

Sleter Nellen

Bitácora

20/11/20

7:31

Midiendo la constante de Planck

Reunión con prof. y ayudante.

-> Discretizamos experimento

-> Tenemos material para hacerlo en casa.

- LEDs

- 2 m x 2 metros

(Isofronic, Radiosback)

- Resistencias de carbón

- 9V (72xAA)

10:14

-> En el taller

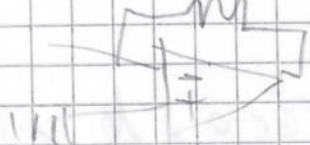
Encuentramos circuito brasileño

mucho complicado.

-> sugiere usar batería o

convertidor de corriente a voltaje.

Negativo C-, (ningún mult. tiene ampl.)



-> Fuente?

LM7812

Fuente o pump

2 pilas AA en serie, 2 veces.  
12:22

$\rightarrow$  Esta demasiado bajada.

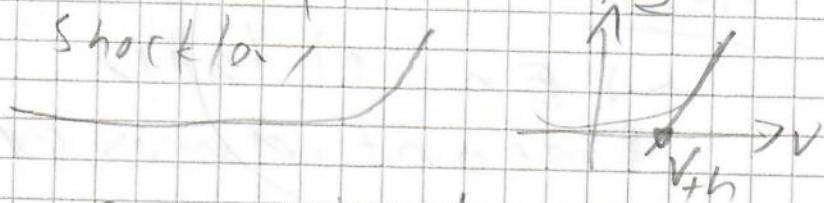
Fuente o pump  $\rightarrow$  compresión  
 $\pm 12V$ .

Discutir más sobre el efecto.

$\rightarrow$  Presentar la teoría y práctica.

Teoría:

$$\text{Queremos } I = J_0 e^{-\frac{eV_h}{kT}} \left( e^{\frac{eV}{kT}} - 1 \right)$$



S:

Ajustar recta suponiendo que

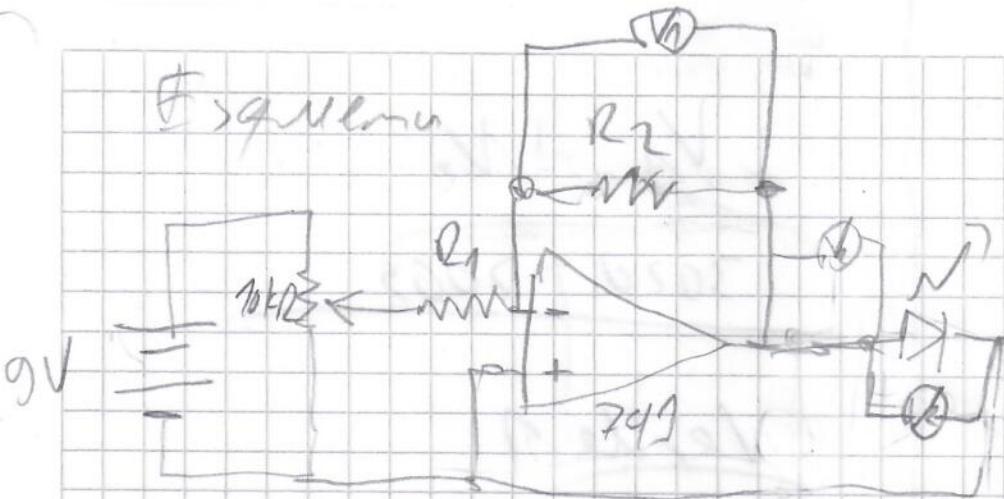
se prende con  $V > V_{th}$ .

$$eV_{th} = E_{emisiva} = h\nu \Rightarrow V_{th} = \frac{h\nu}{e}$$

Experimento

=) midiendo  $V_{th}$  varias veces de distintos modos  
a  $m(V - V_{th}) \rightarrow$ . De estas medidas (y otras) se obtiene la pendiente  $m$  de  $y$  versus  $x$ .

25/11/120



$$R_1 = 102.0 \pm 0.05 \Omega$$

$$R_2 = 102.7 \pm 0.05 \Omega$$

R<sub>obj</sub> d2

V<sub>1</sub>(mV) V<sub>2</sub>(mV)

4.4	10.0
9.6	336.6
9.3	228.8
10.5	544
12.9	294
13.0	895
15.0	1255
18.5	1506
22.2	1595

V <sub>1</sub>	V <sub>2</sub>
29.1	1580
33.1	1602
44.6	1616
64.5	1642
71.2	1648
78.3	1653
80.5	1656
91.3	1663
99.7	1666

$V_1$  |  $V_2$

117.7	1625
129.5	1684
101.2	1656
160.8	1698
188	1709
233	1722
187	1726
229	1735
324	1746
396	1761
471	1775
631	1803
515	1783
717	1815
838	1831
020	1843
716.5	1843
1430	1904

Sexta-feira

$V_1$  |  $V_2$

3020	2063
------	------

Verde 7

$V_1$  [mV] |  $V_2$  [mV]

4.3	-10.0
8.8	16.02
9.3	27.7.0
10.5	46.0
11.5	6.50
13.7	92.2
14.2	1930
15.2	1375
16.2	1508
19.1	1692
21.3	1737
23.5	175
27.4	1768
32.7	1783

<u><math>V_1</math></u>	<u><math>V_2</math></u>	<u><math>V_1</math></u>	<u><math>V_2</math></u>	<u>P</u>
45.0	1808	678	2026	
50.8	1816	916	2026	
58.2	1825	2.89	2.024	
63.6	1830	se satura		
69.2	1835			
70.7	1842			
88.9	1851			2911, 20
55.7	1855	Hasta un corto.		
110.5	1865	Si vuelve el LED amarillo		
122.3	1874	corrección de error de gráfica		
137.7	1879	se tenga que volver a medir		
157.6	1886	todo		
172.5	1894	Amarillo		
195	1903			
214	1911	55.6		53.5
259	1926	59.6		57.5
361	1950	63.0		60.7
462	1961	78.2		76.7
513	1997	80.8		78.8
		55.5		83.6

# Rojo 2

1 1

$V_1$	$V_2$	$V_i \text{ [mV]}$	$V_o \text{ [mV]}$
92.0	84.1	3.1	1489
98.1	94.4	1.5	1461
99.5	97.8	0.3	1397
108.4	106.0	14.3	1557
115.8	119.6	18.3	1568
<u>15.35 /</u>		25.6	1552
Error extraido en opamp		41.8	1602
Se quieren dos dígitos		50.9	1616
Nuevo circuito:		57.4	1621
		63.9	1627
		68.8	1631
		76.12	1637
		83.9	1649
		96.3	1698
(Nota: El error)		104.0	1653
fue que la salida del		127.3	1669
opamp es		147.2	1670
negativa.		1553	1675
No queremos		1648.1	1681
operación inversa de LED).			

