PP1

March 31, 2019

1 Esperienza n. 1

```
In [1]: %run 'Base_Load.ipynb'
```

```
In [2]: def multimetro_sd(x):
            if(isinstance(x, int)):
                s = str(x)
                ret = 1
            else:
                s = str(x)
                for k in range(len(s)):
                    if (s[k] == '.'):
                        a = k
                ret = 4*10**(-len(s[a+1:]))
            sd= 0.5/100*x + ret
            return (sd)


        def multimeasure_multimetro_sd(x): # BISOGNA DARE UNA LISTA, NON UN ARRAY
            sd = []
            for i in range(len(x)):
                ret = multimetro_sd(x[i])
                sd.append(ret)
            sd = array(sd)
            return (sd)
```

```
In [3]: #Multimetro METEX M-4650
```

1.1 Legge di Malus

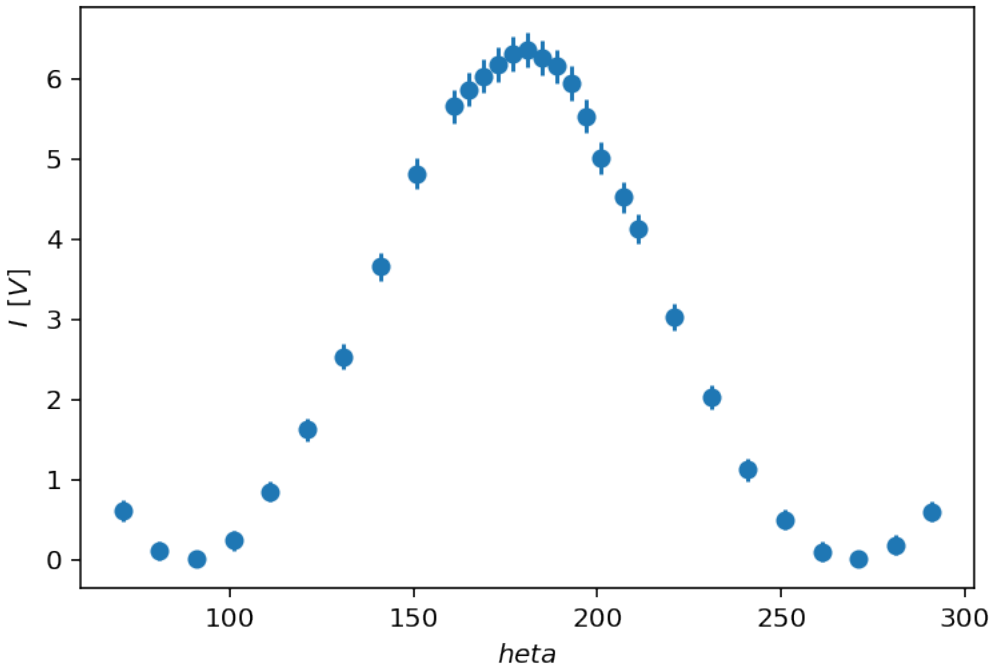
1.1.1 Parte 1 - due polaroid

```
In [4]: b = 7.6e-3#Segnale di buio/fondo
        sdb = multimetro_sd(b)
        thetha_est = 49 #Angolo di riferimento - Estinzione
        theta0 = thetha_est-90 #Angolo di riferimento - 0
        sdtheta0 = 1/sqrt(12)
```

```
I = [0.62, 0.12, 0.022, 0.25, 0.86, 1.63, 2.54, 3.66, 4.82, 5.66, 5.87, 6.04, 6.18, 6.3
theta = array([30, 40, 50, 60, 70, 80, 90, 100, 110, 120, 124, 128 ,132, 136, 140, 144

sdI = multimeasure_multimetro_sd(I)*3
I = asarray(I)
I -= b #Sottraggo il segnale di buio

sdtheta = 1/sqrt(12)
sdtheta = sqrt(sdtheta**2 + sdtheta0**2)

#data = pd.DataFrame(np.column_stack([theta,I,sdtheta*theta/theta,sdI]))
#print(tabulate(data))
```

In [5]:
```
figure(1)
xlabel("$\theta$")
ylabel("$I\,\,\,[V]$")
errorbar(theta, I, yerr = sdI, xerr=sdtheta, fmt='o')
```

