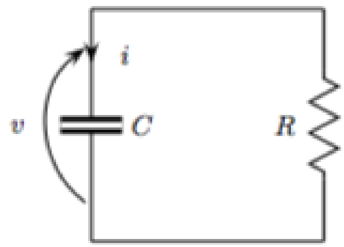
Integrazione numerica applicata alla simulazione circuitale



Consideriamo come base per la presentazione di questo argomento l'equazione di stato del circuito di figura, che descrive la scarica di un condensatore su di un resistore (scarica RC):

$\dot{x} = \lambda x$ (1)

avendo posto $x = v$ e $\lambda = -\frac{1}{RC}$

La soluzione analitica della (1) è:

$$x(t) = x_0 e^{\lambda t}$$

essendo $x_0 = v(t_0)$ la tensione sul condensatore all'istante iniziale t_0.

La soluzione dell'equazione (1) nell'intervallo $t_0 - t_{stop}$ può essere ottenuta in forma approssimata per via numerica.

Chiamiamo x_n un'approssimazione della soluzione all'istante t_n e immaginiamo di voler determinare x_{n+1} al successivo istante di tempo t_{n+1} compiendo uno step temporale $h = t_{n+1} - t_n$.

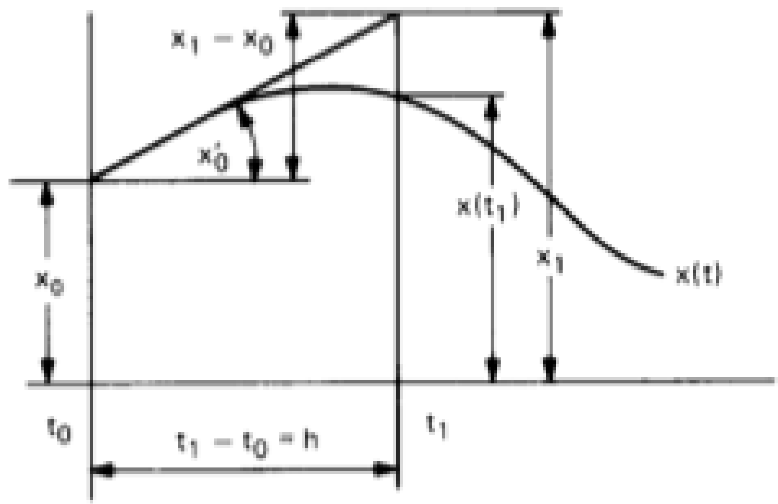
Si inizia per $n = 0$, dove abbiamo il valore esatto $x_0 = x(t_0)$; per i successivi valori di n la soluzione numerica x_n è solo una approssimazione della soluzione esatta $x(t_0 + nh)$.

Integrazione Forward Euler

Un primo modo per determinare il valore di x_1 consiste nell'approssimare numericamente la derivata $\dot{x}_0 = \frac{x_1 - x_0}{h}$ il che significa graficamente determinare il valore approssimato x_1 utilizzando la tangente alla soluzione in x_0:

$x_1 = x_0 + h\dot{x}_0 = x_0 + h(\lambda x_0)$ (2)

la formula di integrazione (2) si chiama Forward Euler (FE).



Come mostra la figura, si commette un errore tanto più grande quanto maggiore è il passo di integrazione h.

Vediamo di seguito qualche passo di integrazione con FE per l'esempio proposto:

Esempio 1: qualche passo di integrazione con il FE

```
R=1000.0;
C=1.0e-3;
lambda=-1/(R*C);
tau = -1.0/lambda;
x0=10.0;         %condizione iniziale
h=0.5;
tstop=5*tau;
time=[0:h:tstop];
numsteps=size(time,2);

fe = @(x)(x+h*lambda*x); %definizione della funzione 'forward Euler formula'
esatta=x0*exp(lambda.*time);%definizione della soluzione analitica esatta


approxFE = zeros(1,numsteps);
%facciamo qualche passo di integrazionne

%calcolo la soluzione numerica per tutto l'intervallo di integrazione
approxFE(1)=x0;
for i=1:numsteps-1
    approxFE(i+1)=fe(approxFE(i));
end;

%vediamo qualche passo in dettaglio
x_old=x0;
for i=1:min(10,numsteps)
    disp(sprintf('passo %d, t=%5.3f, x=%5.3f, xe=%5.3f, errore: %5.3f',i, time(i), approxFE(i)
end
```

