

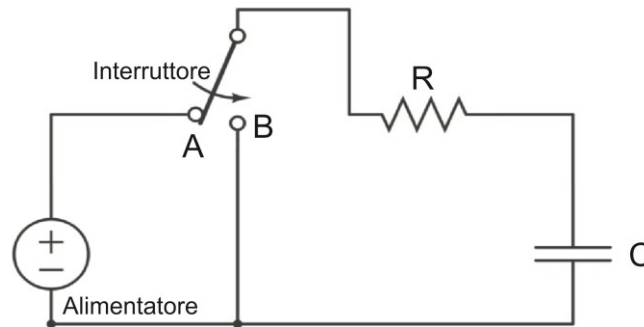
Analisi dei fenomeni transitori nel circuito RC

Scopo dell'esperimento

Lo scopo di questo esperimento è quello di studiare quantitativamente il processo di carica e scarica di un condensatore posto in serie ad un resistore. Mediante un fit dei dati sperimentali si verificherà se la costante di tempo del circuito è compatibile con il valore teorico.

Teoria del circuito RC

Il circuito RC è composto da un condensatore di capacità C e da un resistore di resistenza R posti in serie a un alimentatore di forza elettromotrice f_{em} ; un possibile schema del circuito è mostrato nella figura seguente.



Il processo di carica del condensatore

La *seconda legge di Kirchhoff*, applicata ad un circuito RC, ci dice che durante la fase di carica (con l'interruttore nella posizione A), sulle armature del condensatore si accumula una carica variabile nel tempo $q(t)$, definita attraverso la seguente equazione differenziale:

$$f_{em} - R \frac{dq}{dt} - \frac{q(t)}{C} = 0$$

dove $i(t) = \frac{dq}{dt}$ è la corrente elettrica istantanea uguale alla derivata prima della carica $q(t)$.

Dalla condizione iniziale $q(0) = 0$, condensatore inizialmente scarico, si ottiene la soluzione:

$$q(t) = C f_{em} (1 - e^{-t/\tau})$$

dove $\tau = RC$ è la costante di tempo del circuito.

La differenza di potenziale ai capi del condensatore è definita dalla seguente espressione:

$$V_C(t) = \frac{q(t)}{C} = f_{em} (1 - e^{-t/\tau})$$

Il processo di scarica del condensatore

La *seconda legge di Kirchhoff*, applicata ad un circuito RC, ci dice che durante la fase di scarica (l'alimentatore viene escluso ponendo l'interruttore nella posizione B), la carica $q(t)$ presente sulle armature del condensatore diminuisce nel tempo secondo la seguente equazione differenziale:

$$R \frac{dq}{dt} + \frac{q(t)}{C} = 0$$

dove $i(t) = \frac{dq}{dt}$ è la corrente elettrica istantanea uguale alla derivata prima della carica $q(t)$.

Dalla condizione iniziale $q(0) = C \cdot f_{em}$, condensatore inizialmente carico, si ottiene la soluzione:

$$q(t) = C f_{em} \cdot e^{-t/\tau}$$

dove $\tau = RC$ è la costante di tempo del circuito.

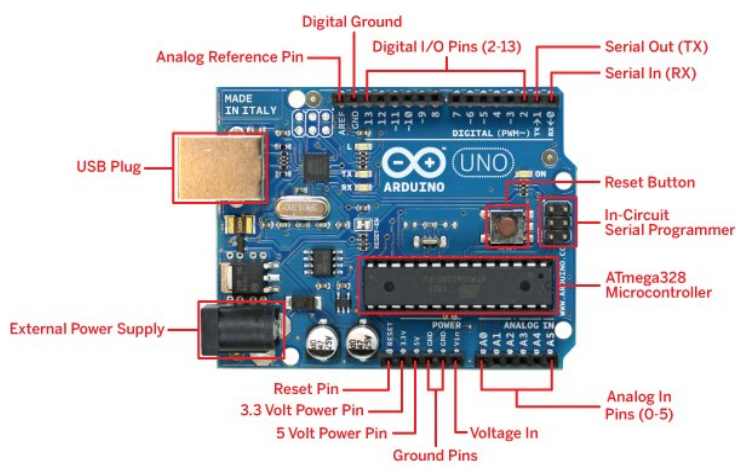
La differenza di potenziale ai capi del condensatore è definita dalla seguente espressione:

$$V_C(t) = \frac{q(t)}{C} = f_{em} \cdot e^{-t/\tau}$$

Esecuzione dell'esperimento

Per costruire il sistema di acquisizione dati si utilizzerà una scheda *Arduino uno R3* su cui verrà caricato lo *sketch* in grado di alimentare il circuito con una tensione f_{em} uguale a $5 V$ e di misurare la differenza di potenziale ai capi del condensatore, in funzione del tempo. Per alimentare il circuito si utilizzerà una porta **digitale** della scheda Arduino, mentre per misurare la tensione ai capi del condensatore si utilizzerà una delle porte **analogiche**. Mediante queste porte Arduino riesce a leggere un potenziale nel range $0-5 V$; il valore misurato dal *convertitore analogico/digitale (ADC)* restituisce un numero intero da 0 a 1023 (*10 bits*), quindi per ottenere il valore in Volt bisogna dividere per 1023.0 e moltiplicare per 5.0 il valore letto dalla porta analogica. I valori della differenza di potenziale ai capi del condensatore e del corrispondente istante di tempo saranno memorizzati in due vettori e stampati sullo schermo, cioè sul monitor seriale di Arduino.

Sketch per Arduino



//Analisi del circuito RC - Sketch per Arduino (completo)

```
const int analogPin = 3; // Definisce la porta A3 per la lettura della tensione
sul condensatore
const int digitalPin = 11; // Definisce il pin per la carica: PIN 11: 1 condensatore,
PIN 10: 2 cond. in parallelo, PIN 9: 2 cond. in serie, pin 6: 1 cond. in serie con
altri due in parallelo
int i; // Definisce la variabile intera i (contatore)
int delays = 10; // intervallo di tempo in ms tra la lettura delle misure
int M = 100; // numero di misure effettuate (M*delays deve essere maggiore o uguale
a 5 tau = 5 R*C, dove R è la resistenza e C la capacità; M <= 400 per problemi di
memoria)
int StartTime; // Definisce la variabile intera StartTime (istante di tempo
iniziale)

// Istruzioni di inizializzazione
void setup()
{
  Serial.begin(9600); // Inizializza la porta seriale a 9600
  Serial.flush(); // Pulisce il buffer della porta seriale
  pinMode(digitalPin, OUTPUT); // Definisce digitalPin come output
  bitClear(ADCSRA, ADPS0); // Istruzioni necessarie per velocizzare
  bitClear(ADCSRA, ADPS2); // il rate di acquisizione analogica
}

// Istruzioni del programma
void loop()
{
  int V[M]; // Definisce l'array intero V (dove vengono salvate le tensioni)
  int t[M]; // Definisce l'array t come intero (dove vengono salvati i tempi)
  Serial.println("\t Aspetto due secondi per scaricare il condensatore");
```

```

digitalWrite(digitalPin, LOW); // Pone digitalPin al valore low
delay(2000); // Aspetta 2 s per scaricare il condensatore ( potrebbe avere
una carica residua non nulla)
digitalWrite(digitalPin, HIGH); // Pone digitalPin a livello alto per caricare
il condensatore
StartTime = millis(); // Pone in Startime il tempo iniziale (valore attuale in
ms misurato dal cronometro di Arduino)

// Fa un ciclo di due letture a vuoto per "scaricare" l'analogPin
for (i = 0; i < 2; i++)
{
  V[i] = analogRead(analogPin);
}

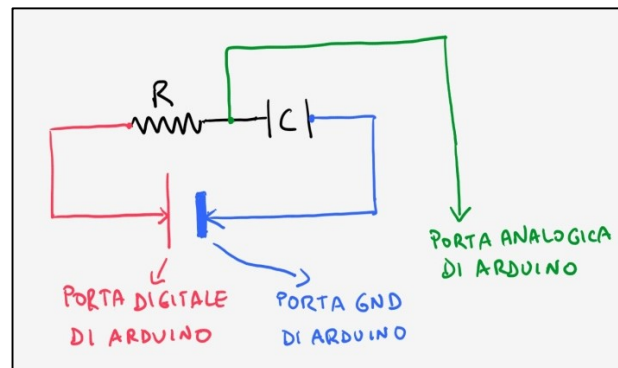
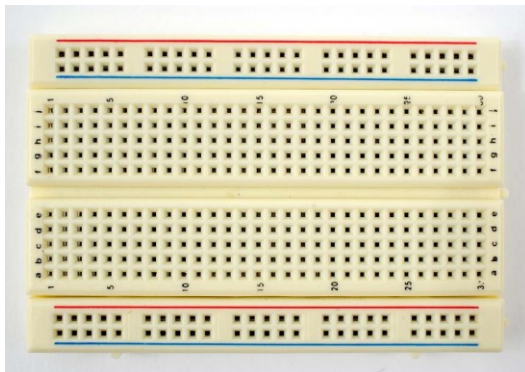
// Ciclo per la carica
for (i = 0; i < M; i++)
{
  t[i] = millis() - StartTime; // Legge il tempo attuale in ms, sottrae lo
StartTime e mette il risultato nel vettore t[i]
  V[i] = analogRead(analogPin); // Legge la tensione e la mette nel vettore V[i]
  delay(delays); // Aspetta delays ms prima della lettura successiva
}
Serial.println("\t Valori fase di CARICA");
// Ciclo per la scrittura su porta seriale dei dati della carica
for (i = 0; i < M; i++)
{
  Serial.print(t[i]/1000.0); // Scrive t[i] in secondi
  Serial.print ( "\t" ); // Mette uno carattere separatore
  Serial.println(V[i] * 5.0 / 1023.0, 3); // Scrive V[i] in Volt con 3 cifre
decimali e va a capo
  delay(1);
}
  delay (2000); //Attende due secondi per completare la carica
  StartTime = millis(); // Pone in Startime il tempo iniziale attuale
  digitalWrite(digitalPin, LOW); // Pone digitalPin a livello basso per
scaricare il condensatore
  // Fa un ciclo di due letture a vuoto per "scaricare" l'analogPin
  for (i = 0; i < 2; i++)
  {
    V[i] = analogRead(analogPin);
  }

// Ciclo per la scarica
for (i = 0; i < M; i++)
{
  t[i] = millis() - StartTime; // Legge il tempo attuale in ms, sottrae lo
StartTime e mette il risultato nel vettore t[i]
  V[i] = analogRead(analogPin); // Legge la tensione e la mette nel vettore V[i]
  delay(delays); // Aspetta delays ms prima della lettura successiva
}
Serial.println("Valori fase di SCARICA");
// Ciclo per la scrittura su porta seriale dei 256 dati della carica
for (i = 0; i < M; i++)
{
  Serial.print(t[i]/1000.0); // Scrive t[i] in secondi
  Serial.print ( "\t" ); // Mette un carattere separatore
  Serial.println(V[i] * 5.0 / 1023.0, 3); // Scrive V[i] in Volt con 3 cifre
decimali e va a capo
  delay(1);
}
  while (1); {
// viene fermato il ciclo Void loop mettendo Arduino in stand-bye
}
}

```

Collegamento del circuito, caricamento dello sketch su Arduino, esecuzione della misura

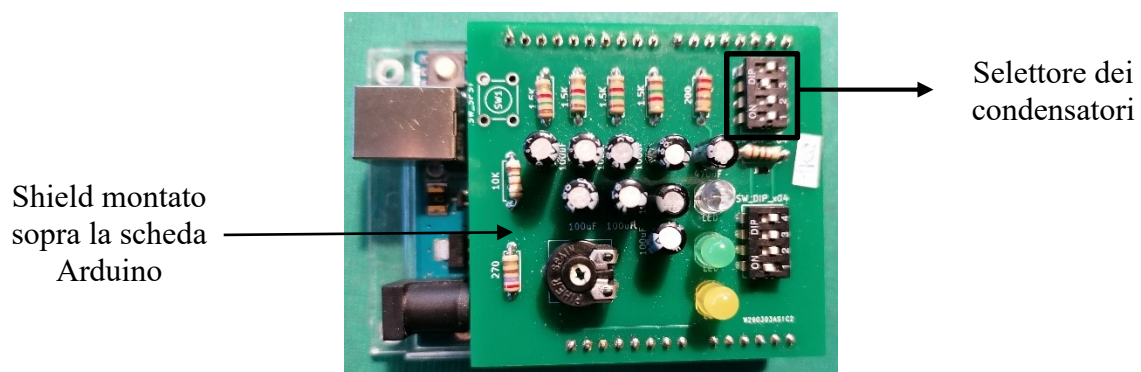
- a) Se viene usata una breadboard vuota e un resistore di resistenza $R = 1,5 \text{ k}\Omega$ e un condensatore di capacità $C = 100 \text{ }\mu\text{F}$, collegare in serie la resistenza R e il condensatore C sulla *breadboard* (figura sotto a sinistra) e realizzare poi il circuito mostrato nella figura sotto a destra effettuando i collegamenti con la scheda Arduino.



La porta digitale (polo positivo) e la porta GND (massa = polo negativo) servono per alimentare il circuito, mentre la porta analogica serve per misurare la tensione (rispetto alla massa) ai capi del condensatore.

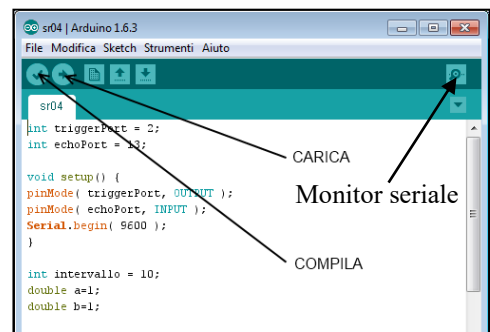
Attenzione il condensatore è polarizzato, cioè ha un polo negativo (evidenziato da una banda nera con scritto "-"), che va collegato al polo negativo dell'alimentatore (in questo caso la porta GND di Arduino)

- b) Se viene usato lo Shield collegato alla scheda Arduino avente le resistenze e i condensatori già predisposti, basta solo cambiare la posizione su ON del selettore dei condensatori posto sul circuito a fianco dei resistori (vedi figura sotto):



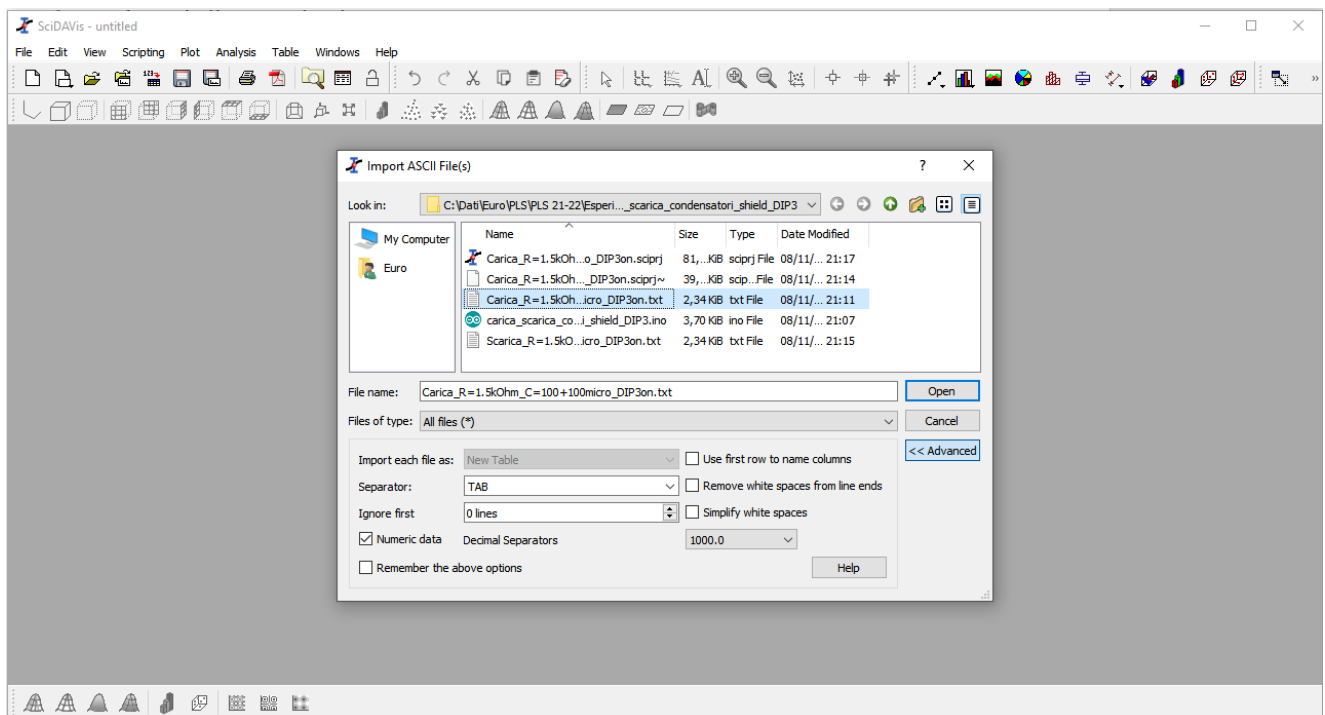
- la posizione n. 4 per selezionare la resistenza $R = 1,5 \text{ k}\Omega$ e un condensatore di capacità $C = 100 \text{ }\mu\text{F}$, alimentati dal PIN digitale n. 11 di Arduino
- la posizione n. 3 per selezionare la resistenza $R = 1,5 \text{ k}\Omega$ e 2 condensatori di capacità $C_1 = C_2 = 100 \text{ }\mu\text{F}$ collegati in parallelo, alimentati dal PIN digitale n. 10 di Arduino ($C = C_1 + C_2 = 200 \text{ }\mu\text{F}$)
- la posizione n. 2 per selezionare la resistenza $R = 1,5 \text{ k}\Omega$ e 2 condensatori di capacità $C_1 = C_2 = 100 \text{ }\mu\text{F}$ collegati in serie, alimentati dal PIN digitale n. 9 di Arduino ($C = (\frac{1}{C_1} + \frac{1}{C_2})^{-1} = 50 \text{ }\mu\text{F}$)
- la posizione n. 1 per selezionare la resistenza $R = 1,5 \text{ k}\Omega$ e 2 condensatori di capacità $C_1 = C_2 = 100 \text{ }\mu\text{F}$ collegati in parallelo e posti in serie ad un terzo condensatore di capacità $C_3 = 100 \text{ }\mu\text{F}$, alimentati dal PIN digitale n. 6 di Arduino ($C = (\frac{1}{C_1 + C_2} + \frac{1}{C_3})^{-1} = 66,7 \text{ }\mu\text{F}$)

Per caricare gli sketch collegare la scheda Arduino alla porta USB del PC e avviare il software *Arduino* che si trova sul *desktop*. Una volta aperto bisogna copiare lo sketch precedente al suo interno, modificarlo opportunamente e poi salvare il file su una cartella. Poi si può verificare la corretta scrittura del codice con il pulsante *compila*; se la compilazione va a buon fine si può caricare il programma sulla scheda con il pulsante *carica* (se non riesce a caricarlo controllare nel menù Strumenti -> Porta COM che sia selezionata la corretta porta seriale). Aprire il monitor seriale per visualizzare i valori della tensione ai capi del condensatore e del tempo, copiarli e incollarli nel Blocco Note di Windows e salvarli in due file di tipo txt, uno per la carica e uno per la scarica.

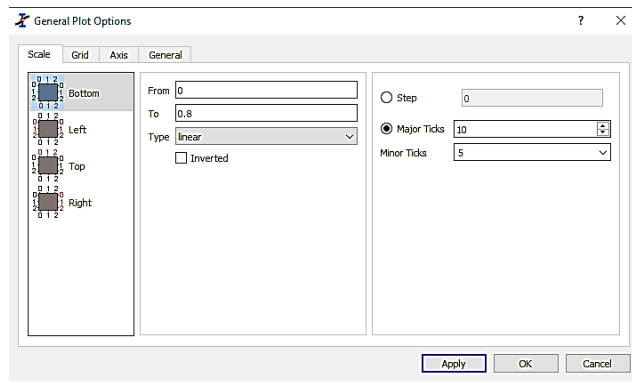


Analisi dei dati sperimentali

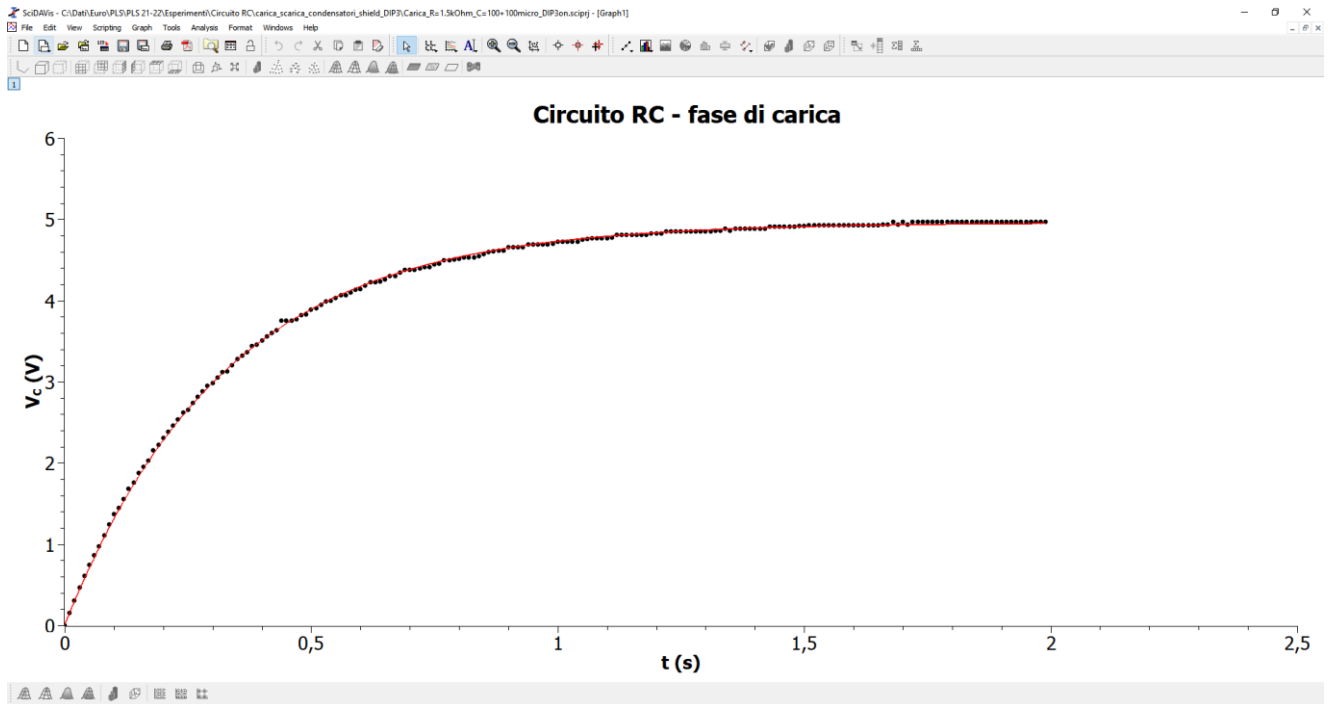
- Aprire il programma *SciDAVis* <http://scidavis.sourceforge.net/>),
- Cancellare la *Table1* (cliccare sulla **X** e poi sul tasto *Delete*)
- Importare il file txt con i dati: menù *File* -> *Import ASCII*, scegliere il file, selezionare le opzioni mostrate nell'immagine seguente (se non compaiono cliccare sul pulsante *Advanced*) e poi cliccare sul pulsante *Open*. Se la prima riga del file txt con i dati non contiene i valori numerici ma un'intestazione nella Sezione "*Ignore first*" selezionare "*1 lines*" invece di "*0 lines*"



