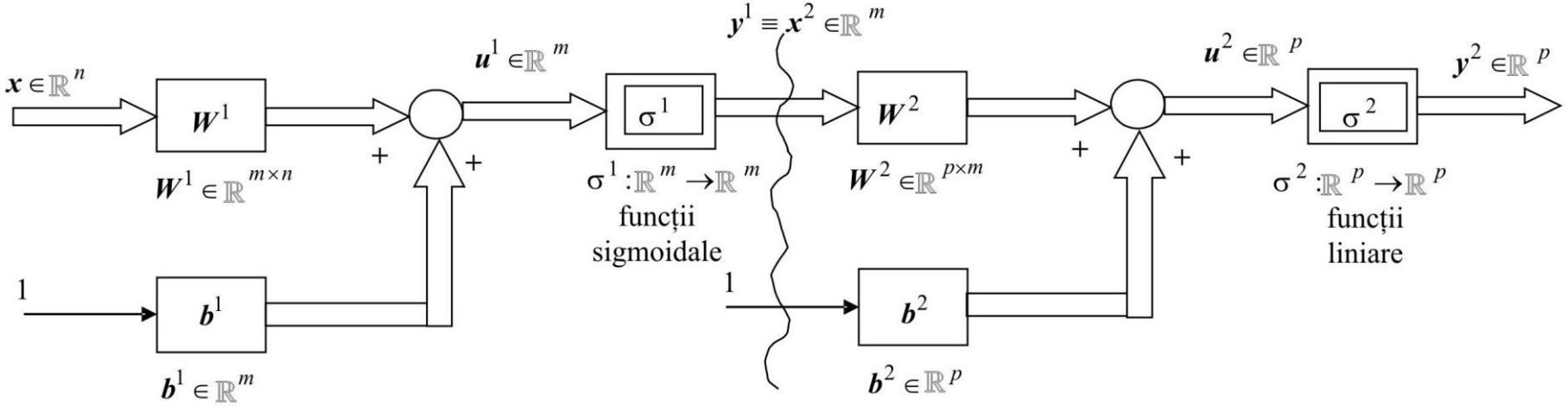
4.1. Proprietatea de aproximare a rețelelor feed-forward cu două straturi



$$\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^p, \quad \boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{\sigma}^2 \; \boldsymbol{W}^2 \boldsymbol{\sigma}^1 (\boldsymbol{W}^1 \boldsymbol{x} + \boldsymbol{b}^1) + \boldsymbol{b}^2 \qquad (5.1.1)$$

$\boldsymbol{x} \in \mathbb{R}^n$ – vectorul de intrare;

$\boldsymbol{y} \in \mathbb{R}^p$ – vectorul de ieşire;

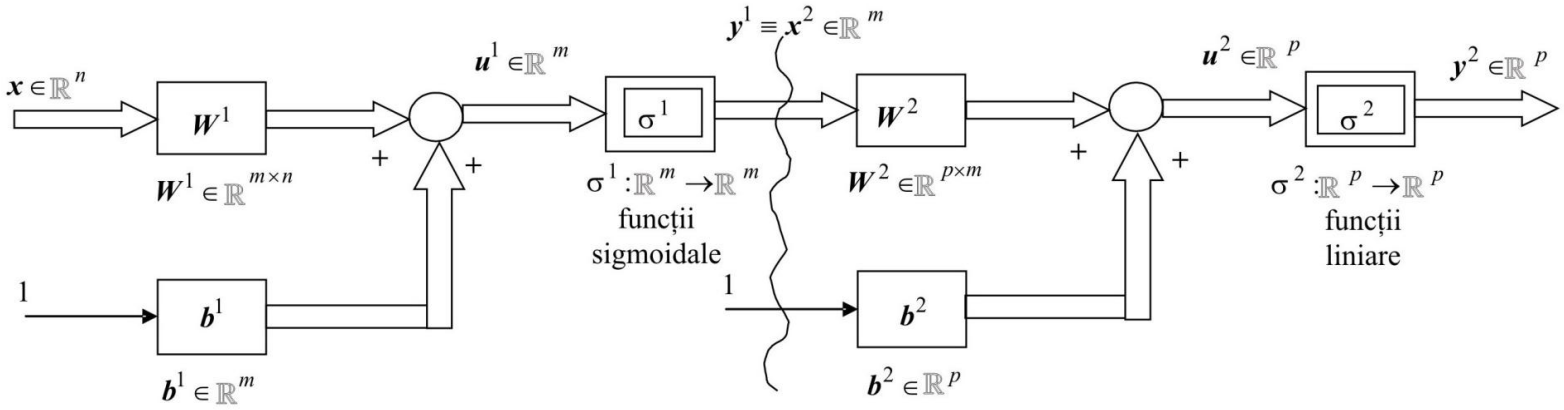
Parametrii rețelei:

$\boldsymbol{W}^1 \in \mathbb{R}^{m \times n}$ – ponderi strat de intrare;

$\boldsymbol{b}^1 \in \mathbb{R}^m$ – deplasări strat de intrare;

$\boldsymbol{W}^2 \in \mathbb{R}^{p \times m}$ – ponderi strat de ieşire;

$\boldsymbol{b}^2 \in \mathbb{R}^p$ – deplasări strat de ieşire;



$\boldsymbol{\sigma}^1 : \mathbb{R}^m \to \mathbb{R}^m$, $\boldsymbol{\sigma}^1(\boldsymbol{u}^1) = \left[\sigma^1(u_1^1) \quad \ldots \quad \sigma^1(u_m^1) \right]^T$ – funcția de activare sigmoidală

$$\boldsymbol{u}^1 = \boldsymbol{W}^1 \boldsymbol{x} + \boldsymbol{b}^1 \overset{not}{=} \left[u_1^1 \ldots u_m^1 \right]^T, \text{ cu} \qquad \sigma^1(u) = \frac{1}{(1+e^u)} \text{ sau } \sigma^1(u) = \frac{e^u - e^{-u}}{e^u + e^{-u}} ;$$

$\boldsymbol{\sigma}^2 : \mathbb{R}^p \to \mathbb{R}^p$, $\boldsymbol{\sigma}^2(\boldsymbol{u}^2) = \left[\sigma^2(u_1^2) \ldots \sigma^2(u_p^2) \right]^T = \boldsymbol{u}^2$, $\sigma^2(u) = u$ – funcția de activare liniară

$$\boldsymbol{u}^2 = \boldsymbol{W}^2 \boldsymbol{x}^2 + \boldsymbol{b}^2 \overset{not}{=} \left[u_1^1 \ldots u_p^1 \right]^T \in \mathbb{R}^p, \; \boldsymbol{x}^2 \equiv \boldsymbol{y}^1 = \boldsymbol{\sigma}^1(\boldsymbol{u}^1).$$

Fie funcția netedă pe o mulțime compactă $S \subseteq \mathbb{R}^n$:

$$\varphi : \mathbb{R}^n \to \mathbb{R}^p, \tag{5.1.2}$$

cunoscută în punctele $\boldsymbol{x}_i \in S \subseteq \mathbb{R}^n$:

$$\varphi(\boldsymbol{x}_i) = \boldsymbol{z}_i, \quad i = 1, \ldots, N.$$

Dacă rețeaua neuronală posedă *un număr adecvat de noduri m* în stratul de intrare, atunci parametrii rețelei pot fi determinați de așa manieră încât $\boldsymbol{f}$ (5.1.1) *să aproximeze* pe domeniul S funcția dată φ (5.1.2) *cu o precizie impusă* (în sensul de „oricât de bună"):

$$\begin{gathered} \forall \varepsilon > 0, \quad \exists m \in \mathbb{N} \text{ şi } \boldsymbol{W}^1 \in \mathbb{R}^{m \times n}, \boldsymbol{b}^1 \in \mathbb{R}^m, \boldsymbol{W}^2 \in \mathbb{R}^{p \times m}, \boldsymbol{b}^2 \in \mathbb{R}^p \text{ a.î.} \\ \left\| \boldsymbol{f}(\boldsymbol{x}) - \varphi(\boldsymbol{x}) \right\| < \varepsilon, \quad \forall \boldsymbol{x} \in S. \end{gathered} \tag{5.1.3}$$

Rețeaua neurală MLP: Se alege numărul de neuroni *m* în stratul de intrare al rețelei

vectori prototip: $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N \in \mathbb{R}^n$,

vectori țintă: $\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_N \in \mathbb{R}^p$

vectori de ieşire: $\boldsymbol{y}_i = \boldsymbol{f}(\boldsymbol{x}_i) \in \mathbb{R}^p, \quad i = 1, \ldots, N,$

vectori eroare: $\boldsymbol{e}_i = \boldsymbol{z}_i - \boldsymbol{y}_i \in \mathbb{R}^p, \quad i = 1, \ldots, N.$

funcția obiectiv MSE: $J_{MSE}\left(\boldsymbol{W}^1, \boldsymbol{b}^1, \boldsymbol{W}^2, \boldsymbol{b}^2\right) = \dfrac{1}{N} \sum_{i=1}^{N} \boldsymbol{e}_i^T \boldsymbol{e}_i.$

$\forall m \Rightarrow \exists\, \boldsymbol{W}^{1*} \in \mathbb{R}^{m \times n}, \boldsymbol{b}^{1*} \in \mathbb{R}^m, \boldsymbol{W}^{2*} \in \mathbb{R}^{p \times m}, \boldsymbol{b}^{2*} \in \mathbb{R}^p$, astfel încât:

$$J_{MSE}\left(\boldsymbol{W}^{1*}, \boldsymbol{b}^{1*}, \boldsymbol{W}^{2*}, \boldsymbol{b}^{2*}\right) \leq J_{MSE}\left(\boldsymbol{W}^1, \boldsymbol{b}^1, \boldsymbol{W}^2, \boldsymbol{b}^2\right).$$

4.2. Antrenarea rețelelor feedforward cu două straturi

5.2.1. Obiectivul antrenării rețelei

- determinarea parametrilor $\boldsymbol{W}^{1*} \in \mathbb{R}^{m \times n}$, $\boldsymbol{b}^{1*} \in \mathbb{R}^{m}$, $\boldsymbol{W}^{2*} \in \mathbb{R}^{p \times m}$, $\boldsymbol{b}^{2*} \in \mathbb{R}^{p}$ care să asigure *minimizarea erorii pătratice globale*
- m va influența calitatea aproximării realizate de rețea pentru funcția φ

5.2.2. Algoritmul de antrenare prin propagare inversă

backpropagation: *minimizarea erorii pătratice medii* (MSE) prin *metoda gradientului*.

Fie $\boldsymbol{v} \in \mathbb{R}^{m(n+1)+p(m+1)} = vect\left[\boldsymbol{W}^1\ \boldsymbol{b}^1\right];\left[\boldsymbol{W}^2\ \boldsymbol{b}^2\right]$, $\boldsymbol{W}^1 \in \mathbb{R}^{m \times n}$, $\boldsymbol{W}^2 \in \mathbb{R}^{p \times m}$, $\boldsymbol{b}^1 \in \mathbb{R}^{m}$, $\boldsymbol{b}^2 \in \mathbb{R}^{p}$

Funcția obiectiv:

$$J(\boldsymbol{v}) = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{e}_i^{\,T}(\boldsymbol{v})\boldsymbol{e}_i(\boldsymbol{v}). \tag{5.2.1}$$

Minimizarea prin metoda gradientului:

$$\boldsymbol{v}_{\text{new}} = \boldsymbol{v}_{\text{old}} - \eta \nabla J(\boldsymbol{v}_{\text{old}}), \tag{5.2.2}$$

$\eta > 0$ - *pasul metodei de optimizare*,

$\nabla J(\boldsymbol{v}_{\text{old}}) = \left.\partial J / \partial \boldsymbol{v}^{\,T}\right|_{\boldsymbol{v}_{\text{old}}}$ - vectorul gradient (normal la suprafețele de nivel constant ale lui J care trec prin $\boldsymbol{v}_{\text{old}}$).

Etapa 0 (Inițializare)

se fixează:

un *număr maxim de iterații* sau *epoci*

o *valoare de prag* $\varepsilon > 0$ care se dorește a fi atinsă pentru funcția criteriu

rată de învățare (eng. *learning rate*) = pasul metodei de optimizare $\eta > 0$

se construiesc:

vectorii prototip $\boldsymbol{X} = [\boldsymbol{x}_1 \dots \boldsymbol{x}_N] \in \mathbb{R}^{n \times N}$, (5.2.3)

vectorii țintă: $\boldsymbol{Z} = [\boldsymbol{z}_1 \dots \boldsymbol{z}_N] \in \mathbb{R}^{p \times N}$. (5.2.4)

Etapa I (Etapa de prezentare)

$$\boldsymbol{y}_i = \boldsymbol{f}(\boldsymbol{x}_i) = \boldsymbol{\sigma}^2 \left(\boldsymbol{W}^2_{\text{old}} \boldsymbol{\sigma}^1 (\boldsymbol{W}^1_{\text{old}} \boldsymbol{x}_i + \boldsymbol{b}^1_{\text{old}}) + \boldsymbol{b}^2_{\text{old}} \right), \; i = 1, \dots, N, \tag{5.2.5}$$

$$\boldsymbol{Y} = [\boldsymbol{y}_1 \dots \boldsymbol{y}_N] \in \mathbb{R}^{p \times N}. \tag{5.2.6}$$

Etapa II (Etapa de verificare)

$$\boldsymbol{e}_i = \boldsymbol{z}_i - \boldsymbol{y}_i \in \mathbb{R}^p, \; i = 1, \dots, N.$$

$$\boldsymbol{E} = [\boldsymbol{e}_1 \dots \boldsymbol{e}_N] = \boldsymbol{Z} - \boldsymbol{Y} \in \mathbb{R}^{p \times N}. \tag{5.2.7}$$

Algoritmul se oprește dacă este îndeplinită una din următoarele două condiții:

($\chi 1$) eroarea pătratică medie este inferioară pragului ε stabilit la startarea algoritmului, adică:

$$J\left(\boldsymbol{W}^2_{\text{old}}, \boldsymbol{W}^1_{\text{old}}, \boldsymbol{b}^1_{\text{old}}, \boldsymbol{b}^2_{\text{old}}\right) = \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{e}_i^T \boldsymbol{e}_i < \varepsilon; \tag{5.2.8}$$

($\chi 2$) a fost atins numărul maxim de iterații stabilit la startarea algoritmului.

Etapa III (Etapa de calcul a vectorilor delta prin propagare inversă)

pentru valorile curente $\boldsymbol{W}_{\text{old}}^1 \in \mathbb{R}^{m\times n}$, $\boldsymbol{W}_{\text{old}}^2 \in \mathbb{R}^{p\times m}$, $\boldsymbol{b}_{\text{old}}^1 \in \mathbb{R}^m$, $\boldsymbol{b}_{\text{old}}^2 \in \mathbb{R}^p$

pentru fiecare vector prototip $\boldsymbol{x}_i$, $i=1,\ldots,N$, se calculează *matricele Jacobian ale funcțiilor de activare* (vectoriale) ale celor două straturi:

- pentru stratul de intrare:

$$\boldsymbol{Q}_i^1 = \text{diag}\ \left(q_i^1(1),\ldots,q_i^1(m)\right),\ \boldsymbol{Q}_i^1 \in \mathbb{R}^{m\times m},\ i=1,\ldots,N, \tag{5.2.9-a}$$

unde:

$$q_i^1(k) = (\sigma^1)(u_i^1(k)) = \frac{1}{(\text{Inv}\,\sigma^1)(y_i^1(k))},\quad k=1,\ldots,m; \tag{5.2.9-b}$$

