Trabajo Fin de Máster
Máster en Sistemas de Energía Térmica

MAXIMIZACIÓN DE LA PRODUCCIÓN DE HIDRÓGENO MEDIANTE TÉCNICAS DE MACHINE LEARNING

Autor: Eduardo Gómez Piñuela
Tutor: David Velázquez Alonso

Dpto. de Termotecnia – Ingeniería Energética
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2020

tsc
Departamento de Teoría de la Señal y Comunicaciones

Trabajo Fin de Máster
Máster en Sistemas de Energía Térmica

MAXIMIZACIÓN DE LA PRODUCCIÓN DE HIDRÓGENO MEDIANTE TÉCNICAS DE MACHINE LEARNING

Autor:
Eduardo Gómez Piñuela

Tutor:
David Velázquez Alonso
Profesor titular

Dpto. de Termotecnia – Ingeniería Energética
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
Sevilla, 2020

ÍNDICE

CAPÍTULO 1: INTRODUCCIÓN AL PROYECTO 5
1.1. Objetivos 6
1.2. Alcance del proyecto 7
1.3. Reseña histórica 8
1.4. El hidrógeno como vector energético 9
1.5. Propiedades del hidrógeno 9
1.6. Producción de hidrógeno a partir de Hidrocarburos. 11
1.7. Proceso de obtención de hidrógeno a partir de gas natural mediante reformado de vapor 13
1.8. Proceso de reformado de vapor: El reformador 16
1.9. Catalizador y funcionamiento del reformador. 17
1.10. Descripción del proceso de obtención de hidrógeno ..... **¡Error! Marcador no definido.**
CAPÍTULO 2: ESTADO DEL ARTE DE LAS REDES NEURONALES 19
2.1. Objetivos 20
2.2. Acerca de las redes neuronales. 20
2.3. Reseñas históricas. 22
2.4. El modelo biológico. 23
2.4.1. Estructura de la neurona 23
2.5. Definiciones de una Red Neuronal Artificial (RNA). 24
2.6. Ventajas que ofrecen las redes neuronales. 25
2.6.1. Aprendizaje adaptativo. 25
2.6.2. Auto-organización. 26
2.6.3. Tolerancia a fallos. 26
2.6.4. Operación en tiempo real. 27
2.6.5. Fácil inserción dentro de la tecnología existente. 27
2.7. Elementos Básicos. 28
2.7.1. Elementos básicos que componen una red neuronal. 28
2.7.2. Niveles o capas de una red neuronal. 29
2.7.3. Función de entrada (input function) 29
2.7.4. Función de activación (activation function). 31
2.7.5. Función de salida (output function). 32
2.1. Aprendizaje, Validación y Codificación. 33
2.1.1. Mecanismos de aprendizaje. 33
2.1.2. Elección del conjunto inicial de pesos. 35
2.1.3. Detención del proceso de aprendizaje. 35

2.1.4. Validación de la red neuronal. 36
2.2. Fenómenos que se deben evitar en el aprendizaje 36
2.2.1. Overtraining 36
2.2.2. Overfitting 38
2.3. Principales Topologías de redes neuronales. 39
2.3.1. Redes monocapa. 39
2.3.2. Redes multicapa. 39
2.3.3. Redes Neuronales con conexiones hacia adelante. 40
2.3.4. El Perceptron. 40
2.3.5. EL Perceptron multicapa 42
2.3.6. Redes ADALINE y MADELINE. 44
2.3.7. Red Backpropagation 46
2.4. Redes Recurrentes. 47
2.5. Redes de Hopfield. 47
2.6. Redes Feedforward. 48
2.7. Redes de Elman. 50
2.8. Red de Jordan 52
2.9. Red recurrente completamente conectada 53
2.10. Aplicaciones de las redes neuronales. 56
CAPÍTULO 3: OBTENCIÓN DEL MODELO 59
3.3. Objetivos 60
3.4. Datos de partida 60
3.5. Pre-procesado de los datos 63
3.5.1. Reducción de la dimensionalidad 63
3.6. Estudio de la complejidad del modelo 66
3.6.1. Estudio de la arquitectura. Número de neuronas. 67
3.6.2. Número de iteraciones o epochs 73
3.7. Resultados y selección del modelo óptimo. 74
CAPÍTULO 4: ANÁLISIS DE SENSIBILIDAD 75
4.3. Objetivos 76
4.4. Introducción a los análisis de sensibilidad 76
4.5. Metodología aplicada 77
4.5.1. Análisis de sensibilidad en el rango completo 78
4.5.2. Análisis de sensibilidad +-2% de la media de cada variable 1
4.6. Resultados del análisis de sensibilidad 1
CAPÍTULO 5: MAXIMIZACIÓN DEL HIDRÓGENO 2

5.1. Objetivos 3
5.2. Introducción a los métodos de optimización 3
5.3. Tipos de métodos de optimización 5
5.3.1. Simulated Annealing (recocido simulado) 6
5.3.2. Parámetros para la configuración del SA 8
5.4. Metodología aplicada 10
5.1. Resultados de la optimización 13
BIBLIOGRAFÍA Y REFERENCIAS 15

MÁSTER EN SISTEMAS DE ENERGÍA TÉRMICA

CAPÍTULO 1:

INTRODUCCIÓN AL PROYECTO

1.1. Objetivos

En la actualidad el hidrógeno se ha convertido en una de las alternativas más eficientes, térmicamente hablando, para ir sustituir a los combustibles fósiles y reducir así el gran impacto ambiental que estos conllevan. Es por ello, que una de las tendencias en muchos sistemas de combustión, para su uso generalmente en sistemas de propulsión, es el hidrógeno. En la actualidad, son cada vez más los dispositivos que utilizan este combustible, por diferentes medios, para la generación térmica y su transformación en trabajo útil.

A raíz de todo esto, diferentes tecnologías son las que se han ido desarrollando para la obtención del hidrógeno, ya que de forma aislada no se encuentra en el medio natural.

El objetivo de este proyecto, y con motivo de lo anterior, es el realizar un modelo de un proceso de refino para la obtención de hidrógeno mediante redes neuronales. Este proceso consiste en un reformado de gas natural mediante vapor para la separación de la cadena de hidrocarburo el hidrógeno.

Posteriormente, y en base a este modelo, se realizará un análisis de sensibilidad con el que visualizar que variables tienen mayor influencia en el modelo.

Por último, se realizará un estudio de optimización de las variables del proceso, para conseguir optimizar un registro de estas que conlleven a un aumento de la producción de hidrógeno a la salida del proceso.

1.2. Alcance del proyecto

El proyecto se centrará en los aspectos ya comentados de la siguiente forma:

- Modelado mediante redes neuronales:

  Obtención del modelo óptimo para describir de la forma más precisa posible el proceso de obtención de hidrógeno utilizando un modelo matemático basado en redes neuronales artificiales para predecir la salida de este gas.

- Análisis de sensibilidad:

  Se estudiarán las variables implicadas en el modelo viendo la importancia de estas sobre el mismo, y como afectan al modelo.

  De este modo, se estudiarán que variables afectan en mayor medida al modelo y de qué manera afecta su variación con respecto a la producción de hidrógeno.

- Maximización del hidrógeno:

  Se realizará un estudio de los registros de las variables de entrada, mediante un algoritmo basado en el recocido simulado para maximizar el valor de producción de hidrógeno buscando aquella configuración óptima de las variables que causan su aumento de la producción, siempre dentro de los límites teóricos.

1.3. Reseña histórica

En la historia de la Química, la primera referencia al hidrógeno como tal, aparece con la identificación de dos gases diferentes como formando parte del agua, por el inglés Henry Cavendish en 1766 [1]. Cavendish le da el nombre de aire inflamable. Más tarde, Antoine Lavoisier le da el nombre por el que lo conocemos, hidrógeno, generador de agua.

Antes de que finalizara el siglo XVIII, el hidrógeno encontró su primera aplicación práctica, como ocurre frecuentemente, por el ejército francés para globos de reconocimiento. Más de un siglo después, Alemania lo empleó en sus dirigibles para cruzar el océano Atlántico e incluso como combustible para la propulsión de los llamados zeppelines. Esta empresa concluyó después de la catástrofe del Hinderburg en 1937.

Más tarde, antes y después de la segunda guerra mundial, el hidrógeno se empleó como combustible de motores de vehículos de todo tipo, incluidos locomotoras y submarinos, pero sin gran éxito. Y ello a pesar de la predicción de Julio Verne en su novela "La isla misteriosa" de que algún día el agua, bajo la forma de sus componentes hidrógeno y oxígeno, serviría como fuente inagotable de energía.

El auge del carbón en el siglo XIX y del petróleo en el siglo XX eliminaron toda posibilidad del uso masivo del hidrógeno [1]. Solamente la industria química, primero con la producción de fertilizantes derivados del amoníaco y después con la necesidad de hidrógeno para eliminar azufre y otros componentes de las gasolinas y gasóleos de locomoción y como complemento en las síntesis de productos derivados del petróleo, mantuvieron en el mundo una producción sustancial del hidrógeno.

Más adelante, ya en la segunda mitad del siglo XX, la preocupación por el ambiente y los posibles efectos sobre el clima, el comienzo de la era espacial y la inquietud sobre el agotamiento de los combustibles fósiles ha traído de nuevo un gran impulso, sobre todo, de la industria automovilista por el empleo en gran escala del hidrógeno, lo que se ha llamado, erróneamente a mi modo de ver, la economía del hidrógeno.

El interés de las industrias aeroespacial y automovilista por el hidrógeno se debe a la llamada pila de combustible, donde el hidrógeno puede quemarse con el oxígeno, transformándose la energía de la combustión en electricidad. Este proceso fue ya descubierto en 1839 por el galés William R. Grove, pero su desarrollo comenzó en la década de los 1960 por la NASA para producir electricidad y agua en algunas de sus misiones espaciales. Actualmente, un gran número de prototipos de las principales marcas de automóviles y autobuses ensayan pilas de combustible de tipos y combustibles diversos.

1.4. El hidrógeno como vector energético

Uno de los objetivos de la sociedad actual [2], en todos sus sectores, es la mejora sostenible de la calidad de vida y, en concreto, la búsqueda de nuevos sistemas energéticos que satisfagan a precios asequibles las demandas de esa sociedad (reducir los agentes contaminantes y asegurar el suministro energético, principalmente).

Como una alternativa a los tradicionales vectores energéticos basados en combustibles fósiles, se plantea ahora la posibilidad de emplear un nuevo vector energético, el hidrógeno, del que se dice que es limpio, renovable y eficiente. Además, empleado conjuntamente con pilas de combustible, el hidrógeno puede producir calor y electricidad teniendo como único residuo agua pura.

Sin embargo, hay que destacar que el hidrógeno no es una fuente de energía primaria, como lo son el carbón o el gas natural; es un vector energético que en la actualidad se puede producir utilizando los sistemas energéticos tradicionales (actualmente la mayor parte del hidrógeno se produce a partir del gas natural).

Así, la producción de hidrógeno a partir de estas fuentes permite una generación distribuida [2], acorde con las economías locales, siempre a través del método más limpio y adecuado en cada caso; pero, además, libera a las regiones y a los países de la necesidad de importar combustibles fósiles y sus derivados, ya que será posible en muchos casos producir localmente el hidrógeno que se necesite.

1.5. Propiedades del hidrógeno

Es el elemento más ligero y abundante en el Universo, y es el combustible de las estrellas y, evidentemente, del Sol, por lo que la energía que éste nos envía es la base de todos los procesos fisicoquímicos y biológicos que tienen lugar en la Tierra.

El hidrógeno está presente en forma molecular o iónica, pero a pesar de su abundancia no está disponible para nosotros. El posible yacimiento más próximo está en Júpiter, inaccesible por el momento. En la Tierra, desgraciadamente, el hidrógeno está combinado en su mayor parte formando agua, no hay hidrógeno libre y la corteza terrestre está formada principalmente por oxígeno, silicio, aluminio y otros elementos menores.

Deben destacarse algunas propiedades que son importantes y deberán ser tenidas en cuenta en muchos de los aspectos que se verán a continuación.

Los primeros son el bajo punto de ebullición y la proximidad a la temperatura crítica, como aparecen en la siguiente tabla.

Los segundos son las bajas densidades del gas y del líquido.

La tercera es el contenido en deuterio, que puede ser una de las bases de la energía nuclear de fusión, lo que no se tratará en esta presentación.

TABLA 1: Algunas propiedades del hidrógeno

- Incoloro. Gas normalmente; sólido,cúbico.
- Punto de fusión: -259,1º C.
- Punto de ebullición: -252,7º C.
- Temperatura crítica: -239,8º C.
- Densidad del líquido: 0,0709 g/cm^3.
- Densidad del gas: 0,0899 Kg/m^3.
- Solubilidad en el agua:

  a 0º C: 2,1 cm^3
  a 80º C: 0,85 cm^3
  } en 100 partes

- Isótopos y abundancias

H-1	99,985%
H-2 (deuterio)	0,015%
H-3 (radiactivo, período 12,26 años)	-

Tabla 1. Propiedades del Hidrógeno.

De las propiedades antes citadas se pueden deducir las dificultades de almacenar el hidrógeno, de transformar el gas en el líquido y de los problemas del almacenamiento del líquido.

- El valor calorífico y el contenido de energía por unidad de masa del hidrogeno es mayor que el de cualquier otro combustible. Esto se debe a su baja densidad. Contiene 120 MJ/Kg mientras que la gasolina contiene 51 MJ/Kg.

- Al combinarse con el oxígeno puede producir electricidad mediante un proceso electroquímico que sobrepasa la eficiencia del ciclo de Carnot por tratarse de un proceso químico y no termodinámico como el que usan los motores de combustión. Este proceso sucedería en la celda o pila de hidrogeno y la electricidad producida alimentaría un motor eléctrico. El ´único residuo de este proceso es H2O.

- Se puede usar en motores de combustión interna muy parecidos a los motores convencionales, siendo menos contaminantes que los hidrocarburos.

- Es un vector de energía respetuoso con el medio ambiente, que trae consigo la reducción de la contaminación. Al contrario que los combustibles fósiles, a los que pretende sustituir, que al quemarse emiten compuestos de azufre, hidrocarburos y CO2, el hidrógeno solo emite agua cuando se combina con el oxígeno en la celda de hidrógeno. Contiene 0.16 MJ/L frente a los 35 MJ/L que contiene la gasolina.

- Los márgenes de explosión para el hidrogeno en aire son mayores (más seguros) que los del metano y similares a los de la gasolina, esto se debe en parte a su baja densidad y a la gran facilidad que tiene para difundirse.

1.6. Producción de hidrógeno a partir de Hidrocarburos.

La producción del hidrógeno consiste en, básicamente, la extracción desde los compuestos fuente tales como: el agua, los hidrocarburos y la biomasa a través de operaciones físico-químicas que necesitan cierta cantidad de energía para su liberación. Actualmente, existen métodos de obtención de hidrógeno más desarrollados que otros. Unos se encuentran en etapa industrial y otros en etapa experimental.

Para la producción de hidrógeno existen distintas alternativas, tanto en la elección del proceso (reformado con vapor, oxidación parcial, gasificación, electrolisis) como de la materia prima (carbón, gas natural, hidrocarburos, agua, alcoholes).

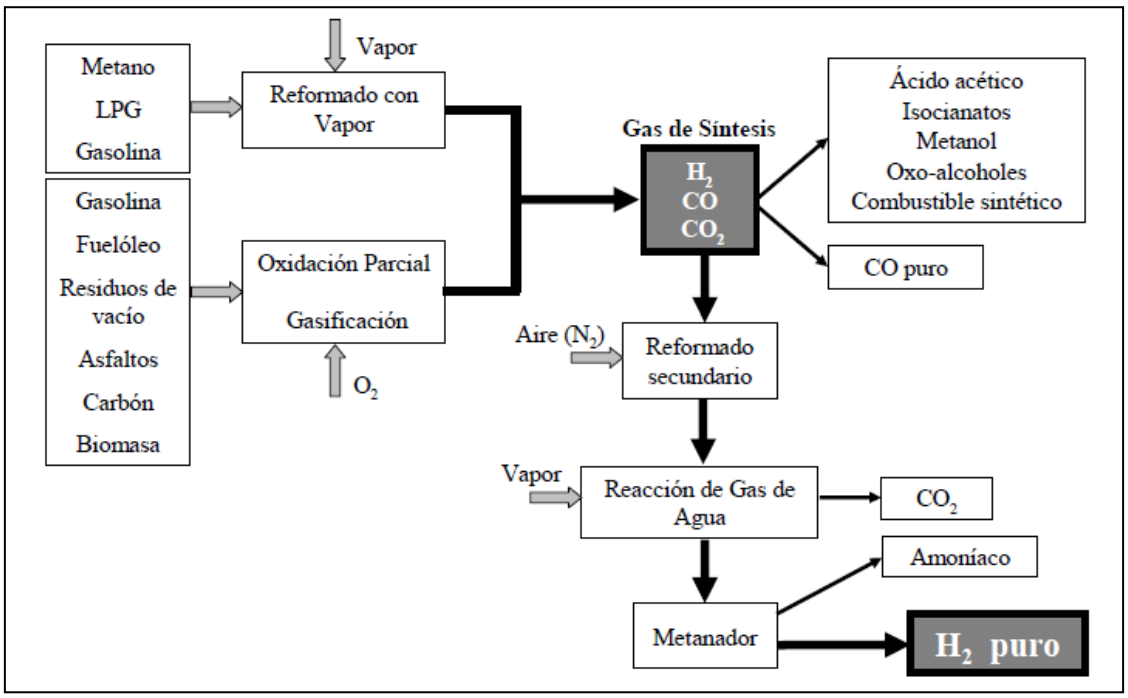


Figura 1. Tecnologías actuales de producción de hidrógeno.

Las técnicas tradicionales para la producción de hidrógeno son:

- **Electrolisis del agua**: La electrólisis del agua es un método de obtención de hidrógeno no muy usado debido a que resulta al menos dos veces más caro que el hidrógeno obtenido por el reformado del gas natural. Sin embargo, es de un gran interés ya que es un método bastante limpio y que podría ser

usado como complemento para la energía producida por centrales solares y eólicas, y además, produce un hidrógeno de elevada pureza.

- **Gasificación:** Consiste en un proceso de combustión incompleta a partir de combustibles sólidos, o líquidos para obtener gas de síntesis (40-65 % de CO y 25-37 % de H2). En el caso del carbón, éste es tratado con vapor de agua de manera que se obtiene monóxido de carbono e hidrógeno. El problema es que se trata de una reacción muy endotérmica (ΔH = 119 kJ/mol)) y necesita temperaturas muy elevadas para que se produzca, por lo que se añade una pequeña cantidad de oxígeno de manera que se produce su oxidación parcial que libera energía:

$$C + H_2O \rightarrow CO + H_2$$

$$C + 1/2\ O_2 \rightarrow CO$$

Para obtener un buen rendimiento de la mezcla gaseosa resultante, es necesaria una temperatura mínima de 700 ºC, por lo que este proceso se suele realizar entre 800 y 1500 ºC. Por tanto, se trata un proceso con un elevado consumo energético, aunque no tan elevado como en el caso de la electrolisis.

- **Reformado con vapor:** El proceso de reformado con vapor funciona preferentemente sobre hidrocarburos ligeros, es decir, con puntos de ebullición inferiores a 200°C, que deben ser limpiados de toda impureza (azufre, arsénico, halógenos, entre otros) antes de su procesamiento. La carga a utilizar puede ser gas natural, metano o gas licuado de petróleo. Básicamente, el proceso consiste en la exposición de la carga a vapor de agua a alta temperatura y a una presión moderada. Como resultado de la reacción química se obtiene fundamentalmente hidrógeno, monóxido y dióxido de carbono y otros compuestos. Estas reacciones ocurren generalmente a temperaturas comprendidas entre los 800 y 950°C con presiones entre 1,5 a 3,0 MPa.

$$CH_4 + H_2O + calor \rightarrow CO + 3H_2$$
$$CO + H_2O \rightarrow CO_2 + H_2 + calor$$

- **Oxidación parcial**: El hidrocarburo se mezcla con oxígeno o vapor de agua, se calientan, se mezclan y queman dentro del reactor a unos 1100°C. La eficiencia de este método es relativamente elevada, pero menos eficiente que el reformado por vapor debido a que opera a elevadas temperaturas y existe el problema de la recuperación del calor. Tiene unos costes de inversión y operación elevados si se opera con oxígeno puro. Sin embargo, el uso de aire en vez de vapor, y debido a que es una reacción exotérmica, hace que no se necesite aporte de energía externa durante el proceso.

$$CH_4 + \tfrac{1}{2} O_2 \rightarrow CO + 2H_2 + calor$$

- **Reformado autotérmico**: Es una combinación de la oxidación parcial y la de reformado por vapor de forma conjunta. Con la correcta combinación de alimentación de entrada, el calor de reacción de la oxidación parcial es suficiente para que se dé la reacción de reformado por vapor. El hidrocarburo reacciona con el vapor de agua y con el aire para producir un gas rico en hidrógeno. Los inconvenientes que presenta este método tienen que ver con el tamaño de todo el equipo en conjunto debido a que requieren un sistema de recuperación del calor.

En la siguiente tabla se muestra un resumen de algunas de las tecnologías actuales para la producción de hidrógeno.

Proceso	Ventajas	Inconvenientes
Electrolisis del agua	Mínimas emisiones de CO_2 Hidrógeno de gran pureza Tecnología probada	Elevados costes de energéticos
Gasificación	Desarrollo de la tecnología Materias primas abundantes Bajo coste de las materias primas	Grandes emisiones de CO_2 Eficacia baja Gran infraestructura
Oxidación parcial	Tecnología utilizada muy avanzada	Emisiones de CO_2 Elevados costes de inversión Gran infraestrucutra
Reformado con vapor	Elevada eficiencia Perfectamente desarrollado a gran escala Hidrógeno a bajo coste	Emisiones de CO_2 según materia prima Gran infraestructura Unidades a pequeña escala no comerciales

Figura 2. Comparativa de los principales procesos de prod. de hidrógeno.

1.7. Proceso de obtención de hidrógeno a partir de gas natural mediante reformado de vapor.

Hoy en día, el hidrógeno se produce principalmente a partir de reformado de metano o gas natural con vapor de agua (Steam Methane Reforming- SMR). (15-40 bares) en reactores tubulares rellenos de un catalizador de Ni [3]. Con el fin de suministrar el calor necesario para la reacción endotérmica, los reactores tubulares se encuentran dentro de un horno o caldera donde se quema gas natural o una corriente gaseosa residual con aire. Como consecuencia de esta combustión se producen cantidades notables de emisiones de CO2. Un esquema del proceso SMR se muestra en la siguiente figura:

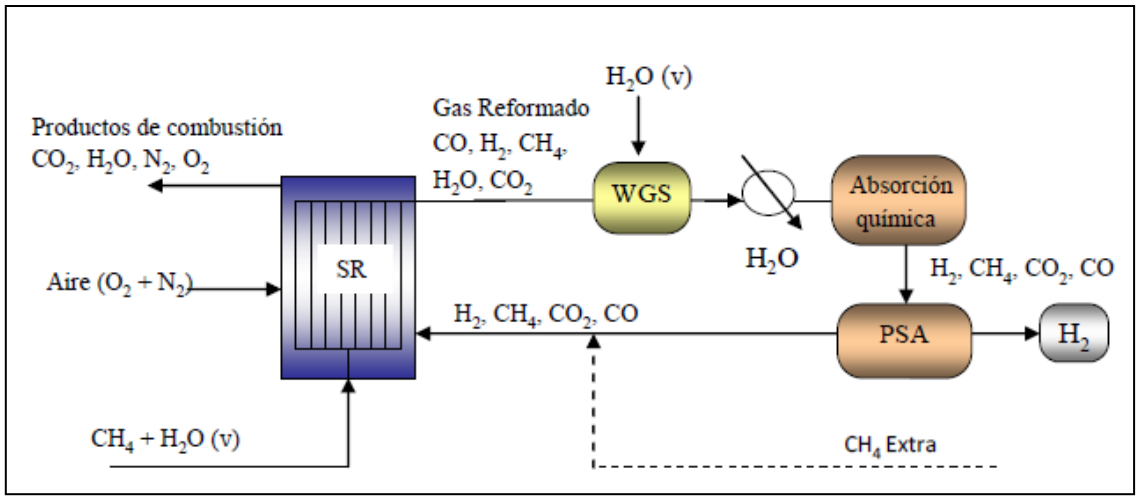


Figura 3. Esquema del reformado de metano con vapor de agua (SMR)

$$CH_4 + H_2O \leftrightarrow CO + 3\,H_2\,, \qquad \Delta H1200K = 226\ kJ/mol$$

En este proceso el gas reformado o gas de síntesis producido en la reacción de reformado se enfría y se lleva a un reactor de intercambio WGS para maximizar la producción de H2 por la reacción:

$$CO + H_2O \leftrightarrow CO_2 + H_2$$

Normalmente la reacción WGS tiene lugar en dos reactores [3]. En el primer reactor la reacción se lleva a cabo a temperaturas relativamente altas, de 400-550 ºC (HT-WGS), y es catalizada por un catalizador de Fe para reducir la concentración de CO hasta un 2-5%. En el segundo reactor (LT-WGS) la temperatura de reacción es menor, 200 ºC-350 ºC, y se utiliza un catalizador de cobre para reducir la concentración de CO hasta un 0.2%.

Los catalizadores más empleados en este proceso a nivel industrial son los basados en níquel (habitualmente, Ni/Al_2O_3), aunque presentan el inconveniente de la deposición de carbón. En general, los metales nobles muestran mejor actividad catalítica y resistencia a la desactivación, pero industrialmente se usan los catalizadores basados en Ni debido a su menor costo y mayor disponibilidad.

La formación de carbón puede causar severos problemas operacionales en un gran número de procesos catalíticos industriales, por lo que debe ser inhibida con el objeto de evitar el bloqueo del lecho catalítico, la aparición de puntos calientes, el envenenamiento y la destrucción textural del catalizador.

A continuación, y tras condensar el agua, está la etapa de purificación de H2 que se puede realizar de dos maneras diferentes dependiendo del grado de pureza requerido:

- Absorción química. Es el proceso clásico. En esta opción el grado de pureza del H2 no es muy alto (≈ 97-99 %). Se utiliza un proceso de absorción

química, por ejemplo, con monoetanolamina (MEA), para retener y separar el CO2 del resto de componentes de la corriente gaseosa, seguido de una etapa de metanación para eliminar el CO y CO2 que puedan quedar en la corriente. Así, en esta opción se obtiene una corriente de H2 y otros componentes minoritarios (CH4, CO, CO2) y una corriente pura de CO2 que se puede almacenar.

- Pressure swing adsorption (PSA). En plantas donde se requiere un alto grado de pureza del H2 se utiliza una unidad de adsorción (Pressure swing adsorption-PSA) donde todos los gases menos el H2 son adsorbidos. Como material adsorbente se usa carbón activado, alúmina o zeolitas. El H2 a la salida de la unidad PSA puede llegar a tener una pureza de 99.999 %. La concentración de CO2 en la corriente de salida de la unidad PSA que se obtiene durante la desorción por bajada de presión es del 40-50%. Esta corriente, que además contiene H2, CH4 y CO, se quema en el horno del reformador para generar parte del calor necesario para las reacciones endotérmicas de reformado.

- Combinación de ambas tecnologías. Mediante la combinación de las dos tecnologías anteriores se pueden obtener corrientes prácticamente puras de CO2 e H2.

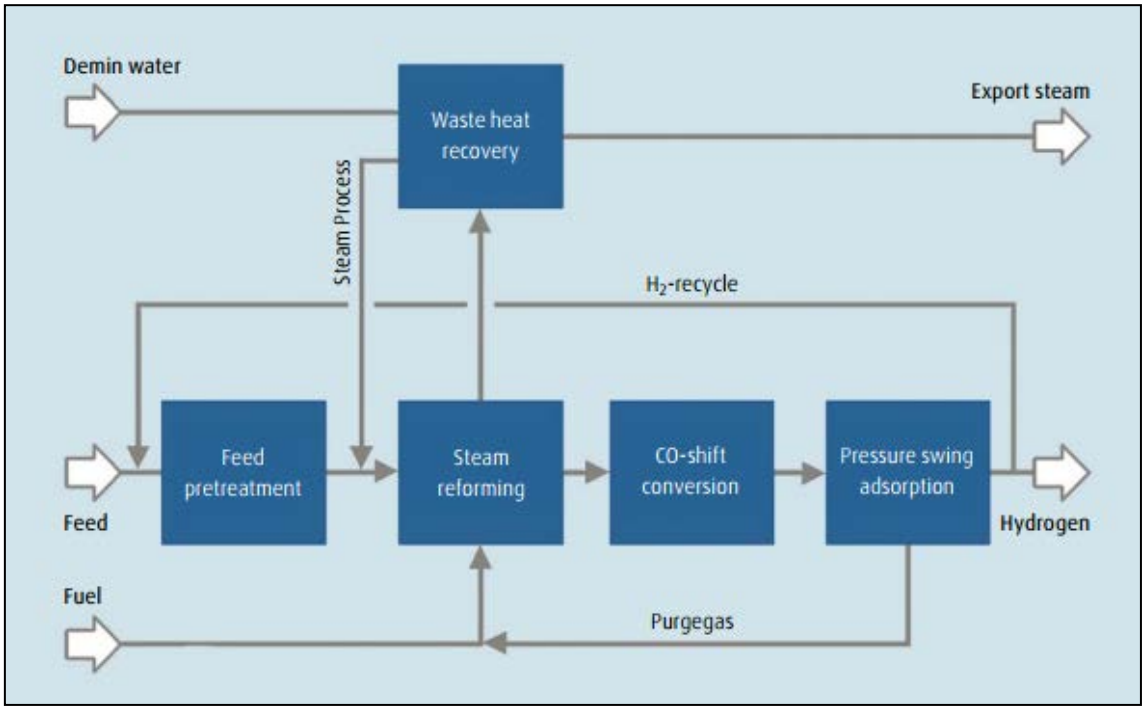


Figura 4. Esquema de reformado de gas natural con vapor

En este proceso, para evitar que el CO2 sea emitido a la atmósfera habría que capturar el CO2 producido en el horno del reformador utilizando alguna de las tecnologías de captura en post-combustión (absorción con aminas). Al escaparse del objetivo de este proyecto, no se desarrollará el tema.

1.8. Proceso de reformado de vapor: El reformador.

La reacción de reformado con vapor es fuertemente endotérmica [4]. Sin embargo, el calor global de la reacción podría ser positivo, nulo o negativo dependiendo de las condiciones de procesamiento.

