TRATAMIENTO DIGITAL DE SEÑALES
Ingeniería de Telecomunicación (4º, 2º c)

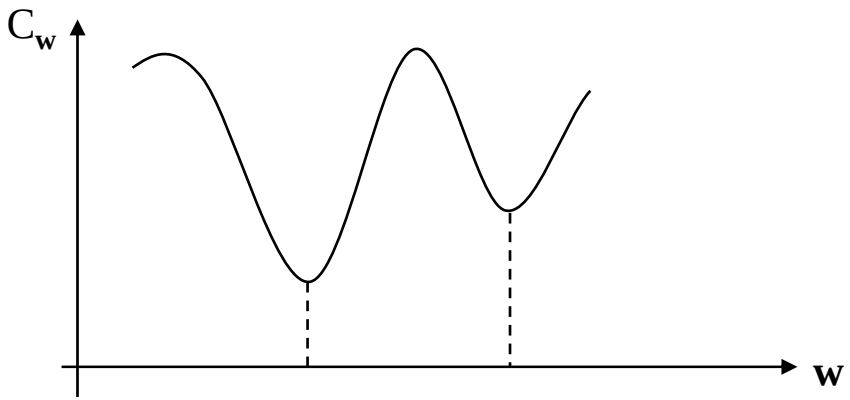
Unidad 9ª: Búsqueda

Aníbal R. Figueiras Vidal
Jesús Cid Sueiro
Ángel Navia Vázquez

Área de Teoría de la Señal y Comunicaciones
Universidad Carlos III de Madrid

Búsqueda. Tipos de Búsqueda

Se ha visto que, en la aproximación máquina, se trata de minimizar un coste definido a partir de las salidas deseadas, $d^{(k)}$, y las que proporciona la máquina, $o^{(k)} = F_{\mathbf{w}}(\mathbf{x}^{(k)})$ (u $o^{(k)} = f_{\mathbf{w}}(\mathbf{x}^{(k)})$), acumulando sus valores según k; en definitiva, una forma $C_{\mathbf{w}}$, cuya variación con **w** es arbitraria:



y se trata de buscar el mínimo absoluto, o, en su defecto, uno relativo aceptable.

No siempre es posible obtener una solución cerrada (como en el caso de regresión lineal, p. ej.): en cuyo caso hay que recurrir a aplicar un algoritmo de búsqueda de dicho mínimo.

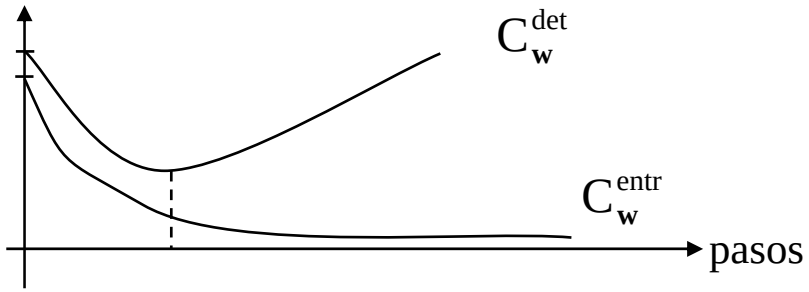
E, incluso cuando sea accesible una solución cerrada, puede interesar aplicar una búsqueda para conseguir generalización: ya que la minimización para las muestras no equivale a la minimización para todo el espacio de observación, y puede producirse sobreajuste; lo que debe evitarse:

- **regularizando** la solución;
- eligiendo un **coste** adecuado;
- o bien, entrenando hasta un cierto límite: lo que es posible, p.ej., mediante la detención de un **algoritmo iterativo** de búsqueda.

Un algoritmo iterativo va modificando (por pasos) **w** con objeto de reducir $C_{\mathbf{w}}$: si se dividen las muestras disponibles en dos conjuntos:

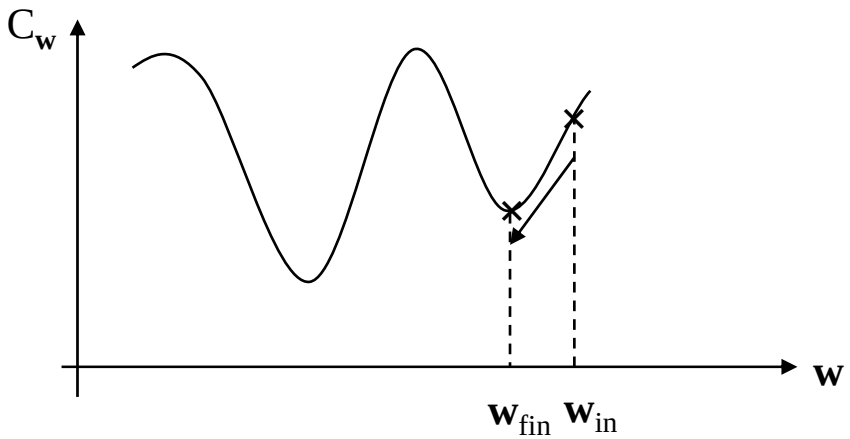
- el de entrenamiento: con cuyos elementos se hace la búsqueda;
- el de detención (o validación): que simplemente se usa para ir midiendo $C_{\mathbf{w}}$

(típicamente, el 80% y 20% de las muestras, respectivamente), se puede observar que, antes de llegar al mínimo coste en el conjunto de entrenamiento, repunta el de validación (al perder capacidad de generalización): ahí se puede detener la búsqueda



Hay dos tipos de métodos de búsqueda:

- **locales**: con los que se puede acceder a un mínimo relativo, a partir del punto en que se inicialice el algoritmo, que minimiza paso a paso el coste; con lo que sólo es posible llegar al mínimo que se encuentre en la cuenca de atracción del punto inicial



- **globales**: que persiguen alcanzar el mínimo absoluto.