$V_1$	$V_2$	$V_{12} \text{ [mV]}$	$V_{12} \text{ [mV]}$	$V_1$	$V_2$
232	1707	0.1	1590	455	1929
230	1712	2.5	1702	730	2044
342	1728	8.7	1713	1728	2116
138	1748	17.7	1725	1378	2153
591	1767	26.6	1756	834	2053
676	1781	36.2	1811	595	2005
843	1812	46.2	1821	1167	2123
945	1825	51.6	1828	Se satur	
7202	1858	63.5	1838	1430	2160
1400	1880	72.1	1845	1340	2152
1691	1912	81.9	1850	Se satur	
2020	1944	96.9	1862	5290	2412
Se satur		105.4	1866	7860	2740
5290	2272	113.2	1870		
5790	2319	140.7	1883		
		164.5	1888		
		205	1910		
		260	1928		
		326	1946		

# Todo son mV

1 1

Amarillo 2

<u>V<sub>1</sub>(mV)</u>	<u>V<sub>2</sub>(mV)</u>	<u>V<sub>1</sub></u>	<u>V<sub>2</sub></u>	<u>V<sub>1</sub></u>	<u>V<sub>2</sub></u>	<u>V<sub>1</sub></u>	<u>V<sub>2</sub></u>	<u>V<sub>1</sub></u>	<u>V<sub>2</sub></u>
0.5	1536	356	1677	35.2	2541	1119	3080		
1.2	1574	550	1970	42.2	2554	8050	4474		
4.4	1626	660	1932	52.2	2523	9560	3543		
8.1	1653	807	1955	63.2	2592	4200	3493		
12.6	1673	915	1974	72.2	2602			<u>IR</u>	
20.0	1694	1184	2016	77.0	2610				
24.3	1703	1412	2050	93.5	2637	<u>V<sub>1</sub></u>	<u>V<sub>2</sub></u>		
34.3	1720	1670	2080	110.2	2658	0.5	879		
40.4	1728	1890	2510	122.2	2632	3.2	857		
49.9	1740	5890	2587	137.0	2688	10.8	551		
60.7	1750	142612		161.7	2732	12.3	976		
70.2	1758	<u>V<sub>1</sub></u>	<u>V<sub>2</sub></u>	214	2759	22.8	989		
84.0	1767	0.7	2228	283	2805	31.6	1004		
91.6	1773	6.5	22120	410	2972	51.8	1026		
121.2	1790	2.0	2330	450	2990	60.1	1039		
136.5	1798	3.4	2325	528	2929	71.8	1043		
157.3	1807	9.0	2441	626	2959	89.3	1052		
193.6	1817	16.1	2479	716	2989	100.7	1057		
206	1824	22.4	2504	824	3015	118.2	1063		
267	1845	31.6	2528	987	3053	147	1073		

$\sigma \approx 10\%$

Verde ↑

<u><math>V_1</math></u>	<u><math>V_2</math></u>	<u><math>V_1</math></u>	<u><math>V_2</math></u>	<u><math>V_1</math></u>	<u><math>V_2</math></u>	<u><math>V_1 + V_2</math></u>
193.4	1083	30.0	2692	1808	198.9	
222	1068	40.6	2666	1895	435	
250	1097	50.8	2683	12		
333	1104	62.2	2700			
434	1113	69.6	2705			
502	1126	75.1	2720			
706	1133	91.4	2733			
828	1139	103.1	2743			
1206	1155	121.8	2758			
1408	1161	143.4	2775			
		168.5	2788			
Saturación						
6150	1243		30.97.20			
6730	1247		Nuevo diseño			

Demasiado sensible

Los huele mejor  
mediante  
observación  
cuando se enciende.

Verde 2

$V_1$      $V_2$   
3.4    2489

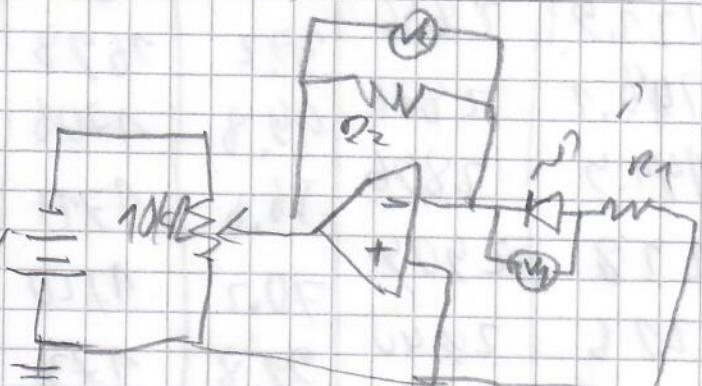
$$R_2 = 800.2 \Omega$$

$$R_1 = 102.0 \text{ k}\Omega$$

9.1    2559  
15.0    2599

(Experimento  
nro 7 con atm)

23.6    2621  
Sustituir  
mejor



Idea: invertir circuito

o siempre min. escala

R = 100.2

Regresando

42012



$V_2$ [mV]	$V_1$ [mV]	$V_1$	$V_2$	$V_3$	$V_4$	
0.7		687	3095	1694	1753	<u>181.56</u>
147	0.					
2510	1.7	759	3052	212	1764	Se satura
2670	9.3	1103	3923	322	1781	de mas alto
2720	20.9	80satura	913	1753	Rápido	
		3690	3555	538	1802	(0, 0)
		4560	3526	635	1820	Circuito 2.
32.7	2756	<u>R0jd 2</u>	918	1840	Med. remo	
43.7	2777	<u>V1</u>		1151	1855	directamente
50.8	2789	<u>0.5</u>	1561	4087	1870	(nando) se
60.8	2802	<u>0.5</u>	7631	se satura	disparar	
80.3	2823	<u>0.5</u>	1600	5969	2039	O se enciende
700.0	2847	<u>13.9</u>	1685			el diodo.
721.9	2856	<u>25.4</u>	1685			→
748.1	2872	<u>37.2</u>	1698			
191.5	2896	<u>49.8</u>	1708			
278	2909	<u>56.6</u>	1711			
29.6	2940	<u>70.2</u>	1720			
389	2970	<u>92.8</u>	1730			
540	3012	<u>115.3</u>	1738			
		<u>140.8</u>	1746			

El riesgo  
de que  
el voltaje  
de saturación  
en vez del  
voltaje  
numérico.

19:08

mediciones directas

Rojos 1  $V_{th} (\text{mV})$

$2030 \pm 50$

Rojos 2

$1960 \pm 50$

Verde 1

$2420 \pm 50$

Verde 2

$2430 \pm 50$

Amarillo

$2160 \pm 50$

Azul 1

$2320 \pm 50$

Azul 2

$2850 \pm 50$

IR

$1150 \pm 50$

3. 12. 20

Andrés dice que tomar la freq como todo el espectro no da resultados.

20:17

→ Encuentré todas las frecuencias de LEN y código de color.

→ Tíene curvas para el efecto de longitud de onda.

20:22

Coinciden con nuestros colores.  
Para  $\sigma$

Hay curvas para cada  $\lambda$  de la forma



aproximar a  $\exp\left(-\frac{(\lambda-\lambda_0)^2}{2\sigma^2}\right)$   
(no está normalizada)

$$\text{A25} \quad \text{si } \lambda - \lambda_0 = \theta \Rightarrow \exp\left(-\frac{\theta^2}{2\sigma^2}\right) = \frac{1}{\sqrt{2}}$$

El error es el ancho de la curva a  $\frac{1}{\sqrt{2}} = 0.667$ .

Con esto se obtiene

LED	$\lambda(\text{nm})$	LED	$\lambda(\text{nm})$
Roj01	$657 \pm 15$	Azul1	$470 \pm 27$
Roj02	$657 \pm 15$	Azul2	$430 \pm 35$
Verde 1	$525 \pm 27$	Infrarrojo	
Verde 2	$525 \pm 27$		
A morillo	$583 \pm 16$		

# Bitacora Práctica 3

Diodos: R, V, Amarillo, Cyan, Azul, Infrarrojo  
 ↑      ↑      ↑  
 dos    dos    dos      "Azules distintos"

Ajuste lineal:  $y = m(x - x_0)$  ← El ajuste a  $x_0$  será  $V_s, sat$

$$(\sigma_y)^2 = \left(\frac{\partial y}{\partial m}\right)^2 \sigma_m^2 + \left(\frac{\partial y}{\partial x}\right)^2 \sigma_x^2 + \left(\frac{\partial y}{\partial x_0}\right)^2 \sigma_{x_0}^2 = (x - x_0)^2 \sigma_m^2 + m^2 (\sigma_x^2 + \sigma_{x_0}^2)$$

$$y = I, \quad x = V$$

$$I = V_s / R \Rightarrow \sigma_I^2 = \frac{V_s^2}{R^2} \sigma_R^2 + \frac{\sigma_{V_s}^2}{R^2}$$

Para ajuste a  $h$ :

$$E = h\nu, \quad V = E/q, \quad v = c/\lambda \Rightarrow V = \frac{hv}{q}$$

$$q, c \text{ se toman como exactas: } q = 1,602176634 \times 10^{-19} C \quad V = \frac{hv}{q} + V_0$$

$$c = 299792458 \text{ m/s}$$

$$\Rightarrow \sigma_V^2 = \frac{v^2}{q^2} \sigma_h^2 + \frac{h^2}{q^2} \sigma_v^2, \quad \sigma_v = \frac{1}{q} \sqrt{(v \sigma_h)^2 + (h \sigma_v)^2} \quad \sigma_V = \sqrt{\left(\frac{v \sigma_h}{q}\right)^2 + \left(\frac{h \sigma_v}{q}\right)^2 + \sigma_{V_0}^2}$$

$$\sigma_v^2 = \frac{C^2}{\lambda^4} \sigma_\lambda^2, \quad \sigma_v = \frac{C}{\lambda^2} \sigma_\lambda = v \frac{\sigma_\lambda}{\lambda}$$

Espectro visible:

$$\text{Rojo: } 625-700 \text{ nm} \Rightarrow \lambda = 662.5 \pm 37.5 \text{ nm}$$

$$\text{Amarillo: } 565-590 \text{ nm}$$

$$\text{Verde: } 500-565 \text{ nm}$$

$$\text{Cyan: } 485-500 \text{ nm}$$

$$\text{Azul: } 450-485 \text{ nm}$$

Propuesta:

$$\lambda = 577.5 \pm 12.5 \text{ nm}$$

$$\lambda = 532.5 \pm 32.5 \text{ nm}$$

$$\lambda = 492.5 \pm 7.5 \text{ nm}$$

$$\lambda = 467.5 \pm 17.5 \text{ nm}$$

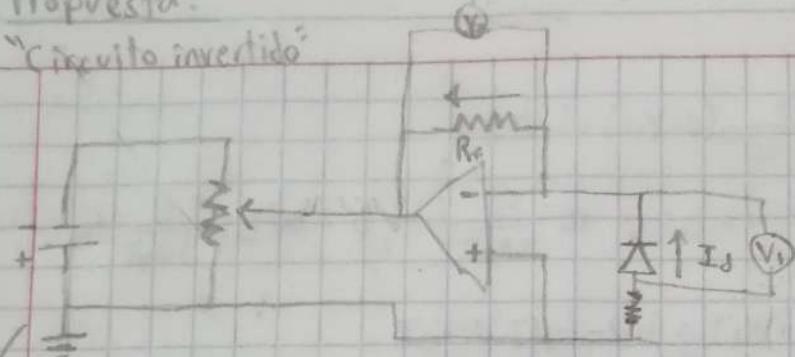
Rango máximo en el que se puede poner la longitud de onda

Nueva propuesta: Gráfica de espectro de diodos

Propuesta:

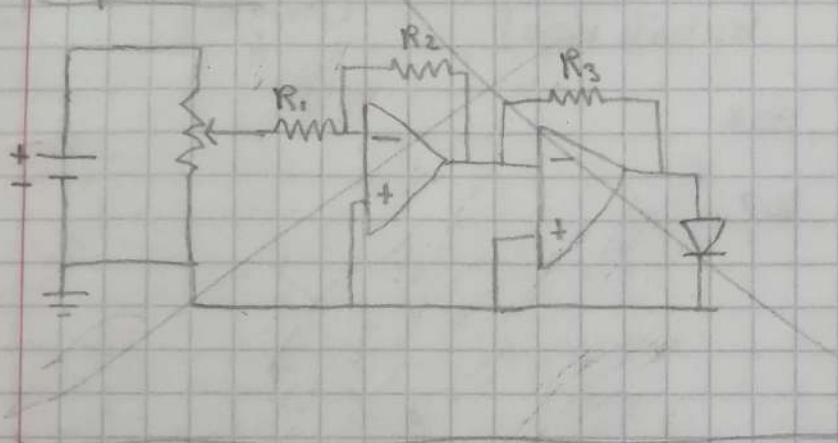
"Circuito invertido"

$$V_2 = -I_d R_f$$



No estamos seguros  
de las polarizaciones  
Si están bien  
las polaridades ✓

Propuesta:



Assigogramm

Arbeits B

Avances bitácora  
gramática  
Vi 11

$$R_1 = 100.2 \Omega$$

Experimento 1: "Muy sensible"

Experimento 2:

Visat [V] ← cuando enciende

Visat [V]

$$\text{Rojo 1} \quad 2.03$$

$$\text{Rojo 1} \quad 1.79$$

$$\text{Verde 1} \quad 2.12 \pm 1$$

$$\text{Verde 1} \quad 2.01$$

$$c = \lambda v$$

$$\text{Amarillo 1} \quad 2.16$$

$$\text{Amarillo 1} \quad 1.91$$

$$\text{Azul 1} \quad 2.72$$

$$\text{Azul 1} \quad 2.75$$

$$\text{IR} \quad 1.19$$

$$\text{IR} \quad 1.12$$

$$\text{Rojo 2} \quad 1.96$$

$$\text{Rojo 2} \quad 1.82$$

Lado P Lado N

$$\text{Verde 2} \quad 3.18 (?)$$

$$\text{Verde 2} \quad 2.43$$

erzeugen = generar

$$\text{Azul 2} \quad 2.89$$

$$\text{Azul 2} \quad 2.75$$

Gesch erhöhen

$$I = I_0 e^{\frac{e(V-V_{th})}{nK_B T}}$$

$$\ln(I) = \frac{e}{nK_B T} (V - V_{th}) + \ln(I_0)$$

$$\text{ie } \ln(I) = mV + b, \quad m = \frac{e}{nK_B T}, \quad b = \ln(I_0) - \frac{e}{nK_B T} V_{th}$$

$$\text{de ahí: } Y(V) = -b/m = eV_{th} - \ln(I_0) nK_B T = hV - \ln(I_0) nK_B T$$

$(E = eV_{th} = hV)$  ordenada cambia en cada diodo!

Depurando  $R_{o1a}$  y  $V_{e1}$  (circuito viejo)

(inviable)

Ent.  $R_{o1b}, R_{o2} \rightarrow R_{o1}, R_{o2}$

$V_{e2}, V_{e3} \rightarrow V_{e1}, V_{e2}$

Proc.: determinante

Carátula presentación:

Univ. Escudo

Fac. Título. Nombre autor

UNAM → Grande  
Fac. → Grande

Solo 1 par de gráficas  $I_d - V_d$

Infrarrojo → IR  
 $M = h/e$   
 $I_d(V_d)$ , lo relevante es  $V_{th}$

```
In [1]: import numpy as np
import matplotlib.pyplot as plt
from scipy.optimize import curve_fit
import os #Para adquirir el path global y guardar figuritas
```

```
In [39]: h_real = 6.62607015e-34 #J*s
e = 1.602176634e-19 #C
c = 299792458 #m/s
#R = 102.7 #Ohm En Los primeros dos experimentos
R = 100.2 #Ohm
sR = 0.05
M_real = h_real/e

def y(x,m,x0): #Modelo Lineal general para ajustar
    return m*(x-x0)

def sy(x,sx,m,sm,x0,sx0):
    return np.sqrt(m**2*(sx**2+sx0**2)+sm**2*(x-x0)**2)

#En deshuso
#def V(nu,M,V0): #Modelo Lineal para ajustar h, q exacta. M = h/q
#    return M*nu+V0
#
#def sV(nu,snu,M,sM,V0,sV0):
#    return np.sqrt((nu*sM)**2+(M*snu)**2+sV0**2)

def V(nu,M): #Modelo Lineal para ajustar h, q exacta. M = h/q
    return M*nu

def sV(nu,snu,M,sM):
    return np.sqrt((nu*sM)**2+(M*snu)**2)
```

```
In [3]: #Lambdas directo del espectro visible
lamb_rojo, slamb_rojo = (625+700)/2, (700-625)/2
lamb_verde, slamb_verde = (500+565)/2, (565-500)/2
lamb_amarillo, slamb_amarillo = (565+590)/2, (590-565)/2
#lamb_cyan, slamb_cyan = (485+500)/2, (500-485)/2
lamb_azul, slamb_azul = (450+485)/2, (485-450)/2
lamb_IR, slamb_IR = 940, 940-760
```

```
In [3]: #Lambdas y desviaciones estándar estimadas de la gráfica técnica
lamb_IR, slamb_IR = 945, 22 #Infrared (1)
lamb_rojo, slamb_rojo = 657, 15 #Se tomó el denominado rojo, pero hay otros tipos de rojo posibles
lamb_verde, slamb_verde = 525, 27 #Ultra Green (A)
lamb_amarillo, slamb_amarillo = 583, 18 #Yellow
lamb_azul, slamb_azul = 430, 35 #Ultra Blue
lamb_cyan, slamb_cyan = 470, 27 #Pure Blue
```