A bajas relaciones vapor-carbono y temperaturas bajas, el reformado con vapor de hidrocarburos superiores seguido de reacciones exotérmicas de WGS y metanación podrían dar un gas rico en metano y un calor global de reacción ligeramente positivo.

En la industria, las reacciones de reformado se realizan habitualmente en un horno calentado en presencia de un catalizador de níquel. El catalizador se coloca en tubos de acero inoxidable con diámetros externos de 100 a 150 mm y de 10 a 13 m de largo. Temperaturas de entrada son de unos 450–650ºC y la temperatura de salida de 800–950ºC. Existen diversos ordenamientos de tubos y quemador.

Su capacidad es el orden de 300000 Nm3 de H2/h. El horno consta de una sección radiante de tipo caja que incluye quemadores y una sección de convección para recuperar el calor residual de los gases de salida que va a la sección radiante. El problema de la deposición de carbono se ha resuelto introduciendo un pre-reformador adiabático.

Todos los hidrocarburos superiores se convierten en el pre-reformador a unos 650ºC reduciendo así el tamaño del reformador tubular. El pre-reformador también ayuda a la flexibilidad de la alimentación. El proceso de reformado con vapor en un reformador tubular tiene una serie de limitaciones.

La termodinámica obliga a temperaturas altas de salida para lograr conversiones grandes de metano. Sin embargo, los catalizadores son potencialmente activos a temperaturas por debajo de 400ºC. Para obviar esta limitación se usan membranas selectivas de hidrógeno en el lecho catalítico.

El hidrógeno producido de esta manera es de baja presión luego habrá que presurizarlo para ciertas aplicaciones y hace al proceso no económico a no ser que el precio de la electricidad sea bajo o si la aplicación del hidrógeno es para una pila de combustible o como combustible a baja presión.

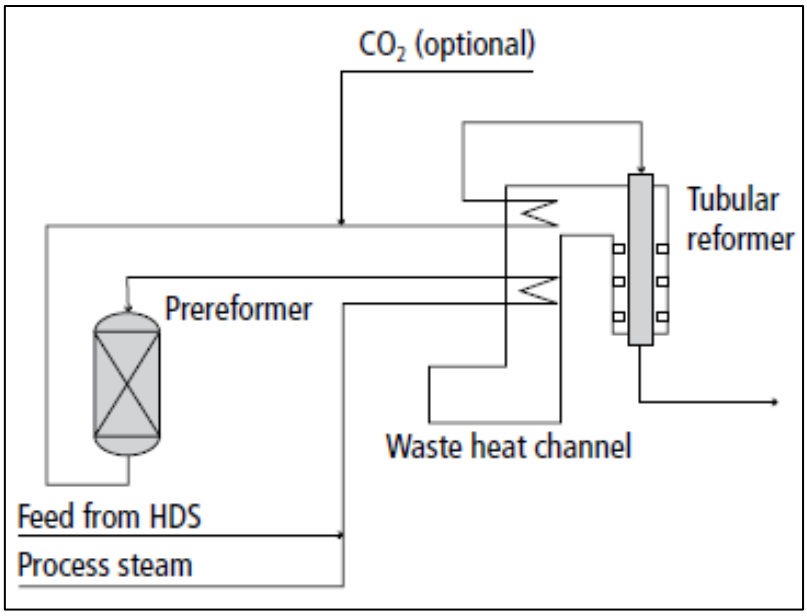


Figura 5. Diagrama de flujo de un proceso con reformador tubular y pre-reformador.

Se han sugerido diversos esquemas para obviar el concepto de reactor tubular. Así, hay esquemas de precalentamiento del gas antes de entrar al reformador adiabático y generalmente esto requiere muchas etapas de precalentamiento debido a la endotermicidad de la reacción de reformado. Otra posibilidad es usar un catalizador recirculante con un lecho para reacción y otro para calentamiento.

Pero para el reformado la velocidad de recirculación tiene que ser muy elevada y el catalizador en los intercambiadores de calor corriente abajo puede dar metanación. Lechos fluidizados con altas velocidades de transferencia de calor y conducciones de calor son otras alternativas.

1.9. Catalizador y funcionamiento del reformador.

Normalmente el catalizador de reformado con vapor contiene níquel, cobalto y metales nobles son también activos [5]. Las propiedades del catalizador dependen de las condiciones de severidad del proceso: 450–950ºC y hasta 30 bar.

La actividad catalítica depende del área activa de níquel, pero la actividad catalítica no es el factor limitante. En la práctica, la utilización de la actividad catalítica intrínseca (el factor de efectividad) es menor del 10% debido a restricciones de transporte.

El factor de efectividad es tan bajo que la actividad es proporcional al área superficial externa del catalizador. Una actividad catalítica alta no es importante en el complejo acoplamiento de la reacción y la transferencia de calor:

Calor transferido = calor de reacción + calor sensible

En la figura 6 se muestran datos de simulación de un reformador tubular. El calor requerido para reformar un hidrocarburo líquido es inferior que para metano, luego se necesitará menos aporte de calor al reformador en condiciones de operación similares. Los hidrocarburos superiores son también más reactivos que el metano, siendo los aromáticos los que tienen menor actividad acercándose a la del metano [5].

La conversión de los hidrocarburos superiores sobre níquel se produce por una adsorción irreversible en la superficie dejando la superficie solamente componentes C1. Las reacciones de reformado podrían estar acompañadas de craqueo térmico (pirólisis) a partir de 600–650ºC.

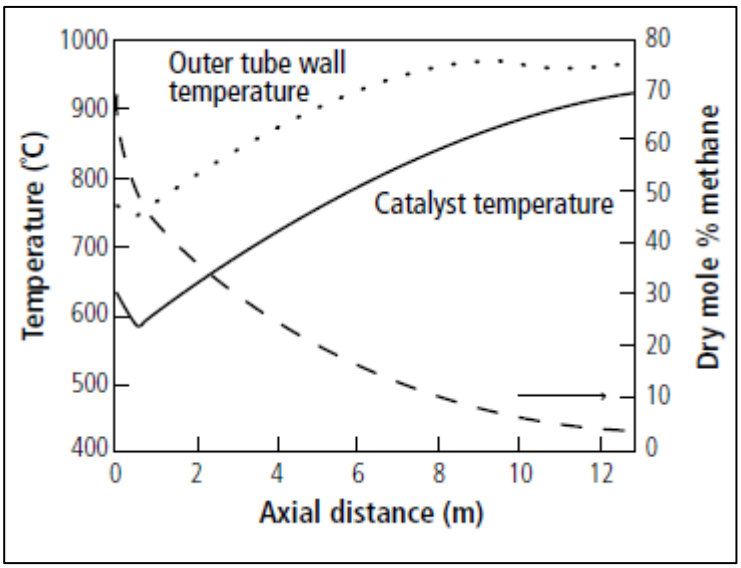


Figura 6. Perfiles de un reformador tubular de una planta de hidrógeno con pre-reformador.

MÁSTER EN SISTEMAS DE ENERGÍA TÉRMICA

CAPÍTULO 2:

ESTADO DEL ARTE DE LAS REDES NEURONALES

2.1. Objetivos

En este capítulo se desarrollará de modo introductorio, una descripción detallada sobre las redes neuronales artificiales, siendo esta la herramienta principal aplicada en el presente proyecto para la predicción de datos.

Es por ello, por lo que se expondrá un "pequeño" estado del arte sobre las redes neuronas artificiales, mostrando así desde los orígenes de esta herramienta matemática, así como sus generalidades y tipologías, configuración, su importancia en el mundo actual y principales aplicaciones.

De este modo, se dará a conocer mejor la gran utilidad de esta herramienta para su uso en el presente proyecto con el objetivo de estimar y maximizar la producción de H2.

2.2. Acerca de las redes neuronales.

El hombre se ha caracterizado siempre por su búsqueda constante de nuevas vías para mejorar sus condiciones de vida. Estos esfuerzos le han servido para reducir el trabajo en aquellas operaciones en las que la fuerza juega un papel primordial. Los progresos obtenidos han permitido dirigir estos esfuerzos a otros campos, como, por ejemplo, a la construcción de máquinas calculadoras que ayuden a resolver de forma automática y rápida determinadas operaciones que resultan tediosas cuando se realizan a mano.

Uno de los primeros en acometer esta empresa fue Charles Babbage [6], quien trató infructuosamente de construir una máquina capaz de resolver problemas matemáticos. Posteriormente otros tantos intentaron construir máquinas similares, pero no fue hasta la Segunda Guerra Mundial, cuando ya se disponía de instrumentos electrónicos, que se empezaron a recoger los primeros frutos. En 1946 se construyó la primera computadora electrónica, ENIAC. Desde entonces los desarrollos en este campo han tenido un auge espectacular.

Estas máquinas permiten implementar fácilmente algoritmos para resolver multitud de problemas que antes resultaban engorrosos de resolver. Sin embargo, se observa una limitación importante: ¿qué ocurre cuando el problema que se quiere resolver no admite un tratamiento algorítmico, como es el caso, por ejemplo, de la clasificación de objetos por rasgos comunes? Este ejemplo demuestra que la construcción de nuevas máquinas más versátiles requiere un enfoque del problema desde otro punto de vista.

Los desarrollos actuales de los científicos se dirigen al estudio de las capacidades humanas como una fuente de nuevas ideas para el diseño de las nuevas máquinas. Así, la inteligencia artificial es un intento por descubrir y describir aspectos de la inteligencia humana que pueden ser simulados mediante máquinas.

Esta disciplina se ha desarrollado fuertemente en los últimos años teniendo aplicación en algunos campos como visión artificial [7], demostración de teoremas, procesamiento de información expresada mediante lenguajes humanos... etc.

Las redes neuronales son más que otra forma de emular ciertas características propias de los humanos, como la capacidad de memorizar y de asociar hechos. Si se examinan con atención aquellos problemas que no pueden expresarse a través de un algoritmo, se observará que todos ellos tienen una característica en común: la experiencia.

El hombre es capaz de resolver estas situaciones acudiendo a la experiencia acumulada. Así, parece claro que una forma de aproximarse al problema consista en la construcción de sistemas que sean capaces de reproducir esta característica humana.

En definitiva, las redes neuronales no son más que un modelo artificial y simplificado del cerebro humano [7], que es el ejemplo más perfecto del que disponemos para un sistema que es capaz de adquirir conocimiento a través de la experiencia. Una red neuronal es "un nuevo sistema para el tratamiento de la información, cuya unidad básica de procesamiento está inspirada en la célula fundamental del sistema nervioso humano: *la neurona*".

Todos los procesos del cuerpo humano se relacionan en alguna u otra forma con la (in)actividad de estas neuronas. Las mismas son un componente relativamente simple del ser humano, pero cuando millares de ellas se conectan en forma conjunta se hacen muy poderosas.

Lo que básicamente ocurre en una neurona biológica es lo siguiente: la neurona es estimulada o excitada a través de sus *entradas* (inputs) y cuando se alcanza un cierto umbral, la neurona se dispara o activa, pasando una señal hacia el *axon*. Posteriores investigaciones condujeron al descubrimiento de que estos procesos son el resultado de eventos electroquímicos.

Como ya se sabe, el pensamiento tiene lugar en el cerebro, que consta de billones de neuronas interconectadas. Así, el secreto de la "inteligencia" -sin importar como se defina- se sitúa dentro de estas neuronas interconectadas y de su interacción.

También, es bien conocido que los humanos son capaces de aprender.

Aprendizaje significa que aquellos problemas que inicialmente no pueden resolverse, pueden ser resueltos después de obtener más información acerca del problema. Por lo tanto, las Redes Neuronales consisten de unidades de procesamiento que intercambian datos o información.

Se utilizan para reconocer patrones, incluyendo imágenes, manuscritos y secuencias de tiempo (por ejemplo: tendencias financieras).

Tienen capacidad de aprender y mejorar su funcionamiento.

2.3. Reseñas históricas.

1936 - Alan Turing. Fue el primero en estudiar el cerebro como una forma de ver el mundo de la computación. Sin embargo, los primeros teóricos que concibieron los fundamentos de la computación neuronal fueron Warren McCulloch, un neurofisiólogo, y Walter Pitts, un matemático, quienes, en 1943, lanzaron una teoría acerca de la forma de trabajar de las neuronas (Un Cálculo Lógico de la Inminente Idea de la Actividad Nerviosa - Boletín de Matemática Biofísica 5: 115-133). Ellos modelaron una red neuronal simple mediante circuitos eléctricos.

1949 - Donald Hebb. Fue el primero en explicar los procesos del aprendizaje (que es el elemento básico de la inteligencia humana) desde un punto de vista psicológico, desarrollando una regla de como el aprendizaje ocurría. Aun hoy, este es el fundamento de la mayoría de las funciones de aprendizaje que pueden hallarse en una red neuronal. Su idea fue que el aprendizaje ocurría cuando ciertos cambios en una neurona eran activados. También intentó encontrar semejanzas entre el aprendizaje y la actividad nerviosa. Los trabajos de Hebb formaron las bases de la Teoría de las Redes Neuronales.

1950 - Karl Lashley. En sus series de ensayos, encontró que la información no era almacenada en forma centralizada en el cerebro sino que era distribuida encima de él.

1956 - Congreso de Dartmouth. Este Congreso frecuentemente se menciona para indicar el nacimiento de la inteligencia artificial.

1957 - Frank Rosenblatt. Comenzó el desarrollo del Perceptron. Esta es la red neuronal más antigua; utilizándose hoy en día para aplicación como identificador de patrones. Este modelo era capaz de generalizar, es decir, después de haber aprendido una serie de patrones podía reconocer otros similares, aunque no se le hubiesen presentado en el entrenamiento. Sin embargo, tenía una serie de limitaciones, por ejemplo, su incapacidad para resolver el problema de la función OR-exclusiva y, en general, era incapaz de clasificar clases no separables linealmente.

1959 - Frank Rosenblatt: Principios de Neurodinámica. En este libro confirmó que, bajo ciertas condiciones, el aprendizaje del Perceptron convergía hacia un estado finito (Teorema de Convergencia del Perceptron).

1960 - Bernard Widroff/Marcian Hoff. Desarrollaron el modelo Adaline (ADAptative LINear Elements). Esta fue la primera red neuronal aplicada a un problema real (filtros adaptativos para eliminar ecos en las líneas telefónicas) que se ha utilizado comercialmente durante varias décadas.

1961 - Karl Steinbeck: Die Lernmatrix. Red neuronal para simples realizaciones técnicas (memoria asociativa).

1969 - Marvin Minsky/Seymour Papert. En este año casi se produjo la "muerte abrupta" de las Redes Neuronales; ya que Minsky y Papert probaron (matemáticamente) que el Perceptrons no era capaz de resolver problemas relativamente fáciles, tales como el aprendizaje de una función no-lineal. Esto demostró que el Perceptron era muy débil, dado que las funciones no-lineales son extensamente empleadas en computación y en los problemas del mundo real.

1974 - Paul Werbos. Desarrolló la idea básica del algoritmo de aprendizaje de *propagación hacia atrás* (backpropagation); cuyo significado quedó definitivamente aclarado en 1985.

1977 - Stephen Grossberg: Teoría de Resonancia Adaptada (TRA). La Teoría de Resonancia Adaptada es una arquitectura de red que se diferencia de todas las demás previamente inventadas. La misma simula otras habilidades del cerebro: memoria a largo y corto plazo.

1985 - John Hopfield. Provocó el renacimiento de las redes neuronales con su libro: "Computación neuronal de decisiones en problemas de optimización."

1986 - David Rumelhart/G. Hinton. Redescubrieron el algoritmo de aprendizaje de propagación hacia atrás (*backpropagation*).

A partir de 1986, el panorama fue alentador con respecto a las investigaciones y el desarrollo de las redes neuronales. En la actualidad, son numerosos los trabajos que se realizan y publican cada año, las aplicaciones nuevas que surgen (sobretodo en el área de control) y las empresas que lanzan al mercado productos nuevos, tanto hardware como software (sobre todo para simulación).

2.4. El modelo biológico.

La teoría y modelado de redes neuronales artificiales está inspirada en la estructura y funcionamiento de los sistemas nerviosos, donde la neurona es el elemento fundamental. Existen neuronas de diferentes formas, tamaños y longitudes. Estos atributos son importantes para determinar la función y utilidad de la neurona. La clasificación de estas células en tipos estándar ha sido realizada por muchos neuroanatomistas.

2.4.1. Estructura de la neurona.

Una neurona es una célula viva y, como tal, contiene los mismos elementos que forman parte de todas las células biológicas. [8] Además, contienen elementos característicos que las diferencian. En general, una neurona consta de un cuerpo celular más o menos esférico, de 5 a 10 micras de diámetro, del que salen de una rama principal, el *axón*, y varias ramas más cortas, llamadas *dendritas*. A su vez, el axón puede producir ramas en torno a su punto de arranque, y con frecuencia se ramifica extensamente cerca de su extremo.

Una de las características que diferencian a las neuronas del resto de las células vivas, es su capacidad de comunicarse. En términos generales, las dendritas y el cuerpo celular reciben señales de entrada; el cuerpo celular las combina e integra y emite señales de salida. El axón transporta esas señales a los terminales axónicos, que se encargan de distribuir información a un nuevo conjunto de neuronas.

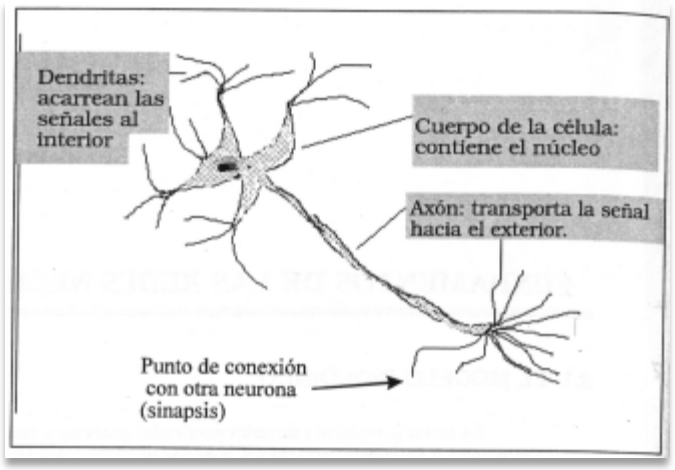


Figura 7. Forma general de una neurona

Por lo general, una neurona recibe información de miles de otras neuronas y, a su vez, envía información a miles de neuronas más. Se calcula que en el cerebro humano exista del orden de 10^{15} conexiones.

2.5. Definiciones de una Red Neuronal Artificial (RNA).

Existen numerosas formas de definir a las redes neuronales [6]; desde las definiciones cortas y genéricas hasta las que intentan explicar más detalladamente qué son las redes neuronales.

Por ejemplo:

1) Una nueva forma de computación, inspirada en modelos biológicos.

2) Un modelo matemático compuesto por un gran número de elementos procesales organizados en niveles.

3) Un sistema de computación compuesto por un gran número de elementos simples, elementos de procesos muy interconectados, los cuales procesan información por medio de su estado dinámico como respuesta a entradas externas.

4) Redes neuronales artificiales son redes interconectadas masivamente en paralelo de elementos simples (usualmente adaptativos) y con organización jerárquica, las cuales intentan interactuar con los objetos del mundo real del mismo modo que lo hace el sistema nervioso biológico.

2.6. Ventajas que ofrecen las redes neuronales.

Debido a su constitución y a sus fundamentos, las redes neuronales artificiales presentan un gran número de características semejantes a las del cerebro [6].

Por ejemplo, son capaces de aprender de la experiencia, de generalizar de casos anteriores a nuevos casos, de abstraer características esenciales a partir de entradas que representan información irrelevante, etc. Esto hace que ofrezcan numerosas ventajas [9], y que este tipo de tecnología se esté aplicando en múltiples áreas.

Entre las ventajas se incluyen:

- **Aprendizaje Adaptativo**. Capacidad de aprender a realizar tareas basadas en un entrenamiento o en una experiencia inicial.

- **Auto-organización.** Una red neuronal puede crear su propia organización o representación de la información que recibe mediante una etapa de aprendizaje.

- **Tolerancia a fallos.** La destrucción parcial de una red conduce a una degradación de su estructura; sin embargo, algunas capacidades de la red se pueden retener, incluso sufriendo un gran daño.

- **Operación en tiempo real**. Los cómputos neuronales pueden ser realizados en paralelo; para esto se diseñan y fabrican máquinas con hardware especial para obtener esta capacidad.

- **Fácil inserción dentro de la tecnología existente**. Se pueden obtener chips especializados para redes neuronales que mejoran su capacidad en ciertas tareas. Ello facilitará la integración modular en los sistemas existentes.

2.6.1. Aprendizaje adaptativo.

La capacidad de aprendizaje adaptativo es una de las características más atractivas de redes neuronales [9]. Esto es, aprenden a llevar a cabo ciertas tareas mediante un entrenamiento con ejemplos ilustrativos.

Como las redes neuronales pueden aprender a diferenciar patrones mediante ejemplos y entrenamientos, no es necesario elaborar modelos a priori ni necesidad de especificar funciones de distribución de probabilidad.

Las redes neuronales son sistemas dinámicos autoadaptativos. Son adaptables debido a la capacidad de autoajuste de los elementos procesales (neuronas) que componen el sistema. Son dinámicos, pues son capaces de estar constantemente cambiando para adaptarse a las nuevas condiciones.

En el proceso de aprendizaje, los enlaces ponderados de las neuronas se ajustan de manera que se obtengan ciertos resultados específicos. Una red neuronal no necesita un algoritmo para resolver un problema, ya que ella puede generar su propia distribución de pesos en los enlaces mediante el aprendizaje.

También existen redes que continúan aprendiendo a lo largo de su vida, después de completado su período de entrenamiento.

La función del diseñador es únicamente la obtención de la arquitectura apropiada. No es problema del diseñador el cómo la red aprenderá a discriminar. Sin embargo, sí es necesario que desarrolle un buen algoritmo de aprendizaje que le proporcione a la red la capacidad de discriminar, mediante un entrenamiento con patrones.

2.6.2. Auto-organización.

Las redes neuronales emplean su capacidad de aprendizaje adaptativo para auto organizar la información que reciben durante el aprendizaje y/o la operación. Mientras que el aprendizaje es la modificación de cada elemento procesal, la auto organización consiste en la modificación de la red neuronal completa para llevar a cabo un objetivo específico.

Cuando las redes neuronales se usan para reconocer ciertas clases de patrones, ellas auto organizan la información usada. Por ejemplo, la red llamada *backpropagation*, creará su propia representación característica, mediante la cual puede reconocer ciertos patrones.

Esta auto organización provoca la generalización: facultad de las redes neuronales de responder apropiadamente cuando se les presentan datos o situaciones a las que no había sido expuesta anteriormente. El sistema puede generalizar la entrada para obtener una respuesta. Esta característica es muy importante cuando se tiene que solucionar problemas en los cuales la información de entrada no es muy clara; además permite que el sistema dé una solución, incluso cuando la información de entrada está especificada de forma incompleta.

2.6.3. Tolerancia a fallos.

Las redes neuronales fueron los primeros métodos computacionales con la capacidad inherente de tolerancia a fallos [9]. Comparados con los sistemas computacionales tradicionales, los cuales pierden su funcionalidad cuando sufren un pequeño error de memoria, en las redes neuronales, si se produce un fallo en un número no muy grande de neuronas y aunque el comportamiento del sistema se ve influenciado, no sufre una caída repentina.

Hay dos aspectos distintos respecto a la tolerancia a fallos:

a) Las redes pueden aprender a reconocer patrones con ruido, distorsionados o incompletos. Esta es una tolerancia a fallos respecto a los datos.

b) Las redes pueden seguir realizando su función (con cierta degradación) aunque se destruya parte de la red.

La razón por la que las redes neuronales son tolerantes a los fallos es que tienen su información distribuida en las conexiones entre neuronas, existiendo cierto grado de redundancia en este tipo de almacenamiento.

La mayoría de los ordenadores algorítmicos y sistemas de recuperación de datos almacenan cada pieza de información en un espacio único, localizado y direccionable. En cambio, las redes neuronales almacenan información no localizada. Por lo tanto, la mayoría de las interconexiones entre los nodos de la red tendrán sus valores en función de los estímulos recibidos, y se generará un patrón de salida que represente la información almacenada.

2.6.4. Operación en tiempo real.

Una de las mayores prioridades, casi en la totalidad de las áreas de aplicación, es la necesidad de realizar procesos con datos de forma muy rápida. Las redes neuronales se adaptan bien a esto debido a su implementación paralela. Para que la mayoría de las redes puedan operar en un entorno de tiempo real, la necesidad de cambio en los pesos de las conexiones o entrenamiento es mínimo.

2.6.5. Fácil inserción dentro de la tecnología existente.

Una red individual puede ser entrenada para desarrollar una única y bien definida tarea (tareas complejas, que hagan múltiples selecciones de patrones, requerirán sistemas de redes interconectadas).

Con las herramientas computacionales existentes (no del tipo PC), una red puede ser rápidamente entrenada, comprobada, verificada y trasladada a una implementación hardware de bajo coste. Por lo tanto, no se presentan dificultades para la inserción de redes neuronales en aplicaciones específicas, por ejemplo, de control, dentro de los sistemas existentes.

De esta manera, las redes neuronales se pueden utilizar para mejorar sistemas en forma incremental y cada paso puede ser evaluado antes de acometer un desarrollo más amplio.

2.7. Elementos Básicos.

2.7.1. Elementos básicos que componen una red neuronal.

A continuación, se puede ver, en la siguiente figura, un esquema de una red neuronal artificial:

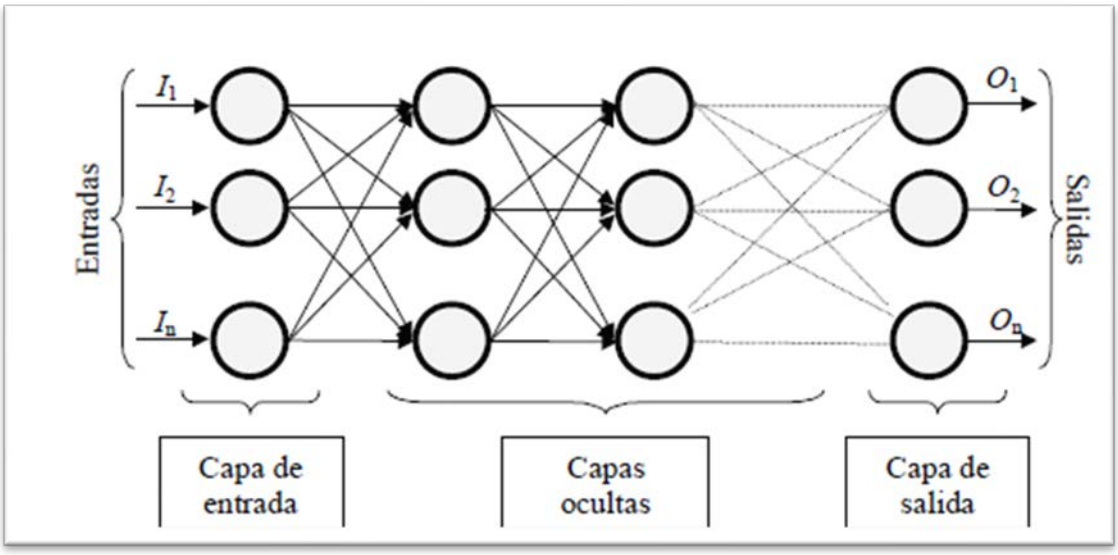


Figura 8. Ejemplo de una red neuronal totalmente conectada.

La misma está constituida por neuronas interconectadas y arregladas en tres capas (esto último puede variar) [6]. Los datos ingresan por medio de la "*capa de entrada*", pasan a través de la "*capa oculta*" y salen por la "*capa de salida*". Cabe mencionar que la capa oculta puede estar constituida por varias capas.

Antes de comenzar el estudio sobre las redes neuronales, se debe aprender algo sobre las neuronas y de cómo ellas son utilizadas por una red neuronal. En la *Figura 3* se compara una neurona biológica con una neurona artificial. En la misma se pueden observar las similitudes entre ambas (tienen entradas, utilizan pesos y generan salidas).

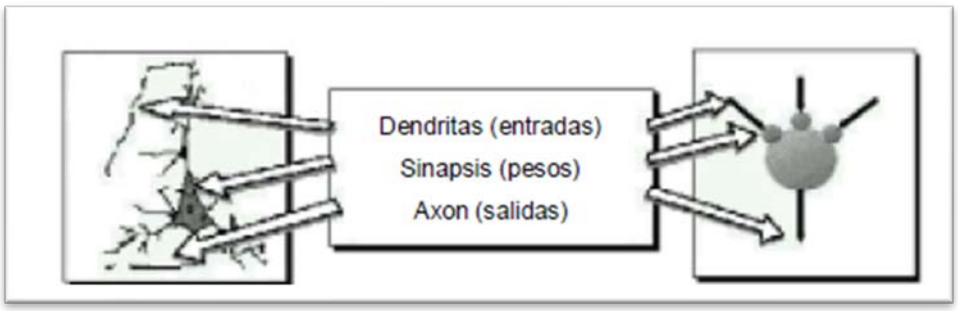


Figura 9. Comparación entre una neurona biológica (izquierda) y una artificial (derecha).

Mientras una neurona es muy pequeña en sí misma, cuando se combinan cientos, miles o millones de ellas pueden resolver problemas muy complejos. Por ejemplo, el cerebro humano se compone de billones de tales neuronas.

2.7.2. Niveles o capas de una red neuronal.

La distribución de neuronas dentro de la red se realiza formando niveles o capas [6], con un número determinado de dichas neuronas en cada una de ellas. A partir de su situación dentro de la red, se pueden distinguir tres tipos de capas:

- ***De entrada*****:** es la capa que recibe directamente la información proveniente de las fuentes externas de la red.

- ***Ocultas*****:** son internas a la red y no tienen contacto directo con el entorno exterior. El número de niveles ocultos puede estar entre cero y un número elevado. Las neuronas de las capas ocultas pueden estar interconectadas de distintas maneras, lo que determina, junto con su número, las distintas topologías de redes neuronales.

- ***De salidas*****:** transfieren información de la red hacia el exterior.

2.7.3. Función de entrada (input function).

La neurona trata a muchos valores de entrada como si fueran uno solo; esto recibe el nombre de *entrada global*. Por lo tanto, ahora nos enfrentamos al problema de cómo se pueden combinar estas simples entradas (*ini*1, *ini*2, ...) dentro de la entrada global, *gini*. Esto se logra a través de la *función de entrada*, la cual se calcula a partir del *vector entrada*.