- pentru stratul de ieşire:

$$\boldsymbol{Q}_i^2 = \text{diag}\ \left(q_i^2(1),\ldots,q_i^2(m)\right),\ \boldsymbol{Q}_i^2 \in \mathbb{R}^{p\times p},\ i=1,\ldots,N, \tag{5.2.10-a}$$

unde:

$$q_i^2(k) = (\sigma^2)(u_i^2(k)) = \frac{1}{(\text{Inv}\,\sigma^2)(y_i^2(k))} = 1,\quad k=1,\ldots,m. \tag{5.2.10-b}$$

funcțiile de activare pe al doilea strat sunt liniare $\Rightarrow$ $\boldsymbol{Q}_i^2 = \boldsymbol{I}$ este *matricea unitate*, $i=1,\ldots,N$.

se definesc *vectorii / matricele delta*:

- pentru stratul de ieşire:

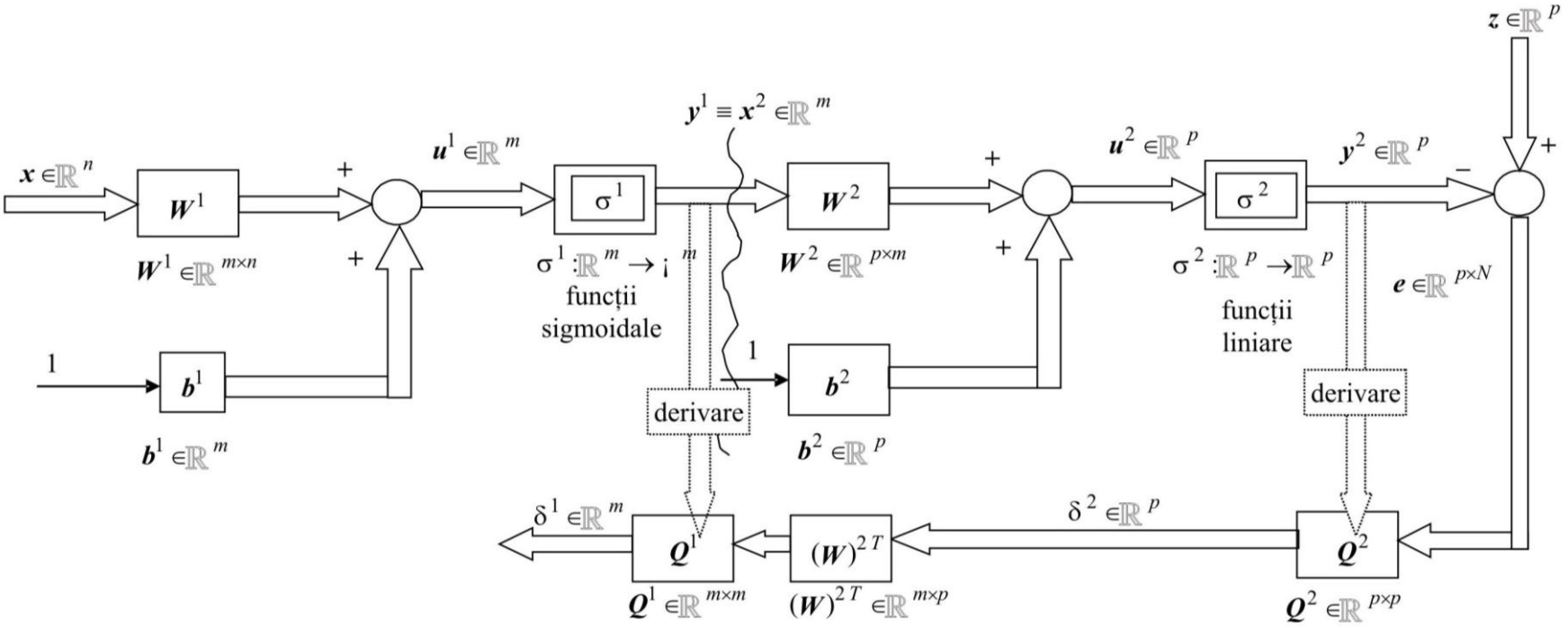
$$\boldsymbol{\delta}_i^2 = \boldsymbol{Q}_i^2 \boldsymbol{e}_i = \boldsymbol{e}_i \in \mathbb{R}^p, \quad i = 1, \ldots, N; \tag{5.2.11}$$

