Trabajo Práctico N°3

Junghanss, Juan Cruz

Métodos Cuantitativos para Ciencias Sociales y Negocios - Prof. Sergio Pernice

15 de abril, 2021

[2]:
```
import numpy as np
from numpy import linalg as LA  # Notar que importamos linalg con nickname LA

np.random.seed(1)
```

[6]:
```
C = np.array([[1,1],[1,0]])
print(C)
```

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

[7]:
```
w, v = LA.eig(C)
print(w)
print(v)
```

```
[ 1.61803399 -0.61803399]
[[ 0.85065081 -0.52573111]
 [ 0.52573111  0.85065081]]
```

[8]:
```
AutoV1_C = v[:, [0]]
AutoV2_C = v[:, [1]]
print(AutoV1_C)
print(AutoV2_C)
```

```
[[0.85065081]
 [0.52573111]]
[[-0.52573111]
 [ 0.85065081]]
```

[9]:
```
# Generamos una matriz A de nxn (n=2 originalmente) al azar y su traspuesta
n = 4
A = 2*(np.random.rand(n,n)-0.5)   #
AT=A.T
print(A)
```

```
print(AT)
```

```
[[-0.16595599  0.44064899 -0.99977125 -0.39533485]
 [-0.70648822 -0.81532281 -0.62747958 -0.30887855]
 [-0.20646505  0.07763347 -0.16161097  0.370439  ]
 [-0.5910955   0.75623487 -0.94522481  0.34093502]]
[[-0.16595599 -0.70648822 -0.20646505 -0.5910955 ]
 [ 0.44064899 -0.81532281  0.07763347  0.75623487]
 [-0.99977125 -0.62747958 -0.16161097 -0.94522481]
 [-0.39533485 -0.30887855  0.370439    0.34093502]]
```

[10]:
```
# Calculamos sus autovectores y autovalores (notar que en general son complejos)
wA, vA = LA.eig(A)
print(wA)
print(vA)
wAT, vAT = LA.eig(AT)
print(wAT)
print(vAT)
```

```
[ 0.58261698+0.j         -0.54240167+0.77148479j -0.54240167-0.77148479j
 -0.2997684 +0.j        ]
[[ 0.77292345+0.j         -0.08254527-0.52961791j -0.08254527+0.52961791j
   0.50677926+0.j        ]
 [-0.09316922+0.j          0.63445966+0.j          0.63445966-0.j
  -0.54727729+0.j        ]
 [-0.44462828+0.j         -0.06665158+0.06047123j -0.06665158-0.06047123j
  -0.38758567+0.j        ]
 [-0.4429611 +0.j         -0.23639613-0.49615496j -0.23639613+0.49615496j
   0.54170074+0.j        ]]
[ 0.58261698+0.j         -0.54240167+0.77148479j -0.54240167-0.77148479j
 -0.2997684 +0.j        ]
[[ 0.67673284+0.j          0.41468483-0.16547562j  0.41468483+0.16547562j
  -0.35941848+0.j        ]
 [-0.1621988 +0.j         -0.17407857-0.363376j   -0.17407857+0.363376j
  -0.03786207+0.j        ]
 [ 0.12567022+0.j          0.7701102 +0.j          0.7701102 -0.j
  -0.89090866+0.j        ]
 [-0.70705813+0.j         -0.0128101 -0.21230877j -0.0128101 +0.21230877j
   0.2750756 +0.j        ]]
```

[11]:
```
# Tomamos la parte simetrica de A: A = (A + AT)/2 + (A - AT)/2 = A_sim +
↪A_antisim
A_sim = (A + AT)/2
A_sim
```

[11]:
```
array([[-0.16595599, -0.13291962, -0.60311815, -0.49321518],
       [-0.13291962, -0.81532281, -0.27492305,  0.22367816],
```

```
 [-0.60311815, -0.27492305, -0.16161097, -0.28739291],
 [-0.49321518,  0.22367816, -0.28739291,  0.34093502]])
```

