Elementi di Informatica e Applicazioni Numeriche T

Equazioni Non Lineari in Matlab

Equazioni Non Lineari in Matlab

Per risolvere equazioni non lineari, Matlab offre due funzioni:

```
function [X, FVAL, FLAG] = fzero(F, X0)
function [X, FVAL, FLAG] = fsolve(F, X0)
```

Entrambe risolvono equazioni nella forma:

$$f(x) = 0$$

- In altre parole, cercano di trovare un valore di x...
- ...Che azzeri la funzione $f(x)$

Una soluzione dell'equazione è detta uno zero della funzione

Equazioni Non Lineari in Matlab

Per risolvere equazioni non lineari Matlab offre due funzioni:

```
function [X, FVAL, FLAG] = fzero(F, X0)
function [X, FVAL, FLAG] = fsolve(F, X0)
```

Per quanto riguarda i parametri, abbiamo che:

- `X0` è la stima iniziale di x da cui partire
- `F` è la funzione da azzerare (passata come parametro)
- `F` deve avere un singolo parametro, i.e.:

```
function z = F(X)
```

- Il parametro (i.e `X`) corrisponde alla variabile x

Equazioni Non Lineari in Matlab

Per risolvere equazioni non lineari Matlab offre due funzioni:

```
function [X, FVAL, FLAG] = fzero(F, X0)
function [X, FVAL, FLAG] = fsolve(F, X0)
```

Per quanto riguarda i valori restituiti abbiamo che:

- **`X`** è l'ultimo valore di x visitato
  - Se tutto è andato bene, è la soluzione
- **`FVAL`** è il valore di **`F`** in corrispondenza di **`X`**
  - Se tutto è andato bene, sarà molto vicino a 0
- **`FLAG`** è un numero che indica come sono andate le cose
  - Se vale `1`, l'algoritmo ha trovato uno zero
  - Altrimenti, qualcosa è andato storto (consultate **`help`**)

Equazioni Non Lineari in Matlab

Per risolvere equazioni non lineari Matlab offre due funzioni:

```
function [X, FVAL, FLAG] = fzero(F, X0)
function [X, FVAL, FLAG] = fsolve(F, X0)
```

Le due funzioni usano una combinazione di metodi numerici

- **`fzero`** è basata (grossomodo) sul metodo della bisezione
  - Cerca di trovare un secondo punto **`x1`**...
  - ...Tale che **`F(X0)`** e **`F(X1)`** abbiamo segni opposti
  - Se ce la fa e se **`F`** è continua, allora converge sempre
- **`fsolve`** usa metodi completamente diversi (i.e. trust region)
  - Non è garantito che converga...
  - ...Ma può essere più veloce di **`fzero`**

Equazioni Non Lineari in Matlab

Per risolvere equazioni non lineari Matlab offre due funzioni:

```
function [X, FVAL, FLAG] = fzero(F, X0)
function [X, FVAL, FLAG] = fsolve(F, X0)
```

Il valore di **`x0`** è molto importante:

- Se è scelto male, può impedire la convergenza
  - Nel caso di **`fzero`**, perché non si riesce a trovare un **`x1`** adeguato
  - Nel caso di **`fsolve`**, per i limiti del metodo stesso
- Se ci sono più soluzioni, può determinare quale sia restituita
  - Il più delle volte, sarà la soluzone più vicina ad **`x0`**…
  - …Ma in generale non è garantito

Di solito, il valore di **`x0`** si sceglie per intuizione o per tentativi

Sistemi di Equazioni Non Lineari

Per risolvere un sistema di equazioni non lineari...

...Cerchiamo lo zero di una funzione vettoriale, ossia risolviamo:

$$F(x) = 0 \qquad \text{con } F : \mathbb{R}^n \mapsto \mathbb{R}^m$$

- Dove n ed m sono il numero di variabili e di equazioni.
- Per esempio:

$$\begin{aligned} x^3 &= e^y \\ \log x &= y + 1 \end{aligned} \longleftrightarrow F((x, y)) = (0, 0)$$

- Con:

$$F((x, y)) = \left(x^3 - e^y, \quad \log x - y - 1 \right)$$

Sistemi di Equazioni Non Lineari

Per i sistemi di equazioni non lineari...

