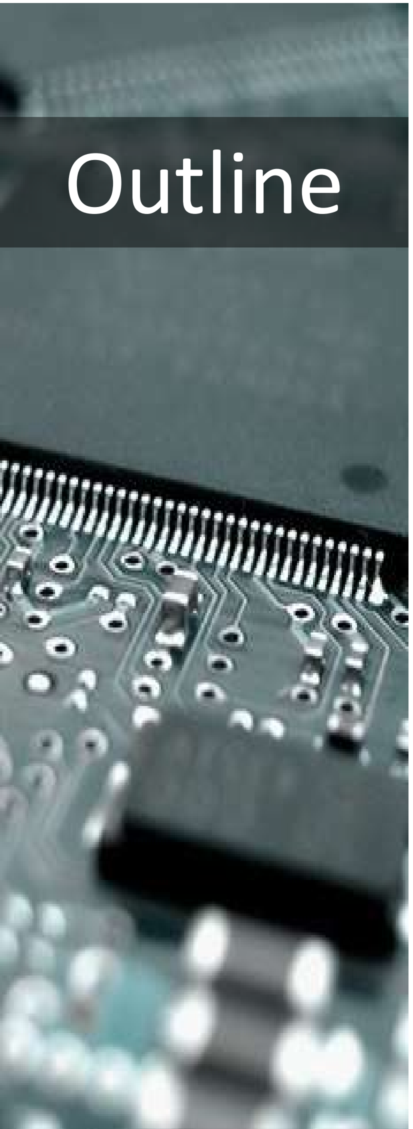
Extremum Seeking

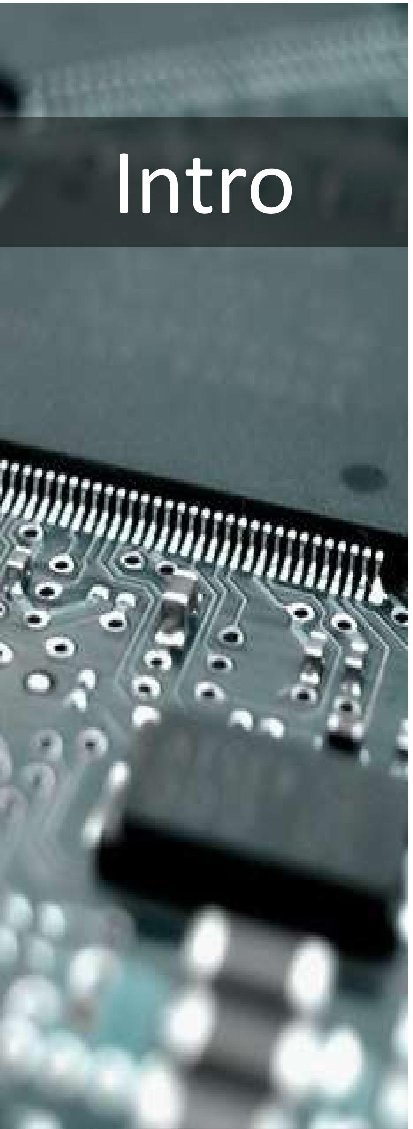
Corso di Tecnologie dei Sistemi di Automazione e Controllo
Laurea Triennale in Ing. Dell'Automazione
28/03/2018

adriano.mele@unina.it



Outline

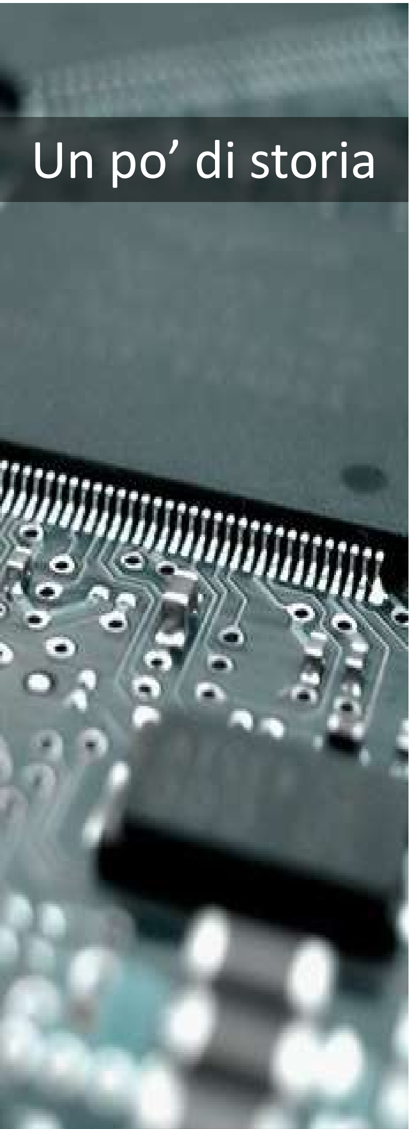
- Cos’è ES
- Un po’ di storia
- Come funziona ES
- PID Tuning con ES
- Esempi in Matlab



Cos’è Extremum Seeking?

