Diego Hernandez Rodriguez
@eu_sou_dieguinho
C311

Esta tarea representa un analisis del modelo $f(x,y) = (x^2+1)^{(y^2+1)} + e^y + e^{-y}$ mediante el uso de los algoritmos, aprendidos en las conferencias de Algoritmos para problemas Irrestrictos y con restricciones, de Maximo Descenso y de Newton.

1 Problema a analizar

El problema a analizar es la ecuacion:

$$f(x,y) = (x^2+1)^{(y^2+1)} + e^y + e^{-y}$$

Su dominio son los reales ya que no tiene punto de indefinicion y es continua en todo su dominio por composicion de funciones continuas elementales.
Esta representa la siguiente funcion en 3D:

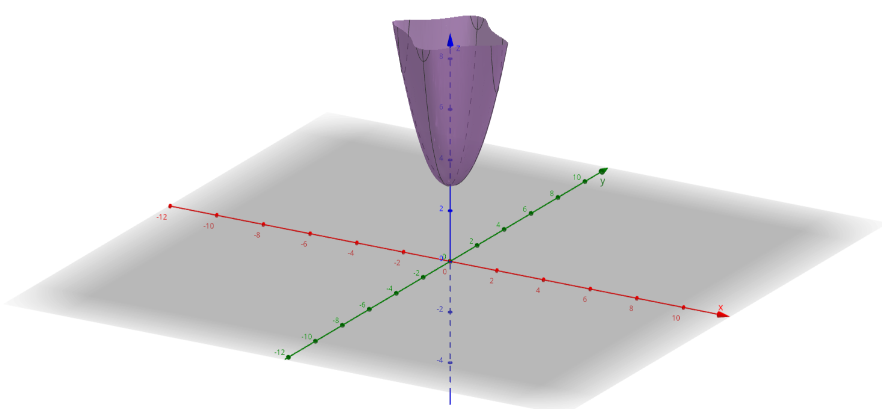


Figura que muestra la continuidad de dicha funcion.

2 Analisis del modelo, soluciones y convexidad

Analizando el problema mas a profundidad, aqui vemos las derivadas parciales de la funcion:

$$\frac{\partial f}{\partial x} = 2x(y^2+1)(x^2+1)^{y^2}$$

$$\frac{\partial f}{\partial y} = 2y\ln(x^2+1)(x^2+1)^{(y^2+1)} + e^y - e^{-y}$$

Sus segundas derivadas parciales:

$$\frac{\partial^2 f}{\partial x^2} = (y^2+1)(2(x^2+1)^{y^2} + (2xy)^2(x^2+1)^{(y^2-1)})$$

$$\frac{\partial^2 f}{\partial y^2} = 2\ln(x^2+1)(x^2+1)^{(y^2+1)}(1+2y^2\ln(x^2+1)) + e^y + e^{-y}$$

$$\frac{\partial^2 f}{\partial x \partial y} = \frac{\partial^2 f}{\partial y \partial x} = 4xy(x^2+1)^{y^2}(1+(y^2+1)\ln(x^2+1))$$

Dicha funcion es continua y estrictamente positiva en todo su dominio.
Ahora para ver la existencia de sus minimos locales vamos a analizar sus derivadas parciales para ver cuando el vector gradiente se hace igual al (0,0).

$\frac{\partial f}{\partial x} = 0$ si y solo si $x = 0$, puesto que $(y^2+1) > 0$ para todo y, $(x^2+1) > 0$ para todo x y $(x^2+1)^{(y^2)} > 0$ para todo x, por tanto $2x$ debe ser igual a 0, por tanto $x = 0$.
Ahora viendo $\frac{\partial f}{\partial y} = 0$ con $x = 0$, tenemos que $\ln(x^2+1) = \ln(1) = 0$, por ende $e^y - e^-y$ tiene que ser igual a 0, osea $y = -y$, lo que implica que y tambien es igual a 0, el punto $P(0,0)$ es un minimo local.
Ahora analizando las derivadas parciales de segundo orden podemos ver que:

$(x^2+1)^{(y^2)} > 0$ para todo x, y

$y^2(x^2+1)^{(y^2-1)} > 0$ para todo x, y

$(y^2+1) > 0$ para todo y

$(2x)^2 > 0$ para todo x

por tanto $\frac{\partial^2 f}{\partial x^2} > 0$ para todo x, y

$\ln(x^2+1) > 0$ para todo x

$(x^2+1)^{(y^2+1)} > 0$ para todo x, y

$\ln(x^2+1)y^2 > 0$ para todo x, y

$e^y + e^y > 0$ para todo y

Por tanto $\frac{\partial^2 f}{\partial y^2} > 0$ para todo x, y

Para el analisis de convexidad de la funcion use las librerias de simpy y numpy de python, las cuales tienen metodo de calculo de la matriz hessiana y en dependencia de esta empieza a ver convexidad en todos los puntos en un rango especifico, lo cual determino no convexidad en numerosos puntos, para verificar se puede compilar el archivo check_convexity.py y ver los resultados.