In [77]: #Con cyan

```
L = ["Ro1","Ro2","Ve1","Ve2","Am1","Azul1","Azul2","IR"]
Nombre = ["rojo 1", "rojo 2", "verde 1", "verde 2", "amarillo", "azul 1", "azul 2", "infrarrojo"]
Lamb = np.array([lamb_rojo,lamb_rojo,lamb_verde,lamb_verde,lamb_amarillo,lamb_cyan,lamb_azul,lamb_IR])*1e-9 #m
sLamb = np.array([slamb_rojo,slamb_rojo,slamb_verde,slamb_verde,slamb_amarillo,slamb_cyan,slamb_azul,slamb_IR])*1e-9 #m
V1th = [0,0,0,0,0,0,0,0]
sV1th = [0,0,0,0,0,0,0,0]
```

## Método 1

In [5]:

```
def Read(i):
    Datos = np.loadtxt("Datos"+L[i]+".txt", skiprows=1).T
    Datos = Datos[:, Datos[0].argsort()] #Ordena datos

    LOW = [25,16,19,10,20,23,10,21]#indice menor
    low = LOW[i] #indice menor
    ind = np.arange(low,len(Datos[0])) #Datos a ajustar desde low hasta el máximo
    ind_extra = np.arange(0,low)
    d_ajustados = len(Datos[0])-low

    V1 = Datos[1][ind]
    V2 = Datos[0][ind]
    sV1 = Datos[3][ind]
    sV2 = Datos[2][ind]
    Ic = V2/R
    sIc = np.sqrt(V2**2*sR**2/(R**4)+(sV2/R)**2)

    V1_extra = Datos[1][ind_extra]
    V2_extra = Datos[0][ind_extra]
    sV1_extra = Datos[3][ind_extra]
    sV2_extra = Datos[2][ind_extra]
    Ic_extra = V2_extra/R
    sIc_extra = np.sqrt(V2_extra**2*sR**2/(R**4)+(sV2_extra/R)**2)

    return V1, Ic, sV1, sIc, V1_extra, Ic_extra, sV1_extra, sIc_extra, d_ajustados
```

```
In [66]: #Rojo1
i = 0
V1, Ic, sV1, sIc, V1_extra, Ic_extra, sV1_extra, sIc_extra, d_ajustados = Read
(i)
popt, pcov = curve_fit(y, xdata = V1, ydata = Ic, sigma = sIc)
m_fit, x0_fit = popt
sm_fit, sx0_fit = np.sqrt(pcov[0][0]), np.sqrt(pcov[1][1])
print("m_fit = "+str(m_fit)+" +- "+str(sm_fit))
print("x0_fit = "+str(x0_fit)+" +- "+str(sx0_fit))

m_fit = 0.08545421947741375 +- 0.0011276096106633958
x0_fit = 1713.9598265831926 +- 1.5476839387967183
```

```
In [96]: #Rojo2
i = 1
V1, Ic, sV1, sIc, V1_extra, Ic_extra, sV1_extra, sIc_extra, d_ajustados = Read
(i)
popt, pcov = curve_fit(y, xdata = V1, ydata = Ic, sigma = sIc)
m_fit, x0_fit = popt
sm_fit, sx0_fit = np.sqrt(pcov[0][0]), np.sqrt(pcov[1][1])
print("m_fit = "+str(m_fit)+" +- "+str(sm_fit))
print("x0_fit = "+str(x0_fit)+" +- "+str(sx0_fit))

m_fit = 0.16156529172551726 +- 0.011506967589816156
x0_fit = 1780.8445484790097 +- 4.464394427173474
```

```
In [99]: #Verde1
i = 2
V1, Ic, sV1, sIc, V1_extra, Ic_extra, sV1_extra, sIc_extra, d_ajustados = Read
(i)
popt, pcov = curve_fit(y, xdata = V1, ydata = Ic, sigma = sIc)
m_fit, x0_fit = popt
sm_fit, sx0_fit = np.sqrt(pcov[0][0]), np.sqrt(pcov[1][1])
print("m_fit = "+str(m_fit)+" +- "+str(sm_fit))
print("x0_fit = "+str(x0_fit)+" +- "+str(sx0_fit))

m_fit = 0.04932791908859819 +- 0.0007583532266485239
x0_fit = 1888.3127232716035 +- 2.507666460481306
```

```
In [102]: #Verde2
i = 3
V1, Ic, sV1, sIc, V1_extra, Ic_extra, sV1_extra, sIc_extra, d_ajustados = Read
(i)
popt, pcov = curve_fit(y, xdata = V1, ydata = Ic, sigma = sIc)
m_fit, x0_fit = popt
sm_fit, sx0_fit = np.sqrt(pcov[0][0]), np.sqrt(pcov[1][1])
print("m_fit = "+str(m_fit)+" +- "+str(sm_fit))
print("x0_fit = "+str(x0_fit)+" +- "+str(sx0_fit))

m_fit = 0.013446659584589954 +- 0.0006951073478863978
x0_fit = 2666.230174816681 +- 4.659784290695643
```

In [85]: #Amarillo1

```
i = 4
V1, Ic, sV1, sIc, V1_extra, Ic_extra, sV1_extra, sIc_extra, d_ajustados = Read(i)
popt, pcov = curve_fit(y, xdata = V1, ydata = Ic, sigma = sIc)
m_fit, x0_fit = popt
sm_fit, sx0_fit = np.sqrt(pcov[0][0]), np.sqrt(pcov[1][1])
print("m_fit = "+str(m_fit)+" +- "+str(sm_fit))
print("x0_fit = "+str(x0_fit)+" +- "+str(sx0_fit))
```

m\_fit = 0.060954826949696575 +- 0.0009134918552428922  
x0\_fit = 1822.0942188119 +- 1.974902156291858

In [87]: #Azul1

```
i = 5
V1, Ic, sV1, sIc, V1_extra, Ic_extra, sV1_extra, sIc_extra, d_ajustados = Read(i)
popt, pcov = curve_fit(y, xdata = V1, ydata = Ic, sigma = sIc)
m_fit, x0_fit = popt
sm_fit, sx0_fit = np.sqrt(pcov[0][0]), np.sqrt(pcov[1][1])
print("m_fit = "+str(m_fit)+" +- "+str(sm_fit))
print("x0_fit = "+str(x0_fit)+" +- "+str(sx0_fit))
```

m\_fit = 0.039416306341173726 +- 0.0014936905064095037  
x0\_fit = 2801.8917644478333 +- 8.075792756863747