Extremum Seeking è un **algoritmo di ottimizzazione**

- Adotta una logica a **ciclo chiuso** per minimizzare o massimizzare una funzione di costo che dipende da uno o più parametri
- Può essere utilizzato in **real-time**
- **Non necessita** della conoscenza della funzione di costo



Un po’ di storia

- **Prima apparizione**: M. Leblanc, *“Sur l’electrification des chemins de fer au moyen de courants alternatifs de frequence elevee”,* Revue Generale de l’Electricite, 1922

- Usato sia in Russia che nell’Occidente già negli anni ‘50-’60

- È stata fornita una **prova della stabilità** di ES, pubblicata nel 2000 (M. Krstic and H. H. Wang, *“Design and stability analysis of extremum seeking feedback for general nonlinear systems”,* Automatica, 2000)

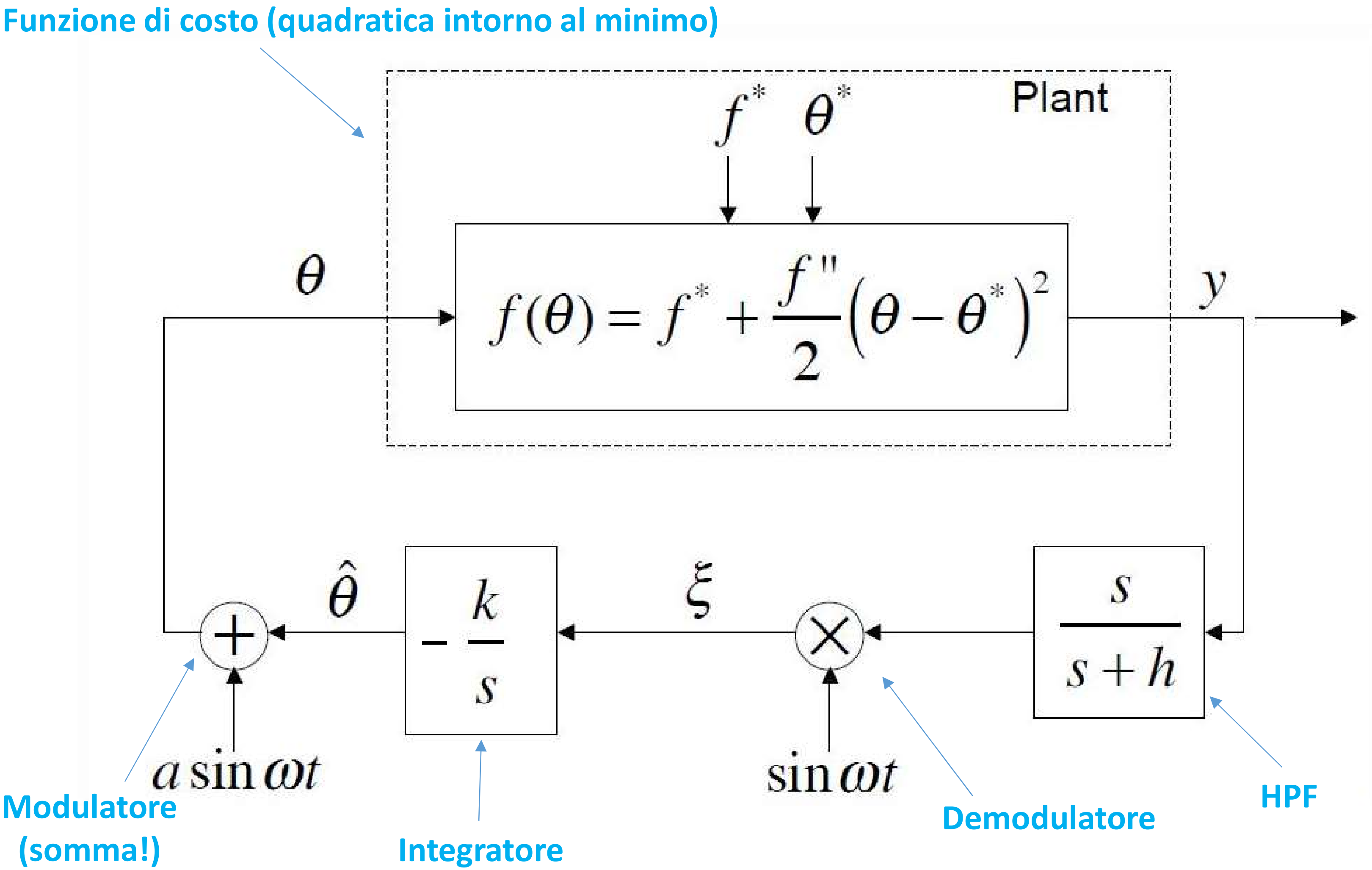
- Da quando la potenza di calcolo lo consente, viene utilizzato anche in applicazioni **real-time**

Come funziona Extremum Seeking?

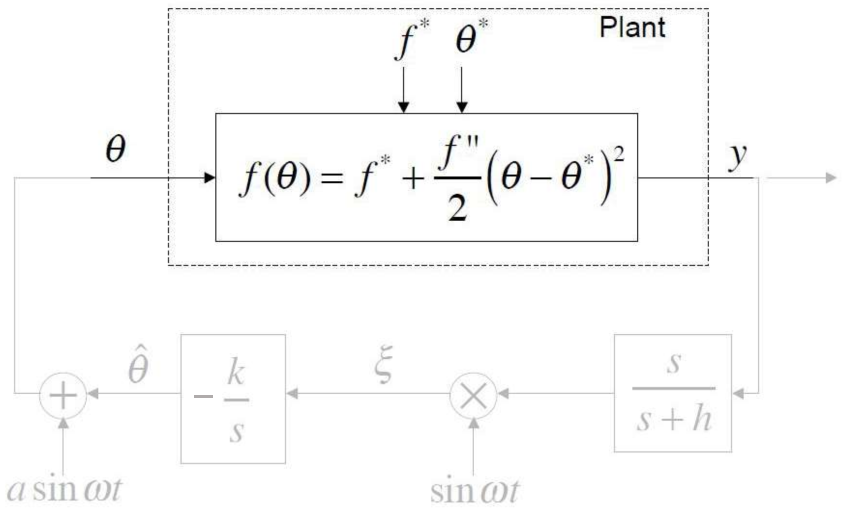
- L'obiettivo di ES è trovare il valore dei parametri θ^* per il quale la funzione di costo $f(\theta)$ ha un **minimo locale**

- ES cerca di stimare il **gradiente** della funzione di costo sfruttando la modulazione con un **segnale test sinusoidale**

Principio di Funzionamento



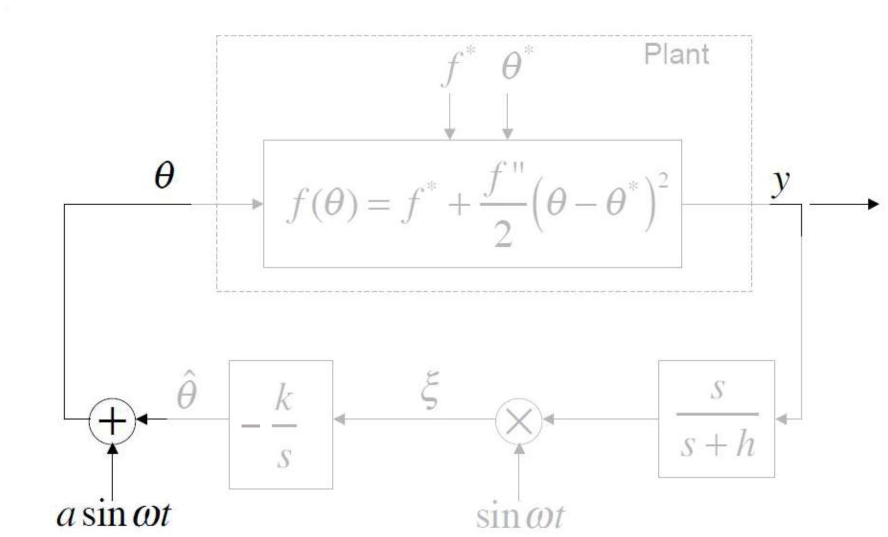
Principio di Funzionamento

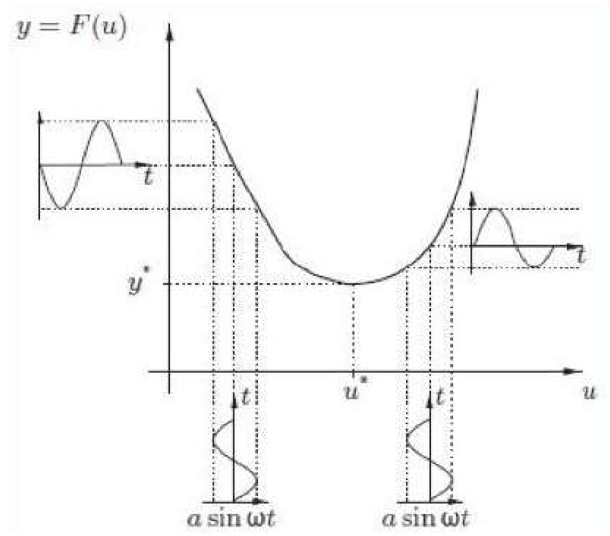


- Supporremo che la funzione di costo abbia una struttura **quadratica** (possiamo immaginare di espanderla in serie di Taylor intorno al punto di minimo arrestandoci al secondo ordine – f' è nulla per il teorema di Fermat)