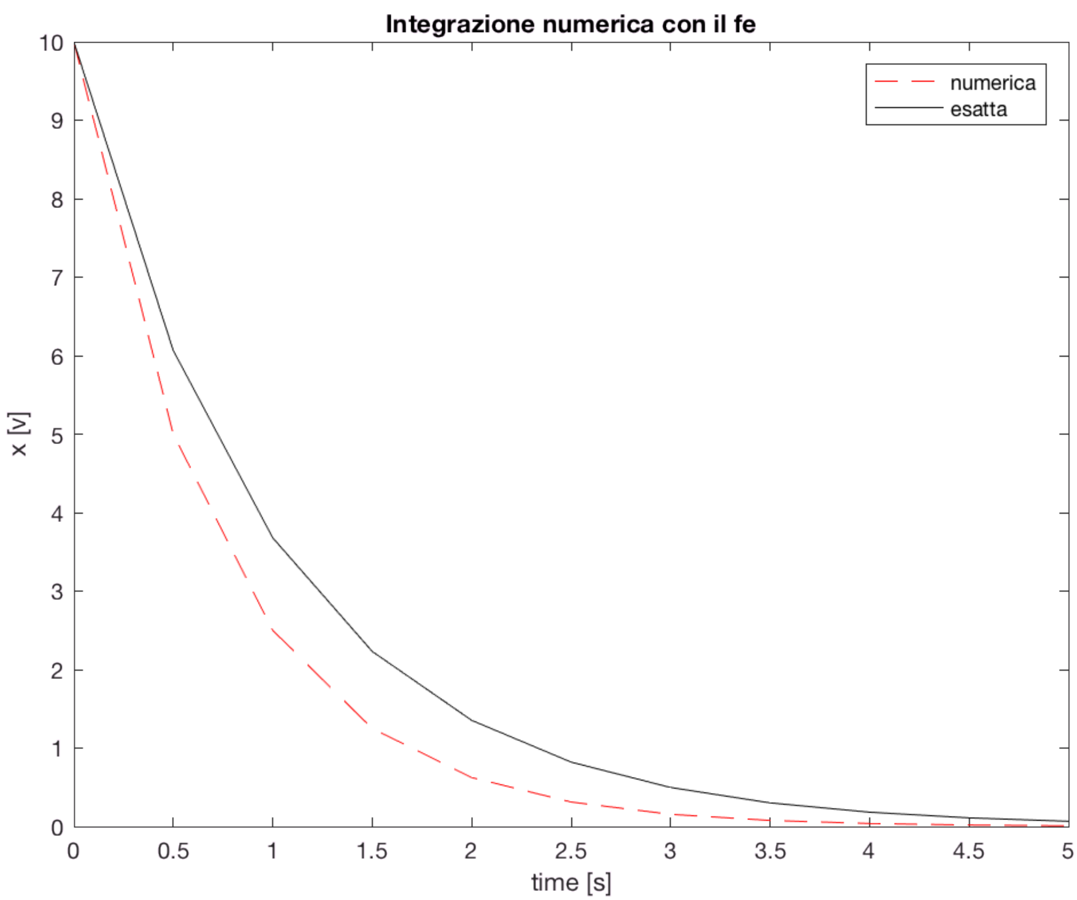
```
passo 1, t=0.000, x=10.000, xe=10.000, errore: 0.000
passo 2, t=0.500, x=5.000, xe=6.065, errore: -1.065
```

```
passo 3, t=1.000, x=2.500, xe=3.679, errore: -1.179
passo 4, t=1.500, x=1.250, xe=2.231, errore: -0.981
passo 5, t=2.000, x=0.625, xe=1.353, errore: -0.728
passo 6, t=2.500, x=0.312, xe=0.821, errore: -0.508
passo 7, t=3.000, x=0.156, xe=0.498, errore: -0.342
passo 8, t=3.500, x=0.078, xe=0.302, errore: -0.224
passo 9, t=4.000, x=0.039, xe=0.183, errore: -0.144
passo 10, t=4.500, x=0.020, xe=0.111, errore: -0.092
```