La función de entrada puede describirse como sigue:

$$inputi = (ini1 \bullet wi1) * (ini2 \bullet wi2) * \ldots (inin \bullet win)$$

donde: * representa al operador apropiado (por ejemplo: máximo, sumatoria, productoria, etc.), *n* al número de entradas a la neurona *Ni* y *wi* al peso.

Los valores de entrada se multiplican por los pesos anteriormente ingresados a la neurona [10]. Por consiguiente, los pesos que generalmente no están restringidos cambian la medida de influencia que tienen los valores de entrada. Es decir, que permiten que un gran valor de entrada tenga solamente una pequeña influencia, si estos son lo suficientemente pequeños.

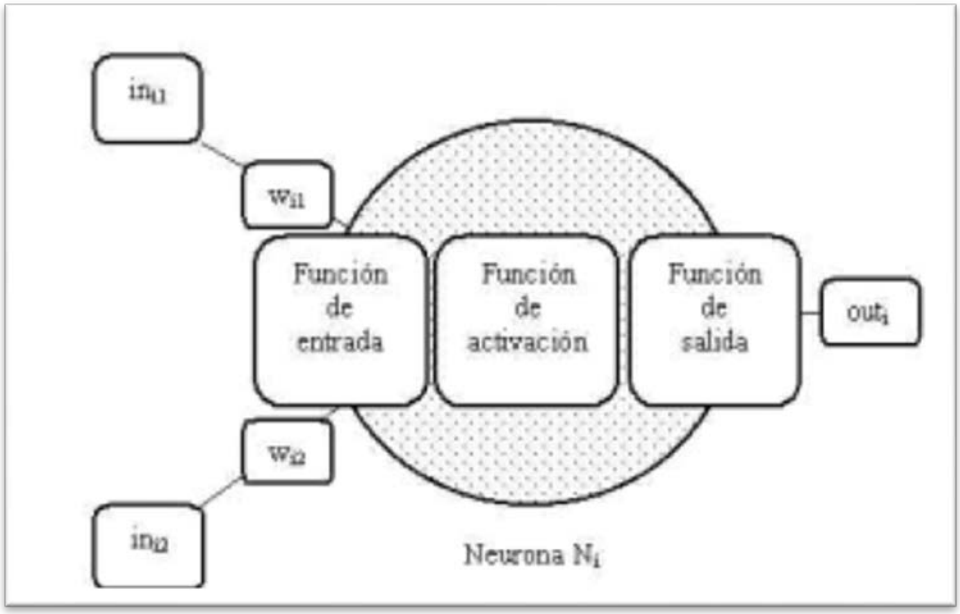


Figura 10. Ejemplo de una neurona con 2 entradas y 1 salida.

La nomenclatura utilizada en la figura anterior es la siguiente: *ini1* = entrada número 1 a la neurona *Ni*; *wi1* = peso correspondiente a *ini1*; *ini2* = entrada número 2 a la neurona *Ni*; *wi2* = peso correspondiente a *ini2*; y *outi* = salida de la neurona *Ni*. El conjunto de todas las *n* entradas *ini* = (*ini1*, *ini2*, ..., *inin*) es comúnmente llamado "*vector entrada*".

Algunas de las funciones de entrada más comúnmente utilizadas y conocidas son:

1) ***Sumatoria de las entradas pesadas****:* es la suma de todos los valores de entrada a la neurona, multiplicados por sus correspondientes pesos.

$$\sum_j \left(n_{ij}\, w_{ij}\right), \quad \text{con } j = 1, 2, \ldots, n$$

2) ***Productoria de las entradas pesadas****:* es el producto de todos los valores de entrada a la neurona, multiplicados por sus correspondientes pesos.

$$\prod_j \left(n_{ij}\, w_{ij}\right), \quad \text{con } j = 1, 2, \ldots, n$$

3**)** ***Máximo de las entradas pesadas:*** solamente toma en consideración el valor de entrada más fuerte, previamente multiplicado por su peso correspondiente.

$$\operatorname{Max}_j \left(n_{ij}\, w_{ij}\right) \quad \text{con } j = 1, 2, \ldots, n$$

2.7.4. Función de activación (activation function).

Una neurona biológica puede estar activa (excitada) o inactiva (no excitada) [6]; es decir, que tiene un "*estado de activación*". Las neuronas artificiales también tienen diferentes estados de activación; algunas de ellas solamente dos, al igual que las biológicas, pero otras pueden tomar cualquier valor dentro de un conjunto determinado.

La *función activación* calcula el estado de actividad de una neurona; transformando la entrada global (menos el umbral, Θi) en un valor (estado) de activación, cuyo rango normalmente va de (0 a 1) o de (–1 a 1). Esto es así, porque una neurona puede estar totalmente inactiva (0 o –1) o activa (1). La función activación, es una función de la entrada global (*gini*) menos el umbral (Θi). Las funciones de activación más comúnmente utilizadas se detallan a continuación:

1) Función lineal:

$$f(x) = \begin{cases} -1 & x \leq -1/a \\ a * x & -1/a < x < 1/a \\ 1 & x \geq 1/a \end{cases}$$

con x = *gini* - Θi, y a > 0.

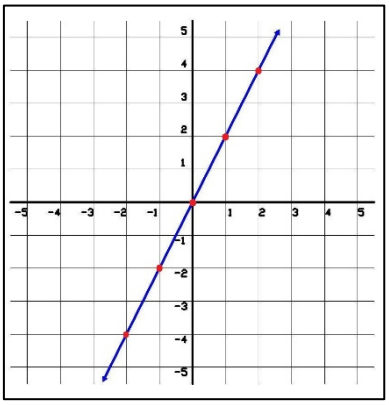


Figura 11. Función de activación lineal

Los valores de salida obtenidos por medio de esta función de activación serán:a·(*gini* - Θi), cuando el argumento de (*gini* - Θi) esté comprendido dentro del rango (-1/a, 1/a). Por encima o por debajo de esta zona se fija la salida en 1 o –1, respectivamente. Cuando a = 1 (siendo que la misma afecta la pendiente de la gráfica), la salida es igual a la entrada.

2) Función sigmoidea:

$$f(x) = \left\{ \frac{1}{1 + e^{-gx}}, \text{con } x = gini - \Theta i \right.$$

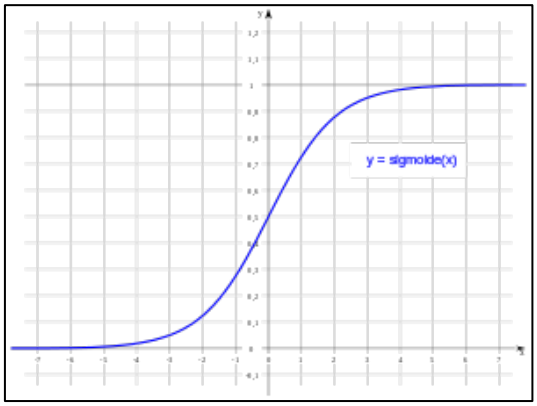


Figura 12.Función de activación sigmoidea

Los valores de salida que proporciona esta función están comprendidos dentro de un rango que va de 0 a 1. Al modificar el valor de g se ve afectada la pendiente de la función de activación.

3) Función tangente hiperbólica:

$$f(x) = \begin{cases} \frac{e^{gx} - e^{-gx}}{e^{gx} + e^{-gx}}, \text{con } x = gini - \Theta i \end{cases}$$

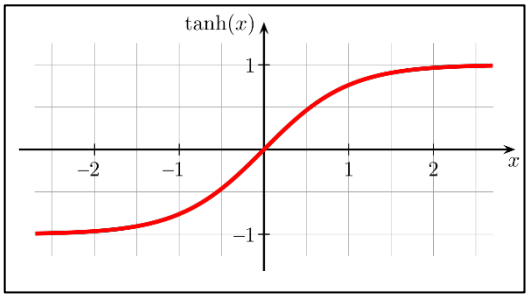


Figura 13. Función de activación tangente parabólica

Los valores de salida de la función tangente hiperbólica están comprendidos dentro de un rango que va de -1 a 1. Al modificar el valor de g se ve afectada la pendiente de la función de activación.

Para explicar porque se utilizan estas funciones de activación se suele emplear la analogía a la aceleración de un automóvil. Cuando un auto inicia su movimiento necesita una potencia elevada para comenzar a acelerar. Pero al ir tomando velocidad, este demanda un menor incremento de dicha potencia para mantener la aceleración. Al llegar a altas velocidades, nuevamente un amplio incremento en la potencia es necesario para obtener una pequeña ganancia de velocidad. En resumen, en ambos extremos del rango de aceleración de un automóvil se demanda una mayor potencia para la aceleración que en la mitad de dicho rango.

2.7.5. Función de salida (output function).

El último componente que una neurona necesita es la *función de salida*. El valor resultante de esta función es la salida de la neurona *i* (*outi*); por ende, la función de salida determina que valor se transfiere a las neuronas vinculadas. Si la función de activación está por debajo de un umbral determinado, ninguna salida se pasa a la neurona subsiguiente. Normalmente, no cualquier valor es permitido como una entrada para una neurona, por lo tanto, los valores de salida están comprendidos en el rango [0, 1] o [-1, 1]. También pueden ser binarios {0, 1} o {-1, 1}.

Dos de las funciones de salida más comunes son:

- Ninguna: este es el tipo de función más sencillo, tal que la salida es la misma que la entrada. Es también llamada *función identidad.*
- Binaria: $\begin{cases} 1 & si\ act_i \geq \xi_i \\ 0 & de\ lo\ contrario \end{cases}$, *donde* ξ_i *es el umbral*

2.1. Aprendizaje, Validación y Codificación.

2.1.1. Mecanismos de aprendizaje.

Se ha visto que los datos de entrada se procesan a través de la red neuronal con el propósito de lograr una salida [6]. También se dijo que las redes neuronales extraen generalizaciones desde un conjunto determinado de ejemplos anteriores de tales problemas de decisión. Una red neuronal debe aprender a calcular la salida correcta para cada constelación (arreglo o vector) de entrada en el conjunto de ejemplos. Este proceso de aprendizaje se denomina: *proceso de entrenamiento o acondicionamiento*. El conjunto de datos (o conjunto de ejemplos) sobre el cual este proceso se basa es, por ende, llamado: *conjunto de datos de entrenamiento*.

Si la topología de la red y las diferentes funciones de cada neurona (entrada, activación y salida) no pueden cambiar durante el aprendizaje, mientras que los pesos sobre cada una de las conexiones si pueden hacerlo; el aprendizaje de una red neuronal significa: *adaptación de los pesos*.

En otras palabras, el aprendizaje es el proceso por el cual una red neuronal modifica sus pesos en respuesta a una información de entrada. Los cambios que se producen durante el mismo se reducen a la destrucción, modificación y creación de conexiones entre las neuronas. En los sistemas biológicos existe una continua destrucción y creación de conexiones entre las neuronas. En los modelos de redes neuronales artificiales, la creación de una nueva conexión implica que el peso de la misma pasa a tener un valor distinto de cero. De la misma manera, una conexión se destruye cuando su peso pasa a ser cero.

Durante el proceso de aprendizaje, los pesos de las conexiones de la red sufren modificaciones, por lo tanto, se puede afirmar que este proceso ha terminado (la red ha aprendido) cuando los valores de los pesos permanecen estables ($dw_{ij}/dt = 0$).

Un aspecto importante respecto al aprendizaje de las redes neuronales es el conocer cómo se modifican los valores de los pesos, es decir, cuáles son los criterios que se siguen para cambiar el valor asignado a las conexiones cuando se pretende que la red aprenda una nueva información.

Hay dos métodos de aprendizaje importantes que pueden distinguirse:

- Aprendizaje supervisado.
- Aprendizaje no supervisado.

2.1.1.1 Aprendizaje supervisado.

El aprendizaje supervisado se caracteriza porque el proceso de aprendizaje se realiza mediante un entrenamiento controlado por un agente externo (supervisor, maestro) que determina la respuesta que debería generar la red a partir de una entrada determinada. El supervisor controla la salida de la red y en caso de que ésta no coincida con la deseada, se procederá a modificar los pesos de las conexiones, con el fin de conseguir que la salida obtenida se aproxime a la deseada.

En este tipo de aprendizaje se suelen considerar, a su vez, tres formas de llevarlo a cabo, que dan lugar a los siguientes aprendizajes supervisados:

1. **Aprendizaje por corrección de error:** Consiste en ajustar los pesos de las conexiones de la red en función de la diferencia entre los valores deseados y los obtenidos a la salida de la red, es decir, en función del error cometido en la salida

2. **Aprendizaje por refuerzo**: Se trata de un aprendizaje supervisado, más lento que el anterior, que se basa en la idea de no disponer de un ejemplo completo del comportamiento deseado, es decir, de no indicar durante el entrenamiento exactamente la salida que se desea que proporcione la red ante una determinada entrada.

3. **Aprendizaje estocástico:** Consiste básicamente en realizar cambios aleatorios en los valores de los pesos de las conexiones de la red y evaluar su efecto a partir del objetivo deseado y de distribuciones de probabilidad.

2.1.1.2 Aprendizaje no supervisado.

Las redes con aprendizaje no supervisado (también conocido como autosupervisado) no requieren influencia externa para ajustar los pesos de las conexiones entre sus neuronas [6]. La red no recibe ninguna información por parte del entorno que le indique si la salida generada en respuesta a una determinada entrada es o no correcta.

Estas redes deben encontrar las características, regularidades, correlaciones o categorías que se puedan establecer entre los datos que se presenten en su entrada. Existen varias posibilidades en cuanto a la interpretación de la salida de estas redes, que dependen de su estructura y del algoritmo de aprendizaje empleado.

En algunos casos, la salida representa el grado de familiaridad o similitud entre la información que se le está presentando en la entrada y las informaciones que se le han mostrado hasta entonces (en el pasado).

En otro caso, podría realizar una clusterización (clustering) o establecimiento de categorías, indicando la red a la salida a qué categoría pertenece la información presentada a la entrada, siendo la propia red quien debe encontrar las categorías apropiadas a partir de las correlaciones entre las informaciones presentadas.

En cuanto a los algoritmos de aprendizaje no supervisado, en general se suelen considerar dos tipos, que dan lugar a los siguientes aprendizajes:

- **Aprendizaje hebbiano:** Esta regla de aprendizaje es la base de muchas otras, la cual pretende medir la familiaridad o extraer características de los datos de entrada. El fundamento es una suposición bastante simple: si dos neuronas *Ni* y *Nj* toman el mismo estado simultáneamente (ambas activas o ambas inactivas), el peso de la conexión entre ambas se incrementa.

- **Aprendizaje competitivo y comparativo:** Se orienta a la clusterización o clasificación de los datos de entrada. Como característica principal del aprendizaje competitivo se puede decir que, si un patrón nuevo se determina que pertenece a una clase reconocida previamente, entonces la inclusión de este nuevo patrón a esta clase matizará la representación de la misma. Si el patrón de entrada se determinó que no pertenece a ninguna de las clases reconocidas anteriormente, entonces la estructura y los pesos de la red neuronal serán ajustados para reconocer la nueva clase.

2.1.2. Elección del conjunto inicial de pesos.

Antes de comenzar el proceso de entrenamiento se debe determinar un estado inicial, lo que significa: escoger un conjunto inicial de pesos para las diversas conexiones entre las neuronas de la red neuronal. Esto puede realizarse por varios criterios; por ejemplo uno de ellos es otorgar un peso aleatorio a cada conexión, encontrándose los mismos dentro de un cierto intervalo. Generalmente un intervalo del tipo [-*n*, *n*], donde *n* es un número natural positivo.

Cabe mencionar que durante el transcurso del entrenamiento los pesos no se encuentran restringidos a dicho intervalo.

2.1.3. Detención del proceso de aprendizaje.

Para determinar cuándo se detendrá el proceso de aprendizaje, es necesario establecer una "*condición de detención"*.

Normalmente el entrenamiento se detiene cuando el cálculo del error cuadrado sobre todos los ejemplos de entrenamiento ha alcanzado un mínimo o cuando para cada uno de los ejemplos dados, el error observado está por debajo de un determinado umbral.

Ya que, para controlar este proceso, la mayor parte de las herramientas de las redes neuronales muestran estos errores utilizando gráficos especiales; los cuales no son utilizados para el aprendizaje, si no que solamente para dar un indicio del proceso en sí mismo.

Otra condición de detención del aprendizaje puede ser cuando un cierto número de ciclos y/o pasos de entrenamiento hayan sido completamente corridos.

Luego de alcanzarse la condición de detención, los pesos no se volverán a cambiar. Entonces podemos decir que la transformación de los datos de entrada a los de salida está resuelta. Esto se puede interpretar como una *función f* oculta en el conjunto de la red neuronal. Esta función es exactamente la "instrucción" de cómo la salida será calculada a partir de una constelación (vector) de entrada.

El orden en que los ejemplos de entrenamiento se presentan a la red neuronal es otro tema importante. En general se ha observado que en la mayoría de los casos es beneficioso realizarlo en forma aleatoria.

2.1.4. Validación de la red neuronal.

Después del proceso de entrenamiento los pesos de las conexiones en la red neuronal quedan fijos. Como paso siguiente se debe comprobar si la red neuronal puede resolver nuevos problemas, del tipo general, para los que ha sido entrenada. Por lo tanto, con el propósito de validar la red neuronal se requiere de otro conjunto de datos, denominado *conjunto de validación o testeo*.

Cada ejemplo del conjunto de evaluación contiene los valores de las variables de entrada, con su correspondiente solución tomada; pero ahora esta solución no se le es otorgada a la red neuronal. Luego se compara la solución calculada para cada ejemplo de validación con la solución conocida.

El nuevo ejemplo utilizado para la validación se identifica como *Eu* y su correspondiente salida correcta como *Au* (*u* indica incógnita, en inglés). Ahora el problema es que hay que decidir cuando la salida de la red neuronal ha de considerarse como correcta.

2.2. Fenómenos que se deben evitar en el aprendizaje

2.2.1. Overtraining.

Uno de los aspectos más importantes en la construcción y desarrollo de las RNA´s es la capacidad de la red para generalizar a partir de ejemplos [11], evitando la simple memorización de patrones de aprendizaje, y proporcionando una respuesta correcta ante individuos no presentados durante la fase de entrenamiento.

Durante el proceso de entrenamiento o aprendizaje de una red neuronal supervisada, del tipo Perceptrón multicapa, los pesos son modificados de forma iterativa de acuerdo con los valores del grupo de entrenamiento, con el objeto de minimizar el error cometido entre la salida obtenida por la red y la salida deseada por el usuario.

De forma característica, en las primeras fases del aprendizaje la red se va adaptando progresivamente al conjunto de datos de entrenamiento, acomodándose al problema y favoreciendo la generalización.

Así, se puede observar que el error que comete la red ante los datos de entrenamiento va descendiendo paulatinamente hasta alcanzar un valor mínimo. Sin embargo, a partir de un momento dado el sistema puede comenzar a ajustarse demasiado a las particularidades irrelevantes (ruido) presentes en los patrones de entrenamiento en vez de ajustarse a la función subyacente que relaciona entradas y salidas.

Llegados a este punto se dice que la red ha sufrido un sobre-entrenamiento o sobreaprendizaje, perdiendo su habilidad de generalizar su aprendizaje a casos nuevos [11].

Ahora bien, es un hecho experimental observable que, si se entrena la red para alcanzar un error de aprendizaje muy reducido (por ejemplo, inferior al 1%), el error en test se degrada, obteniéndose una gráfica similar a la de la siguiente figura.

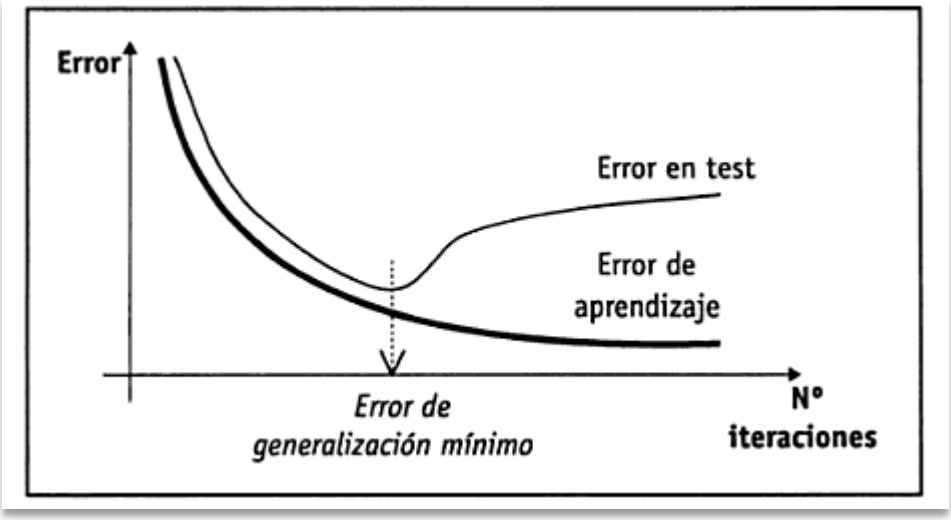


Figura 14. Análisis del fenómeno de sobreaprendizaje de la red.

Con el fin de evitar el problema del sobre-entrenamiento, que puede darse en las redes del tipo Perceptrón multicapa, es aconsejable utilizar un segundo grupo de datos diferentes a los de entrenamiento, denominado grupo de validación, que permita controlar el proceso de aprendizaje.

De este modo, a lo largo del aprendizaje la red va modificando los pesos en función de los datos de entrenamiento y de forma alternada se va obteniendo el error que comete la red ante los datos de validación. Esto permite estimar el error de generalización de la red –es decir, el error que se comete ante patrones diferentes a los utilizados en el entrenamiento—, a partir del error que comete ante los patrones de validación (error de validación) a lo largo del proceso de aprendizaje.

Normalmente, en las primeras fases del entrenamiento el error de validación va disminuyendo progresivamente hasta un punto a partir del cual este error comienza a aumentar, ese punto indica que la red empieza a aprender las particularidades del grupo de entrenamiento –se produce el sobre-entrenamiento. Una práctica común, con el fin de evitar el sobre-entrenamiento, consiste en detener el aprendizaje cuando el error de validación alcanza el punto mínimo.

La utilización de un grupo de validación también permite afinar los parámetros de la red, por ejemplo, para seleccionar el número óptimo de unidades ocultas. Así, la arquitectura que obtenga el menor error de validación será la seleccionada.

2.2.2. Overfitting.

La topología de una red neuronal, es decir, el número de nodos y la ubicación y el número de conexiones entre ellos, tiene un impacto significativo en la performance de la red y su habilidad para generalizar [11]. La densidad de conexiones en una red neuronal determina su habilidad para almacenar información.

Si una red no tiene suficientes conexiones entre nodos, el algoritmo de entrenamiento puede no converger nunca; la red neuronal no es capaz de aproximar la función. Por el otro lado, en una red densamente conectada, puede ocurrir el sobreajuste (overfitting).

El sobreajuste es un problema de los modelos estadísticos donde se presentan demasiados parámetros. Esto es una mala situación porque en lugar de aprender a aproximar la función presente en los datos, la red simplemente puede memorizar cada ejemplo de entrenamiento.

El ruido en los datos de entrenamiento se aprende entonces como parte de la función, a menudo destruyendo la habilidad de la red para generalizar.

Para evitar este problema, se suele utilizar el método de la validación cruzada, y el cúal se explicará en el siguiente sub-apartado.

2.1.1.3 Validación cruzada.

La esencia del aprendizaje back-propagation es codificar un mapeo entrada-salida (representado por un conjunto de ejemplos etiquetados) en los pesos y umbrales de un Perceptrón multicapa [12].

Lo que queremos es que la red se entrene bien de forma tal que aprenda lo suficiente acerca del pasado para generalizar en el futuro. Desde esta perspectiva el proceso de aprendizaje debe elegir la parametrización de la red más acorde a este conjunto de ejemplos.

Más específicamente, podemos ver al problema de selección de la red como una elección, dentro del conjunto de modelos de estructuras candidatas (parametrizaciones), de la mejor estructura de acuerdo a un cierto criterio [12].

En este contexto, una herramienta estándar en estadística conocida como *validación cruzada* provee una guía interesante [13]. El set de entrenamiento se particiona aleatoriamente en dos subsets disjuntos:

- El *subset de estimación*, usado para seleccionar el modelo.
- El *subset de validación*, usado para evaluar o validar el modelo.

La motivación aquí es validar el modelo sobre un set de datos diferente de aquel utilizado para la estimación de los parámetros. De este modo podemos usar el set de entrenamiento para evaluar la performance de varios modelos candidatos, y así elegir el mejor.

El uso de la validación cruzada resulta particularmente interesante cuando debemos diseñar una red neuronal grande que tenga como objetivo a la generalización.

2.3. Principales Topologías de redes neuronales.

La topología o arquitectura de una red neuronal consiste en la organización y disposición de las neuronas en la misma [6], formando capas o agrupaciones de neuronas más o menos alejadas de la entrada y salida de dicha red. En este sentido, los parámetros fundamentales de la red son: el número de capas, el número de neuronas por capa, el grado de conectividad y el tipo de conexiones entre neuronas.

2.3.1. Redes monocapa.

En las redes monocapa, se establecen conexiones entre las neuronas que pertenecen a la única capa que constituye la red. Las redes monocapas se utilizan generalmente en tareas relacionadas con lo que se conoce como autoasociación (regenerar información de entrada que se presenta a la red de forma incompleta o distorsionada).

2.3.2. Redes multicapa.

Las redes multicapas son aquellas que disponen de un conjunto de neuronas agrupadas en varios (2, 3, etc.) niveles o capas.

En estos casos, una forma para distinguir la capa a la que pertenece una neurona consistiría en fijarse en el origen de las señales que recibe a la entrada y el destino de la señal de salida.

Normalmente, todas las neuronas de una capa reciben señales de entrada desde otra capa anterior (la cual está más cerca a la entrada de la red), y envían señales de salida a una capa posterior (que está más cerca a la salida de la red). A estas conexiones se las denomina *conexiones hacia adelante o feedforward*.

Sin embargo, en un gran número de estas redes también existe la posibilidad de conectar la salida de las neuronas de capas posteriores a la entrada de capas anteriores; a estas conexiones se las denomina *conexiones hacia atrás o feedback.*

Estas dos posibilidades permiten distinguir entre dos tipos de redes con múltiples capas: las redes con conexiones hacia adelante o *redes feedforward*, y las redes que disponen de conexiones tanto hacia adelante como hacia atrás o *redes feedforward/feedback*.

2.3.3. Redes Neuronales con conexiones hacia adelante.

En este apartado examinaremos un grupo específico de redes neuronales que tienen una arquitectura similar. Este es el grupo de las redes con conexiones hacia adelante, las cuales se caracterizan por arquitecturas en niveles y conexiones estrictamente hacia adelante entre neuronas. Estas redes son todas *buenos clasificadores de patrones* y utilizan *aprendizaje supervisado.*

Este grupo incluye el Perceptron, las redes ADALINE y MADELINE y la red Backpropagation. El perceptron y las redes ADALINE y MADELINE tienen un importante interés histórico y han abierto el camino para el desarrollo de otras redes neuronales. Por otro lado, la red Backpropagation es probablemente una de las más utilizadas hoy en día.

2.3.4. El Perceptron.

Este fue el primer modelo de red neuronal artificial desarrollado por Rossenblat en 1958 [14]. Despertó un enorme interés en los años 60, debido a su capacidad para aprender a reconocer patrones sencillos: un Perceptron, formado por varias neuronas lineales para recibir las entradas a la red y una neurona de salida, es capaz de decidir cuándo una entrada presentada a la red pertenece a una de las dos clases que es capaz de reconocer.

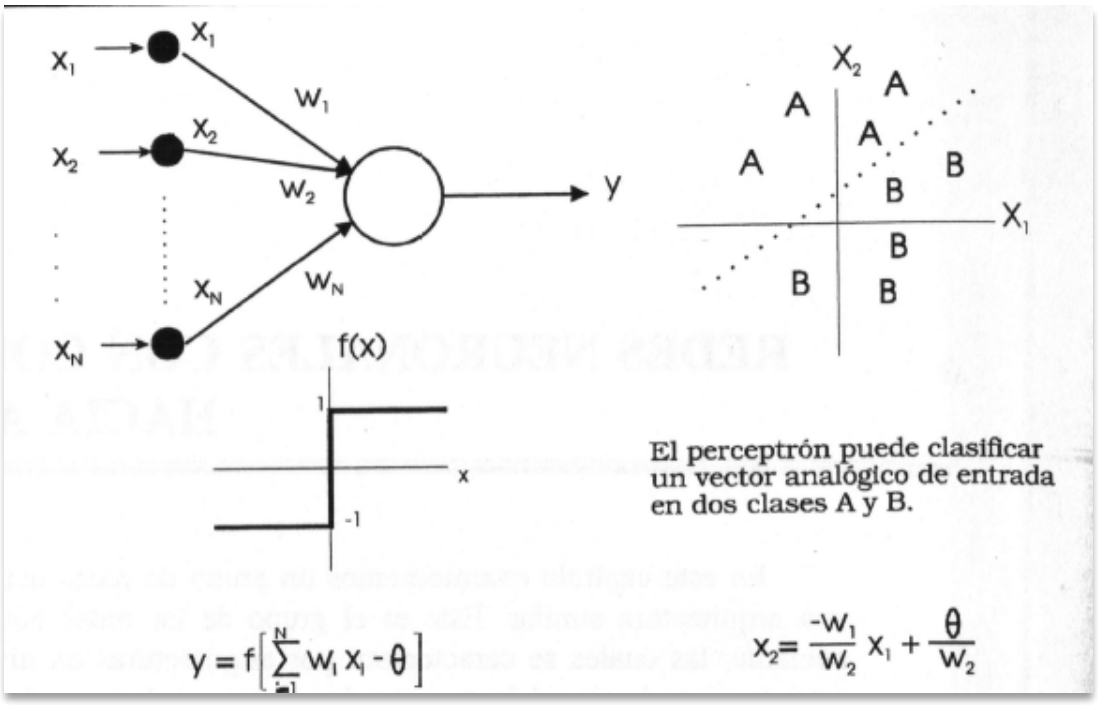


Figura 15. El perceptron.