...Possiamo usare solo **`fsolve`**

```
function [X, FVAL, FLAG] = fsolve(F, X0)
```

In questo caso:

- La funzione **`F`** dovrà avere ancora un singolo parametro:

```
function Z = F(X)
```

- Questa volta, però, **`X`** sarà un vettore anziché uno scalare
- E lo stesso vale per il valore di ritorno **`Z`**

`fsolve` cercherà un vettore **`X`** per cui **`F(X)`** restituisca un vettore nullo

Elementi di Informatica e Applicazioni Numeriche T

Esempio: Stati di Equilibrio
di Sistemi Dinamici Non Lineari

Stati di Equilibrio di Sistemi Non Lineari

Vi ricordate il modello di Shepherd?

$$x^{(k+1)} = \frac{rx^{(k)}}{1 + \left(\frac{x^{(k)}}{N}\right)^2}$$

È un modello tempo-discreto per l'evoluzione di una popolazione:

- $x^{(k)}$ è il numero di individui al passo k-mo
- r è un tasso di crescita
- N è il valore di popolazione per il cui r dimezza

Abbiamo visto come determinare lo stato stabile per simulazione

E se volessimo determinarlo senza simulare?

Stati di Equilibrio di Sistemi Non Lineari

Uno stato di equilibrio deve soddisfare:

$$x = f(x)$$

- Dove f è la funzione di transizione, che è in questo caso non lineare

Portiamo tutti i termini a sx del segno "="

$$x - f(x) = 0$$

- L'espressione $x - f(x)$ è a sua volta una funzione...
- ...Di cui possiamo determinare uno zero con `fzero` o `fsolve`

Vediamo come farlo in pratica...

Una Possibile Soluzione

Supponiamo di avere:

```
function es_shepherd_eq()
    % Dati del problema
    r = 1.72;
    k = 1000;
end

function xf = f(xc, r, k)
    xf = (r .* xc) ./ (1 + (xc./k).^2);
end
```

- A noi non interessa azzerare la funzione di transizione `f`...
- ...Ma l'espressione `x - f(x, r, k)`

Una Possibile Soluzione

Supponiamo di avere:

```
function es_shepherd_eq()
    % Dati del problema
    r = 1.72;
    k = 1000;
    feq = @(x) (x - f(x, r, k)); % r e k dall'ambiente
end

function xf = f(xc, r, k)
    xf = (r .* xc) ./ (1 + (xc./k).^2);
end
```

- Possiamo definire una nuova funzione anonima **`feq`**...
- ...Che denoti il valore di **`x - f(x, r, k)`**

Una Possibile Soluzione

Per il modello di Shepherd, supponiamo di avere:

```
function es_shepherd_eq()
    % Dati del problema
    r = 1.72;
    k = 1000;
    feq = @(x) (x - f(x, r, k));
    [xeq, fval, flag] = fzero(feq, x0)
end

function xf = f(xc, r, k)
    xf = (r .* xc) ./ (1 + (xc./k).^2);
end
```

- **`feq`** espone un solo parametro ⇒ è compatibile con (e.g.) **`fzero`**
- Quindi usiamo **`fzero`** per trovare il valore **`xeq`** che azzera ***`feq`***

Una Possibile Soluzione

Per il modello di Shepherd, supponiamo di avere:

```
function es_shepherd_eq()
    % Dati del problema
    r = 1.72;
    k = 1000;
    feq = @(x) (x - f(x, r, k));
    [xeq, fval, flag] = fzero(feq, x0)
end

function xf = f(xc, r, k)
    xf = (r .* xc) ./ (1 + (xc./k).^2);
end
```

