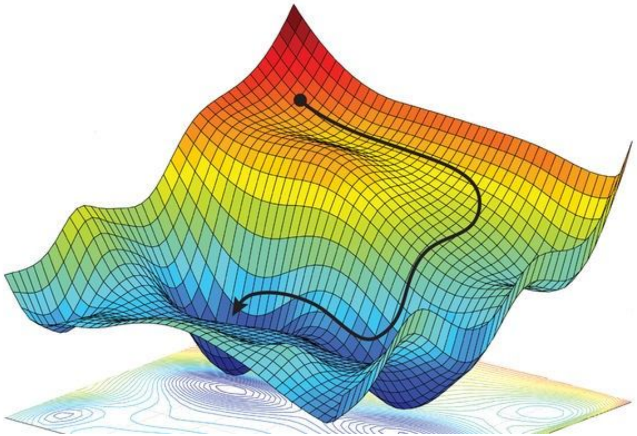
Programación no lineal
Clase 23

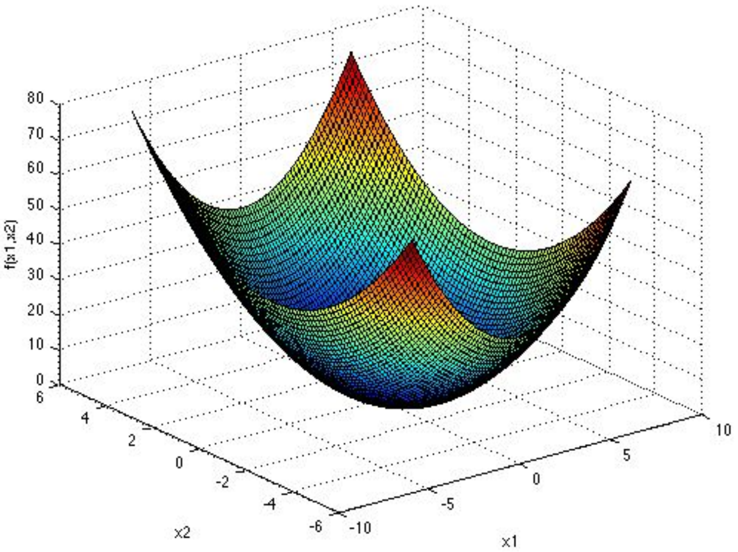
Investigación Operativa UTN FRBA
Curso: I4051
Docente: Martín Palazzo

UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Programación no lineal

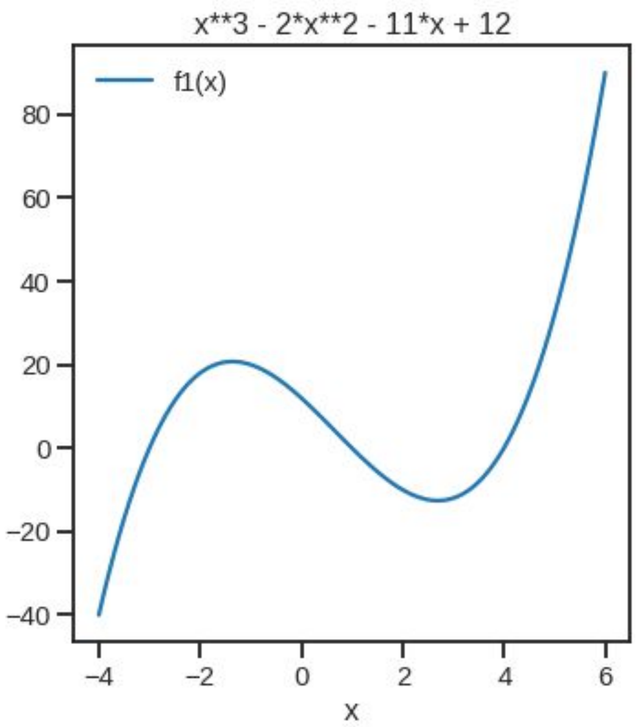


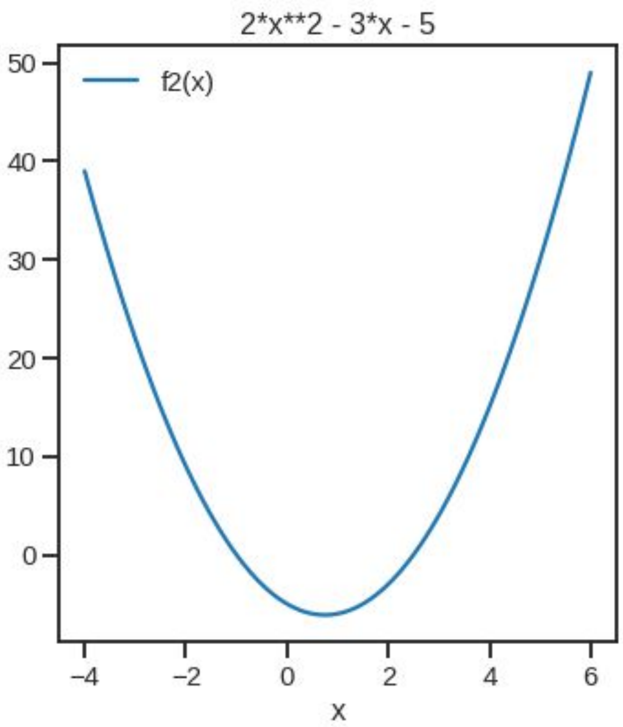
Función no lineal no convexa



Función no lineal convexa (cuadratica)