[12]:
```python
# calculamos los autovalores, que son reales y los autovectores (ortogonales)
wA_sim, vA_sim = LA.eig(A_sim)
print(wA_sim)
print(vA_sim)
```

```
[-1.12609321 -0.77046372  0.68810148  0.4065007 ]
[[-0.54179776 -0.41623142  0.53743231 -0.49434108]
 [-0.58637337  0.77240905 -0.16133151 -0.183092  ]
 [-0.56666064 -0.2552596  -0.04981521  0.78182908]
 [-0.20375764 -0.4061751  -0.82623066 -0.3329377 ]]
```

1 Trabajo practico

1) Comprobar que los autovectores son ortogonales

2) Construir, para n = 2, la forma cuadratica $f(x) = x^T A_{sim} x$, donde $x = (x_1, x_2)$, es un vector de $\mathbb{R}^2$. Graficarla en 3-d en Python y Geogebra.

3) Construir ejes asociados a los autovectores. Comprobar visualmente que son ortogonales. Borrar (en geogebra y Python) los ejes originales y quedarse solo con los ejes correspondiente a los autovectores.
Comparar con la superficie del archivo Geogebra "FormasCuadraticas", donde explcitamente la forma cuadratica no tiene terminos cruzados.
Girar el grafico de modo que convenserse visualmente que en esos ejes la superficie no tiene terminos cruzados.
Que rol juegan los autovalores de A_sim en la forma de la superficie?
Nuevamente comparar con el archivo Geogebra "FormasCuadraticas"

4) Analiticamente mostrar que si parto de una forma cuadratica en n variables:

$$f(x_1, ..., x_n) = \sum_{i=1}^{n} a_i x_i^2 + \sum_{j<i,j=2}^{n} a_{ij} x_i x_j = \mathbf{x}^\top A \mathbf{x}, \quad \text{donde} \quad \mathbf{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$$

y $A \in \mathbb{R}^{n \times n}$ es una matriz simetrica con elementos diagonales dados por a_i y no diagonales dados por $a_{ij}/2$.

5) Supongamos que al vector genérico $\mathbf{x}$ lo expreso como una combinacion lineal de los n autovectores ortonormalizados de A, esto es,

$$\mathbf{x} = \sum_{i=1}^{n} b_i \hat{\mathbf{a}}_i, \quad \text{con} \quad A\hat{\mathbf{a}}_i = \lambda_i \hat{\mathbf{a}}_i, \quad \hat{\mathbf{a}}_i \cdot \hat{\mathbf{a}}_j = \delta_{ij}$$

donde b_i es la coordenada $i-$esima del vector x en la base $\{\hat{\mathbf{a}}_i\}$ y λ_i es el $i-$esimo autovalor de A. Mostrar que

$$\mathbf{x}^\top A\mathbf{x} = \sum_{i=1}^{n} \lambda_i b_i^2$$

Es decir que la forma cuadrática f, expresada en las coordenadas correspondientes a la base de autovectres de A, no tiene términos cruzados!

6) En base a lo probado en el punto anterior, explicar de que depende que el origen $\mathbf{x} = \mathbf{0}$ sea un maximo, un minimo, o una silla de montar.

2 Resolución

1) Comprobar que los autovectores son ortogonales.

Para comprobar que los vectores son ortogonales, podemos realizar el producto escalar entre estos y verificar que el resultado sea 0.

[27]:
```
print("Check for orthogonality between eigenvectors:\n")
print("Dot product between eigenvector 1 and 2: \n",np.dot(vA_sim[:, [0]].
↪T,vA_sim[:,[1]]))
print("Dot product between eigenvector 1 and 3: \n",np.dot(vA_sim[:, [0]].
↪T,vA_sim[:,[2]]))
print("Dot product between eigenvector 1 and 4: \n",np.dot(vA_sim[:, [0]].
↪T,vA_sim[:,[3]]))
print("Dot product between eigenvector 2 and 3: \n",np.dot(vA_sim[:, [1]].
↪T,vA_sim[:,[2]]))
print("Dot product between eigenvector 2 and 4: \n",np.dot(vA_sim[:, [1]].
↪T,vA_sim[:,[3]]))
```

```
Check for orthogonality between eigenvectors:

Dot product between eigenvector 1 and 2:
 [[1.66533454e-16]]
Dot product between eigenvector 1 and 3:
 [[1.66533454e-16]]
Dot product between eigenvector 1 and 4:
 [[-1.66533454e-16]]
Dot product between eigenvector 2 and 3:
 [[3.88578059e-16]]
Dot product between eigenvector 2 and 4:
 [[-2.77555756e-17]]
```

