optim_tarea06

March 25, 2024

1 Curso de Optimización I

1.1 Tarea 6

Descripción:	Fechas
Fecha de publicación del documento:	**Marzo 25, 2024**
Fecha límite de entrega de la tarea:	**Abril 14, 2024**

1.1.1 Indicaciones

- Envie el notebook con los códigos y las pruebas realizadas de cada ejercicio.
- Si se requiren algunos scripts adicionales para poder reproducir las pruebas, agreguelos en un ZIP junto con el notebook.
- Genere un PDF del notebook y envielo por separado.

1.2 Ejercicio 1 (3 puntos)

1. Programe el método de gradiente conjugado lineal, Algoritmo 1 de la Clase 18, para resolver el sistema de ecuaciones $\mathbf{A}\mathbf{x} = \mathbf{b}$, donde $\mathbf{A}$ es una matriz simétrica y definida positiva.

   Haga que la función devuelva el último punto $\mathbf{x}_k$, el último residual $\mathbf{r}_k$, el número de iteraciones k y una variable binaria *bres* que indique si se cumplió el criterio de paro ($bres = True$) o si el algoritmo terminó por iteraciones ($bres = False$).

2. Pruebe el algoritmo para resolver el sistema de ecuaciones

$$\mathbf{A}_1\mathbf{x} = \mathbf{b}_1$$

donde

$$\mathbf{A}_1 = n\mathbf{I} + \mathbf{1} = \begin{bmatrix} n & 0 & \cdots & 0 \\ 0 & n & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & n \end{bmatrix} + \begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & 1 & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \cdots & 1 \end{bmatrix}, \qquad \mathbf{b}_1 = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix},$$

n es la dimensión de la variable independiente $\mathbf{x} = (x_1, x_2, ..., x_n)$, $\mathbf{I}$ es la matriz identidad y $\mathbf{1}$ es la matriz llena de 1's, ambas de tamaño n.

También aplique el algoritmo para resolver el sistema

$$\mathbf{A}_2\mathbf{x} = \mathbf{b}_2$$

donde $\mathbf{A}_2 = [a_{ij}]$ con

$$a_{ij} = exp\left(-0.25(i-j)^2\right), \qquad \mathbf{b}_2 = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}$$

- Use $\mathbf{x}_0$ como el vector cero, el máximo número de iteraciones $N = n$ y una toleracia $\tau = \sqrt{n}\epsilon_m^{1/3}$, donde ϵ_m es el épsilon máquina.
- Pruebe el algoritmo resolviendo los dos sistemas de ecuaciones con $n = 10, 100, 1000$ y en cada caso imprima la siguiente información
- la dimensión n,
- el número k de iteraciones realizadas,
- las primeras y últimas 4 entradas del punto $\mathbf{x}_k$ que devuelve el algoritmo,
- la norma del residual $\mathbf{r}_k$,
- la variable *bres* para saber si el algoritmo puedo converger.

1.2.1 Solución:

[ ]:

```
```

[ ]:

```
```

1.3 Ejercicio 2 (3.5 puntos)

Programar el método de gradiente conjugado no lineal descrito en el Algoritmo 3 de Clase 19 usando la fórmula de Fletcher-Reeves:

$$\beta_{k+1} = \frac{\nabla f_{k+1}^\top \nabla f_{k+1}}{\nabla f_k^\top \nabla f_k}$$

1. Escriba la función que implemente el algoritmo.

- La función debe recibir como argumentos $\mathbf{x}_0$, la función f y su gradiente, el número máximo de iteraciones N, la tolerancia τ, y los parámetros para el algoritmo de backtracking: factor ρ, la constante c_1 para la condición de descenso suficiente, la constante c_2 para la condición de curvatura, y el máximo número de iteraciones N_b.

- Agregue al algoritmo un contador nr que se incremente cada vez que se aplique el reinicio, es decir, cuando se hace $\beta_{k+1} = 0$.

- Para calcular el tamaño de paso α_k use el algoritmo de backtracking usando las condiciones de Wolfe con el valor inicial $\alpha_{ini} = 1$.

- Haga que la función devuelva el último punto $\mathbf{x}_k$, el último gradiente $\mathbf{g}_k$, el número de iteraciones k y una variable binaria $bres$ que indique si se cumpli'o el criterio de paro ($bres = True$) o si el algoritmo terminó por iteraciones ($bres = False$), y el contador \$br $\cdot$.

