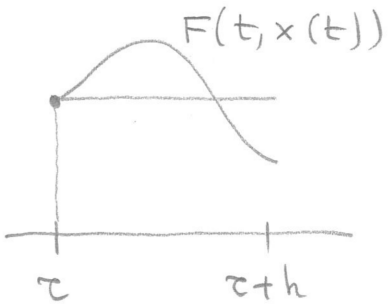
* Metodo di EULERO IMPLICITO *

- Dall'eq diff $\dot{x}(t) = F(t, x(t))$ si deduce, per ogni τ, h:

$$\boxed{x(\tau+h) - x(\tau) = \int_{\tau}^{\tau+h} \dot{x}(t)\, dt}$$

$$= \boxed{\int_{\tau}^{\tau+h} F(t, x(t))\, dt}$$



Ma:

$$\int_{\tau}^{\tau+h} F(t, x(t))\, dt =$$

$$= h\, F(\tau, x(\tau)) + \mathcal{O}(h^2)$$

da cui, eliminando il termine $\mathcal{O}(h^2)$ e ponendo $\tau = t_k$, $h = h_k$:

$$x_{k+1} = x_k + h_k\, F(t_k, x_k)$$

(in questo caso $x(t) = x(t; x_k, t_k)$). Si ottiene ancora il metodo di EULERO ESPLICITO.

Ponendo invece:

$$\int_{\tau}^{\tau+h} F(t, x(t))\, dt = h\, F(\tau+h, x(\tau+h)) + O(h^2)$$

si ottiene (operando come prima):

$$\boxed{x_{k+1} = x_k + h_k F(t_{k+1}, x_{k+1})}$$

Questa relazione definisce il metodo di <u>EULERO IMPLICITO</u>. Il termine si spiega osservando che la relazione definisce x_{k+1} IMPLICITAMENTE: x_{k+1} è <u>uno zero</u> (se esiste) della funzione

$$\boxed{g(z) = z - x_k - h_k F(t_{k+1}, z)}$$

si può (tentare di) approssimare uno zero della funzione $g: \mathbb{R}^n \to \mathbb{R}^n$ utilizzando, ad esempio, il metodo di Newton. In ogni caso, la ricerca di uno zero di g con un metodo iterativo ad un

punto richiede un punto iniziale.
Una scelta ragionevole è l'ultima
approx fornita dal metodo:

$k = 0$

finché $t_k < t_f$ ripeti

- scegli h_k
- $x_{k+1} = \text{newtonNd}(g, x_k, \dots)$

la scelta di h_k può essere fatta basandosi
sulla stima dell'errore locale:

$$EL_k \simeq \frac{1}{2} \| z''(t_{k-1}) \| h_{k-1}^2$$

e quindi:

$$h_k = \sqrt{\frac{2E}{\| z''(t_k) \|}}$$

Una realizzazione del metodo è nel
file LMV-eulero_imp-pv.sci riportato
alle pag seguenti.

```
function [T, X, PASSO, StimaEL]=LMV_eulero_imp_pv(x0, t0, tf, fct, fct2, EL_MAX, dialogo)
//
//  Integra numericamente, sull'intervallo [t0,tf], il problema
//  di Cauchy in R(n):
//  .
//  x = F(t,x)
//  x(t0) = x0
//
//  con il metodo di Eulero all'indietro a passo variabile.
//
//  x0: condizione iniziale (colonna di n elementi)
//  t0: istante iniziale
//  tf: istante finale
//  fct: function per F - fct(t,x) deve essere una colonna
//  fct2: function il cui valore fct2(t,x) è la derivata seconda in t della
//        soluzione dell'equazione differenziale che all'istante t assume valore x.
//  EL_MAX: errore locale massimo consentito
//  dialogo: se "loquace" mostra gli istanti di integrazione
//
//
//  T = [T(1),...,T(N)], nodi
//  X: matrice n x N - la colonna X(:,i) è la soluzione numerica
//     all'istante T(i)
//  PASSO: riga con PASSO(k) = h tale che T(k+1) = T(k) + h
//  StimaEL: riga delle stime dell'errore locale
//
n = length(x0); // determina il numero di equazioni del sistema
h_min = (tf - t0)/1d6; // passo minimo consentito
h_max = (tf - t0)/10;  // passo massimo consentito
T = [];
X = [];
PASSO = [];
StimaEL = [];
//
T(1,1) = t0;
X(:,1) = x0;
StimaEL(1,1) = 0;
//
// ciclo principale
//
while (T(1,$) < tf) & (PASSO(1,$) > h_min | PASSO(1,$) == []),
                      // l'iterazione si arresta se si è
                      // raggiunto tf o se non si è riuscito
                      // a rendere la stima dell'errore
                      // locale inferiore a EL_MAX
    h_max_loc = min(tf - T(1,$), h_max);
    // determina il passo
    Nd2x = norm(fct2(T(1,$),X(:,$)));
    if Nd2x == 0 then
       if PASSO == [] then PASSO(1,$+1) = min(h_min * 100, h_max_loc);
       else PASSO(1,$+1) = min(PASSO(1,$), h_max_loc); end;
    else PASSO(1,$+1) = min(sqrt(2*EL_MAX/Nd2x), h_max_loc);
                // passo per avere StimaEL = EL_MAX (o non superare tf)
    end;
    // calcola nuovo X e T
    h = PASSO(1,$);
    deff("Y=G(Z)","Y=Z-h*fct(T(1,$)+h,Z)-X(:,$)");
    X(:,$+1) = fsolve(X(:,$),G);
    T(1,$+1) = T(1,$) + PASSO(1,$);
    StimaEL(1,$+1) = (1/2) * Nd2x * PASSO(1,$)^2;
    if dialogo == "loquace" then printf("\nT = %3.2e",T($)); end;
end;
if T(1,$) < tf then
   printf("\n\nIntegrazione interrotta a T = %3.2e", T(1,$));
   printf("\n\nh_min = %3.2e , h = %3.2e\n", h_min, PASSO(1,$));
end;
if dialogo == "loquace" then printf("\n"); end;
//
```