Out[5]: <Container object of 3 artists>



Fit da Gnuplot

In [6]:
```
FIT:    data read from "malus.rtf" u (($1)/180*pi):2:(($3)/180*pi):4 xyerrors
        format = x:z:sx:s
        x range restricted to [1.00000 : 5.50000]
```

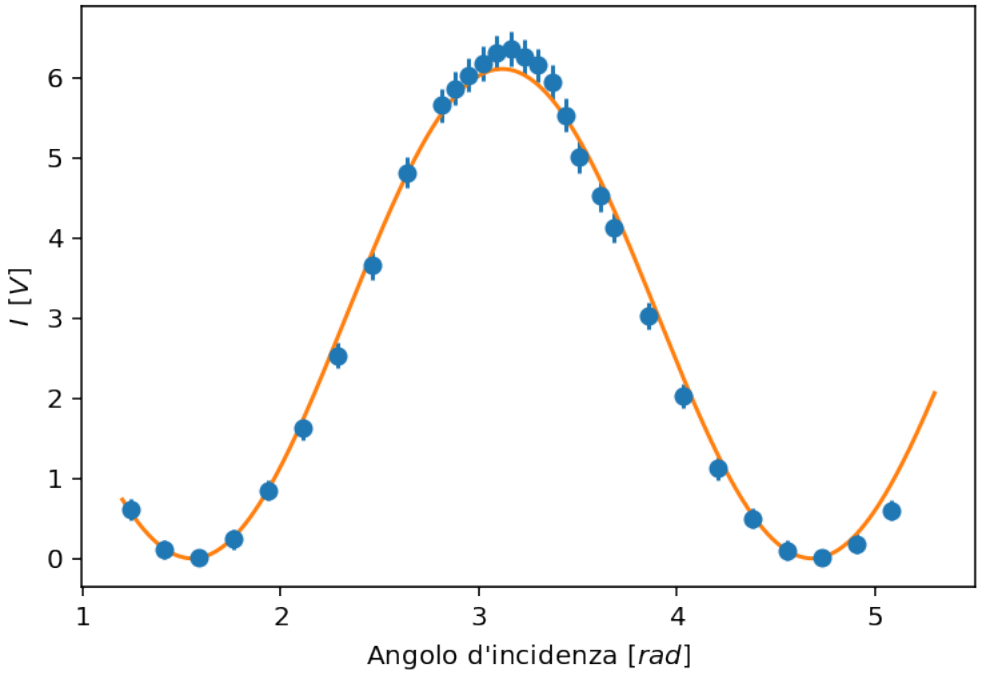
```
        #datapoints = 30
function used for fitting: f(x)
        f(x)=A*((cos(B*x+C))**2)
fitted parameters initialized with current variable values

iter      chisq       delta/lim  lambda   A              B              C
   0 2.7715056171e+01   0.00e+00  7.11e+01    6.107505e+00   1.003870e+00   3.153691e+0
   1 2.7715056171e+01   0.00e+00  7.11e+06    6.107505e+00   1.003870e+00   3.153691e+0

After 1 iterations the fit converged.
final sum of squares of residuals : 27.7151
rel. change during last iteration : 0

degrees of freedom    (FIT_NDF)                        : 27
rms of residuals      (FIT_STDFIT) = sqrt(WSSR/ndf)    : 1.01316
variance of residuals (reduced chisquare) = WSSR/ndf   : 1.02648
p-value of the Chisq distribution (FIT_P)              : 0.425775

Final set of parameters            Asymptotic Standard Error
=======================            ==========================
A               = 6.1075           +/- 0.05848      (0.9575%)
B               = 1.00387          +/- 0.006215     (0.6191%)
C               = 3.15369          +/- 0.02159      (0.6846%)

correlation matrix of the fit parameters:
                A      B      C
A               1.000
B               0.224  1.000
C              -0.193 -0.943  1.000


  File "<ipython-input-6-7de192ae066e>", line 1
FIT:    data read from "malus.rtf" u (($1)/180*pi):2:(($3)/180*pi):4 xyerrors
                ^
SyntaxError: invalid syntax
```

In [7]:
```
A = 6.1075
B = 1.00387
C = 3.15369

def f(x):
    return A*((cos(B*x+C))**2)
```

In [8]:
```
x=linspace(1.2,5.3,1000)
xlabel("Angolo d'incidenza $[rad]$")
ylabel("$I\,\,\,[V]$")
```

```
errorbar(radians(theta), I, yerr = sdI, xerr=radians(sdtheta), fmt='o')
plot(x,f(x),'-')
```