- È buona norma recuperare anche il valore di **`fval`** e **`flag`**
- Ci permettono di capire se qualcosa sia andato storto

Una Possibile Soluzione

Per il modello di Shepherd, supponiamo di avere:

```
function es_shepherd_eq()
    % Dati del problema
    r = 1.72;
    k = 1000;
    feq = @(x) (x - f(x, r, k));
    [xeq, fval, flag] = fzero(f2, x0) % x0 da scegliere
end

function xf = f(xc, r, k)
    xf = (r .* xc) ./ (1 + (xc./k).^2);
end
```

- Il valore di **x0** è da scegliere ed è molto importante
- Può determinare se il metodo converga, ed a quale valore

Una Possibile Soluzione

Per scegliere `x0` in modo più rigoroso:

- È utile disegnare la funzione da azzerare
- Nel caso di un sistema dinamico, dobbiamo disegnare:

$$x - f(x)$$

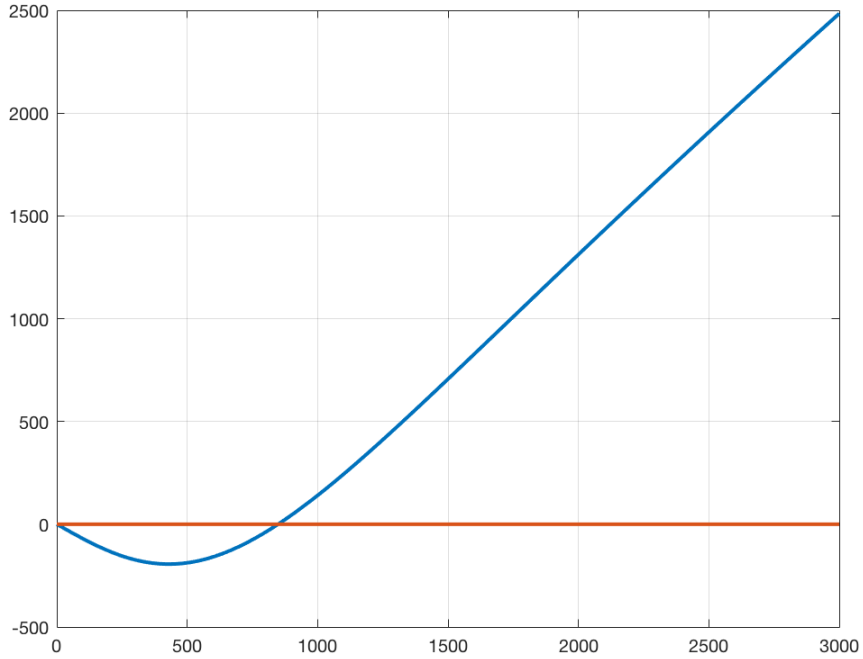
Per il modello di Shepherd, avremo qualcosa del genere:

```
figure();
plot(x, feq(x));
hold on
plot(x, zeros(size(x))); % Linea sullo 0
hold off
```

- Dove **`feq`** è la funzione da azzerare definita in precedenza

Una Possibile Soluzione

Il disegno ottenuto sarà più o meno come segue:



Ci sono due zeri (intersezioni con l'asse delle ascisse)!

Una Possibile Soluzione

Quindi, nel nostro caso l'equazione $x = f(x)$ ha due soluzioni

Ogni soluzione corrisponde ad un punto di equilibrio

- La prima soluzione è sempre $x = 0$ (nessun individuo)
- La seconda soluzione dipende dai dati del problema
- Di solito, il primo equilibrio è instabile, il secondo è stabile
- Per determinare la stabilità, continueremo ad usare la simulazione

In generale, quando si risolvono equazioni non lineari...

