Wild Matrix

Matrices en la vida real

Sebastian Flores
Chief Data Officer | u-planner

Esta charla está disponible en github.com/sebastiandres/charlas para libre descarga.
Puede verse en línea:

- De manera interactiva con binder: **https://bit.ly/2Ub89mU**
- De manera estática en el mismo github: **https://bit.ly/33WmSa2**

¿Data Science?

Data science es una ciencia interdisciplinaria, que se apoya en conocimientos de Matemática, Estadística, Informática, entre otros.

Es como ciencia forense, pero en lugar de cadáveres, se investigan datos.

Es como Jurassic Park, donde se tiene parte del ADN de cada dinosaurio, y para poder completar la información se utilizó otro ADN compatible.

¿Data Science?

Algo que aprendí muy tarde

Las matrices representan combinaciones lineales.
Nada más que eso.
Pero eso es **suficientemente** poderoso.

Combinación lineal de filas

Multiplicar un vector por una matriz permite obtener una combinación lineal de las distintas filas.

Seleccionar la primera fila

$$\begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} a & b \end{pmatrix}$$

Seleccionar la segunda fila

$$\begin{pmatrix} 0 & 1 \end{pmatrix} \begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} c & d \end{pmatrix}$$

Combinación lineal de filas

Multiplicar un vector por una matriz permite obtener una combinación lineal de las distintas filas.

Mezclar ambas filas

$$\begin{pmatrix} \alpha & \beta \end{pmatrix} \begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} \alpha a + \beta c & \alpha b + \beta d \end{pmatrix} = \alpha \begin{pmatrix} a & b \end{pmatrix} + \beta \begin{pmatrix} c & d \end{pmatrix}$$

Combinación lineal de columnas

Multiplicar una matriz por un vector columna (por la derecha) permite obtener una combinación lineal de las columnas de la matriz.

Seleccionar la primera columna

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} a \\ c \end{pmatrix}$$

Seleccionar la segunda columna

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} b \\ d \end{pmatrix}$$

Combinación lineal de columnas

Multiplicar una matriz por un vector columna (por la derecha) permite obtener una combinación lineal de las columnas de la matriz.

Combinación de columnas

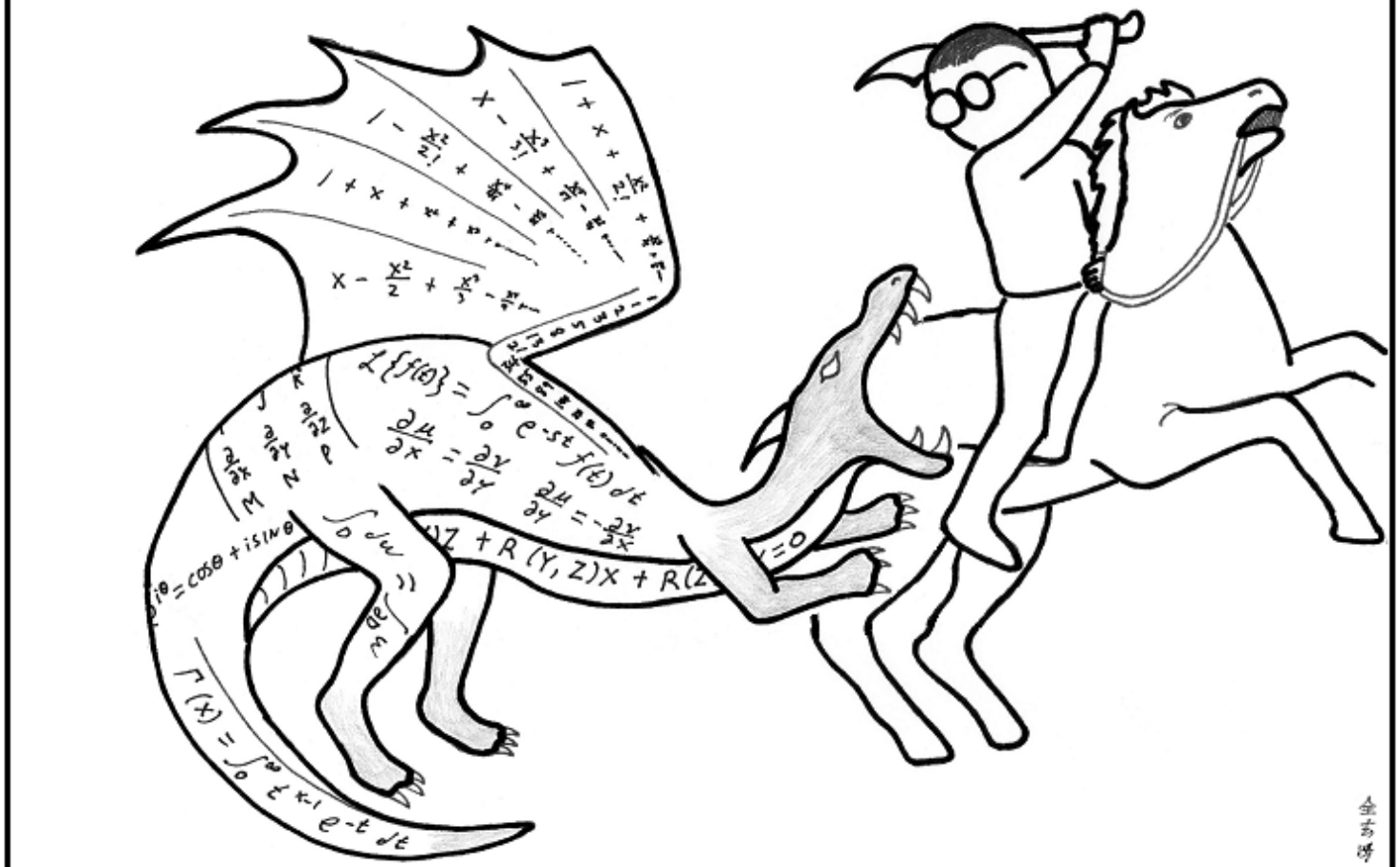
$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} \alpha a + \beta b \\ \alpha c + \beta d \end{pmatrix} = \alpha \begin{pmatrix} a \\ c \end{pmatrix} + \beta \begin{pmatrix} b \\ d \end{pmatrix}$$