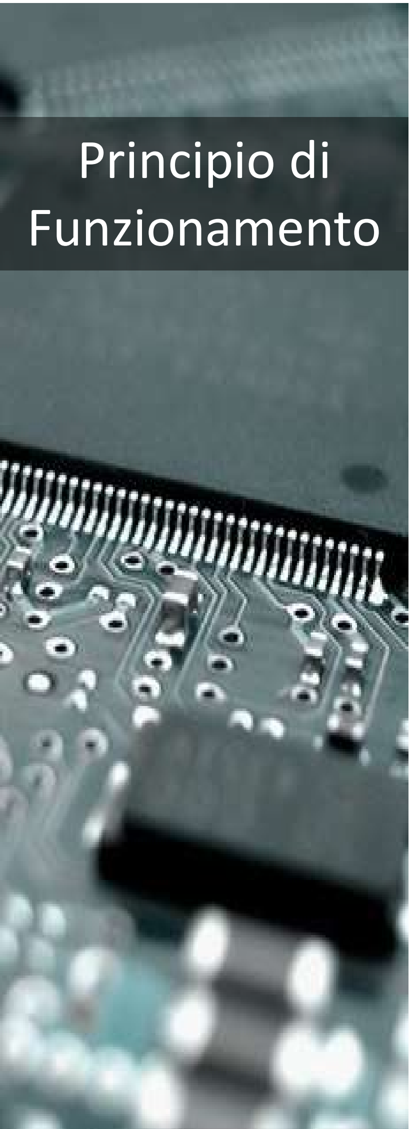
- Supporremo che f'' sia **positiva** (in questo modo, f avrà un **minimo** in θ^*)

Principio di Funzionamento

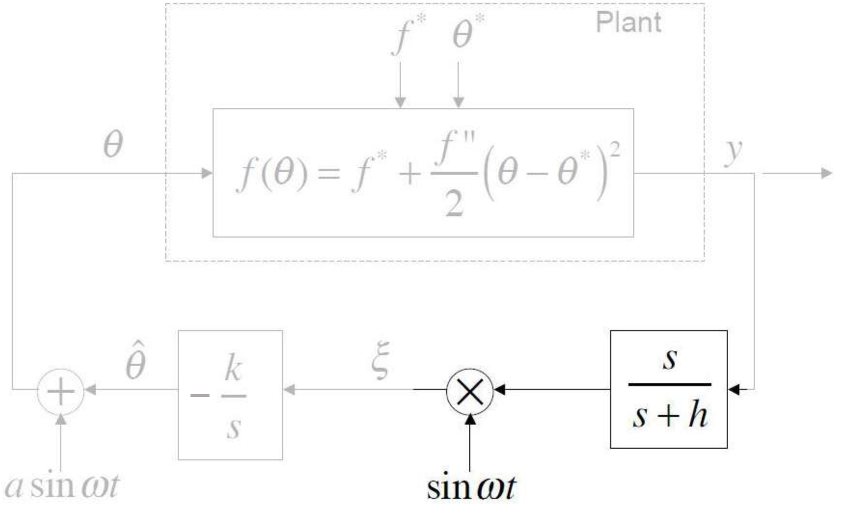




- Una **perturbazione sinusoidale** entra in maniera **additiva** in $f(\theta)$
- L’uscita sarà **approssimativamente sinusoidale**
- Si usa un blocco somma e non prodotto poiché la parte quadratica di f restituisce un termine moltiplicativo (nel doppio prodotto)



Principio di Funzionamento

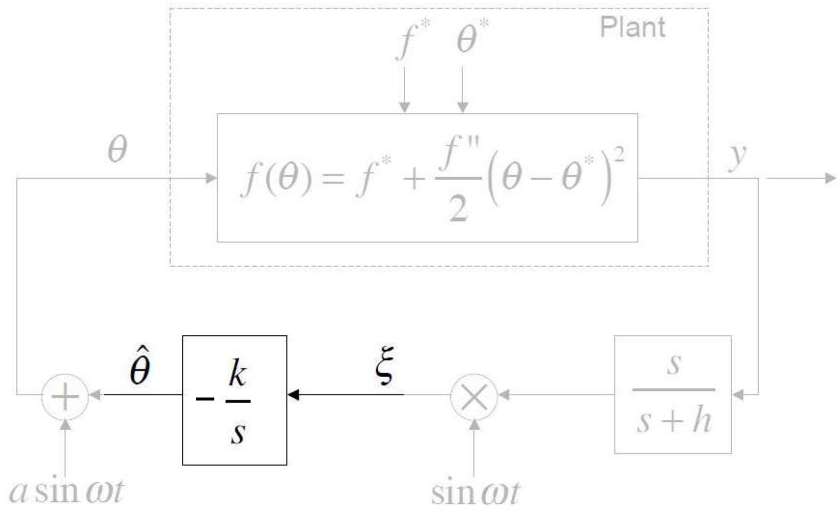


- L'uscita, all'incirca sinusoidale, viene data in ingresso a un **filtro passa-alto**

- L'uscita del filtro viene quindi **demodulata** moltiplicandola nuovamente per una sinusoide a pulsazione ω