...È molto utile analizzare i risultati per verificarne l'attendibilità

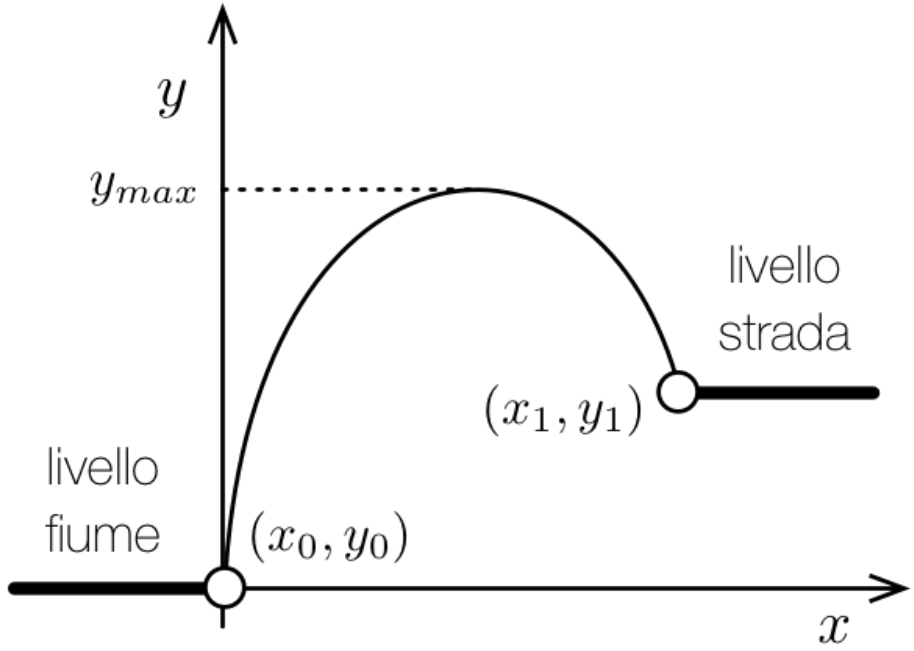
- Potete disegnare la funzione da azzerare
- Oppure simulare un sistema dinamico per verificare la stabilità

Elementi di Informatica e Applicazioni Numeriche T

Esempio: Progettazione di Curve
(Casi Non Lineari)

Esempio: Argine di un Fiume

Supponiamo di dover progettare l'argine di un fiume



- Alle ascisse abbiamo una posizione orizzontale x
- Alle ordinate abbiamo l'altezza y

Esempio: Argine di un Fiume

Supponiamo di dover progettare l'argine di un fiume:

L'argine deve:

- Essere definito da una curva parabolica
- Toccare il fiume in una posizione nota (x_0, y_0)
- Raggiungere il livello della strada nel punto (x_1, y_1)
- Raggiungere l'altezza massima y_{max}

A prima vista sembra simile ai problemi dell'ultima lezione

- Proviamo a risolverlo nello stesso modo...
- ...Impostando un insieme di equazioni

Equazioni per il Problema

Proviamo a formulare equazioni per le condizioni da rispettare:

- La curva deve passare per il punto (x_0, y_0) :

$$\alpha_2 x_0^2 + \alpha_1 x_0 + \alpha_0 = y_0$$

- La curva deve passare per il punto (x_1, y_1) :

$$\alpha_2 x_1^2 + \alpha_1 x_1 + \alpha_0 = y_1$$

- La curva raggiungerà il max nel punto x_3 in cui la derivata si annulla:

$$2\alpha_2 x_3 + \alpha_1 = 0$$

- In corrispondenza di x_3 , l'altezza dovrà essere y_{max} :

$$\alpha_2 x_3^2 + \alpha_1 x_3 + \alpha_0 = y_{max}$$

Equazioni per il Problema

Nel complesso, abbiamo:

$$\alpha_2 x_0^2 + \alpha_1 x_0 + \alpha_0 = y_0$$
$$\alpha_2 x_1^2 + \alpha_1 x_1 + \alpha_0 = y_1$$
$$2\alpha_2 x_3 + \alpha_1 = 0$$
$$\alpha_2 x_3^2 + \alpha_1 x_3 + \alpha_0 = y_{max}$$