Out[8]: [<matplotlib.lines.Line2D at 0x10e718668>]

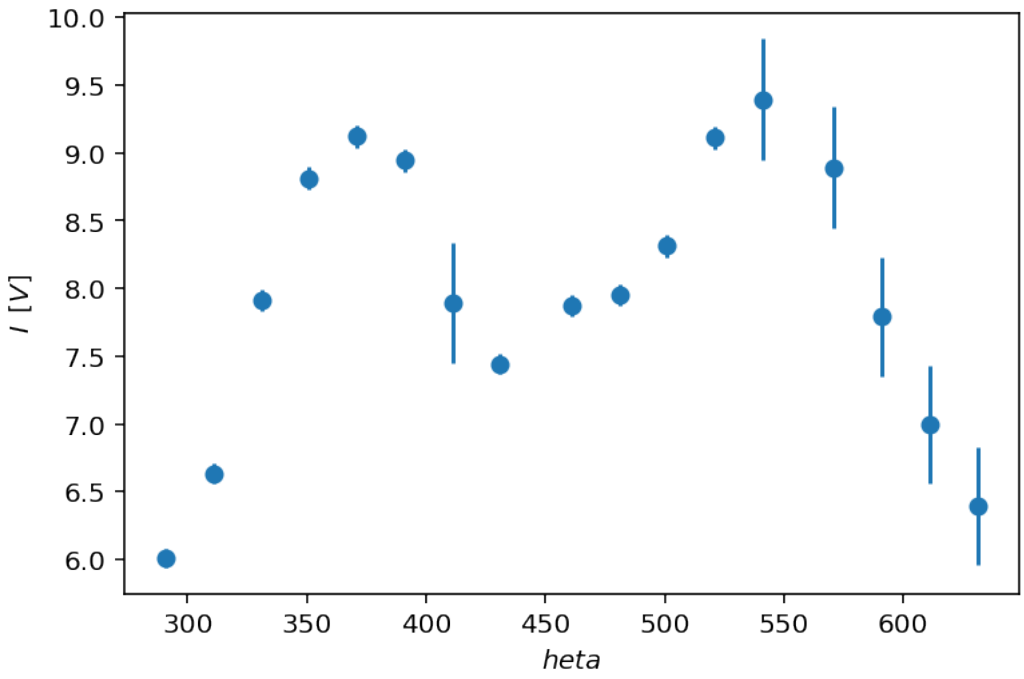


Singolo Polaroid

In [9]:
```
I = [6.02, 6.64, 7.92, 8.82, 9.13, 8.95, 7.90, 7.45, 7.88, 7.96, 8.32, 9.12, 9.40, 8.90
theta = array([250, 270, 290, 310, 330, 350, 360+10, 360+30, 360+60, 360+80, 360+100, 3

sdI = multimeasure_multimetro_sd(I)
I = asarray(I)
I -= b #Sottraggo il segnale di buio
```

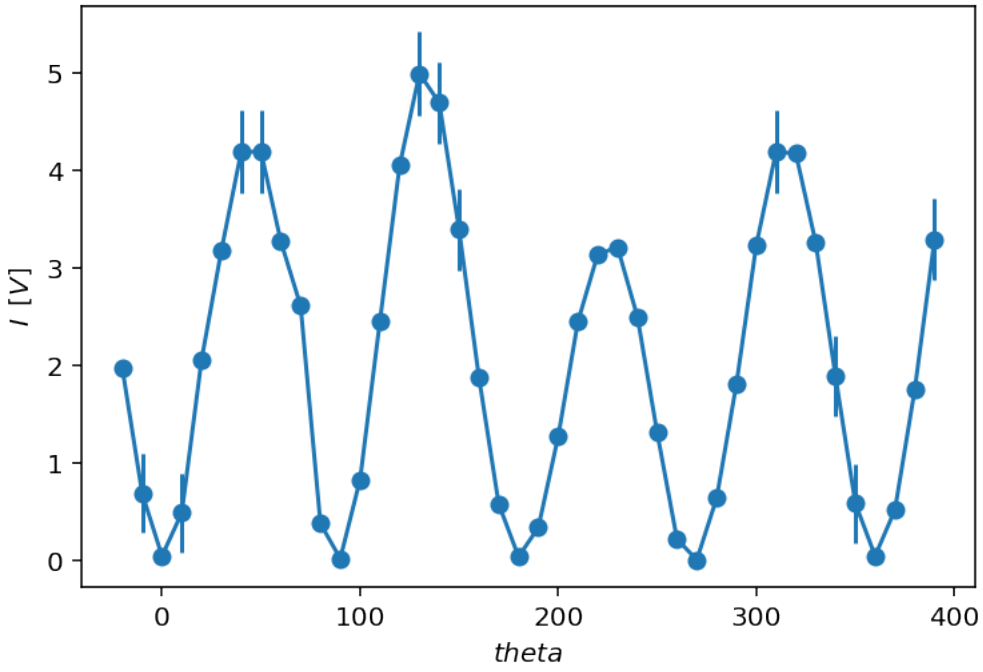
In [11]:
```
figure(1)
xlabel("$\theta$")
ylabel("$I\,\,\,[V]$")
errorbar(theta, I, yerr = sdI, xerr=sdtheta, fmt='o')
```