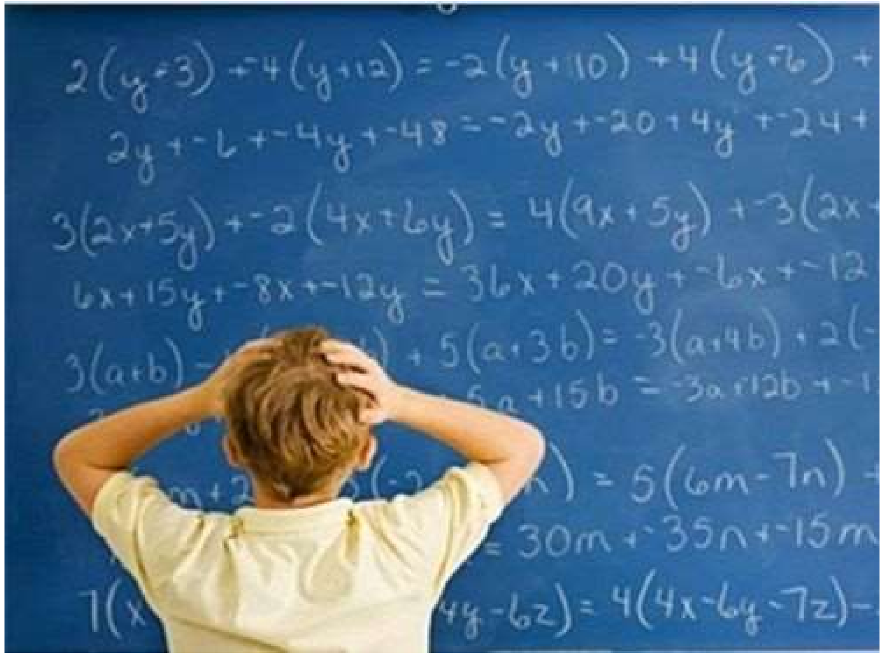
- Si noti che il filtro passa-alto **non è strettamente necessario**, ma migliora le performance eliminando la componente continua di y

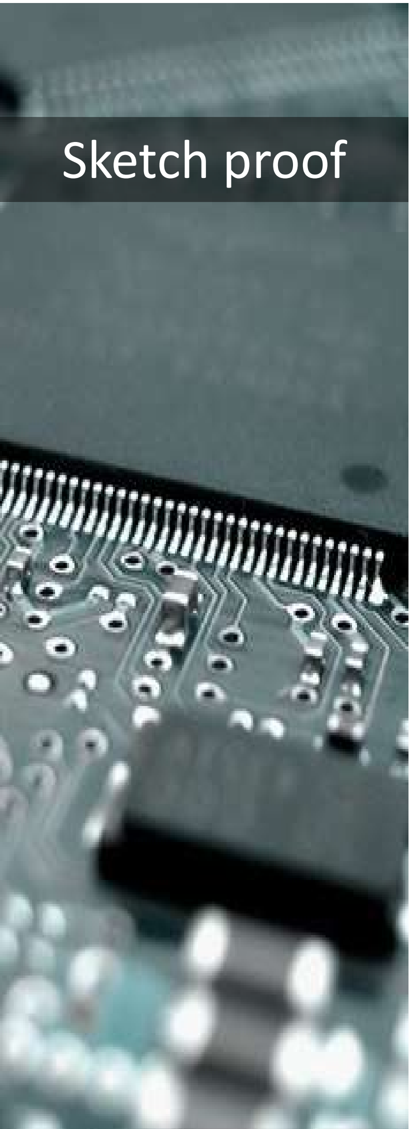
Principio di Funzionamento



- Il segnale viene quindi dato in ingresso a un **integratore**, che elimina le componenti ad alta frequenza introdotte
- L'uscita dell'integratore rappresenta la **stima** dei parametri
- A seconda del segno di k, l'algoritmo convergerà verso un **massimo** (k<0) o un **minimo** (k>0) locale

Sketch proof





Sketch proof

$\tilde{\theta} \triangleq \theta^* - \hat{\theta}$, scalare

$$
\begin{aligned}
y = f(\theta) &= f^* + \frac{f''}{2}\left(\theta^* - \hat{\theta} - a\,\sin\omega t\right)^2 = \\
&= f^* + \frac{f''}{2}\left(\tilde{\theta} - a\sin\omega t\right)^2 = \\
&= f^* + \cancel{\frac{f''}{2}}\tilde{\theta}^2 + \frac{a^2 f''}{2}\sin^2\omega t - a f''\tilde{\theta}\sin\omega t = \\
&\cong f^* - \boxed{a f''\tilde{\theta}\sin\omega t} + \frac{a^2 f''}{4}\left(1 - \cos 2\omega t\right)
\end{aligned}
$$

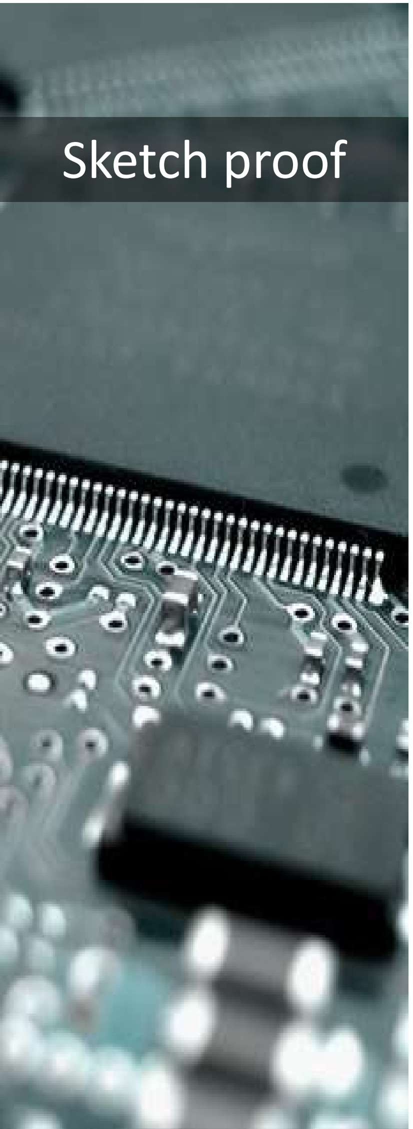
Trascuriamo i termini del secondo ordine (è un'analisi locale)

