

2. PRINCIPALES TIPOS DE REDES NEURONALES

2.1 PERCEPTRÓN

2.1.1 Antecedentes. La primera red neuronal conocida, fue desarrollada en 1943 por Warren McCulloch y Walter Pitts; ésta consistía en una suma de las señales de entrada, multiplicadas por unos valores de pesos escogidos aleatoriamente. La entrada es comparada con un patrón preestablecido para determinar la salida de la red. Si en la comparación, la suma de las entradas multiplicadas por los pesos es mayor o igual que el patrón preestablecido la salida de la red es uno (1), en caso contrario la salida es cero (0). Al inicio del desarrollo de los sistemas de inteligencia artificial, se encontró gran similitud entre su comportamiento y el de los sistemas biológicos y en principio se creyó que este modelo podía computar cualquier función aritmética o lógica.

La red tipo Perceptrón fue inventada por el sicólogo Frank Rosenblatt en el año 1957. Su intención era ilustrar algunas propiedades fundamentales de los sistemas inteligentes en general, sin entrar en mayores detalles con respecto a condiciones específicas y desconocidas para organismos biológicos concretos.



Rosenblatt creía que la conectividad existente en las redes biológicas tiene un elevado porcentaje de aleatoriedad, por lo que se oponía al análisis de McCulloch Pitts en el cual se empleaba lógica simbólica para analizar estructuras bastante idealizadas. Rosenblatt opinaba que la herramienta de análisis más apropiada era la teoría de probabilidades, y esto lo llevó a una teoría de *separabilidad estadística* que utilizaba para caracterizar las propiedades más visibles de estas redes de interconexión ligeramente aleatorias.

El primer modelo de Perceptrón fue desarrollado en un ambiente biológico imitando el funcionamiento del ojo humano, el fotoperceptrón como se le llamó, era un dispositivo que respondía a señales ópticas; como se muestra en la figura 2.1.1 la luz incide en los puntos sensibles (**S**) de la estructura de la retina, cada punto S responde en forma todo-nada a la luz entrante, los impulsos generados por los puntos S se transmiten a las unidades de asociación (**A**) de la capa de asociación; cada unidad A está conectada a un conjunto aleatorio de puntos S, denominados conjunto fuente de la unidad A, y las conexiones pueden ser tanto excitatorias como inhibitorias. Las conexiones tienen los valores posibles +1, -1 y 0, cuando aparece un conjunto de estímulos en la retina, una unidad A se activa si la suma de sus entradas sobrepasa algún valor umbral; si la unidad esta activada, A produce una salida que se envía a la siguiente capa de unidades.



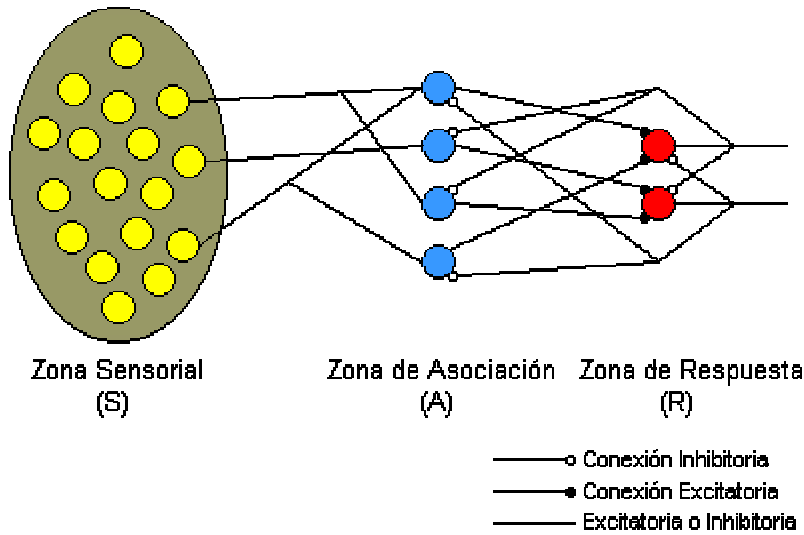


Figura 2.1.1 Modelo del Fotoperceptrón de Rosenblatt

De forma similar, las unidades A están conectadas a unidades de respuesta (R) dentro de la capa de respuesta y la conectividad vuelve a ser aleatorio entre capas, pero se añaden conexiones inhibitorias de realimentación procedentes de la capa de respuesta y que llegan a la capa de asociación, también hay conexiones inhibitorias entre las unidades R. Todo el esquema de conexiones se describe en forma general en un diagrama de Venn, para un Perceptrón sencillo con dos unidades de respuesta como el de la figura 2.1.2.

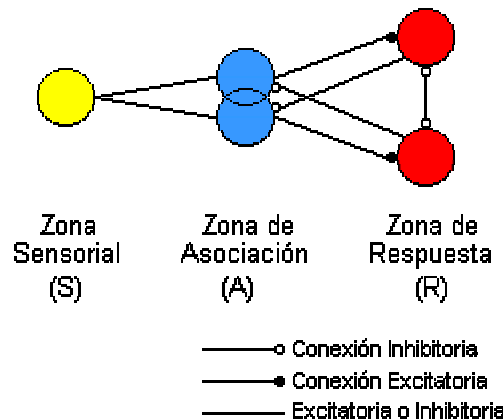


Figura 2.1.2 Esquema de conexiones de un Perceptrón sencillo



El Perceptrón era inicialmente un dispositivo de aprendizaje, en su configuración inicial no estaba en capacidad de distinguir patrones de entrada muy complejos, sin embargo mediante un proceso de aprendizaje era capaz de adquirir esta capacidad. En esencia, el entrenamiento implicaba un proceso de refuerzo mediante el cual la salida de las unidades **A** se incrementaba o se decrementaba dependiendo de si las unidades **A** contribuían o no a las respuestas correctas del Perceptrón para una entrada dada. Se aplicaba una entrada a la retina, y el estímulo se propagaba a través de las capas hasta que se activase una unidad de respuesta. Si se había activado la unidad de respuesta correcta, se incrementaba la salida de las unidades **A** que hubieran contribuido. Si se activaba una unidad **R** incorrecta, se hacía disminuir la salida de las unidades **A** que hubiesen contribuido.