Funciones no lineales no acotadas



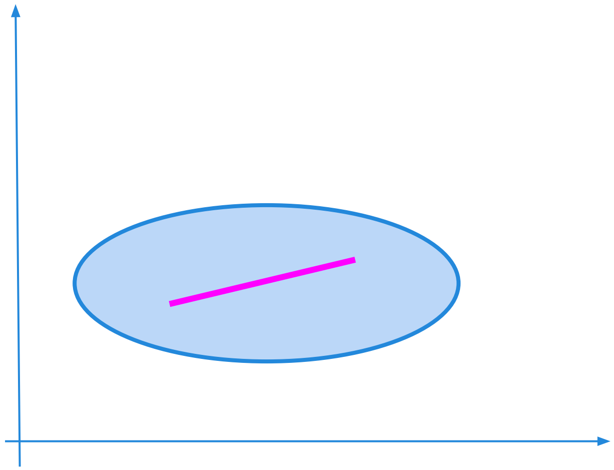


$$f(x) = x^3 - 2x^2 - 11x + 12$$

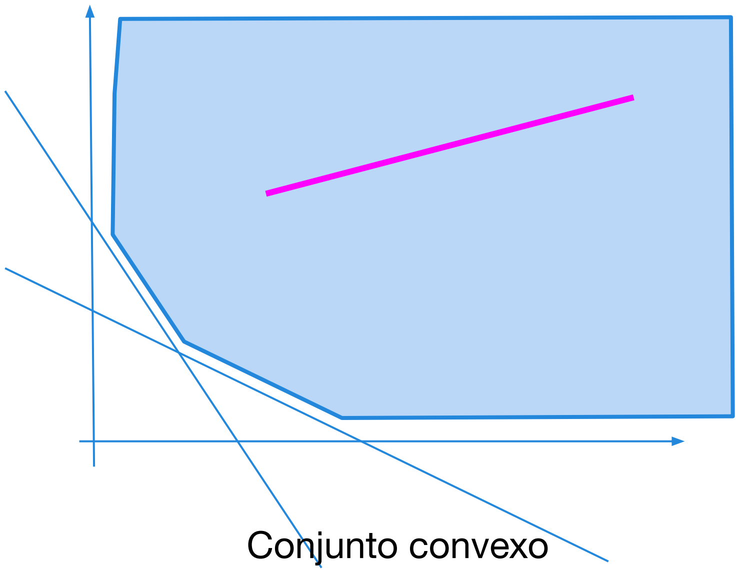
$$f(x) = 2x^2 - 3x - 5$$

Conjunto Convexo

Un conjunto es convexo si la línea que une cualquier par de puntos en el conjunto también pertenece al conjunto.



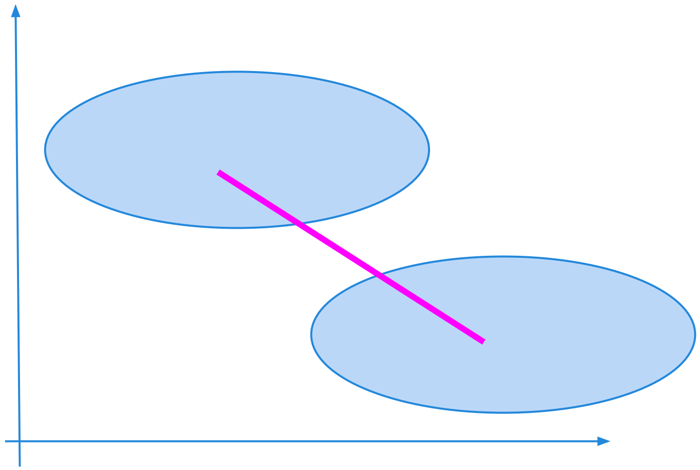
Conjunto convexo



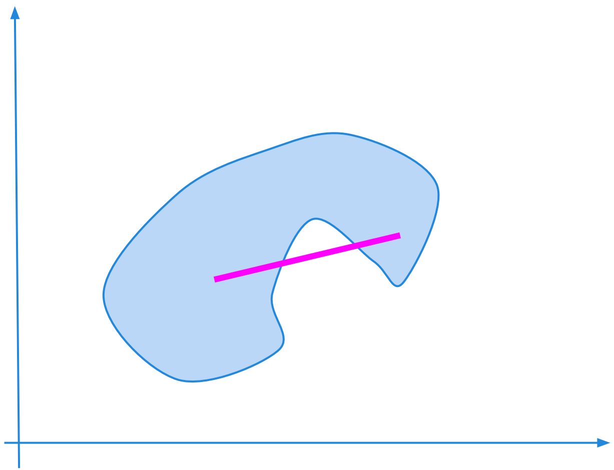
Conjunto convexo

Conjunto no convexo

Un conjunto es convexo si la línea que une cualquier par de puntos en el conjunto también pertenece al conjunto.