¿Porqué aprender Algebra Lineal?

Sólo podemos resolver problemas lineales.
Todo lo demás es reducir el problema a algo que se parezca o se aproxime a un problema lineal.
Desde lo más "clásico" (sistemas de ecuaciones, ecuaciones diferenciales ordinarias, ecuaciones diferenciales parciales, etc.) hasta lo más moderno (inteligencia artificial y Machine Learning): **todo depende de matrices y vectores de datos, cada vez de mayor tamaño.**

¿Cómo aprender Algebra Lineal?

HOW TO STUDY MATH



Don't just read it; fight it!

--- Paul R. Halmos

Matrices en u-exam

El producto u-exam busca asignar de manera eficiente y sin conflictos las evaluaciones de los estudiantes. Cada estudiante toma varias asignaturas, y se necesita que en una única semana se agenden las evaluaciones para cada asignatura:

- Todos los estudiantes deben tener asignadas todas sus evaluaciones, que deben tener asignado un día, hora, sala y supervisor.
- Las evaluaciones de cada estudiante deben repartirse en la semana, y sobre todo, no pueden ocurrir al mismo tiempo. * Para cada asignatura, la evaluación debe quedar en una único día y hora, para todos los estudiantes y en todos los campus, simultáneamente.
- Los estudiantes de una misma sección, deben mantenerse reunidos en la evaluación y deben asignarse a una sala disponible de tamaño y características apropiadas.

- El día y hora asignado a las evaluaciones deben ser compatibles con la disponibilidad de los supervisores de la evaluación.

u-exam | notación

Importante: Una parte crucial del modelamiento de un problema es utilizar una notación apropiada.
En nuestro caso hablaremos de:

- Estudiantes o alumnos.
- Cursos, ramos, materias o asignaturas.

u-exam | notación

- Usar como notación $N_{estudiantes}$ y $N_{asignaturas}$ para los conteos de estudiantes y asignaturas se hará demasiado largo en las fórmulas y sumatorias, aunque haría más legible el código numérico.
- Usar la notación N_e y N_a es ambigua, porque la letra "a" se usa para alumno y asignatura.
- Una convención neutra es utilizar N_e y N_c que es directa de interpretar (ramos/materias no se entiende en otros países).

u-exam | notación

Definamos la matriz E (del inglés "enrollments") de dimensiones N_e estudiantes y N_c asignaturas, donde:

- El elemento $E_{i,j}$ tiene el valor 1 si el estudiante i está tomando la asignatura j.
- El elemento $E_{i,j}$ tiene el valor 0 si el estudiante i NO está tomando la asignatura j.

¿Cuántas filas hay? ¿Cuántas columnas hay?

Por ejemplo, consideremos los siguientes asignaturas:

- CORE102: CIVILIZACIÓN CONTEMPORÁNEA II
- MAT106: CÁLCULO INTEGRAL
- MAT113: ÁLGEBRA LINEAL
- FIS101: FÍSICA I
- TEI101: TALLER EL ARTE DE LA INGENIERÍA
- LID101: LIDERAZGO I
- GYM102: EDUCACIÓN FÍSICA II

Y consideremos los siguiente alumnos:

- Tony Stark
- Thor Odinson
- Henry Pym
- Janet van Dyne
- Bruce Banner

¿Cuánto vale N_e? ¿Cuánto vale N_c?

$N_e = 5$ estudiantes y $N_c = 7$ asignaturas.

Supongamos que toman los siguientes cursos:

- Tony Stark: MAT106, MAT113, FIS101, LID101
- Thor Odinson: CORE102, TE101, LID101, GYM102
- Henry Pym: CORE102, MAT106, MAT113, FIS101, TE101, LID101, GYM102
- Janet van Dyne: CORE102, MAT106, MAT113, FIS101, TE101, LID101, GYM102
- Bruce Banner: CORE102, MAT106, MAT113, FIS101, TE101

¿Cómo quedaría la matriz?

¡Depende de cómo ordenemos los estudiantes y los cursos!