- Le variabili sono $\alpha_2, \alpha_1, \alpha_0$, ma anche x_3
- Ci sono delle espressioni non lineari, i.e. $\alpha_2 x_3, \alpha_2 x_3^2, \alpha_1 x_3$

In sostanza, abbiamo un sistema di equazioni non lineari:

- Quindi, non possiamo impostare il problema in forma matriciale...
- ...Ma possiamo risolverlo con **`fsolve`**

Equazioni per il Problema

Innanzitutto portiamo tutti i membri a sx del segno =:

$$\alpha_2 x_0^2 + \alpha_1 x_0 + \alpha_0 - y_0 = 0$$
$$\alpha_2 x_1^2 + \alpha_1 x_1 + \alpha_0 - y_1 = 0$$
$$2\alpha_2 x_3 + \alpha_1 = 0$$
$$\alpha_2 x_3^2 + \alpha_1 x_3 + \alpha_0 - y_{max} = 0$$

Le espressioni ottenute definiscono i termini di una funzione vettoriale

- La funzione dovrà avere come input un singolo vettore (e.g. "p")
- Il vettore dovrà contenere tutte le variabili:

$$p = (\alpha_2, \alpha_1, \alpha_0, x_3)$$

- L'ordine può essere scelto liberamente

Implementazione

A questo punto possiamo codificare la funzione in Matlab:

```
function z = f(p, x0, x1, y0, y1, ymax)
    % La curva deve passare per (x0, y0)
    z(1) = p(1)*x0^2 + p(2)*x0 + p(3) - y0;
    % La curva deve passare per (x1, y1)
    z(2) = p(1)*x1^2 + p(2)*x1 + p(3) - y1;
    % La derivata si deve annullare in p(4)
    z(3) = 2*p(1)*p(4) + p(2);
    % in p(4) la curva deve valere ymax
    z(4) = p(1)*p(4)^2 + p(2)*p(4) + p(3) - ymax;
end
```

- La variabile restituita `z` è un vettore di 4 elementi
- Con la nostra convenzione, `p(4)` corrisponde ad x_3

Implementazione

La nostra `f(p,x0,x1,y0,y1,ymax)` è la funzione da azzerare

- Ma ha troppi argomenti!
- `fsolve` richiede una funzione con un singolo argomento

Risolviamo il problema con una funzione anonima:

```
x0 = 0;
x1 = 7;
y0 = 0;
y1 = 2;
ymax = 5;
% Espongo un solo parametro
fz = @(p) f(p, x0, x1, y0, y1, ymax);
```

- Il parametro da esporre è `p`, il vettore delle variabili

Implementazione

Per trovare uno zero possiamo usare:

```
[psol, fval, flag] = fsolve(fz, x0)
```

- In questo caso `x0` speficherà un valore per ogni variabile
- Per esempio, potremmo avere: `x0 = [-1, 1, 0, 3]`