Mediante estas investigaciones se pudo demostrar que el Perceptrón era capaz de clasificar patrones correctamente, en lo que Rosenblatt denominaba un entorno diferenciado, en el cual cada clase estaba formada por patrones similares. El Perceptrón también era capaz de responder de manera congruente frente a patrones aleatorios, pero su precisión iba disminuyendo a medida que aumentaba el número de patrones que intentaba aprender.

En 1969 Marvin Minsky y Seymour Papert publicaron su libro: "Perceptrons: An Introduction to Computational Geometry"[20], el cual para muchos significó el final de las redes neuronales. En él se presentaba un análisis detallado del Perceptrón, en



términos de sus capacidades y limitaciones, en especial en cuanto a las restricciones que existen para los problemas que una red tipo Perceptrón puede resolver; la mayor desventaja de este tipo de redes es su incapacidad para solucionar problemas que no sean linealmente separables.

Minsky y Papert se apartaban de la aproximación probabilística de Rosenblatt y volvían a las ideas de cálculo de predicados en el análisis del Perceptrón. Su idea de Perceptrón aparece en la figura 2.1.3

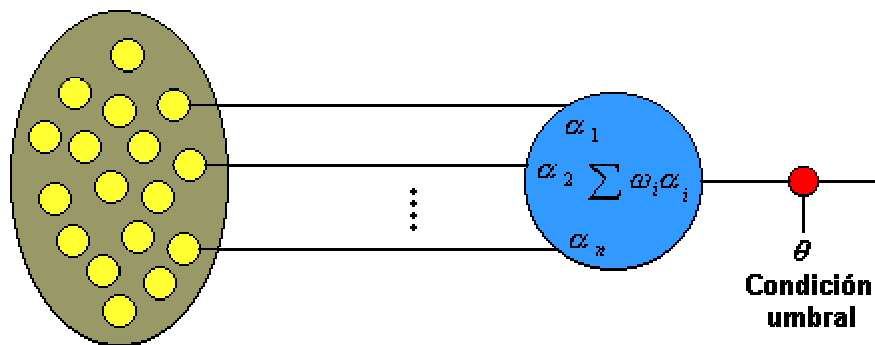


Figura 2.1.3 Perceptrón según Minsky y Papert

La estructura de un Perceptrón sencillo es similar a la del elemento general de procesamiento que se muestra en la figura 2.1.3; en la que se observa la adición de una condición umbral en la salida. Si la entrada neta, a esta condición es mayor que el valor umbral, la salida de la red es 1, en caso contrario es 0.

La función de salida de la red en la figura 2.1.3 es llamada función umbral o función de transferencia



$$f(\text{salida}) = \begin{cases} 1 & \text{si salida} \geq \theta \\ 0 & \text{si salida} < \theta \end{cases} \quad (2.1.1)$$

A pesar de esta limitación, el Perceptrón es aún hoy una red de gran importancia, pues con base en su estructura se han desarrollado otros modelos de red neuronal como la red Adaline y las redes multicapa.

2.1.2 Estructura de la red.

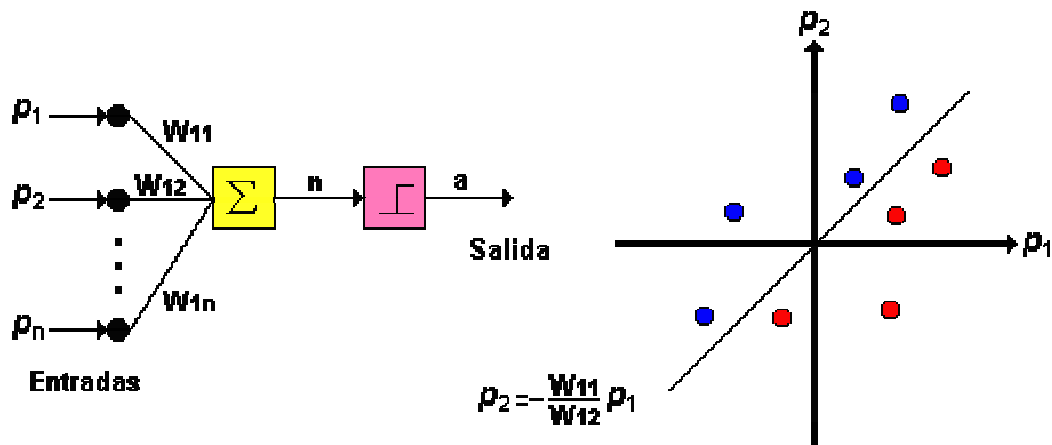


Fig. 2.1.4 Perceptrón

La única neurona de salida del Perceptrón realiza la suma ponderada de las entradas, resta el umbral y pasa el resultado a una función de transferencia de tipo escalón. La regla de decisión es responder +1 si el patrón presentado pertenece a



la clase A, o -1 si el patrón pertenece a la clase B (figura 2.1.4), la salida depende de la entrada neta ($n =$ suma de las entradas p_i ponderadas).

La red tipo Perceptrón emplea principalmente dos funciones de transferencia, *hardlim* con salidas 1, 0 o *hardlims* con salidas 1, -1; su uso depende del valor de salida que se espera para la red, es decir si la salida de la red es unipolar o bipolar; sin embargo la función *hardlims* es preferida sobre la *hardlim*, ya que el tener un cero multiplicando algunas de los valores resultantes del producto de las entradas por el vector de pesos, ocasiona que estos no se actualicen y que el aprendizaje sea más lento.

Una técnica utilizada para analizar el comportamiento de redes como el Perceptrón es presentar en un mapa las regiones de decisión creadas en el espacio multidimensional de entradas de la red, en estas regiones se visualiza qué patrones pertenecen a una clase y cuáles a otra, el Perceptrón separa las regiones por un hiperplano cuya ecuación queda determinada por los pesos de las conexiones y el valor umbral de la función de activación de la neurona, en este caso los valores de los pesos pueden fijarse o adaptarse empleando diferentes algoritmos de entrenamiento.

