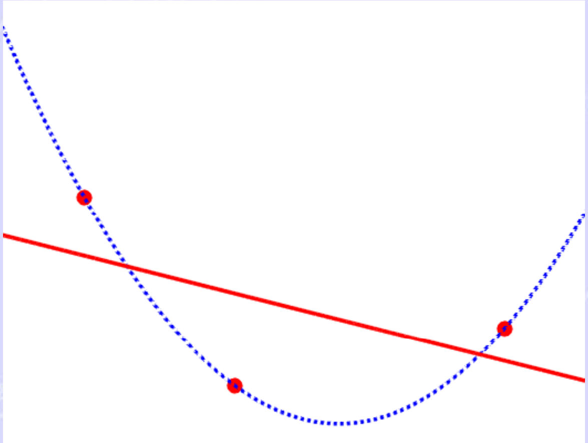
TEMA 3.

AJUSTE de DATOS

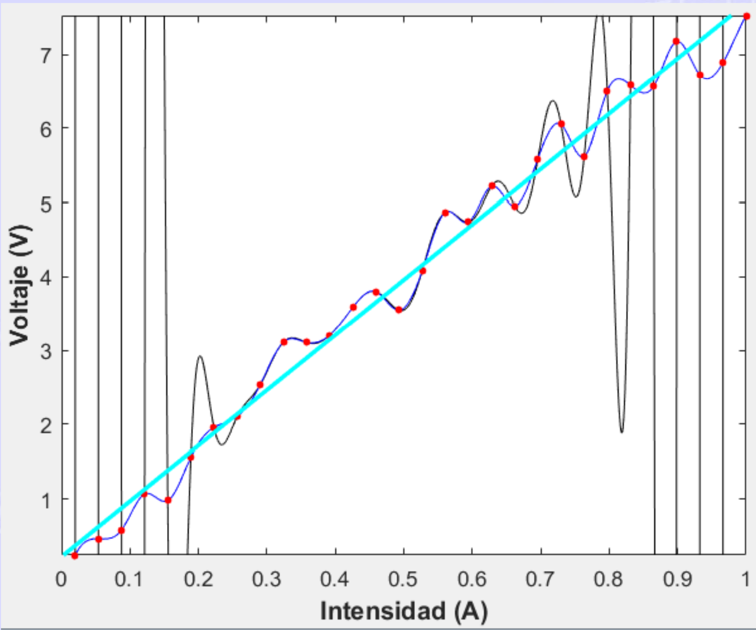
Más condiciones que parámetros

No existe una solución que pase por los puntos (INTERPOLACIÓN)

Buscamos la solución que pase "cerca" de los puntos (AJUSTE)



Ajuste de datos



¿Cuál de estas alternativas es preferible?

Ajuste de datos

Es un problema muy importante en la práctica:

- Reduzco ruido de las medidas, detecto patrón general.
- Tengo redundancia en caso de algún error.
- Describo muchos datos con menos parámetros → Compresión

La pega es que el problema no está bien definido:

HAY UNANIMIDAD EN EL SIGNIFICADO DE PASAR POR ENCIMA (=)

PASAR CERCA (~) PUEDE SIGNIFICAR DISTINTAS COSAS

Hay que definir un criterio de "semejanza"

Ajuste de datos

En interpolación: nº coeficientes (incógnitas) = nº datos (ecuaciones)

En ajuste: permitimos que número de coeficientes < número de datos

nº incógnitas < nº ecuaciones ???

No se podrán verificar exactamente (=) las ecuaciones.
→ Nos conformaremos con que más o menos se cumplan (~).
→ Llegaremos a sistemas lineales **sobredeterminados**:

N ecuaciones

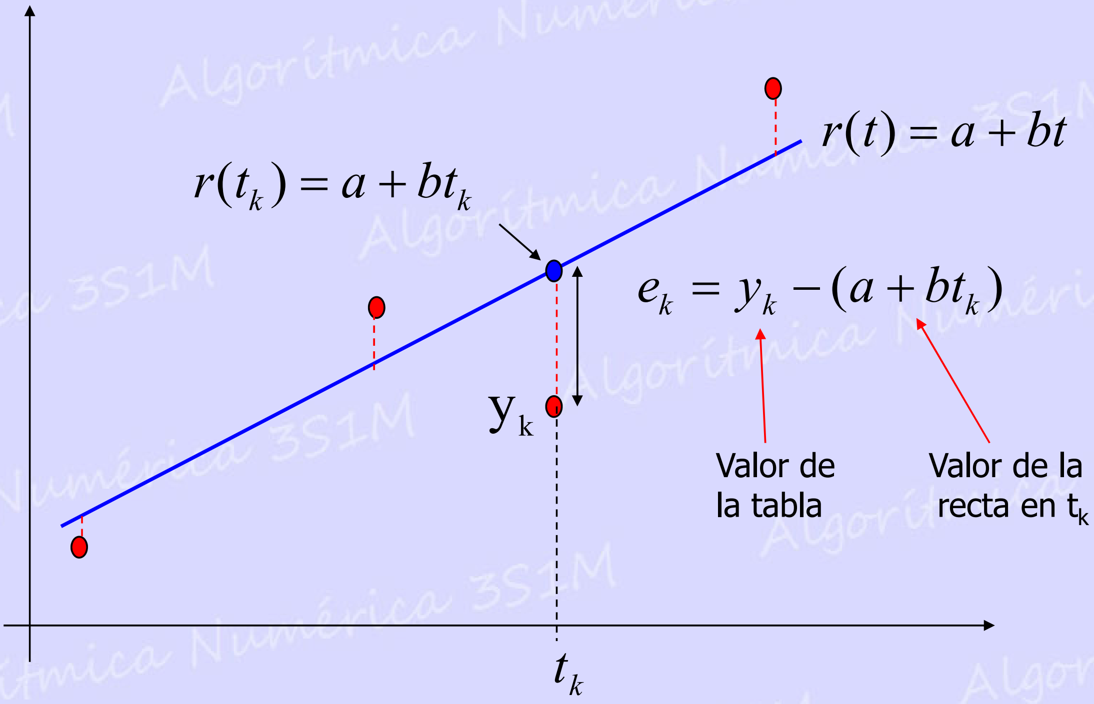
$$\Big(H \Big) \cdot \big(c \big) \approx \Big(v \Big)$$

n incógnitas (n<N)

Tras resolver no pasamos por los puntos sino cerca de ellos

El problema clásico: regresión lineal

"Mejor" ajuste de una recta r(t) a una nube (tabla) de puntos $\{t_k, y_k\}$:



Distancia a minimizar: mínimos cuadrados

$$E = \sum_{k=1}^{N} e_k^2 = \sum_{k=1}^{N} \left(y_k - r(t_k)\right)^2 = \sum_{k=1}^{N} \left(y_k - a - bt_k\right)^2$$