Si cons (nuovamente) il pb di Cauchy

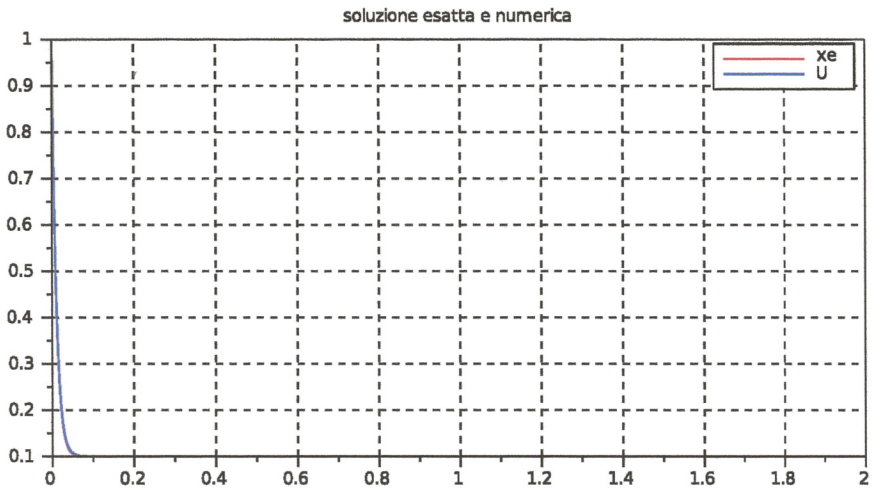
$$\begin{cases} \dot{x} = -100x + 10 \\ x(0) = 1 \end{cases} \qquad \text{su } [0,2]$$

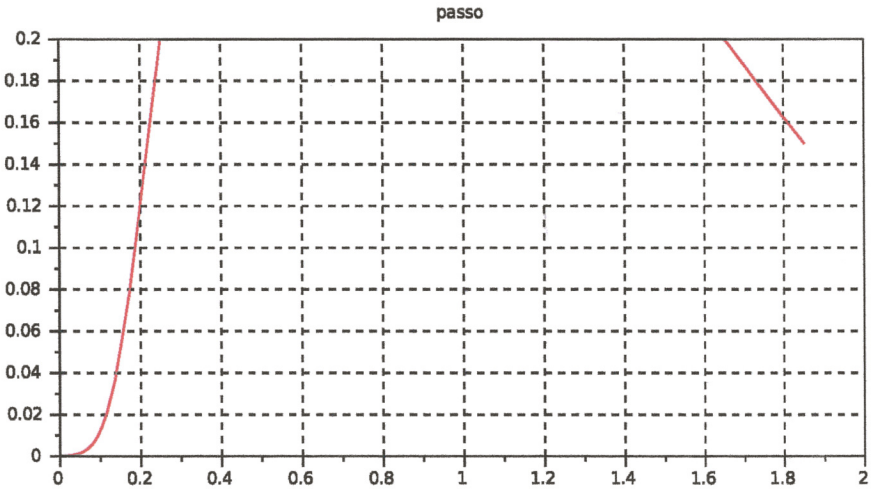
Per appross la soluz si utilizza la proc LMV_eulero_imp_pv, che realizza il metodo di EULERO IMPLICITO a passo variabile, con EL_MAX $= 10^{-4}$.