Para ilustrar el proceso computacional del Perceptrón consideremos la matriz de pesos en forma general.



$$W = \begin{bmatrix} W_{1,1} & W_{1,2} & \dots & W_{1,R} \\ W_{2,1} & W_{2,2} & \dots & W_{2,R} \\ W_{S,1} & W_{S,2} & \dots & W_{S,R} \end{bmatrix} \quad (2.1.2)$$

Los pesos para una neurona están representados por un vector compuesto de los elementos de la i -ésima fila de W

$$w = \begin{bmatrix} w_{i,1} \\ w_{i,2} \\ \vdots \\ w_{i,R} \end{bmatrix} \quad (2.1.3)$$

De esta forma y empleando la función de transferencia *hardlim* la salida de la neurona i de la capa de salida

$$a_i = \text{hardlim}(n_i) = \text{hardlim}(w_i^T p_i) \quad (2.1.4)$$

El Perceptrón, al constar de una sola capa de entrada y otra de salida con una única neurona, tiene una capacidad de representación bastante limitada, este modelo sólo es capaz de discriminar patrones muy sencillos, patrones linealmente separables (concepto que se estudiará en la sección 2.1.4), el caso más conocido es la imposibilidad del Perceptrón de representar la función OR EXCLUSIVA.



2.1.3 Regla de aprendizaje. El Perceptrón es un tipo de red de aprendizaje supervisado, es decir necesita conocer los valores esperados para cada una de las entradas presentadas; su comportamiento está definido por pares de esta forma:

$$\{p_1, t_1\}, \{p_2, t_2\}, \dots, \{p_Q, t_Q\} \quad (2.1.5)$$

Cuando p es aplicado a la red, la salida de la red es comparada con el valor esperado t , y la salida de la red esta determinada por:

$$a = f\left(\sum_i w_i p_i\right) = \text{hardlims}\left(\sum_i w_i p_i\right) \quad (2.1.6)$$

Los valores de los pesos determinan el funcionamiento de la red, estos valores se pueden fijar o adoptar utilizando diferentes algoritmos de entrenamiento de la red.

Como ejemplo de funcionamiento de una red neuronal tipo Perceptrón, se solucionará el problema de la función OR, para esta función la red debe ser capaz de devolver a partir de los cuatro patrones de entrada, a qué clase pertenece cada uno; es decir para el patrón 00 debe devolver la clase cero y para los restantes la clase 1, según la gráfica 2.1.5



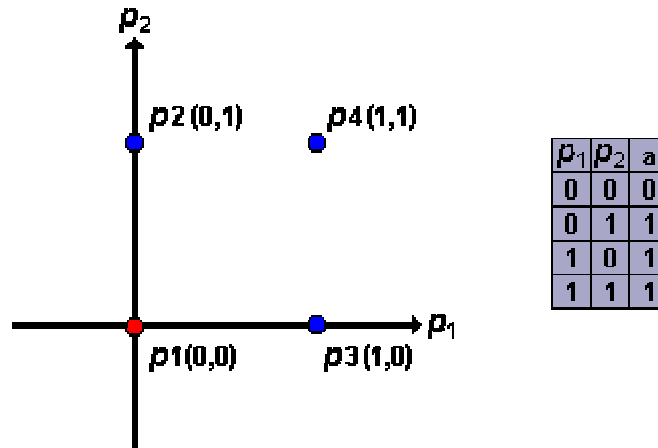


Figura 2.1.5 Función OR

Para este caso las entradas a la red serán valores binarios, la salida de la red está determinada por

$$a = \text{hardlims} \left(\sum_i w_i p_i \right) = \text{hardlims}(w_1 p_1 + w_2 p_2) \quad (2.1.7)$$

Si $w_1 p_1 + w_2 p_2$ es mayor que 0 la salida será 1, en caso contrario la salida será -1 (función escalón unitario). Como puede verse la sumatoria que se le pasa a cada parámetro (entrada total) a la función *hardlim* (función de salida o de transferencia) es la expresión matemática de una recta, donde w_1 y w_2 son variables y p_1 y p_2 son constantes. En la etapa de aprendizaje se irán variando los valores de los pesos obteniendo distintas rectas, lo que se pretende al modificar los pesos de las conexiones es encontrar una recta que divida el plano en dos espacios de las dos clases de valores de entrada, concretamente para la función OR se deben separar los valores 01, 10, y 11 del valor 00; la red



Perceptrón que realiza esta tarea y la gráfica característica pueden observarse en la figura 2.1.6 allí puede verse como las posibles rectas pasarán por el origen de coordenadas, por lo que la entrada 00 quedará sobre la propia recta.

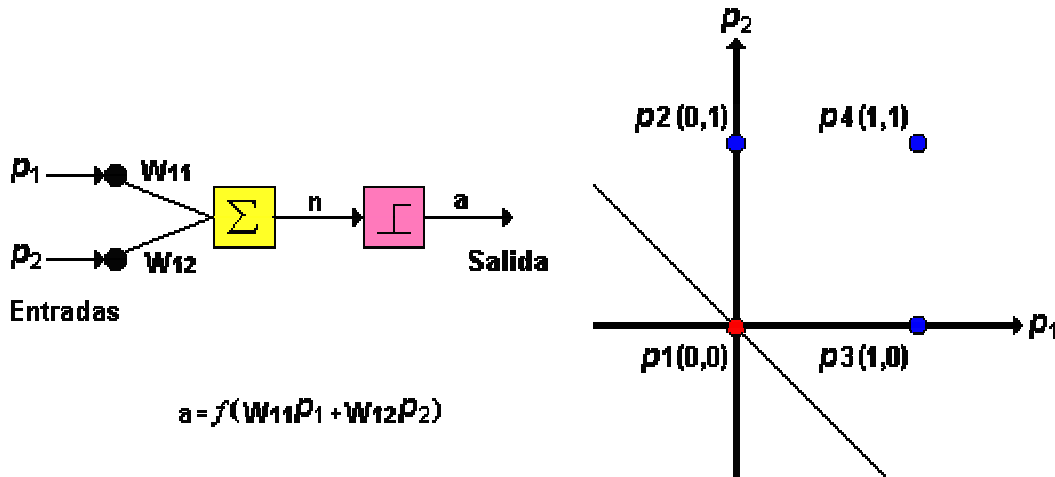


Figura 2.1.6 Perceptrón aplicado a la función OR

Se aplicará este método para resolver también el problema de la función AND, el cual se describe en la siguiente figura

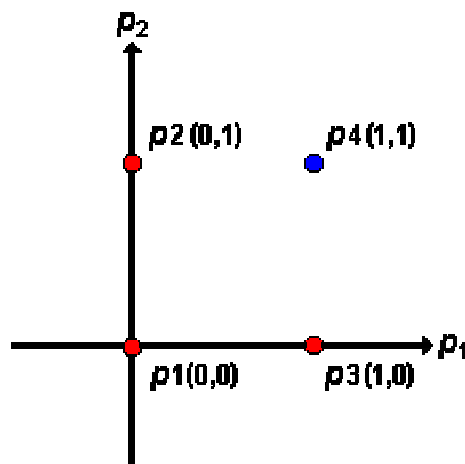


Figura 2.1.7 Espacio de salida de una compuerta AND

Analizando el comportamiento de la AND se llega a la conclusión de que es imposible que una recta que pase por el origen, separe los valores 00,01 y 10 del



valor 11, por lo que se hace necesario introducir un término independiente para realizar esta tarea, a este término se le da el nombre de ganancia y se representa por la letra b , al cual por lo general se le asigna un valor inicial de 1 y se ajusta durante la etapa de aprendizaje de la red; este nuevo término permite desplazar la recta del origen de coordenadas dando una solución para el caso de la función AND y ampliando el número de soluciones de la función OR

Ahora la salida de la neurona esta dada por

$$a = \text{hardlims}(w_1 p_1 + w_2 p_2 + b) \tag{2.1.8}$$

Las soluciones obtenidas para la función AND y la OR, se ven en la figura 2.1.8

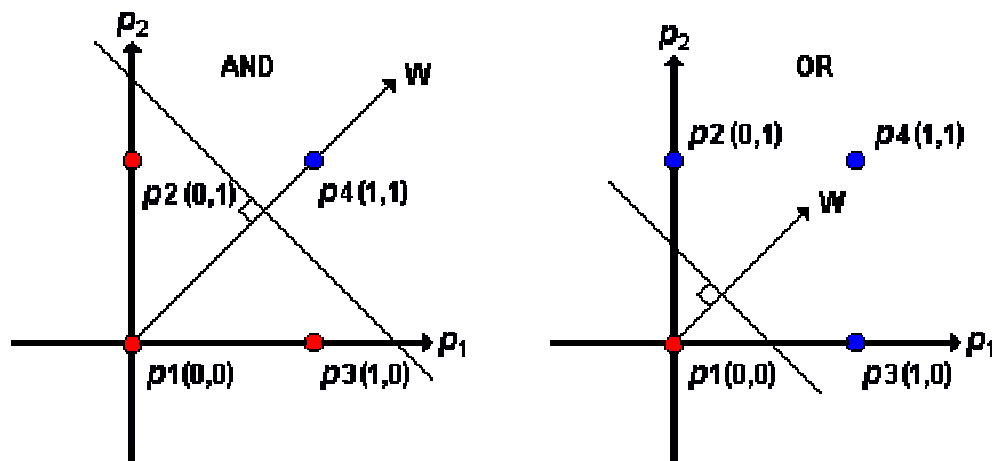


Figura 2.1.8 Solución para una función AND y una OR

En el proceso de entrenamiento el Perceptrón se expone a un conjunto de patrones de entrada y los pesos de la red son ajustados de forma que al final de



entrenamiento se obtengan salidas esperadas para cada unos de esos patrones de entrada.

El algoritmo de entrenamiento del Perceptrón puede resumirse en los siguientes pasos:

- 1 Se inicializa la matriz de pesos y el valor de la ganancia, por lo general se asignan valores aleatorios a cada uno de los pesos w_i y al valor b
- 2 Se presenta el primer patrón a la red, junto con la salida esperada en forma de pares entrada/salida

- 3 Se calcula la salida de la red por medio de

$$a = f(w_1 p_1 + w_2 p_2 + b) \quad (2.1.9)$$

donde f puede ser la función *hardlim* o *hardlims*

- 4 Cuando la red no retorna la salida correcta, es necesario alterar el valor de los pesos, tratando de llevarlo hasta p y así aumentar las posibilidades de que la clasificación sea correcta, una posibilidad es adicionar p a w haciendo que el vector w apunte en la dirección de p , y de esta forma después de repetidas presentaciones de p a la red, w se aproximará asintóticamente a p ; este es el procedimiento adoptado para la regla de aprendizaje del Perceptrón.



El proceso de aprendizaje del Perceptrón puede definirse en tres reglas, las cuales cubren la totalidad de combinaciones de salidas y sus correspondientes valores esperados. Estas reglas utilizando la función de transferencia *hardlim*, se expresan como sigue:

$$\text{Si } t = 1 \text{ y } a = 0, \text{ entonces } {}_1w^{\text{nuevo}} = {}_1w^{\text{anterior}} + p \quad (2.1.10)$$

$$\text{Si } t = 0 \text{ y } a = 1, \text{ entonces } {}_1w^{\text{nuevo}} = {}_1w^{\text{anterior}} - p \quad (2.1.11)$$

$$\text{Si } t = a, \text{ entonces } {}_1w^{\text{nuevo}} = {}_1w^{\text{anterior}} \quad (2.1.12)$$

Las tres condiciones anteriores pueden ser escritas en forma compacta y generalizarse para la utilización de las funciones de transferencia *hardlim* o *hardlims*, generalización que es posible introduciendo el error en las reglas de aprendizaje del Perceptrón:

$$e = t - a \quad (2.1.13)$$

Por lo tanto:

$$\text{Si } e = 1, \text{ entonces } {}_1w^{\text{nuevo}} = {}_1w^{\text{viejo}} + p \quad (2.1.14)$$

$$\text{Si } e = -1, \text{ entonces } {}_1w^{\text{nuevo}} = {}_1w^{\text{anterior}} - p \quad (2.1.15)$$

$$\text{Si } e = 0, \text{ entonces } {}_1w^{\text{nuevo}} = {}_1w^{\text{anterior}} \quad (2.1.16)$$

En una sola expresión la ley puede resumirse así:



$${}_1w^{nuevo} = {}_1w^{anterior} + ep = {}_1w^{anterior} + (t - a)p \quad (2.1.17)$$

Y extendiendo la ley a las ganancias

$$b^{nueva} = b^{anterior} + e \quad (2.1.18)$$

Para ilustrar la regla de aprendizaje del Perceptrón, se dará solución al problema de clasificación de patrones ilustrado en la figura 2.1.9

$$P_1 = \begin{bmatrix} 2 \\ 1 \end{bmatrix} \quad t_1 = 1, \quad P_2 = \begin{bmatrix} 0 \\ -1 \end{bmatrix} \quad t_2 = 1, \quad P_3 = \begin{bmatrix} -2 \\ 1 \end{bmatrix} \quad t_3 = -1, \quad P_4 = \begin{bmatrix} 0 \\ 2 \end{bmatrix} \quad t_4 = -1$$