Scegliere il valore x_0 per i sistemi di equazioni può essere complicato

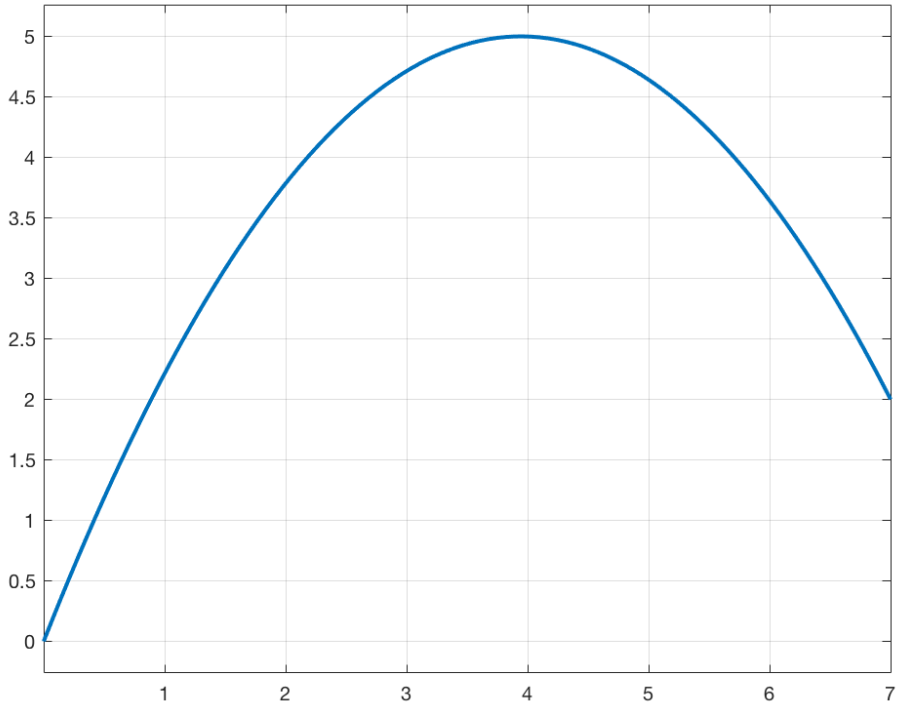
- Ci sono molti valori da decidere...
- ...E disegnare la funzione può essere complicato (troppe dimensioni)

Come linea guida:

- Usate l'intuizione! Specie se il problema ha senso fisico
- Se qualcosa non torna, fate diversi tentativi

Soluzione

Per il nostro problema, l'argine avrà questa forma:



- Il massimo è raggiunto per $x_3 = 3.9446$

Elementi di Informatica e Applicazioni Numeriche T

Esercizio: Equilibri per
il Modello di Beverton-Holt

Esercizio: Equilibri per Beverton-Holt

Si consideri il modello di Beverton-Holt:

$$x^{(k+1)} = \frac{r x^{(k)}}{1 + \frac{x^{(k)}}{N}}$$

Si tratta di un altro modello per l'evoluzione di una popolazione

- $x^{(k)}$ è il numero individui al passo k-mo
- r è un tasso di crescita
- N è un valore della popolazione per cui r si dimezza

Il file **`es_beverton_holt_eq.m`** nello start-kit contiene un simulatore

- Vogliamo provare a determinare lo stato di equilibrio...
- ...Risolvendo una equazione non lineare

Esercizio: Equilibri per Beverton-Holt

Estendete il codice in `es_beverton_holt_eq.m`

Costruite opportunamento la funzione da azzerare

- Disegnate l'andamento della funzione...
- ...Così da individuare visivamente la posizione degli zeri
- Se volete, utilizzate la funzione `plot_target_function`

Trovate lo zero con `fzero` o `fsolve`

- Verificate anche i valori restituiti di `fval` e `flag`

Provate a variare il valore di partenza x_0

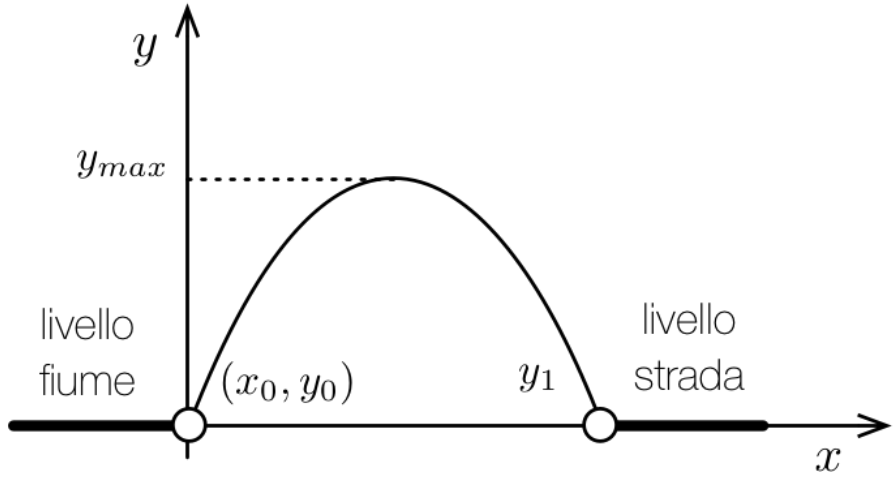
- Osservate se viene trovato uno zero diverso