2. Pruebe el algoritmo usando la siguientes funciones con los puntos iniciales dados:

Función de cuadrática 1: Para $\mathbf{x} = (x_1, x_2, ..., x_n)$

- $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^\top \mathbf{A}_1 \mathbf{x} - \mathbf{b}_1^\top \mathbf{x}$, donde $\mathbf{A}_1$ y $\mathbf{b}_1$ están definidas como en el Ejercicio 1.
- $\mathbf{x}_0 = (0, ..., 0) \in \mathbb{R}^{10}$
- $\mathbf{x}_0 = (0, ..., 0) \in \mathbb{R}^{100}$
- $\mathbf{x}_0 = (0, ..., 0) \in \mathbb{R}^{1000}$

Función de cuadrática 2: Para $\mathbf{x} = (x_1, x_2, ..., x_n)$

- $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^\top \mathbf{A}_2 \mathbf{x} - \mathbf{b}_2^\top \mathbf{x}$, donde $\mathbf{A}_2$ y $\mathbf{b}_2$ están definidas como en el Ejercicio 1.
- $\mathbf{x}_0 = (0, ..., 0) \in \mathbb{R}^{10}$
- $\mathbf{x}_0 = (0, ..., 0) \in \mathbb{R}^{100}$
- $\mathbf{x}_0 = (0, ..., 0) \in \mathbb{R}^{1000}$

Función de Beale : Para $\mathbf{x} = (x_1, x_2)$

$$f(\mathbf{x}) = (1.5 - x_1 + x_1 x_2)^2 + (2.25 - x_1 + x_1 x_2^2)^2 + (2.625 - x_1 + x_1 x_2^3)^2.$$

- $\mathbf{x}_0 = (2, 3)$

Función de Himmelblau: Para $\mathbf{x} = (x_1, x_2)$

$$f(\mathbf{x}) = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2.$$

- $\mathbf{x}_0 = (2, 4)$

Función de Rosenbrock: Para $\mathbf{x} = (x_1, x_2, ..., x_n)$

$$f(\mathbf{x}) = \sum_{i=1}^{n-1} \left[100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2\right] \quad n \geq 2.$$

- $\mathbf{x}_0 = (-1.2, 1.0) \in \mathbb{R}^2$
- $\mathbf{x}_0 = (-1.2, 1.0, ..., -1.2, 1.0) \in \mathbb{R}^{20}$
- $\mathbf{x}_0 = (-1.2, 1.0, ..., -1.2, 1.0) \in \mathbb{R}^{40}$

3. Fije $N = 5000$, $\tau = \sqrt{n}\epsilon_m^{1/3}$, donde n es la dimensión de la variable $\mathbf{x}$ y ϵ_m es el épsilon máquina. Para backtracking use $\rho = 0.5$, $c_1 = 0.001$, $c_2 = 0.01$, $N_b = 500$.

4. Para cada función de prueba imprima

- la dimensión n,
- $f(\mathbf{x}_0)$,

- el número k de iteraciones realizadas,
- $f(\mathbf{x}_k)$,
- las primeras y últimas 4 entradas del punto $\mathbf{x}_k$ que devuelve el algoritmo,
- la norma del vector gradiente $\mathbf{g}_k$,
- la variable *bres* para saber si el algoritmo puedo converger.
- el número de reinicios nr.

1.3.1 Solución:

[ ]:

[ ]:

1.4 Ejercicio 3 (3.5 puntos)

Programar el método de gradiente conjugado no lineal de usando la fórmula de Hestenes-Stiefel:

En este caso el algoritmo es igual al del Ejercicio 2, con excepción del cálculo de β_{k+1}. Primero se calcula el vector $\mathbf{y}_k$ y luego β_{k+1}:

$$\mathbf{y}_k = \nabla f_{k+1} - \nabla f_k$$

$$\beta_{k+1} = \frac{\nabla f_{k+1}^\top \mathbf{y}_k}{\nabla p_k^\top \mathbf{y}_k}$$

1. Repita el Ejercicio 2 usando la fórmula de Hestenes-Stiefel.
2. ¿Hay alguna diferencia que indique que es mejor usar la fórmula de Hestenes-Stiefel respesto a Fletcher-Reeves?
3. La cantidad de reinicios puede indicar que tanto se comporta el algoritmo como el algoritmo de descenso máximo. Agregue un comentario sobre esto de acuerdo a los resultados obtenidos para cada fórmula.

1.4.1 Solución:

[ ]:

[ ]: