

Caratteristica del diodo utilizzando dati sperimentali

Marcello Colozzo - <http://www.extrabyte.info>

Introduzione

Di seguito un'esperienza di laboratorio per la determinazione della caratteristica tensione-corrente di un diodo al silicio.

Plotting dei dati sperimentali

```
SetOptions[
{
  Plot,
  ListLinePlot, ParametricPlot, ListPlot,
  ListLinePlot
},
AxesStyle -> Directive[
  Hue[5 / 6],
  8
],
FrameStyle -> Directive[
  Hue[5 / 6],
  8
]
];
```

Siamo tentati ad utilizzare l'istruzione **Import**, ma così facendo non otteniamo un vettore (lista):

```
dati = Import["diodo.txt"]
```

```
0.112 0.536
0.087 0.524
0.158 0.550
0.222 0.565
0.296 0.576
0.348 0.585
0.742 0.620
0.957 0.630
1.420 0.650
2.350 0.673
4.820 0.708
8.010 0.734
```

Utilizziamo **ReadList**

```
? ReadList
```

`ReadList["file"]` reads all the remaining expressions in a file, and returns a list of them.

`ReadList["file", type]` reads objects of the specified type

from a file, until the end of the file is reached. The list of objects read is returned.

`ReadList["file", {type1, type2, ...}]` reads objects with a sequence of types, until the end of the file is reached.

`ReadList["file", types, n]` reads only the first *n* objects of the specified types. >>

```

lista = ReadList["diodo.txt"]

{0.060032, 0.045588, 0.0869, 0.12543, 0.170496,
 0.20358, 0.46004, 0.60291, 0.923, 1.58155, 3.41256, 5.87934}

Clear[lista]

```

Però bisogna specificare il tipo di oggetti:

```

lista = ReadList[
  (*nomefile. Attenzione: bisogna specificare la directory*)
  "diodo.txt",
  (*tipo di oggetti*)
  Number
]

{0.112, 0.536, 0.087, 0.524, 0.158, 0.55, 0.222, 0.565, 0.296, 0.576, 0.348, 0.585,
 0.742, 0.62, 0.957, 0.63, 1.42, 0.65, 2.35, 0.673, 4.82, 0.708, 8.01, 0.734}

```

Si potrebbe utilizzare la specifica **Real** (*Mathematica* tratta i numeri approssimati in maniera simile al FORTRAN):

```

Clear[lista]

lista = ReadList[
  (*nomefile. Attenzione: bisogna specificare la directory*)
  "diodo.txt",
  (*tipo di oggetti*)
  Real
]

{0.112, 0.536, 0.087, 0.524, 0.158, 0.55, 0.222, 0.565, 0.296, 0.576, 0.348, 0.585,
 0.742, 0.62, 0.957, 0.63, 1.42, 0.65, 2.35, 0.673, 4.82, 0.708, 8.01, 0.734}

```

Tuttavia, *Mathematica* tratta una notazione puntata automaticamente come un reale. Ad esempio se scriviamo **2**, *Mathematica* memorizza l'intero 2 (**Integer**). Ma se scriviamo **2.**, *Mathematica* lo memorizza come **Real**.

Importati i dati, dobbiamo disporli nel giusto ordine per poter plottare la caratteristica tensione-corrente (la tensione in ascissa, e la corrente in ordinate). Generiamo prima delle coppie ordinate:

```

Clear[lista]

lista = ReadList[
  "diodo.txt",
  {Number, Number}
]

{{0.112, 0.536}, {0.087, 0.524}, {0.158, 0.55}, {0.222, 0.565},
 {0.296, 0.576}, {0.348, 0.585}, {0.742, 0.62}, {0.957, 0.63},
 {1.42, 0.65}, {2.35, 0.673}, {4.82, 0.708}, {8.01, 0.734}}

```

Ma non è l'ordine giusto perché in ascissa deve esserci la d.d.p. ai capi del diodo. Ad esempio, la prima coppia deve essere **{{ 0.536, 0.112}}**. A questo punto, ricordiamo che per *Mathematica* una lista è fondamentalmente una matrice (v. file http://www.extrabyte.info/manuale_matematica001.pdf). Infatti:

```
lista // MatrixForm
```

$$\begin{pmatrix} 0.112 & 0.536 \\ 0.087 & 0.524 \\ 0.158 & 0.55 \\ 0.222 & 0.565 \\ 0.296 & 0.576 \\ 0.348 & 0.585 \\ 0.742 & 0.62 \\ 0.957 & 0.63 \\ 1.42 & 0.65 \\ 2.35 & 0.673 \\ 4.82 & 0.708 \\ 8.01 & 0.734 \end{pmatrix}$$

Calcoliamo allora la matrice trasposta

```
lista1 = lista // Transpose
```

```
{ {0.112, 0.087, 0.158, 0.222, 0.296, 0.348, 0.742, 0.957, 1.42, 2.35, 4.82, 8.01},  
  {0.536, 0.524, 0.55, 0.565, 0.576, 0.585, 0.62, 0.63, 0.65, 0.673, 0.708, 0.734} }
```

Vediamo sotto forma di matrice:

```
lista1 // MatrixForm
```

$$\begin{pmatrix} 0.112 & 0.087 & 0.158 & 0.222 & 0.296 & 0.348 & 0.742 & 0.957 & 1.42 & 2.35 & 4.82 & 8.01 \\ 0.536 & 0.524 & 0.55 & 0.565 & 0.576 & 0.585 & 0.62 & 0.63 & 0.65 & 0.673 & 0.708 & 0.734 \end{pmatrix}$$

che è appunto la matrice trasposta di quella iniziale. Però lasciamo nel formato lista:

```
lista1
```

```
{ {0.112, 0.087, 0.158, 0.222, 0.296, 0.348, 0.742, 0.957, 1.42, 2.35, 4.82, 8.01},  
  {0.536, 0.524, 0.55, 0.565, 0.576, 0.585, 0.62, 0.63, 0.65, 0.673, 0.708, 0.734} }
```

Permutiamo la seconda riga con la prima:

```
lista2 = lista1 // Reverse
```

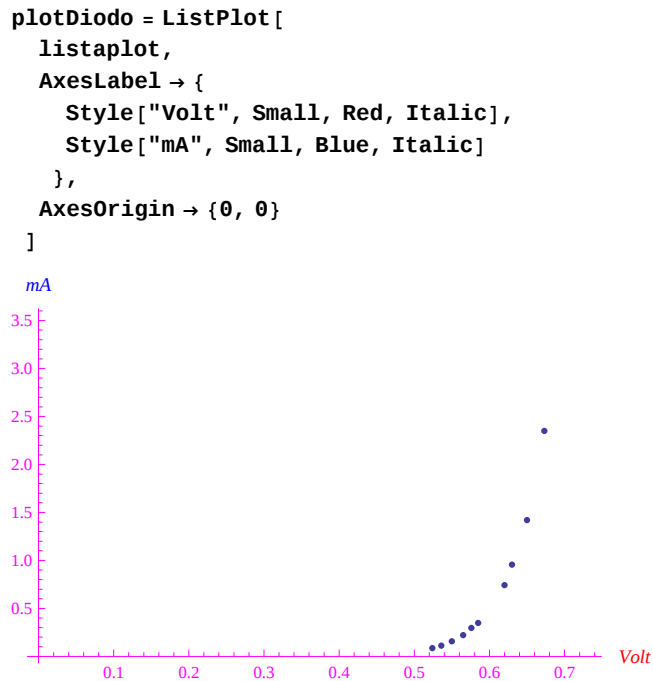
```
{ {0.536, 0.524, 0.55, 0.565, 0.576, 0.585, 0.62, 0.63, 0.65, 0.673, 0.708, 0.734},  
  {0.112, 0.087, 0.158, 0.222, 0.296, 0.348, 0.742, 0.957, 1.42, 2.35, 4.82, 8.01} }
```

Vediamo che ora i valori della d.d.p. compaiono nell'ordine giusto. Il problema consiste nell'accoppiarli con quelli della corrente (seconda riga). Di nuovo, facciamo la trasposta:

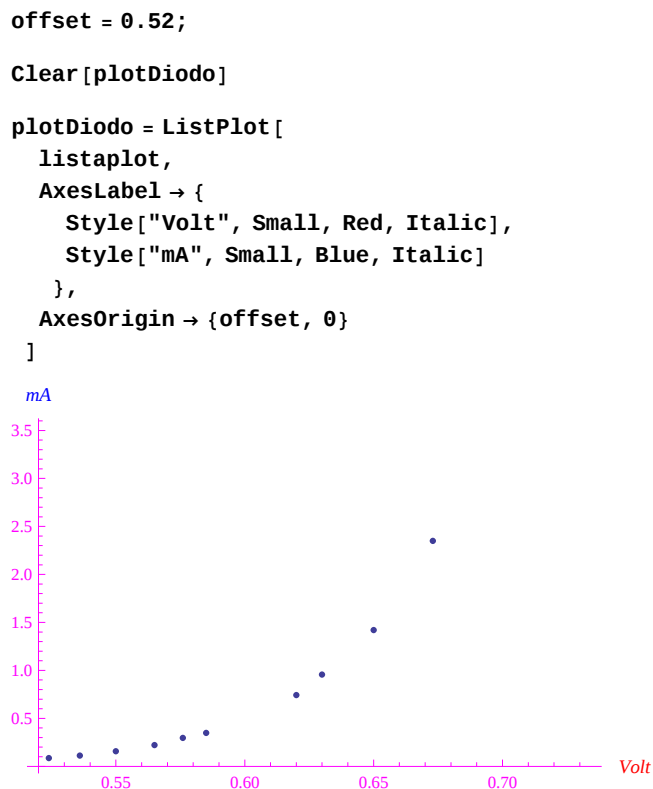
```
listaplot = lista2 // Transpose
```

```
{ {0.536, 0.112}, {0.524, 0.087}, {0.55, 0.158}, {0.565, 0.222},  
  {0.576, 0.296}, {0.585, 0.348}, {0.62, 0.742}, {0.63, 0.957},  
  {0.65, 1.42}, {0.673, 2.35}, {0.708, 4.82}, {0.734, 8.01} }
```

Ci siamo! Non dobbiamo fare altro che plottare i dati:



Dal grafico vediamo una tensione di offset dell'ordine di 0.52 V, per cui resettiamo l'origine degli assi coordinati:



Fitting dei dati

Separiamo le righe della matrice `listaplot`

```
{tensione, corrente} = listaplot // Transpose
{{0.536, 0.524, 0.55, 0.565, 0.576, 0.585, 0.62, 0.63, 0.65, 0.673, 0.708, 0.734},
 {0.112, 0.087, 0.158, 0.222, 0.296, 0.348, 0.742, 0.957, 1.42, 2.35, 4.82, 8.01}}
```

tensione

```
{0.536, 0.524, 0.55, 0.565, 0.576, 0.585, 0.62, 0.63, 0.65, 0.673, 0.708, 0.734}
```

corrente

```
{0.112, 0.087, 0.158, 0.222, 0.296, 0.348, 0.742, 0.957, 1.42, 2.35, 4.82, 8.01}
```