In [89]: #Azul2

```
i = 6
V1, Ic, sV1, sIc, V1_extra, Ic_extra, sV1_extra, sIc_extra, d_ajustados = Read(i)
popt, pcov = curve_fit(y, xdata = V1, ydata = Ic, sigma = sIc)
m_fit, x0_fit = popt
sm_fit, sx0_fit = np.sqrt(pcov[0][0]), np.sqrt(pcov[1][1])
print("m_fit = "+str(m_fit)+" +- "+str(sm_fit))
print("x0_fit = "+str(x0_fit)+" +- "+str(sx0_fit))
```

m\_fit = 0.04324573968167225 +- 0.0026229092360033133  
x0\_fit = 2878.9018180118096 +- 8.698030044131542

In [91]: #IR

```
i = 7
V1, Ic, sV1, sIc, V1_extra, Ic_extra, sV1_extra, sIc_extra, d_ajustados = Read(i)
popt, pcov = curve_fit(y, xdata = V1, ydata = Ic, sigma = sIc)
m_fit, x0_fit = popt
sm_fit, sx0_fit = np.sqrt(pcov[0][0]), np.sqrt(pcov[1][1])
print("m_fit = "+str(m_fit)+" +- "+str(sm_fit))
print("x0_fit = "+str(x0_fit)+" +- "+str(sx0_fit))
```

m\_fit = 0.5937737750075394 +- 0.08962577526595315  
x0\_fit = 1132.8137087298692 +- 4.432573118138958

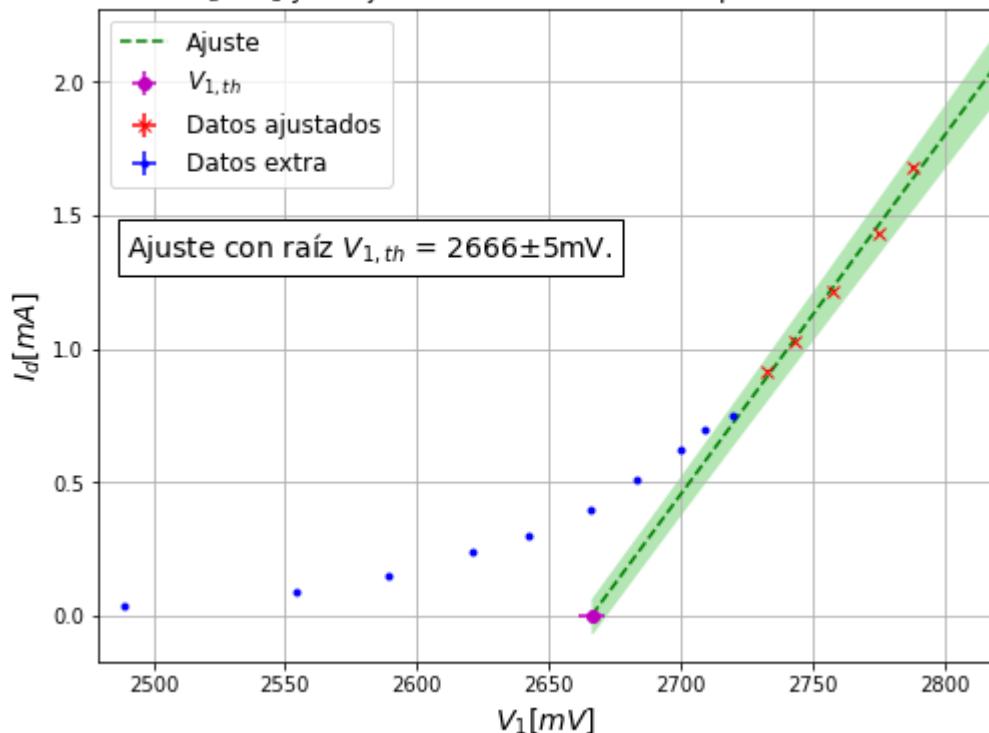
In [106]: #Gráfica individual

```
V_cont = np.linspace(x0_fit,V1.max()+30,100)

fig = plt.figure(figsize=(8,6))
plt.errorbar(x0_fit,0,0,sx0_fit,"mo",label=r"$V_{1,th}$")
plt.errorbar(V1,Ic,sIc,sV1,"rx",label="Datos ajustados")
plt.errorbar(V1_extra,Ic_extra,sIc_extra,sV1_extra,"b.",label="Datos extra") # Datos extra que fueron medidos, pero no ajustados
y_fit = y(V_cont,m_fit,x0_fit)
plt.plot(V_cont,y_fit,"g--",label="Ajuste")
sy_fit = sy(V_cont,0,m_fit,sm_fit,x0_fit,sx0_fit)
plt.fill_between(V_cont,y_fit+sy_fit,y_fit-sy_fit,color=(0.7,0.9,0.7))
plt.title(r"Relación $I_d - V_1$ y su ajuste a los últimos "+str(d_ajustados)+
" datos para el diodo "+r"\n"+Nombre[i]+ ".",
          fontsize=13)
plt.xlabel(r"$V_1[mV]$",fontsize=14)
plt.ylabel(r"$I_d[mA]$",fontsize=14)
plt.xlim(V1_extra.min()-10,V1.max()+30)
POSX = [1400,1560,1600,2490,1550,2050,2755,830] #DatosR1a, DatosR1b , DatosR2...
2...
POSY = [14,13,10,1.35,12,8,8,60] #DatosR1a, DatosR1b, DatosR2...
posx , posy = POSX[i], POSY[i]
props=dict(boxstyle='square', facecolor='white', alpha=1)
plt.text(posx, posy, r"Ajuste con raíz $V_{1,th}$ = "+"{:.0f}".format(x0_fit)+
"$\pm$"+"{:.0f}".format(sx0_fit)+"mV.",
         fontsize=14, bbox=props)
plt.legend(fontsize=12)
plt.grid()
plt.show()

#Para guardar la gráfica
fig.savefig("C:\Andrés\IX Semestre\Lab. Contemporánea I\Práctica 3\Plots\ "[:-1]+str(i)+".png",bbox_inches='tight') #El [-1] es para quitar el espacio, necesario para que se comente bien por el \
```

