Determinando la aceleración

Con el Tracker

Derivadas númericas

El Tracker mide x_t, y calcula:

Velocidad

$$v_t = \frac{x_t - x_{t-1}}{\Delta t}$$

Aceleración

$$a_t = \frac{v_t - v_{t-1}}{\Delta t}$$

$$= \frac{\frac{x_t - x_{t-1}}{\Delta t} - \frac{x_{t-1} - x_{t-2}}{\Delta t}}{\Delta t}$$

$$= \frac{x_t - 2x_{t-1} + x_{t-2}}{\Delta t^2}$$

Promedio de aceleraciones

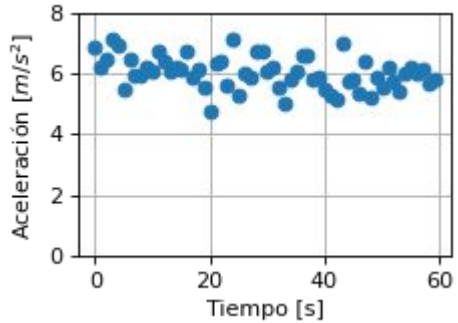
Hipótesis: la aceleración es constante.

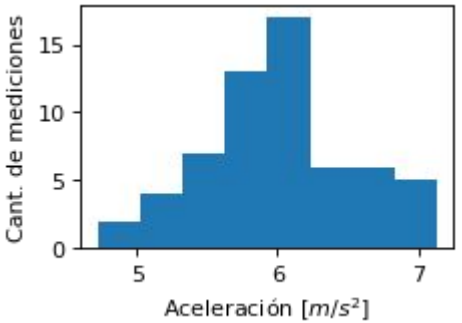
$$\bar{a} = \frac{1}{N-3} \sum_{i=2}^{N} a_i \qquad a_t = \frac{x_t - 2x_{t-1} + x_{t-2}}{\Delta t^2}$$