Podemos observar que la relación de cada par de vectores (1,2), (1,3), (1,4), (2,3) y (2,4) es ortogonal entre sí.

2) Construir, para n = 2, la forma cuadratica $f(x) = x^T A_{sim} x$, donde $x = (x_1, x_2)$, es un vector de $\mathbb{R}^2$. Graficarla en 3-d en Python y Geogebra.

La forma cuadratica o polinomio homogeneo cuadratico podemos calcularlo como:

$f(x1, x2) = xT \cdot A \cdot x$ donde $x = (x1, x2)$; A_simetrica = np.array([[2,5],[5,8]]).

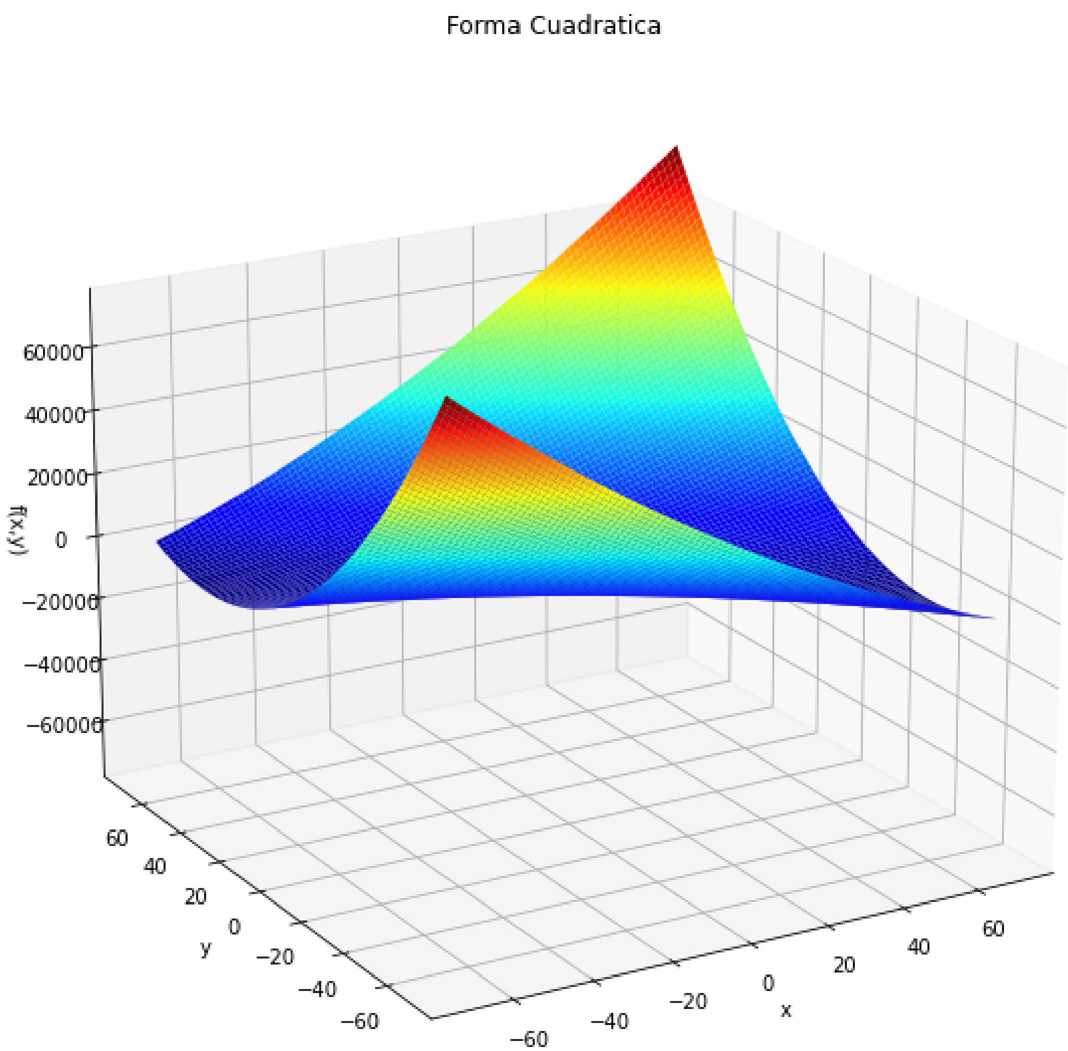
Nuestro cálculo, técnicamente, sería entonces algo como quadratic_form = np.dot(np.dot(x.T,A),x)