3 Descripcion de los algoritmos

Los metodos utilizados son el de Newton y el de Maximo descenso:

El metodo de Maximo descenso es un algoritmo iterativo que busca el minimo siguiendo la direccion opuesta del vector gradiente, el maximo descenso de la funcion, consiste en comenzar con un punto x_0 y luego $x_{(k+1)} = x_k – a_k df(x_k)$ para cada k iteracion. Este algoritmo tiene un bajo costo computacional dado que depende unicamente de las segundas derivadas, tiene una convergencia rapida en pocas iteraciones pero no es tan preciso.

El Metodo de Newton en cambio usa informacion de las derivadas de segundo orden, de la matriz Hessiana, aproximando la funcion localmente por una cuadratica, comienza en un punto x_0 inicial y $x_{(k+1)} = x_k – (d^2 f(x_k))^{-1} df(x_k)$ luego en cada iteracion, dadas sus caracteristicas tiene resultados bastante precisos y muestra una gran robustez bajo condiciones adversas, aunque requiere de mayor cantidad de iteraciones y de mayor costo computacional ya que depende de la matriz hessiana.

4 Experimentacion

En el archivo config.json tengo un conjunto de categorias, cada una con 24 puntos para la experimentacion con los algoritmos, a continuacion un breve resumen de lo visto con ambos algoritmos en cada categoria:

4.1 Cerca del minimo

Son puntos cercanos al minimo global (0,0) para probar convergencia rapida cerca del optimo y medir velocidad de convergencia en condiciones optimas.
Ambos tuvieron 10 de 24 puntos convergentes, ambos fallaron cuando $x < 0$, aunque Newton tuvo una convergencia en menor cantidad de iteraciones.

4.2 Lejos del minimo

Son puntos alejados del minimo para probar robustez de los metodos desde inicializaciones malas.
Mucha mayor robustez con el metodo de Newton, puesto que tuvo muchos mas puntos convergentes, solo que a veces diverge al infinito, Maximo descenso solo tuvo un punto convergente.

4.3 Valores grandes en x

Probamos con valores grandes de x y menores en y puesto que funciones que crecen rapido en un termino pueden resultar desafiantes para la algoritmos.
5 puntos mas que convergieron con Newton que con Maximo descenso, puesto que este metodo se maneja mejor en valores grandes de x y en menor cantidad de iteraciones.

4.4 Valores grandes en y

Aqui tenemos valores grandes de y para probar como se comportan los algoritmos ante la exponencialidad de e^y, puesto que estos casos son numericamente delicados.

Ambos tuvieron dificultades con valores grandes de y, pero Newton tuvo mejores resultados, aunque tuvo mayor cantidad de iteraciones.

4.5 Zona Plataforma

Estos son regiones de gradiente bajo por lo cual son zonas de curvatura suave, ya que muchos algoritmos no saben a donde dirigirse con gradientes pegados a 0.
Maximo descenso tuvo convergencia en casi todos los puntos y en no muchas mas iteraciones que Newton, que fue un poco mas rapido.

4.6 Puntos criticos estacionarios

Puntos donde el gradiente es 0 o casi 0, ya que los metodos deberian ser capaces de no estancarse en los puntos sillas.
Ambos fallan en los puntos sobre los ejes donde el gradiente es 0, por lo cual tuvieron pocos puntos convergiendo.

4.7 Direcciones diagonales

Son puntos alineados con las diagonales y antidiagonales, dado que en problemas reales el optimo no esta alineado con los ejes y testear asi en trayectorias que cumplan esa no alineacion.
Newton claramente superior en cuestion de robustez en direcciones diagonales, aunque Maximo descenso tuvo muchas menos iteraciones

4.8 Cerca de los ejes coordenados

Probar con puntos muy cercanos a los ejes coordenados puesto que este tipo de puntos pueden causar problemas numericos.
Los dos metodos tienen bastantes fallos cerca de los ejes pero Newton es mas rapido.

4.9 Valores intermedios

Son valores moderados para testeo general ya que suelen ser los casos de uso mas comunes.
En este caso Newton es mas rapido pero Maximo descenso converge mas a menudo.

4.10 Patrones especiales

Son patrones de puntos especiales para verificar debilidades especificas de los algoritmos forzando comportamientos limites.
Newton es mucho mas robusto en este tipo de patrones complejos.