$\cos 2\omega t = 1 - \ 2\sin^2\omega t$

Compare un termine moltiplicativo tra $\tilde{\theta}$ e $\sin\omega t$

$$
\frac{s}{s+h}[y] \cong -a f''\tilde{\theta}\sin\omega t - \frac{a^2 f''}{4}\cos 2\omega t
$$

Il filtro HPF elimina i termini DC (deve valere h < ω)



Sketch proof

$$\xi \cong -af''\tilde{\theta}\sin^2\omega t - \frac{a^2 f''}{4}\sin\omega t\cos 2\omega t =$$
$$= -\frac{af''\tilde{\theta}}{2}(1-\cos 2\omega t) - \frac{a^2 f''}{8}(\sin 3\omega t - \sin\omega t)$$

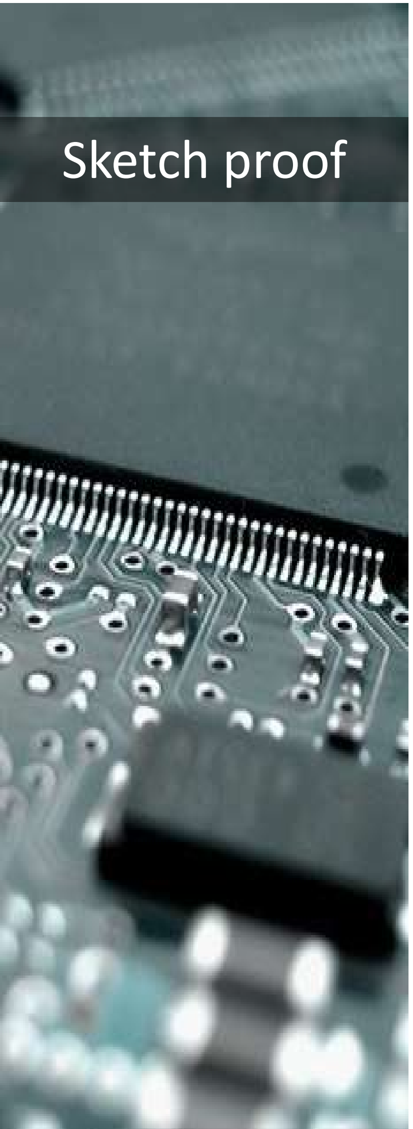
$$\boxed{\begin{aligned}&\cos 2\omega t = 1 - 2\sin^2\omega t\\ &\sin\alpha\cos\beta = \frac{1}{2}[\sin(\alpha+\beta)+\sin(\alpha-\beta)]\end{aligned}}$$

$$\tilde{\theta} \triangleq \theta^* - \hat{\theta} \Rightarrow \dot{\tilde{\theta}} = -\dot{\hat{\theta}}$$

$$\hat{\theta} = -\frac{k}{s}[\xi] \Rightarrow k\xi = -\dot{\hat{\theta}} = \dot{\tilde{\theta}}$$

$$\tilde{\theta} = \frac{k}{s}[\xi] = \frac{\mathrm{k}}{\mathrm{s}}\left[-\frac{af''\tilde{\theta}}{2}(1-\cancel{\cos 2\omega t}) - \frac{a^2 f''}{8}(\cancel{\sin 3\omega t} - \cancel{\sin\omega t})\right] \cong \frac{1}{s}\left[-\frac{akf''\tilde{\theta}}{2}\right]$$

L’integratore ha un effetto passa-basso
(con una scelta opportuna di k)



Sketch proof

$$\dot{\tilde{\theta}} = -ak\frac{\boldsymbol{f''}}{\boldsymbol{2}}\tilde{\boldsymbol{\theta}}$$

L'errore converge a zero con una dinamica determinata dai parametri *k* e *a (akf'' > 0)*