El resultado es $2x^2 + 10xy + 8y^2$, que será nuestra función a continuación.

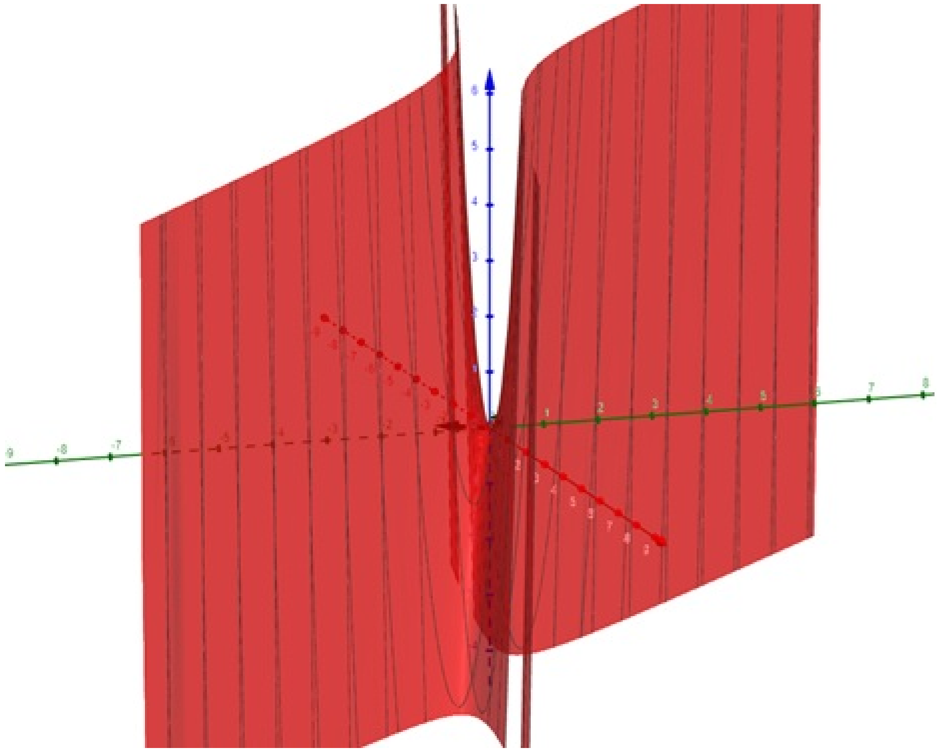
Analíticamente, en Python sería:

[3]:
```
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

# Boundaries for each of the two variables in our domain
x_interval = (-70, 70)
y_interval = (-70, 70)

# Sample points within each of these intervals:
x_points = np.linspace(x_interval[0], x_interval[1], 100)
y_points = np.linspace(y_interval[0], y_interval[1], 100)

# Cartesian product of these two sampled sets to produce two arrays
# that (when stacked) form a set of ordered pairs we can compute a function on:
X, Y = np.meshgrid(x_points, y_points)

# Define our Quadratic Form as a Python function of two scalar inputs

def forma_cuadratica(x, y):
    """
    La forma cuadratica o polinomio homogeneo cuadratico podemos calcularlo como:
 ↪ f(x1,x2) = xT·A·x
    donde x=(x1,x2) ; A_simetrica = np.array([[2,5],[5,8]])
    Nuestro cálculo, técnicamente, sería entonces algo como quadratic_form = np.
 ↪dot(np.dot(x.T,A),x)
    El resultado es 2·x^2 + 10·x·y + 8·y^2, que será nuestra función a␣
 ↪continuación:
    """
    return 2*x**2 + 10*x*y + 8*y**2

# Produce a vectorized version of the function that can be called on vectors or␣
 ↪matrices of inputs
forma_cuadratica_vectorizada = np.vectorize(forma_cuadratica)

# Compute an output array from our two domain arrays:
Z = forma_cuadratica_vectorizada(X, Y)
```

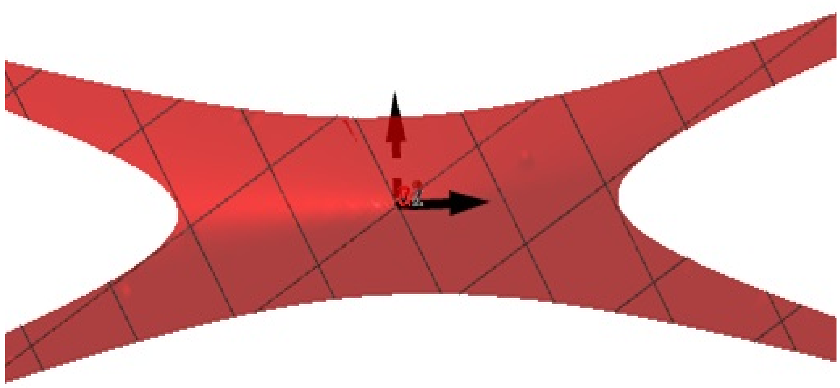
```python
# Plot the Function
plt.figure(figsize=(20, 10))
ax = plt.axes(projection='3d')
ax.view_init(20,-120)
ax.plot_surface(X, Y, Z, rstride=1, cstride=1, cmap='jet', edgecolor=None)
ax.set(xlabel='x', ylabel='y', zlabel='f(x,y)', title='Forma Cuadratica')
ax.set_zlim3d([-75000, 75000])
plt.show()
```