Out[11]: <Container object of 3 artists>



1.1.2 Parte 2 - tre polaroid

```
In [12]: thetha_est = 140 #Angolo di riferimento - Estinzione
         theta0 = thetha_est #Angolo di riferimento - 0
         sdtheta0 = 1

         #0.010, 0.124, 0.41, 0.73, 0.89, 0.86, 0.65, 0.35, 0.104, 0.022, 0.146, 0.439, 0.75,
         #228, 238, 248, 260, 270, 280, 290, 300, 310, 320, 330, 340, 350, 10, 20, 30, 40, 50,
         I = [1.98, 0.70, 0.058, 0.50, 2.07, 3.19, 4.2, 4.2, 3.28, 2.62, 0.39, 0.031, 0.84, 2.4
         theta = array([120, 130, 140, 150, 160, 170, 180, 190, 200, 210, 220, 230, 240, 250, 1

         sdI = multimeasure_multimetro_sd(I)
         I = asarray(I)
         I -= b #Sottraggo il segnale di buio

         sdtheta = 1
         sdtheta = sqrt(sdtheta**2 + sdtheta0**2)

In [13]: figure(1)
         xlabel("$theta$")
         ylabel("$I\,\,\,[V]$")
         errorbar(theta, I, yerr = sdI, fmt='o-')

Out[13]: <Container object of 3 artists>
```

