Universidad Industrial de Santander

Escuela de Física

Asignación: Momento de Inercia Para N partículas

Mecánica Clásica

Autor:
Alejandro Martínez Portilla

Santander Colombia
Mayo 2025

Introducción

En este trabajo se analizan los momentos de orden cero (masa total), uno (centro de masa) y dos (tensor de inercia), así como la diagonalización del tensor (cálculo de autovalores y autovectores) para un sistema de partículas cuyas masas y posiciones vienen dadas en el archivo `datosmasas.csv`. Primero se aborda el caso **2D** (planos x, y) y luego el caso **3D** (coordenadas x, y, z).

1 Caso 2D

1.1 Datos del sistema

El archivo `datosmasas.csv` contiene n partículas enumeradas con su número de partícula, masa y coordenadas (x, y, z). En el caso 2D nos enfocamos solo en las dos primeras coordenadas (x, y).

Renombramos internamente `masas` a `mass` (masa de cada partícula).

1.2 Momento de orden cero: Masa total

Definición:

$$M = \sum_{i=1}^{n} m_i,$$

donde m_i es la masa de la partícula i.

Cálculo numérico en Python:

```
import pandas as pd
import numpy as np

# Cargar datos:
df = pd.read_csv('datosmasas.csv')
df = df.rename(columns={'masas':'mass'})

# Masa total:
M = df['mass'].sum()
print("Masa␣total␣M␣=", M)
```

Resultado obtenido:

$$\boxed{M = 4627.0}$$

1.3 Momento de orden uno: Centro de masa

El centro de masa $\mathbf{r}_{CM} = (x_{CM}, y_{CM})$ se define como:

$$x_{CM} = \frac{1}{M}\sum_{i=1}^{n} m_i\, x_i, \qquad y_{CM} = \frac{1}{M}\sum_{i=1}^{n} m_i\, y_i.$$

En estas expresiones, (x_i, y_i) son las coordenadas de la partícula i.

Cálculo numérico en Python:

```
# Centro de masa 2D
x_cm = (df['mass'] * df['x']).sum() / M
y_cm = (df['mass'] * df['y']).sum() / M
print("Centro de masa 2D = (", x_cm, ",", y_cm, ")")
```

Resultado obtenido:

$$\boxed{(x_{CM},\, y_{CM}) = (\,825.8152150421439,\; 776.9185217203371\,).}$$

Físicamente, este punto es aquel en que se concentraría toda la masa para que el sistema se comporte ante fuerzas externas de la misma manera que la distribución original.

1.4 Momento de orden dos: Tensor de inercia en 2D

Trasladamos el origen al centro de masa: $\mathbf{r}'_i = (x'_i, y'_i) = (x_i - x_{CM},\, y_i - y_{CM})$. El tensor de inercia en 2D se define como

$$\mathbf{I}_{2D} \;=\; \sum_{i=1}^{n} m_i \Big[(x_i'^2 + y_i'^2)\, \mathbf{I}_2 \;-\; \begin{pmatrix} x_i'^2 & x'_i y'_i \\ x'_i y'_i & y_i'^2 \end{pmatrix} \Big],$$

donde $\mathbf{I}_2$ es la matriz identidad 2×2. De este modo, las componentes quedan:

$$I_{xx} = \sum_{i=1}^{n} m_i\, y_i'^2,$$

$$I_{yy} = \sum_{i=1}^{n} m_i\, x_i'^2,$$

$$I_{xy} = I_{yx} \;=\; -\sum_{i=1}^{n} m_i\, x'_i\, y'_i.$$

Cálculo numérico en Python:

```
# Posiciones relativas 2D
x_rel = df['x'] - x_cm
y_rel = df['y'] - y_cm

# Componentes del tensor de inercia 2D
I_xx = (df['mass'] * y_rel**2).sum()
I_yy = (df['mass'] * x_rel**2).sum()
I_xy = - (df['mass'] * x_rel * y_rel).sum()

I_tensor_2d = np.array([[I_xx, I_xy],
                        [I_xy, I_yy]])
print("Tensor de inercia 2D =")
print(I_tensor_2d)
```