Hallar los coeficientes (a,b) que minimicen E, el error global del ajuste

¿Hallar un mínimo?

$$\frac{\partial E}{\partial a} = 2\sum_{k=1}^{N} \left(y_k - a - bt_k\right)\left(-1\right) = 0$$

$$\frac{\partial E}{\partial b} = 2\sum_{k=1}^{N} \left(y_k - a - bt_k\right)\left(-t_k\right) = 0$$

Distancia a minimizar : mínimos cuadrados

$$E = \sum_{k=1}^{N} e_k^2 = \sum_{k=1}^{N} \left(y_k - r(t_k)\right)^2 = \sum_{k=1}^{N} \left(y_k - a - bt_k\right)^2$$

Hallar los coeficientes (a,b) que minimicen E, el error global del ajuste

¿Hallar un mínimo?

$$\sum_{k=1}^{N} \left(y_k - a - bt_k\right) = 0 \qquad \sum_{k=1}^{N} y_k = a\sum_{k=1}^{N} 1 + b\sum_{k=1}^{N} t_k$$

$$\sum_{k=1}^{N} \left(y_k - a - bt_k\right)\cdot t_k = 0 \qquad \Longrightarrow \qquad \sum_{k=1}^{N} t_k y_k = a\sum_{k=1}^{N} t_k + b\sum_{k=1}^{N} t_k^2$$

Distancia a minimizar : mínimos cuadrados

$$E = \sum_{k=1}^{N} e_k^2 = \sum_{k=1}^{N} \left(y_k - r(t_k)\right)^2 = \sum_{k=1}^{N} \left(y_k - a - bt_k\right)^2$$

Hallar los coeficientes (a,b) que minimicen E, el error global del ajuste

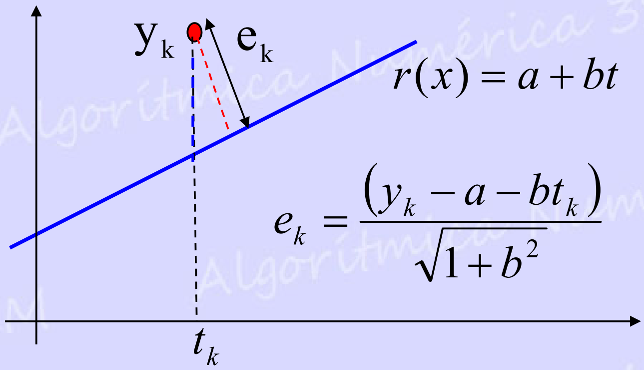
¿Hallar un mínimo?

$$\begin{pmatrix} N & \sum_{k=1}^{N} t_k \\ \sum_{k=1}^{N} t_k & \sum_{k=1}^{N} t_k^2 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} \sum_{k=1}^{N} y_k \\ \sum_{k=1}^{N} y_k t_k \end{pmatrix} \qquad \begin{matrix} a = \ldots \\ b = \ldots \end{matrix}$$

Los valores $\sum_{k=1}^{N} t_k, \sum_{k=1}^{N} t_k^2, \sum_{k=1}^{N} y_k, \sum_{k=1}^{N} y_k t_k$ salen de los datos de la tabla.

Objeciones a esta solución

¿No sería mejor definir el error como la distancia mínima a la recta?



Además, el criterio a minimizar $E = \sum_{k=1}^{N} e_k^2$ es bastante arbitrario.

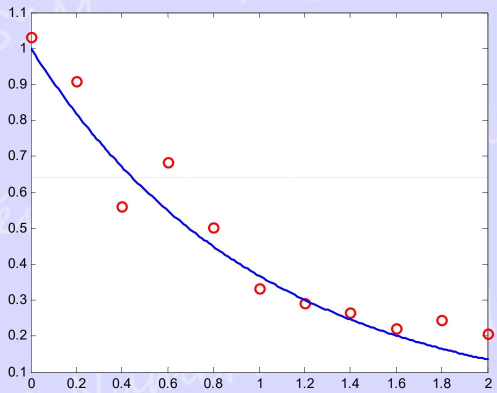
Otras posibilidades tan buenas (o mejores) $E = \sum_{k=1}^{N} |e_k|$ $\quad E = \max\{|e_k|\}$

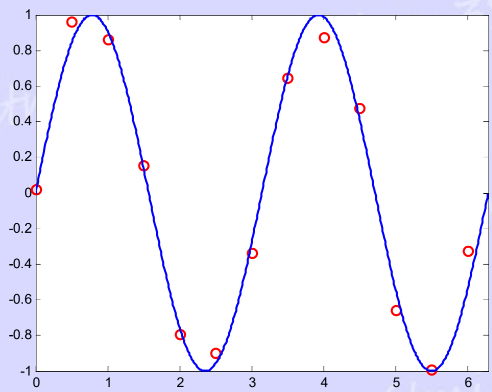
¿Por qué usamos mínimos cuadrados? Porque es (mucho) más fácil

Garantizamos que el problema se reduce a **resolver un sistema lineal.**

Ampliarlo a otros casos

¿Podríamos generalizar este ajuste de mínimos cuadrados a otras curvas?





Revisitar el problema de la recta

Planteamos el problema de la recta como si fuese una interpolación:

$$\boxed{\boxed{\begin{array}{l} a+bt_1 \approx y_1 \\ a+bt_2 \approx y_2 \end{array}} \\ \begin{array}{l} a+bt_3 \approx y_3 \\ \ldots \\ a+bt_N \approx y_N \end{array}}$$

→ Esto es lo que podemos hacer con interpolación.

→ Esto es imposible verificar con sólo 2 coeficientes.

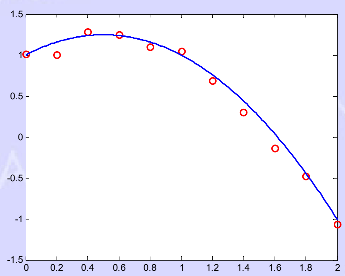
Pero si puedo plantearlo cambiando el = por un ~

$$\begin{pmatrix} 1 & t_1 \\ 1 & t_2 \\ 1 & t_3 \\ \ldots & \ldots \\ 1 & t_N \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} \approx \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \ldots \\ y_N \end{pmatrix} \qquad \begin{pmatrix} \\ \mathrm{H} \\ \\ \end{pmatrix} \begin{pmatrix} \mathrm{a} \\ \mathrm{b} \end{pmatrix} \approx \begin{pmatrix} \\ \mathrm{v} \\ \\ \end{pmatrix}$$

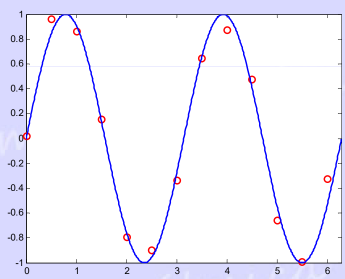
Base {1 , t}

Fácilmente generalizable a otros casos

$$\begin{pmatrix} 1 & t_1 & t_1^2 \\ 1 & t_2 & t_2^2 \\ 1 & t_3 & t_3^2 \\ \ldots & \ldots & \ldots \\ 1 & t_N & t_N^2 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \end{pmatrix} \approx \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \ldots \\ y_N \end{pmatrix}$$



$$\begin{pmatrix} 1 & \sin(t_1) & \cos(t_1) \\ 1 & \sin(t_2) & \cos(t_2) \\ 1 & \sin(t_3) & \cos(t_3) \\ \ldots & \ldots & \ldots \\ 1 & \sin(t_N) & \cos(t_N) \end{pmatrix} \begin{pmatrix} a \\ b \\ c \end{pmatrix} \approx \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \ldots \\ y_N \end{pmatrix}$$



La matriz H se construye como en la interpolación, pero ahora es más alta (más condiciones / ecuaciones) que ancha (incógnitas / coefs.)

¿Cómo resolver el sistema sobredeterminado H·c $\approx$ v al que llegamos?

¿Cómo resolver el sistema sobredeterminado?

$$\begin{pmatrix} \\ H \\ \\ \end{pmatrix} \begin{pmatrix} \bar{c} \end{pmatrix} \approx \begin{pmatrix} \\ v \\ \\ \end{pmatrix}$$

Para resolver H·c ≅ v hay que elegir un criterio.

Definimos el vector de residuos o errores como:

$$\bar{r} = \bar{v} - H \cdot \bar{c}$$

En interpolación conseguíamos hacer 0 todas sus componentes (pasabamos por todos los puntos)

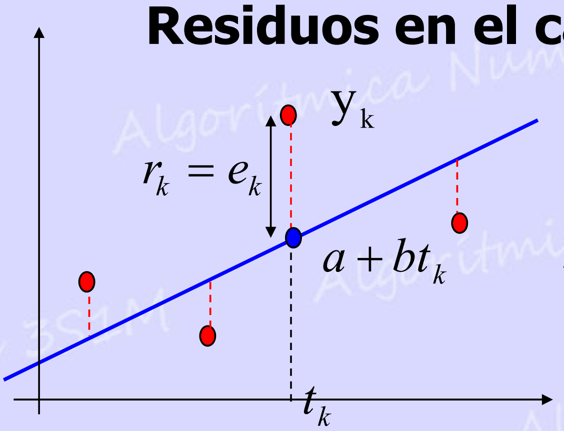
En ajuste no vamos a conseguir anularlo (no pasamos por los puntos).

Se trata de hacerlo pequeño ¿en qué sentido?

<u>COMO ANTES, USAREMOS EL CRITERIO de MÍNIMOS CUADRADOS:</u> la solución <u>c</u> debe minimizar la norma al cuadrado del vector de residuos:

$$\|\bar{r}\|_2^2 = \|\bar{v} - H \cdot \bar{c}\|_2^2$$

Residuos en el caso de la recta anterior



$$\bar{r} = \begin{pmatrix} y_1 \\ y_2 \\ \ldots \\ y_k \end{pmatrix} - \begin{pmatrix} 1 & t_1 \\ 1 & t_2 \\ \ldots & \ldots \\ 1 & t_k \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} y_1 - a - bt_1 \\ y_2 - a - bt_2 \\ \ldots \\ y_k - a - bt_k \end{pmatrix} = \begin{pmatrix} e_1 \\ e_2 \\ \ldots \\ e_k \end{pmatrix}$$

La k-ésima componente de <u>r</u> es: $r_k = (\bar{v} - H\bar{c})_k = y_k - (a + bt_k) = e_k$

La norma de <u>r</u> (al cuadrado) es: $\|\bar{r}\|_2^2 = \|\bar{v} - H\bar{c}\|_2^2 = \sum_k (\bar{v} - H\bar{c})_k^2 = \sum_k e_k^2$

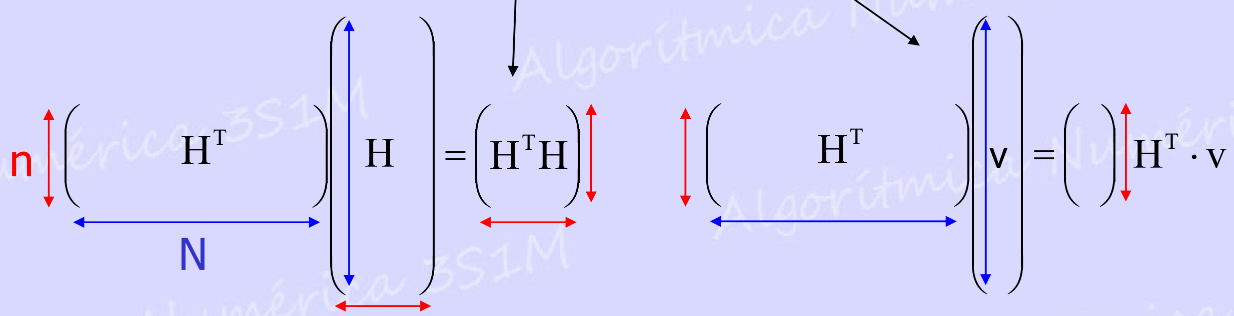
Minimizar $||r||^2$ equivale al criterio usado antes (minimizar $\sum_k e_k^2$) en el ajuste de la recta.

La solución del sistema sobredeterminado debe coincidir con la solución encontrada antes para el ajuste a una recta.

Solución de mínimos cuadrados

La solución $\underline{c}$ que minimiza la norma del vector de residuos $\underline{r} = (\underline{v} - H \cdot \underline{c})$ es la solución de las llamadas ecuaciones normales:

$$\left(H^T H\right) \cdot \overline{c} = H^T \cdot \overline{v}$$



No importa el número (N) de puntos iniciales presentes en el ajuste.

Al final terminamos con un sistema cuyas dimensiones se corresponde con el número de coeficientes n (típicamente n << N)

Caso de la recta anterior: 2 coefs (a,b) → terminábamos con un sistema 2 x 2

Ecuaciones Normales para el ajuste a una recta

$$H^T H = \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ t_1 & t_2 & t_3 & \dots & t_N \end{pmatrix} \begin{pmatrix} 1 & t_1 \\ 1 & t_2 \\ 1 & t_3 \\ \dots & \dots \\ 1 & t_N \end{pmatrix} = \begin{pmatrix} N & \sum t_k \\ \sum t_k & \sum t_k^2 \end{pmatrix}$$

$$H^T \overline{v} = \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ t_1 & t_2 & t_3 & \dots & t_N \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \dots \\ y_N \end{pmatrix} = \begin{pmatrix} \sum y_k \\ \sum t_k y_k \end{pmatrix}$$

$$\left(H^T H\right) \cdot \overline{c} = H^T \cdot \overline{v} \Rightarrow \begin{pmatrix} N & \sum t_k \\ \sum t_k & \sum t_k^2 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} \sum y_k \\ \sum t_k y_k \end{pmatrix}$$

Mismo sistema que salía antes.