La única neurona de salida del Perceptron 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 a la clase A, o -1 si el patrón pertenece a la clase B (figura 15). La salida dependerá de la entrada neta (suma de las entradas χ_i ponderadas) y del valor umbral Θ. La idea de Perceptron según Rosenblatt se muestra en la siguiente figura:

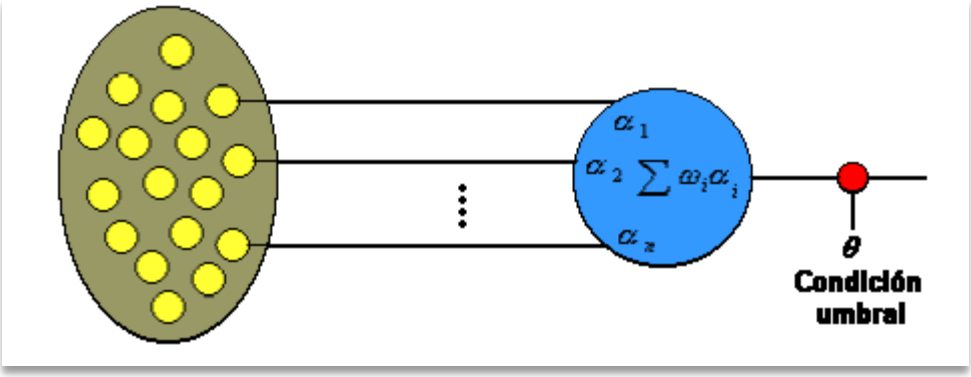


Figura 16. Conexión del Perceptron

Una técnica utilizada para analizar el comportamiento de redes como el Perceptron es representar en un mapa las regiones de decisión creadas en el espacio multidimensional de entradas a la red. En estas regiones se visualiza qué patrones pertenecen a una clase y cuáles a otra.

El Perceptron 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 [14]. En este caso, los valores de los pesos pueden fijarse o adaptarse utilizando diferentes algoritmos de entrenamiento de la red.

Sin embargo, el Perceptron, al constar de una capa de entrada y otra de salida con una única neurona, tiene capacidad de representación bastante limitada. Este modelo sólo es capaz de discriminar patrones muy sencillos, linealmente separables. El caso más conocido es la imposibilidad del Perceptron de representar la función OR-EXCLUSIVA.

La separabilidad lineal limita a las redes con sólo dos capas a la resolución de problemas en los cuáles el conjunto de puntos (correspondientes a los valores de entrada) sean separables geométricamente. En el caso de dos entradas, la separación se lleva a cabo mediante una línea recta. Para tres entradas, la separación se realiza mediante un plano en el espacio tridimensional, y así sucesivamente hasta el caso de N entradas, en el cuál el espacio N-dimensional es dividido en un hiperplano.

2.3.5. EL Perceptron multicapa.

En un principio el desarrollo del Perceptrón llevó a la generación de un nuevo tipo de red cuya modificación principal respecto a la estructura del Perceptrón se basa en el uso de varias capas de neuronas artificiales, en vez de usar una sola capa. Este hecho significativo no hubiera servido de nada sin el cambio de la función de activación de las neuronas artificiales pasando de una función no diferenciable como era la activación logística a una función diferenciable y no lineal como lo es la sigmoide.

El recurso de este tipo de función de activación introdujo un nuevo paradigma en el procesamiento de los sistemas neuronales permitiendo a las redes neuronales aprender las variaciones no lineales de los distintos tipos de ambientes, que, en su mayoría, presentan variaciones del tipo no lineal.

En este momento se puede entender la importancia de este suceso ya que la mayor parte del tiempo el flujo de datos en las redes de comunicaciones sucede aleatorio y discontinuo.

Este tipo de características son precisamente las que el nuevo sistema neuronal nos permitirá asimilar. Este sistema neuronal considerado también una red neuronal se conoce en la literatura como Perceptrón Multicapa debido a que parte del principio del Perceptrón simple. A continuación, se muestra la arquitectura de esta red neuronal:

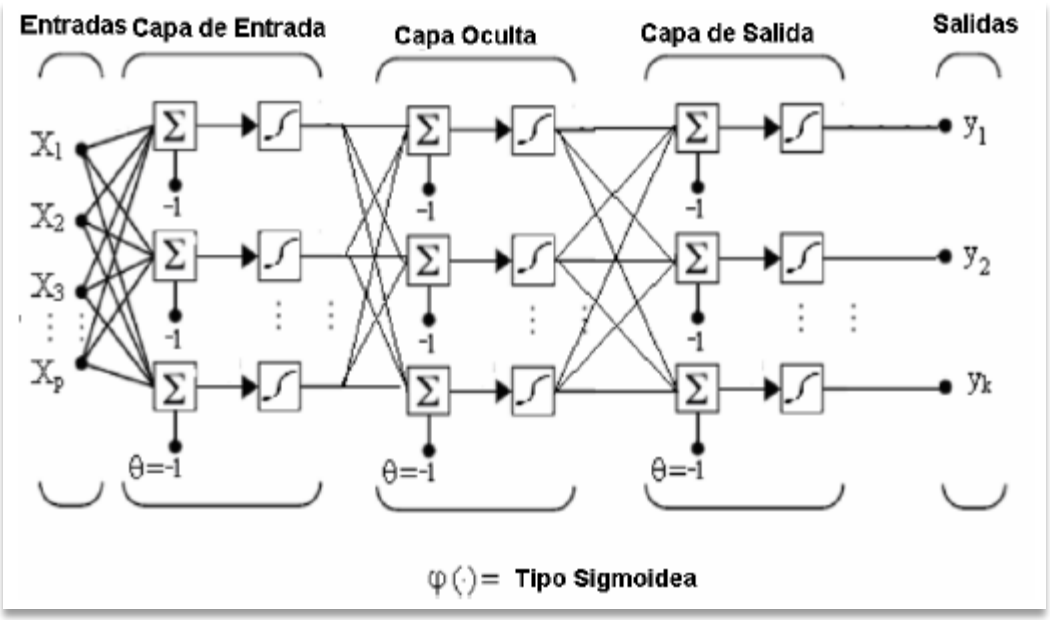


Figura 17. Arquitectura Perceptron Multicapa.

Al observar la arquitectura del Perceptrón Multicapa se puede observar que las múltiples entradas conectadas en la primera capa son mapeadas en las salidas en función de las distintas capas de neuronas intermedias y de los parámetros libres de la red. Se puede ver como una caja negra que realiza una operación sobre las entradas, produciendo un rango de salidas en función de los parámetros libres.

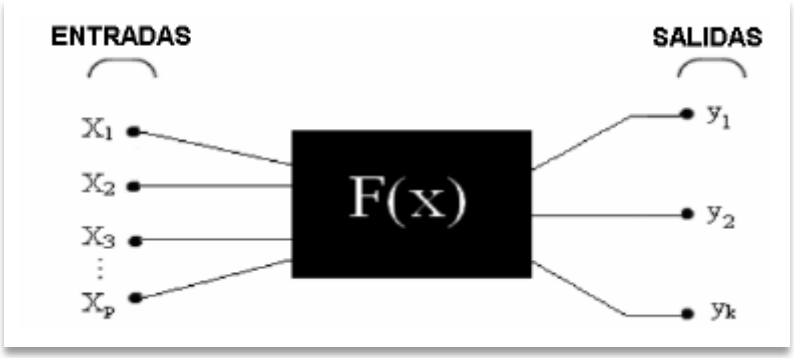


Figura 18. Perceptron multicapa visto como una caja negra.

Se infiere de la arquitectura que el algoritmo de entrenamiento de una red con tales características deberá ser planeado para que los cambios en los parámetros libres sean tales que el error en las unidades básicas de la estructura sea mínimo de manera que el conjunto de los cambios produzca un error global que tienda al mínimo. Se buscará entonces el límite en el cual la configuración de los parámetros libres produzca un error mínimo.

Tomando en cuenta este razonamiento, la evolución al Perceptrón Multicapa tuvo que basar su éxito en el diseño del algoritmo de entrenamiento que lograra minimizar el error al modificar adecuadamente los pesos y umbrales. La historia marco como primer paso, estudiar la forma de minimizar el error en una capa de neuronas lineales, que se conoce también como filtro lineal.

El análisis de este tipo de red neuronal que posee elementos lineales nos permite deducir un algoritmo más complejo para entrenar a una red como la Perceptrón Multicapa que posee elementos no lineales.

Las limitaciones del Perceptron multicapa son:

- No extrapola bien, es decir, si la red se entrena mal o de manera insuficiente, las salidas pueden ser imprecisas.

- La existencia de mínimos locales en la función de error dificulta considerablemente el entrenamiento, pues una vez alcanzado un mínimo el entrenamiento se detiene aunque no se haya alcanzado la tasa de convergencia fijada.

2.3.6. Redes ADALINE y MADELINE.

Las redes ADALINE (ADAptative LINear Element) y MADELINE (Multiple ADALINE) fueron desarrolladas por Bernie Widrow en la Universidad de Stanford poco despés de que Rosenblatt desarrollara el Perceptron. Las arquitecturas de ADALINE y MADELINE son esencialmente las mismas que las de Perceptron.

Ambas estructuras usan neuronas con funciones de transferencia escalón. La red ADALINE está limitada a una única neurona de salida, mientras que MADELINE puede tener varias. La diferencia fundamental respecto al Perceptron se refiere al mecanismo de aprendizaje.

ADALINE y MADELINE utilizan la denominada regla "*Delta de Widrow-Hoff*" o regla del mínimo error cuadrado medio (LMS), basada en la búsqueda de una expresión del error entre la salida deseada y la salida lineal obtenida antes de aplicarle la función de activación escalón (frente a la salida binaria utilizada en el caso del Perceptron).

Debido a esta nueva forma de evaluar el error, estas redes pueden procesar información analógica, tanto de entrada como de salida, utilizando una función de activación lineal o sigmoidal.

En cuanto a la estructura de la red ADALINE, que es casi idéntica a la del Perceptron elemental, sus autores la consideran formada por un elemento denominado *combinador adaptativo lineal* (ALC), que obtiene una salida lineal (s) que puede ser aplicada a otro elemento de conmutación bipolar, de forma que si la salida del ALC es positiva, la salida de la red ADALINE es 1; si la salida es negativa, entonces la salida de la red ADALINE es -1; tal y como se muestra en la siguiente figura:

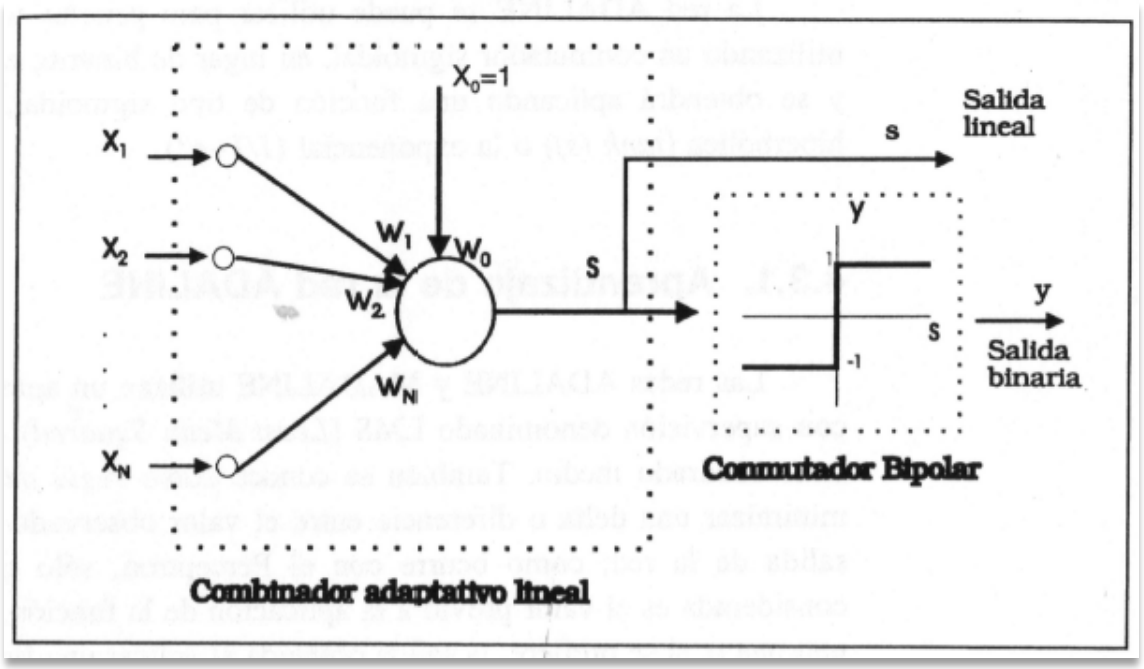


Figura 19. Estructura de la red ADALINE

El ALC realiza el cálculo de la suma ponderada de las entradas:

$$s = w_0 + \sum_{j=1}^{N} w_j w_j$$

Como en el caso del Perceptron, el umbral de la función de transferencia se representa a través de una conexión ficticia de peso w_0. Si tenemos en cuenta que para esta entrada se toma el valor de $x_0=1$, se puede escribir la anterior ecuación de la forma:

$$s = w_0 + \sum_{j=1}^{N} w_j w_j = X \cdot W^T$$

Esta es la salida lienal que genera el ALC. La salida binaria correspondiente de la red ADALINE es, por tanto:

$$y(t+1) = \begin{cases} +1 & S > 0 \\ y(t) & S = 0 \\ -1 & S > 0 \end{cases}$$

La red ADALINE se puede utilizar para generar una salida analógica utilizando un conmutador sigmoidal, en lugar de binario; en tal caso, la salida se obtendrá aplicando una función de tipo sigmoidal, como la tangente hiperbólica o la exponencial.

2.3.7. Red Backpropagation

En 1986, Rumelhart, Hinton y Williams [15],basándose en los trabajos de otros investigadores [16] formalizaron un método para que una red neuronal aprendiera la asociación que existe entre los patrones de entrada a la misma y las clases correspondientes, utilizando más niveles de neuronas que los que utilizó Rosenblatt para desarrollar el perceptron. Este método, conocido en general como *backpropagation* (propagación del error hacia atras), está basado en la generalización de la regla delta y, a pesar de sus propias limitaciones, ha ampliado de forma considerable el rango de aplicaciones de las redes neuronales.

El algoritmo de propagación hacia atrás, o retropropagación, es una regla de aprendizaje que se puede aplicar en modelos de redes con más de dos capas de células. Una característica importante de este algoritmo es la representación interna del conocimiento que es capaz de organizar en la capa de entrada y la salida de la red.

De forma simplificada, el funcionamiento de una red *backpropagation* consiste en un aprendizaje de un conjunto predefinido de pares de entradas-salidas dados como ejemplo, empleando un ciclo *propagación-adaptación* de dos fases: primero se aplica un patrón de entrada como estímulo para la primera capa de las neuronas de la red, se va propagando a través de todas las capas superiores hasta generar una salida, se compara el resultado obtenido en las neuronas de salida con la salida que se desea obtener y se calcula un valor del error para cada neurona de salida.

A continuación, estos errores se transmiten hacia atrás, partiendo de la capa de salida, hacia todas las neuronas de la capa intermedia que contribuyan directamente a la salida, recibiendo el porcentaje de error aproximado a la participación de la neurona intermedia en la salida original. Este proceso se repite, capa por capa, hasta que todas las neuronas de la red hayan recibido un error que describa su aportación relativa al error total.

Basándose en el valor del error recibido, se reajustan los pesos de conexión de cada neurona, de manera que en la siguiente vez que se presente el mismo patrón, la salida esté más cercana a la deseada, es decir, el error disminuya.

La importancia de la red *backpropagation* consiste en su capacidad de autoadaptar los pesos de las neuronas de las capas intermedias para aprender la relación que existe entre un conjunto de patrones dados como ejemplo y sus salidas correspondientes.

Para poder aplicar esa misma relación, después del entrenamiento, a nuevos vectores de entrada con ruido o incompletas, dando una salida activa si la nueva entrada es parecida a las presentadas durante el aprendizaje. Esta característica importante, que se exije a los sistemas de aprendizaje, es la capacidad de *generalización*, entendida como la facilidad de dar salidas satisfactorias a entradas que el sistema no ha visto nunca en su fase de entrenamiento.

La red debe encontrar una representación interna que le permita generar las salidas deseadas cuando se le dan entradas de entrenamiento, y que pueda aplicar, además, a entradas no presentadas durante la etapa de aprendizaje para clasificarlas según las características que compartan con los ejemplos de entrenamiento.

2.4. Redes Recurrentes.

Los problemas de predicción constituyen una clase especial de los problemas de aproximación funcional en los que los valores de las variables de salida (predicciones) se determinan a partir de valores de salida obtenidos en instantes anteriores. Por ejemplo, la predicción de los índices bursátiles basándose en valores anteriores.

Para tratar con esta clase de problemas se introducen unidades de proceso que evolucionan en tiempo discreto, es decir, en el tiempo $t \in \{0,1,2,\ldots\}$, de manera sus salidas son transmitidas como entradas en el tiempo $t+1$. Se comportan así como un autómata finito. Sin embargo, el problema está en cómo entrenar dicho autómata para conseguir las secuencias de valores de salida deseados. Las redes neuronales recurrentes constan de una capa de sensores (recogen la información de las entradas) una capa intermedia de unidades ocultas y una capa de unidades de salida.

Además, contienen conexiones sinápticas que van de los sensores de entrada a las unidades ocultas (también pueden ir directamente a las unidades de salida) y de las unidades ocultas a las unidades de salida, y se permiten conexiones sinápticas entre las unidades de proceso de una misma capa, sobre todo entre las unidades de la capa oculta.

Se han propuestos varios modelos de redes recurrentes juntos con sus algoritmos de entrenamiento, como los modelos de Elman (1990) [17], Jordan (1986) [18], Pineda (1987) [19],o de Rumelhart [15], Hinton y Williams (1986) cuyo algoritmo de aprendizaje es esencialmente el algoritmo de retropropagación.

2.5. Redes de Hopfield.

Sin duda, uno de los principales responsables del desarrollo que ha experimentado el campo de la computación neuronal ha sido J. Hopfield [20], quien construyó un modelo de red con el número suficiente de simplificaciones como para poder extraer analíticamente información sobre las características relevantes del sistema, conservando las ideas fundamentales de las redes construidas en el pasado y presentando una serie de funciones básicas de sistemas neuronales básicas.

Con su aportación, Hopfield redescubrió el mundo casi olvidado de las redes autoasociativas, caracterizadas por una nueva arquitectura y un nuevo funcionamiento, a las que tuvo que añadir otro tipo de reglas de aprendizaje.

La Red de Hopfield es una red recurrente, es decir, existe realimentación entre las neuronas. De esta forma, al introducir un patrón de entrada, la información se propaga hacia adelante y hacia atrás, produciéndose una dinámica (se puede ver en la Figura 21). En algún momento, la evolución se detendrá en algún estado estable. En otros casos, es posible que la red no se detenga nunca.

Las redes recurrentes deben cumplir tres objetivos:

1. Dado cualquier estado inicial, deben converger siempre a un estado estable.

2. El dominio de atracción de cada estado estable debe estar perfectamente delimitado y cumplir algún criterio de métrica (por ejemplo, que el estado final sea el más cercano al inicial).

3. Debe poder tener cualquier número de estados estables.

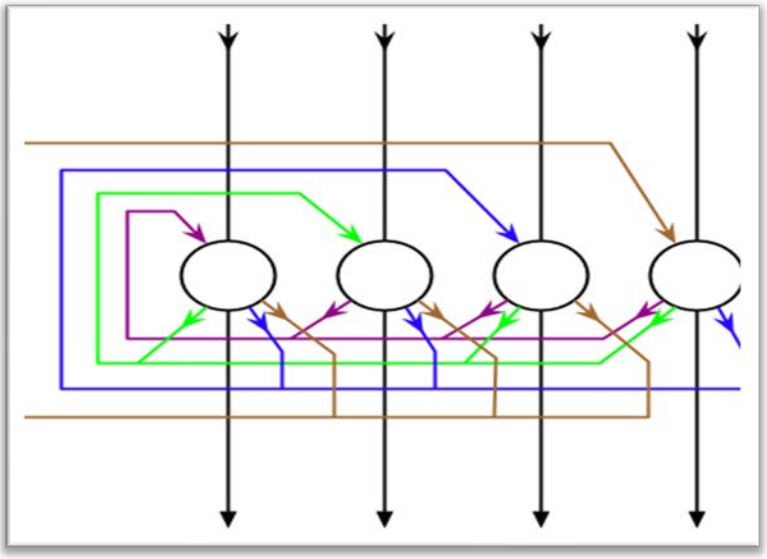


Figura 20. Estructura de una red de Hopfield.

2.6. Redes Feedforward.

Como su nombre lo indica, en este tipo de redes se empieza con un vector de entradas el cual es equivalente en magnitud al número de neuronas de la primera capa de la red, las cuales procesan dicho vector elemento por elemento en paralelo.

La información, modificada por los factores multiplicativos de los pesos en cada neurona, es transmitida hacia delante por la red pasando por las capas ocultas (si hay) para finalmente ser procesada por la capa de salida. Es por eso que este tipo de redes reciben su nombre.

Es importante mencionar que las redes *feedforward* son las más sencillas en cuanto a implementación y simulación, pero su desempeño es bueno para aplicaciones en los que no se requiera que la red retenga información de eventos pasados como ayuda para evaluar eventos futuros.

Cada vector de entrada presentado como entrenamiento para este tipo de redes es una entidad aislada del resto y, al final de dicho periodo de prueba, la red estará lista para comenzar a identificar y clasificar patrones, reconocer imágenes o cualquier otra aplicación que se le quiera dar.

Al iniciar las investigaciones sobre redes neuronales, las redes *feedforward* fueron las que recibieron más atención de parte de los investigadores porque sus características en cuanto a tiempos de procesamiento hacían viables simulaciones con los equipos computacionales de la época. Comparadas con las otras redes, las FFNN *(Feedforward Neural Network)* son una opción cuyo balance costo-velocidad y costo-exactitud es tal que da mayor ventaja al costo que a los otros parámetros.

En este tipo de redes no existen interconexiones entre capas más allá de la conexión directa hacia delante para propagar la información. No hay rutas de retroalimentación para desempeñar la función de memoria de la red.

En la siguiente figura se aprecia una FFNN (*feedforward neural network)* en donde todas las neuronas de una capa están interconectadas con las neuronas de la capa siguiente, iniciando con la capa principal y los elementos del vector *Xk*, proporcionando su información (las salidas *smn(t+1*)) propagándola hacia delante dentro de la red. El vector de pesos *Wk* es actualizado conforme los *epochs* pasan mientras el entrenamiento prosigue, y al final del mismo los pesos individuales *w11… w1n, w21…w2n*, etc, asumen sus valores finales para iniciar el trabajo de la FFNN con datos de entrada nuevos una vez que se ha llegado a una o varias salidas globales *son(t+1)*. La función de umbral *signum* (±1) se encuentra a la salida de la red para convertir *son(t+1)*en el valor final *yk.*

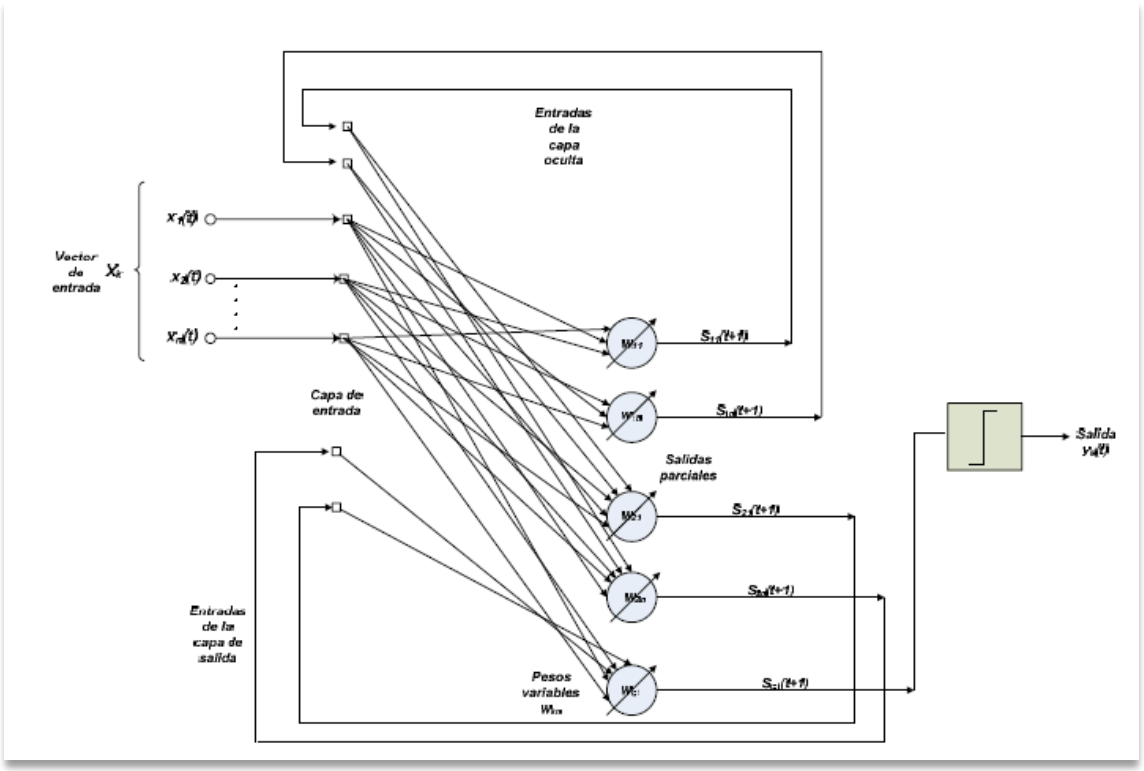


Figura 21. ejemplo de una arquitectura de red feedforward.

2.7. Redes de Elman.

Las redes *feedforward* tienen ciertas limitaciones inherentes a su diseño que pueden ser mejoradas con un cambio de arquitectura. Las redes Elman [17] son conocidas como redes recurrentes simples (SRN) y son precisamente una mejora de las redes *feedforward* debido a la inclusión de retroalimentación entre las capas inmediatas contiguas.

Los nombres de las variables (ver FIGURA X) corresponden a las de la Figura X-1(Redes feedforward), pero en este caso la retroalimentación simple hace que dichas redes posean una memoria de los eventos inmediatos anteriores, y éstos afectan las actualizaciones sucesivas de los pesos en cada una de las capas de la red.

De esta manera, los algoritmos de aprendizaje utilizados pueden mejorar el desempeño del sistema de detección de intrusos para conseguir mejores porcentajes generales al final de una sesión de entrenamiento con un conjunto de datos igual al de la red *feedforward.*

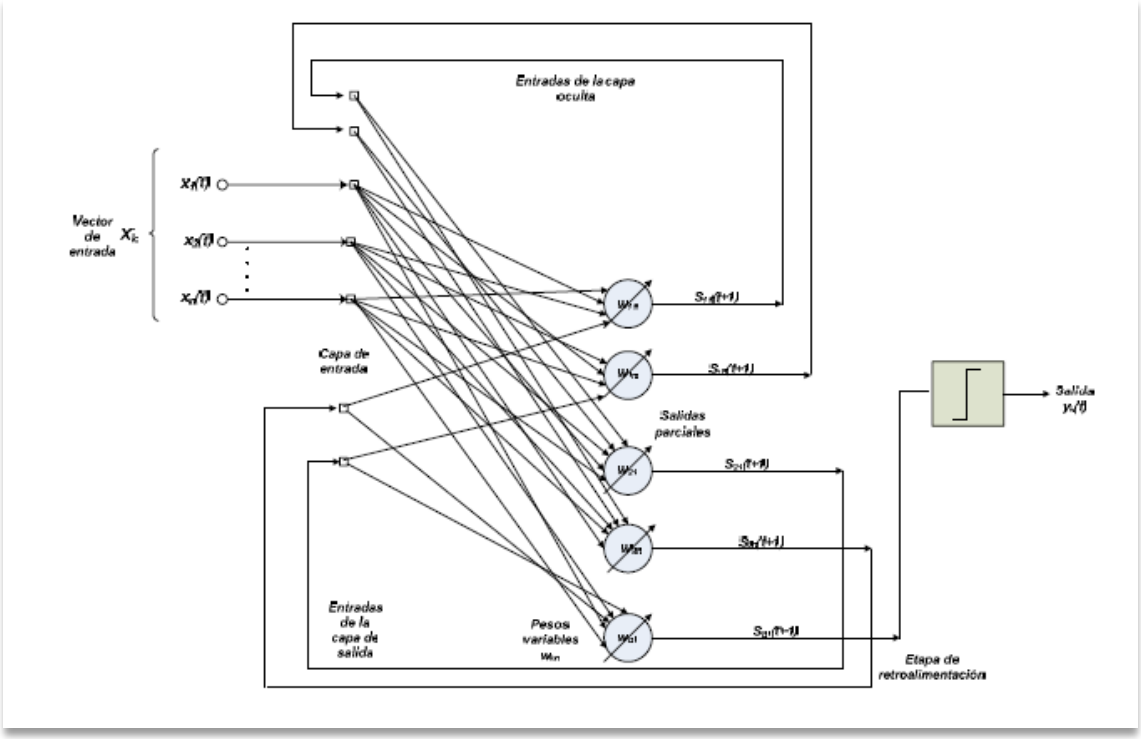


Figura 22. ejemplo de arquitectura de una red de Elman

Las redes Elman poseen varias características que las hacen superiores a las FFNN. En primer lugar, para una meta de desempeño dada con un margen de error pueden converger a dicho valor más rápido que las redes *feedforward.*

Eso significa que el número de iteraciones que los datos de entrenamiento y posteriormente los datos de prueba deben realizar dentro de la red es menor. También, el porcentaje de efectividad de estas redes es significativamente menor al de las FFNN debido a que los caminos de retroalimentación generan un mejor comportamiento no-lineal durante el aprendizaje supervisado.

Es menos probable que un IDS basado en redes Elman identifique un ataque cuando en realidad se trata de código normal, o que una vez identificado el ataque lo clasifique en una categoría errónea.

Sin embargo, estas ventajas tienen un costo importante: el tiempo de procesamiento. Las redes Elman tienen un tiempo de procesamiento mayor que el de las redes *feedforward*, y esta característica las hace más lentas en aplicaciones donde el número de neuronas es grande tanto en las capas de entrada y salida como en las capas ocultas.

De hecho, el incluir más de una capa oculta en una red Elman para la resolución de un problema sólo debe ser contemplado si la complejidad del mismo es substancial.

A modo de esquema gráfico, se puede visualizar la arquitectura de una red parcialmente recurrente de Elman en la siguiente imagen.