4.11 Regiones en cuadrantes

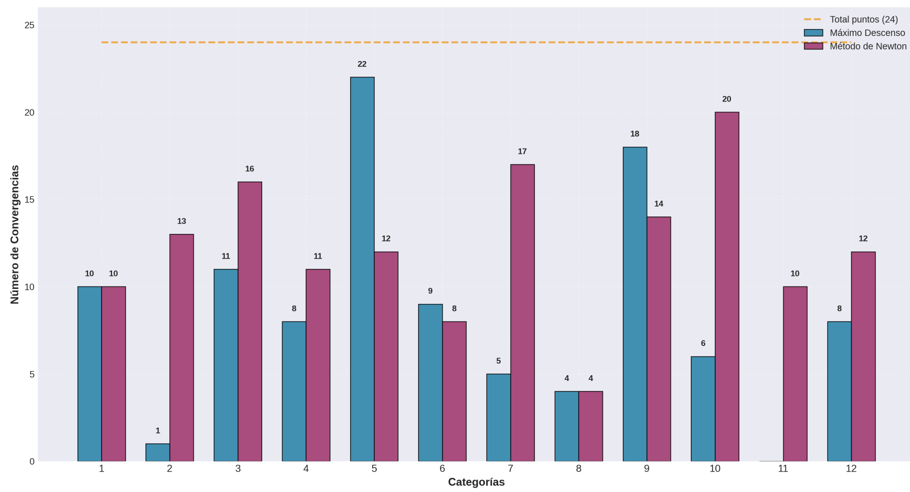
Son puntos distribuidos en los 4 cuadrantes para asegurar una cobertura completa del espacio de busqueda.
Newton es la unica opcion viable en cuadrantes con valores grandes, dado que Maximo descenso no tuvo valores convergentes.

4.12 Escala logaritmica

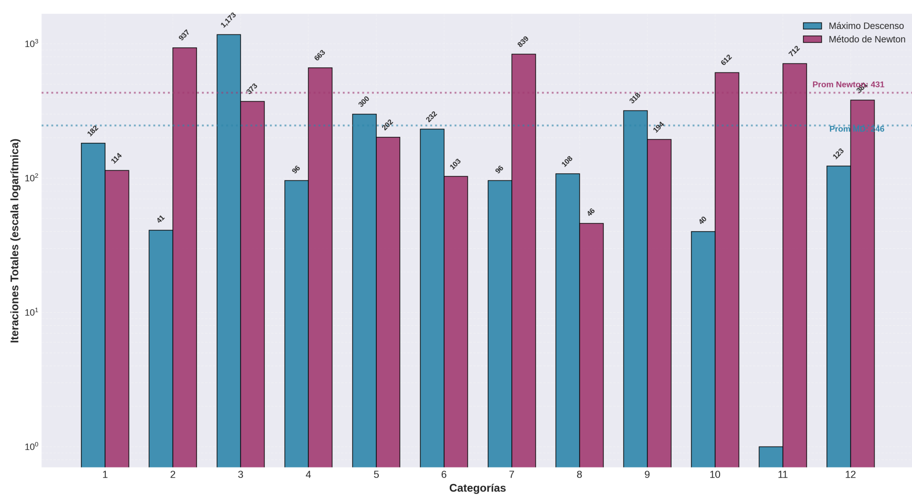
Estos son puntos espaciados logaritmicamente ya que los algoritmos deben funcionar independientemente de la escala.
Como los gradientes son nulos en la mayoria de los casos ambos fallan, teniendo solo 1 punto convergente cada uno.

A continuacion unas tablas que muestran los resultados de evaluar los algoritmos en los puntos mencionados anteriormente:

CONVERGENCIAS POR CATEGORÍA - Máximo Descenso vs Newton



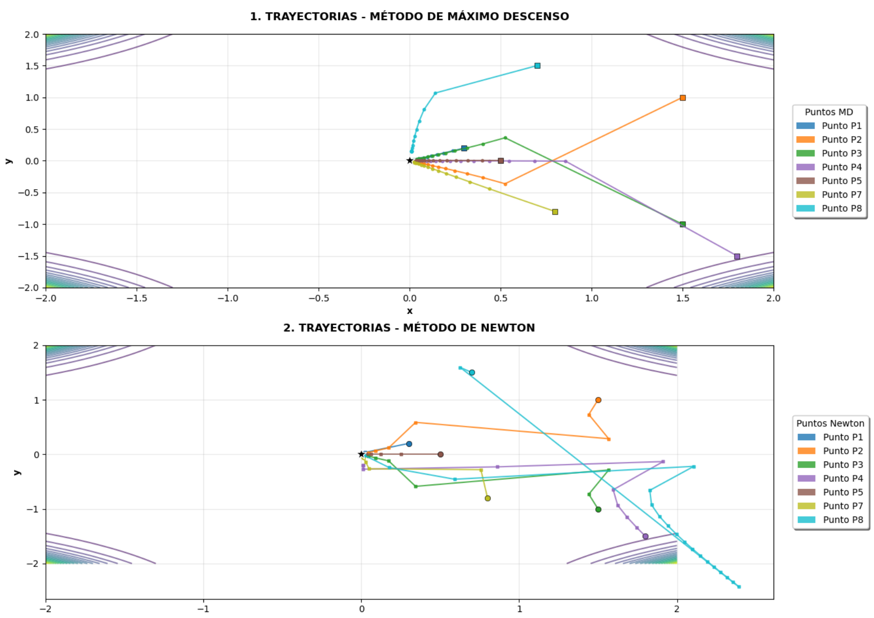
ITERACIONES POR CATEGORÍA - Máximo Descenso vs Newton