Problema 1

Sea la tabla de datos:

ti	-1	0	1	2
fi	-2	-1	0	3

Plantear el sistema sobredeterminado obtenido al intentar ajustar los datos con:

$$f(t) = A + Bt + Ct^2$$

$$f(t) = A + Bt^2$$

$$f(t) = A\cos(t) + B\sin(t)$$

Sea la tabla de datos:

t_k	-1	0	1	2
y_k	-2	-1	0	3

Plantear el sistema sobredeterminado obtenido al ajustar los datos con un polinomio de grado 2 (parábola)

$p(t) = A + B{\cdot}t + C{\cdot}t^2$ → Combinación lineal de la base $\{1, t, t^2\}$

Matriz 4 x 3: 4 condiciones
3 coeficientes

$$\begin{pmatrix} 1 & -1 & 1 \\ 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 2 & 4 \end{pmatrix} \begin{pmatrix} A \\ B \\ C \end{pmatrix} \approx \begin{pmatrix} -2 \\ -1 \\ 0 \\ 3 \end{pmatrix}$$

¿Cómo hacemos esto en MATLAB?

Construimos H y v como lo hacíamos en la interpolación.

xk	-1	0	1	2
yk	-2	-1	0	3

```
xk =[-1  0 1 2]'; %Vector columna
yk =[-2 -1 0 3]'; %Vector columna
H = [xk.^0 xk.^1 xk.^2];
```

Una vez que tenemos H y v hay que resolver las ecuaciones normales:

$$H^T H \cdot c = H^T y \qquad \Rightarrow \qquad c = \left(H^T H\right)^{-1} \cdot \left(H^T y\right)$$

```
Q = (H' * H); b = H'*yk;   % Operador ' transpone matriz
size(Q), ans = 3 x 3       % (H'*H) es de tamaño 3x3
c = inv(Q)*b;              % o c = Q\b
```

El vector c contiene los tres coeficientes A,B,C de la solución

TODAVÍA más fácil

Construimos H y v como hacíamos con interpolación.

xk	-1	0	1	2
yk	-2	-1	0	3

```
xk =[-1  0 1 2]'; %Vector columna
yk =[-2 -1 0 3]'; %Vector columna
H = [xk.^0 xk.^1 xk.^2];
```

En MATLAB nos vale el mismo comando para resolver el caso exacto (interpolación) como el caso aproximado (ajuste):

```
c = H \ yk; % Coeficientes del ajuste de mínimos cuadrados
```

- Si H es cuadrada (invertible) corresponde a hacer: $c = H^{-1} y$
- Si H no es cuadrada, solución de mínimos cuadrados:

$$c = \left(H^T H\right)^{-1} H^T y$$

Cambiar la base usada es sencillo

Sea la tabla de datos:

ti	-1	0	1	2
fi	-2	-1	0	3

Plantear el sistema sobredeterminado obtenido al intentar ajustar los datos con:

1) $u(t) = A + Bt + Ct^2$

$$\begin{pmatrix} 1 & -1 & 1 \\ 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 2 & 4 \end{pmatrix} \begin{pmatrix} \mathrm{A} \\ \mathrm{B} \\ \mathrm{C} \end{pmatrix} \approx \begin{pmatrix} -2 \\ -1 \\ 0 \\ 3 \end{pmatrix}$$

2) $u(t) = A + Bt^2$

$$\begin{pmatrix} 1 & 1 \\ 1 & 0 \\ 1 & 1 \\ 1 & 4 \end{pmatrix} \begin{pmatrix} \mathrm{A} \\ \mathrm{B} \end{pmatrix} \approx \begin{pmatrix} -2 \\ -1 \\ 0 \\ 3 \end{pmatrix}$$

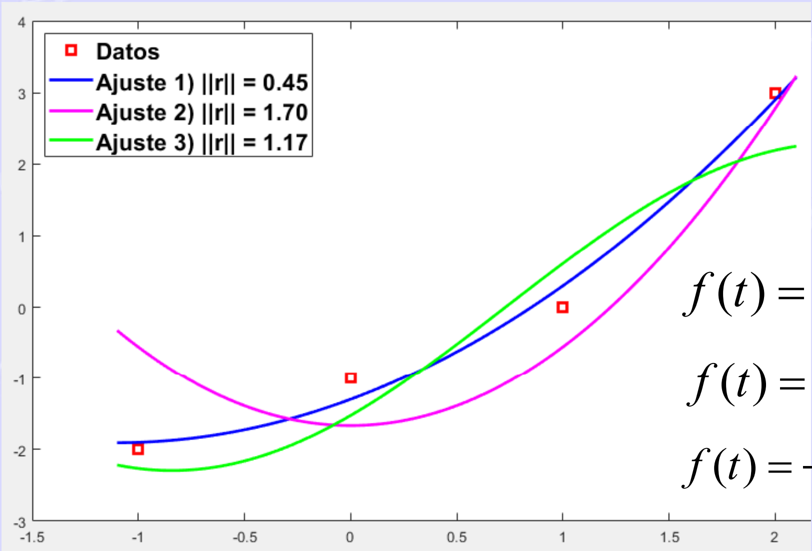
3) $u(t) = A\cos(t) + B\sin(t)$

$$\begin{pmatrix} \cos(-1) & \sin(-1) \\ \cos(0) & \sin(0) \\ \cos(1) & \sin(1) \\ \cos(2) & \sin(2) \end{pmatrix} \begin{pmatrix} \mathrm{A} \\ \mathrm{B} \end{pmatrix} \approx \begin{pmatrix} -2 \\ -1 \\ 0 \\ 3 \end{pmatrix}$$

Soluciones y gráficas

```
tt=(-1.1:0.01:2.1); % Para grafica
u1 = c(1)+c(2)*tt+c(3)*tt.^2;   % En cada caso se usarían
u2 = c(1)+c(2)*tt.^2;           % los coeficientes obtenidos
u3 = c(1)*sin(tt)+c(2)*cos(tt); % con la H de cada problema

plot(tk,yk,'rs',tt,u1,'b',tt,u2,'m‘,tt,u3,'b','LineWidth',2);
```



$$f(t) = -1.3 + 1.1 \cdot t + 0.5 \cdot t^2$$

$$f(t) = -1.67 + 1.11 \cdot t^2$$

$$f(t) = -1.53 \cdot \cos(t) + 1.71 \cdot \sin(t)$$

¿Qué nos queda por ver en ajuste?