Procedento in questo modo si ottiene la soluzione approssimata dell'equazione (1) per tutti i punti dell'intervallo di integrazione.

Rappresentiamo le due soluzioni graficamente:

```
plot(time, approxFE, 'r--',time, esatta,'k')
title('Integrazione numerica con il fe')
xlabel('time [s]')
ylabel('x [v]')
legend('numerica', 'esatta')
```



Integrazione Backward Euler

Una alternativa al fe consiste nell'approssimare numericamente la derivata $\dot{x}_1 = \frac{x_1 - x_0}{h}$ il che significa graficamente determinare il valore approssimato x_1 utilizzando la tangente alla soluzione in x_1:

$x_1 = x_0 + h\dot{x}_0 = x_0 + h(\lambda x_1)$ (3)

ricavando x_1 dalla (3) si ottiene:

$$x_1 = (1 - \lambda h)^{-1} x_0$$

la formula di integrazione (3) si chiama Backward Euler (BE).

Vediamo di seguito qualche passo di integrazione con BE per l'esempio proposto:

Esempio 1: qualche passo di integrazione con il BE

```
be = @(x)(x/(1-h*lambda)); %definizione della funzione 'Backward Euler formula'


approxBE = zeros(1,numsteps);
%facciamo qualche passo di integrazionne

%calcolo la soluzione numerica per tutto l'intervallo di integrazione
approxBE(1)=x0;
for i=1:numsteps-1
    approxBE(i+1)=be(approxBE(i));
end;

%vediamo qualche passo in dettaglio
for i=1:min(10,numsteps)
    disp(sprintf('passo %d, t=%5.3f, x=%5.3f, xe=%5.3f, errore: %5.3f',i, time(i), approxBE(i)
end
```