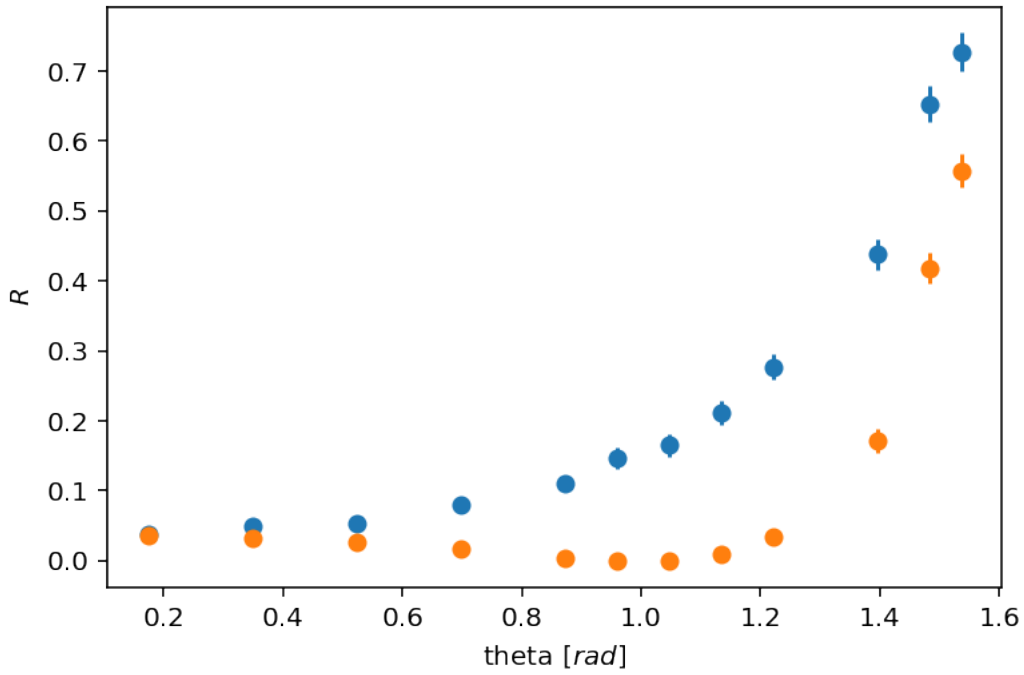


1.2 Angolo di Brewster

```
In [14]: theta_p0 = 52 #Annullameto TM
         theta_s0 = 322 #Annullameto TE

         theta0 = 80 #Riferimento angolo del Plexiglass
         sdtheta0 = 1/sqrt(12)

         Is0 = 9.23-b #Intensità TM senza Plexiglass OLD: 8.27
         Ip0 = 8.87-b #Intensità TE senza Plexiglass OLD: 5.23
         sdIp0 = multimetro_sd(Ip0)
         sdIs0 = multimetro_sd(Is0)

In [15]: Is = [0.358, 0.456, 0.502, 0.748, 1.028, 1.36, 1.53, 1.96, 2.56, 4.05, 6.03, 6.71] #T
         Ip = [0.328, 0.297, 0.242, 0.153, 0.043, 7.7e-3, 9.6e-3, 93e-3, 0.302, 1.52, 3.71, 4.9
         theta = -array([70, 60, 50, 40, 30, 25, 20, 15, 10, 0, -5, -8 ]) + theta0 #Angolo del

         sdtheta = 1/sqrt(12)*theta/theta
         sdtheta = sqrt(sdtheta**2 + sdtheta0**2)

         sdIp = multimeasure_multimetro_sd(Ip)*3
         Ip = asarray(Ip)
         Ip -= b #Sottraggo il segnale di buio

         Rp = Ip/Ip0
```

```
sdRp = Rp*(sdIp/Ip + sdIp0/Ip0)

sdIs = multimeasure_multimetro_sd(Is)*3
Is = asarray(Is)
Is -= b #Sottraggo il segnale di buio

Rs = Is/Is0
sdRs = Rs*(sdIs/Is + sdIs0/Is0)

figure(1)
xlabel("theta $[rad]$")
ylabel("$R$")
errorbar(radians(theta), Rs, yerr = sdRs, fmt='o')
errorbar(radians(theta), Rp, yerr = sdRp, fmt='o')

#data = pd.DataFrame(np.column_stack([theta,Rs,Rp,sdtheta,sdRs,sdRp]))
#print(tabulate(data))
```

Out[15]: <Container object of 3 artists>



In [16]:
```
def fs(x,n):
    return abs((cos(x)-sqrt(n**2-(sin(x))**2))/(cos(x)+sqrt(n**2-(sin(x))**2)))**2

def fp(x,n):
    return abs((-n**2*cos(x)+sqrt(n**2-(sin(x))**2))/(n**2*cos(x)+sqrt(n**2-(sin(x))**
```

Fit TE

```
In [ ]: FIT:    data read from "brewster.rtf" u (($1)/180*pi):2:(($4)/180*pi):5 xyerror
                format = x:z:sx:s
                x range restricted to [0.00000 : 1.60000]
                #datapoints = 12
        function used for fitting: fs(x)
                fs(x)=abs((cos(x)-sqrt(n**2-(sin(x))**2))/(cos(x)+sqrt(n**2-(sin(x))**2)))**2
        fitted parameters initialized with current variable values

        iter      chisq       delta/lim  lambda   n
           0 4.7198517783e+01   0.00e+00  6.62e+01    1.487689e+00
           1 4.7198515431e+01  -4.98e-03  6.62e+00    1.487689e+00

        After 1 iterations the fit converged.
        final sum of squares of residuals : 47.1985
        rel. change during last iteration : -4.98301e-08

        degrees of freedom    (FIT_NDF)                        : 11
        rms of residuals      (FIT_STDFIT) = sqrt(WSSR/ndf)    : 2.07142
        variance of residuals (reduced chisquare) = WSSR/ndf   : 4.29077
        p-value of the Chisq distribution (FIT_P)              : 1.98283e-06

        Final set of parameters            Asymptotic Standard Error
        =======================            ==========================
        n               = 1.48769          +/- 0.01343      (0.9029%)
```

```
In [17]: ns = 1.48769
          sdns = 0.01343
```