El caso básico lo tenemos cubierto (y desde el punto de vista de su resolución en MATLAB es prácticamente idéntico a lo que hicimos en la interpolación).

Estudiaremos algunas variantes del problema de ajuste básico:

A) El ajuste no se plantea con funciones del tipo $u(t) = \sum_{k=1}^{n} c_k b_k(t)$

¿Cómo resolver un mejor ajuste con una función del tipo $u(t) = Ae^{\beta t}$?

B) Ajuste con restricciones: el problema tiene una mezcla de condiciones estrictas (hay que cumplirlas de forma exacta) y otras aproximadas. Mezclamos aspectos de interpolación (estamos obligados a pasar por ciertos puntos) y de ajuste (nos basta con pasar cerca de los otros).

C) USO de PESOS: sin llegar al caso anterior de tener algunas condiciones exactas, es posible que ciertos datos sean más importantes que otros (p.e. han sido tomados por un observador más experimentado). ¿Cómo reflejar esa diferencia de calidades en el ajuste?

Ajuste con una exponencial $u(t)=A\cdot e^{\beta t}$

¿Ajustar una tabla de datos $\{t_k, y_k\}$ a una exponencial $u(t)=A\cdot e^{\beta\cdot t}$?

Tenemos 2 parámetros pero no podemos escribir $u(t) = A\cdot(\ ) + \beta\cdot(\ )$

El objetivo es lograr que: $y_k \approx u(t_k) = Ae^{\beta t_k}$

Si aplicamos logaritmos: $\log(y_k) \approx \log(Ae^{\beta t_k}) = \log A + \beta\cdot t_k = a + b\cdot t_k$

Nuevo problema: ajustar los datos $\{t_k, \log(y_k)\}$ con una recta $(a+b\cdot t_k)$

$$\begin{pmatrix} 1 & t_1 \\ 1 & t_2 \\ 1 & t_3 \\ \dots & \dots \\ 1 & t_N \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} \approx \begin{pmatrix} \log(y_1) \\ \log(y_2) \\ \log(y_3) \\ \dots \\ \log(y_N) \end{pmatrix} \qquad \Big(\ H \ \Big) \begin{pmatrix} a \\ b \end{pmatrix} \approx \Big(\ v \ \Big)$$

$$\left.\begin{array}{l} A = e^a \\ \beta = b \end{array}\right\} \Rightarrow u(t) = Ae^{\beta t}$$

Problema 2

Ajustar la tabla

t_k	0	1	2	3
y_k	4.5	2.4	1.5	1

con $u(t) = Ae^{\beta t}$

a) Modificar problema convirtiéndolo en un ajuste a una recta.

$$y_k \approx u(t_k) = Ae^{\beta t_k} \Rightarrow \log(y_k) \approx \log(A) + \beta \cdot t_k = a + bt_k$$

El nuevo problema es ajustar con una recta (a+bt) la tabla:

t_k	0	1	2	3
$\log(y_k)$	1.5041	0.8755	0.4055	0.000

Tras resolver (con MATLAB) convertir los parámetros de la recta (a,b) a los parámetros originales:

a = log(A) → A = exp(a)

b = β → β = b

Problema 2

Determinar el sistema sobredeterminado a resolver.

$$\begin{pmatrix} 1 & 0 \\ 1 & 1 \\ 1 & 2 \\ 1 & 3 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} \approx \log \begin{pmatrix} 4.5 \\ 2.4 \\ 1.5 \\ 1 \end{pmatrix} = \begin{pmatrix} 1.5041 \\ 0.8755 \\ 0.4055 \\ 0.0000 \end{pmatrix} = \mathrm{v}$$

Plantear el sistema de ecuaciones normales:

$$\mathrm{H^T H} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & 1 \\ 1 & 2 \\ 1 & 3 \end{pmatrix} = \begin{pmatrix} 4 & 6 \\ 6 & 14 \end{pmatrix} \qquad \mathrm{H^T v} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 \end{pmatrix} \begin{pmatrix} 1.5041 \\ 0.8755 \\ 0.4055 \\ 0.0000 \end{pmatrix}$$

Solución $\bar{\mathrm{c}} = \left(\mathrm{H^T H}\right)^{-1} \mathrm{H} \cdot \bar{\mathrm{v}}$

Solución del Problema en MATLAB

```
% DATOS TABLA (VECTORES COLUMNA)
tk = [0 1 2 3]'        % abcisas {tk}
yk = [4.5 2.4 1.5 1]’ % ordenadas {yk}

H = [tk.^0 tk];    % MATRIZ H [1’s t’s]
v = log(yk);       % VECTOR v del problema lineal “equivalente”

% OPCION A (Comando \ de MATLAB)
c = H\v

% OPCION B (Sol a través de las ecuaciones normales)
Q = H'*H; vv = H'*v; % Matriz y vector de las ecs normales
c=inv(Q)*(vv),    % Solución ecs normales

a=c(1); b=c(2);
A = exp(a), beta=b,  % Parámetros iniciales del problema
```

Con ambas opciones → mismo resultado c = [1.4436 -0.4982]

Y por lo tanto: A = `exp(1.4436) = 4.2359`, `beta = -0.4982`

Ventajas / Inconvenientes de este enfoque

VENTAJA: hemos encontrado una solución

DESVENTAJA: en realidad es la solución de OTRO problema !!

Estoy minimizando $E = \sum_k \left(\log(y_k) - \alpha - \beta t_k\right)^2$,

no es lo que pedían en el problema original: $E = \sum_k \left(y_k - Ae^{\beta t_k}\right)^2$

¿Cómo es que lo aceptamos?

a) Mejor resolver algo similar de una forma fácil que quedarnos sin alcanzar ninguna solución porque no sabemos hacer nada.

b) El criterio de mínimos cuadrados ya era un poco arbitrario, ¿por qué no modificarlo un poco más?

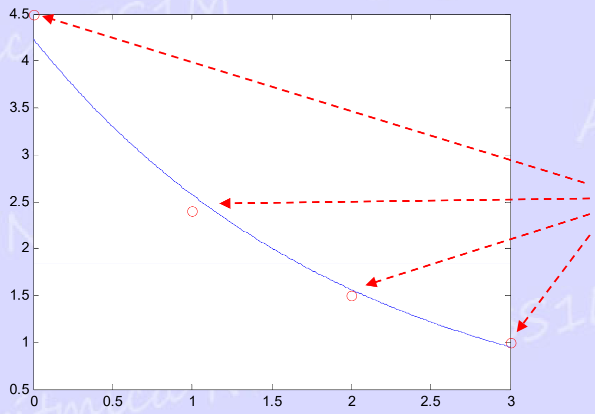
¿Se puede resolver el problema original? **SI, pero es más difícil**

¿Cuáles son las consecuencias de este “compromiso”?