Utilizziamo la relazione $i = i_0 e^{\alpha V}$, dove i_0 è la corrente di saturazione inversa, mentre α è una costante positiva. Dobbiamo prendere il logaritmo dell'intensità di corrente. La funzione **Log** ha l'attributo **Listable** ovvero agisce sulle liste:

```
listalog = {tensione, Log[corrente]}
```

```
{{0.536, 0.524, 0.55, 0.565, 0.576, 0.585, 0.62, 0.63, 0.65, 0.673, 0.708, 0.734},
 {-2.18926, -2.44185, -1.84516, -1.50508, -1.2174, -1.05555,
 -0.298406, -0.0439519, 0.350657, 0.854415, 1.57277, 2.08069}}
```

```
listalog // MatrixForm
```

```
( 0.536    0.524    0.55    0.565    0.576    0.585    0.62    0.63    0.65
 -2.18926 -2.44185 -1.84516 -1.50508 -1.2174 -1.05555 -0.298406 -0.0439519 0.350657
```

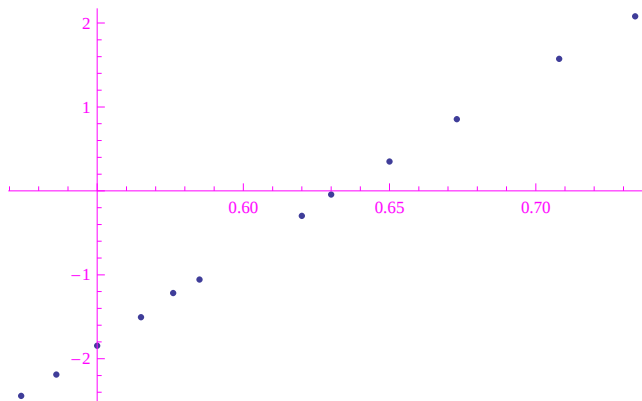
Ordiniamo a coppie

```
listalog1 = listalog // Transpose
```

```
{{0.536, -2.18926}, {0.524, -2.44185}, {0.55, -1.84516}, {0.565, -1.50508},
 {0.576, -1.2174}, {0.585, -1.05555}, {0.62, -0.298406}, {0.63, -0.0439519},
 {0.65, 0.350657}, {0.673, 0.854415}, {0.708, 1.57277}, {0.734, 2.08069}}
```

Proviamo a plottare

```
ListPlot[
 listalog1
]
```



che è l'andamento corretto, giacché la relazione tra $\ln(i)$ e V è lineare. Dobbiamo perciò eseguire il fitting di questi dati. Rammentiamo la sintassi del comando

?Fit

```
Fit[data, funs, vars] finds a least-squares fit to
a list of data as a linear combination of the functions funs of variables vars. >>
```

Per quanto precede, dobbiamo chiedere a *Mathematica* di fittare i dati in scala semilogaritmica attraverso una funzione lineare. Quindi:

```

f[V_] = Fit[
  (*dati da fittare*)
  listalog1,
  (*funzione di fitting*)
  {1, V},
  (*variabile indipendente*)
  V
]
-13.7368 + 21.6437 V

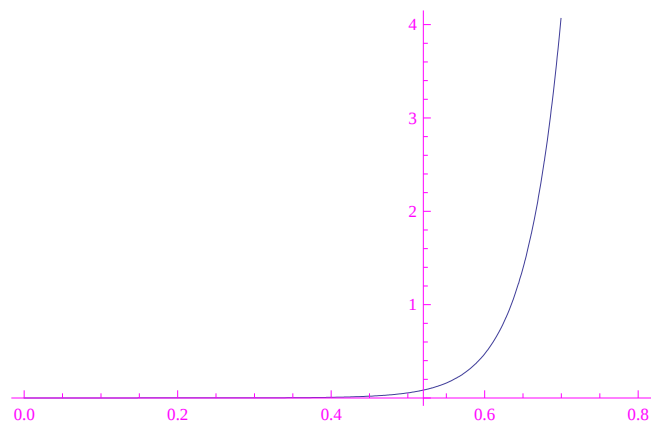
```

Cioè $i(V) = e^{-13.7368} e^{21.6437 V}$

```

plotFit = Plot[
  Exp[f[V]],
  {V, 0, 0.8},
  AxesOrigin -> {offset, 0}
]

```



```
Show[plotDiodo, plotFit]
```

