Preparación para para el parcial

Algunas consideraciones importantes:

- En algunos problemas tendrán que analizar un conjunto de datos. Podrán hacerlo en código Python (disponible en las computadoras del aula del parcial). Si prefieren usar Origin, lleven su propia laptop con el programa instalado ya que esas computadoras **no** lo tienen instalado.
- Se les brindarán códigos generales para ajustes, armados de histogramas, cálculos de estimadores, graficado. Si prefieren usar sus propios códigos (**recomendado**), podrán hacerlo accediendo a su Drive o Colab. Pueden llevarlo también en un *pendrive* o en su propia computadora. En ese caso, asegúrense de llevar códigos preparados para:
  - Graficar datos con las barras de error, ejes e indicaciones adecuadas.
  - Realizar ajustes y estimadores asociados a los mismos.
  - Armar histogramas.
  - Graficar funciones.
  - Calcular todos los estimadores vistos en la materia.
- Ustedes deberán generar para entregar los problemas un archivo PDF con las figuras y respuestas. Para eso conviene primero editar el parcial con algún editor de texto. Asegurense de poder hacer todo el proceso en forma individual (practiquen como hacerlo si tienen dudas).

Acá agregamos algunos problemas tipo parcial para que resuelvan antes de la clase próxima en forma individual y generen el archivo PDF correspondiente con las respuestas.

Problema 1

Los datos de la Tabla 1 fueron obtenidos a partir de un experimento de un resorte realizando un movimiento oscilatorio. Se fue variando la masa y, en cada caso, se midió la frecuencia angular w con una frecuencia de adquisición de 1000 1/s. Por otro lado, se pesó cada objeto obteniendo el valor de la masa m con una precisión del 4%. En un resorte ideal sin rozamiento, se espera que se cumpla la relación:

Tabla 1: Valores nominales medidos (no estan incluidas las incertezas en cada uno).

	m (g)	w (1 / s)		m (g)	w (1 / s)		m (g)	w (1 / s)		m (g)	w (1 / s)
0	1005.0	0.064775	5	169.1	0.157080	10	88.6	0.216662	15	60.7	0.261799
1	497.4	0.091061	6	150.3	0.169816	11	83.5	0.224399	16	57.5	0.261799
2	341.9	0.112200	7	129.7	0.184800	12	69.8	0.232711	17	54.9	0.273182
3	251.0	0.130900	8	108.0	0.196350	13	70.8	0.241661	18	53.5	0.285599
4	195.7	0.146121	9	94.9	0.202683	14	63.3	0.251327	19	52.1	0.285599

$$w^2 = \frac{k}{m}$$

Puede cargar los datos con:

```python
import numpy as np

m, w = np.loadtxt("problema1.csv", delimiter=",", skiprows=1).T
```

1. La tabla muestra los 16 valores nominales obtenidos para cada variable. Evalue la incerteza de cada magnitud física medida y de aquellas que considere necesarias para realizar el ajuste.

La incerteza Δm en la masa m está dada como un error relativo. Podemos calcular el valor absoluto como:

```python
dm = 0.04 * m
```

Para obtener la incerteza Δw en la frecuencia angular w, tenemos que considerar la incerteza dado por la frecuencia de adquisición $f = 1000s^1$ que nos limita la incerteza Δt en los tiempos a $\Delta t = 1/f = 0.001\,s = 1\,ms$.

Asumiendo que solo se midió un periodo T considerando únicamente sus extremos, el error en el periodo sería Δt.

Podemos calcular la frecuencia angular w a partir del periodo T como:

$$w = \frac{2\pi}{T}$$

y, por propagación de errores, la incerteza Δw:

$$\Delta w = \frac{dw}{dT}\,\Delta T = \frac{2\pi}{T^2}\,\Delta T = \frac{w^2}{2\pi}\,\Delta T$$

```
dw = w**2 / (2*np.pi) * 0.001
```