$$= \frac{1}{(N-3)\Delta t^2} \sum_t x_t - 2x_{t-1} + x_{t-2}$$

$$= \frac{1}{(N-3)\Delta t} \left(x_N - x_{N-1} - x_2 + x_1 \right)$$

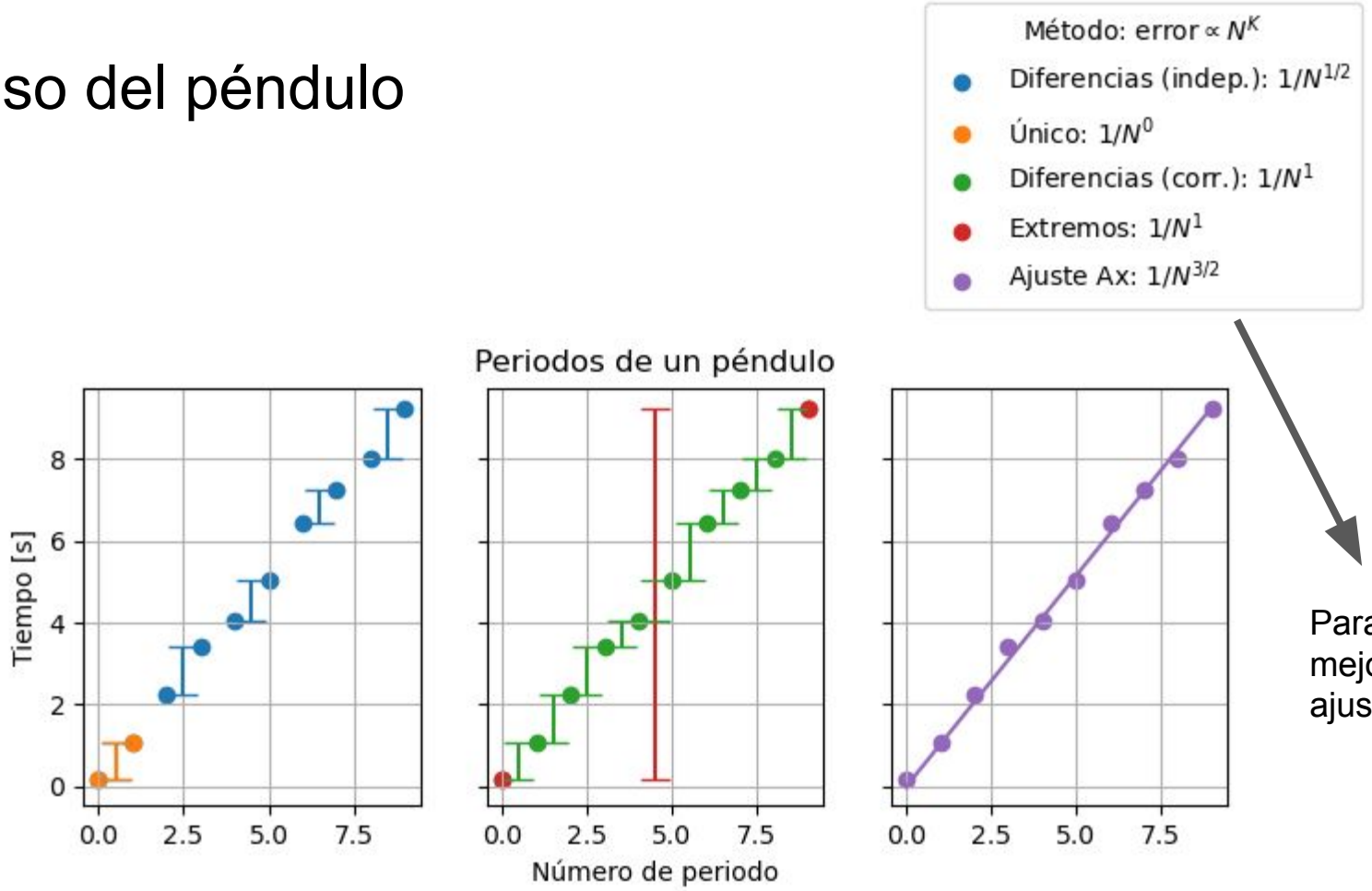
$$= \frac{v_N - v_2}{(N-3)\Delta t}$$





De los N valores, solo usamos 4 valores de posición de los extremos.

Caso del péndulo



Para otro día: era mejor hacer un ajuste Ax+B.

Modelo físico

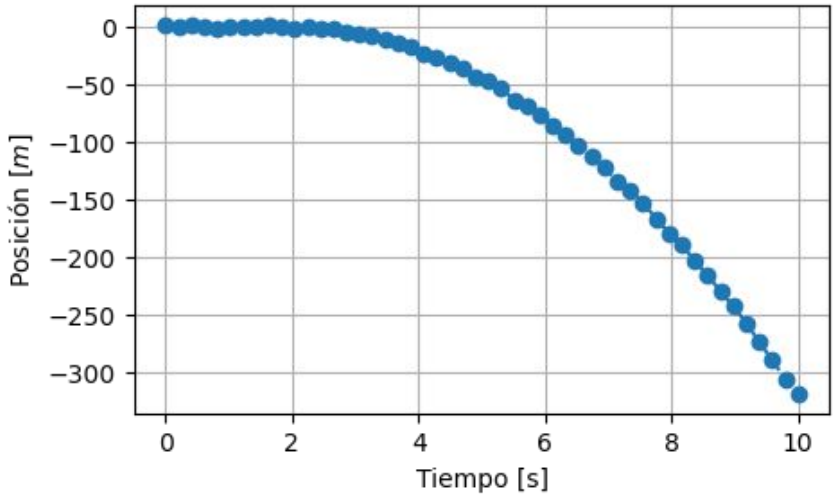
Si la aceleración es constante, $a(t) = a$

Entonces,

$$v(t) = at + v_0$$

$$x(t) = \frac{1}{2}at^2 + v_0 t + x_0$$

¿Incluimos v_0 y x_0?