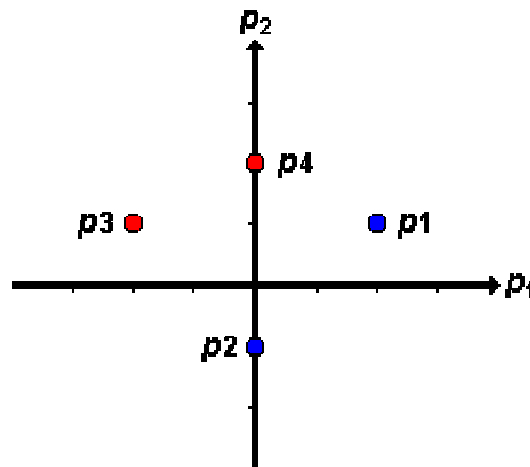


Figura 2.1.9 Patrones de entrenamiento

En este caso las salidas toman valores bipolares de 1 o -1, por lo tanto la función de transferencia a utilizar será *hardlims*. Según la dimensiones de los patrones de entrenamiento la red debe contener dos entradas y una salida.



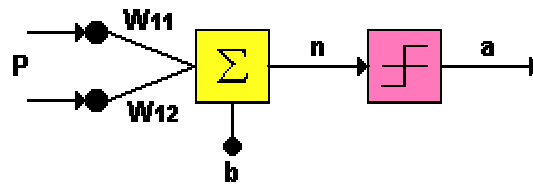


Figura 2.1.10 Red Perceptrón que resolverá el problema de clasificación de patrones

Para decidir si una red tipo Perceptrón puede aplicarse al problema de interés, se debe comprobar si el problema es linealmente separable, esto puede determinarse gráficamente de la figura 2.1.9, en donde se observa que existe un gran número de líneas rectas que pueden separar los patrones de una categoría de los patrones de la otra, el siguiente paso es asumir arbitrariamente los valores para los pesos y ganancias iniciales de entrada a la red; el proceso terminará cuando se hayan obtenido los pesos y ganancias finales que permitan a la red clasificar correctamente todos los patrones presentados.

Los valores iniciales asignados aleatoriamente a los parámetros de la red son:

$$W = [-0.7 \quad 0.2] \quad b = [0.5]$$

Con base en el procedimiento descrito anteriormente, el proceso de aprendizaje de la red es el siguiente:

- Iteración 0



La red clasificará los patrones de entrenamiento según la característica de decisión mostrada en la figura 2.1.11, la cual depende de los valores de los pesos y ganancias iniciales.

Interceptos con los ejes: $\frac{-b}{W_{11}} = -2.5$ $\frac{-b}{W_{21}} = 0.71$

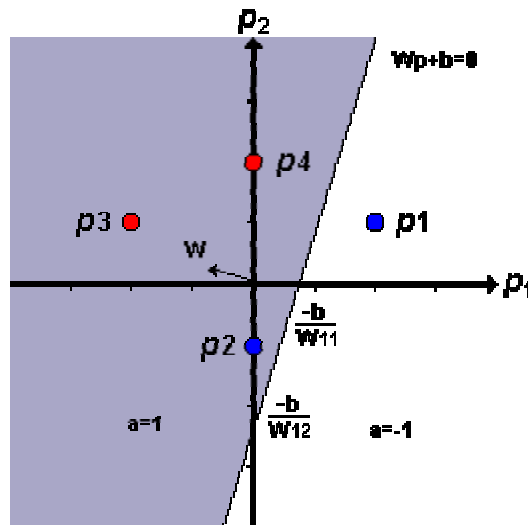


Figura 2.1.11 Clasificación de los patrones de acuerdo a la iteración 0

Como puede verse, la característica de decisión es ortogonal al vector de pesos W . La red clasifica incorrectamente los patrones $p1$, $p3$ y $p4$; en esta iteración; a continuación presentamos a la red el patrón de entrenamiento $p1$.

- Iteración 1

$$W^0 = [-0.7 \quad 0.2] \quad b^0 = [0.5]$$

$$a = \text{hardlims} \left([-0.7 \quad 0.2] \begin{bmatrix} 2 \\ 1 \end{bmatrix} + [0.5] \right) \quad a = \text{hardlims}(-0.7) = -1$$

$$e = t - a = 1 - (-1) = 2$$



De la iteración 0 $p1$ estaba mal clasificado, la actualización de pesos permite que este patrón sea clasificado correctamente.

$$W^1 = W^0 + ep^T \quad W^1 = [-0.7 \quad 0.2] + 2[2 \quad 1] = [3.3 \quad 2.2]$$

$$b^1 = b^0 + e \quad b^1 = 0.5 + 2 = 2.5$$

La iteración 1 lleva a la característica de decisión de la figura 2.1.12

Interceptos con los ejes: $\frac{-b}{W_{11}} = -0.75$ $\frac{-b}{W_{21}} = -1.13$

Como se observa el patrón de entrenamiento $p1$ ha sido clasificado correctamente, y casualmente los patrones $p2$ y $p3$ fueron correctamente ubicados, pues aún no han sido presentados a la red.

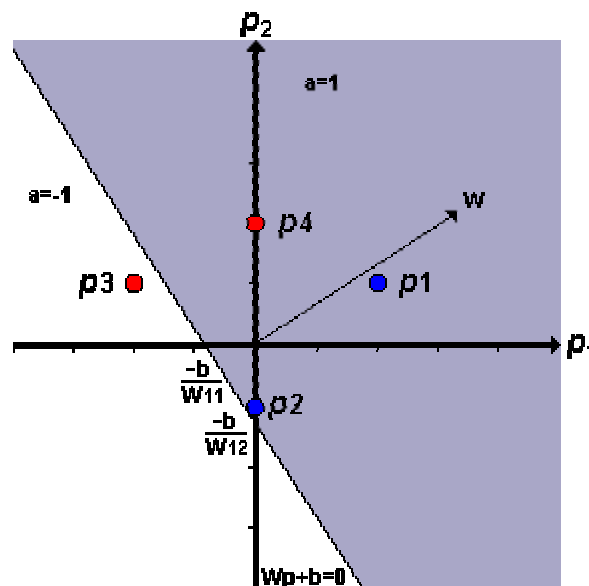


Figura 2.1.12 Característica de decisión de la iteración 1



- Iteración 2

Se presenta p_2 a la red, y es clasificado correctamente, como se observó gráficamente

$$W^1 = [3.3 \quad 2.2] \quad b^1 = [2.5]$$
$$a = \text{hardlims} \left([3.3 \quad 2.2] \begin{bmatrix} 0 \\ -1 \end{bmatrix} + [2.5] \right) \quad a = \text{hardlims}(0.3) = 1$$
$$e = t - a = 1 - (1) = 0$$