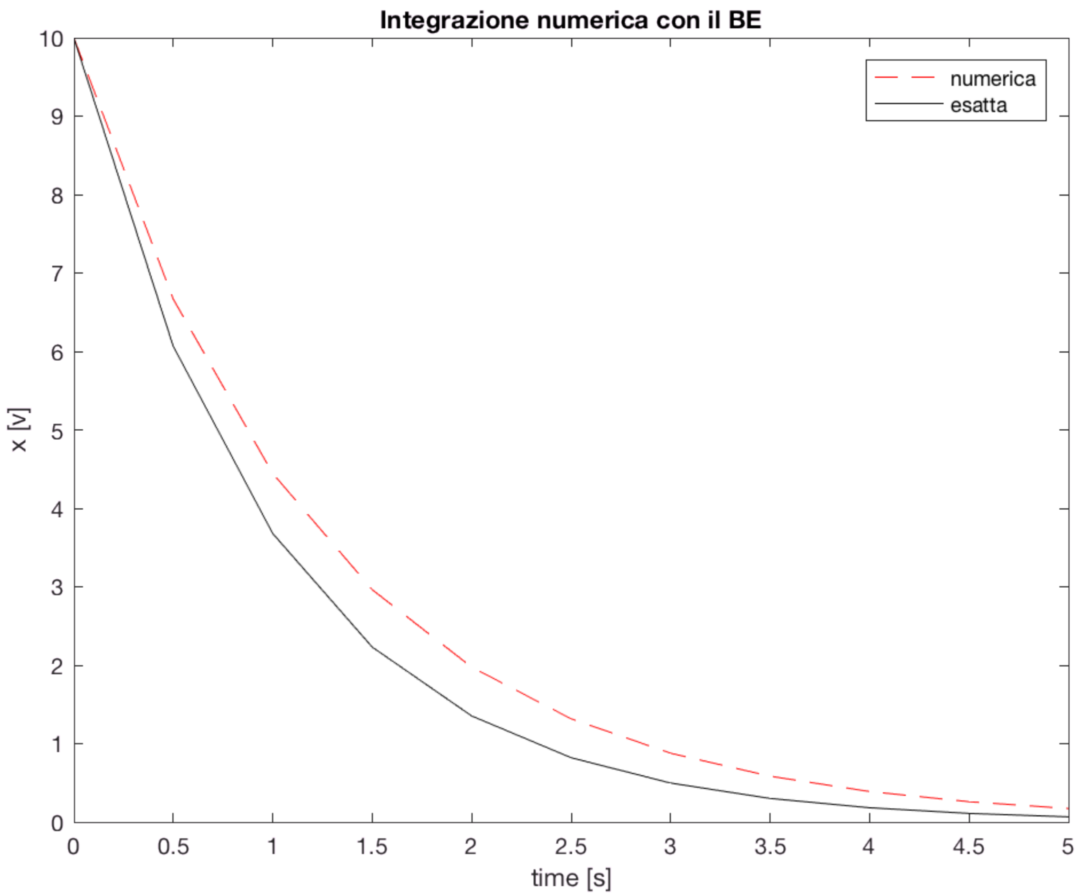
```
passo 1, t=0.000, x=10.000, xe=10.000, errore: 0.000
passo 2, t=0.500, x=6.667, xe=6.065, errore: 0.601
passo 3, t=1.000, x=4.444, xe=3.679, errore: 0.766
passo 4, t=1.500, x=2.963, xe=2.231, errore: 0.732
passo 5, t=2.000, x=1.975, xe=1.353, errore: 0.622
passo 6, t=2.500, x=1.317, xe=0.821, errore: 0.496
passo 7, t=3.000, x=0.878, xe=0.498, errore: 0.380
passo 8, t=3.500, x=0.585, xe=0.302, errore: 0.283
passo 9, t=4.000, x=0.390, xe=0.183, errore: 0.207
passo 10, t=4.500, x=0.260, xe=0.111, errore: 0.149
```

Procedento in questo modo si ottiene la soluzione approssimata dell'equazione (1) per tutti i punti dell'intervallo di integrazione.

Rappresentiamo le due soluzioni graficamente:

```
plot(time, approxBE, 'r--',time, esatta,'k')
title('Integrazione numerica con il BE')
```

```
xlabel('time [s]')
ylabel('x [v]')
legend('numerica', 'esatta')
```



Integrazione con Trapezoidal Rule

Visto che l'errore commesso con il FE è sempre di sovrastima della soluzione e quello commesso con il BE di sottostima, una scelta alternativa potrebbe essere quella di approssimare la derivata con la media delle derivate in x_0 ed x_1:

$\frac{x_1 - x_0}{h} = \frac{x'_0 + x'_1}{2}$ (4)

ricavando x_1 dalla (4) si ottiene:

$$x_1 = \frac{1 + \lambda h/2}{1 - \lambda h/2} x_0$$

nota come Trapezoidal Rule (TR)

Vediamo di seguito qualche passo di integrazione con TR per l'esempio proposto:

Esempio 1: qualche passo di integrazione con il TR

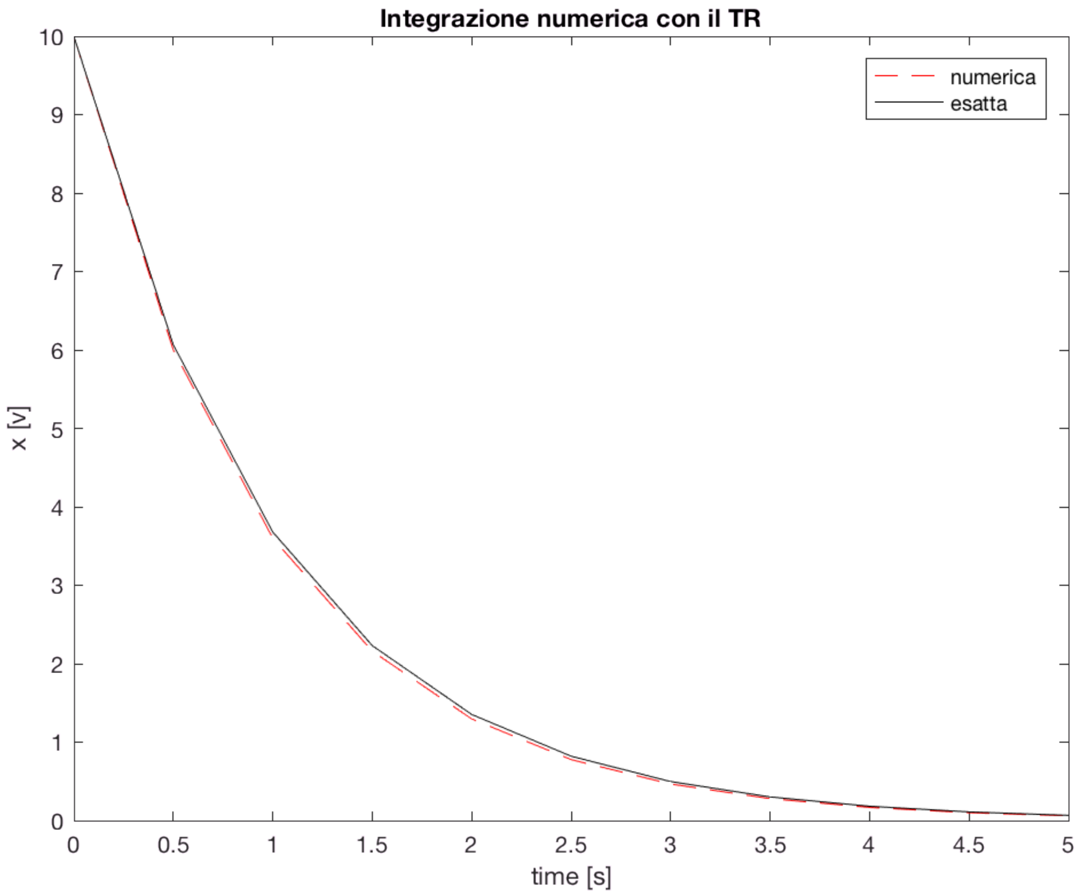
```
tr = @(x)((1+lambda*h/2)/(1-lambda*h/2)*x); %definizione della funzione 'trapezoidal rule'


approxTR = zeros(1,numsteps);
%facciamo qualche passo di integrazionne

%calcolo la soluzione numerica per tutto l'intervallo di integrazione
approxTR(1)=x0;
for i=1:numsteps-1
    approxTR(i+1)=tr(approxTR(i));
end;

%vediamo qualche passo in dettaglio
for i=1:min(10,numsteps)
    disp(sprintf('passo %d, t=%5.3f, x=%5.3f, xe=%5.3f, errore: %5.3f',i, ...
        time(i), approxTR(i), esatta(i), approxTR(i)-esatta(i)));
end
```

```
passo 1, t=0.000, x=10.000, xe=10.000, errore: 0.000
passo 2, t=0.500, x=6.000, xe=6.065, errore: -0.065
passo 3, t=1.000, x=3.600, xe=3.679, errore: -0.079
passo 4, t=1.500, x=2.160, xe=2.231, errore: -0.071
passo 5, t=2.000, x=1.296, xe=1.353, errore: -0.057
passo 6, t=2.500, x=0.778, xe=0.821, errore: -0.043
passo 7, t=3.000, x=0.467, xe=0.498, errore: -0.031
passo 8, t=3.500, x=0.280, xe=0.302, errore: -0.022
passo 9, t=4.000, x=0.168, xe=0.183, errore: -0.015
passo 10, t=4.500, x=0.101, xe=0.111, errore: -0.010
```

Procedento in questo modo si ottiene la soluzione approssimata dell'equazione (1) per tutti i punti dell'intervallo di integrazione.