Fit TM

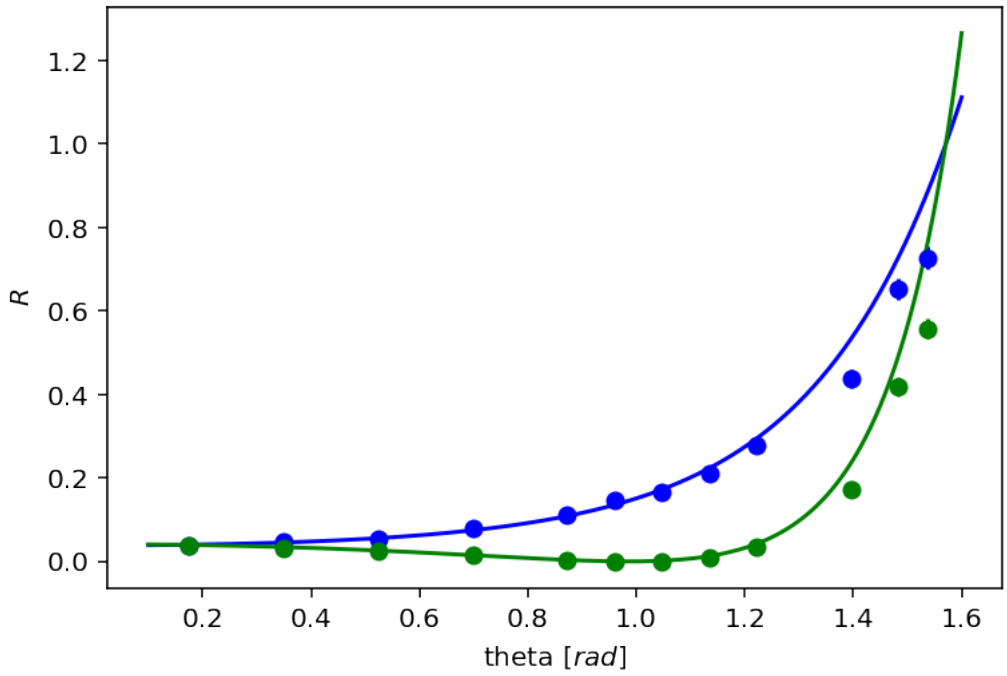
```
In [ ]: FIT:    data read from "brewster.rtf" u (($1)/180*pi):3:(($4)/180*pi):6 xyerror
                format = x:z:sx:s
                x range restricted to [0.00000 : 1.60000]
                #datapoints = 12
        function used for fitting: fp(x)
                fp(x)=abs((-n**2*cos(x)+sqrt(n**2-(sin(x))**2))/(n**2*cos(x)+sqrt(n**2-(sin(x)
        fitted parameters initialized with current variable values

        iter      chisq       delta/lim  lambda   n
           0 5.5604540642e+01   0.00e+00  4.65e+01    1.487689e+00
           2 5.0566459617e+01   0.00e+00  4.65e+08    1.509099e+00

        After 2 iterations the fit converged.
        final sum of squares of residuals : 50.5665
        rel. change during last iteration : 0

        degrees of freedom    (FIT_NDF)                        : 11
```

```
rms of residuals      (FIT_STDFIT) = sqrt(WSSR/ndf)     : 2.14405
variance of residuals (reduced chisquare) = WSSR/ndf    : 4.59695
p-value of the Chisq distribution (FIT_P)               : 4.94976e-07

Final set of parameters            Asymptotic Standard Error
=======================            ==========================
n                = 1.5091          +/- 0.01912      (1.267%)
```

In [18]:
```
np = 1.5091
sdnp = 0.01912
```

In [19]:
```
x=linspace(0.1,1.6,1000)
xlabel("theta $[rad]$")
ylabel("$R$")
errorbar(radians(theta), Rs, yerr = sdRs, fmt='bo')
errorbar(radians(theta), Rp, yerr = sdRp, fmt='go')
plot(x,fs(x,ns),'b-')
plot(x,fp(x,np),'g-')
```

Out[19]: [<matplotlib.lines.Line2D at 0x1515dea7b8>]



1.2.1 Visibilità

In [22]:
```
V = (Rs-Rp)/(Rs+Rp)
sd = sqrt(sdRs**2 + sdRp**2)
```

```
sdV = V*(sd/(Rs-Rp)+sd/(Rs+Rp))

#data = pd.DataFrame(column_stack([theta,V,sdtheta,sdV]))
#print(tabulate(data))
```

