Principali comandi MATLAB utili per il corso di **Controlli Automatici**

In questo documento sono raccolti i principali comandi Matlab utilizzati nel corso; per maggiore comodità, sono riportati facendo riferimento al loro impiego piuttosto che come mero elenco di comandi. Si ricorda che informazioni dettagliate sulla loro sintassi possono essere ottenute semplicemente digitando **`help nome del comando`** nella *Command Window* di Matlab (ad esempio **`help tf`**).

Funzioni di trasferimento e loro proprietà

- Modello ingresso-uscita: definizione "simbolica" della funzione di trasferimento
  - Dopo aver introdotto la variabile complessa s per mezzo del comando **`tf`**, è possibile scrivere direttamente la funzione di trasferimento di un sistema LTI a tempo continuo in modo simbolico.

    Esempio: La funzione

    $$G(s) = \frac{1+s}{3s^2+5s+2}$$

    viene definita in Matlab dai seguenti comandi:

    ```
    >> s=tf('s')
    >> G=(1+s)/(3*s^2+5*s+2)
    ```
  - È possibile analogamente scrivere la funzione di trasferimento di un sistema LTI a tempo discreto, dopo avere introdotto la variabile complessa z ed aver contestualmente definito il passo di campionamento Ts per mezzo del medesimo comando **`tf`**:

    ```
    >> z=tf('z',Ts)
    ```

- Calcolo della fdt ad anello chiuso di un sistema in retroazione
  - Il comando **`feedback`** genera il sistema retroazionato negativamente (rappresentato dalla sua fdt W), avente F1 come fdt del ramo diretto e F2 come fdt del ramo in retroazione, secondo la seguente sintassi:

    ```
    >> W=feedback(F1,F2)
    ```
    - Per applicare retroazione positiva, è sufficiente dichiarare +1 come terzo argomento del comando:

      ```
      >> W=feedback(F1,F2,+1)
      ```

    N.B.: Il comando **`feedback`** può essere utilizzato per sistemi sia a tempo continuo sia a tempo discreto.

- Calcolo del guadagno stazionario
  - Il comando **`dcgain`** applicato al sistema LTI G ne calcola il "guadagno in continua", cioè il valore della sua fdt G(s) per s = 0:

    ```
    >> K0=dcgain(G)
    ```

- Tale comando può essere utilizzato per calcolare il *guadagno stazionario di* G(s) *di tipo h*, definito come:

$$K_G = \lim_{s \to 0} \left\{ s^h \cdot G(s) \right\}$$

tenendo conto del numero *h* di poli in s = 0 di G(s):

```
>> KG=dcgain(s^h*G)
```

- Calcolo delle singolarità (poli e zeri) di una fdt
  - Il comando **zero** applicato al sistema LTI G ne calcola gli zeri di trasmissione, cioè gli zeri della sua fdt G(s); i comandi **pole** e **damp** ne calcolano i poli, indicandone nel primo caso parte reale e parte immaginaria, nel secondo caso anche i corrispondenti valori di pulsazione naturale e fattore di smorzamento:

```
>> zero(G)
>> pole(G)
>> damp(G)
```

Tracciamento di diagrammi ed analisi della stabilità

Tutti i comandi sotto elencati possono essere applicati a sistemi sia a tempo continuo sia a tempo discreto.

- Diagrammi di Bode
  - Il comando **bode** applicato al sistema LTI G (senza ulteriori argomenti) ne traccia i diagrammi di Bode, scegliendo automaticamente l'intervallo di pulsazioni di interesse ed il numero di punti:

```
>> bode(G)
```

  - Per tracciare i diagrammi di Bode in un intervallo di pulsazioni prescelto, è possibile indicare direttamente i valori minimo e massimo di tale intervallo:

```
>> bode(G,{wmin,wmax})
```

oppure definire (precedentemente o direttamente all'interno del comando **bode**) il corrispondente vettore di pulsazioni generato dal comando **logspace**:

```
>> w=logspace(x1,x2,N)
>> bode(G,w)
```

ove il vettore `w` risulta definito da N punti spaziati in scala logaritmica di valore compreso fra 10^{x1} e 10^{x2}.

  - Il comando **bode** può essere utilizzato anche per calcolare (senza rappresentazione grafica) modulo e fase di una funzione complessa ad una pulsazione `w` o in un intervallo di pulsazioni, definito dal vettore `w` precedentemente generato con **logspace**:

```
>> [m,f]=bode(G,w);
```

I vettori `m` e `f` contengono rispettivamente il modulo (in unità naturali) e la fase (in gradi) di G(s) nell'intervallo di pulsazioni specificato. Per ottenere il modulo in decibel è sufficiente applicare il comando **log10**:

```
>> m_dB=20*log10(m);
```

- Diagramma di Nyquist
  - Il comando **nyquist** applicato al sistema LTI G (senza ulteriori argomenti) ne traccia il diagramma di Nyquist, scegliendo automaticamente l'intervallo di pulsazioni di interesse ed il numero di punti:

    ```
    >> nyquist(G)
    ```

    N.B.: Il diagramma di Nyquist tracciato può risultare incompleto, perché mancante delle semicirconferenze all'infinito associate ad eventuali poli di G(s) posti sull'asse immaginario. I dettagli del diagramma possono essere analizzati con successive operazioni di zoom. Si consiglia la successiva applicazione del comando **axis equal** per garantire la medesima scala su entrambi gli assi cartesiani:

    ```
    >> axis equal
    ```
  - Per tracciare il diagramma di Nyquist in un intervallo di pulsazioni prescelto, è possibile indicare direttamente i valori minimo e massimo di tale intervallo oppure definire il corrispondente vettore di pulsazioni, analogamente a quanto visto per il comando **bode**.

- Diagramma di Nichols
  - Il comando **nichols** applicato al sistema LTI G (senza ulteriori argomenti) ne traccia il diagramma di Nichols scegliendo automaticamente l'intervallo di pulsazioni di interesse ed il numero di punti:

    ```
    >> nichols(G)
    ```
  - Per tracciare il diagramma di Nichols in un intervallo di pulsazioni prescelto, è possibile indicare direttamente i valori minimo e massimo di tale intervallo oppure definire il corrispondente vettore di pulsazioni, analogamente a quanto visto per il comando **bode**.
  - Per tracciare la carta di Nichols, sovrapponendola al diagramma di Nichols del sistema, è sufficiente aggiungere la griglia direttamente sulla finestra grafica (selezionando "Grid" nel menu a tendina, apribile cliccando con il tasto destro del mouse in qualunque punto dell'area del grafico) oppure applicare il comando **ngrid**:

    ```
    >> ngrid
    ```

- Margini di stabilità
  - Il comando **margin** applicato alla funzione d'anello $G_a(s)$ (rappresentata in Matlab dal sistema LTI Ga) ne traccia i diagrammi di Bode, mettendone in evidenza i margini di stabilità:

    ```
    >> margin(Ga)
    ```

    I valori dei margini e delle pulsazioni alle quali sono letti sono riportati sopra il diagramma del modulo.

N.B.: L'informazione sull'effettiva stabilità del sistema ad anello chiuso è sicuramente corretta solo per i sistemi per i quali è possibile dedurre la stabilità dalla sola lettura dei margini sui diagrammi di Bode della funzione d'anello, senza la necessità di svolgere un'analisi completa della stabilità con il criterio di Nyquist.

- I valori del margine di guadagno m_G (Gm) e della pulsazione ω_π (Wcg) alla quale viene calcolato ed i valori del margine di fase m_φ (Pm) e della pulsazione di cross-over ω_c (Wcp) possono essere salvati in quattro corrispondenti variabili applicando il comando margin con la seguente sintassi:

```
>> [Gm,Pm,Wcg,Wcp]=margin(Ga)
```

Risposta di sistemi LTI ad ingressi canonici

- Risposta al gradino (per sistemi a tempo continuo e a tempo discreto)
  - Il comando **step** applicato al sistema LTI G (senza ulteriori argomenti) ne simula la risposta al gradino unitario, scegliendo automaticamente il numero di istanti temporali e la durata della simulazione; l'andamento temporale della risposta viene riportato su un apposito grafico:

    ```
    >> step(G)
    ```
  - È possibile altresì specificare la durata della simulazione da t = 0 a t = TFINAL (con scelta automatica degli istanti temporali intermedi):

    ```
    >> step(G,TFINAL)
    ```

    oppure assegnare l'intero vettore del tempo T per la simulazione:

    ```
    >> step(G,T)
    ```
  - L'evoluzione dell'uscita del sistema ed il vettore dei tempi della simulazione possono essere salvati nelle corrispondenti variabili Y e T applicando il comando **step** con la seguente sintassi:

    ```
    >> [Y,T]=step(G);
    ```

    In questo caso non viene prodotta alcuna uscita grafica. Per visualizzare l'andamento dell'uscita è necessario applicare il comando **plot**:

    ```
    >> plot(T,Y)
    ```

Sistemi digitali

- Discretizzazione di un sistema LTI a tempo continuo
  - Il comando **c2d** converte il sistema LTI a tempo continuo G in un sistema a tempo discreto GD, con tempo di campionamento TS, secondo il metodo specificato come terzo argomento:

    ```
    >> GD=c2d(G,TS,METHOD)
    ```

    `METHOD` è una stringa corrispondente al metodo prescelto fra quelli disponibili:
    - `'zoh'`: discretizzazione con inserimento di un filtro di tenuta di ordine zero
    - `'foh'`: discretizzazione con inserimento di un filtro di tenuta del primo ordine
    - `'tustin'`: approssimazione bilineare

- `'prewarp'`: approssimazione bilineare con pre-compensazione alla pulsazione ω_c specificata come quarto argomento:
  ```
  >> GD=c2d(G,TS,'prewarp',wc)
  ```
- `'matched'`: discretizzazione secondo la corrispondenza zeri-poli