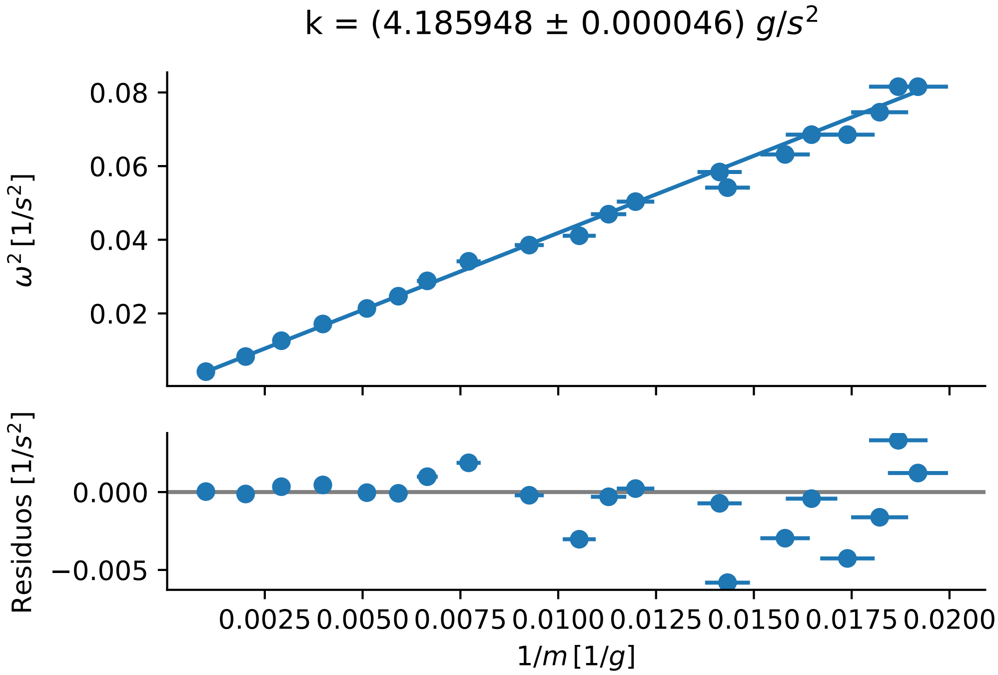
2. Obtenga el resultado de k realizando el ajuste de los datos que considere más conveniente. Justifique.

Como el modelo esperado es $\omega^2 = k/m$, vamos a realizar un ajuste lineal $y = kx$, donde $y = \omega^2$ y $x = 1/m$.

Por propagación de errores, tenemos $\Delta y = 2\omega\,\Delta\omega$ y $\Delta x = \Delta m/m^2$.

```
x = 1/m
y = w**2
dx = 1/m**2 * dm
dy = 2 * w * dw
```

```
import labo1


def func(x, k):
    return k * x


r = labo1.curve_fit(func, x=x, y=y, y_err=dy, rescale_errors=False)
fig, axes = r.plot_with_residuals(x_err=dx)
axes[0].set(ylabel=r"$\omega^2 \, [1/s^2]$")
axes[1].set(ylabel=r"Residuos $[1/s^2]$", xlabel=r"$1/m \, [1/g]$")
fig.align_labels()
fig.suptitle("k = ({} ± {}) $g/s^2$".format(*labo1.to_significant_figures(*r["k"])));
```



Como el ajuste por cuadrados mínimos solo considera los errores en y, es importante que los errores en x sean despreciables:

$$\Delta y \gg k\Delta x$$

Equivalentemente,

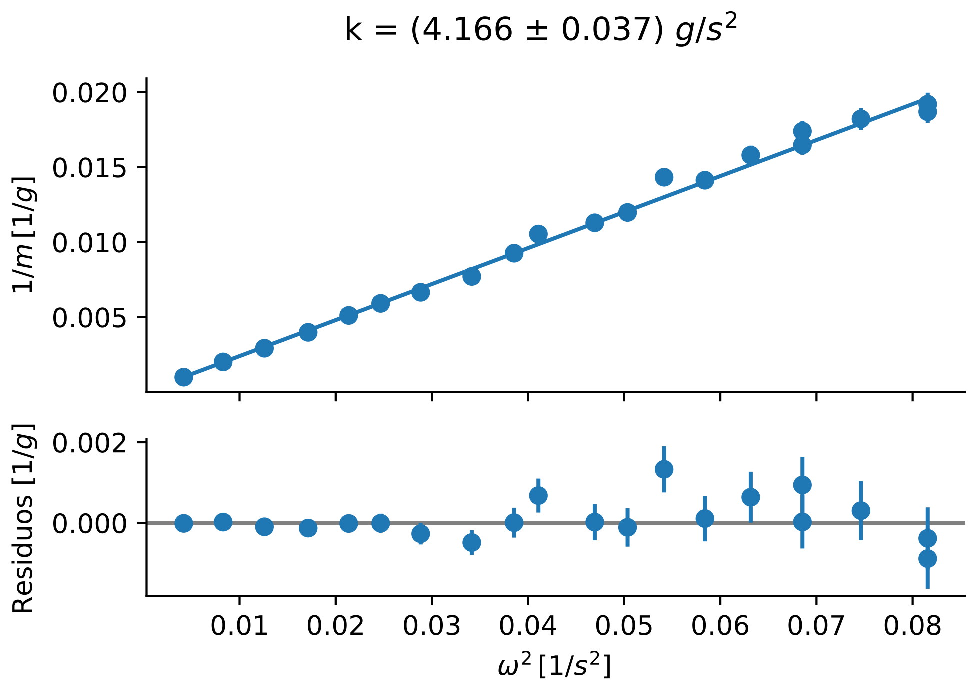
$$\frac{\Delta y}{k\Delta x} \gg 1$$

```
k, dk = r["k"]

dy / (k * dx)
```

```
array([0.00051926, 0.000714  , 0.00091807, 0.00107026, 0.00116071,
       0.00124596, 0.00139924, 0.00155612, 0.00155422, 0.00150215,
       0.00171308, 0.00179368, 0.00167225, 0.00189954, 0.00191037,
       0.00207057, 0.00196141, 0.00212777, 0.0023693 , 0.0023073 ])
```

Como vemos, esto no se cumple. Por lo tanto, cambiamos los roles en el ajuste para que considere el error de la otra variable.

```
def func(x, k):
    return 1/k * x

r = labo1.curve_fit(func, x=y, y=x, y_err=dx, rescale_errors=False)
fig, axes = r.plot_with_residuals(x_err=dy)
axes[0].set(ylabel=r"$1/m \, [1/g]$")
axes[1].set(ylabel=r"Residuos $[1/g]$", xlabel=r"$\omega^2 \, [1/s^2]$")
fig.align_labels()
fig.suptitle("k = ({} ± {}) $g/s^2$".format(*labo1.to_significant_figures(*r["k"])));
```



3. Evalúe la calidad del ajuste y discuta la validez del modelo.

A simple vista, el ajuste parece ser bueno. En los residuos, no se observa ninguna estructura ni parecería haber *outliers*.

Podemos realizar el calculo del χ^2 reducido para evaluar la "dispersión promedio" de los residuos, que esperamos que este alrededor de 1:

```
residuos = r.y - r.func(r.x, *r.params)
chi2 = np.sum((residuos / r.y_err) ** 2)
chi2_reducido = chi2 / (r.y.size - r.params.size)
chi2_reducido
```

```
0.9328220629234224
```

Problema 2

Con el fin de comprar un mantel, se realizaron las siguientes mediciones del diámetro D de una mesa con una cinta métrica de apreciación 1 mm:

	0	1	2	3	4	5	6	7	8	9
D (cm)	36.0	136.1	136.3	135.6	136.0	136.1	136.2	136.3	136.0	135.9

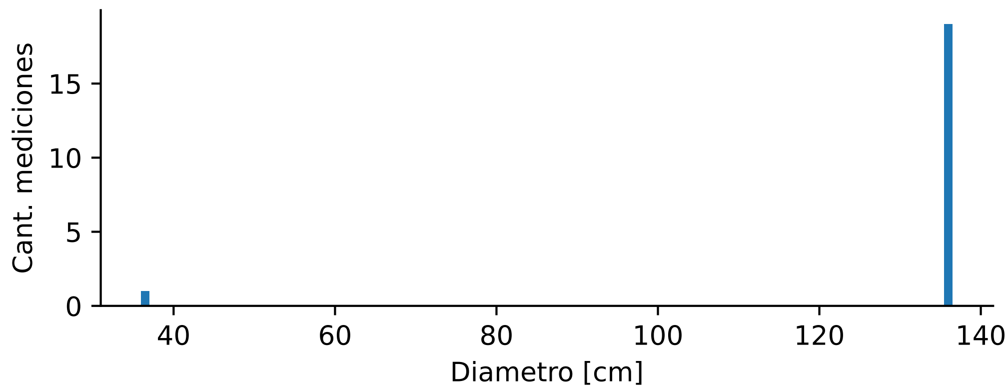
	10	11	12	13	14	15	16	17	18	19
D (cm)	135.8	135.7	136.2	136.5	136.4	136.4	135.8	135.9	135.7	135.6