Resultado obtenido:

$$\mathbf{I}_{2D} = \begin{pmatrix} I_{xx} & I_{xy} \\ I_{xy} & I_{yy} \end{pmatrix} = \begin{pmatrix} 9.63660148 \times 10^{8} & -9.11747911 \times 10^{8} \\ -9.11747911 \times 10^{8} & 9.58535589 \times 10^{8} \end{pmatrix}.$$

Interpretación: I_{xx} es la resistencia a rotar alrededor del eje x (en el plano), I_{yy} la resistencia a rotar alrededor del eje y, y I_{xy} mide el acoplamiento entre rotaciones en x e y. Como $I_{xy} \neq 0$, la base cartesiana no diagonaliza el tensor.

1.5 Diagonalización: autovalores y autovectores en 2D

Al ser $\mathbf{I}_{2D}$ simétrica real, sus autovalores son reales y los autovectores ortonormales. Resolvemos

$$\mathbf{I}_{2D}\,\mathbf{v} = \lambda\,\mathbf{v}.$$

Cálculo numérico en Python:

```
# Autovalores y autovectores 2D
eigvals_2d, eigvecs_2d = np.linalg.eig(I_tensor_2d)
print("Autovalores 2D =", eigvals_2d)
print("Autovectores 2D (columnas) =")
print(eigvecs_2d)
```

Resultados obtenidos:

$$\text{Autovalores 2D: } \lambda_1 \approx 1.87284938 \times 10^9, \quad \lambda_2 \approx 4.93463569 \times 10^7.$$

$$\text{Matriz de autovectores 2D: } V = \begin{pmatrix} 0.70809967 & 0.70611250 \\ -0.70611250 & 0.70809967 \end{pmatrix}.$$

Cada columna de V es un autovector unitario ortogonal al otro. Por ejemplo, $\mathbf{v}_1 = (0.70809967, -0.70611250)$ y $\mathbf{v}_2 = (0.70611250, 0.70809967)$.

La base cartesiana no es propia para éste sistema

Dado que en $\mathbf{I}_{2D}$ el término $I_{xy} \neq 0$, la matriz no está diagonal en la base $\{(1,0),(0,1)\}$. Por tanto, la base cartesiana $\{\mathbf{e}_x, \mathbf{e}_y\}$ *no* es una base propia. En cambio, la base de *autovectores* $\{\mathbf{v}_1, \mathbf{v}_2\}$ es la base ortonormal de ejes principales de inercia.

Matriz de transformación a la base de autovectores

La matriz de transfomación de la base cartesiana que da lugar a una matriz de inercia diagonal, es decir, la forma más simple de describir la distribución de masas es :

$$V = \begin{pmatrix} 0.70809967 & 0.70611250 \\ -0.70611250 & 0.70809967 \end{pmatrix}.$$

Como V es ortonormal ($V^T V = I$), su inversa es $V^{-1} = V^T$. Entonces

$$V^T\,\mathbf{I}_{2D}\,V \;=\; \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix}.$$

Esta matriz diagonal contiene los momentos principales de inercia.

2 Caso 3D

2.1 Momento de orden cero y centro de masa

En 3D, para cada partícula i con masa m_i y coordenadas (x_i, y_i, z_i), el proceso es análogo:

$$M = \sum_{i=1}^{n} m_i, \quad (x_{CM}, y_{CM}, z_{CM}) = \left(\frac{\sum_i m_i x_i}{M}, \frac{\sum_i m_i y_i}{M}, \frac{\sum_i m_i z_i}{M} \right).$$

Cálculo numérico en Python (ya tenemos x_{CM} y y_{CM}, solo faltaba z_{CM}):

```
# Centro de masa 3D
z_cm = (df['mass'] * df['z']).sum() / M
print("Centro de masa 3D = (", x_cm, ",", y_cm, ",", z_cm, ")")
```