- Rinominare le colonne della tabella *Table1*: cliccare sulla colonna 1, poi nella sezione *Description* nel campo *Name* impostare t(s) e cliccare su *Apply*; allo stesso modo rinominare Vc(V) la colonna 2.
- Controllare che nella sezione *Type* il campo *Type* sia di tipo *Numeric*, il campo *Format* sia settato a *Decimal* e poi impostare il corretto numero di cifre decimali (campo *Decimal Digits*).
- Per costruire il grafico tensione-tempo Vc(t) selezionare con il mouse tutti i valori della prima e seconda colonna, poi cliccare sul menù *Plot* -> *Scatter*.
- Inserire il titolo del grafico e degli assi (basta fare doppio click col mouse..), cancellare la legenda, impostare a zero il valore iniziale della scala degli assi x e y (menù *Format* -> *Axes* -> *Scale* -> *Bottom*, scrivere 0 nella casella *From* e cliccare sul pulsante *Apply*, poi cliccare su *Left* per fare la stessa operazione sull'asse y) (vedi figure seguenti). Nel menù *Format* -> *Axes* -> *Axis* potete cambiare anche i *Font* degli assi (tipo, dimensione,.. del carattere)



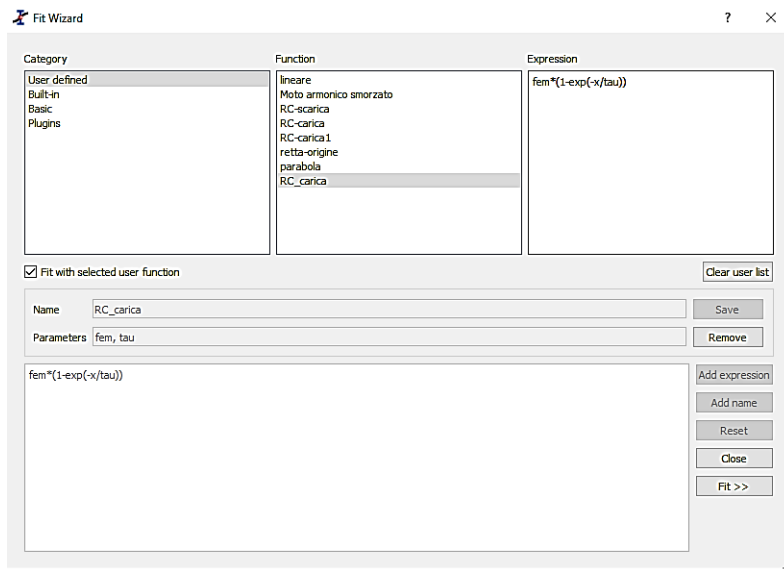
- Apparirà un grafico simile a quello sottostante