Este patrón ha sido clasificado correctamente y por lo tanto no hay actualización del set de entrenamiento

$$W^2 = W^1 + ep^T \quad W^2 = [3.3 \quad 2.2] + 0[0 \quad -1] = [3.3 \quad 2.2]$$
$$b^2 = b^1 + e \quad b^2 = 2.5 + 0 = 2.5$$

- Iteración 3

Se presenta p_3 a la red y es clasificado correctamente, como se observó gráficamente

$$W^2 = [3.3 \quad 2.2] \quad b^2 = [2.5]$$



$$a = \text{hardlims}\left(\left[3.3 \quad 2.2\right]\begin{bmatrix} -2 \\ 1 \end{bmatrix} + [2.5]\right) \quad a = \text{hardlims}(-1.9) = -1$$

$$e = t - a = -1 - (-1) = 0$$

Como se esperaba, no hubo error en la clasificación de este patrón, y esto lleva a que no haya actualización de los pesos de la red

$$W^3 = W^2 + eP^T \quad W^3 = [3.3 \quad 2.2] + 0[-2 \quad 1] = [3.3 \quad 2.2]$$

$$b^3 = b^2 + e \quad b^3 = 2.5 + 0 = 2.5$$

- Iteración 4

Se presenta a la red $p4$,

$$W^3 = [3.3 \quad 2.2] \quad b^3 = [2.5]$$

$$a = \text{hardlims}\left(\left[3.3 \quad 2.2\right]\begin{bmatrix} 0 \\ 2 \end{bmatrix} + [2.5]\right) \quad a = \text{hardlims}(6.9) = 1$$

$$e = t - a = -1 - (1) = -2$$

La red ha clasificado incorrectamente este patrón y por lo tanto deben modificarse pesos y ganancias

$$W^4 = W^3 + ep^T \quad W^4 = [3.3 \quad 2.2] - 2[0 \quad 2] = [3.3 \quad -1.8]$$

$$b^4 = b^3 + e \quad b^4 = 2.5 - 2 = 0.5$$



En esta iteración la red se comportara de acuerdo a la característica de decisión de la figura 2.1.13

Interceptos con los ejes: $\frac{-b}{W_{11}} = -0.15$ $\frac{-b}{W_{21}} = 0.27$

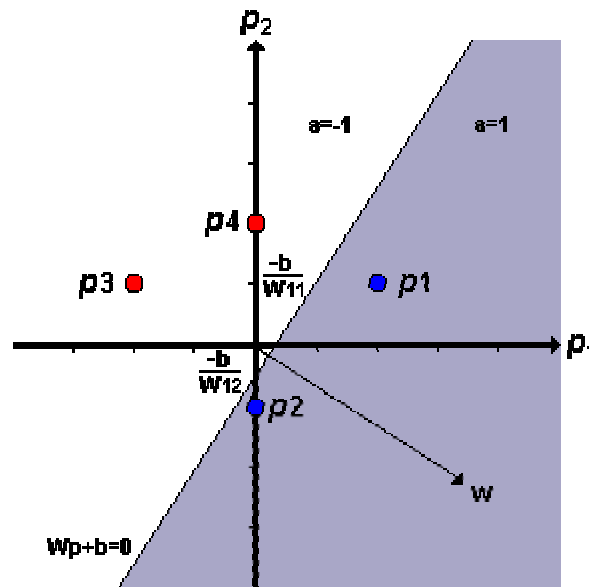


Figura 2.1.13 Característica de decisión final

De la figura 2.1.13 se observa que la red ha clasificado correctamente los patrones de entrenamiento, después de entrenada la red con los pesos y ganancias finales, cualquier otro valor de entrada será clasificado según la característica de decisión mostrada.

Es de importancia notar que en este caso los patrones de entrada se encuentran en dos dimensiones y por lo tanto es fácil determinar gráficamente cuando han sido clasificados correctamente, en el caso que los patrones se encuentren en tres dimensiones esta visualización se dificulta y en el caso de que los patrones sean de orden superior la visualización resulta imposible; para estos casos se debe



comprobar matemáticamente que el error correspondiente a cada patrón de entrenamiento para los pesos finales es nulo.

2.1.4 Limitación de la red Perceptrón.

En la sección 2.1.1, se planteó la restricción que existe para los tipos de problemas que una red Perceptrón puede solucionar, como se dijo esta red puede resolver solamente problemas que sean linealmente separables, esto es problemas cuyas salidas estén clasificadas en dos categorías diferentes y que permitan que su espacio de entrada sea dividido en estas dos regiones por medio de un hiperplano de características similares a la ecuación del Perceptrón, es decir

$$wp + b = 0 \tag{2.1.19}$$

Ejemplos de problemas de este tipo son las funciones lógicas OR y AND estudiadas anteriormente; para ilustrar más claramente que significa que un problema sea linealmente separable se analizará un caso que no lo sea, el caso de la compuerta XOR, el cual se visualiza en la figura 2.1.14

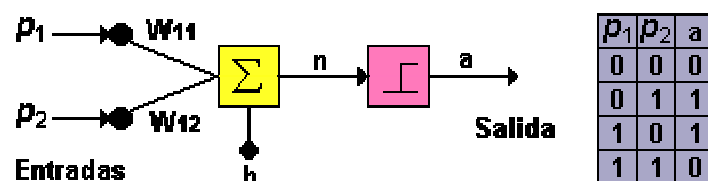


Figura 2.1.14 Compuerta XOR



Se pretende que para los valores de entrada 00 y 11 se devuelva la clase 0 y para los patrones 01 y 10 la clase 1. Como puede verse de la figura 2.1.15 el problema radica en que no existe ninguna línea recta que separe los patrones de una clase de los de la otra