Resultado obtenido:

$$(x_{CM},\, y_{CM},\, z_{CM}) = (\,825.8152150421439,\ 776.9185217203371,\ 15.503349902744759\,).$$

2.2 Tensor de inercia en 3D

Definimos las posiciones relativas al centro de masa $\mathbf{r}'_i = (x'_i, y'_i, z'_i) = (x_i - x_{CM},\, y_i - y_{CM},\, z_i - z_{CM})$. El tensor de inercia 3D es:

$$\mathbf{I}_{3D} = \sum_{i=1}^{n} m_i \left[(x_i'^2 + y_i'^2 + z_i'^2)\, \mathbf{I}_3 - \begin{pmatrix} x_i'^2 & x'_i y'_i & x'_i z'_i \\ y'_i x'_i & y_i'^2 & y'_i z'_i \\ z'_i x'_i & z'_i y'_i & z_i'^2 \end{pmatrix} \right],$$

donde $\mathbf{I}_3$ es la identidad 3×3. Entonces las componentes se obtienen explícitamente como:

$$\begin{aligned}
I_{xx} &= \sum_i m_i\, (y_i'^2 + z_i'^2), \\
I_{yy} &= \sum_i m_i\, (x_i'^2 + z_i'^2), \\
I_{zz} &= \sum_i m_i\, (x_i'^2 + y_i'^2), \\
I_{xy} &= I_{yx} = -\sum_i m_i\, x'_i\, y'_i, \\
I_{xz} &= I_{zx} = -\sum_i m_i\, x'_i\, z'_i, \\
I_{yz} &= I_{zy} = -\sum_i m_i\, y'_i\, z'_i.
\end{aligned}$$

Cálculo numérico en Python:

```
# Posiciones relativas 3D
x_rel3 = df['x'] - x_cm
y_rel3 = df['y'] - y_cm
z_rel3 = df['z'] - z_cm

# Construccin del tensor de inercia 3D
I_tensor_3d = np.zeros((3,3))
for i in range(len(df)):
    m = df.loc[i, 'mass']
    r = np.array([x_rel3[i], y_rel3[i], z_rel3[i]])
    I_tensor_3d += m * (np.dot(r, r) * np.eye(3) - np.outer(r, r))
```

```
print("Tensor de inercia 3D =")
print(I_tensor_3d)
```

Resultado obtenido:

$$\mathbf{I}_{3D} = \begin{pmatrix} 1.065503468 \times 10^9 & -9.11747911 \times 10^8 & 7.14204864 \times 10^6 \\ -9.11747911 \times 10^8 & 1.060378910 \times 10^9 & 1.92959724 \times 10^6 \\ 7.14204864 \times 10^6 & 1.92959724 \times 10^6 & 1.922195737 \times 10^9 \end{pmatrix}.$$

Dado que hay términos no nulos fuera de la diagonal ($I_{xy}, I_{xz}, I_{yz} \neq 0$), la base cartesiana no es base propia de $\mathbf{I}_{3D}$.

2.3 Diagonalización: autovalores y autovectores en 3D

Para $\mathbf{I}_{3D}$ (matriz simétrica real 3×3), resolvemos $\mathbf{I}_{3D}\,\mathbf{v} = \lambda\,\mathbf{v}$.

Cálculo numérico en Python:

```
# Autovalores y autovectores 3D
eigvals_3d, eigvecs_3d = np.linalg.eig(I_tensor_3d)
print("Autovalores 3D =", eigvals_3d)
print("Autovectores 3D (columnas) =")
print(eigvecs_3d)
```

Resultados obtenidos:

$$\text{Autovalores 3D: } \begin{cases} \lambda_1 \approx 1.51166481 \times 10^8, \\ \lambda_2 \approx 1.97495158 \times 10^9, \\ \lambda_3 \approx 1.92196006 \times 10^9. \end{cases}$$

$$\text{Matriz de autovectores 3D: } V_3 = \begin{pmatrix} -0.70611307 & -0.70654139 & -0.04694255 \\ -0.70808985 & 0.70421170 & 0.05190995 \\ 0.00361904 & -0.06989384 & 0.99754787 \end{pmatrix}.$$

Cada columna de V_3 es un autovector unitario (ortogonal al resto).