Ajuste cuadrático

```
datos = np.loadtxt(...)  # Cargan
datos = datos[desde:hasta]  # Recortan
t, x, y = datos[:,0], datos[:,1], datos[:,2]

# Ajustan
def cuadratica(x, a, b, c):
    return a * x**2 + b * x + c

params, cov = curve_fit(cuadratica, t, y)
sigmas = np.sqrt(np.diag(cov))

a, error_a = params[0], sigmas[0]
```

Importante: a *curve_fit* le tienen que llegar los datos recortados, donde sigue una parábola.

No lo vamos a usar hoy, pero…

¿Cómo sabe *curve_fit* cuál es el error de la aceleración si no le dimos los errores en **y**?

Extras

Error de las mediciones a partir de los residuos

Cuadrados mínimos para una constante:

$$S(p) = \sum_i \Big(y_i - f(x_i)\Big)^2 = \sum_i \Big(y_i - p\Big)^2$$

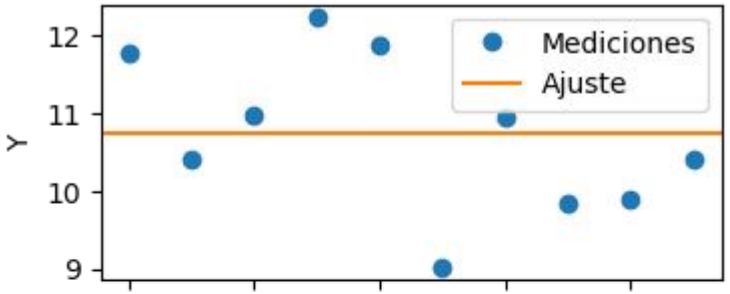
El p óptimo:

$$p_{min} = \bar{y} = \frac{1}{N}\sum_i y_i$$

Desviación estándar:

$$s_y = \sqrt{\frac{1}{N}\sum_i (y_i - \bar{y})^2} = \sqrt{\frac{1}{N} S(\bar{y})}$$

Pero: $$S(\bar{y}) = \sum_i r_i^2$$



La desviación estándar se calcula de los residuos.

Error de las mediciones a partir de los residuos

Cuadrados mínimos para una lineal:

$$S(p) = \sum_i \left(y_i - f(x_i)\right)^2 = \sum_i \left(y_i - p\right)^2$$

Ax

El p óptimo:

$$p_{min} = \bar{y} = \frac{1}{N}\sum_i y_i$$

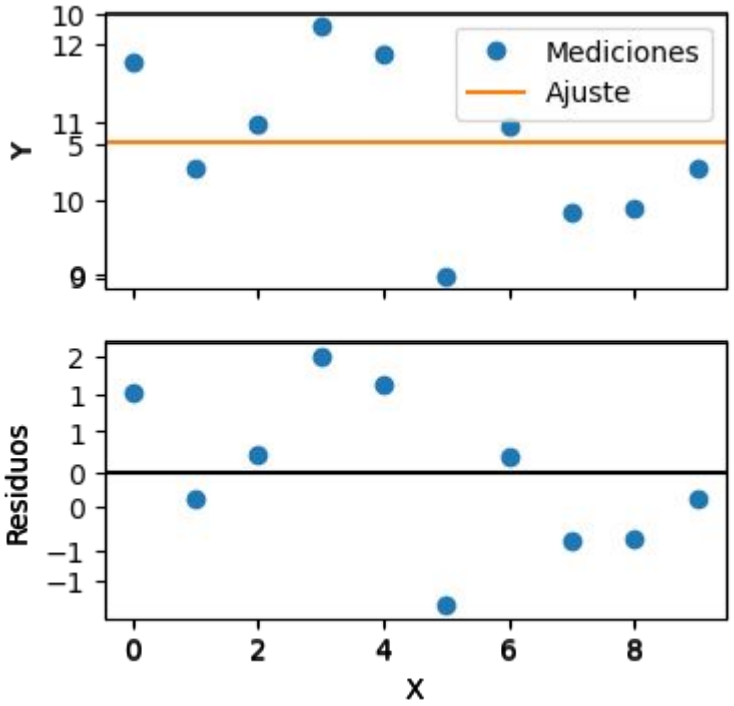
A_{min}

Desviación estándar:

$$s_y = \sqrt{\frac{1}{N}\sum_i (y_i - \bar{y})^2} = \sqrt{\frac{1}{N} S(\bar{y})}$$