Puede cargar los datos con:

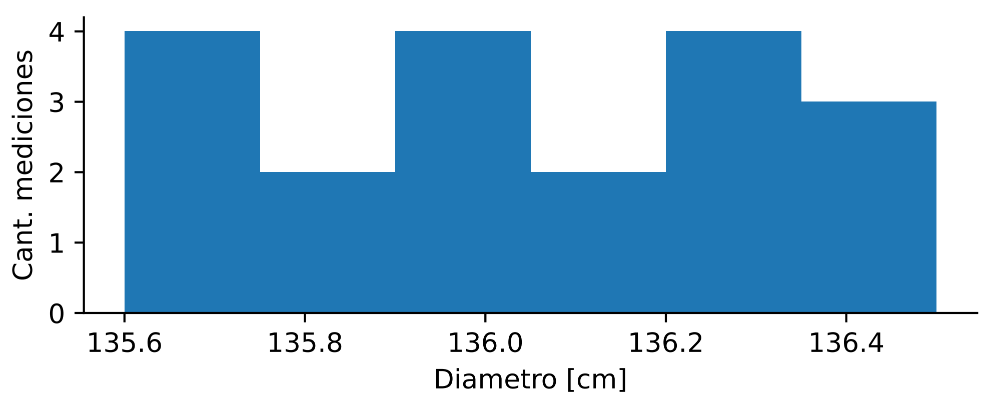
```python
import numpy as np

datos = np.loadtxt("problema2.csv", skiprows=1)
```

1. Realicen un histograma.

```python
import matplotlib.pyplot as plt
plt.rc("figure", figsize=(6, 2))
plt.rc("axes.spines", top=False, right=False)

plt.hist(datos, bins=100)
plt.xlabel("Diametro [cm]")
plt.ylabel("Cant. mediciones")
None
```




```
datos_validos = datos[datos > 100]

plt.hist(datos_validos, bins="sturges")
plt.xlabel("Diametro [cm]")
plt.ylabel("Cant. mediciones")
None
```



2. Con los datos que consideren válidos, calculen la media y la desviación estándar.

```
media = np.mean(datos_validos)
desv_est = np.std(datos_validos)
error_media = desv_est / np.size(datos_validos)**0.5

print("     media = ({} ± {}) cm".format(*labo1.to_significant_figures(media, error_media)))
print("desv. est. = {} cm".format(labo1.to_significant_figures(desv_est)))
```

```
     media = (136.026 ± 0.063) cm
desv. est. = 0.27 cm
```

3. Reporten el valor medido del diámetro de la mesa con su incerteza.

Como considero que la variación en la medición se debe a errores aleatorios, puedo tomar el promedio con su error como resultado (136.026 ± 0.063) cm

4. ¿Cómo esperarían que cambien los puntos anteriores (1, 2 y 3) si realizaran 100 mediciones en total?

Rescpecto al punto 1, esperaría que el histograma tenga una altura mayor por tener más mediciones, pero mantenga su forma.

Respecto al punto 2, la media $\bar{x}$ y la desviación estándar σ no deberían cambiar considerablemente, mientras que el error en la media se reduciría a $\sigma/\sqrt{100} \approx 0.027$ cm ya que aumentó la cantidad de mediciones.

Respecto al punto 3, podríamos determinar el diametro de la mesa con mayor precisión.

> ⚠ Advertencia
>
> Todo esto asume que los errores son aproximadamente simétricos respecto del valor real. Al medir un diámetro, podría darse que las mediciones sean iguales o mayores al valor real.