$$\Delta^2 = [\boldsymbol{\delta}_i^2 \ldots . \boldsymbol{\delta}_N^2] \in \mathbb{R}^{p \times N}, \tag{5.2.13}$$

- pentru stratul de intrare:

$$\boldsymbol{\delta}_i^1 = \boldsymbol{Q}_i^1 \ (\boldsymbol{W}^2)^T \boldsymbol{\delta}_i^2 \ \in \mathbb{R}^m, \quad i = 1, \ldots, N. \tag{5.2.12}$$

$$\Delta^1 = [\boldsymbol{\delta}_i^1 \ldots . \boldsymbol{\delta}_N^1] \in \mathbb{R}^{m \times N}. \tag{5.2.14}$$



Etapa IV (Etapa de actualizare a parametrilor)

➢ se calculează *matricele de actualizare* a parametrilor aplicând *regula delta* pe fiecare strat:

- pentru stratul de ieşire

ponderi $$\boldsymbol{D}_{\boldsymbol{W}^2} = \Delta^2 (\boldsymbol{X}^2)^T \in \mathbb{R}^{p\times m}, \tag{5.2.15-a}$$

deplasări $$\boldsymbol{D}_{\boldsymbol{b}^2} = \Delta^2 \underbrace{1 \ldots 1}_{N \text{ elemente}}^{T} \in \mathbb{R}^{p}; \tag{5.2.15-b}$$

cu $$\boldsymbol{x}_i^2 = \boldsymbol{y}_i^1 \in \mathbb{R}^m,\ i = 1,\ldots,N \ \Rightarrow\ \boldsymbol{X}^2 = [\boldsymbol{x}_1^2 \ldots \boldsymbol{x}_N^2] \in \mathbb{R}^{m\times N}. \tag{5.2.17}$$

- pentru stratul de intrare

ponderi $$\boldsymbol{D}_{\boldsymbol{W}^1} = \Delta^1 \boldsymbol{X}^T \in \mathbb{R}^{m\times n}, \tag{5.2.16-a}$$

deplasări $$\boldsymbol{D}_{\boldsymbol{b}^1} = \Delta^1 \underbrace{1 \ldots 1}_{N \text{ elemente}}^{T} \in \mathbb{R}^{m}. \tag{5.2.16-b}$$

➢ se determină noile valori ale parametrilor rețelei:

- pentru stratul de ieşire

$$\boldsymbol{W}^2_{\text{new}} = \boldsymbol{W}^2_{\text{old}} + \eta \boldsymbol{D}_{\boldsymbol{W}^2},\ \ \boldsymbol{b}^2_{\text{new}} = \boldsymbol{b}^2_{\text{old}} + \eta \boldsymbol{D}_{\boldsymbol{b}^2}; \tag{5.2.18}$$

- pentru stratul de intrare

$$\boldsymbol{W}^1_{\text{new}} = \boldsymbol{W}^1_{\text{old}} + \eta \boldsymbol{D}_{\boldsymbol{W}^1},\ \ \boldsymbol{b}^1_{\text{new}} = \boldsymbol{b}^1_{\text{old}} + \eta \boldsymbol{D}_{\boldsymbol{b}^1}. \tag{5.2.19}$$

➢ se trece la o nouă iterație cu:

$$\boldsymbol{W}^2_{\text{new}} \to \boldsymbol{W}^2_{\text{old}},\quad \boldsymbol{b}^2_{\text{new}} \to \boldsymbol{b}^2_{\text{old}},\quad \boldsymbol{W}^1_{\text{new}} \to \boldsymbol{W}^1_{\text{old}},\quad \boldsymbol{b}^1_{\text{new}} \to \boldsymbol{b}^1_{\text{old}}. \tag{5.2.20}$$