### Relación $I_d - V_1$ y su ajuste a los últimos 5 datos para el diodo verde2.



```
In [52]: #Recopilación de voltajes de Vth
for i in range(0,8):
    V1, Ic, sV1, sIc, V1_extra, Ic_extra, sV1_extra, sIc_extra, d_ajustados =
    Read(i)
    popt, pcov = curve_fit(y, xdata = V1, ydata = Ic, sigma = sIc)
    m_fit, x0_fit = popt
    sm_fit, sx0_fit = np.sqrt(pcov[0][0]), np.sqrt(pcov[1][1])

    V1th[i], sV1th[i] = x0_fit*1e-3, sx0_fit*1e-3 #Convertimos a V
    nu = c/Lamb #Hz
    snu = nu*sLamb/Lamb #Hz

    print(V1th)
    print(sV1th)

[1.71395983 1.78084455 1.88831272 2.66623017 1.82209422 2.80189176
 2.87890182 1.13281371]
[0.00154768 0.00446439 0.00250767 0.00465978 0.0019749 0.00807579
 0.00869803 0.00443257]
```

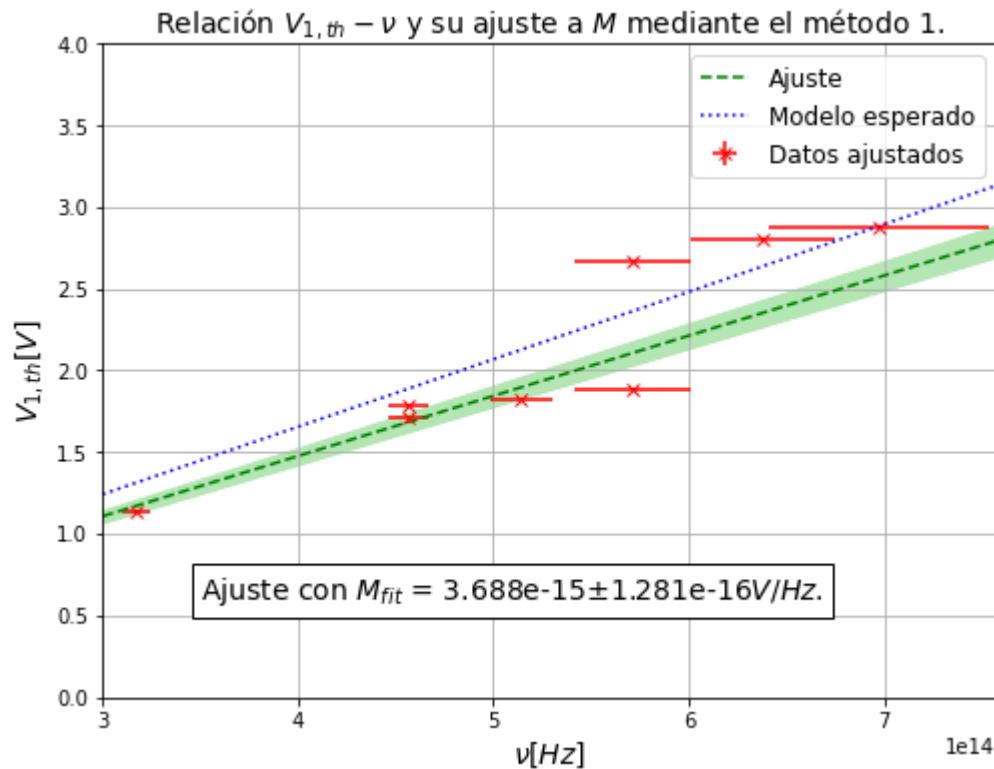
In [53]: #Ajuste a  $h$

```
popt, pcov = curve_fit(V, xdata = nu, ydata = V1th, sigma = sV1th)
M_fit = popt[0]
sM_fit = np.sqrt(pcov[0][0])
h_fit, sh_fit = M_fit*e, sM_fit*e
print("M_fit = "+str(M_fit)+" +- "+str(sM_fit))
print("h_fit = "+str(h_fit)+" +- "+str(sh_fit))
print("h_real = "+str(h_real))
print("Diferencia porcentual a la real: "+"{:3f}%".format(100*np.abs(h_fit-h_real)/h_real)+"")
```

```
M_fit = 3.687859400758826e-15 +- 1.2813465379821185e-16
h_fit = 5.908602161373032e-34 +- 2.0529434832117436e-35
h_real = 6.62607015e-34
Diferencia porcentual a la real: 10.828%
```

```
In [54]: numin, numax = 3e14, nu.max()+0.6e14
nu_cont = np.linspace(numin,numax,100)

plt.figure(figsize=(8,6))
plt.errorbar(nu,V1th,sV1th,snu,"rx",label="Datos ajustados")
y_fit = V(nu_cont,M_fit)
plt.plot(nu_cont,y_fit,"g--",label="Ajuste")
sy_fit = sV(nu_cont,0,M_fit,sM_fit)
plt.fill_between(nu_cont,y_fit+sy_fit,y_fit-sy_fit,color=(0.7,0.9,0.7))
plt.plot(nu_cont,V(nu_cont,M_real),"b:",label="Modelo esperado")
plt.title(r"Relación  $V_{1,\text{th}} - \nu$  y su ajuste a  $M$  mediante el método 1."
,fontsize=14)
plt.xlim(numin,numax)
plt.ylim(0,4)
plt.xlabel(r" $\nu$ [Hz]",fontsize=14)
plt.ylabel(r" $V_{1,\text{th}}$ [V]",fontsize=14)
posx , posy = 3.5e14, 0.6
props=dict(boxstyle='square', facecolor='white', alpha=1)
plt.text(posx, posy, r"Ajuste con  $M_{\text{fit}} = " + "{:.3e}".format(M_fit) + "\pm" + "{:.3e}" .format(sM_fit)+r"V/Hz".",
         fontsize=14, bbox=props)
plt.legend(fontsize=12)
plt.grid()
plt.show()$ 
```



## Método 2

In [55]: #Con cyan

```
V1th = np.array([2.03,1.96,2.42,2.43,2.16,2.72,2.89,1.19]) #V
sV1th = np.array([0.05,0.05,0.05,0.05,0.05,0.05,0.05,0.05]) #V
Lamb = np.array([lamb_rojo,lamb_rojo,lamb_verde,lamb_verde,lamb_amarillo,lamb_cyan,lamb_azul,lamb_IR])*1e-9 #m
sLamb = np.array([slamb_rojo,slamb_rojo,slamb_verde,slamb_verde,slamb_amarillo,slamb_cyan,slamb_azul,slamb_IR])*1e-9 #m
nu = c/Lamb #Hz
snu = nu*sLamb/Lamb #Hz
```