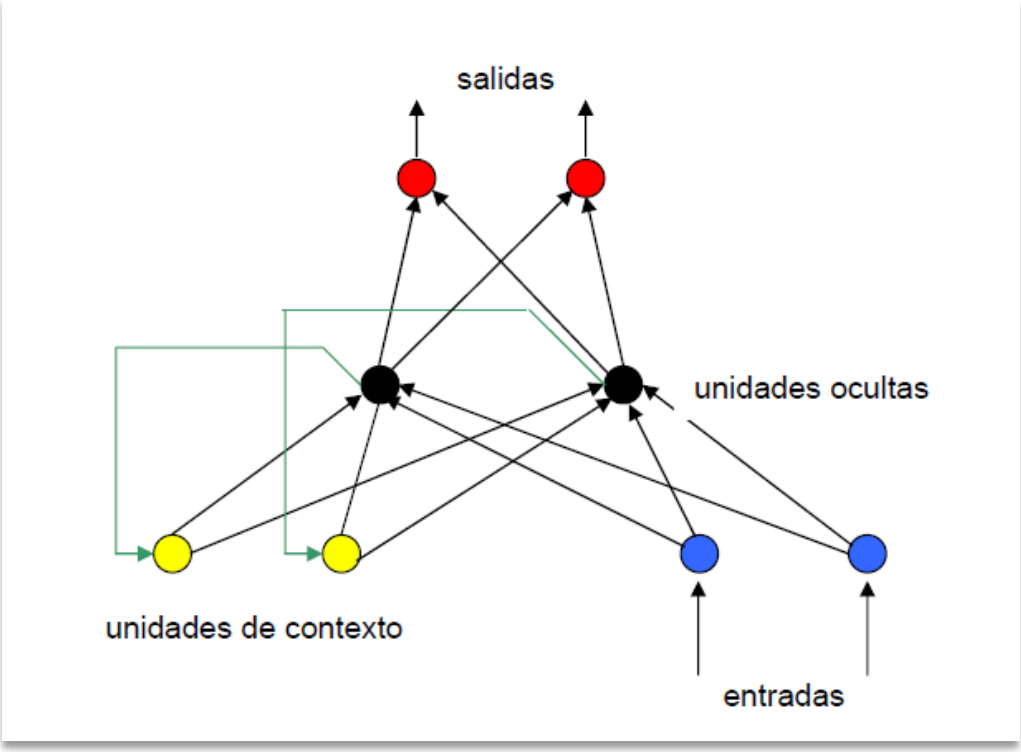


Figura 23. ejemplo de arquitectura de una red de Elman

En conclusión, este tipo de redes son un compromiso entre desempeño y rapidez. No son tan eficaces como las redes neuronales recurrentes completamente conectadas, pero tampoco requieren tiempos de procesamiento tan alto como ellas.

En cuanto a las redes *feedforward,* hay aplicaciones como las de reconocimiento de imágenes (lectores ópticos, reconocimiento de firmas, fotografía) en las que la no-linearidad del problema es un factor secundario y en el que excesivo tiempo de procesamiento sería perjudicial para la solución del problema.

2.8. Red de Jordan.

La Red Jordan se diferencia a la anterior, en que en vez de tomar la realimentación de la salida de las capas enterradas, se toma de la salida de la propia Red. Principalmente una de las ventajas por la que se a elegido este tipo de Red, como se comentó anteriormente, es por el ahorro de tiempo de cálculo y que respecto a la Red Elman es más rápida a la hora de la simulación.

El tipo de entrenamiento que se lleva a cabo es el backpropagation con velocidad de aprendizaje variable, de un número máximo de epoch's a 0, durante un número determinado de epoch's.

2.9. Red recurrente completamente conectada.

Las redes neuronales recurrentes completamente conectadas, a diferencia de las redes Elman, tienen caminos de retroalimentación entre todos los elementos que las conforman. Una sola neurona está entonces conectada a las neuronas posteriores en la siguiente capa, las neuronas pasadas de la capa anterior y a ella misma a través de vectores de pesos variables que sufren alteraciones en cada *epoch* con el fin de alcanzar los parámetros o metas de operación.

La complejidad de este tipo de redes es alta en comparación con una red *feedforward*, por ejemplo, ya que en esta última la red sólo es capaz de transmitir la información hacia las capas siguientes resultando en un efecto de propagación hacia atrás *en el tiempo.*

Las redes neuronales recurrentes, en cambio, realizan el intercambio de información entre neuronas de una manera mucho más compleja y por sus características, dependiendo del tipo de algoritmo de entrenamiento que se elija, pueden propagar la información hacia delante en el tiempo, lo cual equivale a predecir eventos.

Esta es una característica muy importante para ciertas aplicaciones, como los IDS, ya que la capacidad de predicción de eventos significativos (en este caso, ataques a la red) basada en las entradas anteriores al sistema le proporciona un beneficio importante a la seguridad del mismo.

La arquitectura básica de una RNN se muestra en la figura 26. Una característica importante es la inclusión de *delays (z-1)* a la salida de las neuronas en las capas intermedias; las salidas parciales *smn(t+1)* se convierten en valores *smn(t),* un instante de tiempo anterior, y así se retroalimentan a todos los componentes de la red, guardando información de instantes de tiempo anteriores.

Puede observarse cómo todos los nodos están interconectados entre sí y también con los nodos anteriores a ellos a través de conexiones directas y también *delays* antes de cada capa, o memorias temporales. El diagrama ha sido simplificado para no incurrir en excesiva complejidad, pero cada una de las capas está representada por cierto número de neuronas.

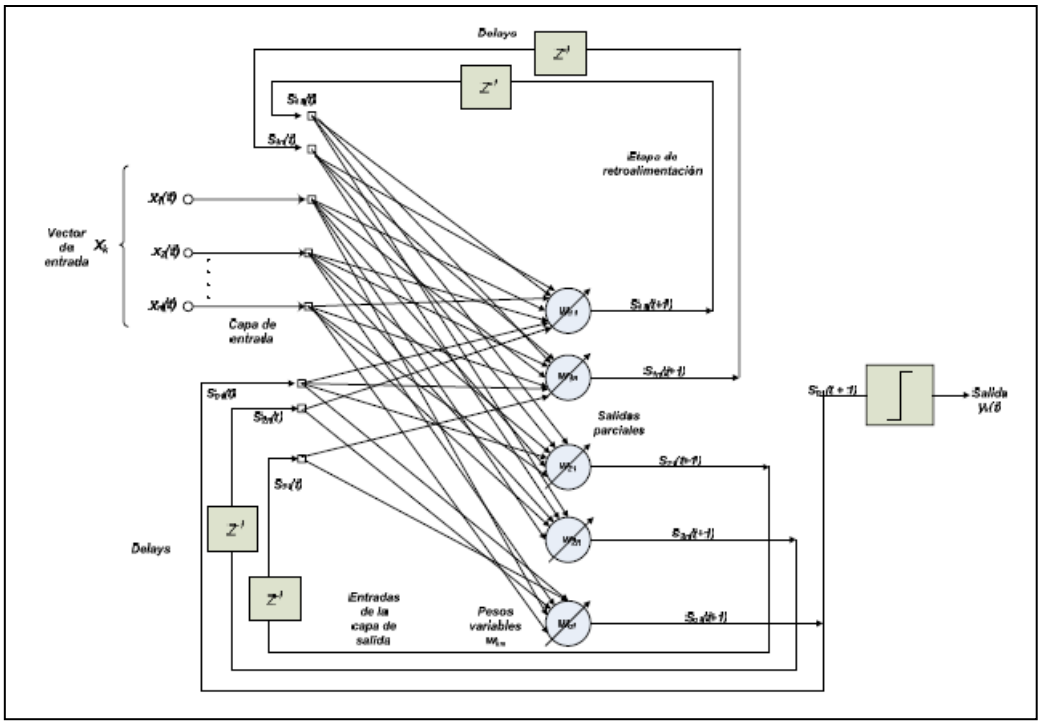


Figura 24. ejemplo de arquitectura de una red recurrente completamente conectada

Las redes neuronales recurrentes son más eficaces para resolver problemas con no-linealidades temporales significativas. Son especialmente útiles en aplicaciones tales como el reconocimiento de patrones secuenciales, cambiantes en el tiempo, ya que las capacidades de predicción y mapeo de las RNN así lo permiten.

Se ha hecho énfasis en que, a diferencia de las redes *feedforward* en las que la meta es que la red converja a un valor de estados fijo en cierto periodo de tiempo, las RNN tienen comportamiento variable en el tiempo y esto ofrece posibilidades de resolución de problemas diferentes a aquellos con una arquitectura tradicional.

En los sistemas biológicos, los cuales fueron las bases conceptuales de las redes neuronales, el número de interconexiones entre todas las neuronas es muy grande. Las RNN pueden aproximarse más a ese comportamiento que los demás tipos de redes, pero por lo general la complejidad intrínseca de éstas requiere tiempos de procesamiento muy superiores.

Las características variables de sus estados internos a través del tiempo también hacen muy importante considerar en qué momento deben actualizarse los pesos: al final de cada *epoch* (*epoch-wise training)* o continuamente. En el segundo caso, encontrar el instante adecuado para la actualización es un reto significativo.

A modo de resumen y curiosidad en algunos casos, se muestra a continuación una serie de tablas donde se recogen los modelos de redes neuronales más importantes iniciales y que posteriormente han dado lugar a nuevas variaciones de dichos modelos, permitiendo así adaptar cada modelo a los nuevos casos concretos de aplicación.

De este modo, actualmente se tiene un modelo óptimo para cada caso de aplicación.

Modelo de Red	Topología	Aprendizaje			Asociación Auto/Hetero	Información de entrada y salida	Autor(es)
		On/Off Line	Supervisa / No supervisa	Regla			
ADALINE/MADELINE	2 capas - Feedforward	Off	Supervisado	Corrección error ("Least Mean Square")	Heteroasociado	E: Analógica, S: Binaria	Widrow Hoff 1960
Adaptive Bidireccional Associatiive Memory. Aba,.	2 capas - Feedforward/Feedback	On	No Supervisado	Hebbiano diferencial	Heteroasociado	Analogica	Kosko 1987
Adaptive Heuristic Crtitic- AHC	3 Capas - Feedforward	On	Supervisado	Refuerzo por ajuste lateral	Heteroasociado	E: Analógica, S: Binaria	Barto 1983
Adaptive Resonance Theory 1.ART1	2 capas - Feedforward/Feedback - Conex. Laterales	On	No Supervisado	Competitivo (Resonancia) Adaptativa	Heteroasociado	Binarias	Carpenter 1986
Adaptive Resonance Theory 2.ART2	2 capas - Feedforward/Feedback - Conex. Laterales	On	No Supervisado	Competitivo (Resonancia) Adaptativa	Heteroasociado	Analogica	Carpenter 1986

Tabla 3: Clasificación de las redes neuronales artificiales más importantes.

Modelo de Red	Topología	Aprendizaje			Asociación Auto/Hetero	Información de entrada y salida	Autor(es)
		On/Off Line	Supervisa / No supervisa	Regla			
Aditive Grossberg	1 Capa Conex. Laterales auto-recurrentes	On	Supervisado	Hebbiano Competitivo	Autoasociado	Analogica	Grossberg
Associative Reward Penalty. ARP	2 capas - Feedforward	On	Supervisado	Refuerzo estocástico	Heteroasociado	E: Analógica, S: Binaria	Barto 1985
Back-Propagation	N capas - Feedforward	Off	Supervisado	Corrección error (regla delta)	Heteroasociado	Analogica	Rumelhart 1986
Recurrent-Backpropagation	N capas - Feedforward/Feedback	Off	Supervisado	Corrección error (regla delta)	Heteroasociado	Analogica	Pineda 1987
Boltzman Machine	1 Capa conex. Laterales - 3 capas feedforward	Off	Supervisado	Estocástico	Heteroasociado	Binarias	Hinton Ackley Sejnowski 1984
Bidirectional Associative Memory Bam.	2 capas - Feedforward/Feedback	On	No Supervisado	Hebbiano	Heteroasociado	Binarias	Kosko 1988

Tabla 4: Clasificación de las redes neuronales artificiales más importantes.

Modelo de Red	Topología	Aprendizaje			Asociación Auto/Hetero	Información de entrada y salida	Autor(es)
		On/Off Line	Supervisa / No supervisa	Regla			
Fuzzy Associative Memory. FAM	2 capas - Feedforward/Feedback	Off	No supervisado	Hebbiano borroso	Heteroasociado	Analogica	Kosko 1987
Continous Hopfield	1 Capa Conex. Laterales	Off	No supervisado	Hebbiano	Autoasociado	Analogica	Hoppfield 1984
Discrete Hopfield	1 Capa Conex. Laterales	Off	No supervisado	Hebbiano	Autoasociado	Binarias	Hoppfield 1982
Learn Matrix LM	1 Capa Crossbar	Off	No supervisado	Hebbiano	Heteroasociado	Binarias	Steinbuch 1961
Learning Vector Quantizer. LVQ	2 capas - Feedforward/Feedback - Conex. Laterales	Off	No supervisado	Competitivo	Heteroasociado	Analogica	Kohonen 1981
Linear Associative Memory. LRP	2 capas - Feedforward	Off	No supervisado	Hebbiano	Heteroasociado	Analogica	Anderson 1968

Tabla 5: Clasificación de las redes neuronales artificiales más importantes.

2.10. Aplicaciones de las redes neuronales.

Las redes neuronales pueden utilizarse en un gran número y variedad de aplicaciones, tanto comerciales como militares [6].

Se pueden desarrollar redes neuronales en un periodo de tiempo razonable, con la capacidad de realizar tareas concretas mejor que otras tecnologías.

Cuando se implementan mediante hardware (redes neuronales en chips VLSI), presentan una alta tolerancia a fallos del sistema y proporcionan un alto grado de paralelismo en el procesamiento de datos. Esto posibilita la inserción de redes neuronales de bajo coste en sistemas existentes y recientemente desarrollados.

Hay muchos tipos diferentes de redes neuronales; cada uno de los cuales tiene una aplicación particular más apropiada.

Algunas aplicaciones comerciales son:

- Biología:
  - Aprender más acerca del cerebro y otros sistemas.
  - Obtención de modelos de la retina.

- Empresa:
  -Evaluación de probabilidad de formaciones geológicas y petrolíferas.
  - Identificación de candidatos para posiciones específicas.

- Explotación de bases de datos.
- Optimización de plazas y horarios en líneas de vuelo.
-Optimización del flujo del tránsito controlando convenientemente la temporización de los semáforos.
- Reconocimiento de caracteres escritos.
- Modelado de sistemas para automatización y control.

- Medio ambiente:
  - Analizar tendencias y patrones.
  - Previsión del tiempo.

- Finanzas:
  - Previsión de la evolución de los precios.
  - Valoración del riesgo de los créditos.
  - Identificación de falsificaciones.
  - Interpretación de firmas.

- Manufacturación:
  -Robots automatizados y sistemas de control (visión artificial y sensores de presión, temperatura, gas, etc.).
  - Control de producción en líneas de procesos.
  - Inspección de la calidad.

- Medicina:
  - Analizadores del habla para ayudar en la audición de sordos profundos.
  - Diagnóstico y tratamiento a partir de síntomas y/o de datos analíticos (electrocardiograma, encefalogramas, análisis sanguíneo, etc.).
  - Monitorización en cirugías.
  - Predicción de reacciones adversas en los medicamentos.
  - Entendimiento de la causa de los ataques cardíacos.

  Militares:
  - Clasificación de las señales de radar.
  - Creación de armas inteligentes.
  - Optimización del uso de recursos escasos.
  - Reconocimiento y seguimiento en el tiro al blanco.

La mayoría de estas aplicaciones consisten en realizar un reconocimiento de patrones, como ser: buscar un patrón en una serie de ejemplos, clasificar patrones, completar una señal a partir de valores parciales o reconstruir el patrón correcto partiendo de uno distorsionado. Sin embargo, está creciendo el uso de redes neuronales en distintos tipos de sistemas de control.

Desde el punto de vista de los casos de aplicación, la ventaja de las redes neuronales reside en el procesado paralelo, adaptativo y no lineal.

El dominio de aplicación de las redes neuronales también se lo puede clasificar de la siguiente forma: asociación y clasificación, regeneración de patrones, regresión y generalización, y optimización.

MÁSTER EN SISTEMAS DE ENERGÍA TÉRMICA

CAPÍTULO 3:

OBTENCIÓN DEL MODELO

3.3. Objetivos

A lo largo de este capítulo se desarrollará un modelo matemático y óptimo mediante el uso de redes neuronales artificiales que permitirá estimar la producción de hidrógeno a una configuración de variables dada. Esta producción de hidrógeno estará basada en un proceso de obtención de este gas a partir del reformado de gas natural mediante vapor.

Para realizar el estudio se utilizarán las variables y registros facilitados por el tutor, estarán basados en datos reales, y que facilitarán, valga la redundancia, unos resultados más realistas sobre lo que sería el proceso de refino.

Previamente al estudio, se realizará un pre-procesado de los datos con la finalidad de asegurar la integridad de estos datos que se utilizarán en el modelo y mejorar además la eficiencia del proceso de estudio; de forma que se intentará reducir así la cantidad de información que se procesará, sin que se pierda precisión en el modelo posterior.

Para ello se realizará un estudio de los datos iniciales con los siguientes objetivos:

- Evitar datos que no aporten información al modelo.
- Evitar posibles ausencias de información.
- Reducir la dimensionalidad de los datos sin afectar al estudio posterior.
- Estudio del tamaño de la muestra.

Se mostrará en el anexo del proyecto el algoritmo implementado para la realización del presente estudio para cualquier comprobación.

3.4. Datos de partida

Los datos utilizados, como se ha comentado anteriormente, han sido facilitados por el tutor a partir de una base de datos real. Es por ello, por lo que los valores de estos datos se han considerado confidenciales, y no se mostrarán a lo largo del presente proyecto.

No obstante, se mostrarán las variables que han sido utilizadas para el estudio con el fin de poder tratar con cierta identidad los resultados que se reflejarán en el proyecto. De esta manera se identificarán las variables que intervienen en el proceso.

En la siguiente tabla se especifican las variables de entrada al proceso, así como la producción de hidrógeno a la salida del proceso.

	Variables	Descripción	Unidades
Entradas	T_GASPRO_IN_R3	Temperatura del gas producido en R3	ºC
	R_VAP/GASPRO_PREREF	relación vapor y gas producido de consigna	%
	T_GASPRO_OUT_H1	Temperatura del gas producido en H1	ºC
	R_VAP/GASPRO_TOTAL	Relación vapor y gas producido total	%
	T_GASPRO_OUT_E1	Temperatura del gas producido en E1	ºC
	FACTOR_OPERACIÓN	Factor de operación	ppm VOL
	T_ADESM_IN_V7	Temperatura en V7	ºC
	T_AGUA_IN_V5	Temperatura de agua en V5	ºC
	T_VAP_OUT_E24	Temperatura del vapor producido en E24	ºC
	T_GASPRO_IN_R1	Temperatura del gas producido en R1	ºC
	CAPACIDAD	Capacidad	-
	R_H2/GN_IN_PLANT	Relación de H2 y gas natural	%
	M_OFFGAS_OUT_V4	Caudal másico de gas producido en V4	[kg/h]
	PCI_GN	Valor de PCI del gas natural	[kcal/nm3]
	PCI_OFFG	Valor de PCI del gas natural	[kcal/nm3]
	P_GN_IN_VALVFC002	presión del gas natual en el punto FC0002	[kg/cm2g]
	T_GASPRO_OUT_R4	Tª del gas producido a la salida de R4	ºC
Salida	M_H2_OUT_E8B	Caudal másico de salida de H2	Kg/h

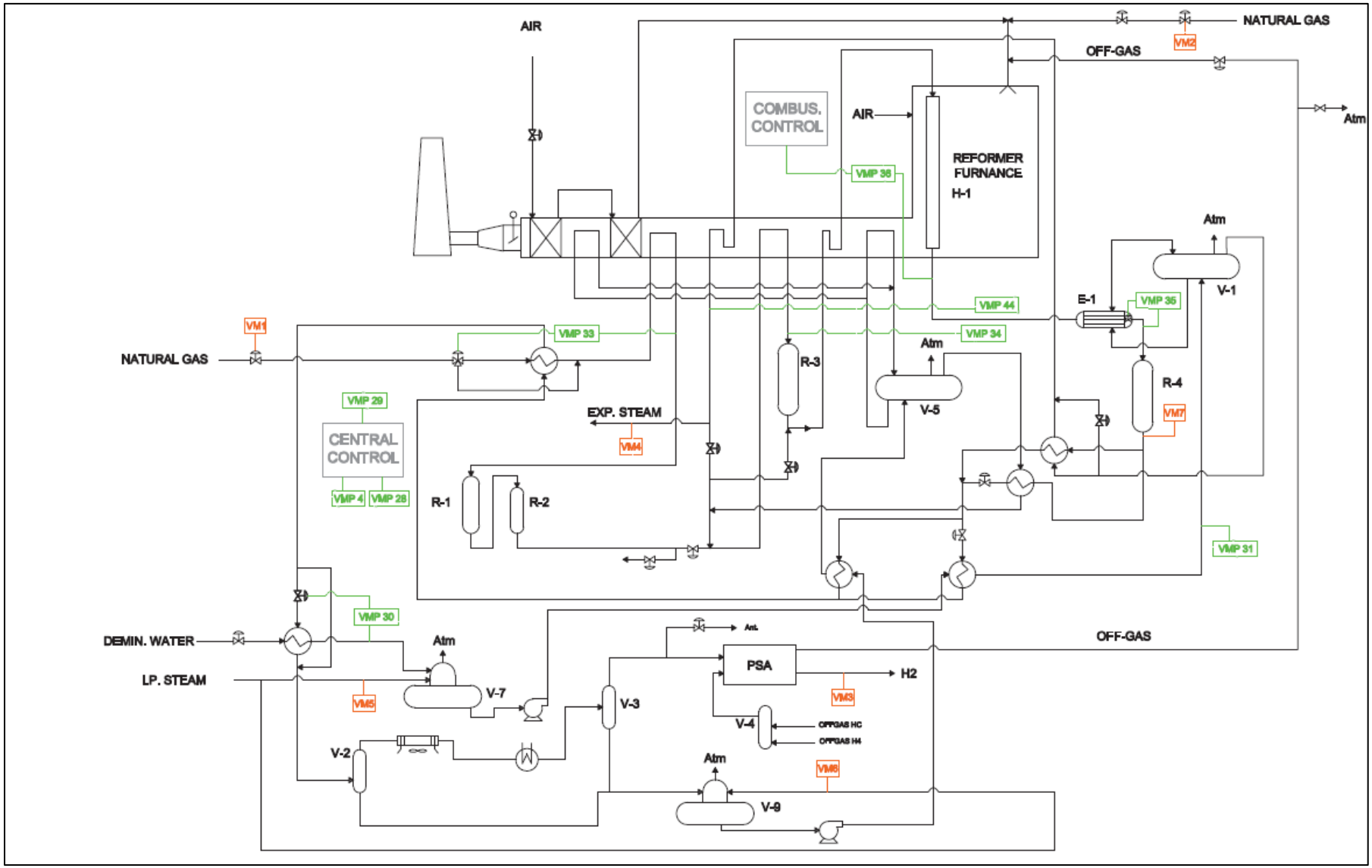
Tabla 6. Variables utilizadas en el proceso de obtención de hidrógeno.

Se puede observar que existen un total de 17 variables de entrada, y una sola única variable de salida. Aunque no aparecen en la table, cabe mencionar que se cuentan con un total de 3361 registros (valores) para cada una de esas variables.

De esta manera, quedará definida nuestra base de datos y que será utilizada en el presente proyecto para futuros estudios.

Las variables mostradas hacen referencia a diferentes puntos del proceso de obtención de hidrógeno.

Estos puntos se describen el en diagrama que se muestra a continuación.


AIR
NATURAL GAS
VM2
OFF-GAS
Atm
COMBUS. CONTROL
AIR
VMP 36
REFORMER FURNANCE
H-1
Atm
V-1
E-1
VMP 35
VMP 44
VMP 34
VM1
VMP 33
NATURAL GAS
R-3
Atm
V-5
R-4
VM7
VMP 29
EXP. STEAM
VM4
CENTRAL CONTROL
VMP 4
VMP 28
R-1
R-2
VMP 31
VMP 30
DEMIN. WATER
LP. STEAM
Atm
V-7
VM5
V-3
PSA
H2
VM3
V-4
OFFGAS HC
OFFGAS H4
OFF-GAS
V-2
Atm
VM6
V-9

3.5. Pre-procesado de los datos

Los datos que han sido proporcionados se han considerado como hipótesis de partida, correctos y adecuados en su totalidad para el estudio, y por tanto, no será necesario realizar un estudio discriminatorio de los datos para eliminar datos que pudieran ser susceptibles de no aportar información o modificar la información útil de manera errónea causando desviaciones en el modelo.

En muchas ocasiones, es necesario estudiar la muestra total de los datos de manera que se deben eliminar o no, aquellos datos que resulten anómalos a la distribución de estos, por las razones que sea. Las razones que suelen afectar a la aparición de estos datos "curiosos", pueden ser cambios puntuales de la operación de la planta, averías, cambios de equipos, desviaciones en la composición del gas u otros factores de carácter puntual, que si no se analizan minuciosamente, no es posible garantizar la integridad de todos los datos que se utilizarán.

Es por ello, que siempre se debe analizar los datos de partida y estudiar cuales son realmente válidos.

En este caso, no se realizará ningún estudio previo de estos datos y se considerarán que estos ya han sido filtrados, y, por tanto, serán válidos para su utilización en los futuros estudios del presente proyecto.

3.5.1. Reducción de la dimensionalidad

El objetivo de este apartado no es más que reducir el peso de los datos a tratar con la finalidad de mejorar la eficiencia en el proceso de obtención del modelo óptimo.

Para el estudio de tratar de reducir la dimensionalidad, se van a realizar dos métodos:

- Análisis de componentes principales (PCA, por sus siglas en inglés)
- Reducción del tamaño de la muestra

5.5.1.1 Análisis de componentes principales

El análisis de componentes principales (ACP) consiste en expresar un conjunto de variables en un conjunto de combinaciones lineales de factores no correlacionados entre sí. Este método permite representar los datos originales (individuos y variables) en un espacio de dimensión inferior al del espacio original, mientras se limita la pérdida de información.

En el análisis de componentes principales lo que hace es establecer una serie de combinaciones lineales de las características o variables de las que dispone el modelo. El grado de estas combinaciones va desde 1 hasta el número de variables presentes en los datos iniciales.

Posteriormente, se estudia mediante los autovalores y los auto vectores que nivel de representatividad tiene cada una de estas combinaciones en el espacio de origen, y en el espacio de la combinación.

Se debe establecer un frente de pareto entre % de representatividad y grado de reducción de la dimensionalidad, de manera que se pueda elegir la combinación de ambas más conveniente.

Tras realizar el estudio de PCA mediante Matlab, se obtiene que, concretamente nos quedamos con 8 características (una combinación lineal de grado 8), la cual tiene un porcentaje de representatividad del 94.7651 %.

Una vez tomada esa decisión de partida, se obtiene la matriz de transformación W con un tamaño de 8x17, siendo nuestra matriz de datos original de 8x3361.

Esta tiene dicho tamaño porque al multiplicar nuestra matriz de datos original, los cuales están expresados como vectores de 17x1, es multiplicada por una matriz de transformación W, obteniendo que W(8x17) * D(17x1) y, por tanto, se obtiene un vector de 8x1 que es el objetivo. De esta forma, se reduce la dimensionalidad, es decir reducir el número de características con las que se trabajarán.

La aplicación de dicha matriz sobre nuestros datos devuelve las imágenes de estos, en el espacio de la combinación elegida, la cual tendrá un número menor de variables, reduciendo con ellos el peso de los datos. Se habla de imágenes ya que estamos realizando un cambio de espacio del espacio de nuestros datos al espacio de los datos reducidos, eso se hace mediante una transformación lineal o matriz de cambio W, siendo los transformados de los puntos en el otro espacio las imágenes de estos.

Todo este proceso viene implementado en la función PCA de Matlab, dicha función nos brinda toda la información anteriormente nombrada.

Es por ello, por lo que se utilizará dicha función para implementar el método.

5.5.1.2 Tamaño de la muestra

En este apartado, se realizará un estudio de la muestra para ajustar el tamaño de la misma a las necesidades del proyecto.

Cabe mencionar que cuando se toma una muestra de un conjunto mayor, esta goza de cierta representatividad sobre el conjunto original, se dice que esta explica en mayor o menor grado el conjunto de datos del que procede.

Además, a partir de cierto tamaño de muestra la representatividad de esta se estanca y no crece más, es decir, se ha obtenido un subgrupo que representa muy bien a la población, y no se puede obtener un grupo reducido que la explique mejor.

Para explicar analíticamente esto es necesario introducir dos conceptos, error empírico y error de generalización.

- El **error empírico** es aquel obtenido a partir de probar un modelo generado a partir de los datos de una muestra, con los datos de esa misma muestra. Es, por tanto, un estimador del riesgo de predicción muy optimista, dado a que el modelo ha sido generado con los mismos datos con los que se prueba, es obvio que funcionará muy bien, pero esto no es realista.

- El **error de generalización** es aquel obtenido a partir de probar un modelo generado a partir de los datos de una muestra, con datos externos a esta muestra, pero provenientes de la misma población. Es, por tanto, un estimador del riesgo menos optimista, pero es mucho más fiable, puesto que prueba el desempeño real del modelo con datos externos.

Para realizar este estudio se han realizado estimaciones de la producción de hidrógeno utilizando un modelo de regresión de grado 1. Para ello, con los diferentes tamaños de muestra se ha utilizado el grado más pequeño puesto que lo que se pretende observar es indiferente al modelo de estimación que se utiliza en este estudio.

Para cada distinto tamaño de muestra se ha obtenido el error empírico y el error de generalización, para ello:

- En el caso del error empírico, se han calculado los coeficientes para una regresión de grado 1, una vez hallados se ha procedido a la estimación de la producción de hidrógeno con TODOS los datos de la muestra original.

- En el caso del error de generalización se ha realizado lo mismo, sólo que se ha dividido la muestra en dos conjuntos, uno de training y uno de test, de manera que los datos con los que se obtienen los coeficientes no son los mismos con los que se prueba la calidad de la estimación de la producción de hidrógeno.

Posteriormente se han graficado todos los valores de ambos errores para cada uno de los tamaños de muestra.

A continuación, se muestra la gráfica que representa el estudio del tamaño de la muestra, en la que se refleja el comportamiento del error empírico y de generalización conforme aumentan los datos de la muestra.