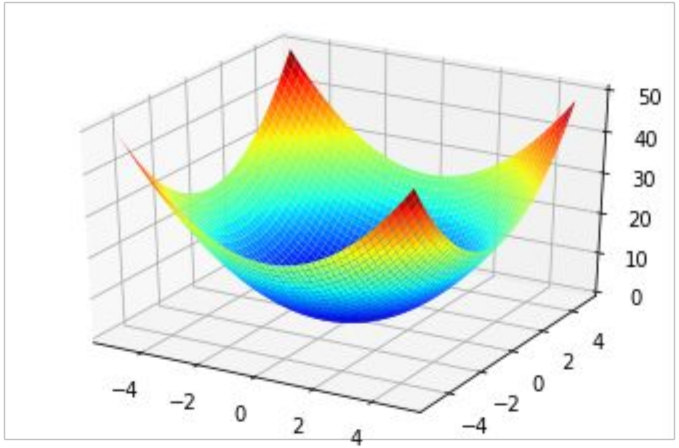


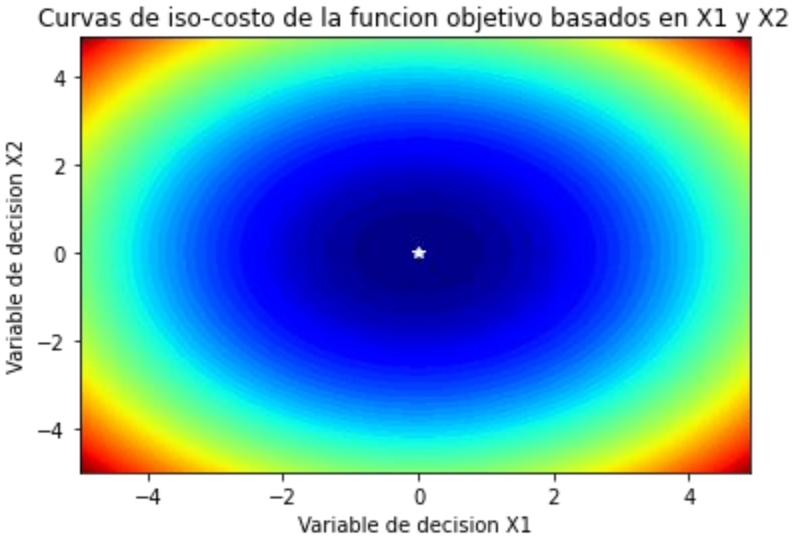
Conjunto no convexo



Conjunto no convexo

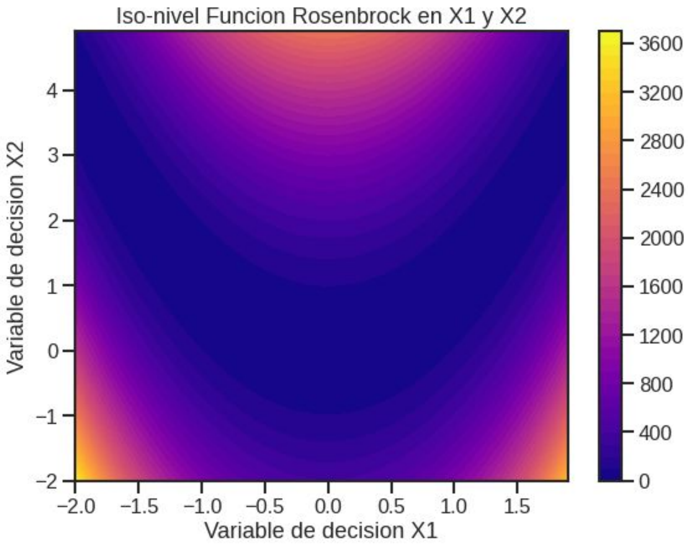
Función objetivo cuadratica no lineal: 2 variables de decision

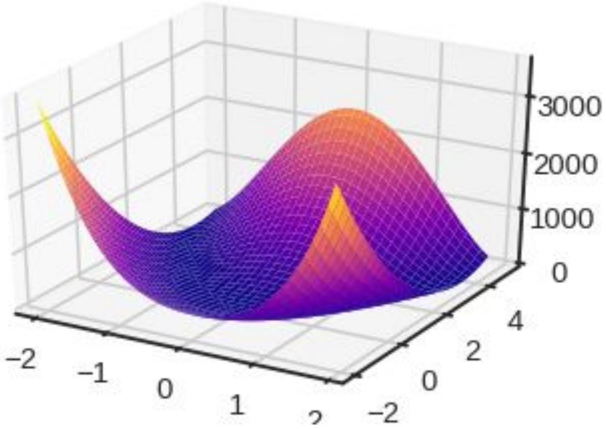




$$f(x_1, x_2) = x_1^2 + x_2^2$$

Función no lineal no acotada: Funcion de Rosenbrock

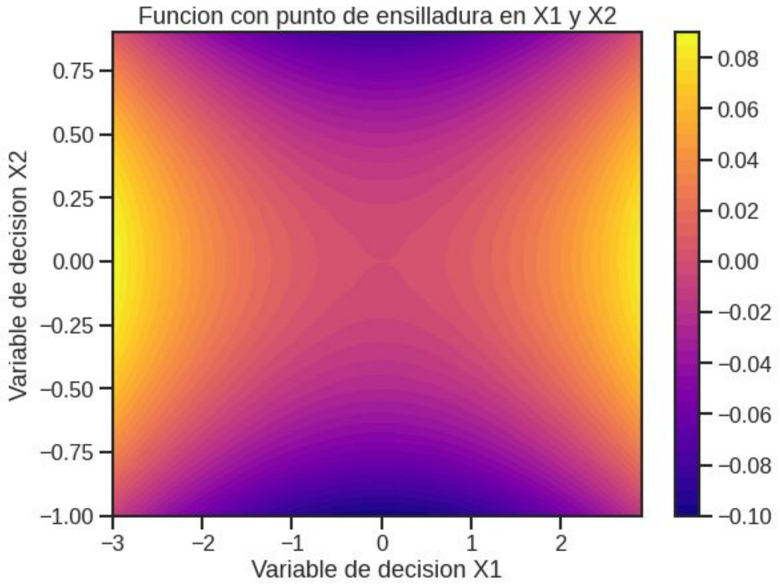


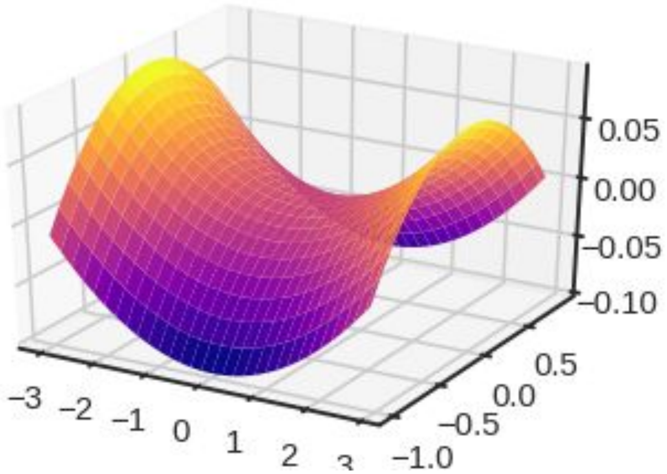


$$f(x) = (a - x_1)^2 + b(x_2 - x_1^2)$$

$$a = 1 \qquad\qquad b = 100$$

Función no lineal no acotada: Función con punto de ensilladura





$$f(x) = x_1^2 - 10x_2^2$$

Modelos de Inventario con optimización convexa

Caso de estudio

Centro de Distribucion

Que nivel de inventario maneja un centro de distribucion asi? Qué preguntas tienen que hacerse para tratar de responder la primer pregunta? Existen políticas de inventario?

Inventarios en distintas industrias

Terminal Automotriz

Consumo masivo

Auto-partista

Retail o E-commerce

Que política de inventario se determina para cada industria? Es igual en todos los casos? De que depende? Qué preguntas tienen que hacerse para poder contestar la primer pregunta?

Sistemas de inventarios

Arribo de insumos **Almacenamiento de insumos** **Despacho de insumos**

Un sistema de inventarios está caracterizado por un flujo de insumos que arriban al lugar de depósito, se almacenan por un tiempo determinado (usualmente finito) y luego son despachados.

Si consideramos solo un insumo\item nuestro problema tiene una sola variable a considerar. Si tenemos múltiples productos\insumos\items entonces nuestro problema tiene alta dimensión.

Sistemas de inventario

Un sistema industrial, de cualquier naturaleza, suele tener un inventario de bienes y recursos que le permita desarrollar su funcionamiento continuo.

En algunos casos un nivel muy bajo del nivel de inventario puede ocasionar interrupciones de alto costo. En otros casos un exceso del nivel de inventario puede generar altos costos en capital inmovilizado.

Por estas razones una de las variables operativas a determinar en un sistema industrial es el nivel de inventarios para operar.

Hay dos preguntas importantes a considerar respecto a la reposición de insumos:

- Cuanto pedir?
- Cuándo pedir?

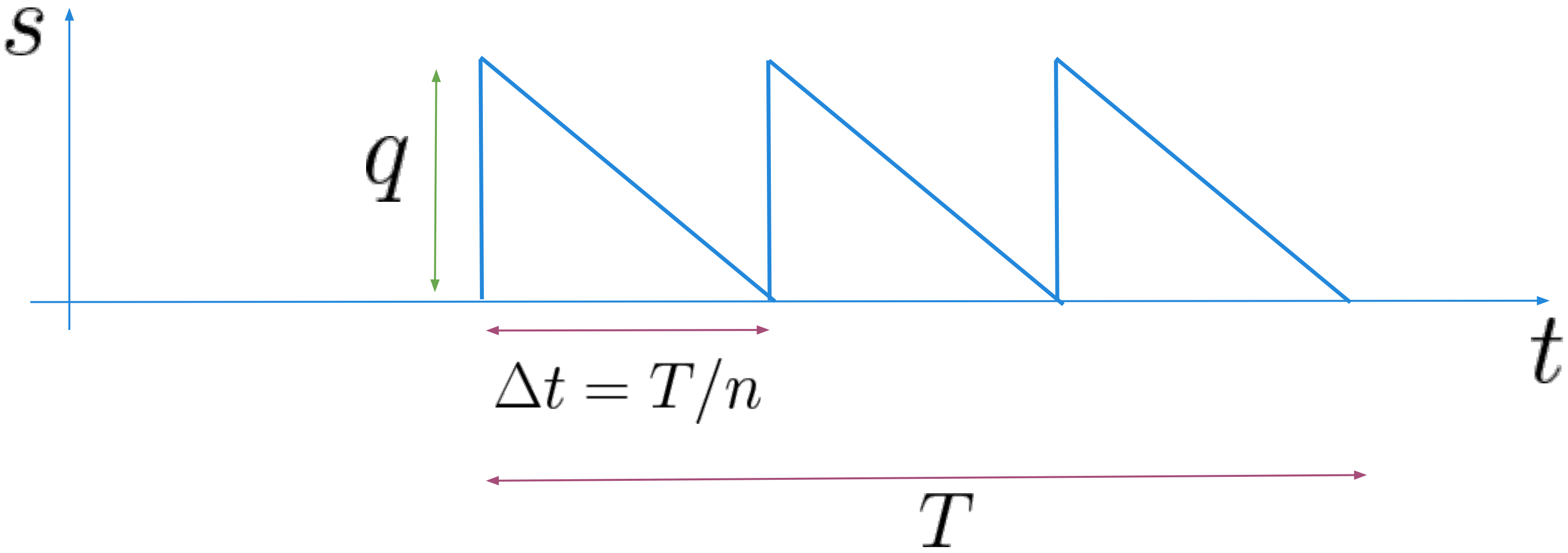
Los extremos mencionados dan la pauta de que existe un costo de inventarios, por esta razón vamos a querer determinar la política de inventarios que **minimice el costo**.

Sistemas de inventarios

$$Q^{(t=i)} = \begin{bmatrix} q_1 \\ q_2 \\ \dots \\ q_n \end{bmatrix} \qquad S^{(t=i)} = \begin{bmatrix} s_1 \\ s_2 \\ \dots \\ s_n \end{bmatrix} \qquad D^{(T)} = \begin{bmatrix} d_1 \\ d_2 \\ \dots \\ d_n \end{bmatrix}$$

Podemos modelizar el problema pensando que la cantidad a pedir regularmente en el tiempo de cada item es “q”, la cantidad de inventario de cada item en un tiempo t es “s” y la demanda de cada item en una ventana de tiempo es “d”. Si tenemos “n” items entonces nuestro problema es n-dimensional.

Modelo de inventario fundamental



En este modelo fundamental suponemos que hay un solo item n=1, la reposición de insumos es instantánea y que no existe stock de seguridad. El inventario se consume a una tasa D de unidades/tiempo (demanda).

Modelizar un problema de inventarios

$$f(q) = z \rightarrow \min$$

Simplificando el problema, si suponemos que la demanda en un periodo dado T es un dato conocido y constante entonces podemos modelizar al **costo** del inventario en función de la cantidad "q" de unidades a solicitar en cada pedido a realizar en el periodo T. Por esta razón, vamos a utilizar herramientas de optimización para encontrar el valor de "q" que minimiza la función objetivo costo "f".

Costo del inventario

Costo Inventarios = Costo Compra/reposición + Costo Adquisición + Costo de almacenamiento + Costo faltante

El costo de inventario es una función con 3 componentes. El costo de compra/reposición se basa en el precio unitario de cada producto y la cantidad de producto a solicitar en cada orden. El costo de adquisición representa el costo fijo que requiere **cada** orden de compra (departamento de compras + recepción + control de calidad, etc). El costo de almacenamiento representa el costo de mantener una existencia de inventario disponible, incluye el interés sobre el capital inmovilizado, el costo de manipulación en el depósito y el mantenimiento del depósito.

Costo del inventario

$$\text{Costo Adq.} \quad = b_i.D$$

$$\text{Costo pedido} \quad = k.\frac{D_i}{q_i}$$

$$\text{Costo alm.} \quad = \frac{1}{2}.q_i.i_i.b_i$$

$$\frac{D_i}{q_i} = n$$

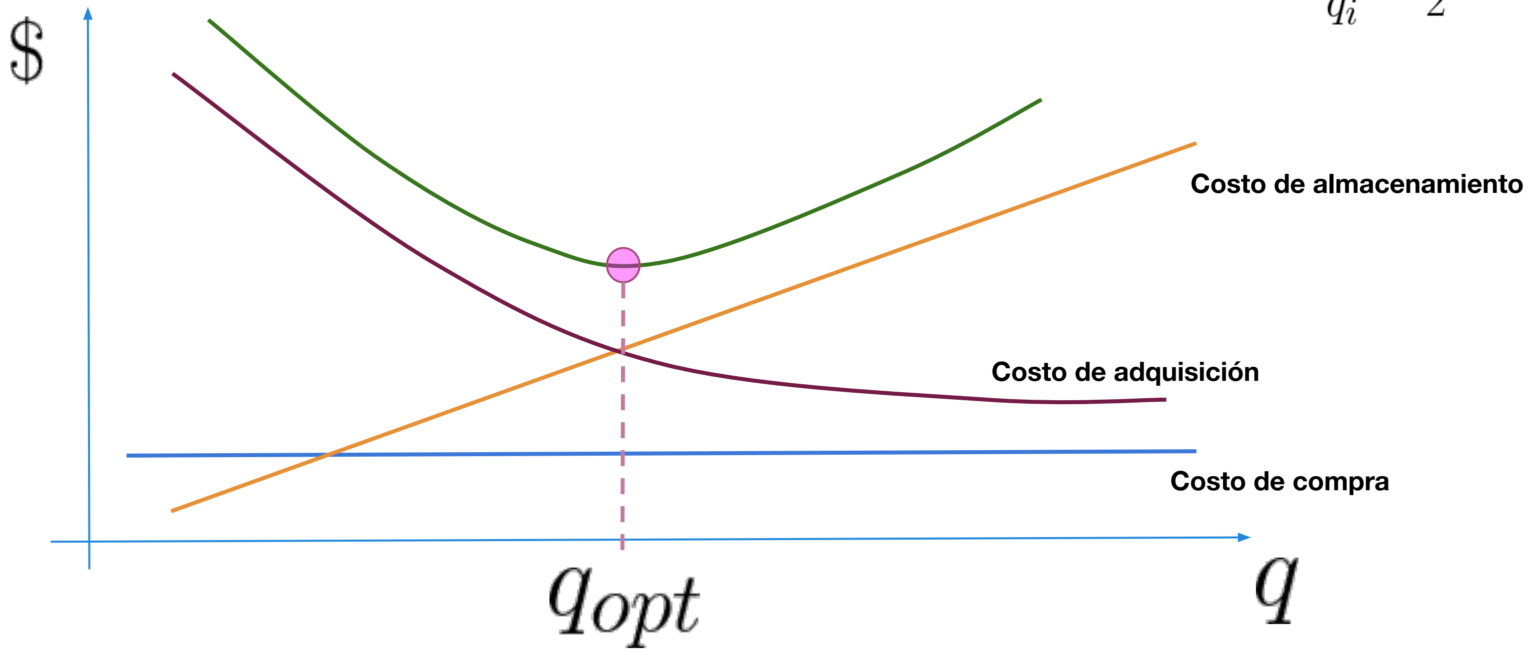
n :	cant. veces a pedir en T
D_i :	Demanda del producto i en T
b_i :	Precio de adquisicion por unidad de i
i_i :	Tasa Anual de almacenamiento
k :	Costo de orden de compra
q_i :	Cantidad a pedir en cada orden
T :	Periodo de analisis

$$z_i = b_i.D_i + k.\frac{D_i}{q_i} + \frac{1}{2}.q_i.i.b_i$$

Función objetivo a minimizar -> costo total de almacenamiento en todo el periodo T

Costo Total Esperado

$$\mathbf{z}_i = P_i.q_i + k.\frac{D_i}{q_i} + \frac{1}{2}.q_i.b.P_i$$



Minimo Costo Total Esperado

Para calcular el **mínimo** Costo Total Esperado (CTE) en el periodo T del sistema de inventarios tendremos las siguientes consideraciones:

- Suponemos que la función de costo es continua, diferenciable y convexa. Podemos entonces suponer que existe un mínimo global y encontrarlo en la coordenada donde la derivada es = 0.

$$\frac{d\mathbf{z}}{dq} = \frac{d(b.D)}{dq} + \frac{d(k.\frac{D}{q})}{dq} + \frac{d(\frac{1}{2}.q.i.b)}{dq} = 0$$

- El costo de compra en el periodo T es una constante ya que no importa la cantidad de pedidos que se hagan ni cuanto se pida en cada uno, la cantidad a comprar en el periodo de análisis siempre será la misma y será la necesaria para satisfacer la demanda. Además su derivada en función de q (cantidad a pedir) es cero.
- Entonces el **mínimo** del CTE dependerá del costo de adquisición y el costo de almacenamiento.

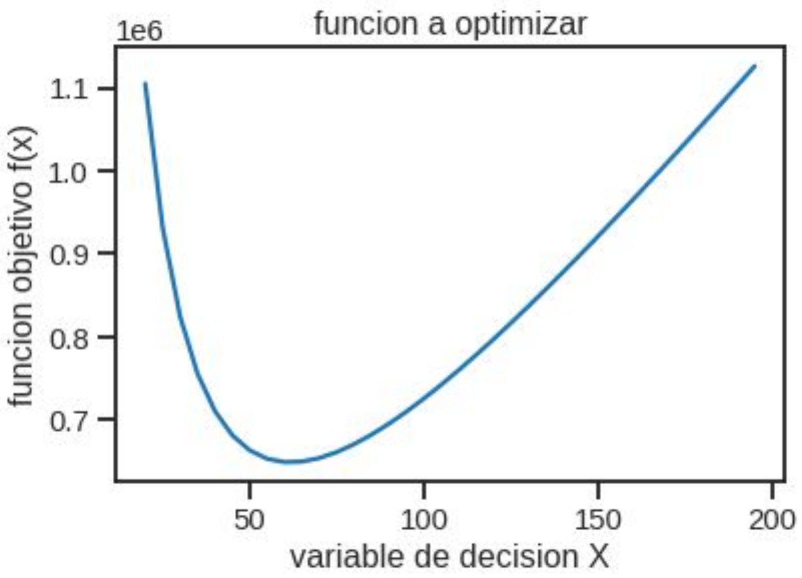
Lote optimo

$$q_{\text{opt}} = \sqrt{\frac{2Dk}{b \cdot i}}$$

Entonces para un solo item con el modelo de inventarios fundamental más básico podemos calcular deterministicamente cual es el tamaño del lote de reposición óptimo que minimiza el costo total esperado.