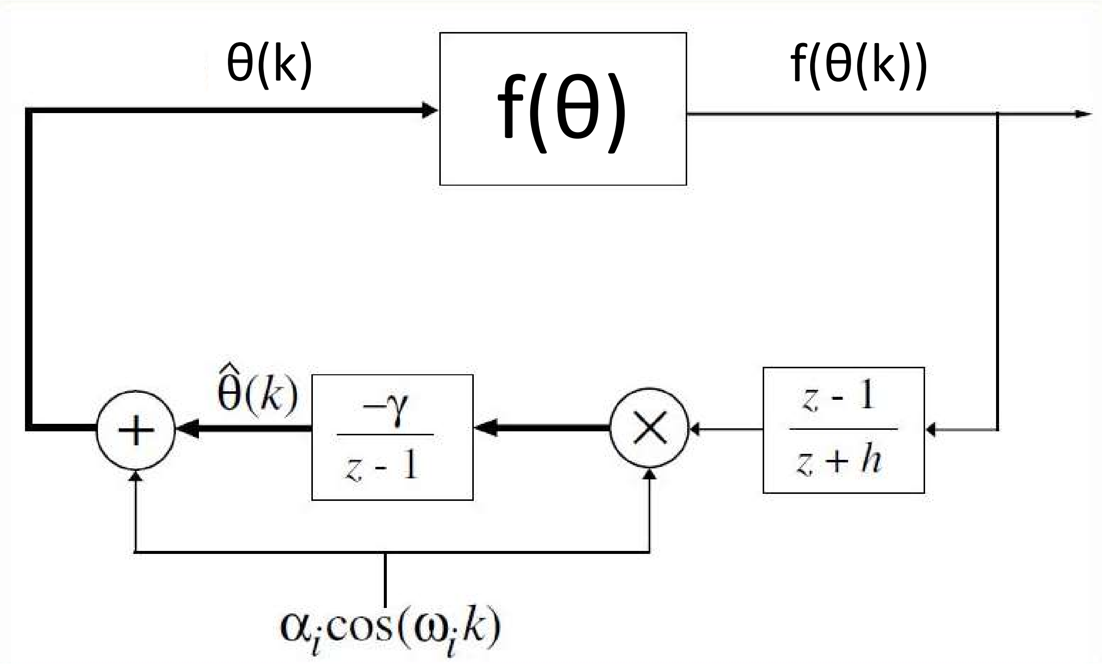
Osservazione: $f''\tilde{\theta}$ è il gradiente di f rispetto a $\tilde{\theta}$ quando $a = 0$

L'informazione sul gradiente è contenuta nel doppio prodotto $af''\tilde{\theta}\sin\omega t$, l'algoritmo punta a estrarre questa informazione sfruttando una tecnica di modulazione/demodulazione

ES è un metodo per stimare il gradiente di *f* **senza conoscere la *f* stessa**.
L'algoritmo di ottimizzazione usato è il classico **metodo del gradiente**!