Observații:

1° Inițializarea parametrilor rețelei (adică $\boldsymbol{W}_{\text{old}}^{1} \in \mathbb{R}^{m\times n}$, $\boldsymbol{W}_{\text{old}}^{2} \in \mathbb{R}^{p\times m}$, $\boldsymbol{b}_{\text{old}}^{1} \in \mathbb{R}^{m}$, $\boldsymbol{b}_{\text{old}}^{2} \in \mathbb{R}^{p}$ la prima iterație a algoritmului) se face cu valori *subunitare arbitrare*; în acest mod se asigură senzitivitate maximă pentru funcțiile sigmoidale din primul strat al rețelei.

2° Convergența algoritmului de propagare inversă se realizează în condițiile specifice metodei gradientului (ca tehnică de optimizare), fiind puternic dependentă de valoarea ratei de învățare η şi anume:

- pentru η inadecvat mic, deplasarea către optim, în spațiul parametrilor, este foarte lentă,
- pentru η inadecvat mare, deplasarea către optim, în spațiul parametrilor, se realizează cu oscilații (eventual cu pierderea stabilității algoritmului).

3° În cazul când algoritmul se încheie prin satisfacerea condiției (χ1), acuratețea parametrilor furnizați $\boldsymbol{W}_{f}^{1} \in \mathbb{R}^{m\times n}$, $\boldsymbol{W}_{f}^{2} \in \mathbb{R}^{p\times m}$ (ponderi) şi $\boldsymbol{b}_{f}^{1} \in \mathbb{R}^{m}$, $\boldsymbol{b}_{f}^{2} \in \mathbb{R}^{p}$ (deplasări) depinde de valoarea pragului ε stabilit pentru eroarea pătratică globală. Valoarea finală a lui J (implicit alegerea valorii de prag ε) trebuie corelată cu magnitudinea valorilor țintă $\boldsymbol{z}_i$, $i=\overline{1,N}$. De exemplu, dacă $\boldsymbol{z}_i$ este de ordinul 10^6 în unitățile de măsură, eroarea poate fi aleasă de ordinul 10^4-10^3; nu vom utiliza drept criteriu de antrenare un prag de ordinul 10^{-2} pentru J deoarece acesta nu ar putea fi atins practic niciodată.

4° Suprafața erorii $J(\boldsymbol{v})$ definită prin (10) uzual posedă *minime locale* în care căutarea minimului poate eşua, dacă rata de învățare are o valoare prea mică. Astfel de minime locale nu pot fi părăsite decât prin schimbarea valorii ratei de învățare, sau prin restartarea algoritmului cu alte valori inițiale ale parametrilor.

5^0 Numărul de neuroni din stratul de intrare se alege în funcție de "gradul de neliniaritate" care se constată la funcția φ din examinarea eşantioanelor (5.1.4). În alegerea arhitecturii rețelei (adică a numărului de neuroni de pe primul strat) pot părea două arhitecturi care trebuie verificate şi eliminate:

a. Subaproximarea (eng. *underfitting*): care are loc pentru un număr prea mic de neuroni. În cazul $n = p = 1$, graficul realizat de rețea nu prezintă un număr suficient de puncte de inflexiune. Pentru a asigura calitatea aproximării este suficient la J să fie foarte mic (nu va fi de aceeaşi valoare ca atunci când trec prin vecinătatea tuturor punctelor).
b. Supraaproximarea (eng. *overfitting*): se alege un număr mare de neuroni în cazul funcției scalare $n = p = 1$. Numărul excesiv de neuroni creează prea multe puncte de inflexiune, care nu ar putea exista pentru un fenomen real.

Alegerea adecvată a numărului de neuroni se bazează pe experiență şi, eventual, pe teste de factură încercare-eroare (eng. *trial and error*) efectuate pe problema concretă. ■