Los métodos de búsqueda globales son, como es fácil comprender, complejos, delicados y lentos: computacionalmente muy costosos; por lo que sólo se aplican en casos en que merezca la pena perseguir el mínimo absoluto (ya que, si basta un buen mínimo relativo, suele ser suficiente repetir búsquedas locales con distintas inicializaciones y seleccionar la mejor solución). Tienen fundamentos muy distintos: pero todos incluyen un mecanismo de **exploración** del espacio de soluciones (**w**) y uno de **explotación** de los resultados favorables de dicha exploración.

Búsqueda local

A. Algoritmos de gradiente

Si se trabajase mediante procedimientos analíticos, es decir, conociendo $\overline{C}(\mathbf{w})$ (el coste medio, que incluye la promediación sobre todo el espacio muestral), y $\mathbf{g}(\mathbf{w})=\nabla_{\mathbf{w}}\overline{C}(\mathbf{w})$ es su gradiente, está claro que proceder de la forma

$$\mathbf{w}^{\text{nuevo}} = \mathbf{w}^{\text{ant}} - \eta \mathbf{g}\left(\mathbf{w}^{\text{ant}}\right) \qquad (\eta > 0)$$

conducirá a un mínimo local, si η es suficientemente pequeño.

Este procedimiento se conoce como **algoritmo del descenso más pendiente** ("steepest descent"): requiere conocer $\overline{C}(\mathbf{w})$, lo que no es habitual.

En situaciones muestrales, se puede trabajar con el coste para las muestras; aunque puede procederse para la totalidad (o subconjuntos del total), es frecuente y recomendable hacerlo secuencialmente (muestra a muestra: ciclando éstas tantas veces como se precise, y extendiendo de este modo {k}):

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} - \eta \mathbf{g}^{(k)}\left(\mathbf{w}^{(k)}\right)$$

que es el **algoritmo (secuencial) de gradiente**. En esta regla, $\mathbf{g}^{(k)}$ es el gradiente del coste evaluado para la k-ésima muestra.

(Nótese que es adaptativo si k es el tiempo).

Discutiremos su comportamiento en media, sustituyendo $\mathbf{g}^{(k)}$ por $\mathbf{g}$: pero conviene aclarar que al coste mínimo teórico se añadirá un **error de desajuste** debido al efecto del muestreo.

En las cercanías del mínimo buscado, $\mathbf{w}_0$, se podrá escribir:

$$C(\mathbf{w}) = C(\mathbf{w}_0) + (\mathbf{w} - \mathbf{w}_0)^T \mathbf{g}(\mathbf{w}_0) + \frac{1}{2}(\mathbf{w} - \mathbf{w}_0)^T H_{\mathbf{w}}(\mathbf{w}_0)(\mathbf{w} - \mathbf{w}_0)$$

siendo $H_{\mathbf{w}}$ la matriz hessiana de $C(\mathbf{w})$

$$H_{\mathbf{w}}]_{i_1 i_2} = \frac{\partial^2 C(\mathbf{w})}{\partial w_{i_1} \partial w_{i_2}}$$

y, por ser $\mathbf{w}_0$ un mínimo (local), el segundo término de la derecha es nulo; descomponiendo $H_{\mathbf{w}}(\mathbf{w}_0)$ en sus vectores propios

$$H_{\mathbf{w}}(\mathbf{w}_0)\, \mathbf{v}_i = \lambda_i \mathbf{v}_i$$

y escribiendo

$$\mathbf{w} - \mathbf{w}_0 = \sum_i \alpha_i \mathbf{v}_i$$

se tiene

$$C(\mathbf{w}) = C(\mathbf{w}_0) + \frac{1}{2}\left(\sum_i \alpha_i \mathbf{v}_i\right)^T H_{\mathbf{w}}(\mathbf{w}_0)\left(\sum_i \alpha_i \mathbf{v}_i\right) = C(\mathbf{w}_0) + \frac{1}{2}\sum_i \lambda_i \alpha_i^2$$

luego $\mathbf{w}_0$ es un mínimo si $H_{\mathbf{w}}(\mathbf{w}_0)$ es definida positiva: $\lambda_i > 0$, $\forall$ i

Tomando ahora gradientes sobre

$$C(\mathbf{w}) = C(\mathbf{w}_0) + \frac{1}{2}(\mathbf{w} - \mathbf{w}_0)^T H_{\mathbf{w}}(\mathbf{w}_0)(\mathbf{w} - \mathbf{w}_0)$$

resulta

$$\mathbf{g}(\mathbf{w}) = H_{\mathbf{w}}(\mathbf{w}_0)(\mathbf{w} - \mathbf{w}_0) = \sum_i \lambda_i \alpha_i \mathbf{v}_i$$

con lo que el algoritmo $\left(\mathbf{w}^{\text{nuevo}} = \mathbf{w}^{\text{ant}} - \eta \mathbf{g}\left(\mathbf{w}^{\text{ant}}\right)\right)$ resulta

$$\sum_i \alpha_i^{\text{nueva}} \mathbf{v}_i = \sum_i \alpha_i^{\text{ant}} \mathbf{v}_i - \eta \sum_i \lambda_i \alpha_i^{\text{ant}} \mathbf{v}_i$$

y, siendo los vectores propios linealmente independientes,

$$\alpha_i^{\text{nueva}} = (1 - \eta\lambda_i)\alpha_i^{\text{ant}}$$

Así, para la correspondiente $\mathbf{v}_i$, la búsqueda empieza en un cierto punto $\alpha_i^{(0)}$ y sigue una progresión geométrica de razón $1-\eta\lambda_i$; que converge si

$$|1 - \eta\lambda_i| < 1 : \qquad \eta < 2/\lambda_i \quad (< 2/\lambda_{\max})$$

Nótese que la convergencia de la sucesión de $\mathbf{v}_i$ es lo más rápida posible cuando $\eta=1/\lambda_i$. Sin embargo, ésta puede ser una mala elección para otras componentes.

Ejercicio:

Definiendo la velocidad de convergencia como

$$v_\eta = \min_i \left| \frac{1}{1-\eta\lambda_i} \right|$$

demuestre que el valor de η *que maximiza* v_η *es*

$$\eta_{opt} = \arg\left\{ \max_\eta \{ v_\eta \} \right\} = \frac{2}{\lambda_{min} + \lambda_{max}}$$

La velocidad no es el único factor a tener en cuenta en la elección del paso. En general, cuanto mayor sea η, mayor será el **error de desajuste** debido al procesado muestra a muestra.

Para el caso de coste cuadrático y estimador lineal:

$$C(\mathbf{w}) = \frac{1}{2}E\left\{\left(s-\mathbf{w}^T\mathbf{x}\right)^2\right\}$$

$$\frac{\partial C}{\partial w_i} = -E\left\{\left(s-\mathbf{w}^T\mathbf{x}\right)x_i\right\}$$

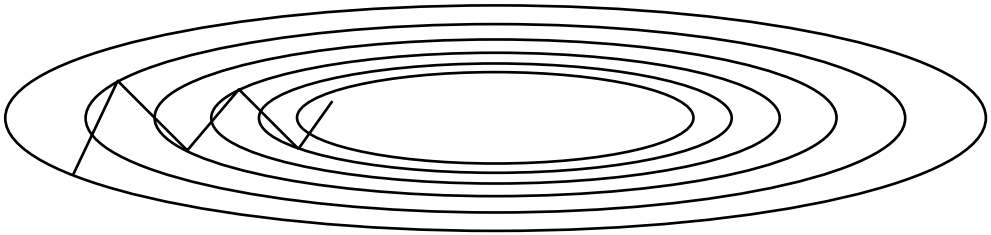
$$\frac{\partial^2 C}{\partial w_i \partial w_j} = E\left\{x_i x_j\right\}$$

el hessiano es la matriz de autocorrelación R_{xx}; y el algoritmo secuencial de gradiente

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \eta\left(s^{(k)} - \mathbf{w}^{(k)^T}\mathbf{x}^{(k)}\right)\mathbf{x}^{(k)}$$

que se conoce como **LMS** ("Least Mean Squares") o de **Widrow-Hoff**: muy frecuentemente aplicado para tiempo discreto k, y del que hay que destacar su robustez y su notable capacidad de seguimiento en casos no estacionarios.

La principal debilidad de estos algoritmos radica en su ineficiencia cuando H tiene autovalores muy dispersos (recuérdese la discusión de convergencia): lo que ocurre entonces es que la superficie del coste en el entorno del mínimo tiene aspecto fuertemente hiperelíptico, y los sucesivos gradientes implican una ineficaz búsqueda en “zig-zag”.



T: Formas de evitar este inconveniente de los algoritmos de gradiente.

Hay muchas variantes de estos algoritmos para evitar (otras) limitaciones: p. ej., para acelerar la búsqueda cuando se atraviesa una zona llana de C(**w**)

- puede aplicarse el **algoritmo del momento**

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} - \eta \mathbf{g}^{(k)}\left(\mathbf{w}^{(k)}\right) + \beta\left[\mathbf{w}^{(k)} - \mathbf{w}^{(k-1)}\right]$$

  en que el último término proporciona "inercia" a la actualización de los parámetros

- puede realizarse gestión del escalón: aumentándolo suavemente cuando el error va decreciendo, y reduciéndolo rápidamente si el error crece

T: Estudiar y aplicar algoritmos de gestión del escalón.

También es popular el **algoritmo NLMS** ("Normalized LMS"), en el que se evita la dependencia del tamaño de las muestras utilizando como escalón

$1/\left(\varepsilon + \left\|\mathbf{x}^{(k)}\right\|_2^2\right)$ (ε evita problemas numéricos)

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \frac{1}{\varepsilon + \left\|\mathbf{x}^{(k)}\right\|_2^2}\left(s^{(k)} - \mathbf{w}^{(k)^T}\mathbf{x}^{(k)}\right)\ \mathbf{x}^{(k)}$$

(nótese que se satisface "naturalmente" la condición de convergencia, ya que $\mathrm{tr}(R_{xx}) > \lambda_{max}$, y $\left\|\mathbf{x}^{(k)}\right\|_2^2$ es la traza muestral).

Nociones de otras búsquedas locales

A. Búsqueda en línea

Se elige una dirección de búsqueda $\mathbf{d}^{(k)}$, y se aplica

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \eta^{(k)}\mathbf{d}^{(k)}$$

eligiendo $\eta^{(k)}$ para la máxima reducción del coste

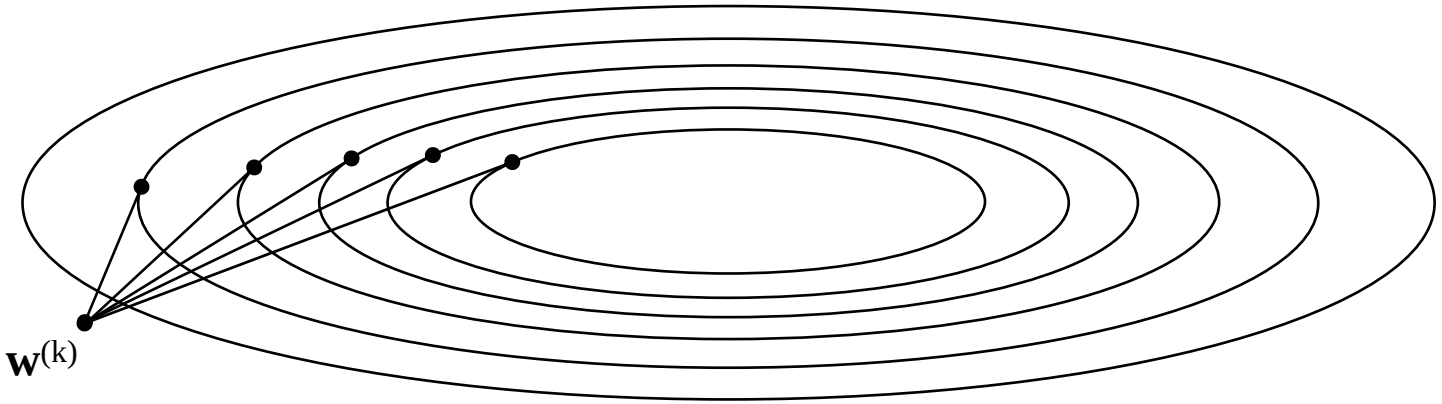
$$\frac{\partial C\left(\mathbf{w}^{(k+1)}\right)}{\partial \eta^{(k)}} = \left[\frac{\partial C\left(\mathbf{w}^{(k+1)}\right)}{\partial \mathbf{w}^{(k+1)}}\right]^T \frac{\partial \mathbf{w}^{(k+1)}}{\partial \eta^{(k)}} = \mathbf{g}^T\left(\mathbf{w}^{(k+1)}\right)\frac{\partial\left(\mathbf{w}^{(k)} + \eta^{(k)}\mathbf{d}^{(k)}\right)}{\partial \eta^{(k)}} =$$

$$= \mathbf{g}^T\left(\mathbf{w}^{(k+1)}\right)\mathbf{d}^{(k)} = 0$$

obteniéndose así los **algoritmos de descenso más rápido**.

En la práctica, $\eta^{(k)}$ se determina minimizando $C(\mathbf{w}^{(k)} + \eta^{(k)}\mathbf{d}^{(k)})$ por búsqueda directa unidimensional.

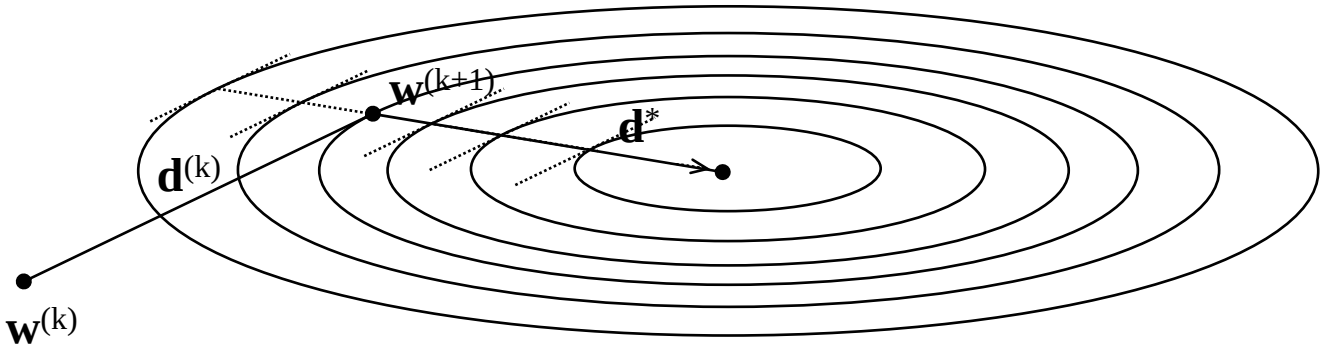
La condición $\mathbf{g}^{T}\left(\mathbf{w}^{(k+1)}\right)\mathbf{d}^{(k)} = 0$ implica que, en el punto de destino, la dirección de búsqueda es siempre tangente a la curva de nivel:



¿Cómo determinar la dirección de búsqueda?

Obsérvese cómo al elegir $\mathbf{d}^{(k)} = -\mathbf{g}(\mathbf{w}^{(k)})$, los gradientes consecutivos son ortogonales, lo que suele llevar a búsqueda en zig-zag.

Sin embargo, nótese también que, admitiendo forma del coste de 2º orden, en cualquier punto situado en la dirección óptima, **d***, el gradiente es nulo en la dirección de búsqueda anterior.



Lo anterior puede imponerse como condición: elíjase $\mathbf{d}^{(k+1)}$ de tal modo que

$$\mathbf{g}^{T}\left(\mathbf{w}^{(k+2)}\right)\mathbf{d}^{(k)} = 0$$

Éste es el fundamento de los algoritmos de gradiente conjugado.

B. **Algoritmos de gradiente conjugado**

Son búsquedas en línea (óptimas) en que se eligen las $\mathbf{d}^{(k)}$ de modo que no se perturbe lo progresado en pasos anteriores: concretamente, obligando en cada paso a que el gradiente en el destino sea ortogonal a la dirección de búsqueda previa

$$\mathbf{g}^{T}\left(\mathbf{w}^{(k+2)}\right)\mathbf{d}^{(k)} = \mathbf{g}^{T}\left(\mathbf{w}^{(k+1)} + \eta^{(k+1)}\mathbf{d}^{(k+1)}\right)\mathbf{d}^{(k)} = 0$$

de donde, admitiendo forma de 2º orden:

$$\begin{aligned}\mathbf{g}\left(\mathbf{w}^{(k+2)}\right) &= \mathrm{H}_{\mathbf{w}}\left(\mathbf{w}_0\right)\left(\mathbf{w}^{(k+2)} - \mathbf{w}_0\right) = \\ &= \mathrm{H}_{\mathbf{w}}\left(\mathbf{w}_0\right)\left(\mathbf{w}^{(k+1)} + \eta^{(k+1)}\mathbf{d}^{(k+1)} - \mathbf{w}_0\right) = \\ &= \mathrm{H}_{\mathbf{w}}\left(\mathbf{w}_0\right)\left(\mathbf{w}^{(k+1)} - \mathbf{w}_0\right) + \eta^{(k+1)}\mathrm{H}_{\mathbf{w}}\left(\mathbf{w}_0\right)\mathbf{d}^{(k+1)} = \\ &= \mathbf{g}\left(\mathbf{w}^{(k+1)}\right) + \eta^{(k+1)}\mathrm{H}_{\mathbf{w}}\left(\mathbf{w}_0\right)\mathbf{d}^{(k+1)}\end{aligned}$$

con lo que la condición queda

$$\mathbf{d}^{(k)^{T}}\mathbf{g}\left(\mathbf{w}^{(k+2)}\right) = \mathbf{d}^{(k)^{T}}\mathbf{g}\left(\mathbf{w}^{(k+1)}\right) + \eta^{(k+1)}\mathbf{d}^{(k)^{T}}\mathrm{H}_{\mathbf{w}}\left(\mathbf{w}_0\right)\mathbf{d}^{(k+1)} = 0$$

y siendo 0 el primer sumando del segundo miembro,

$$\mathbf{d}^{(k)^T} \mathrm{H}_{\mathbf{w}}\left(\mathbf{w}_0\right)\mathbf{d}^{(k+1)} = 0$$

que es la condición de conjugación. Si se aplica consecutivamente en un espacio de dimensión P (nº de parámetros) bajo la hipótesis de superficie de 2º orden, da lugar a P direcciones conjugadas que permiten resolver el problema en P pasos (no es así si la superficie no es de 2º orden).

Las formas prácticas de estos algoritmos se obtienen de aplicar la condición de conjugación a

$$\mathbf{d}^{(k+1)} = -\mathbf{g}\left(\mathbf{w}^{(k+1)}\right) + \beta^{(k)}\mathbf{d}^{(k)}$$

sustituyendo en $\mathbf{d}^{(k+1)^T} \mathrm{H}_{\mathbf{w}}\left(\mathbf{w}_0\right)\mathbf{d}^{(k)} = 0$, se puede despejar

$$\beta^{(k)} = \frac{\mathbf{g}\left(\mathbf{w}^{(k+1)}\right) \mathrm{H}_{\mathbf{w}}\left(\mathbf{w}_0\right)\mathbf{d}^{(k)}}{\mathbf{d}^{(k)^T} \mathrm{H}_{\mathbf{w}}\left(\mathbf{w}_0\right)\mathbf{d}^{(k)}}$$

Obviamente, $\beta^{(k)}$ no se puede obtener de esta expresión teórica, ya que no se conoce $\mathbf{H}_{\mathbf{w}}(\mathbf{w}_0)$:

- si se sustituye $\mathbf{H}_{\mathbf{w}}(\mathbf{w}_0)\mathbf{d}^{(k)}$ por $[\mathbf{g}(\mathbf{w}^{(k+1)}) - \mathbf{g}(\mathbf{w}^{(k)})]/\eta^{(k)}$
  (igualdad que se obtiene de la expresión de $\mathbf{g}(\mathbf{w}^{(k+2)})$ en la p. 9.18, cambiando k+2 por k+1), se llega al **algoritmo de Hestenes-Stiefel**

$$\beta_{HS}^{(k)} = \frac{\mathbf{g}^{T}\left(\mathbf{w}^{(k+1)}\right)\left[\mathbf{g}\left(\mathbf{w}^{(k+1)}\right) - \mathbf{g}\left(\mathbf{w}^{(k)}\right)\right]}{\mathbf{d}^{(k)T}\left[\mathbf{g}\left(\mathbf{w}^{(k+1)}\right) - \mathbf{g}\left(\mathbf{w}^{(k)}\right)\right]}$$

- si ahora se considera la expresión de $\mathbf{d}^{(k+1)}$ en la p. 9.19, se traspone y se multiplican por $\mathbf{g}(\mathbf{w}^{(k+1)})$ ambos términos, se tiene:

$$\mathbf{d}^{(k+1)T}\mathbf{g}\left(\mathbf{w}^{(k+1)}\right) = -\mathbf{g}^{T}\left(\mathbf{w}^{(k+1)}\right)\mathbf{g}\left(\mathbf{w}^{(k+1)}\right) + \beta^{(k)}\mathbf{d}^{(k)T}\mathbf{g}\left(\mathbf{w}^{(k+1)}\right)$$

  y como el segundo sumando del término de la derecha es nulo (por descenso más rápido), se deduce cambiando k+1 por k:

$$\mathbf{d}^{(k)T}\mathbf{g}\left(\mathbf{w}^{(k)}\right) = -\mathbf{g}^{T}\left(\mathbf{w}^{(k)}\right)\mathbf{g}\left(\mathbf{w}^{(k)}\right)$$

que, llevada al denominador de $\beta_{HS}^{(k)}$, considerando además que $\mathbf{d}^{(k)T}\mathbf{g}(\mathbf{w}^{(k+1)})=0$, proporciona el **algoritmo de Polak-Ribiere**

$$\beta_{PL}^{(k)} = \frac{\mathbf{g}^T\left(\mathbf{w}^{(k+1)}\right)\left[\mathbf{g}\left(\mathbf{w}^{(k+1)}\right)-\mathbf{g}\left(\mathbf{w}^{(k)}\right)\right]}{\mathbf{g}^T\left(\mathbf{w}^{(k)}\right)\mathbf{g}\left(\mathbf{w}^{(k)}\right)}$$

– por último, si de la condición de descenso más rápido de la p. 9.15

$$\mathbf{g}^T\left(\mathbf{w}^{(k+1)}\right)\mathbf{d}^{(k)} = 0$$

se inserta la expresión de $\mathbf{d}^{(k)}$ de la p. 9.19, se tiene

$$\mathbf{g}^T\left(\mathbf{w}^{(k+1)}\right)\left[-\mathbf{g}\left(\mathbf{w}^{(k)}\right)+\beta^{(k-1)}\mathbf{d}^{(k-1)}\right]=0$$

de donde

$$-\mathbf{g}^T\left(\mathbf{w}^{(k+1)}\right)\mathbf{g}\left(\mathbf{w}^{(k)}\right)+\beta^{(k-1)}\mathbf{g}^T\left(\mathbf{w}^{(k+1)}\right)\mathbf{d}^{(k-1)} = 0$$

con lo que, como es nulo el segundo sumando del primer término, lo es el primero: eliminándolo en el numerador de $\beta_{PL}^{(k)}$, queda establecido el **algoritmo de Fletcher-Reeves:**

$$\beta_{FR}^{(k)} = \frac{\mathbf{g}^T\left(\mathbf{w}^{(k+1)}\right)\mathbf{g}\left(\mathbf{w}^{(k+1)}\right)}{\mathbf{g}^T\left(\mathbf{w}^{(k)}\right)\mathbf{g}\left(\mathbf{w}^{(k)}\right)}$$

Estos algoritmos

- son eficaces si la aproximación cuadrática es razonable: si no, pueden ser peores que uno de gradiente;
- deben “reinicializarse” (mediante gradiente) cada cierto número de pasos, para evitar los efectos de acumulación de errores.

C. El Método de Newton

Supuesto que nos encontramos en $\mathbf{w}^*$ y que es aceptable

$$C(\mathbf{w}) = C(\mathbf{w}^*) + (\mathbf{w} - \mathbf{w}^*)^T \mathbf{g}(\mathbf{w}^*) + \frac{1}{2}(\mathbf{w} - \mathbf{w}^*)^T H_{\mathbf{w}}(\mathbf{w}^*)(\mathbf{w} - \mathbf{w}^*)$$

tomando gradientes

$$\mathbf{g}(\mathbf{w}) = \mathbf{g}(\mathbf{w}^*) + H_{\mathbf{w}}(\mathbf{w}^*)(\mathbf{w} - \mathbf{w}^*)$$

que será nulo si $\mathbf{w}$ es el mínimo buscado; de donde

$$\mathbf{w} = \mathbf{w}^* - H_{\mathbf{w}}^{-1}(\mathbf{w}^*)\mathbf{g}(\mathbf{w}^*)$$

cuya aplicación de forma iterada

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} - H_{\mathbf{w}}^{-1}(\mathbf{w}^{(k)})\mathbf{g}(\mathbf{w}^{(k)})$$

es el algoritmo correspondiente a este método.