Consideremos el siguiente orden
CORE102, MAT106, MAT113, FIS101, TE101, LID101, GYM102:

- Tony Stark: (MAT106, MAT113, FIS101, LID101) -> (0,1,1,1,0,1,0)
- Thor Odinson: (CORE102, TE101, LID101, GYM102) -> (1,0,0,0,1,1,1)
- Henry Pym: (CORE102, MAT106, MAT113, FIS101, TE101, LID101, GYM102) -> (1,1,1,1,1,1,1)
- Janet van Dyne: (CORE102, MAT106, MAT113, FIS101, TE101, LID101, GYM102) -> (1,1,1,1,1,1,1)
- Bruce Banner: (CORE102, MAT106, MAT113, FIS101, TE101) -> (1,1,1,1,1,0,0)

Obtenemos entonces la siguiente matriz:

$$E = \begin{pmatrix} 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 \end{pmatrix}$$

¿Cómo contar la cantidad de inscripciones por estudiante?
¿Cantidad de inscripciones por asignatura?
¿Cantidad de inscripciones totales?

u-exam: contando inscripciones por estudiante

Para contar las inscripciones por estudiante, necesitamos sumar los elementos de cada fila. Eso es equivalente a ...

¡una multiplicación con un vector por la derecha!

$$E = \begin{pmatrix} 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 4 \\ 4 \\ 7 \\ 7 \\ 5 \end{pmatrix}$$

u-exam: contando los estudiantes por curso

Para contar la cantidad de estudiante por curso, necesitamos sumar los elementos de cada columna. Eso es equivalente a ...

¡una multiplicación con un vector fila por la izquierda!

$$E = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 4 & 4 & 4 & 4 & 4 & 4 & 3 \end{pmatrix}$$

u-exam: contando las inscripciones totales

¿Cómo podríamos contar el total de inscripciones?

$$\begin{pmatrix}1 & 1 & 1 & 1 & 1\end{pmatrix}\begin{pmatrix}0 & 1 & 1 & 1 & 0 & 1 & 0\\ 1 & 0 & 0 & 0 & 1 & 1 & 1\\ 1 & 1 & 1 & 1 & 1 & 1 & 1\\ 1 & 1 & 1 & 1 & 1 & 1 & 1\\ 1 & 1 & 1 & 1 & 1 & 0 & 0\end{pmatrix}\begin{pmatrix}1\\1\\1\\1\\1\\1\\1\end{pmatrix} = \begin{pmatrix}4 & 4 & 4 & 4 & 4 & 4 & 3\end{pmatrix}\begin{pmatrix}1\\1\\1\\1\\1\\1\\1\end{pmatrix} = \begin{pmatrix}27\end{pmatrix}$$

u-exam: contando las inscripciones totales

Matemáticamente, lo anterior se obtiene por lo siguiente:

$$E_{sum} = \sum_{i=1}^{N_e} \sum_{j=1}^{N_c} E_{ij} = \sum_{i=1}^{N_c} 1 \sum_{j=1}^{N_c} E_{ij} 1$$

u-exam: contando los créditos por estudiante

¿Cómo podríamos la cantidad de créditos que tomó cada estudiante?

Para contar los créditos por estudiante, necesitamos ponderar los elementos de cada fila. Eso es equivalente a una multiplicación por un vector columna por la derecha.

$$E = \begin{pmatrix} 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 3 \\ 3 \\ 3 \\ 3 \\ 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 10 \\ 6 \\ 15 \\ 15 \\ 13 \end{pmatrix}$$

¿Qué representa $E^T \cdot E$?

- ¿Cuáles son sus dimensiones?

Como E tiene N_e filas y N_c columnas, $E^T \cdot E$ debe tener N_c filas y N_c columnas.
De alguna manera, esta matriz está relacionando cursos con cursos.
Pero ¿cómo?

¿Qué representa $E^T \cdot E$?

- ¿Cómo podemos interpretar $E^T \cdot E$?

Definamos la matriz $A = E^T \cdot E$
El elemento $A_{i,j}$ está definido como

$$A_{i,j} = \sum_{k=1}^{N_e} (E^T)_{i,k} (E)_{k,j}$$

¿Alguna idea?

¿Qué representa $E^T \cdot E$?

$$A_{i,j} = \sum_{k=1}^{N_e} (E^T)_{i,k}(E)_{k,j} = \sum_{k=1}^{N_e} E_{k,i}E_{k,j}$$

Recordemos que $E_{k,j}$ toma únicamente los valores 0 y 1, y por lo tanto, $E_{k,i}E_{k,j}$ sólo es igual a 1 cuando ambos términos son iguales a 1, es decir, si el estudiante k está en ambos cursos.
La suma sobre todos los elementos de k representa entonces que para cursos i y j fijos, estamos tomando todos los estudiantes y contando (sumando) sólo aquellos que están simultáneamente en i y j.

¡La suma anterior dará el total de estudiantes que existen en común entre el curso i y el curso j!
Observación: Lo anterior funciona porque E toma los valores 0 y 1.
Si E_{kj} fueran las horas que invierte el estudiante k en el curso j, entonces la interpretación no sería correcta.