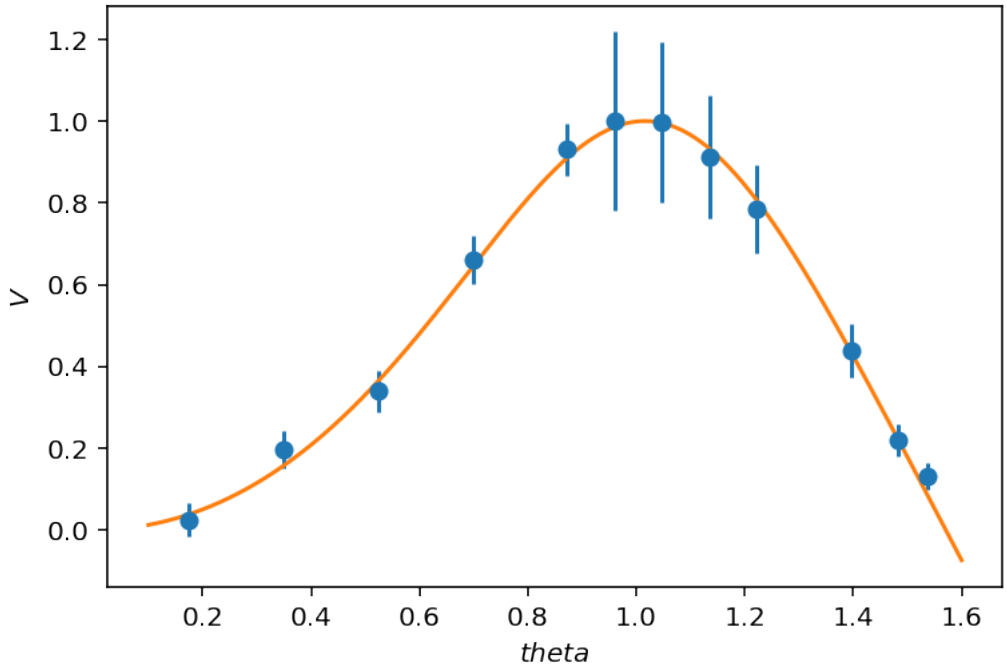
Fit Visibilità

```
In [ ]: FIT:    data read from "visibility.rtf" u (($1)/180*pi):2:(($4)/180*pi):4 xyerror
                format = x:z:sx:s
                x range restricted to [0.00000 : 1.60000]
                #datapoints = 12
        function used for fitting: V(x)
                V(x)=(fs(x)-fp(x))/(fs(x)+fp(x))
                fs(x)=abs((cos(x)-sqrt(n**2-(sin(x))**2))/(cos(x)+sqrt(n**2-(sin(x))**2)))**2
                fp(x)=abs((-n**2*cos(x)+sqrt(n**2-(sin(x))**2))/(n**2*cos(x)+sqrt(n**2-(sin(x)
        fitted parameters initialized with current variable values

        iter      chisq       delta/lim  lambda   n
           0 3.1194622056e+00   0.00e+00  6.28e+00    1.612424e+00
           1 3.1194622054e+00  -4.18e-06  6.28e-01    1.612424e+00

        After 1 iterations the fit converged.
        final sum of squares of residuals : 3.11946
        rel. change during last iteration : -4.17516e-11

        degrees of freedom    (FIT_NDF)                        : 11
        rms of residuals      (FIT_STDFIT) = sqrt(WSSR/ndf)    : 0.532529
        variance of residuals (reduced chisquare) = WSSR/ndf   : 0.283587
        p-value of the Chisq distribution (FIT_P)              : 0.989051

        Final set of parameters            Asymptotic Standard Error
        =======================            ==========================
        n               = 1.61242          +/- 0.03946      (2.447%)
```

```
In [23]: nv = 1.61242
         sdnv = 0.03946
```

```
In [26]: V = (Rs-Rp)/(Rs+Rp)
         xlabel("$theta$")
         ylabel("$V$")
         errorbar(radians(theta), V, yerr = sdV, fmt='o')

         x=linspace(0.1,1.6,1000)
         plot(x,(fs(x,nv)-fp(x,nv))/(fs(x,nv)+fp(x,nv)))
```

```
Out[26]: [<matplotlib.lines.Line2D at 0x1516228710>]
```


1.2
1.0
0.8
0.6
0.4
0.2
0.0
V
0.2
0.4
0.6
0.8
1.0
1.2
1.4
1.6
theta