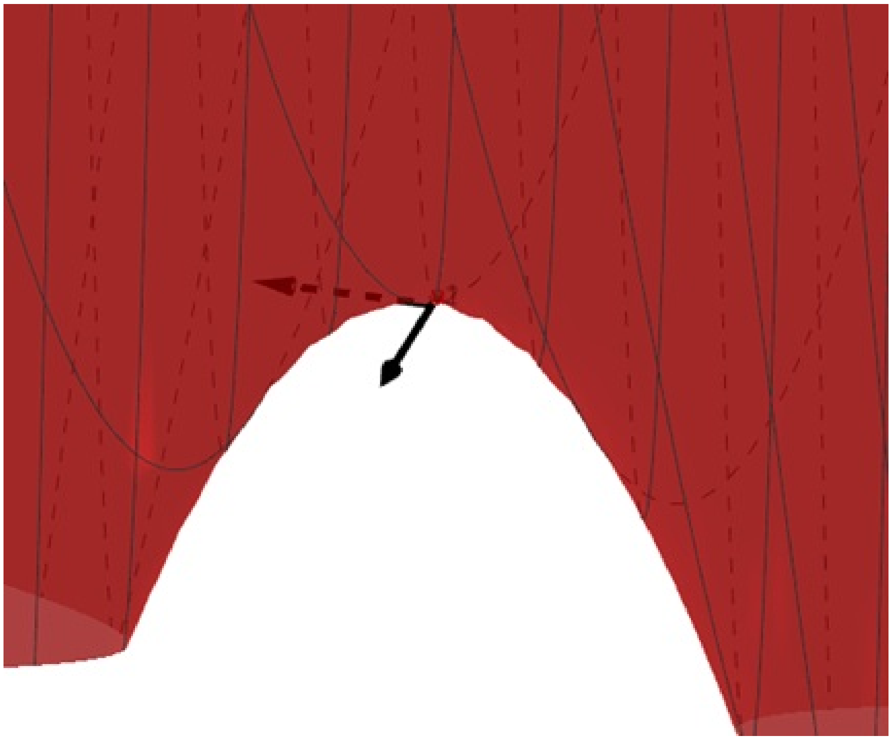


Las imágenes de GeoGebra asociadas a la forma cuadrática $2x^2 + 10xy + 8y^2$ se pueden ver a continuación:

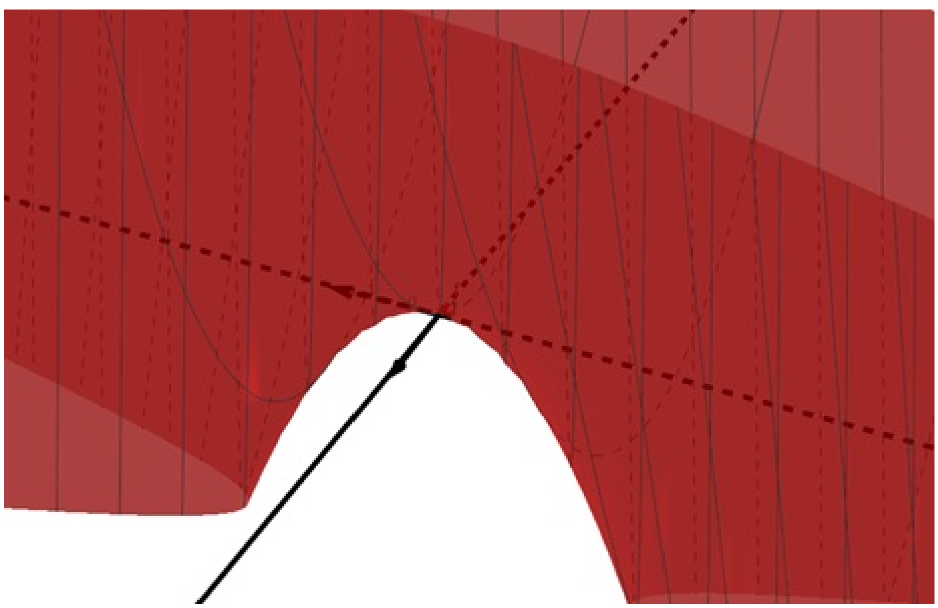


Junto con sus autovectores (ortogonales):





Construyendole “ejes” a los autovectores:



3) Construir ejes asociados a los autovectores. Comprobar visualmente que son ortogonales. Borrar (en geogebra y Python) los ejes originales y quedarse solo con los ejes correspondiente a los autovectores.
   Comparar con la superficie del archivo Geogebra “FormasCuadraticas”, donde explcitamente la forma cuadratica no tiene terminos cruzados.
   Girar el grafico de modo que convenserse visualmente que en esos ejes la superficie no tiene terminos cruzados.
   Que rol juegan los autovalores de A_sim en la forma de la superficie?
   Nuevamente comparar con el archivo Geogebra “FormasCuadraticas”

[5]:
```python
A_simetrica = np.array([[2,5],[5,8]])    # Tomamos la matriz A (simetrica) del␣
 ↪ejercicio anterior

eV, eW = LA.eig(A_simetrica)     # Calculamos sus autovalores y autovectores

print("Eigenvalues: ", eV,"   \n Eigenvectors: \n",eW)
print("\n Verificamos ortogonalidad entre eigenvectors: ", np.dot(eW[:,[0]].T,␣
 ↪eW[:,[1]]))
```

```
Eigenvalues:  [-0.83095189 10.83095189]
 Eigenvectors:
 [[-0.87019991 -0.49269881]
 [ 0.49269881 -0.87019991]]

 Verificamos ortogonalidad entre eigenvectors:  [[0.]]
```

¿Qué rol cumplen los autovectores de A_simetrica en la forma cuadrática?