¿Qué representa $E^T \cdot E$?

$$\begin{pmatrix} 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 \end{pmatrix}^T \begin{pmatrix} 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 \end{pmatrix} =$$

$$= \begin{pmatrix} 4 & 3 & 3 & 3 & 4 & 3 & 3 \\ 3 & 4 & 4 & 4 & 3 & 3 & 2 \\ 3 & 4 & 4 & 4 & 3 & 3 & 2 \\ 3 & 4 & 4 & 4 & 3 & 3 & 2 \\ 4 & 3 & 3 & 3 & 4 & 3 & 3 \\ 3 & 3 & 3 & 3 & 3 & 4 & 3 \\ 3 & 2 & 2 & 2 & 3 & 3 & 3 \end{pmatrix}$$

Por supuesto, la matriz es simétrica. Esta matiz se llama "matriz de clashing" o "matriz de conflicto" y permite conocer que asignaturas no pueden programarse a la misma hora (y cuales sí pueden programarse de manera simultánea).

Recordemos que la matriz era la siguiente:

$$E = \begin{pmatrix} 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 \end{pmatrix}$$

¿Qué representa $E \cdot E^T$?

- ¿Dimensiones?
- ¿Interpretación?

De manera completamente análoga, $E \cdot E^T$ posee dimensiones N_e filas y N_e columnas, y su elemento i, j representa la cantidad de cursos que el estudiante i tiene en común con el estudiante j.

¿Cómo hacer esto en el computador?

In [2]:

```
import numpy as np
E = np.matrix([ [0 , 1 , 1 , 1 , 0 , 1 , 0 ],
                [1 , 0 , 0 , 0 , 1 , 1 , 1 ],
                [1 , 1 , 1 , 1 , 1 , 1 , 1 ],
                [1 , 1 , 1 , 1 , 1 , 1 , 1 ],
                [1 , 1 , 1 , 1 , 1 , 0 , 0 ],
              ])
print(E)
```

```
[[0 1 1 1 0 1 0]
 [1 0 0 0 1 1 1]
 [1 1 1 1 1 1 1]
 [1 1 1 1 1 1 1]
 [1 1 1 1 1 0 0]]
```

¿Cómo hacer esto en el computador?

Multiplicación por vector fila por la izquierda:

In [3]:

```
vf = np.matrix([1,1,1,1,1])
print(vf.shape, E.shape)
print(vf * E)
```

```
(1, 5) (5, 7)
[[4 4 4 4 4 4 3]]
```

¿Como hacer esto en el computador?

Multiplicación por vector columna por la derecha:

In [4]:

```
vc = np.matrix([1,1,1,1,1,1,1])
print(E.shape, vc.shape)
print(E * vc.T)
```

```
(5, 7) (1, 7)
[[4]
 [4]
 [7]
 [7]
 [5]]
```

¿Cómo hacer esto en el computador?

Contar el total de inscripciones:

In [5]:

```
vf = np.matrix([1,1,1,1,1])
vc = np.matrix([1,1,1,1,1,1,1]).T
print(vf.shape, E.shape, vc.shape)
print(vf * E * vc)
```

```
(1, 5) (5, 7) (7, 1)
[[27]]
```

¡Ojo! Obtenemos una matriz de 1x1

¿Cómo hacer esto en el computador?

Cantidad de estudiantes en común entre distintos cursos, $E^T \cdot E$:

In [6]:

```
print(E.T * E)
```

```
[[4 3 3 3 4 3 3]
 [3 4 4 4 3 3 2]
 [3 4 4 4 3 3 2]
 [3 4 4 4 3 3 2]
 [4 3 3 3 4 3 3]
 [3 3 3 3 3 4 3]
 [3 2 2 2 3 3 3]]
```

¿Cómo hacer esto en el computador?

Cantidad de cursos en común entre estudiantes, $E \cdot E^T$:

In [7]:

```
print(E * E.T)
```

```
[[4 1 4 4 3]
 [1 4 4 4 2]
 [4 4 7 7 5]
 [4 4 7 7 5]
 [3 2 5 5 5]]
```

u-exam | datos reales

Consideremos un conjunto de datos real, con las siguientes características:

- 4 campus
- Aproximadamente 11 mil estudiantes.
- Cerca de 900 asignaturas distintas.
- ¿Cuántas inscripciones, aproximadamente?

Considerando 5 inscripciones por estudiante, deberían ser cerca de 55 mil inscripciones.

u-exam | datos real

Carguemos la matriz con datos reales:

In [8]:

```
import numpy as np
E = np.matrix(np.loadtxt("E.csv", delimiter=";"))
print(E.shape)
```

```
(11249, 919)
```

In [9]:

```
print(E[:5,:7]) # Mostrando los primeros 5 estudiantes y los primeros 7 cursos
```

```
[[1. 0. 0. 0. 0. 0. 0.]
 [1. 0. 0. 0. 0. 0. 0.]
 [1. 0. 0. 0. 0. 0. 0.]
 [1. 0. 0. 0. 0. 0. 0.]
 [1. 0. 0. 0. 0. 0. 0.]]
```

u-exam | datos real

En realidad, no necesitamos que sean "números flotantes" (floats) basta con que sean números enteros (ints). Esto nos permitiría incluso ahorrar RAM, pero cuidado, puede traer efectos colaterales si no se maneja con cuidado.