No se aplica directamente por la carga que supone

- calcular H_w en cada paso
- invertir la matriz

y en su lugar se utilizan

* los métodos Seudo-Newton: reduciendo H_w a sus términos diagonales (haciendo positivos los negativos; añadiendo una cte. ε para evitar inestabilidades numéricas)
* los métodos Quasi-Newton: basados en aproximar iterativamente H_w^{-1} imponiendo la "condición Quasi-Newton"

$$\mathbf{w}^{(k+1)} - \mathbf{w}^{(k)} = H_{\mathbf{w}}^{-1}\left(\mathbf{w}^{(k)}\right)\left[\mathbf{g}\left(\mathbf{w}^{(k+1)}\right) - \mathbf{g}\left(\mathbf{w}^{(k)}\right)\right]$$

T: Discutir los métodos Q-N

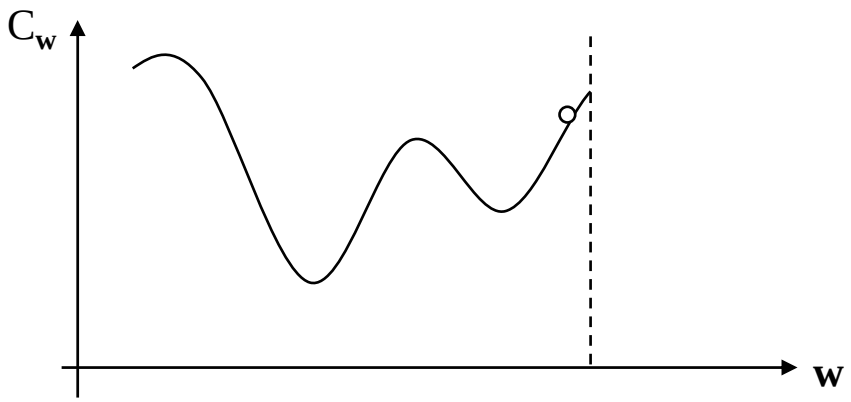
- *de Davidson-Fletcher-Powell (DFP)*
- *de Broyden-Fletcher-Goldfarb-Shanno (BFGS)*

Búsqueda Global

A. **Temple Simulado** (“Simulated Annealing”)

(Como se procede con el temple del acero: enfriando lentamente para alcanzar un estado de mínima energía interna)

Se intuye con el símil del movimiento de una bola por una superficie de coste: con una agitación fuerte recorre toda la superficie, y si se va disminuyendo poco a poco tenderá a quedarse en el mínimo absoluto.



Aprendizaje de Boltzmann

1. Inicializar **w**
2. Inicializar la temperatura T a T_0 (valor alto)
3. Aplicar un observable al azar y obtener C
4. Elegir un parámetro al azar y modificarlo (según $G(0, T^2/2\pi)$, p. ej.) obteniendo $C+\Delta C$
5. - si $\Delta C < 0$: aceptar el cambio
   - si $\Delta C > 0$ y $\exp(-\Delta C/T) > U[0,1]$: aceptar el cambio
   - si $\Delta C > 0$ y $\exp(-\Delta C/T) < U[0,1]$: no aceptarlo
6. Volver a 4 hasta agotar los parámetros
7. Volver a 3 hasta agotar los observables
8. Aplicar un criterio de parada:
   - si se cumple, terminar
   - si no, reducir la temperatura según $T(l)=T_0/\ln(1+l)$, y volver a 3

T: Métodos de aceleración del Aprendizaje de Boltzmann

B. **Algoritmos Genéticos**

Propuestos por Holland a mediados de los 70, son búsquedas Neo-Darwinianas, con exploración por **cruce** y **mutación** sobre poblaciones de candidatos a la solución, y explotación mediante selección para **supervivencia**.

Algoritmo Genético Básico (AGB)