¿La base cartesiana es propia?

Dado que en $\mathbf{I}_{3D}$ los productos de inercia (I_{xy}, I_{xz}, I_{yz}) no son todos cero, la base cartesiana $\{\mathbf{e}_x, \mathbf{e}_y, \mathbf{e}_z\}$ *no* es una base propia. En cambio, la base dada por los autovectores $\{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3\}$ es la base ortonormal de ejes principales de inercia.

Matriz de transformación a la base de autovectores

Sea

$$V_3 \;=\; [\,\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3\,] = \begin{pmatrix} -0.70611307 & -0.70654139 & -0.04694255 \\ -0.70808985 & 0.70421170 & 0.05190995 \\ 0.00361904 & -0.06989384 & 0.99754787 \end{pmatrix}.$$

Como V_3 está ortonormalizado ($V_3^T V_3 = I_3$), su inversa es V_3^T. Entonces

$$V_3^T \, \mathbf{I}_{3D} \, V_3 \;=\; \begin{pmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{pmatrix}.$$

Los vectores $\{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3\}$ son los *ejes principales de inercia.*

Conclusiones

- Se calculó la masa total del sistema discreto: $M = 4627.0$.
- El centro de masa en 2D es $(x_{CM}, y_{CM}) = (825.8152,\ 776.9185)$ y en 3D (825.8152, 776.9185, 15.50
- El tensor de inercia en 2D no está diagonal en la base cartesiana; sus componentes son:
$$I_{2D} = \begin{pmatrix} 9.63660148 \times 10^8 & -9.11747911 \times 10^8 \\ -9.11747911 \times 10^8 & 9.58535589 \times 10^8 \end{pmatrix}.$$
- En 2D, los autovalores (momentos principales) son $\lambda_1 \approx 1.8728 \times 10^9$ y $\lambda_2 \approx 4.9346 \times 10^7$. La base cartesiana no es base propia, y la base de autovectores (matriz V) es:
$$V = \begin{pmatrix} 0.70809967 & 0.70611250 \\ -0.70611250 & 0.70809967 \end{pmatrix}.$$
- El tensor de inercia en 3D tampoco es diagonal en la base cartesiana; sus valores numéricos son:
$$I_{3D} = \begin{pmatrix} 1.065503468 \times 10^9 & -9.11747911 \times 10^8 & 7.14204864 \times 10^6 \\ -9.11747911 \times 10^8 & 1.060378910 \times 10^9 & 1.92959724 \times 10^6 \\ 7.14204864 \times 10^6 & 1.92959724 \times 10^6 & 1.922195737 \times 10^9 \end{pmatrix}.$$
- En 3D, los autovalores principales de inercia son $\{1.5117 \times 10^8,\ 1.9749 \times 10^9,\ 1.9220 \times 10^9\}$. La base de autovectores (matriz V_3) es:
$$V_3 = \begin{pmatrix} -0.70611307 & -0.70654139 & -0.04694255 \\ -0.70808985 & 0.70421170 & 0.05190995 \\ 0.00361904 & -0.06989384 & 0.99754787 \end{pmatrix}.$$
- La matriz de cambio de base cartesiana $\rightarrow$ ejes principales es V^T en 2D y V_3^T en 3D, lo que diagonaliza los tensores y revela los ejes principales de inercia.

References

[1] Goldstein, H., Poole, C., & Safko, J. (2002). *Classical Mechanics* (3rd ed.). Addison-Wesley. Capítulo 5: "The Rigid Body Equations of Motion" (Tensor de inercia, Sección 5.3).