In [10]:

```
E_float = np.matrix(np.loadtxt("E.csv", delimiter=";"))
E_int = np.matrix(np.loadtxt("E.csv", delimiter=";", dtype=np.int8))
```

In [11]:

```
print(E_float.dtype)
print(E_int.dtype)
```

```
float64
int8
```

In [12]:

```
print(E_float.shape)
print(E_int.shape)
```

```
(11249, 919)
(11249, 919)
```

In [13]:

```
print(E_float[:1,:1].nbytes)
print(E_int[:1,:1].nbytes)
print(E_float.nbytes / E_int.nbytes)
```

```
8
1
8.0
```

u-exam | datos real

Volvamos a cargar los datos, considerando elementos de tipo entero:

In [14]:

```
%%time
E = np.matrix(np.loadtxt("E.csv", delimiter=";", dtype=int))
```

```
CPU times: user 8.31 s, sys: 385 ms, to
tal: 8.7 s
Wall time: 7.75 s
```

In [15]:

```
print("Tamaño: ", E.shape) # Estudiantes, cursos
print("Tipo elementos: ", E.dtype) # 64 bits
print("Tamaño de un elemento: ", E[:1,:1].nbytes) # tamaño de cada elemento
print("Tamaño matriz (en Mb): ", E.nbytes/(1024*1024)) # tamaño de matriz en la RAM
print("Algunos elementos:\n", E[:5,:7]) # Algunos elementos
```

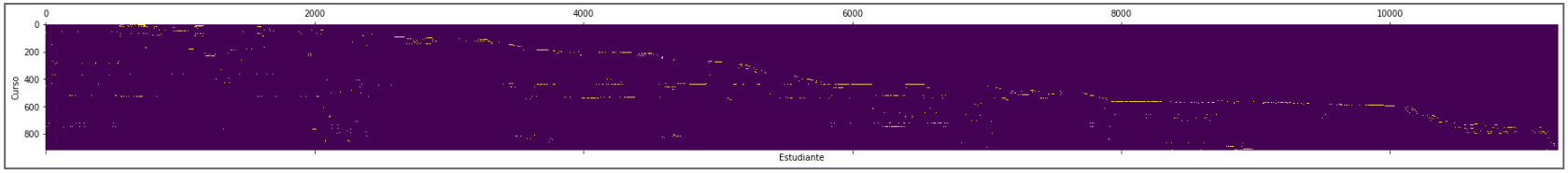
```
Tamaño:  (11249, 919)
Tipo elementos:  int64
Tamaño de un elemento:  8
Tamaño matriz (en Mb):  78.87139129638672
Algunos elementos:
 [[1 0 0 0 0 0 0]
 [1 0 0 0 0 0 0]
 [1 0 0 0 0 0 0]
 [1 0 0 0 0 0 0]
 [1 0 0 0 0 0 0]]
```

u-exam | datos real

Tratemos de mirar la matriz usando algunas herramientas gráficas:

In [16]:

```
from matplotlib import pyplot as plt
fig, ax = plt.subplots(figsize=(32,32))
ax.matshow(E.T, aspect='equal')
plt.box(False)
plt.xlabel("Estudiante")
plt.ylabel("Curso")
plt.show()
```

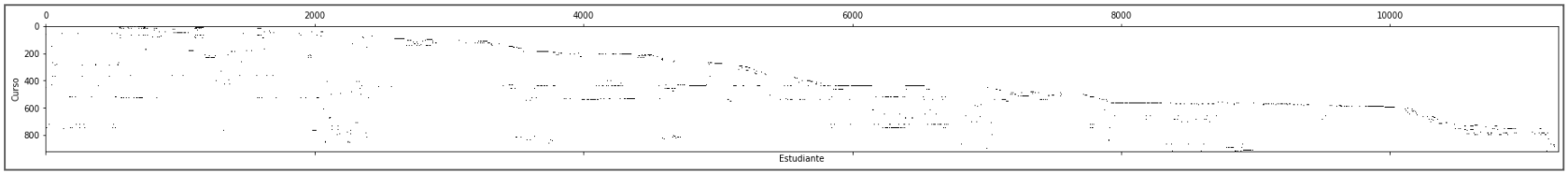


u-exam | datos real

Tratemos de mirar la matriz usando algunas herramientas gráficas:

In [17]:

```
from matplotlib import pyplot as plt
fig, ax = plt.subplots(figsize=(32,32))
ax.matshow(E.T, cmap=plt.cm.Greys, aspect='equal')
plt.xlabel("Estudiante")
plt.ylabel("Curso")
plt.show();
```