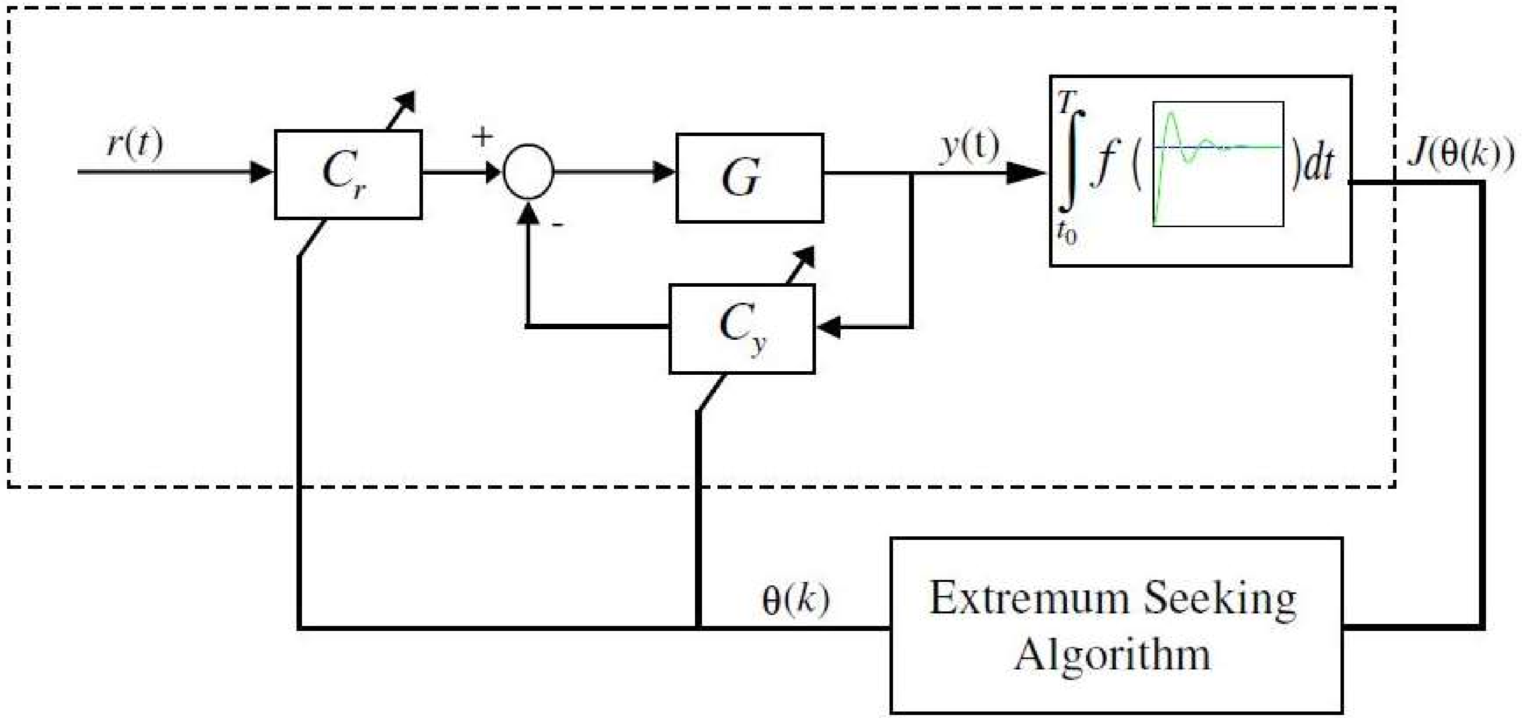
Taratura di un PID con ES

Useremo la versione **tempo-discreta** dell'algoritmo ES



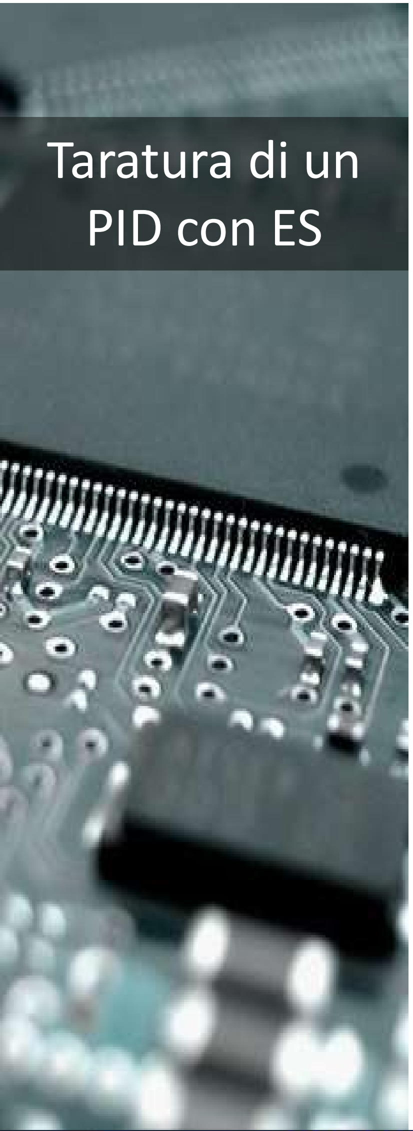
Nel caso scalare, si può mostrare che $\theta(k+1) = \left(1 - \alpha^2 \gamma \frac{f''}{2}\right) \tilde{\theta}(k)$

Taratura di un PID con ES



$$C_r = K_p\left(1 + \frac{1}{sT_I}\right) \qquad C_y = K_p\left(1 + \frac{1}{sT_I} + sT_D\right) \qquad \theta \triangleq [K_p\ T_I\ \ T_D]^T$$

NOTA: l'azione derivativa agisce solo sul segnale di uscita



Taratura di un PID con ES

Possibili funzioni di costo:

ISE $$\boxed{f(\theta) = \frac{1}{T - t_0}\int_{t_0}^{T} e^2(t,\theta)dt}$$

$$f(\theta) = \frac{1}{T - t_0}\int_{t_0}^{T} te^2(t,\theta)dt$$ **ITSE**

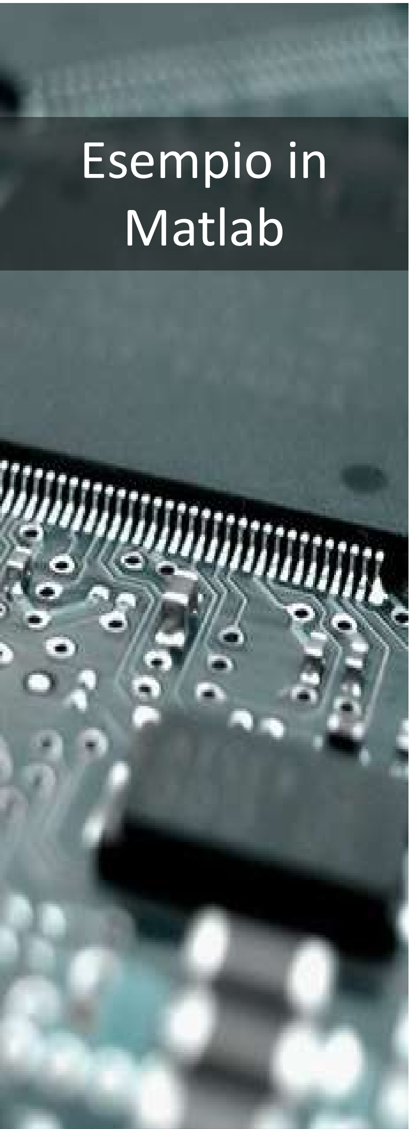
IAE $$f(\theta) = \frac{1}{T - t_0}\int_{t_0}^{T} |e|(t,\theta)dt$$

$$f(\theta) = \frac{1}{T - t_0}\int_{t_0}^{T} t|e|(t,\theta)dt$$ **ITAE**

Condizione iniziale:

$t_0 \cong T_{peak}$ (escludiamo l'errore in transitorio)

θ_0 calcolato con Z&N



Esempio in Matlab