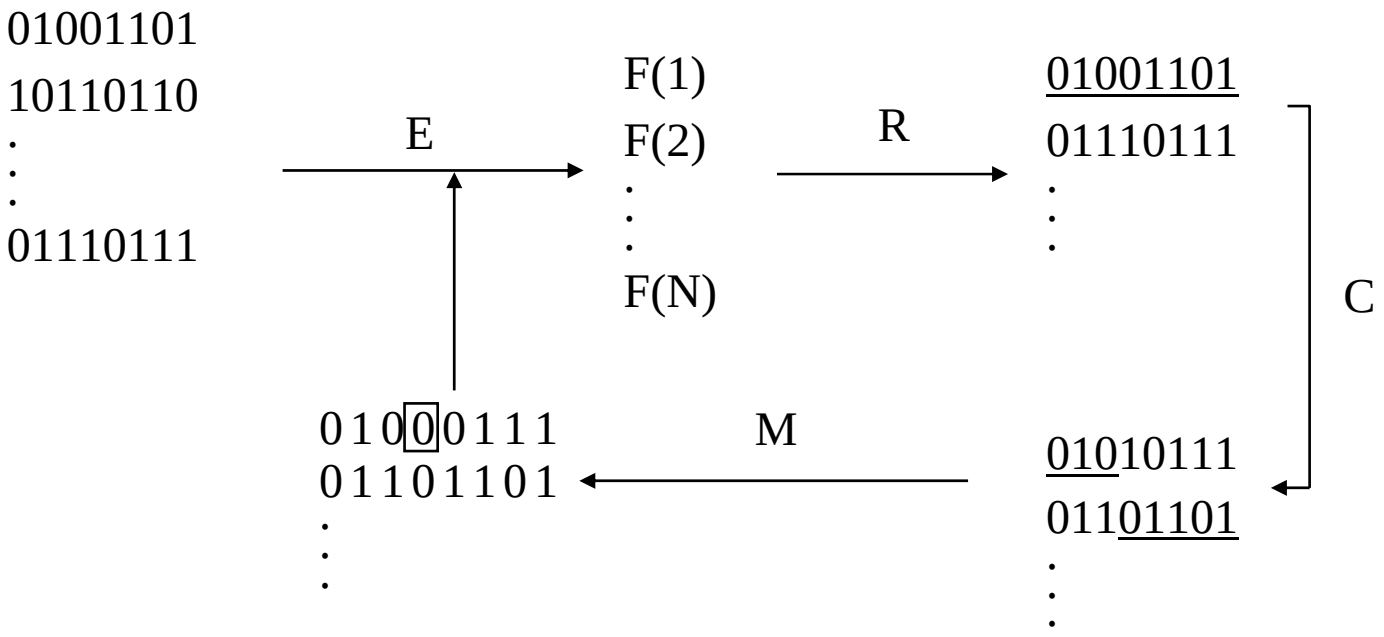
(Las componentes de la solución (**w**) se expresan en binario)

Inicialización:

1. Construir una población inicial aleatoria de tiras binarias
2. Evaluación: aplicar una medida de calidad: ajuste ("fitness") a un objetivo
3. Detención: aplicar un criterio de parada; si no se cumple
4. Reproducción: generar una nueva población, sorteando el paso de los individuos de acuerdo con su calidad
5. Cruce: tomar aleatoriamente dos individuos, cortarlos por un (mismo) punto aleatorio, y cruzar los trozos
6. Mutación: por sorteo (de baja probabilidad) para cada bit de la población
7. Volver a 2

(Reproducción, cruce, mutación: operadores)

Ilustración



Formatos típicos:

- Tamaño de la población: alrededor de 50 individuos
- Tasa de cruce: 0.6 - 0.8
- Tasa de mutación: 10^{-2} - 10^{-3}
- Longitud de los individuos: hasta varios cientos

Con estas características, el algoritmo busca una solución global, progresando (con oscilaciones) hacia mejores valores de calidad tanto del mejor individuo como en media para la población.

Tienen gran potencia de búsqueda.

Características de los AG

Limitaciones

* Son ciegos: no es fácil incorporar información previa
* Es difícil introducir restricciones sobre las soluciones: se suele hacer aplicando operadores que las mantengan
* Tienen gran carga computacional, debido a la evaluación
* Su aplicación directa no ofrece generalización, al calcular directamente la calidad
* Son proclives al estancamiento
  - por pérdida de variedad "genética" (en los bits homólogos de todos los individuos), que imposibilita la exploración;

    se puede corregir: por aumento de la mutación o por renovación de parte de la población

- por aparición de un superindividuo (muy bueno para su generación, pero no como solución final): que tiende a ocupar toda la población por su ventaja en la reproducción;

  se combate: equilibrando la reproducción; p. ej., procediendo según rango en la evaluación, y no de modo directamente proporcional a la calidad.

  (con este tipo de precaución, se pueden aplicar estrategias elitistas, pasando el mejor individuo de cada generación directamente a la siguiente, para evitar pérdidas de difícil reparación)

* No tienen capacidad de seguimiento.

  Pero pueden combinarse con búsquedas locales (que, además, serán vía para obtener generalización), según principios

  - Lamarckistas: se heredan los resultados;
  - Baldwinianas: no se hereda la búsqueda local, pero se emplea en la evaluación.

* El AGB es de aplicación limitada por la codificación binaria: la codificación "real" requiere otros operadores para cruce (incluyendo fragmentación de cada posición) y mutación.

Ventajas

(aparte de dicha potencia de búsqueda)

* No requieren formulación analítica: la evaluación puede llevarse a cabo mediante la aplicación directa de las soluciones al problema (p. ej., haciendo jugar las estrategias que se diseñen para un juego)

* Pueden introducirse muchos otros operadores de búsqueda "ad hoc" para el problema; así como codificaciones diferentes de las tiras de bits

T: Otros métodos de búsqueda global son:

- *las estrategias evolutivas*
- *las búsqueda "Tabú"*
- *los algoritmos "branch and bound"*

Descríbase su aplicación y discútanse sus principales características.