In [65]: print(snu)

```
[1.04179249e+13 1.04179249e+13 2.93674245e+13 2.93674245e+13
 1.58765486e+13 3.66428084e+13 5.67481667e+13 7.38549769e+12]
```

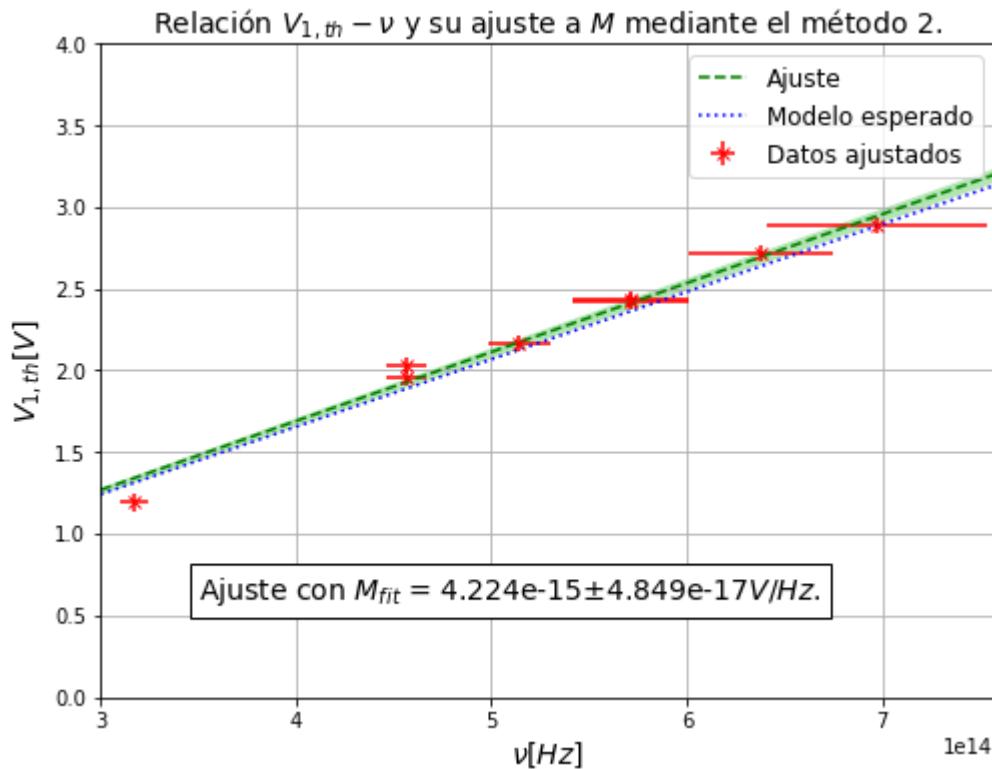
In [57]: #Ajuste a h

```
popt, pcov = curve_fit(V, xdata = nu, ydata = V1th, sigma = sV1th)
M_fit = popt[0]
sM_fit = np.sqrt(pcov[0][0])
h_fit, sh_fit = M_fit*e, sM_fit*e
print("M_fit = "+str(M_fit)+" +- "+str(sM_fit))
print("h_fit = "+str(h_fit)+" +- "+str(sh_fit))
print("h_real = "+str(h_real))
print("Diferencia porcentual a la real: "+"{:.3f}").format(100*np.abs(h_fit-h_real)/h_real)+"%"
```

```
M_fit = 4.22431790349848e-15 +- 4.849292561101135e-17
h_fit = 6.7681034395731315e-34 +- 7.769423232826255e-36
h_real = 6.62607015e-34
Diferencia porcentual a la real: 2.144%
```

```
In [50]: numin, numax = 3e14, nu.max()+0.6e14
nu_cont = np.linspace(numin,numax,100)

plt.figure(figsize=(8,6))
plt.errorbar(nu,V1th,sV1th,snu,"rx",label="Datos ajustados")
y_fit = V(nu_cont,M_fit)
plt.plot(nu_cont,y_fit,"g--",label="Ajuste")
sy_fit = sV(nu_cont,0,M_fit,sM_fit)
plt.fill_between(nu_cont,y_fit+sy_fit,y_fit-sy_fit,color=(0.7,0.9,0.7))
plt.plot(nu_cont,V(nu_cont,M_real),"b:",label="Modelo esperado")
plt.title(r"Relación  $V_{1,\text{th}} - \nu$  y su ajuste a  $M$  mediante el método 2."
,fontsize=14)
plt.xlim(numin,numax)
plt.ylim(0,4)
plt.xlabel(r" $\nu$ [Hz]",fontsize=14)
plt.ylabel(r" $V_{1,\text{th}}$ [V]",fontsize=14)
posx , posy = 3.5e14, 0.6
props=dict(boxstyle='square', facecolor='white', alpha=1)
plt.text(posx, posy, r"Ajuste con  $M_{\text{fit}} = " + "{:.3e}".format(M_fit) + "\pm" + "{:.3e}" .format(sM_fit)+r"V/Hz".",
         fontsize=14, bbox=props)
plt.legend(fontsize=12)
plt.grid()
plt.show()$ 
```



## Generación de tablas con corrientes

```
In [6]: def Write(i):
    """
        Escribe la tabla de datos con V1, V2 e Ic ordenados.
    """
    Datos = np.loadtxt("Datos"+L[i]+".txt",skiprows=1).T
    Datos = Datos[:, Datos[0].argsort()] #Ordena datos

    ind = np.arange(0,len(Datos[0])) #Indice total de datos
    V1 = Datos[1][ind]
    V2 = Datos[0][ind]
    sV1 = Datos[3][ind]
    sV2 = Datos[2][ind]
    Ic = V2/R
    sIc = np.sqrt(V2**2*sR**2/(R**4)+(sV2/R)**2)

    DatosW = np.array([V1,sV1,V2,sV2,Ic,sIc]).T
    np.savetxt("Datos_tabla_"+L[i]+".txt",DatosW,fmt=".0f\t%.1f\t%.1f\t%.2f\t
    %.2e\t%.2e") #Formato de cada columna, con tabulación
```

```
In [13]: Write(7)
```