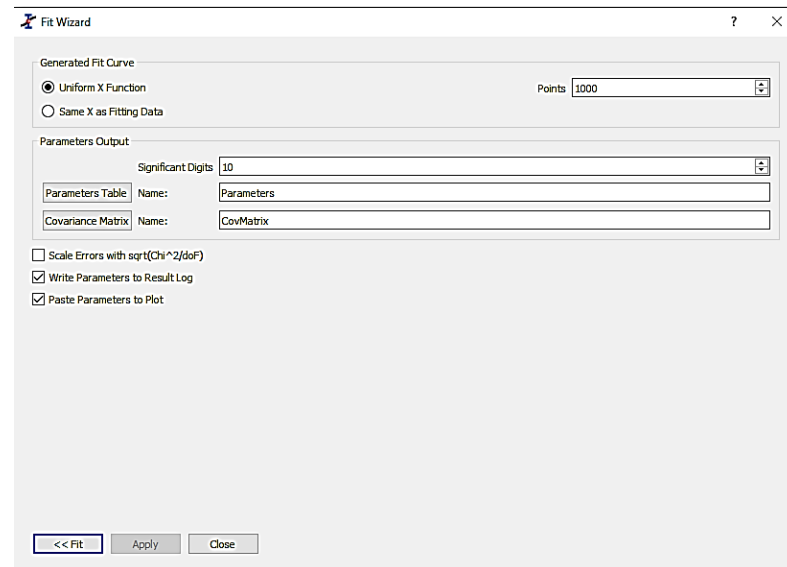
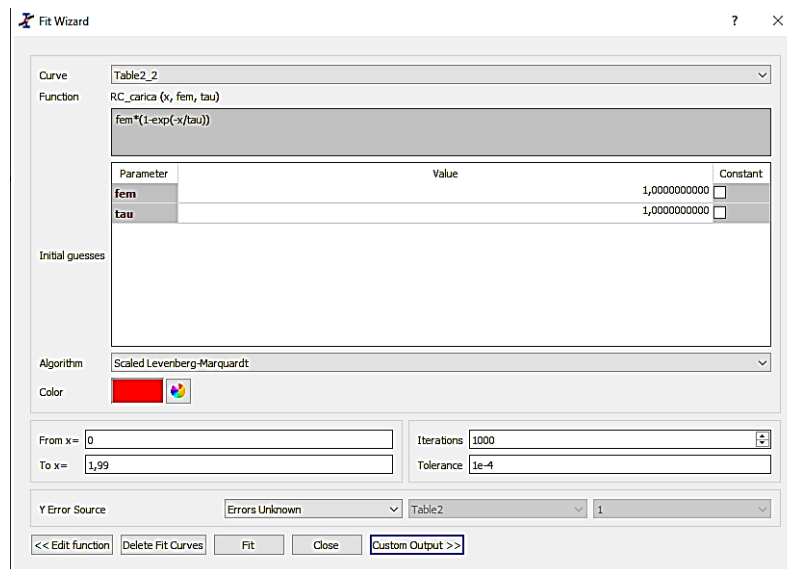
- Fare il fit dei dati, cioè ottenere l'equazione della curva teorica che meglio si adatta ai dati sperimentali, nel caso della fase di carica:

$$V_C(t) = \frac{q(t)}{C} = f_{em} \left(1 - e^{-t/\tau}\right)$$

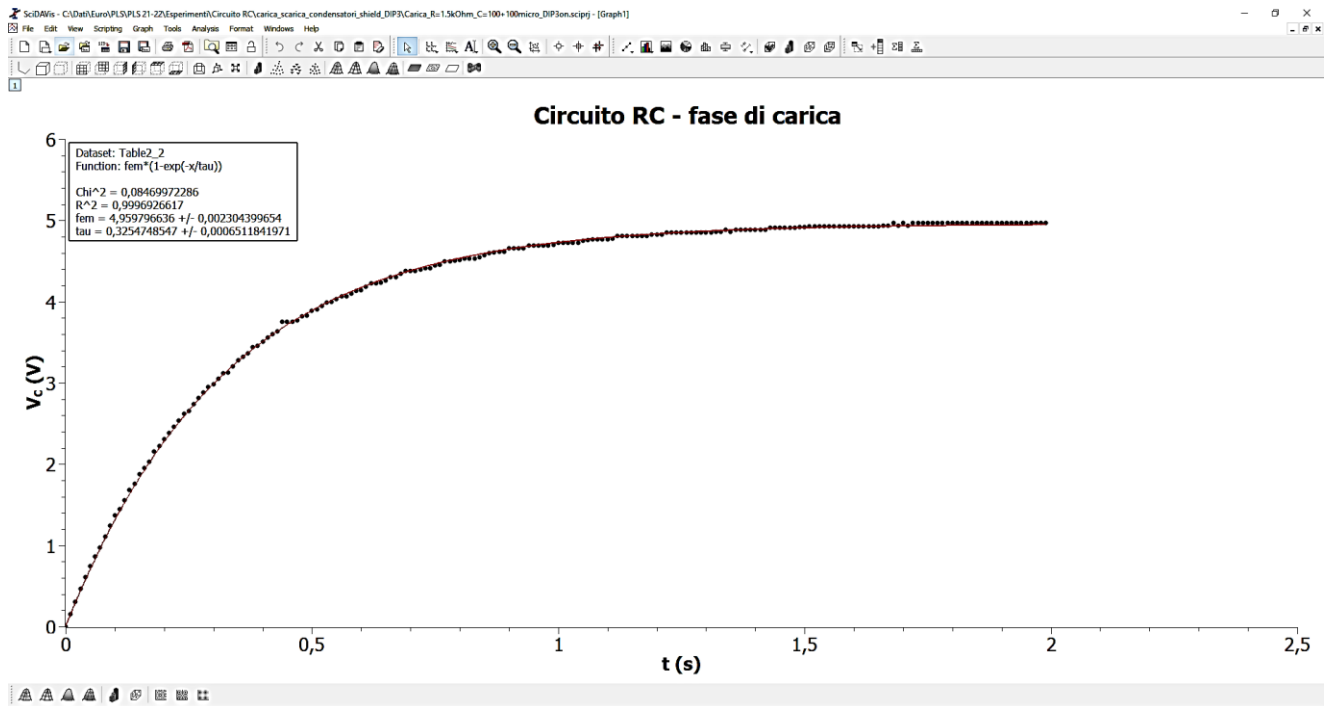
- Selezionare il menu *Analysis* -> *Fit Wizard* -> *Category* -> *User defined*, nella casella *Name* scegliere un nome per la funzione di fit (nel nostro caso *RC_carica*), nella casella *Parameters* scegliere un nome per i parametri di fit (nel nostro caso *fem* e *tau*), nello spazio bianco sottostante scrivere l'equazione della funzione, per esempio -> $fem*(1-\exp(-x/tau))$ per la carica del condensatore, e cliccare sul pulsante *Save*, spuntare il flag *Fit whit selected user function* e infine premere il pulsante *Fit* (vedi figura seguente).



- Premere il tasto *Custom Output* e accertarsi che siano selezionate le caselle “*Write Parameters to Result Log*” e “*Paste Parameters to Plot*” e poi cliccare sul pulsante *Fit* per tornare alla finestra precedente.



- Inserire un valore di partenza “ragionevole dal punto di vista fisico” per i parametri della funzione di fit: nel nostro caso va bene lasciare il valore di default **1** per entrambi, poi premere più volte il tasto *Fit* finché i valori dei parametri non cambiano più e infine premere il tasto *Close*.
- Sul grafico appariranno in rosso la curva teorica che meglio si adatta ai dati sperimentali fine e una finestra con i valori dei parametri di fit con le relative incertezze; fare doppio clic sulla casella e poi su *Font* per aumentare la dimensione del carattere. Nell’esempio mostrato sotto



- Salvare il file del progetto (menù *File* -> *Save Project As*).
- Esportare il file con l’immagine (in formato jpeg) del grafico (menù *File* -> *Export Graph* -> *Current*).
- Ripetere le stesse operazioni per costruire il grafico tensione-tempo $V_C(t)$ e fare il fit dei dati della fase di scarica:

$$V_C(t) = \frac{q(t)}{C} = f_{em} \cdot e^{-t/\tau}$$

Conclusioni

- Assumere come valore sperimentale ottenuto per la costante di tempo tau (τ_{sp}) la media aritmetica tra i valori ottenuti dal fit per la carica e la scarica, e confrontarlo col valore teorico ottenuto dalla formula $\tau = RC$ per i 4 circuiti selezionati.