Función de costo de inventarios de 1 item

```
def costo_stock(p, x1, k, d, b):
  return p*x1 +  k*d*(1/x1) + 0.5*x1*b*p
```



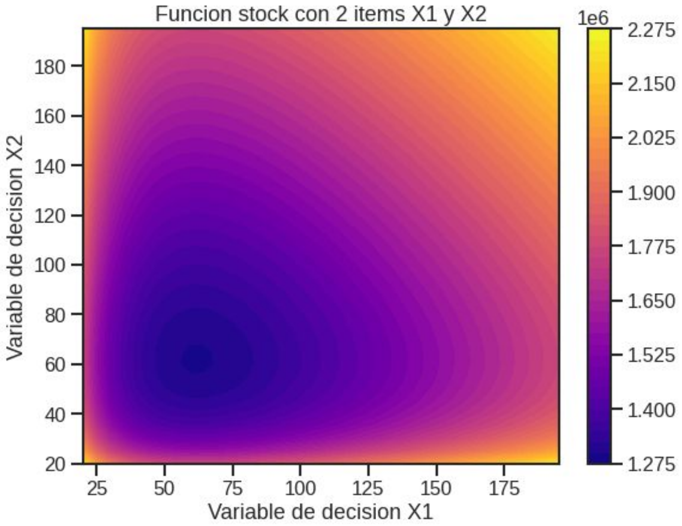
Inventarios: multi-item sin restricción

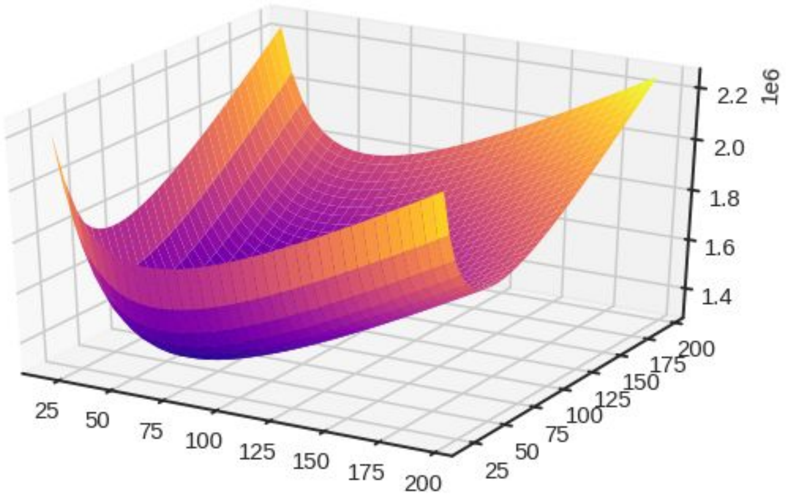
- En el caso de tener "m" items sin restricción de volumen la solución que determine el Costo Total Esperado óptimo estará dada por un vector de "m" posiciones donde cada una determinará la cantidad óptima a solicitar de cada item.
- Para calcular el vector de items que minimice el costo lo que haremos es calcular de manera univariada/independiente el lote optimo de cada item y su respectivo costo.
- Una vez calculados los costos de cada item sumaremos cada término para obtener un costo total.

$$\mathrm{CTE}_o = \sum_{i=1}^{m} b_i D_i + \frac{1}{2} . q_i . c_{1i} . T + k_i \frac{D_i}{q_i}$$

$$Q_o = [q_{o1}, q_{o2}, \ldots, q_{om}]$$

Función de costo de inventarios de 2 item






```
def costo_stock_2d(p1,p2, x1, x2, k1, k2, d1, d2, b1, b2):
  costo_item1 = p*x1 +  k*d*(1/x1) + 0.5*x1*b*p
  costo_item2 = p*x2 +  k*d*(1/x2) + 0.5*x2*b*p
  return
```

Descenso por gradiente

Metodo para resolver: optimización no lineal

Optimización por gradiente descendiente

$$x \in \mathbb{R}^d$$

$$\min_x f(x)$$

$$\nabla f(x_0)$$

Gradiente de F(x)

$$\gamma$$

Paso de optimización

$$x_1 = x_0 - \gamma \nabla f(x_0)$$

$$x_{n+1} = x_n - \gamma \nabla f(x_n)$$

$$x_{n+2} = x_{n+1} - \gamma \nabla f(x_{n+1})$$

$$f(x_n) > f(x_{n+1})$$

Supongamos que queremos optimizar la función no lineal, continua y diferenciable f(x) definida en el dominio de X. Se partirá desde la solución inicial x0 (llamado semilla) actualizando iterativamente la solución del problema en dirección al gradiente descendiente con un paso *gamma* de manera tal que la nueva solución presente un valor de la función objetivo menor.