Rappresentiamo le due soluzioni graficamente:

```
plot(time, approxTR, 'r--',time, esatta,'k')
title('Integrazione numerica con il TR')
xlabel('time [s]')
ylabel('x [v]')
legend('numerica', 'esatta')
```



Ordine di integrazione ed Errore locale di Troncamento

Deriviamo adesso alcune proprietà per le tre formule appena presentate.

Innanzitutto osserviamo che, in forma più generale, le tre formule (FE, BE, TR) sono tutte descritte dalla:

$b_0 x'_0 + b_1 x'_1 = (x_1 - x_0)/h$ (5)

dove:

$b_0 = 1, \quad b_1 = 0$ per il FE

$b_0 = 0, \quad b_1 = 1$ per il BE

$b_0 = 1/2, \quad b_1 = 1/2$ per il TR

Riscriviamo la (5) con i valori esatti della soluzione $x(t)$ nei due istanti di tempo t_0 e $t_1 = t_0 + h$:

$h(b_0 x'(t_0) + b_1 x'(t_0 + h)) - x(t_0 + h) + x(t_0) = 0$ (6)

Possiamo sviluppare in serie di Taylor i termini presenti nella formula:

$$x(t_0 + h) = x(t_0) + \frac{h}{1!}x'(t_0) + \frac{h^2}{2!}x''(t_0) + \frac{h^3}{3!}x'''(t_0)$$

$$x'(t_0 + h) = x'(t_0) + \frac{h}{1!}x''(t_0) + \frac{h^2}{2!}x'''(t_0) + \frac{h^3}{3!}x''''(t_0)$$

sostituiti nella (6) e riorganizzati portando a sinistra i termini sino alla derivata prima si ha:

$$hx'(t_0)[b_0 + b_1 - 1] = h^2x''(t_0)[1/2 - b_1] + h^3x'''(t_0)[1/6 - b_1/2]$$

Per soddisfare questa relazione occorre annullare tutti i termini:

$c_1 = b_0 + b_1 - 1 = 0$ Condizione sempre soddisfatta con tutti i tre metodi

$c_2 = 1/2 - b_1 = 0$ (7)

la condizione (7) non è soddisfatta con FE (in questo caso si ha $c_2 = 1/2$)

la condizione (7) non è soddisfatta per il BE (in questo caso si ha $c_2 = -1/2$)

Dal momento che sia FE che BE hanno nulli i coefficienti della derivata di ordine 0 e di ordine 1, introduciamo il concetto di ordine di integrazione p come la massima derivata con coefficiente nullo. Il termine non nullo (quindi $c_2h^2x''(t_0)$ sia per FE che per TR) rappresenta

l'errore di troncamento.

Per il TR la (7) è invece soddisfatta ed il primo termine che non si annulla a destra è $c_3 = -1/12$; il TR è quindi un metodo di integrazione di ordine 2 e l'errore di troncamento vale $-h^3x'''(t_0)/12$.

Il fatto che TR nell'esempio ha prodotto risultati migliori è dovuto al fatto il TR è un metodo di integrazione più accurato sia di FE che di BE che sono di ordine 1 dal momento che l'errore di troncamento è minore.

Stabilità dei metodi di integrazione

Sempre con riferimento all'equazione (1), che ha una soluzione asinoticamente stabile quando $\lambda < 0$, occorre verificare anche la stabilità numerica del metodo di integrazione usato.