Elementi di Informatica e Applicazioni Numeriche T

Esercizio: Argine di un Fiume (2)

Esercizio: Argine di un Fiume (2)

Consideriamo di nuovo l'esempio di costruzione di un argine



Come in precedenza:

- Alle ascisse abbiamo una posizione orizzontale x
- Alle ordinate abbiamo l'altezza y

Esercizio: Argine di un Fiume (2)

L'argine deve:

- Essere definito da una curva parabolica f
- Toccare il fiume in una posizione nota (x_0, y_0)
- Toccae la strada in un punto (x_1, y_1) ...
- ...Di cui non è nota la coordinata x_1
- Raggiungere l'altezza massima y_{max}
- Avere una superficie complessiva pari a s

La superficie sarà data da:

$$S = \int_{\underbrace{x_0}_{=0}}^{x_1} f(x)\, dx$$

Esercizio: Argine di un Fiume (2)

Il file `es_river_bank2.m` contiene i dati del problema:

- Impostate il problema di definizione della curva
- Vedrete che otterrete un sistema non lineare
- Risolvetelo con gli strumenti messi a disposizione da Matlab...
- ...Dopo aver introdotto una o più funzioni opportunamente definite
- Disegnate l'andamento della curva dell'argine

Suggerimento: usate una funzione "normale" più una anonima

- Non è obbligatorio farlo, ma verrà comodo in futuro

Elementi di Informatica e Applicazioni Numeriche T

Esercizio: Fluido Comprimibile in Condizioni Isoentropiche

Esercizio: Gas Isoentropico

In un fluido comprimibile isoentropico all'equilibrio...

...Pressione e temperature si distribuiscono secondo le equazioni:

$$p = p_0 \left[1 + \left(1 - \frac{1}{\gamma} \right) \frac{M}{RT_0} g(z_0 - z) \right]^{\frac{\gamma}{\gamma - 1}}$$

$$T = T_0 \left[1 + \left(1 - \frac{1}{\gamma} \right) \frac{M}{RT_0} g(z_0 - z) \right]$$

- P, T sono la pressione e temperatura a quota z
- P_0, T_0 sono la pressione e temperatura a quota z_0
- R è la costante dei gas perfetti
- M è la massa del gas
- γ è un parametro che caratterizza il gas

Esercizio: Gas Isoentropico

Se sono noti $P_0, T_0, \gamma, M, z_0 - z$ determinare P e T è facile:

- Perché le due equazioni sono esplicite in P e T
- Vale a dire: P e T compaiono da sole a sx dell'uguale

Supponiamo invece di disporre di P, T, P_0, T_0 e $z_0 - z$

- A partire dai dati noti (e.g. misurati con qualche strumento)...
- ...Vogliamo determinare le caratteristiche del gas (i.e γ e M)
- Le due equazioni date sono implicite in γ e M:
- Perché γ e M non possono essere isolate a sx del segno "="

Quindi dobbiamo risolvere un sistema di due equazioni non lineari

Esercizio: Gas Isoentropico

Il file `es_isentropic.m` contiene i dati del problema

- Impostate il problema di definizione di γ e M
- Risolvetelo con gli strumenti messi a disposizione da Matlab...
- ...Dopo aver introdotto una o più funzioni opportunamente definite
- Disegnate quindi l'andamento della curva dell'argine

Suggerimento: usate una funzione "normale" più una anonima

- Non è obbligatorio farlo, ma verrà comodo in futuro