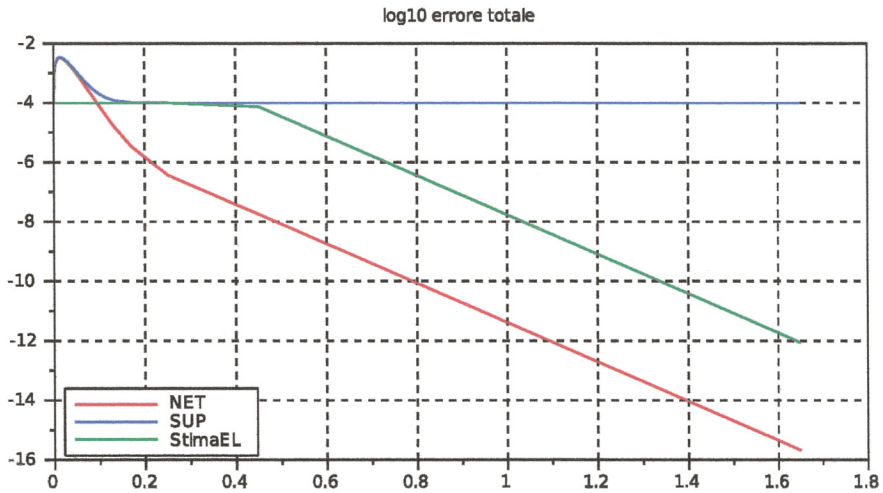
Si ottengono i grafici riportati alla pagina successiva.

Da un confronto con gli analoghi grafici ottenuti utilizzando la proc LMV_TS_1_pv, che realizza il metodo di Eulero ESPLICITO si constata che:

- sono <u>scomparse</u> le oscillazioni della soluzione numerica intorno al valore asintotico della soluz esatta $\left(\frac{1}{10}\right)$
- per $t > 0{,}3$ il passo scelto dalla procedura risulta costante e pari a 0,2 corrisp







Problema: dx/dt = -L x + 10 , L = 1.000D+02

Procedura: LMV_eulero_imp_pv

SUP(k) = EL_MAX + SUP(k-1)*exp(-L*PASSO(k-1))

EL_MAX = 1.000D-04

Errore totale massimo = 3.299D-03

Numero passi = 1.520D+02

ad una successione:

$$x_k - \frac{1}{10} = \left(\frac{1}{1+100h}\right)^k \left(x_0 - \frac{1}{10}\right)$$

monotona decrescente $\left(0 < \frac{1}{1+100h} < 1\right)$

- la stima EL ha andamento decrescente per $t > 0{,}3$ — nel tratto finale decresce in modo lineare

Oss:

(1) $h = 0{,}2$ è il MASSIMO passo consentito dalla procedura; visto l'andamento di stima EL (nel tratto finale inferiore al valore imposto 10^{-4}) la procedura "vorrebbe" aumentare il passo oltre 0,2.

(2) La quantità $\frac{1}{1+100h} \in (0,1)$ per OGNI $h > 0$: contrariamente a quanto accade utilizz il metodo di Eulero esplicito, la succ generata a passo costante è SEMPRE stabile!

(3) L'errore totale massimo risulta $\sim 3 \cdot 10^{-3}$ come nel caso di Eulero esplicito.

(4) Il numero di passi è 152, inferiore ai 226 necessari al metodo esplicito. Si osserva anche che dei 152 passi solo pochi (quanti?) sono necessari per attraversare la soluzione da 0,3 a 2.

Le realizzazioni viste di TS(1), TS(2) e Eulero implicito richiedono all'utilizzatore l'inserimento (fct2 nei casi TS(1) e Eulero implicito, fct3 nel caso TS(2)) di una funzione utilizzata SOLAMENTE per la scelta del passo (dunque non necessaria per il calcolo di x_{k+1}). Questo può essere evitato con realizzazioni meno "ingenue" dei vari metodi.