- Forwar Euler

consideriamo i primi passi del FE:

$$x_1 = (1 + \lambda h)x_0$$

$$x_2 = (1 + \lambda h)x_1 = (1 + \lambda h)^2 x_0$$

$$\dots$$

$$x_n = (1 + \lambda h)x_{n-1} = (1 + \lambda h)^n x_0$$

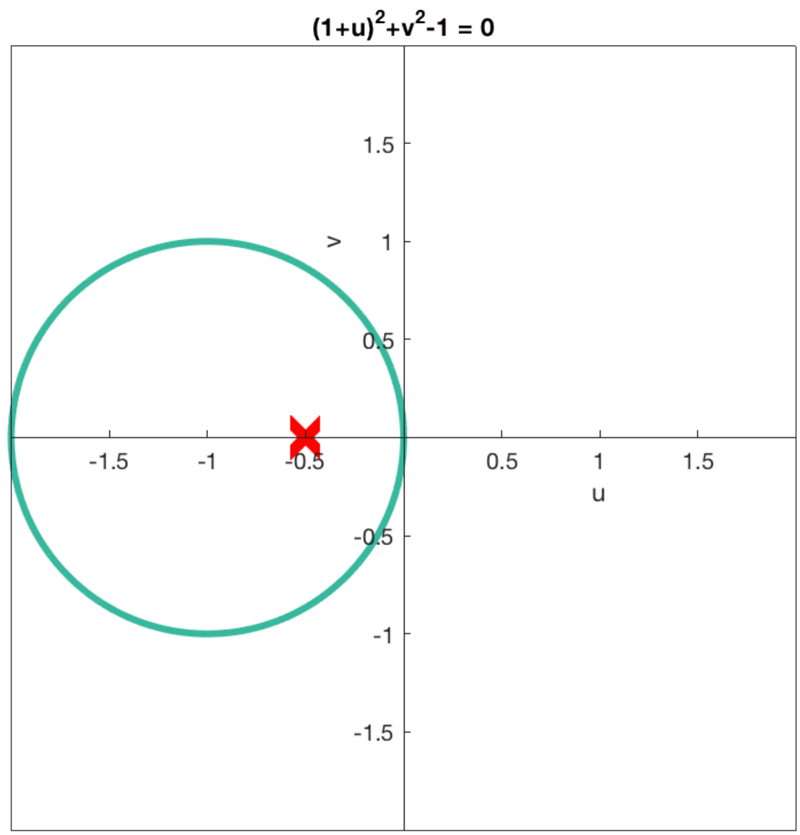
poichè $x(t) \to 0$ quando $t \to \infty$ anche $x_n \to 0,$ quando $n \to \infty$; perchè ciò accada dovra essere:

$$|1 + \lambda h| \le 1$$

poichè in generale λ può essere anche un numero complesso, posto $q = \lambda h = u + jv$ la condizione di stablità diventa:

$$(1 + u)^2 + v^2 \le 1$$

```
ez1=ezplot('(1+u)^2+v^2-1',[-2,2],[-2,2]);
set(ez1,'LineWidth',3);
axis equal
ax=gca;
ax.XAxisLocation='origin';
ax.YAxisLocation='origin';
hold on;
plot(lambda*h,0,'rx','MarkerSize',16,'LineWidth', 5);
hold off
```



- Backward Euler

consideriamo i primi passi del BE:

$$x_1 = (1 - \lambda h)^{-1} x_0$$

$$x_2 = (1 - \lambda h)^{-1} x_1 = (1 - \lambda h)^{-2} x_0$$

$$\ldots$$

$$x_n = (1 - \lambda h)^{-1} x_{n-1} = (1 - \lambda h)^{-n} x_0$$

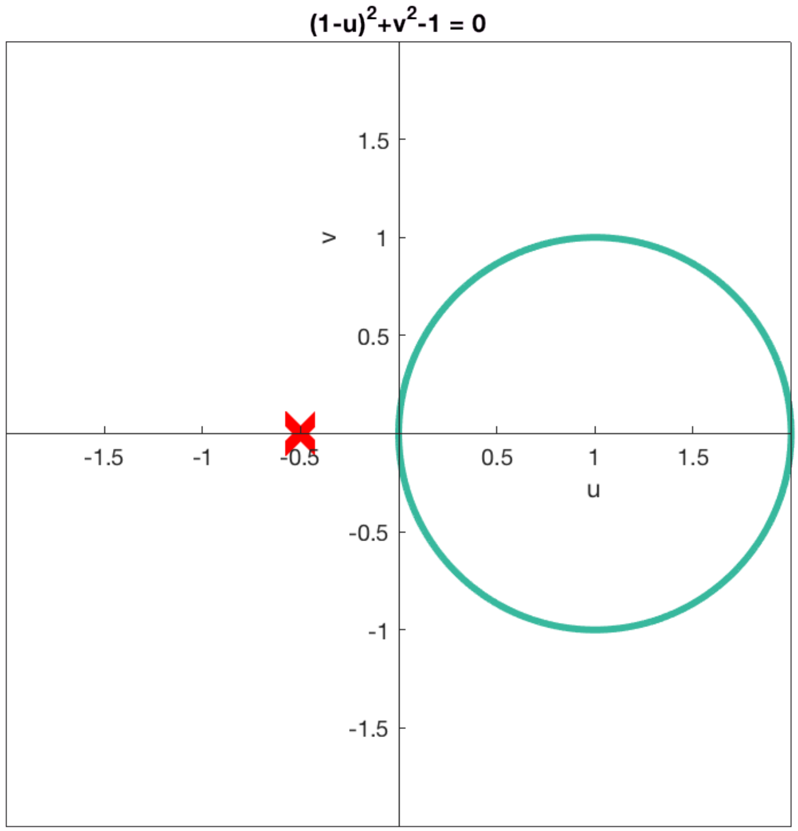
poichè $x(t) \to 0 \text{ quando } t \to \infty$ anche $x_n \to 0, \text{ quando } n \to \infty$; perchè ciò accada dovra essere:

$$|1 - \lambda h| \geq 1$$

poichè in generale λ può essere anche un numero complesso, posto $q = \lambda h = u + jv$ la condizione di stablità diventa:

$$(1 - u)^2 + v^2 \geq 1$$

```
ez2=ezplot('(1-u)^2+v^2-1',[-2,2],[-2,2]);
set(ez2,'LineWidth',3);
axis equal
ax=gca;
ax.XAxisLocation='origin';
ax.YAxisLocation='origin';
hold on;
plot(lambda*h,0,'rx','MarkerSize',16,'LineWidth', 5);
hold off
```



- Trapezoidal rule

consideriamo i primi passi del TR:

$$x_1 = \frac{1+\lambda h/2}{1-\lambda h/2} x_0$$

$$x_2 = \frac{1+\lambda h/2}{1-\lambda h/2} x_1 = (\frac{1+\lambda h/2}{1-\lambda h/2})^2 x_0$$

$$\ldots$$

$$x_n = (\frac{1+\lambda h/2}{1-\lambda h/2})^n x_0$$

poichè $x(t) \to 0 \text{ quando } t \to \infty$ anche $x_n \to 0, \text{ quando } n \to \infty$; perchè ciò accada dovra essere:

$$\left| \frac{1+\lambda h/2}{1-\lambda h/2} \right| \leq 1$$

poichè in generale λ può essere anche un numero complesso, posto $q = \lambda h = u + jv$ la condizione di stablità diventa:

$$\frac{(1+u/2)^2+v^2/4}{(1-u/2)^2+v^2/4} \leq 1$$

```
ez3=ezplot('((2+u)^2+v^2)/((2-u)^2+v^2)-1',[-2,2],[-2,2]);
set(ez3,'LineWidth',3);
```

```
axis equal
ax=gca;
ax.XAxisLocation='origin';
ax.YAxisLocation='origin';
hold on;
plot(lambda*h,0,'rx','MarkerSize',16,'LineWidth', 5);
hold off
```