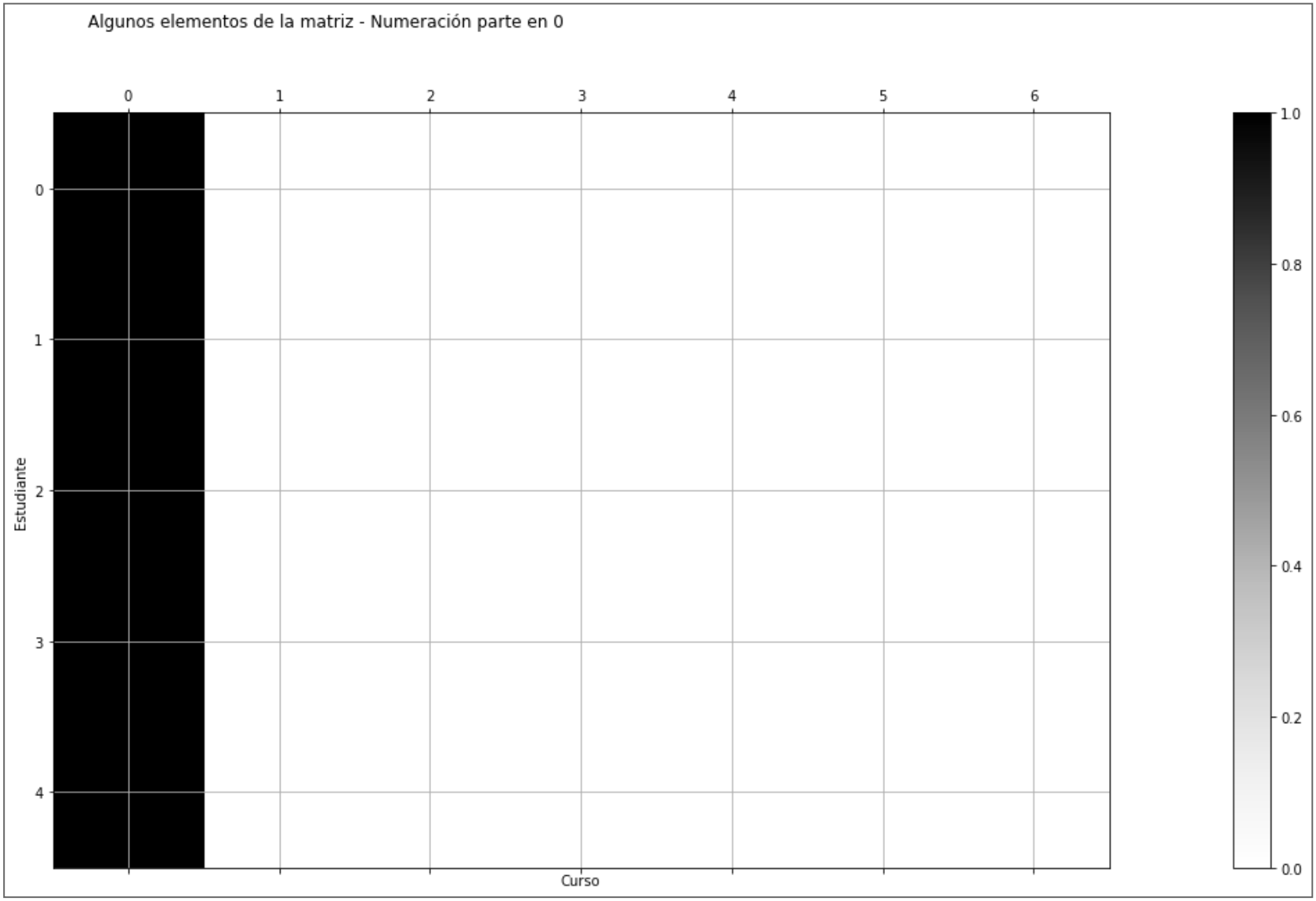


u-exam | datos real

Tratemos de examinar algunos elementos de la matriz usando algunas herramientas gráficas **de manera apropiada**:

In [18]:

```
from matplotlib import pyplot as plt
fig, ax = plt.subplots(figsize=(32,10))
cax = ax.matshow(E[:5,:7], cmap=plt.cm.Greys, aspect='equal')
plt.colorbar(cax)
plt.xlabel("Curso")
plt.ylabel("Estudiante")
plt.grid(True)
plt.suptitle("Algunos elementos de la matriz - Numeración parte en 0")
plt.show()
```


Algunos elementos de la matriz - Numeración parte en 0
0
1
2
3
4
5
6
Estudiante
Curso
1.0
0.8
0.6
0.4
0.2
0.0

u-exam | inscripciones totales

Calculemos la cantidad de inscripciones totales usando varios métodos distintos y midiendo el tiempo que toma realizar la operación.
Recordemos que la matriz es de un tamaño mediano, $11,500 * 900$ aproximadamente.

u-exam | inscripciones totales

Versión ingenua:

In [19]:

```
%%time
# Version Naive
Ne, Nc = E.shape
E_sum = 0
for i in range(Ne):
    for j in range(Nc):
        E_sum = E_sum + E[i,j]
print("Cantidad de inscripciones: ", E_sum)
```

```
Cantidad de inscripciones:  51252
CPU times: user 8.18 s, sys: 51.8 ms, t
otal: 8.23 s
Wall time: 7.8 s
```

El tiempo de ejecución depende del computador usado.

u-exam | inscripciones totales

Versión matricial:

$$E_{sum} = \sum_{i=1}^{n}\sum_{j=1}^{n} E_{ij} = \sum_{i=1}^{n_c} 1 \sum_{i=1}^{n_e} E_{ij} 1$$