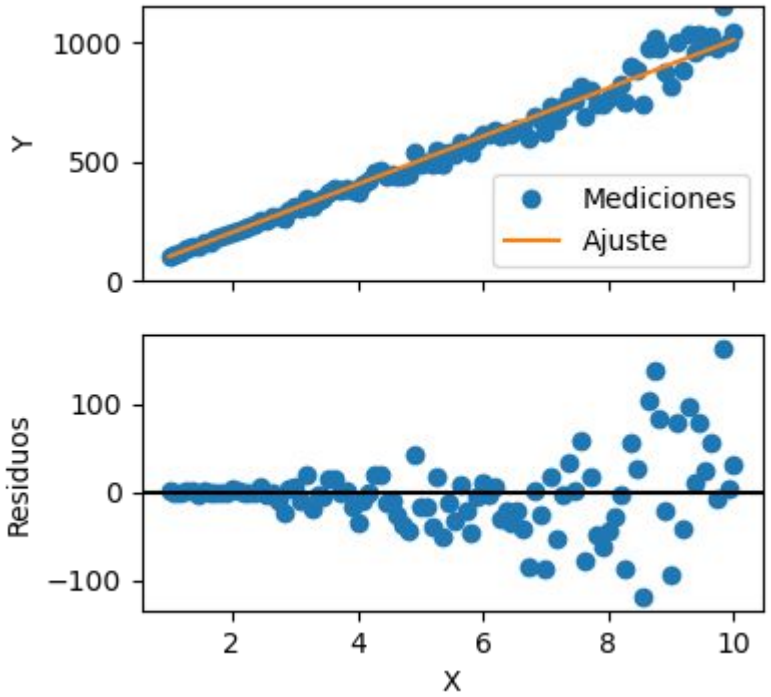
Pero: $S(\bar{y}) = \sum_i r_i^2$

$A_{min}\, x$



La desviación estándar se calcula de los residuos.

¡Cuidado! Errores distintos



El error depende de x (o de y).

Ponderamos la suma de cuadrados mínimos:

$$\chi^2 = \sum_i \left(\frac{y_i - f(x_i)}{\sigma_i} \right)^2$$

En Python:

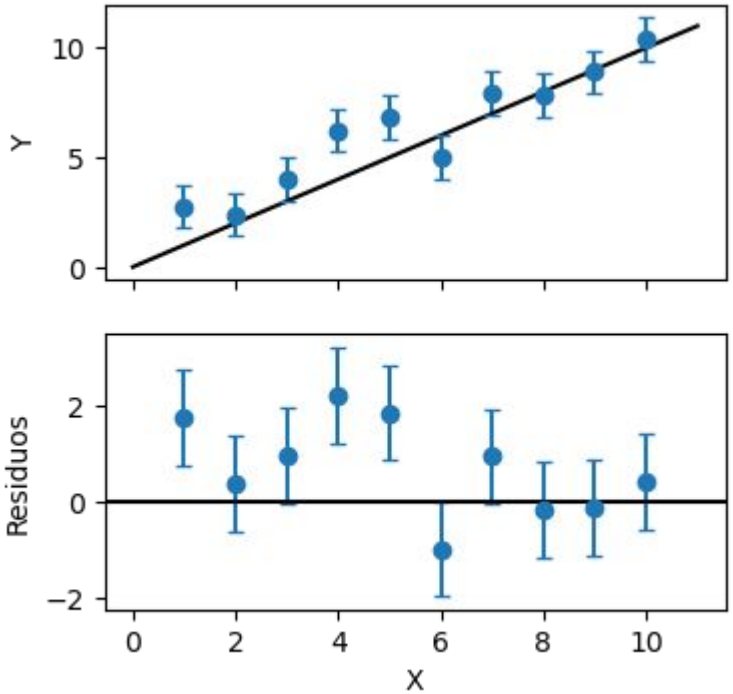
```
curve_fit(funcion, t, y)

curve_fit(funcion, t, y, sigma=y_err)
```