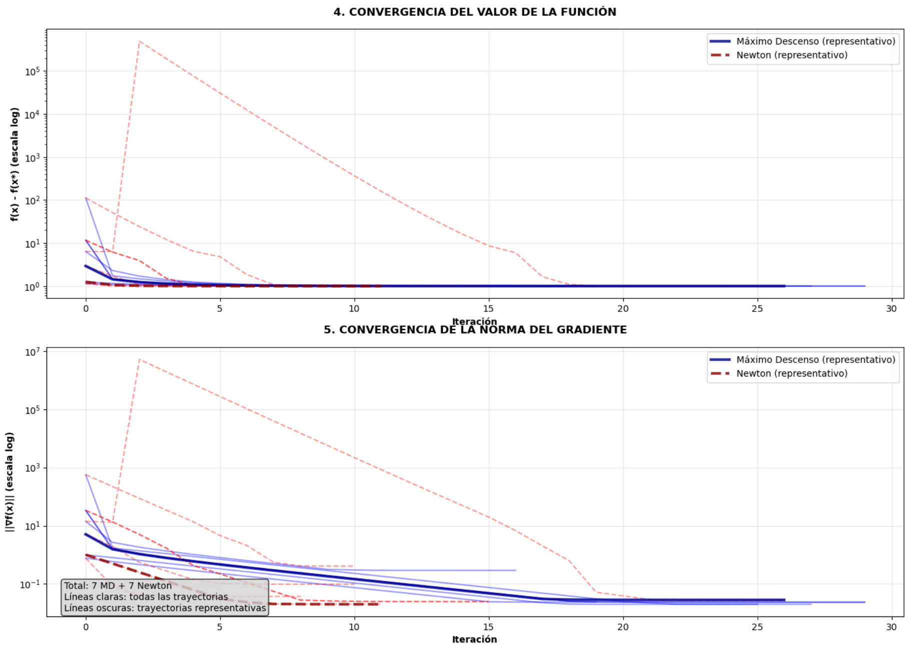


En el init_points.json se puede encontrar un conjunto de puntos los cuales estan seleccionados de la siguiente forma:

- El punto 1 es un punto cercano al minimo
- Los puntos del 2 al 4 son puntos lejos del minimo
- Los 5 y 6 se encuentran en los ejes
- Los 7 y 8 son puntos cerca de puntos potenciales criticos

En las siguientes tablas se evidencian los resultados de convergencia y de trayectoria obtenidos con estos puntos, para ver un comportamiento mas detallado de los algoritmos en estos casos puntuales:




4. CONVERGENCIA DEL VALOR DE LA FUNCIÓN
f(x) - f(x*) (escala log)
Iteración
Máximo Descenso (representativo)
Newton (representativo)
5. CONVERGENCIA DE LA NORMA DEL GRADIENTE
||∇f(x)|| (escala log)
Iteración
Máximo Descenso (representativo)
Newton (representativo)
Total: 7 MD + 7 Newton
Líneas claras: todas las trayectorias
Líneas oscuras: trayectorias representativas

5 Conclusiones

Los resultados de dichos algoritmos pueden verse con el codigo que puedes encontrar en el `https://github.com/IamSkale/tarea_MO` donde en el config.json puedes modificar los puntos que quieras verificar y obtener resultados distintos a los mostrados en este informe.

A modo de resumen pudimos ver con este codigo que el metodo de Newton es mucho mas preciso que el metodo de Maximo Descenso, puesto que tuvo un mayor indice de puntos convergentes en cada categoria de las analizadas anteriormente, ademas se vio mayor robustez desde puntos en condiciones no optimas y convergencia mas rapida cerca del optimo, en cambio el algoritmo de Maximo descenso tuvo mayor eficiencia en cantidad de iteraciones, es mas estable numericamente y tiene menor costo computacional.
Dicho esto se recomienda el uso de estos algoritmos pero cada uno en casos especificos, por ejemplo:

- Newton cuando la precision sea prioridad.
- Maximo descenso cuando los recursos son mas limitados.