In [20]:

```
%%time
vector_uno_cursos = np.matrix(np.ones([Nc,1]))
vector_uno_estudiantes = np.matrix(np.ones([Ne,1]))
E_sum = (vector_uno_estudiantes.T * E) * vector_uno_cursos
print(E_sum)
```

```
[[51252.]]
CPU times: user 85.4 ms, sys: 29.7 ms,
total: 115 ms
Wall time: 55 ms
```

In [21]:

```
%%time
vector_uno_cursos = np.matrix(np.ones([Nc,1], dtype=int))
vector_uno_estudiantes = np.matrix(np.ones([Ne,1], dtype=int))
E_sum = (vector_uno_estudiantes.T * E) * vector_uno_cursos
print(E_sum)
```

```
[[51252]]
CPU times: user 332 ms, sys: 6.94 ms, t
otal: 339 ms
Wall time: 57.6 ms
```

Destacan 2 cosas: existe el fenómeno de "casting" en el cual el numero real (flotante) toma precedencia sobre el entero, y que el resultado sigue siendo una matriz (de 1 fila por 1 columna).

u-exam | inscripciones totales

Versión con funciones nativas:

Usar la función suma ya implementada.

In [22]:

```
%%time
# Versión nativa - matricial
E_sum = E.sum()
print("Cantidad de inscripciones: ", E_sum)
```

```
Cantidad de inscripciones:  51252
CPU times: user 46.2 ms, sys: 2.75 ms,
total: 49 ms
Wall time: 8.14 ms
```

u-exam | inscripciones totales

Resultado:
Puede existir una diferencia de hasta 1000 veces en el tiempo de ejecución de un conteo tan sencillo como la suma de los elementos de una matriz.
Seguir el principio **KISS: Keep It Simple, Stupid!**

u-exam | Estudiantes en común entre asignaturas

Versión ingenua

La versión ingenua es **demasiado** lenta. Consideremos calcularla solo para los primeros 10 cursos.

In [23]:

```
%%time
# Versión Naive : Toma demasiado tiempo!
N_aux = 10
ETE_v2 = np.zeros([N_aux, N_aux])
for i in range(N_aux):
    print("Fila %d de %d. Porcentaje de las filas completado: %.2f" %(i+1, N_aux, 100.*(i+1)/N_aux))
    for j in range(N_aux):
        for k in range(Ne):
            ETE_v2[i,j] = ETE_v1[i,j] + E[k,i]*E[k,j]
```

```
Fila 1 de 10. Porcentaje de las filas c
ompletado: 10.00
```

```
---------------------------------------------------------------------------
NameError
Traceback (most recent call last)
<timed exec> in <module>

NameError: name 'ETE_v1' is not defined
```

In [24]:

```
print(ETE_v2[:5,:5])
```

```
[[0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]]
```

u-exam | Estudiantes en común entre asignaturas

Versión ingenua

¿Cuánto tiempo hubiera sido necesario?

Para una matriz de 10 por $10 = 100$ combinaciones de cursos necesitamos aproximadamente 2 segundos.

Para una matriz de 919 cursos por 919 cursos existen $844,561$ combinaciones y necesitamos 2.2 * 844,561 / 100 segundos, es decir, ¡más de 4 horas!.

In [25]:

```
print(919 * 919)
t = 2.0
print(t * 919 * 919 / 100 / 3600)
```

```
844561
4.692005555555556
```

u-exam | Estudiantes en común entre asignaturas

Versión matricial

In [26]:

```
E.dtype
```

Out[26]:

```
dtype('int64')
```

In [27]:

```
%%time
# Versión matricial (nativa)
ETE_v3 = E.T * E
```

```
CPU times: user 1min 17s, sys: 688 ms,
total: 1min 18s
Wall time: 1min 18s
```

In [28]:

```
print(ETE_v3[:5,:5])
```

```
[[   5    0    0    0    0]
 [   0  545    0    0    0]
 [   0    0  135   40    2]
 [   0    0   40   97    2]
 [   0    0    2    2   50]]
```

Podemos calcular todas el resultado para todos los cursos mediante multiplicación matricial en menos de 90 segundos.

u-exam | Estudiantes en común entre asignaturas

Versión matricial

Veamos el resultado

In [29]:

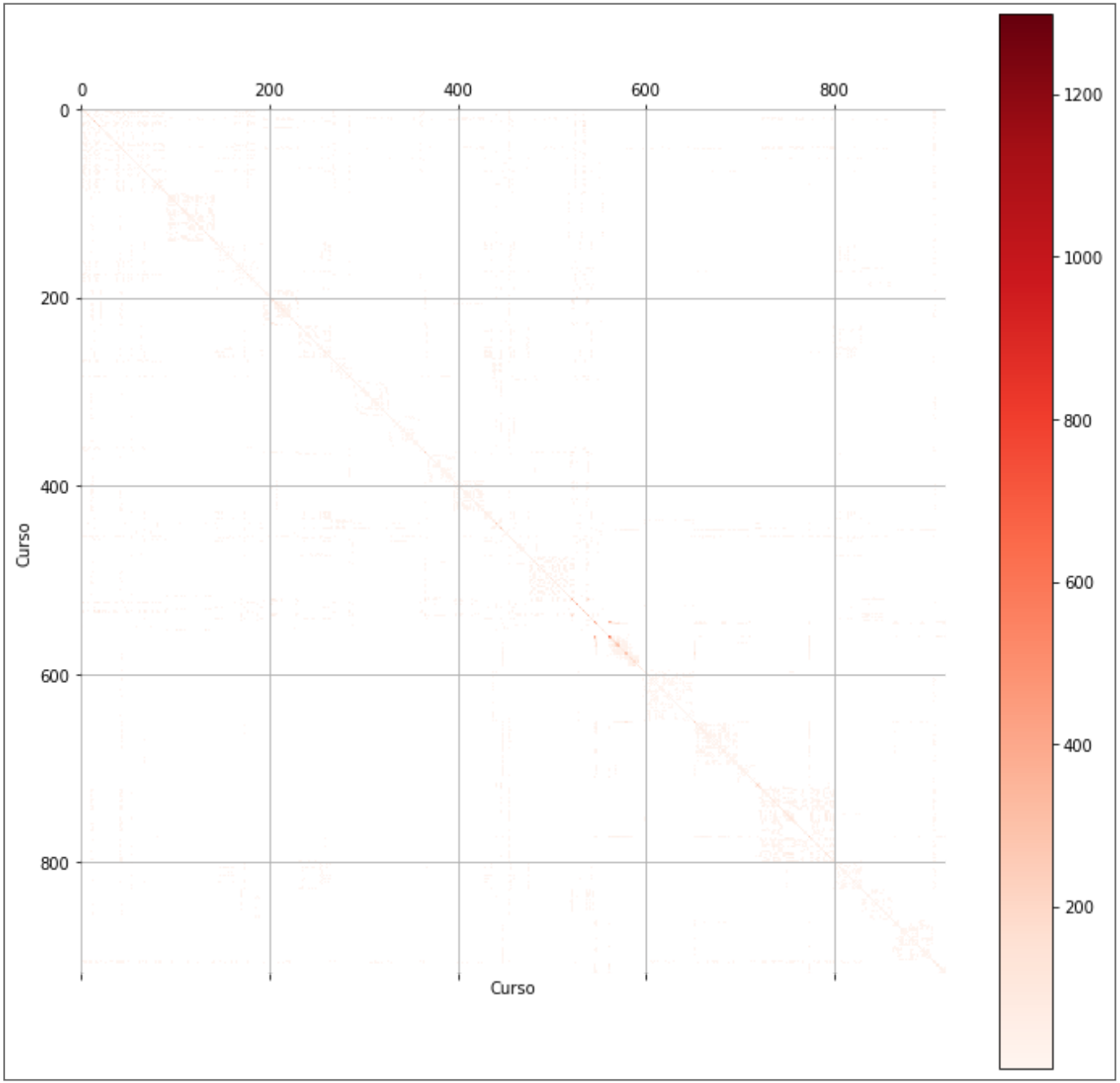
```
from matplotlib import pyplot as plt

def plot_matrix(A):
    fig, ax = plt.subplots(figsize=(12,12))
    A_aux = A.astype(float)
    masked_matrix = np.ma.array(A_aux, mask=(A_aux==0))
    cmap = plt.cm.Reds
    cmap.set_bad(color='white')
    cax = ax.matshow(masked_matrix, cmap=cmap, aspect='equal', interpolation='none')
    plt.colorbar(cax)
    plt.box(False)
    plt.grid(True)
    plt.xlabel("Curso", fontsize=10)
    plt.ylabel("Curso")
    plt.show()
```

In [30]:

```
plot_matrix(ETE_v3[:,:])
```

```
/miniconda3/lib/python3.7/site-packages/ipykernel_launcher.py:8: MatplotlibDeprecationWarning: You are modifying the state of a globally registered colormap. In future versions, you will not be able to modify a registered colormap in-place. To remove this warning, you can make a copy of the colormap first. cmap = copy.copy(mpl.cm.get_cmap("Reds"))
```


0
200
400
600
800
Curso
0
200
400
600
800
Curso
1200
1000
800
600
400
200

u-exam | ¿Porqué hacemos esto?

Para asignar las evaluaciones de manera compatible, es necesario:

- Asignar primero las evaluaciones que son más difíciles, que son aquellas que tienen mayor cantidad de alumnos y tienen la mayor duración ($E^T E$).
- Para los cursos seleccionados, es necesario obtener los estudiantes en común, y buscar separar las fechas de sus evaluaciones para esos estudiantes lo más posible verificando que no coincidan con asignaciones previas (EE^T).

Acá sólo hemos considerado la matriz de estudiantes y cursos, pero también es necesario:

- Considerar las secciones a las que perteneccen los estudiantes, y asignar cada sección en un horario a una sala de capacidad compatible y de tipo de sala apropiado.

- Considerar que el horario de la evaluación debe ser compatible con la disponibilidad de los supervisores que cuidarán la evaluación.

Las otras características también son representadas mediante matrices sobre las cuales las operaciones de álgebra lineal permiten obtener soluciones en un tiempo acotado a pesar de la dificultad del problema.

Resumen y conclusiones

- Algebra lineal y matrices son **muy** importantes.
- Uso de librerías y operaciones nativas resulta crucial para rapidez numérica.
- Interpretación y modelamiento son más interesantes y más complejos que operatoria a ciegas.