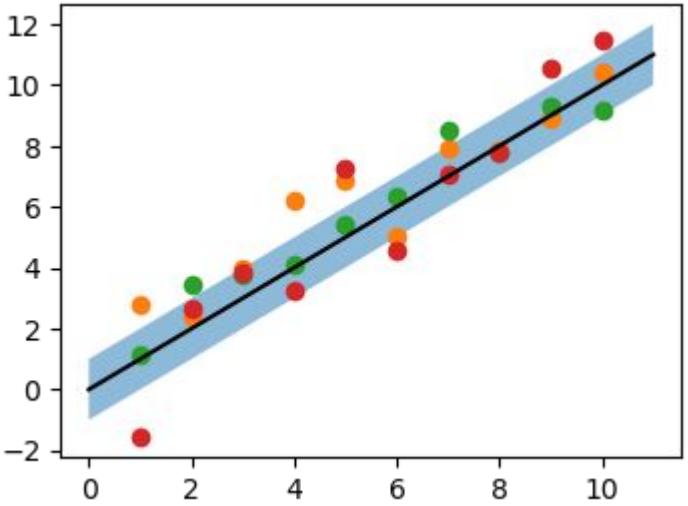
Los errores **y_err** son relativos.

Extras

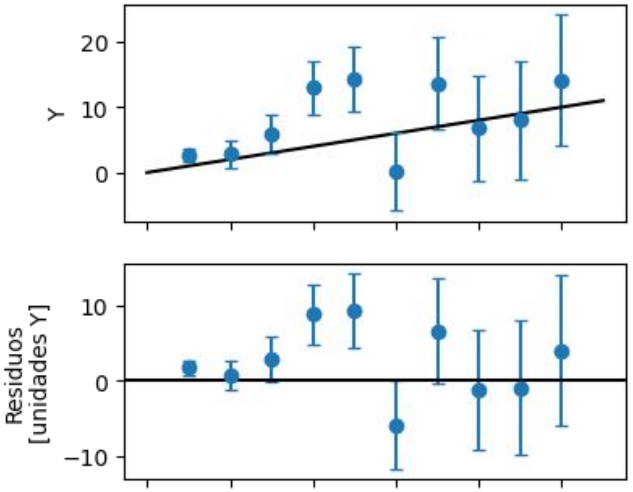
¿Cuántos puntos tienen que “coincidir” con la recta?



Aproximadamente el 68% (si el error es gaussiano)



Extras: residuos normalizados



$$\chi^2 = \sum_i \left(\frac{y_i - f(x_i)}{\sigma_i} \right)^2$$

$$r_i = y_i - f(x_i)$$

$$r_i = \frac{y_i - f(x_i)}{\sigma_i}$$

Extras: ajuste no lineal

En cuadrados mínimos, se diferencia entre funciones lineales y no lineales.

¿Qué función es lineal?

$$y(x) = Ax$$

$$y(x) = Ax^2 + Bx + C$$

$$y(\alpha) = A\tan(\alpha - B)$$

Lineal en los parámetros:

$$\frac{\partial^2 y}{\partial A^2} = 0$$

Si no es lineal, hay que “ayudar” a la función dándole parámetros iniciales:

```
def funcion(x, A, B):
    return A * np.tan(x - B)

curve_fit(funcion, t, y, p0=[10, 0.5])
```