Refiriendonos a las figuras del ejercicio anterior, son los “ejes” que nos indican la dirección de expansión/compresión (y/o que de volteo) la forma cuadrática. Estos ejes a lo largo de los cuales actúa esta transformación lineal nos permite entenderla mejor (los eigenvalues le dan los factores por los cuales se produce esta compresión).

Cuantas más direcciones tenga a lo largo de las cuales se entienda el comportamiento de una transformación lineal, más fácil será comprenderla; por lo que se desea tener tantos vectores propios linealmente independientes como sea posible asociados a una sola transformación lineal.

4) Analiticamente mostrar que si parto de una forma cuadratica en n variables:

$$f(x_1, ..., x_n) = \sum_{i=1}^{n} a_i x_i^2 + \sum_{j<i,j=2}^{n} a_{ij} x_i x_j = \mathbf{x}^\top A\mathbf{x}, \quad \text{donde} \quad \mathbf{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$$

y $A \in \mathbb{R}^{n\times n}$ es una matriz simetrica con elementos diagonales dados por a_i y no diagonales dados por $a_{ij}/2$.

Si tenemos una forma cuadrática de n variables, sabemos que su expresión general matricial es: $\mathbf{x}^T \cdot \mathbf{A} \cdot \mathbf{x}$. Procederemos a trabajar esta última expresión antes, para luego validar analíticamente la demostración.

Supongamos dos matrices $\mathbf{C}, \mathbf{D}$:

$$(\mathbf{C} \cdot \mathbf{D})_i = \sum_{l=1}^{n} \mathbf{C}_{jl} \cdot \mathbf{D}_{li}$$

Entonces, en nuestro contexto:

$$\left(\mathbf{x}^T \cdot \mathbf{A}\right)_i = \sum_{m=1}^{n} x_m \cdot a_{ml}$$

O bien, podría ser:

$$\left(\mathbf{x}^T \cdot \mathbf{A} \cdot \mathbf{x}\right) = \sum_{m=1}^{n} \sum_{l=1}^{n} x_m \cdot a_{ml} \cdot x_l$$

Desarrollado esto, ahora partamos la sumatoria anterior en donde $m = l$ y $m \neq l$, y sea también $a_{ii} = a_i$

$$\sum_{m=1}^{n} \sum_{l=1}^{n} x_m a_{ml} x_l = \sum_{m=1}^{n} a_m x_m^2 + \sum_{m=l}^{n} a_{ml} x_m x_l$$

De esta manera, comprobamos de forma general que:

$$f(x_1, ..., x_n) = \sum_{m=1}^{n} a_m x_m^2 + \sum_{m=l}^{n} a_{ml} x_m x_l = \sum_{m=1}^{n} \sum_{l=1}^{n} x_m a_{ml} x_l = \left(\mathbf{x}^T \cdot \mathbf{A} \cdot \mathbf{x}\right)$$

5) Supongamos que al vector genérico $\mathbf{x}$ lo expreso como una combinacion lineal de los n autovectores ortonormalizados de A, esto es,

$$\mathbf{x} = \sum_{i=1}^{n} b_i \hat{\mathbf{a}}_i, \quad \text{con} \quad A\hat{\mathbf{a}}_i = \lambda_i \hat{\mathbf{a}}_i, \quad \hat{\mathbf{a}}_i \cdot \hat{\mathbf{a}}_j = \delta_{ij}$$

donde b_i es la coordenada $i-$esima del vector x en la base $\{\hat{\mathbf{a}}_i\}$ y λ_i es el $i-$esimo autovalor de A. Mostrar que

$$\mathbf{x}^\top A\mathbf{x} = \sum_{i=1}^{n} \lambda_i b_i^2$$

Es decir que la forma cuadrática f, expresada en las coordenadas correspondientes a la base de autovectres de A, no tiene términos cruzados.