Solucion Lote Optimo con Gradiente descendiente

$$z_i = b_i.D_i + k.\frac{D_i}{q_i} + \frac{1}{2}.q_i.i.b_i$$

$$\frac{d\mathbf{z}}{dq} = \frac{d(b.D)}{dq} + \frac{d(k.\frac{D}{q})}{dq} + \frac{d(\frac{1}{2}.q.i.b)}{dq}$$

$$\frac{d\mathbf{z}}{dq} = 0.5 \cdot i \cdot b_i - q^{-2}$$

Optimizando con scipy

Usando scipy.optimize()

scipy.optimize()

Función objetivo definida en python

$$f(x)$$

Variables de decisión y conjunto factible

$$x$$

$$g(x) \leq 0$$

$$h(x) = 0$$

scipy.optimize()

Punto X optimo

$$x^*$$

$$\nabla f(x)$$

Gradiente de f(x)

- Determina la pendiente de la funcion

$$\nabla^2 f(x)$$

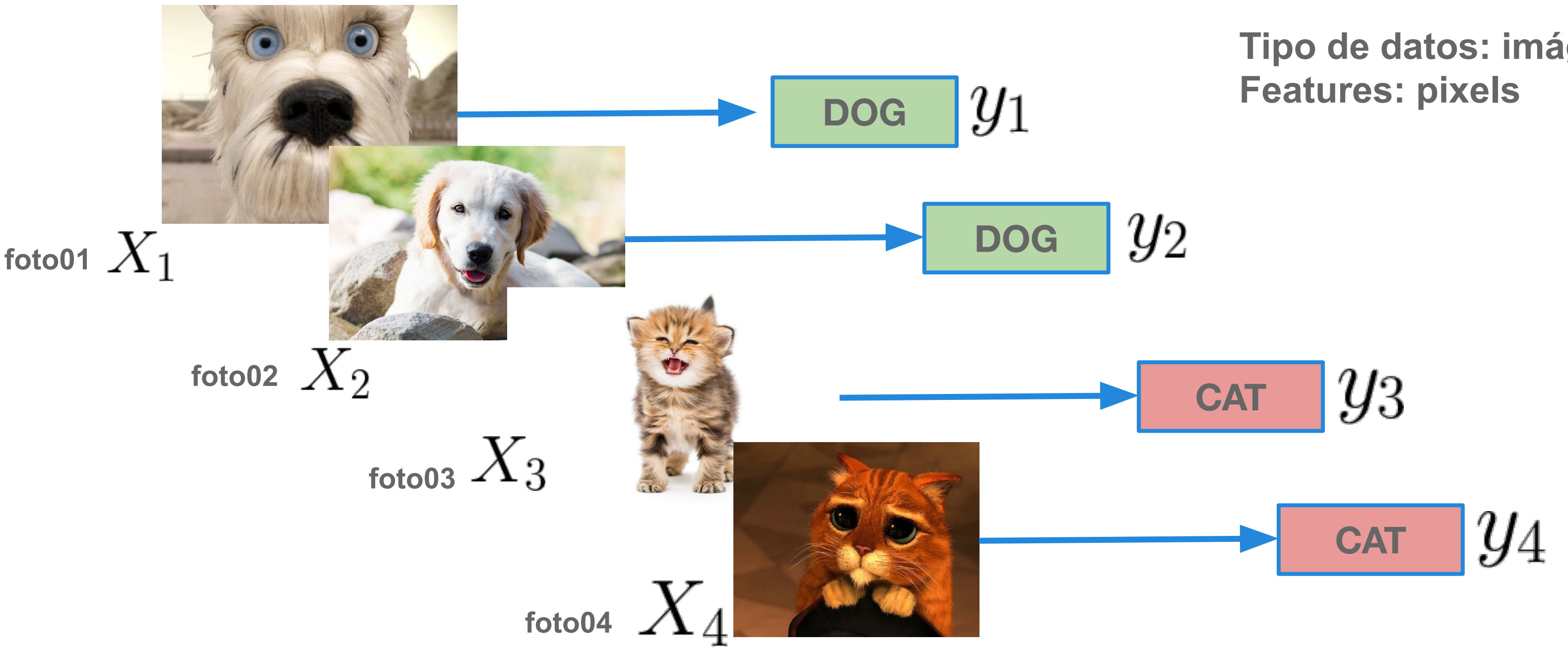
Hessiano de f(x)

- Determina la curvatura de la funcion

Caso de estudio: Optimizacion en Deep Learning

Caso de estudio

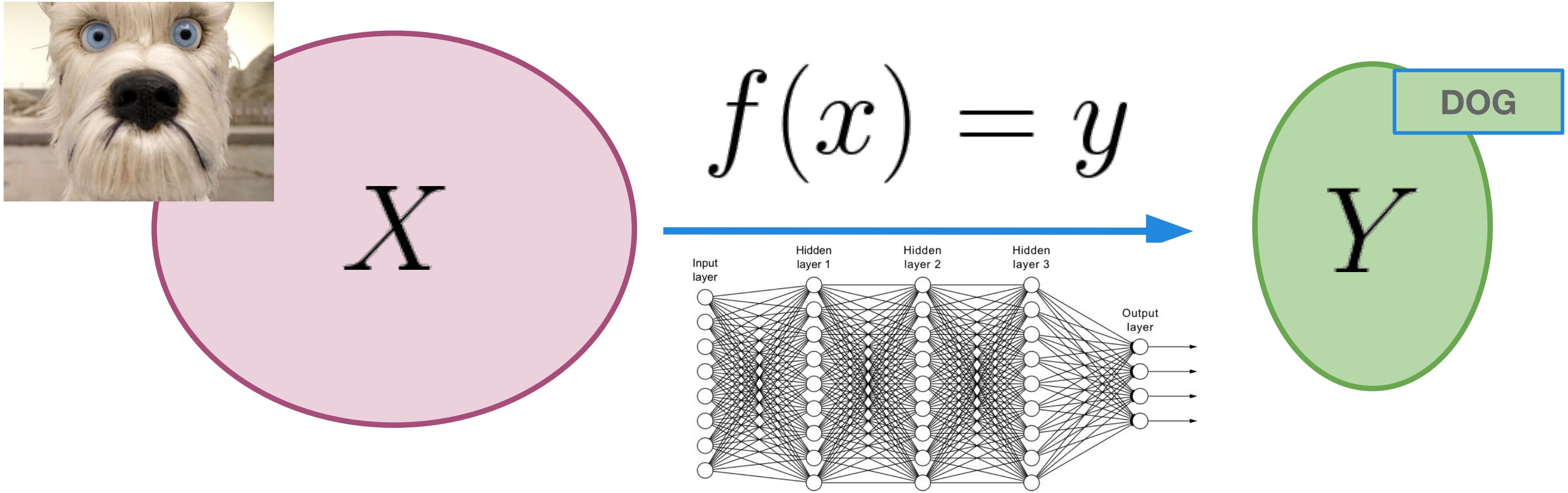
Caso de estudio: Machine Learning & aprendizaje supervisado



Tipo de datos: imágenes
Features: pixels

Cada foto viene acompañada de una etiqueta (label). Queremos aprender una f(x) que explique con el menor error la etiqueta de una foto nueva.