Gráfica de la solución

```
% Gráficos
tk = [0 1 2 3]'        % abcisas {tk}
yk = [4.5 2.4 1.5 1 ]' % ordenadas {yk}

tt=(0:0.01:3);      % malla fina para pintar gráfica
uu=A*exp(beta*tt); % Evalúo función ajuste u(t) en tt para pintar
figure(1); plot(tk,yk,'ro',tt,uu,'b')
```



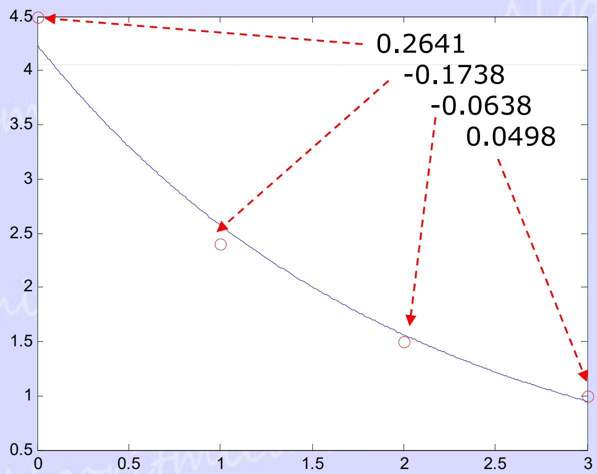
¿No crecen los errores al acercamos al origen?

Residuos de la solución encontrada

Si en el problema resuelto el vector v ≠ {yk} originales el vector de residuos no se puede calcular como res = (v – H*c)

Hay que recurrir a la definición: $e_k = y_k - u(t_k) = \left(y_k - Ae^{\beta t_k}\right)$

```
res=yk-A*exp(beta*tk) % residuos = diferencias entre datos y u(t)
```



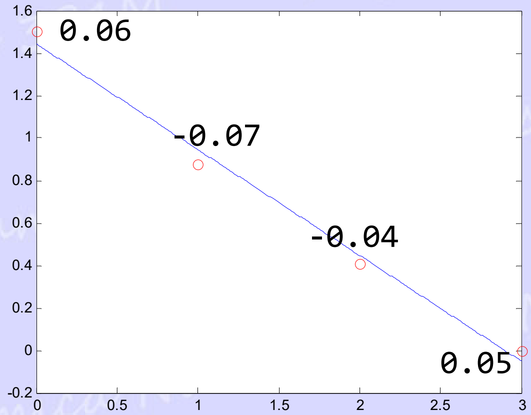
Efectivamente, los residuos (errores) aumentan al acercarnos al origen.

¿De donde viene esta tendencia?

Lo que hacemos y lo que nos gustaría hacer

Ajusto log(datos) a recta

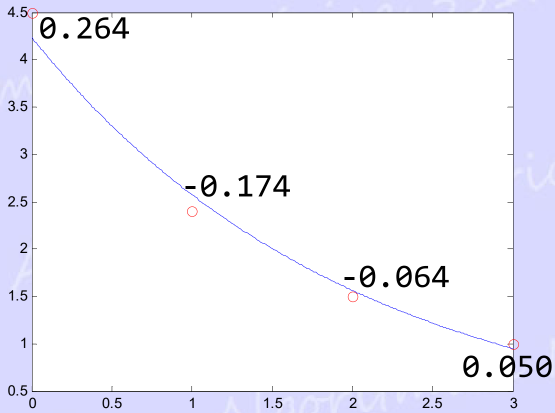
$$\log(datos) \approx recta$$



Esta recta ES EL MEJOR AJUSTE que existe posible (con el criterio de mínimos cuadrados) a los puntos dados por log(datos)

Espero que exp(recta_ajuste) pase cerca de datos originales

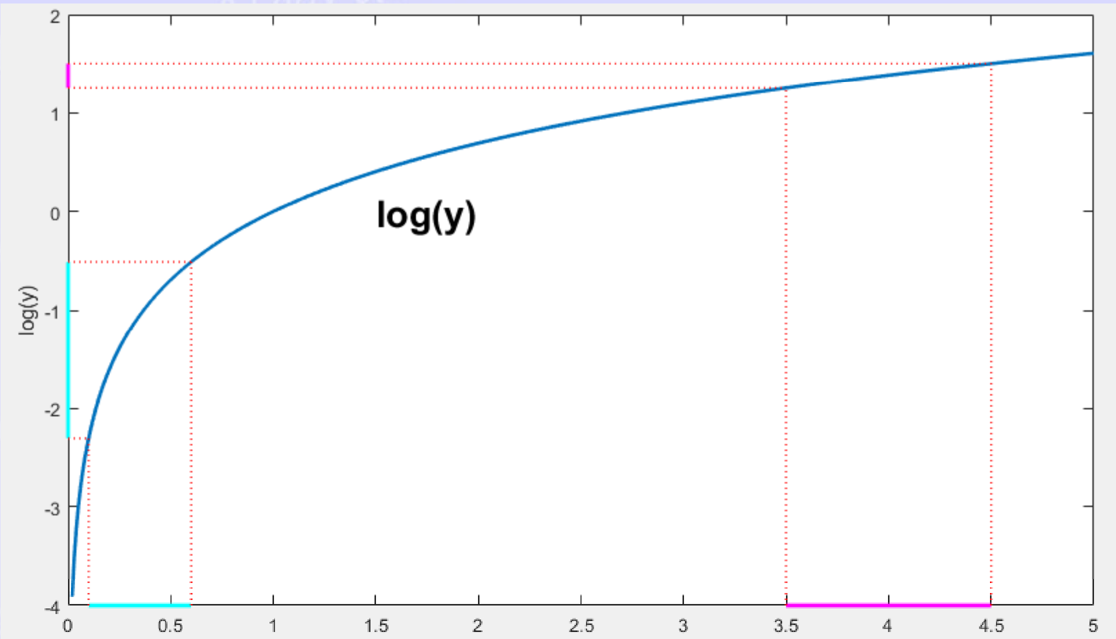
$$datos \approx e^{recta}$$



Este ajuste es RAZONABLE, pero es casi seguro que esta exponencial NO es el mejor ajuste (según mínimos cuadrados) a los datos originales.

El problema original nos pide minimizar: $E = \sum_k \left(y_k - Ae^{\beta t_k}\right)^2$

En nuestra solución minimizamos: $E = \sum_k \left(\log(y_k) - \alpha - \beta t_k\right)^2$



¿Y si queremos resolver bien el problema?

¿Y si quiero minimizar $E = \sum_k \left(y_k - Ae^{\beta t_k}\right)^2$ y no $E = \sum_k \left(\log(y_k) - \alpha - \beta t_k\right)^2$?

Hay que usar algoritmos de optimización que tratan de encontrar el mínimo de una función de "coste".