Ad esempio, per un metodo di ORDINE 2 (ovvero in cui $el_{k+1} = C_k h_k^3 + \mathcal{O}(h_k^4)$),

si può procedere come segue:

- $k = 0$
- scegli h_*
- <u>finchè</u> $t_k < t_f$ <u>ripeti</u>

1) calcola EL_*, stima di EL utilizzando h_* (dell'errore locale che la proc commetterebbe se utilizzasse passo h_*)

2) calcola $h_k = \left(\frac{E}{EL_*}\right)^{\frac{1}{3}} h_*$

(con questo passo la procedura commette un err locale pari ad E)

3) calcola x_{k+1} utilizz h_k e

$t_{k+1} = t_k + h_k$

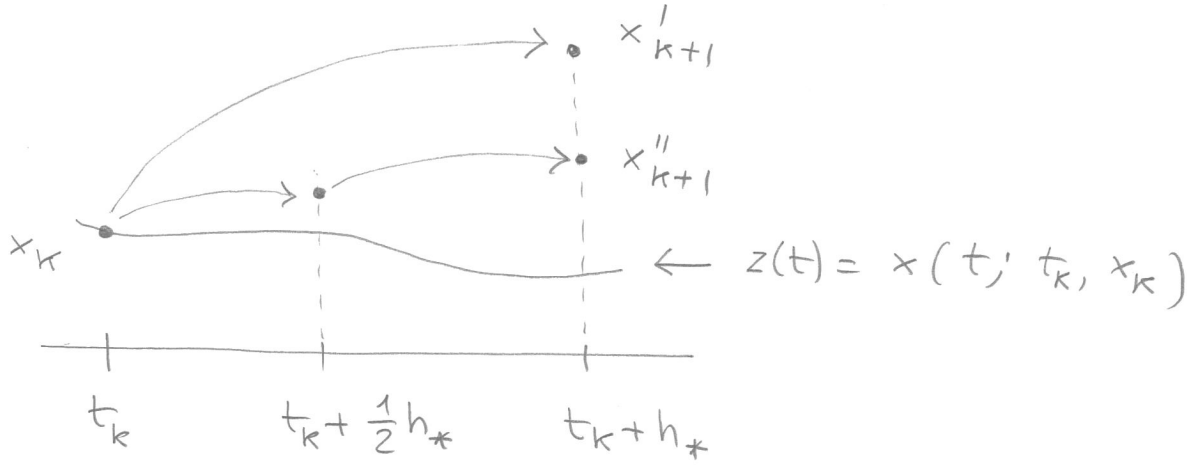
4) $h_* = h_k$

5) $k \leftarrow k+1$

<u>Oss</u>: (1) <u>Come calcolare</u> EL_*:

(a) calcolare x'_{k+1} usando h_* e x_k; si ottiene:

$$x'_{k+1} - z(t_{k+1}) = C_k h_*^3 + O(h_*^4)$$



(b) calcolare x''_{k+1} con DUE passi del metodo numerico, ciascuno con passo $\frac{1}{2}h_*$; si ottiene:

$$x''_{k+1} - z(t_{k+1}) \simeq 2\, c_k \left(\frac{1}{2}h_*\right)^3 + \mathcal{O}(h_*^4)$$

(c) Infine:

$$x'_{k+1} - x''_{k+1} = \left[x'_{k+1} - z(t_{k+1})\right] -$$

$$- \left[x''_{k+1} - z(t_{k+1})\right] =$$

$$= \underbrace{c_k \left(h_*^3 - \frac{h_*^3}{4}\right)}_{= \frac{3}{4} c_k h_*^3} + \mathcal{O}(h_*^4)$$

(d) Si ottiene la STIMA dell'errore locale comunque con passo h_*:

$$\boxed{EL_* = c_k h_*^3 \simeq \frac{4}{3}\left(x'_{k+1} - x''_{k+1}\right)}$$

(2) Come scegliere h_k:

(a) $EL_* \simeq C_k h_*^3 \Rightarrow C_k \simeq \frac{EL_*}{h_*^3}$

(b) imponendo $C_k h_k^3 = E$

si ottiene $h_k = \left(\frac{E}{C_k}\right)^{\frac{1}{3}}$

(c) dunque:

$$h_k = \left(\frac{E\, h_*^3}{EL_*}\right)^{\frac{1}{3}} = h_* \left(\frac{E}{EL_*}\right)^{\frac{1}{3}}$$

Si noti che: SE $EL_* > E$ (errore locale con passo h_* troppo grande) ALLORA $h_k < h_*$ (si diminuisce il passo); SE $EL_* < E$ (errore locale con passo h_* troppo piccolo) ALLORA $h_k > h_*$ (si aumenta il passo).