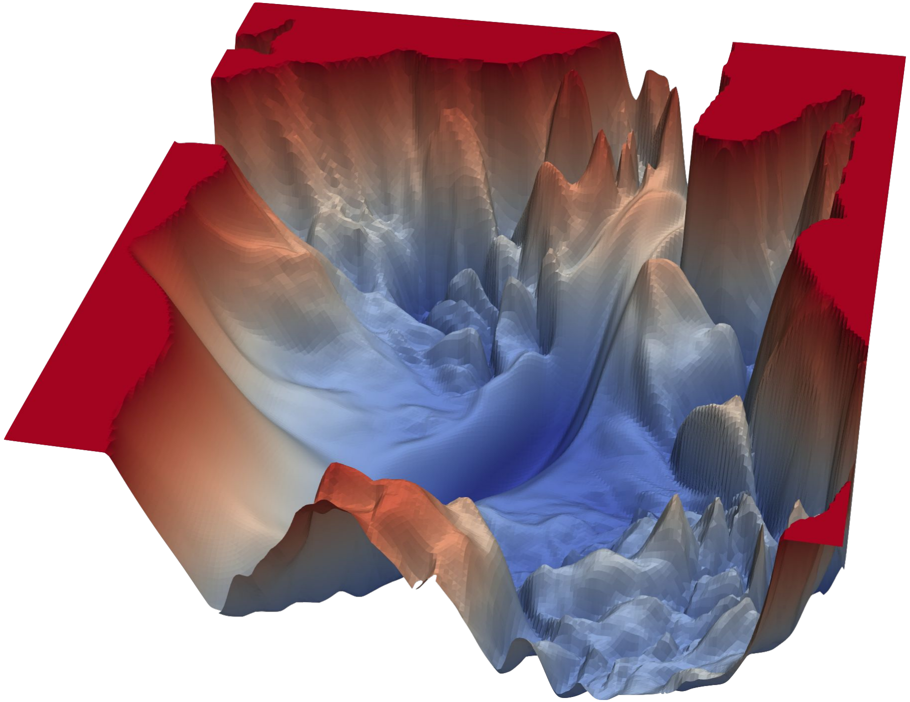
Caso de estudio: Machine Learning & aprendizaje supervisado



Queremos aprender una f(x) que explique lo mejor posible la etiqueta de una foto leyendo únicamente la foto (sin saber cual es la verdadera etiqueta).

Es decir, queremos aprender una función que **minimice** el error de predicción de etiquetas.

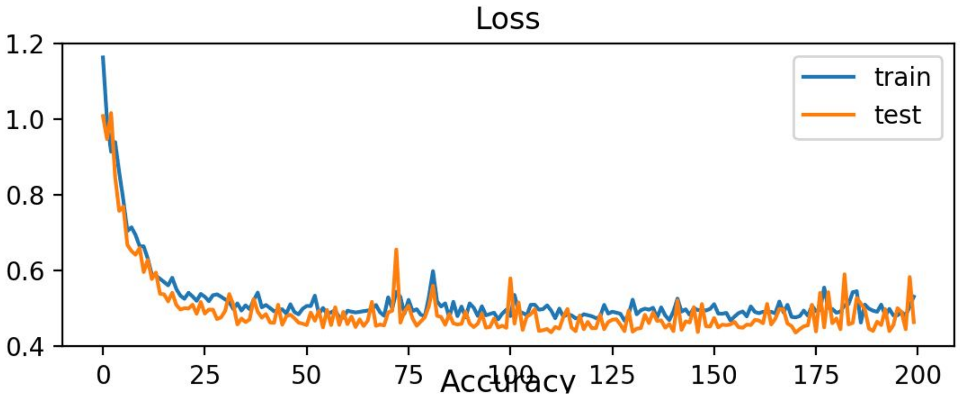
Caso de estudio: Machine Learning & aprendizaje supervisado

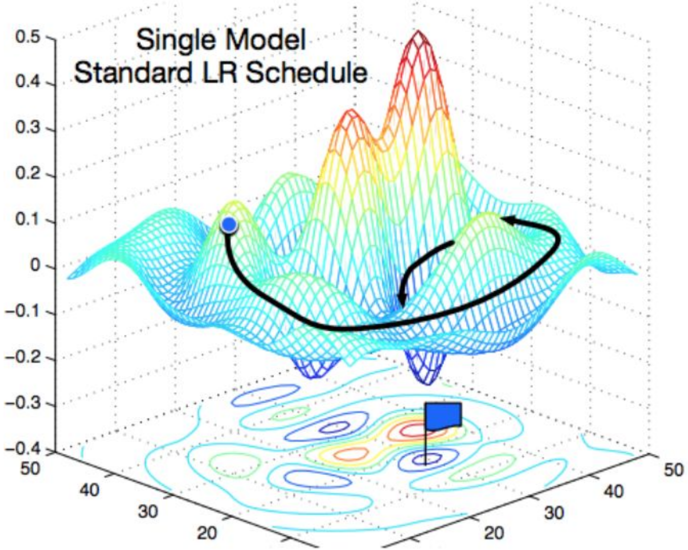


*https://www.cs.umd.edu/~tomg/projects/landscapes/

Cuando usamos redes neuronales la función objetivo es el error de predicción de la misma. Esta funcion presenta muchos minimos locales por eso trataremos de llegar a un mínimo local que sea aceptable usando el gradiente descendiente.

Caso de estudio: Machine Learning & aprendizaje supervisado





La funcion objetivo es el error del modelo. Queremos minimizarlo aunque no es una funcion convexa! Por esa razon usamos gradiente descendiente para ir paso-a-paso moviendonos en la direccion donde el gradiente desciende, es decir donde el error de la funcion objetivo decrece.