(\Lambda)} \tag{8}$$

$$\frac{\partial N_g}{\partial u_i} = v_i P_i \varphi_i^{(\Lambda+1)} \tag{9}$$

$$\frac{\partial N_g}{\partial w_{ij}} = x_j P_i g^{(1)}(s_i) + w_{i0} g^{(2)}(s_i) \prod_{k=1, k \neq j}^{n} w_{ik} \tag{10}$$

Uma vez que a derivada do erro em relação a todos os parâmetros da rede foi definida, é então direto empregar quase qualquer técnica de minimização. Por exemplo, é possível usar o método de descida mais íngreme (ou seja, o algoritmo de retropropagação ou qualquer uma de suas variantes), ou a técnica do gradiente conjugado ou outras técnicas propostas na literatura. Em nossos experimentos, empregamos o método BFGS [ ] que é quadraticamente convergente e demonstrou um excelente desempenho. Deve-se notar também que para um determinado ponto da grade, as derivadas de cada rede (ou rede de gradiente) em relação aos parâmetros podem ser obtidas simultaneamente no caso em que hardware paralelo está disponível. Além disso, no caso de retropropagação, o modo on-line ou em lote de atualizações de peso pode ser empregado.