En nuestro caso la función de coste sería: $f(A,\beta) = \sum_k \left(y_k - Ae^{\beta t_k}\right)^2$

El algoritmo busca en el espacio de los coeficientes y trata de encontrar los valores de (A, β) para los que esa función tiene un mínimo.

Estos algoritmos de optimización suelen ser algoritmos iterativos que necesitan buenas hipótesis iniciales para evitar quedar atrapados en mínimos locales.

La solución aproximada del método anterior sería una buena hipótesis inicial para este tipo de algoritmos.

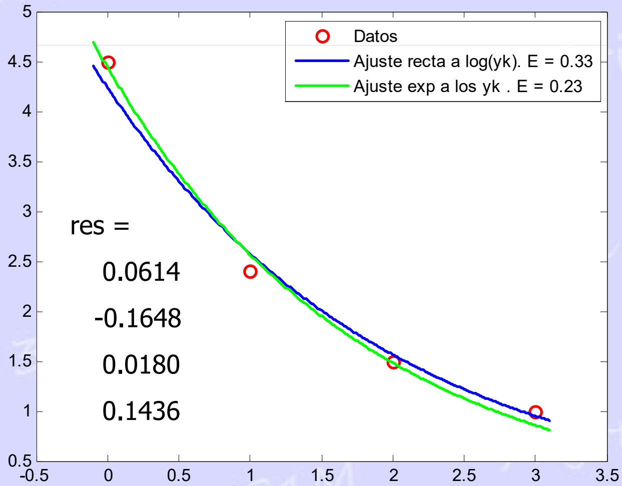
Resolución en MATLAB

Escribo una función objetivo cuya entrada es un vector X (con los parámetros a determinar) y su salida es el valor a minimizar

```
function E=f_obj(X)
  tk = [0 1 2 3]'; yk= [4.5 2.4 1.5 1]';
  A=X(1); b=X(2);
  res = yk - A*exp(b*ti);  % Residuos
  E = norm(res); % norma de residuos
return
```

Llamo a fminsearch, pasándole como parámetros la función a minimizar y una hipótesis inicial para A y b.

```
X0=[4.2359 , -0.4982];
X=fminsearch(@f_obj,X0);
A=X(1); beta=X(2);
res=yi-A*exp(beta*ti);
e2=norm(res);
```



Otro ejemplo: Prob 3b)

Ajustar tabla de datos $\{t_k, y_k\}$ usando funciones $\boxed{u(t) = \dfrac{A}{1 + B\cos(t)}}$

$$y_k \approx \frac{A}{1 + B\cos(t_k)} \Rightarrow \frac{1}{y_k} \approx \frac{1 + B\cos(t_k)}{A} = \alpha \cdot 1 + \beta \cos(t_k)$$

$$\alpha = 1/A$$

$$\beta = B/A$$

$$\begin{pmatrix} 1/y_1 \\ 1/y_2 \\ \dots \\ \dots \\ 1/y_N \end{pmatrix} \approx \begin{pmatrix} 1 & \cos(t_1) \\ 1 & \cos(t_2) \\ 1 & \cos(t_3) \\ \dots & \dots \\ 1 & \cos(t_N) \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix}$$

Tras resolver α y β, volver a los coeficientes originales $\begin{cases} A = 1/\alpha \\ B = A\beta = \beta/\alpha \end{cases}$

Otra forma de resolver el mismo problema

$$y_k \approx \frac{A}{1 + B\cos(t_k)}$$

$$y_k \cdot (1 + B\cos(t_k)) \approx A \Rightarrow y_k \approx A - B \cdot y_k \cdot \cos(t_k)$$

$$\begin{pmatrix} y_1 \\ y_2 \\ \dots \\ \dots \\ y_N \end{pmatrix} \approx \begin{pmatrix} 1 & -y_1 \cdot \cos(t_1) \\ 1 & -y_2 \cdot \cos(t_2) \\ 1 & -y_3 \cdot \cos(t_3) \\ \dots & \dots \\ 1 & -y_N \cos(t_N) \end{pmatrix} \begin{pmatrix} A \\ B \end{pmatrix}$$

Ahora el problema se plantea con los mismos coeficientes A y B originales.

Resultados del Prob 3b) con datos

Ajustar la tabla de datos

t_k	0	1	2	3	4
y_k	0.41	0.40	0.55	0.72	0.62

con funciones del tipo: $u(t) = \dfrac{A}{1 + B\cos(t)}$

Opción A) $\dfrac{1}{y_k} \approx \dfrac{1 + B\cos(t_k)}{A}$

$$\begin{pmatrix} 1/y_1 \\ 1/y_2 \\ \dots \\ \dots \\ 1/y_N \end{pmatrix} \approx \begin{pmatrix} 1 & \cos(t_1) \\ 1 & \cos(t_2) \\ 1 & \cos(t_3) \\ \dots & \dots \\ 1 & \cos(t_N) \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} \qquad \begin{array}{l} A = 1/\alpha \\ B = \beta/\alpha \end{array}$$

Opción B) $y_k \approx A - B \cdot y_k \cdot \cos(t_k)$

$$\begin{pmatrix} y_1 \\ y_2 \\ \dots \\ \dots \\ y_N \end{pmatrix} \approx \begin{pmatrix} 1 & -y_1 \cdot \cos(t_1) \\ 1 & -y_2 \cdot \cos(t_2) \\ 1 & -y_3 \cdot \cos(t_3) \\ \dots & \dots \\ 1 & -y_N \cos(t_N) \end{pmatrix} \begin{pmatrix} A \\ B \end{pmatrix}$$

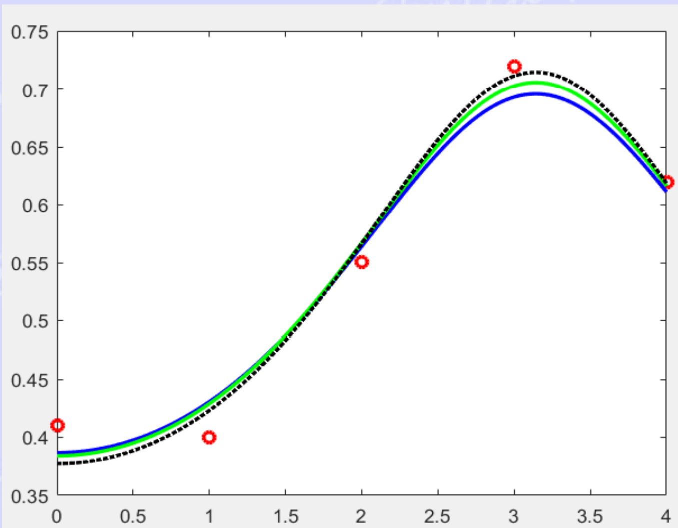
Resultados de este ejemplo con datos

Opción A) A=0.4972 B=0.2854 ||res|| = 0.050

Opción B) A=0.4975 B=0.2951 ||res|| = 0.046

Opcion C) Usar un algoritmo de optimización para minimizar ||res||:

A=0.4938 B=0.3089 ||res|| = 0.044



Problema 3a)

Ajustar una tabla de datos $\{t_k, y_k\}$ usando curvas del tipo: $\boxed{u(t) = Ate^{Bt}}$

$$y_k \approx A \cdot t_k \cdot e^{Bt_k} \Rightarrow \left(\frac{y_k}{t_k}\right) \approx A \cdot e^{Bt_k}$$

$$\log\left(\frac{y_k}{t_k}\right) = \log(A) + Bt_k = \alpha \cdot 1 + \beta \cdot t_k$$

$$\begin{pmatrix} \log(y_1/t_1) \\ \log(y_2/t_2) \\ \dots \\ \dots \\ \log(y_N/t_N) \end{pmatrix} \approx \begin{pmatrix} 1 & t_1 \\ 1 & t_2 \\ 1 & t_3 \\ \dots & \dots \\ 1 & t_N \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} \longrightarrow \begin{array}{l} A = e^{A} \\ B = \beta \end{array}$$

Problema 3d)

Ajustar una tabla de datos $\{t_k, y_k\}$ con $\boxed{u(t) = A\sin(t+\varphi)}$

$$y_k \approx A\sin(t_k + \varphi) = A\cos(\varphi) \cdot \sin(t_k) + A\sin(\varphi) \cdot \cos(t_k)$$

$$y_k \approx a \cdot \sin(t_k) + b \cdot \cos(t_k)$$

$$\begin{pmatrix} y_1 \\ y_2 \\ \dots \\ \dots \\ y_N \end{pmatrix} \approx \begin{pmatrix} & H & \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix}$$

Tras hallar (a, b) → recuperamos los parámetros originales (A, φ):

$$\begin{array}{l} a = A\cos(\varphi) \\ b = A\sin(\varphi) \end{array} \longrightarrow \begin{array}{l} A = \sqrt{a^2 + b^2} \\ \tan(\varphi) = \dfrac{b}{a} \end{array}$$

Otro ejemplo más: problema 3d)

Se considera el problema de ajustar una serie de datos {tk,yk} usando curvas con la siguiente forma:

$$u(t) = \frac{At^2}{B+t^2}$$

- Modificar las expresiones anteriores para convertirlas en ajustes según un modelo lineal en sus coeficientes:

$$\begin{pmatrix} \\ \mathrm{H} \\ \\ \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} \\ \mathrm{v} \\ \\ \end{pmatrix}$$

- Dar la expresión de la matriz H y vector b en cada caso, indicando el posible cambio de parámetros entre (A, B) y (α, β)

Problema 3d)

c) $$y = \frac{At^2}{B+t^2} \Rightarrow \frac{1}{y_k} \approx \frac{B+t_k^2}{At_k^2} = \alpha\left(\frac{1}{t_k^2}\right) + \beta \cdot 1$$

$$\begin{pmatrix} 1/\mathrm{y}_1 \\ 1/\mathrm{y}_2 \\ \ldots \\ \ldots \\ 1/\mathrm{y}_N \end{pmatrix} \approx \begin{pmatrix} 1/\mathrm{t}_1^2 & 1 \\ 1/\mathrm{t}_2^2 & 1 \\ 1/\mathrm{t}_3^2 & 1 \\ \ldots & \ldots \\ 1/\mathrm{t}_N^2 & 1 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix}$$

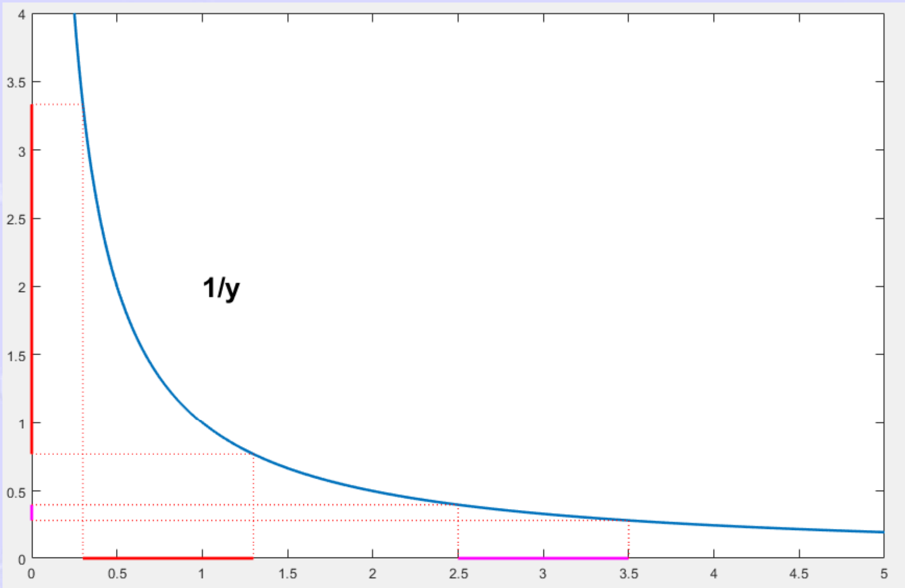
$$\alpha = B/A$$
$$\beta = 1/A$$

Tras resolver, volver a los parámetros iniciales:

$$\mathrm{A} = 1/\beta$$
$$\mathrm{B} = \mathrm{A}\alpha = \alpha/\beta$$

Problema original quería minimizar: $E = \sum_k \left(y_k - \frac{At_k^2}{B + t_k^2} \right)^2$

Con esta solución estamos minimizando: $E = \sum_k \left(\frac{1}{y_k} - \frac{B + t_k^2}{At_k^2} \right)^2$



Problema 3d) de otra forma

$$y = \frac{At^2}{B + t^2} \Rightarrow y_k \cdot \left(B + t_k^2\right) \approx At_k^2 \Rightarrow y_k \cdot t_k^2 \approx At_k^2 - B \cdot y_k$$

Este enfoque tiene la ventaja de que los parámetros sobre los que se resuelve el ajuste son los A y B originales.

No hay que convertir de α y β de vuelta a A y B como antes.

$$\begin{pmatrix} y_1 \cdot t_1^2 \\ y_2 \cdot t_2^2 \\ y_3 \cdot t_3^2 \\ \ldots \\ y_N \cdot t_N^2 \end{pmatrix} \approx \begin{pmatrix} t_1^2 & -y_1 \\ t_2^2 & -y_2 \\ t_3^2 & -y_3 \\ \ldots & \ldots \\ t_N^2 & -y_N \end{pmatrix} \begin{pmatrix} A \\ B \end{pmatrix}$$

Como siempre recordad que ambos “métodos” van a dar soluciones distintas (ya que cada uno está minimizando errores distintos).

<u>Probablemente ninguno de ellos sea la solución del problema original.</u>