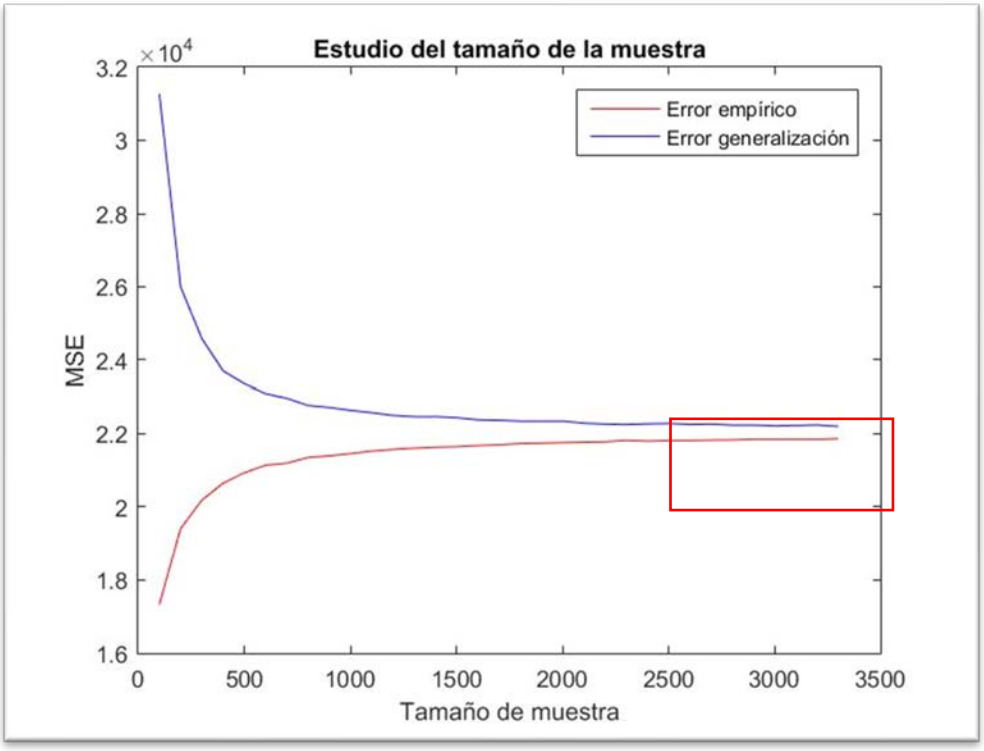


Figura 25. Estudio del tamaño de la muestra

Se puede observar que conforme crece el tamaño de la muestra, el error de generalización (azul) decrece hasta llegar a un punto en el que no se reduce más.

Ese es el tamaño de muestra óptimo, y en ese punto el error que se obtiene es el error subyacente en los datos o ruido, este es el error irreducible en cualquier modelo de Machine Learning.

Para nuestro estudio se conservará un tamaño de muestra de 2900 registros, reduciendo así levemente la población original de 3361 registros.

3.6. Estudio de la complejidad del modelo

A continuación, se va a desarrollar detalladamente todo el proceso por el cual se va a obtener el modelo óptimo de predicción para la obtención de hidrógeno de nuestro proceso.

Todo el proceso que se describe a continuación se ha implementado bajo la interface de Matlab, por lo que los resultados que se mostrarán serán obtenidos a partir de las simulaciones con este software.

En primer lugar, se establecerá la arquitectura de la red neuronal, tal y como se descrito el Capítulo 2 de este proyecto "*Estado del arte de las redes neuronales*".

En segunda instancia, se definirá el número de iteraciones óptimo que deberá realizarse para que el error sea mínimo, y no se produzcan efectos de sobre-entrenamiento.

3.6.1. Estudio de la arquitectura. Número de neuronas.

Para definir la arquitectura de una red neuronal deben definirse, tal y como se ha visto en el Capítulo 2, el número de capas ocultas y el número de neuronas por capa.

Como hipótesis de partida, cabe destacar que se ha establecido una sola capa oculta para la red, ya que, en estudios previos realizados por el tutor, la mejor configuración para este tipo de análisis de predicción se daba con una única capa oculta.

Es por ello, por lo que se prestablecerá como hipótesis inicial.

En primer lugar, se analizan la distribución de los datos de cada variable mediante el uso de cajas de bigotes. Estas son una presentación visual que describen varias características importantes, al mismo tiempo, tales como la dispersión y simetría de los datos.

Una gráfica de este tipo consiste en una caja rectangular, donde los lados más largos muestran el recorrido intercuartílico.

Este rectángulo está dividido por un segmento vertical que indica donde se posiciona la mediana y por lo tanto su relación con los cuartiles primero y tercero (se recuerda que el segundo cuartil coincide con la mediana).

Esta caja se ubica a escala sobre un segmento que tiene como extremos los valores mínimo y máximo de la variable.

Las líneas que sobresalen de la caja se llaman bigotes. Estos bigotes tienen un límite de prolongación, de modo que cualquier dato o caso que no se encuentre dentro de este rango es marcado e identificado individualmente

A continuación, se muestran las variables sin normalizar en este tipo de gráfico.

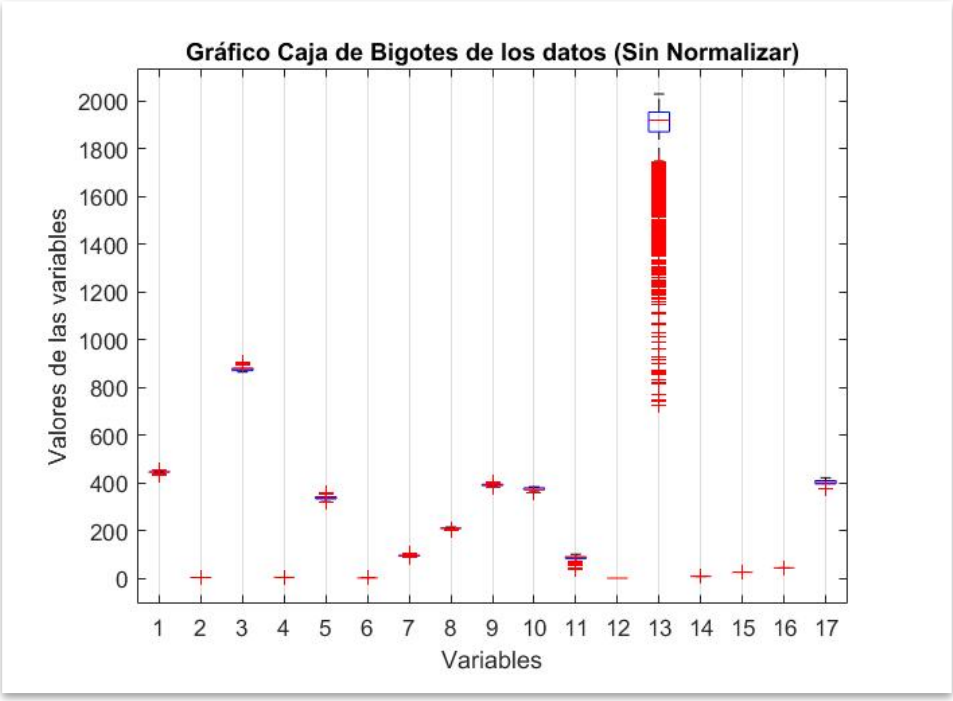


Figura 26. Gráfico de caja de bigotes con los datos sin normalizar

Como se puede observar, los rangos de los dominios de las variables son muy dispares, por lo que existirían variables que afectarían más al modelo que otras.

A continuación, se muestran las variables sin normalizar en este tipo de gráfico.

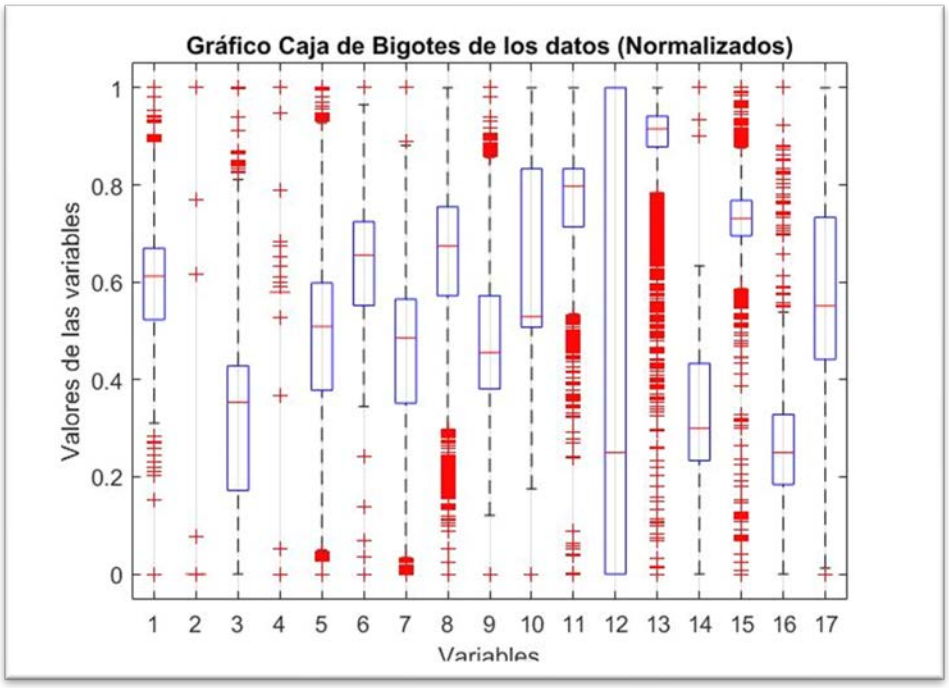


Figura 27. Gráfico de caja de bigotes con los datos normalizados

Como se puede observar, después de la normalización de los datos, los rangos de los dominios entre variables son ahora mucho más parecidos, por lo cual, el peso que egercerán cada una de las variables sobre el modelo será más equilibrado.

A continuación, se definen en el programa una serie de parámetros para configurar el algoritmo de entrenamiento del modelo. De este modo, se puede pre-seleccionar de forma manual una serie de parámetros que tendrán sus implicaciones en la simulación del algoritmo de entrenamiento.

Tal y como se ha descrito en el capítulo 2 "Estado del arte de las redes neuronales", es necesario definir el porcentaje de datos que será utilizado para entrenamiento, validación y test.

De este modo, se definirán los siguientes porcentajes de entrenamiento, de manera que seleccionará un 70%, 72,5% y 75% de los datos.

Los porcentajes restantes se utilizarán para validación y test, de manera que se estudiarán 3 casos para ver la influencia del porcentaje de entrenamiento en el modelo. El porcentaje de datos de entrenamiento puede mostrar en ocasiones efectos de sobre-aprendizaje o por el contrario, de infro-aprendizaje sí el porcentaje de entrenamiento no recoge los datos adecuados.

De este modo se obtiene que:

- 70 % entrenamiento, 15 % validación y 15 % test.
- 72,5 % entrenamiento, 13,75 % validación y 13,75 % test.
- 75 % entrenamiento, 12,5 % validación y 12,5 % test.

Para evaluar el rendimiento del algoritmo, se utilizará como error de validación para los datos de test el MSE (*Mean Square Error*). De esta forma, se evaluará el modelo con una serie de datos que no se han utilizado en el entrenamiento

Una vez definidos los porcentajes de entrenamiento, validación y test que se utilizarán en el algoritmo para el modelo, se establecen los parámetros que darán entidad a la arquitectura de la red.

De manera que se establece

- Número de capas ocultas: 1
- Número máximo de neuronas: 100

Se define un número máximo de hasta 100 neuronas porque el algoritmo realizará simulaciones para el modelo con una arquitectura tal que, esta tendrá una capa oculta e irá analizando desde 1 hasta 100 neuronas.

De este modo, se podrá comprobar como varía el error de validación y seleccionar la arquitectura óptima del modelo.

Para evitar mínimos locales se realizan algoritmos menos "codiciosos" o "ambiciosos", y, además, se debe repetir varias veces el experimento, aumentando las probabilidades de que una de estas iteraciones pueda saltar este mínimo local.

Se debe comentar, que en el algoritmo utilizado (que se muestra en el Anexo I) existen una serie de parámetros que son de tipo aleatorio, y, por tanto, en cada iteración dará unos valores distintos. Por ello, es necesario repetir el experimento varias veces.

Para conseguir un resultado más "fino", se repetirá la ejecución una serie de veces y luego se promedian los resultados, de manera que el resultado será más fiel a la realidad. De otro modo, si nos contentáramos con el resultado de la primera ejecución solamente, nos arriesgaríamos a que dicho resultado no fuera el más correcto, sino un resultado excepcional que solo se diera de vez en cuando.

Por ello, y como justificación de lo anterior, se establece para el algoritmo para evitar estos problemas y mejorar la eficiencia del modelo los siguientes parámetros:

- Número de Boostrap: 5
- Número de promedios: 10

Es por ello, por lo que se establece un total de 5 repeticiones con la misma configuración con el objetivo de evitar esos mínimos locales, y un total de 10 repeticiones adicionales de cada experimento con el fin de mejorar el rendimiento de la red neuronal.

Una vez se ha especificado la configuración de la arquitectura que definirán los modelos de redes neuronales para su estudio, debe especificarse el método de entrenamiento que se utilizará.

Tras una extensa búsqueda bibliográfica, y contrastándolo con la experiencia en procesos similares con el tutor, se estableció para este proyecto que el algoritmo de mejor convergencia y rapidez era el de Levenberg-Marquardt.

Existen números algoritmos de entrenamiento como se ha visto en el Capítulo 2, sin embargo, al no formar parte del presente proyecto, no se desarrollará el contenido de este algoritmo, ya que, además, Matlab lo incorpora en su programa. Por lo que se utilizará de forma sencilla simplemente especificándolo.

Una vez se ha definido la configuración de la arquitectura, se simulará en Matlab para obtener los resultados referentes al estudio de la arquitectura de las redes frente al MSE. De esta manera, se podrá comprobar cuál es la arquitectura óptima del modelo.

El algoritmo de Levenberg-Marquardt utiliza los datos de entrenamiento y validación para reajustar los pesos de la función subyacente. Es por ello, por lo que se ha reservado un porcentaje de datos, que no ha visto el modelo, para ser utilizados como test, y obtener un error de validación del modelo.

A continuación, se muestra la gráfica referente al estudio de la arquitectura:

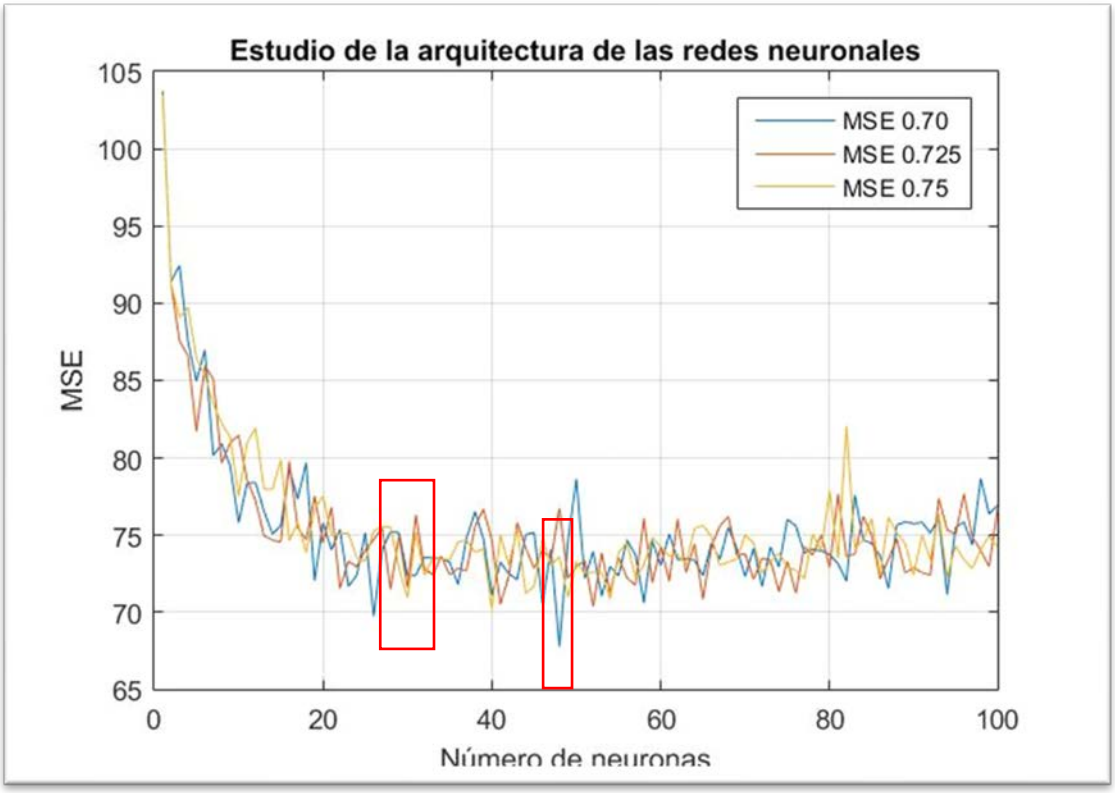


Figura 28. Estudio de la arquitectura

Como se puede observar en la gráfica anterior, se obtiene un mínimo valor de MSE para aproximadamente 45 neuronas, sin embargo, se observa que en ese tramo no han convergido ninguno de los otros dos modelos de redes con porcentajes de entrenamiento similares. Es por ello por lo que este punto (número de neuronas) no se considera adecuado para ser elegido para nuestro modelo, ya que el resultado no es consistente para los tres porcentajes de entrenamiento.

Cabe mencionar, que, si observamos los resultados para 30 neuronas, se puede ver que los 3 modelos convergen en esta zona de error mínimo de manera que el resultado en dicho punto si muestra una coherencia y una consistencia que aportan fiabilidad a esa cantidad de neuronas.

Esto indica que esta configuración se adapta mejor al modelo independientemente del tamaño del conjunto de training, de manera que con dicha complejidad la red generaliza mejor.

Además, al ser un número menor de neuronas que el anterior caso, hará que la velocidad de procesamiento sea mucho mayor.

A continuación, se muestra un zoom de la gráfica anterior para la zona cercana a 30 neuronas:

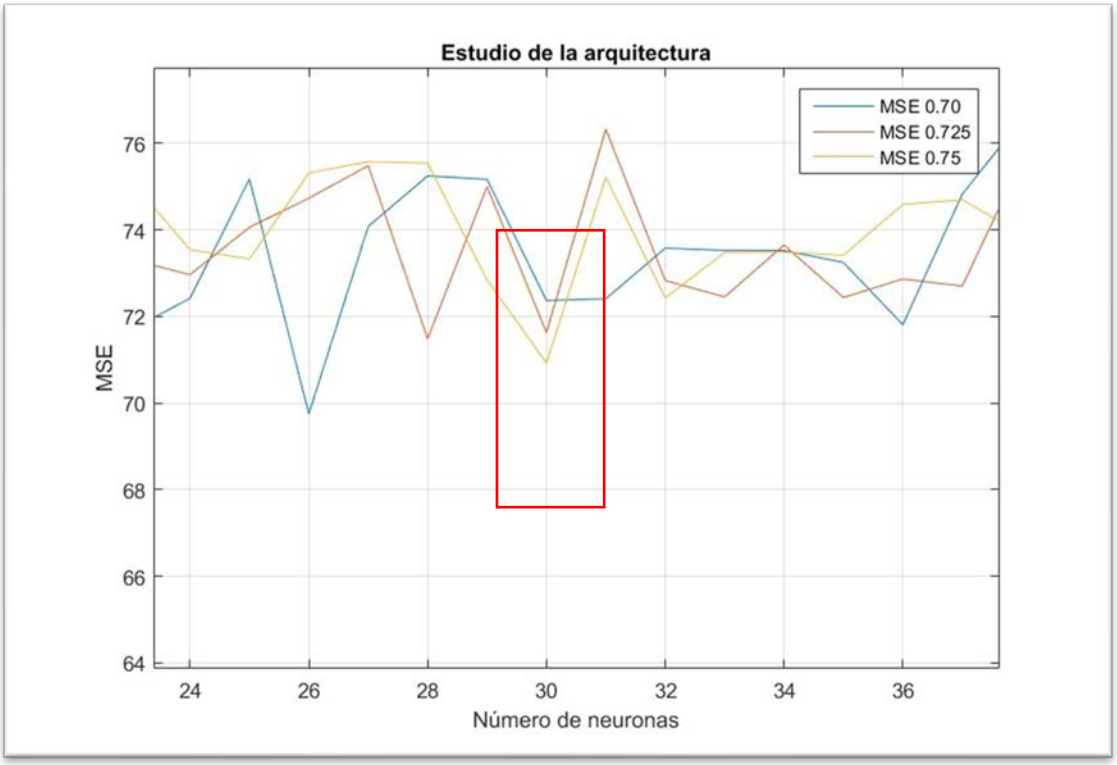


Figura 29. Número de neuronas (zoom)

Como se puede observar en la gráfica anterior, se ve como los 3 modelos convergen para una cantidad de treinta neuronas, obteniendo un valor MSE de aproximadamente 71, es decir, un error medio de 8,71 aproximadamente frente a un valor de hidrógeno cuya magnitud es de 5x10E3 aprox.

Siendo un error pequeño comparado con la magnitud del valor que tiene la variable que se está estimando.

Siendo un muy buen resultado el que se obtiene del modelo.

Por todo ello, se considerará la configuración de la arquitectura en una capa oculta y 30 neuronas como óptimo para él una de las partes del estudio de la complejidad.

3.6.2. Número de iteraciones o epochs

Una vez se conoce la configuración óptima de la arquitectura, es preciso estudiar el número óptimo de iteraciones en las que el algoritmo debe reajustar los pesos de la función subyacente con el fin de no producir un overfitting, ya que, por defecto, en Matlab vienen prefijados en 1.000 iteraciones.

Para ello se establecerá la arquitectura óptima obtenida en el apartado anterior, y manteniendo los parámetros de boostrap y promedios utilizados, se modificará en el algoritmo el parámetro que hace referencia a las iteraciones o epochs.

Se realizará, por tanto, un estudio de las iteraciones para la arquitectura anterior, que irá desde 500 iteraciones hasta 5000, en saltos de 100 en 100.

Finalmente, y del mismo modo que en el apartado anterior, se validará cada vez el modelo con los datos de test obteniendo un valor de MSE, siendo aquel con MSE el valor óptimo de iteraciones.

A continuación, se muestra la gráfica del estudio de iteraciones o epochs:

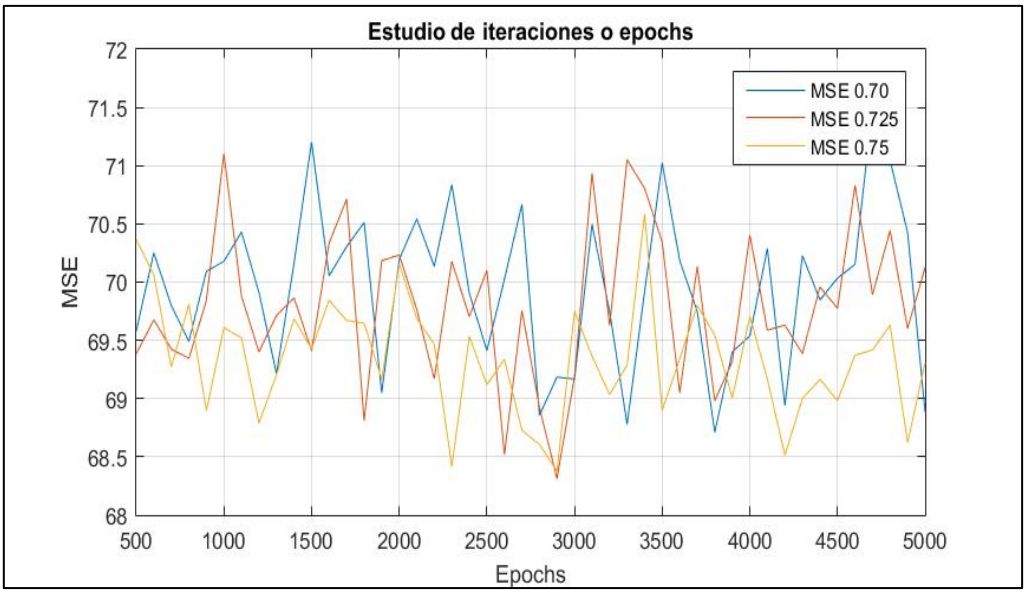


Figura 30. Estudio de las iteraciones o epochs

Como se puede observar en la figura anterior, se obtiene que para un valor de 2900 iteraciones convergen los 3 modelos de entrenamiento obteniendo un valor aproximado de MSE algo inferior a 68,5.

3.7. Resultados y selección del modelo óptimo.

Una vez se han definido todos los parámetros del algoritmo y realizado el estudio de la complejidad del modelo, y gracias a la ayuda de Matlab como la herramienta principal de las simulaciones, se puede confirmar que se ha obtenido un modelo óptimo de predicción de hidrógeno con un error de generalización de XXXX. Siendo este error muy pequeño en comparación con la magnitud de la variable que define la producción de H2.

Por ello, se crea en Matlab, y para los próximos estudios de este proyecto, una estructura que contiene la función del modelo de red neuronal óptimo.

A modo de resumen, se establece que la configuración óptima de la red neuronal para la predicción de H2 es:

- Número de capas ocultas: 1
- Número óptimo de neuronas: 30
- Número de iteraciones: 2900
- Algoritmo de entrenamiento: Levenberg-Marquardt

Finalmente queda definida así el modelo óptimo para la predicción de hidrógeno.

Este modelo óptimo obtenido, será el que se utilizará para los próximos estudios del presente proyecto.

MÁSTER EN SISTEMAS DE ENERGÍA TÉRMICA

CAPÍTULO 4:

ANÁLISIS DE SENSIBILIDAD

4.3. Objetivos

A lo largo de este capítulo se desarrollará un análisis de sensibilidad para las variables de entrada, con el fin de poder ver la influencia sobre el modelo de estas variables.

De este modo, se puede estudiar que variables tienen mayor importancia en el modelo y de qué manera afecta su variación con respecto a la producción de hidrógeno.

Para ello, se estudiarán dos casos en los que se evaluará cada variable, siendo esta modificada, y manteniendo el resto constantes, para ver su implicación en el modelo.

Se mostrará en el anexo del proyecto el algoritmo implementado para la realización del presente estudio para cualquier comprobación.

4.4. Introducción a los análisis de sensibilidad

El análisis de sensibilidad es una herramienta utilizada en el modelado para analizar cómo los diferentes valores de un conjunto de variables independientes afectan a una variable específica bajo ciertas condiciones. En general, del análisis de sensibilidad hay muchos ejemplos, ya que se utiliza en una amplia gama de campos, desde biología y geografía hasta economía e ingeniería.

El análisis de sensibilidad es especialmente útil en el estudio de cualquier proceso donde la salida se considera como una función opaca de varias entradas. Una función o proceso opaco es aquel que, por alguna razón, no puede ser estudiado y analizado.

En el presente proyecto, con el análisis de sensibilidad lo que se pretende es estudiar cómo afectan al comportamiento del modelo los cambios en la entrada del mismo.

Si se quiere estudiar la influencia de una variable dada sobre un modelo, se debe de realizar dicho análisis sometiendo al modelo a la tarea de la estimación del parámetro de salida a partir de distintos valores de las variables de entrada, con la intención de ver las variaciones que se suceden en la salida del mismo.

De esta manera se estudia cómo afecta una variable al modelo, o lo que es lo mismo, cuán importante es la misma.

4.5. Metodología aplicada

Se procederá a la estimación del hidrógeno para una población de distintas configuraciones, que sólo variarán en el valor de la variable la cual se quiere estudiar, de manera que se pueda observar claramente cómo afecta dicha variable al resultado del proceso de estimación del hidrógeno.

Para ello se han generado distintas poblaciones de datos, en cada una de ellas la variable que es estudiada es la única que varía con respecto a las otras (se mantienen constantes).

En ambos casos, la matriz de datos se ha obtenido a partir del mejor registro de la base de datos para la producción de hidrógeno.

Se ha considerado como mejor registro aquel que dio lugar al valor máximo de la producción de hidrógeno. Esto es debido a que el objetivo es maximizar la producción de hidrógeno, y, por tanto, se quiere conseguir ver la implicación en el modelo de la variación de cada una de estas variables en el registro óptimo para la producción de hidrógeno.

Como se acaba de comentar, el registro óptimo se ha obtenido a partir de la producción de hidrógeno, y por ello se presenta a continuación un gráfico de tipo histograma, en el que se mostrará la frecuencia y distribución de todos los valores de esta variable.

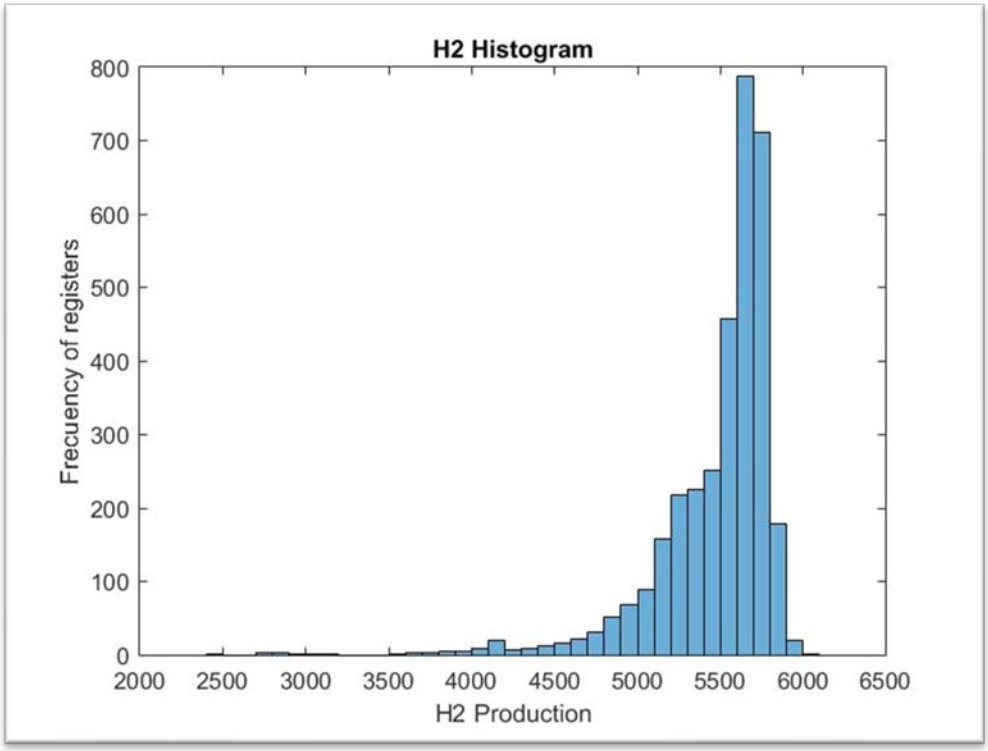


Figura 31. Histograma de la producción de H2

Como se puede apreciar en la figura anterior, los registros con mayores frecuencias se encuentran en el intervalo 5600-5700. Se tomará, por tanto, el valor máximo dentro de este intervalo.