Para demostrar que llegamos a $\mathbf{x}^\top A\mathbf{x} = \sum_{i=1}^{n} \lambda_i b_i^2$ partamos de la siguiente expresión:

$$\mathbf{x}^\top A\mathbf{x} = \left(\sum_{i=1}^{n} b_i a_i\right)^T \cdot A \cdot \left(\sum_{i=1}^{n} b_i a_i\right)$$

Por propiedad de las sumatorias, podemos operar:

$$\mathbf{x}^\top A\mathbf{x} = \left(\sum_{i=1}^{n} b_i a_i\right)^T \cdot \left(\sum_{i=1}^{n} b_i \cdot A \cdot a_i\right)$$

$$\mathbf{x}^\top A\mathbf{x} = \sum_{i=1}^{n} b_i^T a_i^T \cdot b_i A a_i$$

Sabemos que $A \cdot \hat{a}_i = \lambda_i \cdot \hat{a}_i$ y que $b_i^T = b_i$:

$$\mathbf{x}^\top A\mathbf{x} = \sum_{i=1}^{n} b_i \cdot a_i^T \cdot b_i \cdot \lambda_i a_i$$

Operando algebraicamente, entre otras cosas vemos que $a_i^T \cdot a_i = 1$:

$$\mathbf{x}^\top A\mathbf{x} = \sum_{i=1}^{n} b_i^2 \cdot \lambda_i$$

De esta forma, hemos probado que $\mathbf{x}^\top A\mathbf{x} = \sum_{i=1}^{n} b_i^2 \cdot \lambda_i$, lo que nos indica que la forma cuadrática asociada a la matriz A, no posee términos cruzados (o sea, términos lineales).

6) En base a lo probado en el punto anterior, explicar de que depende que el origen $\mathbf{x} = \mathbf{0}$ sea un máximo, un mínimo, o una silla de montar.

Considerando una forma cuadrática sin términos lineales (cruzados) $\sum_{i=1}^{n} \lambda_i b_i^2$, podemos determinar que el punto crítico $(0, 0)$ es decir, el origen, será un máximo, mínimo o punto de silla según los siguientes parámetros:

- Un máximo será un punto crítico donde f cerca de este sea siempre más pequeña.
- Un mínimo será un punto crítico donde f cerca de este sea siempre más grande.
- Un punto de silla será un punto crítico donde f cerca de este sea más grande en algunas direcciones y más pequeña en otras.

Si bien lo anterior es una explicación sumamente intuitiva, podemos directamente ligarlo con los signos de nuestra forma cuadrática: sea q una forma cuadrática de n variables.

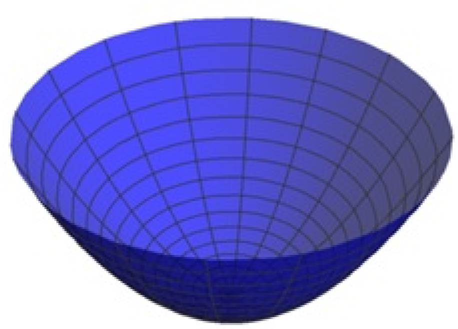
El punto crítico en el origen será un mínimo local de q si y solo si q es positiva definida.

El origen será un máximo local si y solo si q es negativa definida.

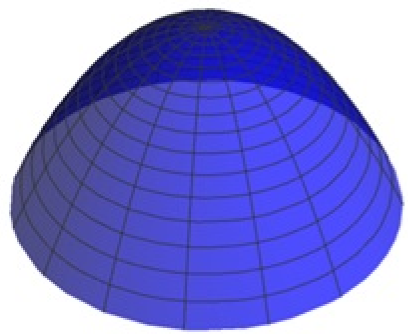
Para otros casos, por ejemplo f(x,y)=x·y, deducimos que tendremos un punto de silla.

Gráficamente:

La función $f(x, y) = x^2 + y^2$ posee un mínimo local en el origen $(0, 0)$



La función $f(x,y) = -x^2 - y^2$ posee un máximo local en el origen $(0,0)$



La función $f(x,y) = x^2 - y^2$ posee un punto de silla en el origen $(0,0)$