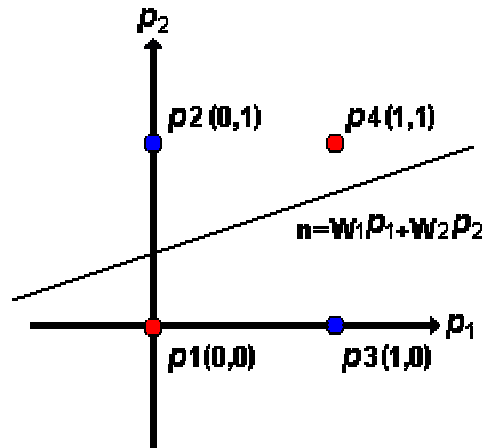


Figura 2.1.15 Plano formado por el problema de la XOR

Los cuatro puntos en la figura son las posibles entradas de la red; la línea divide el plano en dos regiones, por lo que se podría clasificar los puntos de una región como pertenecientes a la clase que posee salida 1 (puntos azules) y los de la otra región como pertenecientes a la clase que posee salida 0 (puntos rojos), sin embargo no hay ninguna forma de posicionar la línea para que los puntos correctos para cada clase se encuentren en la misma región. El problema de la compuerta XOR no es linealmente separable y una red tipo Perceptrón no está en capacidad de clasificar correctamente los patrones de esta función, debido a esta limitación del Perceptrón y a su amplia publicación en el libro de Minsky y Papert, el estudio de las redes neuronales se estancó durante casi 20 años.



El proceso para determinar si un problema es linealmente separable o no, se realiza gráficamente sin problema, cuando los patrones de entrada generan un espacio de dos dimensiones, como en el caso de las funciones AND, OR o de la XOR; sin embargo, esta visualización se dificulta cuando el conjunto de patrones de entrada es de tres dimensiones, y resulta imposible de observar gráficamente cuando los patrones de entrada son de dimensiones superiores; en este caso se requiere plantear condiciones de desigualdad que permitan comprobar la separabilidad lineal de los patrones, esto se realiza con base en la ecuación de salida del Perceptrón

$Wp + b \geq 0$, para aquellos patrones cuya salida deseada sea 1

$Wp + b < 0$, para aquellos patrones cuya salida deseada sea 0

En el caso de la XOR, teniendo en cuenta los valores de la tabla al lado derecho de la figura 2.1.14, estas desigualdades se expresan así:

$$0 * W_{1,1} + 0 * W_{2,1} + b < 0 \quad (p_1) \qquad 1 * W_{1,1} + 0 * W_{2,1} + b \geq 0 \quad (p_3)$$

$$0 * W_{1,1} + 1 * W_{2,1} + b \geq 0 \quad (p_2) \qquad 1 * W_{1,1} + 1 * W_{2,1} + b < 0 \quad (p_4)$$

Si no hay contradicción en las desigualdades anteriores, el problema es linealmente separable. Como se observa de las desigualdades 2, 3 y 4, es imposible que $W_{2,1} \geq 0$, $W_{1,1} \geq 0$ y que su suma sea menor que cero, esta es una



forma alternativa de comprobar que el problema de la XOR no es linealmente separable. El aporte de esta técnica se aprecia mejor para problemas cuyo espacio de entrada sea de dimensiones mayores.

La solución al problema de clasificación de patrones de la función XOR se encontraría fácilmente si se descompone el espacio en tres regiones: una región pertenecería a una de las clases de salida y las otras dos pertenecen a la segunda clase, así que si en lugar de utilizar únicamente una neurona de salida se utilizaran dos, se obtendrían dos rectas por lo que podrían delimitarse tres zonas; para poder elegir entre una zona u otra de las tres, es necesario utilizar otra capa con una neurona cuyas entradas serán las salidas de las neuronas anteriores; las dos zonas o regiones que contienen los puntos (0,0) y (1,1) se asocian a una salida nula de la red y la zona central se asocia a la salida con valor 1, de esta forma es posible encontrar una solución al problema de la función XOR, por tanto se ha de utilizar una red de tres neuronas, distribuidas en dos capas para solucionar este problema.

En la figura 2.1.16 se observa un esquema de lo que sería una red Perceptrón multicapa, con los valores de pesos y ganancias que clasifican correctamente los patrones de la compuerta XOR



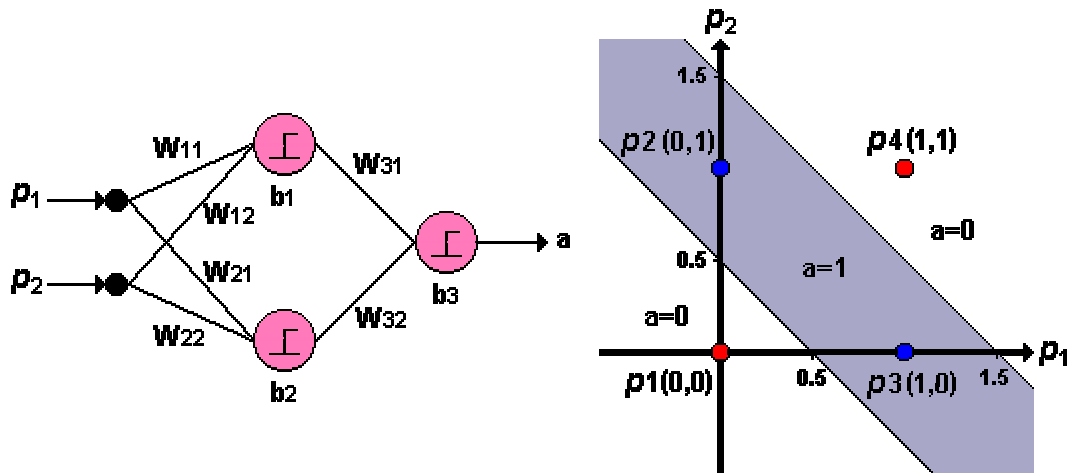


Figura 2.1.16 Perceptrón multicapa para la XOR

Los valores de la matriz de pesos y del vector de ganancias son:

$$\begin{array}{ll}
 w_{11}=1 & w_{12}=1 \\
 w_{21}=1 & w_{22}=1 \\
 w_{31}=1 & w_{32}=-1.5 \\
 b_1=0.5 & b_2=1.5 & b_3=0.5
 \end{array}$$

2.1.5 Perceptrón multicapa. En el problema de la función XOR se explicó como un Perceptrón multicapa había sido implementado para hallar una solución, el esquema general de un Perceptrón multicapa puede encontrarse generalizando la figura 2.4.1 a una red con múltiples entradas y que incluya una entrada adicional representada por la ganancia b , este esquema general se ve en la figura 2.1.17 en



donde se notan las conexiones entre sus nodos de entrada y las neuronas de salida.