Una vez se tiene el valor máximo de hidrógeno, se obtiene, conociendo su índice del registro, aquel registro de las variables de entrada que coincide con dicho índice. De esta forma se obtiene finalmente, el registro óptimo que se considerará para el estudio del presente proyecto.

Para estudiar cómo afectan estas variables en el modelo, se evaluarán en el modelo óptimo de red neuronal obtenido en el capítulo anterior.

Los casos que se estudiarán en el presente proyecto son:

4.5.1. Análisis de sensibilidad en el rango completo

En este primer caso, el análisis de sensibilidad se ha realizado variando cada variable en su rango completo y manteniendo constantes el resto para el mejor registro de la producción de hidrógeno.

Se han obtenido un total de 17 matrices (hay 17 variables) con 1000 registros para su estudio.

De este modo, se analizará cada una de estas variables en su rango completo, y se evaluará como afecta su variación en el modelo. De manera que, obteniendo las distintas matrices, se evaluarán en el modelo óptimo de red neuronal obtenido en el capítulo anterior.

A continuación, se muestran los resultados obtenidos tras la evaluación de las matrices en el modelo óptimo. Donde cada gráfica hará referencia a cada una de las variables de estudio.

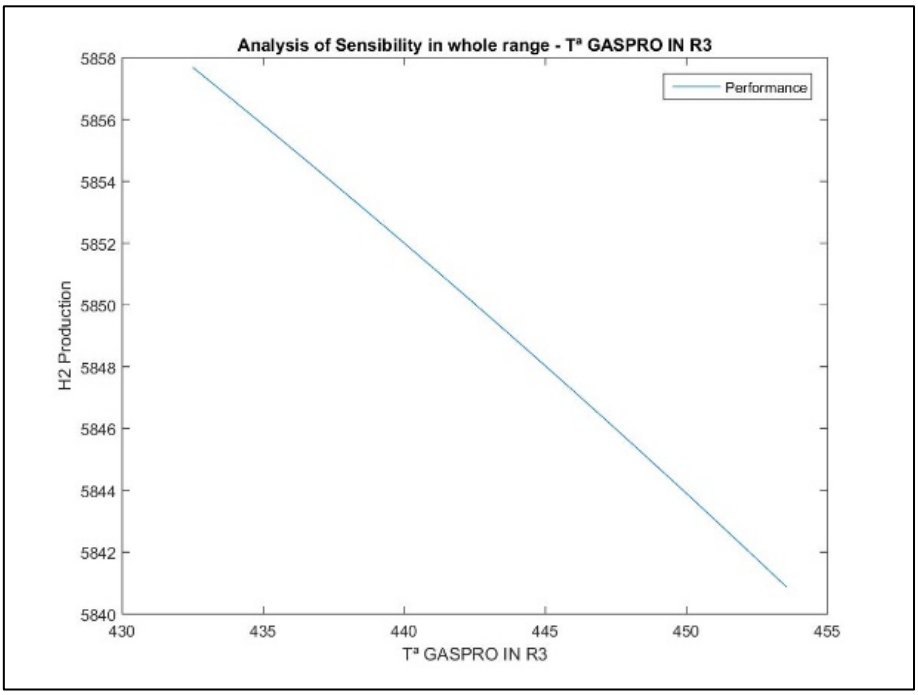


Figura 32. Tª Gaspro en R3

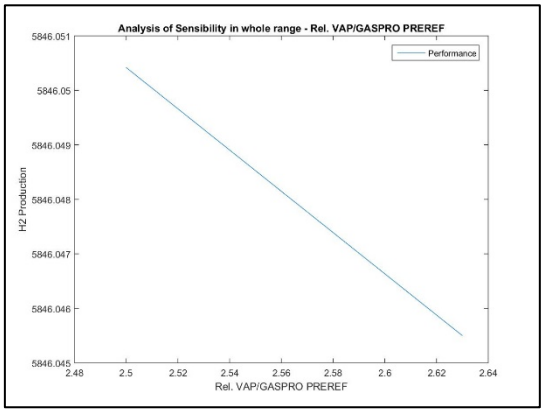


Figura 33. Ref. Vap/Gaspro Preref

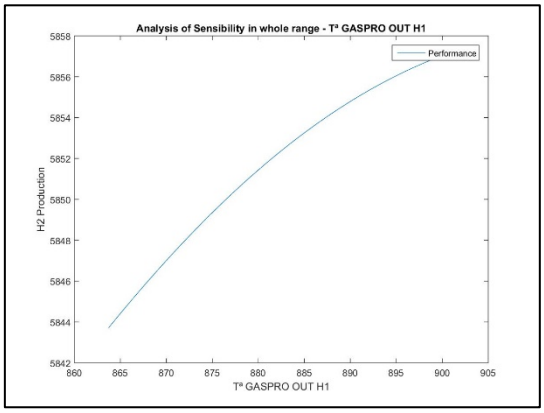


Figura 34. Tª Gaspro out H1

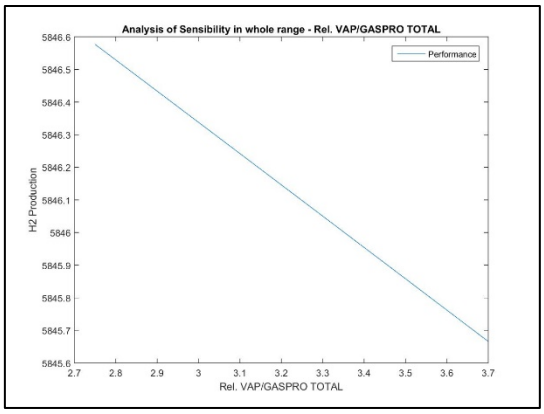


Figura 35. Rel. Vap/Gaspro Total

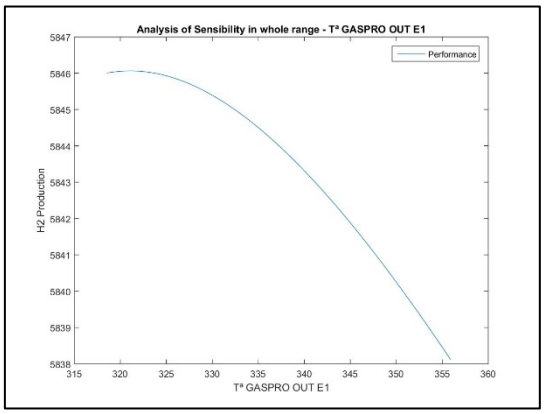


Figura 36. Tª Gaspro out E1

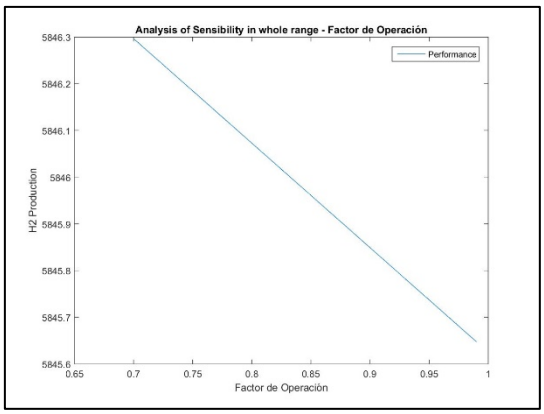


Figura 37. Factor de operación

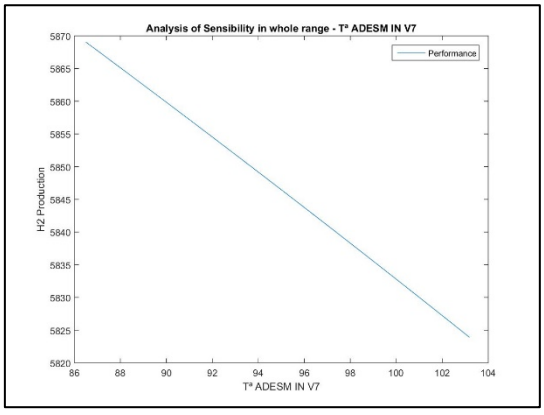


Figura 38. Tª Adesm in V7

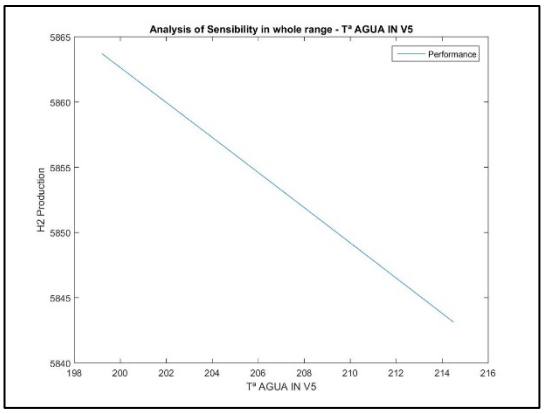


Figura 39. Tª Agua in V5

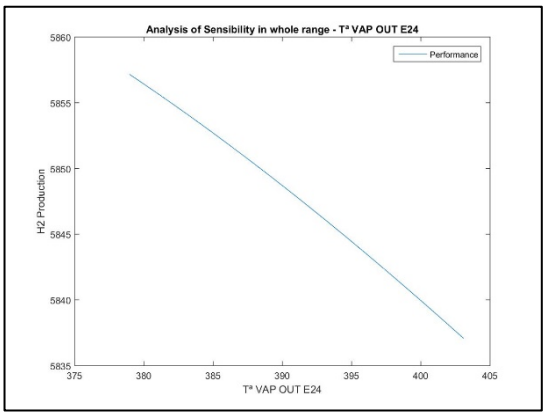


Figura 40. Tª Vap Out E24

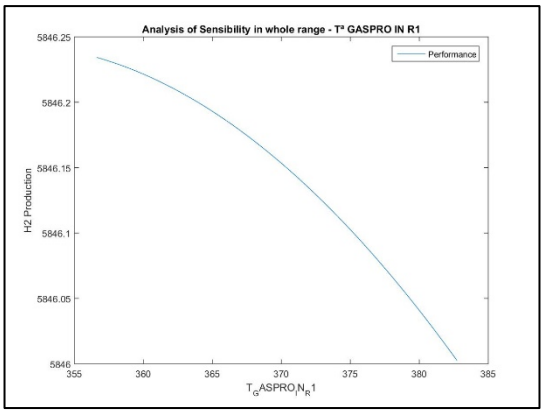


Figura 41. Tª Gaspro in R1

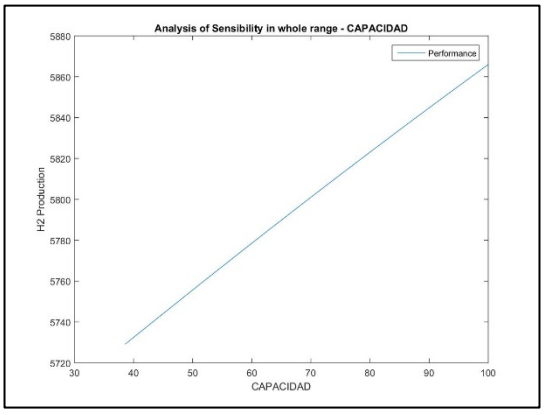


Figura 42. Capacidad

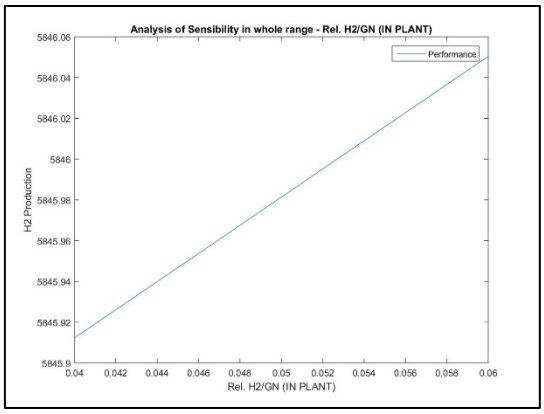


Figura 43. Rel. H2/Gn in plant

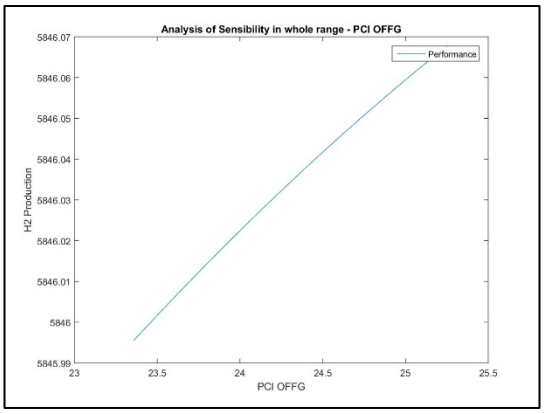


Figura 46. PCI OFFG

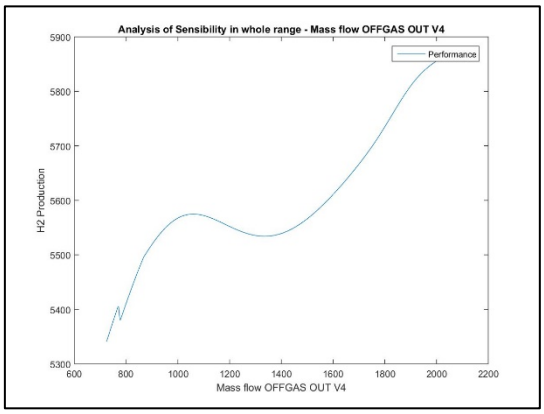


Figura 44. Mass Flow Offgas out in V4

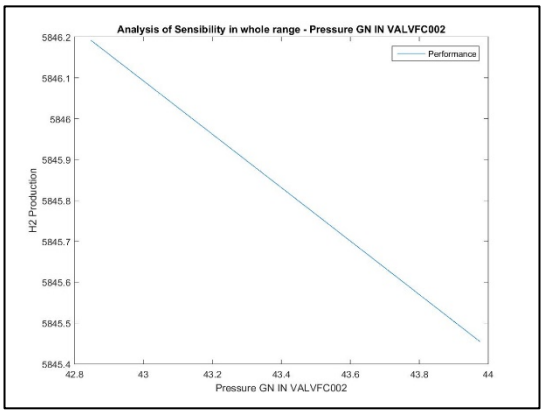


Figura 47. Pressure GN in valvf 002

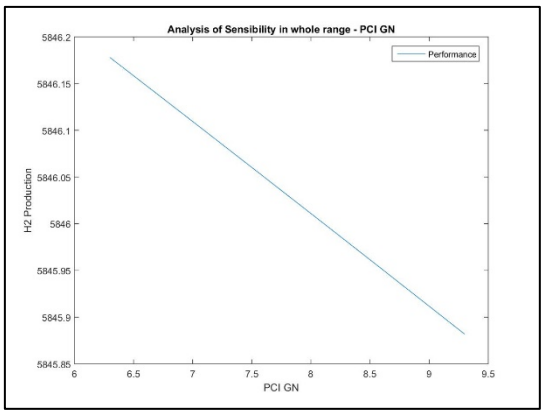


Figura 45. PCI GN

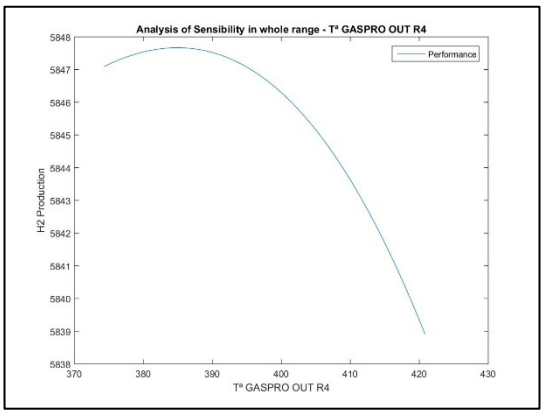


Figura 48. Tª Gaspro out in R4

Como se puede observar en las gráficas anteriores, la mayoría de las variables tienen una influencia destacable en el modelo, ya que, dependiendo del valor de estas, la producción de hidrógeno se ve afectada en mayor o menor medida.

En muchos casos podemos apreciar que ambas variables tienen un coeficiente de correlación negativo, es decir, una crece a medida que la otra disminuye.

Esto significa que, cuando dicha variable aumenta, influye negativamente en la producción de hidrógeno, de manera que si se quiere maximizar la producción de hidrógeno sería efectivo disminuir dicha variable.

4.5.2. Análisis de sensibilidad +-2% de la media de cada variable

En este segundo caso, el análisis de sensibilidad se ha realizado variando cada variable, partiendo del registro óptimo de hidrógeno, en un +-2%; y se ha mantenido el resto constantes en los valores del registro óptimo.

Del mismo modo que en el caso anterior,

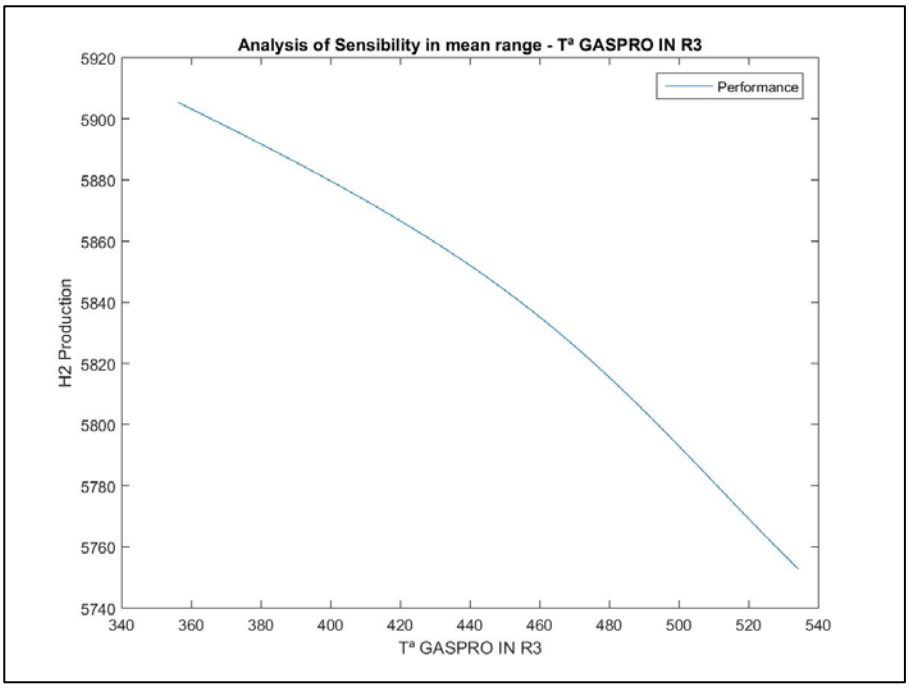


Figura 49. Tª Gaspro in R3

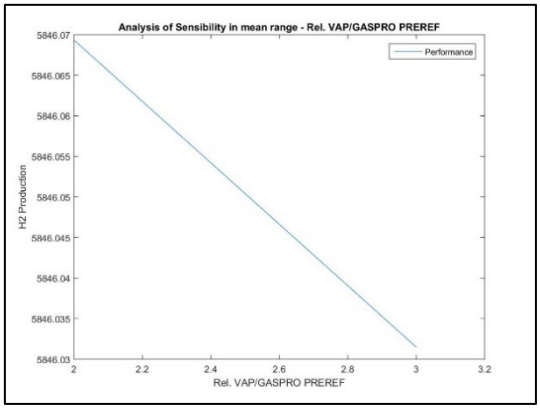


Figura 50. Ref. Vap/Gaspro Preref

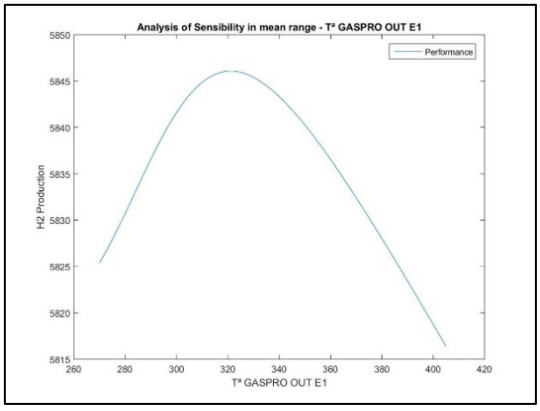


Figura 51. Tª Gaspro out in E1

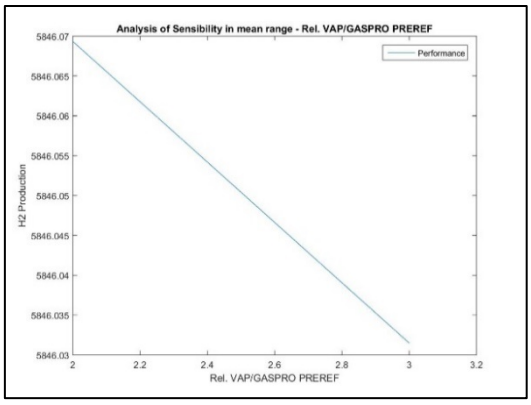


Figura 51. Rel. Vap/Gaspro Total

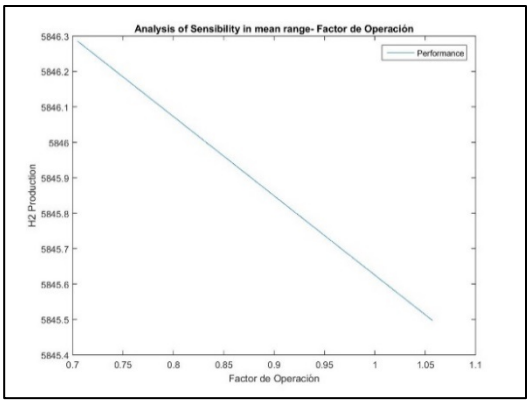


Figura 53. Factor de operación

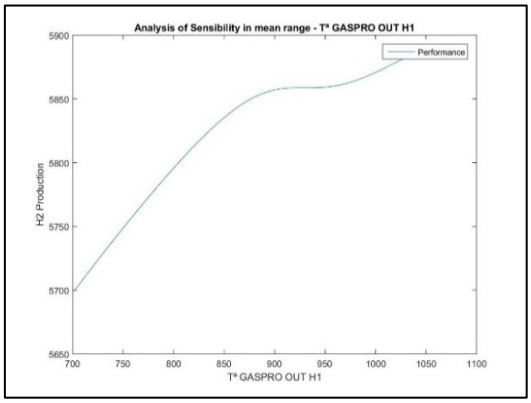


Figura 52. Tª Gaspro out H1

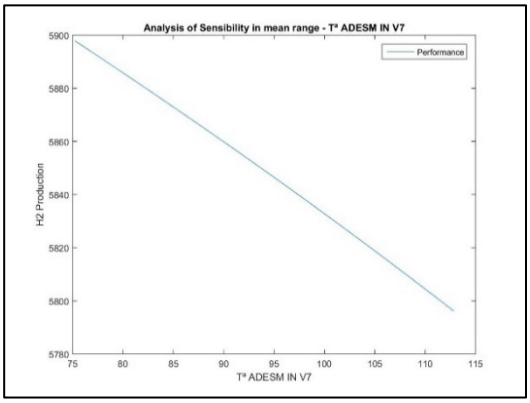


Figura 54. Tª Adesm in V7

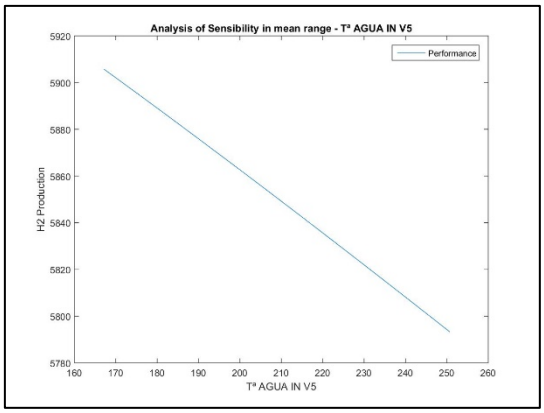


Figura 55. Tª Agua in V5

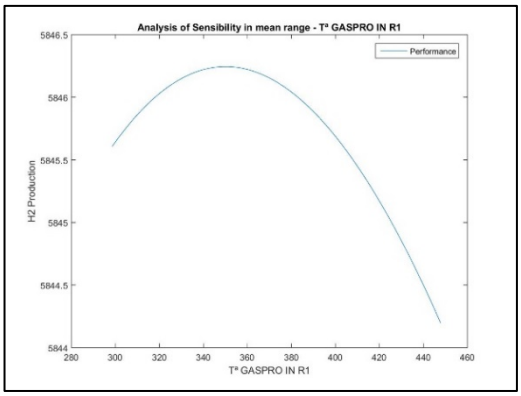


Figura 57. Tª Gaspro in R1

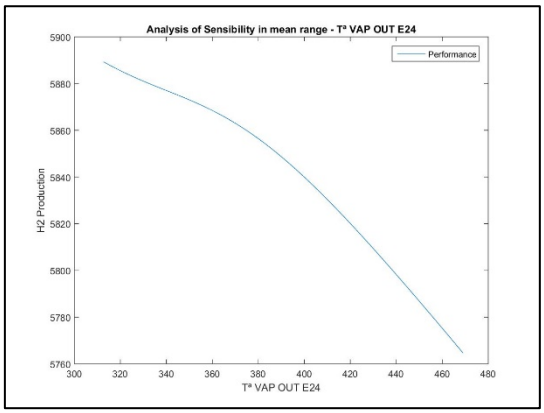


Figura 56. Tª Vap Out E24

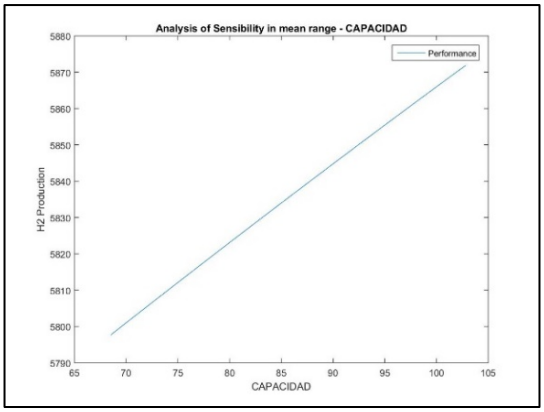


Figura 58. Capacidad

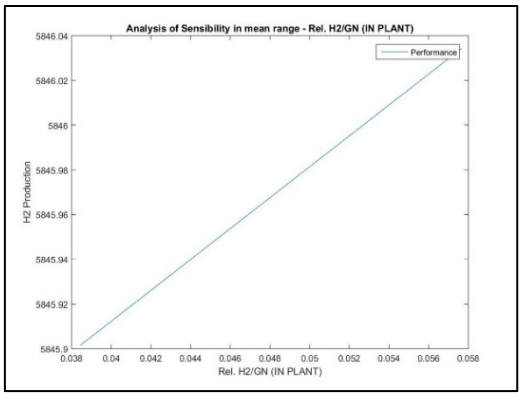


Figura 59.Rel. H2/Gn in plant

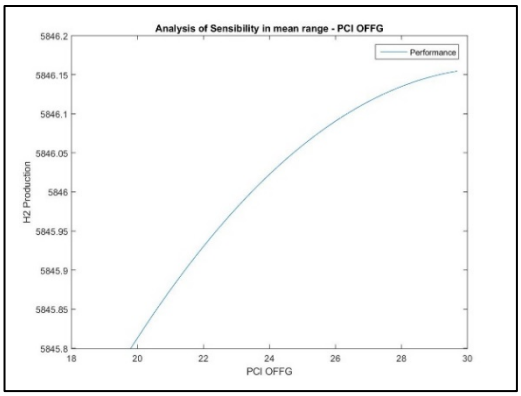


Figura 62. PCI OFFG

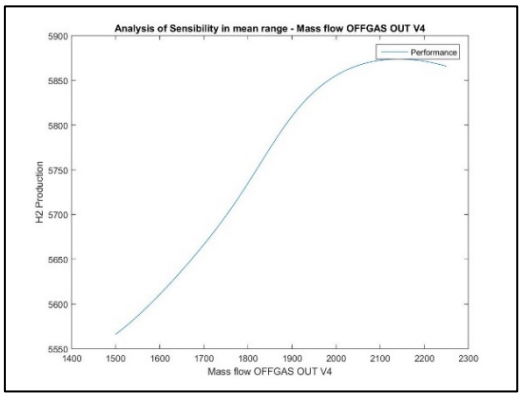


Figura 60. Mass Flow Offgas out in V4

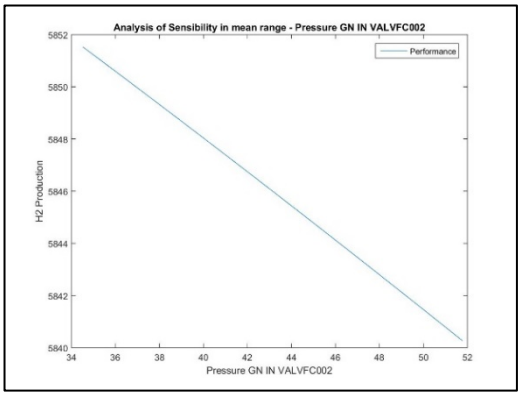


Figura 63.Pressure GN in valvf 002

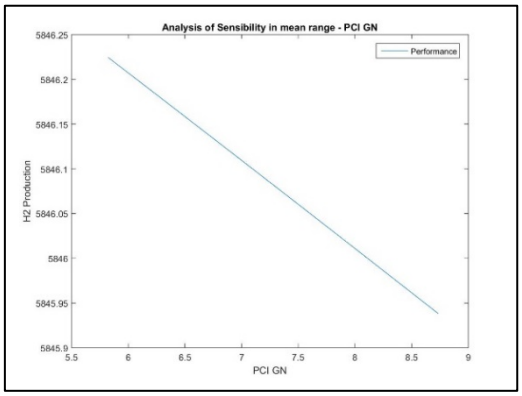


Figura 61. PCI GN

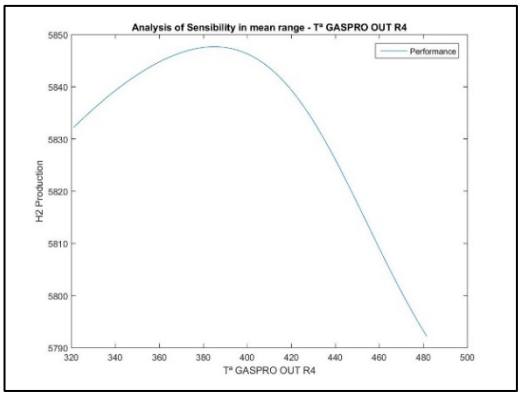


Figura 64.Tª Gaspro out in R4

En las gráficas anteriores, se puede observar del mismo modo que en el caso anterior, que todas las variables tienen una influencia muy destacable en el modelo, y que, la gran mayoría tiene un coeficiente de correlación negativo.