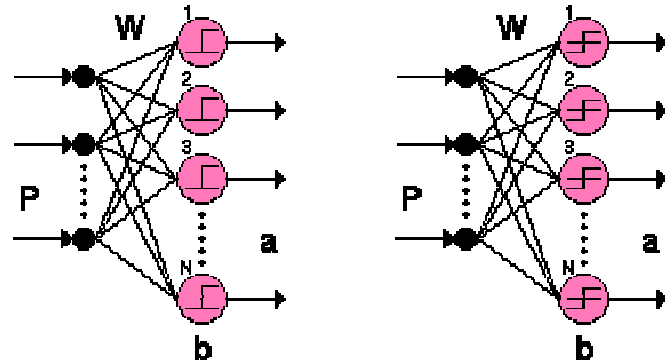


Figura 2.1.17 Conexiones del Perceptrón

Un Perceptrón multicapa es una red con alimentación hacia delante, compuesta de varias capas de neuronas entre la entrada y la salida de la misma, esta red permite establecer regiones de decisión mucho más complejas que las de dos semiplanos, como lo hace el Perceptrón de un solo nivel.

Un esquema simplificado del modelo del Perceptrón de la figura 2.1.17 se observa en la figura 2.1.18

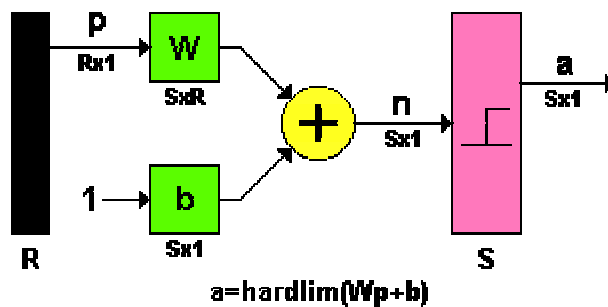


Figura 2.1.18 Notación compacta para la red tipo Perceptrón



La salida de la red está dada por:

$$a = \text{hardlim}(\mathbf{W} * \mathbf{p} + b) \quad (2.1.20)$$

Donde

W: Matriz de pesos asignada a cada una de las entradas de la red de dimensiones $S \times R$, con S igual al número de neuronas, y R la dimensión del vector de entrada

p: Vector de entradas a la red de dimensiones $R \times 1$

b: Vector de ganancias de la red de dimensiones $S \times 1$

Las capacidades del Perceptrón multicapa con dos y tres capas y con una única neurona en la capa de salida se muestran en la figura 2.1.19 extraída del libro de Hilera J y Martínez V [11]. En la segunda columna se muestra el tipo de región de decisión que se puede formar con cada una de las configuraciones, en la siguiente se indica el tipo de región que se formaría para el problema de la XOR, en las dos últimas columnas se muestran las regiones formadas para resolver el problema de clases mezcladas y las formas más generales para cada uno de los casos.



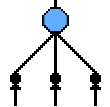
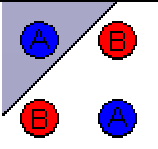
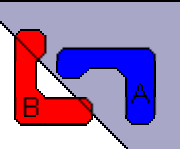
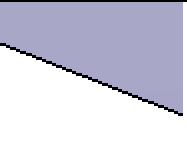
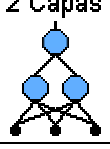
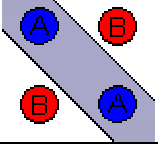
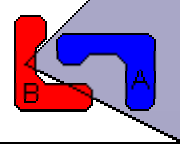
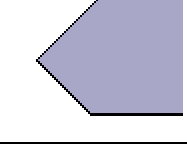
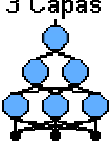
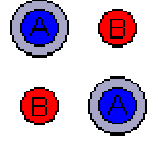
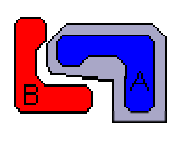
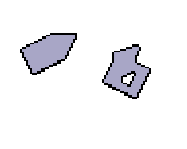
Estructura	Regiones de Decisión	Problema de la XOR	Clases con Regiones Mezcladas	Formas de Regiones más Generales
<p>1 Capa</p> 	Medio Plano Limitado por un Hiperplano			
<p>2 Capas</p> 	Regiones Cerradas o Convexas			
<p>3 Capas</p> 	Complejidad Arbitraria Limitada por el Número de Neuronas			

Figura 2.1.19 Distintas formas de las regiones generadas por un Perceptrón multicapa

El Perceptrón básico sólo puede establecer dos regiones separadas por una frontera lineal en el espacio de entrada de los patrones; un Perceptrón con dos capas, puede formar cualquier región convexa en este espacio. Las regiones convexas se forman mediante la intersección de regiones compuestas por cada neurona de la segunda capa, cada uno de estos elementos se comporta como un Perceptrón simple, activándose su salida para los patrones de un lado del hiperplano, si el valor de los pesos de las conexiones entre las neuronas de la segunda capa y una neurona del nivel de salida son todos igual a 1, y la función de salida es de tipo *hardlim*, la salida de la red se activará sólo si las salidas de todos los nodos de la segunda capa están activos, esto equivale a ejecutar la función lógica AND en el nodo de salida, resultando una región de decisión



intersección de todos los semiplanos formados en el nivel anterior. La región de decisión resultante de la intersección será una región convexa con un número de lados a lo sumo igual al número de neuronas de la segunda capa.

A partir de este análisis surge el interrogante respecto a los criterios de selección para las neuronas de las capas ocultas de una red multicapa, este número en general debe ser lo suficientemente grande como para que se forme una región compleja que pueda resolver el problema, sin embargo no debe ser muy grande pues la estimación de los pesos puede ser no confiable para el conjunto de los patrones de entrada disponibles. Hasta el momento no hay un criterio establecido para determinar la configuración de la red y esto depende más bien de la experiencia del diseñador.

La regla de aprendizaje del Perceptrón para una red multicapa es una generalización de las ecuaciones (2.1.17) y (2.1.18)

$${}_1\mathbf{W}^{nuevo} = {}_1\mathbf{W}^{anterior} + \mathbf{ep}^T \quad (2.1.21)$$

$$\mathbf{b}^{nueva} = \mathbf{b}^{anterior} + \mathbf{e} \quad (2.1.22)$$