De manera que ya se tiene constancia de que variables influyen positivamente y cuales negativamente en la estimación de hidrógeno de nuestro modelo.

4.6. Resultados del análisis de sensibilidad

Tras realizar el estudio de sensibilidad en ambos casos, se comprueba que la gran mayoría de las variables tienen una gran influencia en el modelo, lo que quiere decir que la selección de las variables facilitada por el tutor es realmente correcta, pues la producción de hidrógeno depende en gran medida de estas variables.

Gracias al análisis de sensibilidad, se puede saber que variables de manera independiente, tienen mayor influencia en el modelo. Aunque lo ideal, sería estudiar el análisis de sensibilidad en una combinación lineal de las variables, no es del alcance de este proyecto, por lo tanto, se le dará importancia a aquellas variables que tras el presente estudio tienen mayor influencia.

MÁSTER EN SISTEMAS DE ENERGÍA TÉRMICA

CAPÍTULO 5:

MAXIMIZACIÓN DEL HIDRÓGENO

5.1. Objetivos

El objetivo de este capítulo es el de maximizar el valor de producción de hidrógeno buscando aquella configuración óptima de las variables que causan su aumento de la producción, siempre dentro de los límites teóricos.

Para ello, se utilizarán los límites de las variables facilitadas por el tutor, de manera que, algunas de estas, estarán optimizadas entre los valores que, por experiencia y comportamiento real, han sido considerados como límites superior e inferior de cada variable.

5.2. Introducción a los métodos de optimización

Existen áreas del conocimiento, especialmente relacionadas con la bio-medicina y la ingeniería, donde surgen problemas relacionados con la mejora de determinados procesos o la obtención de mejores soluciones (generales o particulares).

Todos ellos son problemas que presentan una característica en común, donde tratan de optimizar (maximizar o minimizar) un determinado valor. Por esto, se llaman problemas de optimización. Los problemas de optimización pueden clasificarse en función de diferentes factores como son su complejidad, la existencia o no de restricciones, su carácter estático o dinámico, lineal o no lineal, mono-objetivo o multi-objetivo, etc.

En cuanto a las técnicas de búsquedas, estas se pueden clasificar en función de si aseguran obtener el resultado óptimo (técnicas exactas) o si por el contrario permiten obtener soluciones cercanas al optimo (técnicas aproximadas).

Tomando en cuenta que la optimización combinatoria consiste en encontrar la mejor solución (óptima) de entre un conjunto finito de soluciones alternativas. La calidad u optimalidad de las soluciones vendrá definida por la capacidad de dichas soluciones para minimizar o maximizar una determinada función, denominada función objetivo, compuesta por un conjunto determinado de variables definidas sobre un conjunto discreto.

Esta disciplina tiene numerosas aplicaciones en ámbito como las ciencias, la ingeniería, la industria, la logística, la administración de organizaciones, etc.

Como ejemplo podemos mencionar, entre otros, el diseño de redes de telecomunicación, la asignación de tareas a procesadores, el enrutamiento y carga de vehículos en redes de distribución, la planificación de la producción, la selección de carteras financieras, la planificación de la generación de electricidad, el diseño de proteínas, la distribución de ambulancias en una región para asegurar un cierto nivel de servicio a su población, etc. La gran variedad de aplicaciones requiere la ayuda de optimizadores que han provocado un gran interés en las últimas décadas en el desarrollo de nuevos métodos.

Esto ha volcado el interés hacia las técnicas aproximadas que permiten obtener una solución "casi" óptima en un tiempo razonable; en particular al uso de las metaheurísticas que se pueden definir como estrategias inteligentes generales para diseñar o mejorar procedimientos heurísticos para la resolución de problemas con un alto rendimiento.

Dentro de los algoritmos no exactos podemos encontrar tres tipos: los heurísticos constructivos (también llamado voraces), los métodos de búsqueda local (o métodos de seguimientos del gradiente) y las metaheurísticas. Existen diferentes formas de clasificar y describir las técnicas metaheurísticas dependiendo de las características que se seleccionen se pueden obtener: basadas en la naturaleza o no basadas en la naturaleza, basadas en memoria o sin memoria, con función objetivo estática o dinámica, etc.

En el presente proyecto, nos vamos a centrar en las metaheurísticas que están dadas por el número de soluciones que procesan en un determinado momento durante el proceso de búsqueda, es decir, aquellas "basadas en trayectoria" y las "basadas en población" (o poblacionales).

A continuación, se describen las características básicas de las metaheurísticas:

- Son estrategias que guían el proceso de búsqueda.
- Exploran eficientemente el espacio de búsqueda con el objetivo de encontrar soluciones próximas al óptimo global.
- las técnicas que constituyen las metaheurísticas varían entre métodos de búsqueda local simple a complejos métodos de aprendizaje.
- Pueden incorporar mecanismos para evitar quedar atrapado en óptimos locales del espacio de búsqueda.
- Permiten un nivel de descripción abstracto, no específico del problema.
- Pueden hacer uso de conocimiento del dominio específico de forma que las heurísticas son controladas por una estrategia de nivel superior.
- Las metaheurísticas avanzadas utilizan la experiencia de búsqueda (haciendo uso de alguna forma de memoria) para guiar la búsqueda.

5.3. Tipos de métodos de optimización

Existen gran variedad de métodos o algoritmos de optimización, entre los que podemos hacer una distinción, aquellos basados en trayectoria y aquellos que se basan en poblaciones.

Un aspecto diferenciador entre metaheurísticas, radica en el número de soluciones que se utilizan en el proceso de optimización.

Metaheurísticas como Simulated Annealing (enfriamiento simulado) o Búsqueda Tabú, utilizan una sola solución durante el proceso de búsqueda, por lo que se suelen denominar métodos de trayectoria ya que la solución describe una trayectoria desde la solución de partida hasta encontrar la solución final.

Por otro lado, técnicas como los algoritmos genéticos, hacen uso de un conjunto de soluciones (población) que son optimizadas de forma simultánea durante la búsqueda.

Normalmente los métodos poblacionales suelen conseguir soluciones de más calidad debido a la ventaja que supone explorar simultáneamente diferentes áreas del espacio de búsqueda, sin embargo, requiere de un estudio más exhaustivo de la población de datos inicial para evitar el uso de información no adecuada para el estudio.

A continuación, se muestra en las siguientes imágenes el ejemplo de las metahurísticas basadas en trayectorias y en poblaciones:

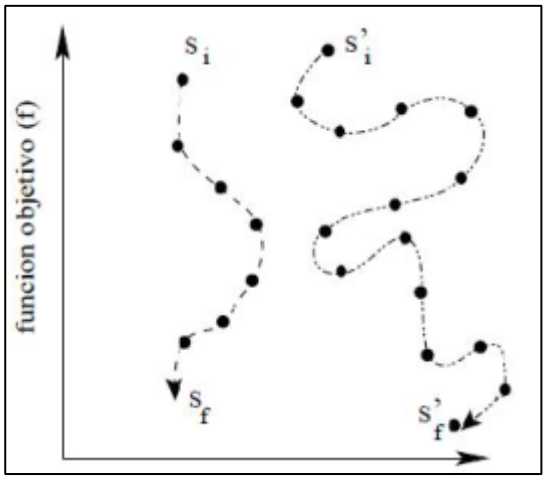


Figura 65. Metahurísticas basadas en trayectorias

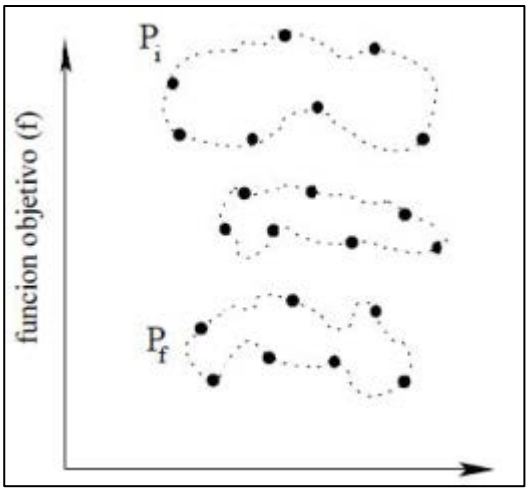


Figura 66. Metahurísticas basadas en trayectorias

En el presente proyecto, se utilizará el método de Simulated Annealing debido a la rapidez que ofrecen sus algoritmos para obtener unos resultados muy realistas, y además, basado en la experiencia trabajada en procesos similares del tutor en estudios de este tipo.

5.3.1. Simulated Annealing (recocido simulado)

Esta metaheurística aparece a principios de los años 80 (Kirkpatrick et al. 1983). Se trata de un modelo de resolución para la optimización de problemas de tipo combinatorio con mínimos locales. Consiste en generar aleatoriamente una solución cercana a la solución actual (o en el entorno de la solución) y la acepta como buena si consigue reducir una determinada función de costo, o con una determinada probabilidad de aceptación.

Esta probabilidad de aceptación se irá reduciendo con el número de iteraciones y está relacionada también con el grado de empeoramiento del costo, es decir, en el algoritmo Simulated Annealing se pueden aceptar soluciones que empeoran la solución actual, solo que esta aceptación dependerá de una determinada probabilidad que depende de un parámetro, denominado temperatura, del estado del sistema.

El nombre de SA viene de la idea en que está basado en un algoritmo diseñado en los años 50 para simular el enfriamiento de material (un proceso denominado "recocido").

El proceso de SA es el siguiente: parte de una solución inicial que de modo paulatino es transformada en otras que a su vez son mejoradas al introducirles pequeñas perturbaciones o cambios (tales como cambiar el valor de una variable o intercambiar los valores que tienen dos variables).

Si este cambio da lugar a una solución "mejor" que la actual, se sustituye ésta por la nueva, continuando el proceso hasta que no es posible ninguna nueva mejora. Esto significa que la búsqueda finaliza en un óptimo local, que no tiene por qué ser forzosamente el global.

Un modo de evitar este problema es permitir que algunos movimientos sean hacia soluciones peores. Pero por si la búsqueda está realmente yendo hacia una buena solución, estos movimientos "de escape" deben realizarse de un modo controlado.

En SA esto se realiza controlando la frecuencia de los movimientos de escape mediante una función de probabilidad que hará disminuir la probabilidad de esos movimientos hacia soluciones peores conforme avanza la búsqueda (y por tanto estamos previsiblemente más cerca del óptimo global).

La fundamentación de este control se basa en el trabajo de Nicolás Metropolis en el campo de la termodinámica estadística.

Básicamente, Metropolis modeló el proceso de recocido mencionado anteriormente simulando los cambios energéticos en un sistema de partículas conforme decrece la temperatura, hasta que converge a un estado estable (congelado).

La ley de la termodinámica dice que a una temperatura t la probabilidad de un incremento energético de magnitud δE se puede aproximar por P[δE] = exp (-δE /kt) siendo k una constante física denominada de Boltzmann.

En el algoritmo de Metropolis se genera una perturbación aleatoria en el sistema y se calculan los cambios de energía resultantes: si hay una caída energética, el cambio se acepta automáticamente; por contra, si se produce un incremento energético, el cambio será aceptado con una probabilidad dada por la formula anterior.

El proceso se repite durante un número predefinido de iteraciones en series decrecientes de temperaturas, hasta que el sistema esté "frío".

Los algoritmos de enfriamiento lento simulado tienen algunas ventajas con respecto a otras técnicas de optimización global.

Entre las ventajas de estos algoritmos se pueden citar:

- Lo relativamente sencillo que resulta implementar este tipo de problemas.

- Su aplicabilidad a la mayoría de los problemas de optimización con una estructura combinatoria.

- Su capacidad para ofrecer soluciones razonablemente buenas a la mayoría de los problemas, aunque hay una cierta dependencia de la planificación del enfriamiento y los movimientos que se realicen.

Por otra parte, también se pueden citar algunos aspectos que pueden limitar su utilización:

- Se necesita elegir con mucho cuidado los movimientos que se realizan, así como los parámetros que se van a utilizar para tratarlo, como por ejemplo la tasa de enfriamiento.

- Una ejecución del problema puede requerir mucho tiempo de cálculo.

- Puede que sea necesario realizar muchas ejecuciones para encontrar una solución satisfactoria.

- Dependiendo de los parámetros elegidos, las soluciones que se van encontrando pueden ser poco estables, "saltando" mucho de unas a otras sin encontrar una solución buena con la rapidez suficiente lo que obliga a retocar los parámetros con las distintas ejecuciones.

5.3.2. Parámetros para la configuración del SA

- Temperatura inicial

La temperatura inicial es la que se adopta para comenzar el proceso de enfriamiento y se irá bajando en tanto avance la evolución. Está debe ser lo suficientemente alta para permitir el libre cambio de soluciones vecinas y, sobre todo, que la solución final sea independiente de la solución inicial.

Una regla práctica que se le suele poner a la temperatura inicial es que debe tener un valor tal que la proporción inicial de vecinos aceptados (tanto de mejora como de no mejora) tenga un alto valor, es decir, inicialmente todos los vecinos se aceptan con una probabilidad cercana a 1.

Lógicamente hay que elegir esta temperatura con cuidado porque si se elige una temperatura inicial demasiado alta se perderá tiempo realizando muchos movimientos que se aceptan siempre, lo que no resulta de mucho interés ya que no se está persiguiendo ningún valor concreto.

Por otro lado, si se elige una temperatura inicial demasiado baja el proceso se mueve poco desde la solución inicial por lo que se puede quedar atrapado desde un inicio en una región desde la que sólo se pueda llegar a un óptimo local, por lo que se perderá la posibilidad de explorar otras regiones donde se podría encontrar alguno de los óptimos globales del sistema.

- Proceso de enfriamiento

El proceso de enfriamiento es el mecanismo por el cual la temperatura va tendiendo a 0. Es decir, la probabilidad de aceptación de soluciones peores tiende a 0. El proceso de enfriamiento, por tanto, determina cómo se modifica la temperatura durante la ejecución del algoritmo.

Hay dos modelos básicos que se pueden utilizar para el proceso de enfriamiento. En el primero podríamos ir enfriando muy lentamente con cada iteración de la cadena de manera que en un número de pasos extremadamente alto la temperatura tienda a 0, o utilizar un proceso de enfriamiento más rápido, pero antes de aplicar un cambio en la temperatura hay que dejar que el sistema llegue a un estado en el que se estabilice la solución.

Hay que intentar durante el proceso de enfriamiento que haya una alta aceptación al principio (de forma que se pueda realizar un proceso de exploración inicial en el espacio de búsqueda) e ir reduciendo está aceptación hacia el final.

- Número de repeticiones

El número de repeticiones del bucle interno nos va a dar el número de vecinos que se visitan antes de reducir la temperatura del sistema. El proceso debe ser dinámico, de forma que se aumente el número de repeticiones según se reduce la temperatura. De esta forma se conseguirá explotar la vecindad hacia el final del proceso.

Un mecanismo habitual para conseguirlo es repetir el bucle hasta que se cumpla una de las siguientes condiciones:

- Se aceptan un cierto número de evaluaciones. De esta forma el bucle termina en caso de que se hayan analizado un determinado número de evaluaciones del algoritmo.

- Se hayan obtenido la solución óptima. De esta forma el bucle termina si se da el caso que se ha conseguido la mejor solución.

Para que se pueda explorar con cierta completitud todos los vecinos.

- Condición de parada

Esta condición nos indica cuando damos por terminado el algoritmo de cálculo. Hay que tener en cuenta que, al reducirse la temperatura, la probabilidad de aceptar soluciones peores tiende a 0. Así mismo, la probabilidad de encontrar soluciones mejores también tiende a 0.

En este sentido las condiciones para terminar el algoritmo deberían tener en cuenta que ya se han realizado un determinado número de iteraciones sin mejorar la solución. La idea es evitar seguir insistiendo reiteradamente en la vecindad de una solución para la que ya se lleva tiempo intentando explorar.

- Espacio de soluciones

La definición del modelo del problema determinará la estructura del espacio de soluciones y, por tanto, la forma en que se va a poder recorrer, los algoritmos que se pueden utilizar para crear soluciones, etc.

Lógicamente para cada problema hay que estudiar de forma diferenciada cómo generar este espacio de soluciones, en relación con la facilidad para generar una vecindad y el cálculo de la función de costo asociada a una determinada solución.

- Solución inicial

La solución inicial, en general es cualquier solución válida del problema. Al ser un problema de optimización combinatoria, el modelo permite expresar una solución como una combinación de partes de este. De esta forma, la solución inicial más sencilla resulta de elegir una combinación aleatoria de elementos del problema que conformen una solución con la estructura del modelo elegido.

- Estructura de vecindad

Dada una solución, hay que poder conseguir una nueva solución mediante un pequeño cambio en la solución original. Además, la forma de generar una vecindad nos debe poder asegurar que a partir de una solución cualquiera se debe poder llegar a cualquier otra ya sea directa o indirectamente.

Esta característica es vital para dar validez al proceso de convergencia del algoritmo. Asimismo, la solución vecina debería ser suficientemente pequeña como para poder realizar una búsqueda en pocas iteraciones, como suficientemente grande para generar mejoras sustanciales en pocos movimientos.

Desde el punto de vista computacional la generación de un vecino debería ser rápida, de forma que se puedan calcular un mayor número de vecinos por unidad de tiempo. Así mismo, sería muy apropiado que el vecino generado siempre sea una solución factible, lo que puede resultar complicado.

5.4. Metodología aplicada

Para el presente estudio se estudiarán aquellas variables que han sido consideradas por el tutor como optimizables, y por tanto serán objeto del estudio. Estas variables, han sido elegidas como consideración en base a la experiencia y al extenso conocimiento sobre el proceso de obtención de hidrógeno al que se hace referencia en el proyecto.

Para ello, se utilizarán los límites de las variables facilitadas por el tutor, de manera que, algunas de estas, estarán optimizadas entre los valores que, por experiencia y comportamiento real, han sido considerados como límites superior e inferior de cada variable.

A continuación, se muestra una tabla con la lista de variables que serán objeto de estudio para optimizar un registro de las variables y maximizar así la producción de hidrógeno.

nº variable lista	Variable	Unidades	Min	Max
1	T_GASPRO_IN_R3	ºC	420	470
2	R_VAP/GASPRO_PREREF	%	2,5	4
3	T_GASPRO_OUT_H1	ºC	870	875
4	R_VAP/GASPRO_TOTAL	%	2,75	4
5	T_GASPRO_OUT_E1	ºC	320	350
6	FACTOR_OPERACIÓN		0,7	1
7	T_ADESM_IN_V7	ºC	90	100
8	T_AGUA_IN_V5	ºC	190	210
9	T_VAP_OUT_E24	ºC	400	430
10	T_GASPRO_IN_R1	ºC	360	380

Tabla 8. Lista de variables para la optimización.

Para el proceso de optimización se ha usado el algoritmo, ya mencionado anteriormente, basado en trayectorias y conocido como Simulated Annealing.

Todo el estudio se ha implementado con algoritmos bajo la interface de Matlab para su simulación.

Para la función que define el grado de optimización y expresa como de buena es la configuración dada, se utilizará el modelo de red neuronal óptimo obtenido en el capítulo 3 del presente proyecto.

Es por ello por lo que se irán evaluando los nuevos registros obtenidos por el algoritmo de Simulated Annealing en el modelo de red neuronal, para buscar mejores ejemplares y con mayor rapidez, y lograr así maximizar la producción de hidrógeno.

Partiendo de un ejemplar (registro) previamente seleccionado, se pretende obtener un ejemplar "mejor" el cual optimice la producción de hidrógeno.

El registro inicial, con sus 17 características, y que se utiliza para el proceso de maximización, será aquel registro de mayor frecuencia que tenga un valor máximo en la producción de hidrógeno.

De esta manera, tendremos 17 variables con los mejores registros obtenidos.

Sin embargo, como se ha comentado anteriormente, solo se optimizarán las variables de la 1 a la 10; y el resto de variables se mantendrán constantes en este mejor registro y que se pretende maximizar. Las variables que serán optimizadas, se les modificará de forma aleatoria un pequeño porcentaje de manera que se obtendrán ejemplares muy parecidos al mejor.

Esto se conseguirá aplicando pequeños cambios de forma aleatoria a las 10 primeras variables (que serán las optimizadas), y evaluando si estos cambios han producido alguna mejora.

Los nuevos ejemplares se consiguen a partir de este registro óptimo, de la siguiente manera:

- Se modifica cada ejemplar de las variables de la 1 a la 10, aplicando la regla de estadística llamada 68-65-99,7[1], para partir de elementos muy parecidos al mejor.

- Estas modificaciones serán aleatorias en cada iteración, obteniendo así un ejemplar distinto para evaluar.

- Se evaluará en la mejor red, quedándonos con el mejor resultado obtenido en la producción de hidrógeno evaluando cada iteración en la mejor red neuronal obtenida en el capítulo 2.

- Se evaluará hasta disminuir la temperatura al mínimo.

- Se realizará un total de 20 hilos o trayectorias para intentar evitar máximos locales y buscar siempre un resultado mejor.

Si dichos cambios ocasionaron un ejemplar de mayor calidad que el previo, entonces seguiremos el proceso con este nuevo ejemplar, si no, este nuevo ejemplar es descartado y proseguimos con el anterior.

Para evitar que el algoritmo sea demasiado avaricioso y que pueda caer en óptimos locales con facilidad hemos desarrollado un sistema estocástico que permite al algoritmo elegir ejemplares peores con cierta probabilidad. De esta manera podríamos escapar de un óptimo local en el caso de que cayéramos en él.

Para evadir aún más los óptimos locales, se ha replicado este proceso varias veces (20 veces), iniciándolo cada vez con un ejemplar distinto.

Esto aumenta considerablemente las posibilidades de encontrar o acercarnos al óptimo global.

Para conseguir esto en un intervalo de tiempo razonable, se ha paralelizado el código de manera que varios procesos de recocido simulado son ejecutados paralelamente, consiguiendo una ganancia de tiempo crítica.

Finalmente, se compara cada uno de los resultados independientes para obtener el mejor de ellos.

[1] En estadística, la regla **68-95-99.7**, también conocida como regla empírica, es una abreviatura utilizada para recordar el porcentaje de valores que se encuentran dentro de una banda alrededor de la media en una distribución normal con un ancho de dos, cuatro y seis veces la desviación típica.

Para configurar el algoritmo tal y como se ha comentado anteriormente, se le ha introducido una temperatura de inicio del sistema de T= 10.000; esta irá disminuyendo con cada iteración.

5.1. Resultados de la optimización

Tras simular en Matlab el algoritmo de Simulated Annealing siguiendo el proceso descrito, se obtiene una gráfica en la que se mostrará como según va variando el registro de entrada, mientras disminuye la temperatura, la producción de hidrógeno va aumentando de forma considerable.

En la gráfica que se muestra a continuación se puede el resultado tras el proceso de optimización de hidrógeno, de manera que:

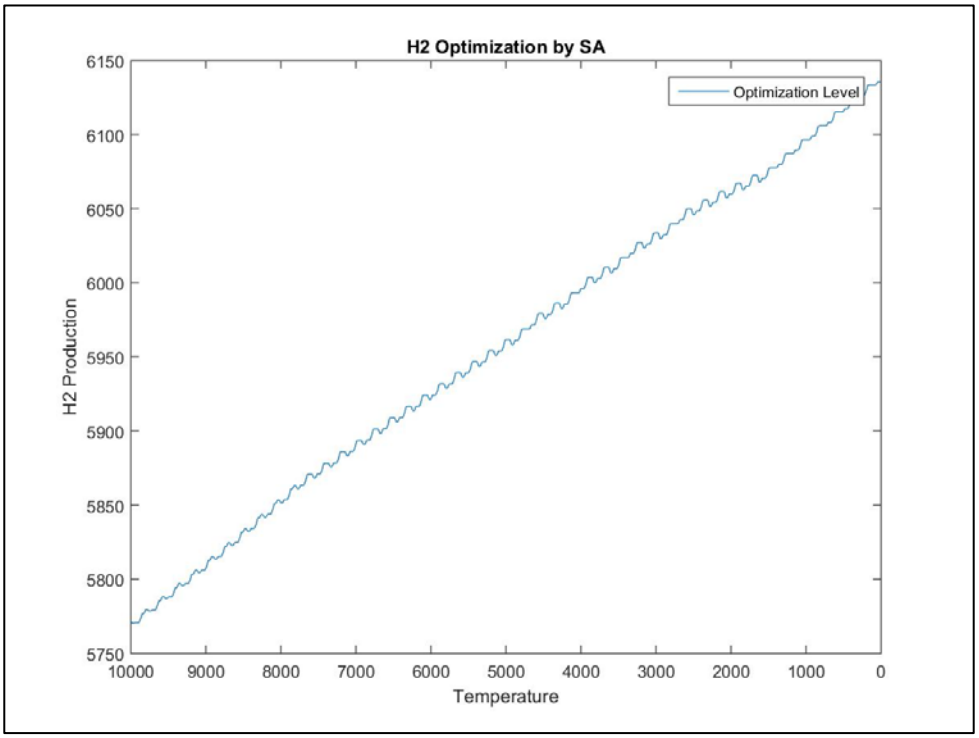


Figura 67. Maximización de hidrógeno mediante Simulated Annealing

En el gráfico anterior se puede observar la mejora a lo largo del proceso de optimización, conforme disminuye la temperatura global del sistema se optimiza el valor de producción de H2.

Los pequeños “baches” que se observan se deben a ese sistema estocástico comentado anteriormente para evadir óptimos locales.

Cuando la temperatura del sistema se acerca a 0, estos “baches” van desapareciendo ya que el algoritmo sólo aceptará especímenes que mejoren al actual.

Finalmente se obtiene un valor para un registro de entrada que maximiza la producción de hidrógeno hasta un total aproximado de 6.140 kg/h, habiendo aumentado desde el valor máximo original de 6.0038 kg/h; es decir, ha aumentado un 2,28%. Demostrando de este modo, la validez del algoritmo utilizado y su importancia a la hora de maximizar funciones.

Cabe destacar que, si se realizasen más pruebas, modificando la aleatoriedad de los saltos del algoritmo, las iteraciones y la temperatura inicial, seguramente se podrían obtener mejoras en cuanto a la producción de hidrógeno. Sin embargo, por restricciones de tiempo, no será objeto de este proyecto un estudio más exhaustivo de la combinatoria de los parámetros del algoritmo.

Queda, por tanto, finalizado el estudio y demostrando una mejora en la producción de hidrógeno tras implementar un algoritmo de Simulated Annealing con respecto a la producción original.

BIBLIOGRAFÍA Y REFERENCIAS

[1] L. G. Jodra, "El hidrógeno, combustible del futuro," *Real Acad. Ciencias Exactas, Físicas y Nat.*, vol. 99, pp. 49–67, 2005.

[2] "Soluciones al Cambio Climático: una Visión Panorámica," 2007. [Online]. Available: http://www.abengoa.es/htmlsites/boletines/es/diciembre2007ext/hidrogeno.htm. [Accessed: 23-Nov-2019].

[3] M. Angel and P. Castillo, "PRODUCCIÓN DE H2 CON CAPTURA DE CO2 POR REFORMADO DE CH4 INTEGRADO CON UN SISTEMA CHEMICAL-LOOPING COMBUSTION TESIS DOCTORAL," 2013.

[4] N. Morlanés Sánchez and F. Vte Melo Faus, "Obtención de hidrógeno mediante reformado catalítico de nafta con vapor de agua."

[5] L. G. Moreno, "PRODUCCIÓN CATALÍTICA DE HIDRÓGENO MEDIANTE REFORMADO CON VAPOR DE COMPUESTOS OXIGENADOS MEMORIA Que para optar al grado de Doctora presenta LOURDES GARCÍA MORENO," Madrid, 2017.

[6] D. J. Matich, "Redes Neuronales: Conceptos Básicos y Aplicaciones.," *Historia Santiago.*, p. 55, 2001.

[7] J. Viviana and V. Garzón, "Aplicación de las redes neuronales artificiales y teoría de juegos al cambio climático con control óptimo," vol. 29, no. 2, pp. 371–381, 2016.

[8] S. Haykin, "Neural Networks. A comprehesive Foundation," 1994.

[9] E. Varela Arregocés Edwin Campbells Sánchez and U. Simón Bolívar Barranquilla -Atlántico, "Redes Neuronales Artificiales: Una Revisión del Estado del Arte, Aplicaciones Y Tendencias Futuras Artificial Neural Networks: A Brief Review," *Investig. y Desarro. en TIC*, vol. 2, pp. 18–27, 2011.

[10] R. Rojas ; R. Rojas and ; R Rojas, "A Systematic Introduction: Neural Networks," Springer-Verlag, 1996.

[11] R. Florez Lopez and J. M. Fernández Fernández, *Las redes neuronales artificiales, fundamentos teóricos y aplicaciones prácticas*. netbiblo, 2008.

[12] A. Fiszlew, "Generación automática de redes neuronales con ajuste de parámetros basado en algoritmos genéticos," Universidad de Buenos Aires, 2002.

[13] M. Stone, "Cross-Validatory choice and Assessment of Statiscal Predictions," *R. Statiscal Soc.*, 1974.

[14] F. Rosenblatt, "THE PERCEPTRON: A PROBABILISTIC MODEL FOR INFORMATION STORAGE AND ORGANIZATION IN THE BRAIN 1," 1958.

[15] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[16] P. Werbos, "Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Science. Thesis (Ph. D.). Appl. Math. Harvard University," 1974.

[17] J. L. Elman, "Finding Structure in Time," *Cogn. Sci.*, vol. 14, no. 2, pp. 179–211, Mar. 1990.

[18] D. Servan-Schreiber, A. Cleeremans, and J. Mcclelland, "Graded State Machines: The Representation of Temporal Contingencies in Simple Recurrent Networks," *Mach. Learn.*, vol. 7, pp. 161–193, Sep. 1991.

[19] F. J. Pineda, "Generalization of back-propagation to recurrent neural networks," *Phys. Rev. Lett.*, vol. 59, no. 19, pp. 2229–2232, Nov. 1987.

[20] N. Ansari and E. Hou, "Hopfield Neural Networks BT - Computational Intelligence for Optimization," N. Ansari and E. Hou, Eds. Boston, MA: Springer US, 1997, pp. 27–45.