Objetivos:

- Comparar los algoritmos del gradiente conjugado.
- Ver que llegan en ℓ iteraciones, donde ℓ es el número de eigenvalores distintos.

Recordamos la teoría. Sean $A \in \mathbb{R}^{n\times n}$ simétrica positiva definida y $\vec{\boldsymbol{b}} \in \mathbb{R}^n$.
Vimos que la única solución del sistema

$$A\vec{\boldsymbol{x}} = \vec{\boldsymbol{b}}$$

también es el único mínimo de la función

$$\phi(\vec{\boldsymbol{x}}) \coloneqq \frac{1}{2}\vec{\boldsymbol{x}}^\top A\vec{\boldsymbol{x}} - \vec{\boldsymbol{x}}^\top\vec{\boldsymbol{b}}\,.$$

Sabemos que

- $\nabla\phi(\vec{\boldsymbol{x}}) = A\vec{\boldsymbol{x}} - \vec{\boldsymbol{b}} =: \vec{\boldsymbol{r}}(\vec{\boldsymbol{x}})$
- Dada una dirección $\vec{\boldsymbol{d}}$, la solución del problema

  $$\min_{\alpha\in\mathbb{R}} \phi(\vec{\boldsymbol{x}} + \alpha\vec{\boldsymbol{d}})$$

  es

  $$\alpha_\star = -\frac{\vec{\boldsymbol{d}}^T\nabla\phi(\vec{\boldsymbol{x}})}{\vec{\boldsymbol{d}}^T A\vec{\boldsymbol{d}}} = -\frac{\vec{\boldsymbol{d}}^T\vec{\boldsymbol{r}}(\vec{\boldsymbol{x}})}{\vec{\boldsymbol{d}}^T A\vec{\boldsymbol{d}}}\,.$$
- Tomando un vector canónico, digamos $\vec{\boldsymbol{e}}_j$, tenemos

  $$\alpha_\star = \frac{-[\vec{\boldsymbol{r}}(\vec{\boldsymbol{x}})]_j}{A_{jj}}\,.$$

  y el nuevo punto $\vec{\boldsymbol{x}} + \alpha_\star\vec{\boldsymbol{e}}_j$ solo cambia en una entrada.

Observaciones y algoritmos.

- Si $A \in \mathbb{R}^{n\times n}$ es diagonal y positiva definida, entonces los vectores canónicos son A-conjugados.

  En este caso, el método de direcciones conjugadas (con los vectores canónicos como base A-conjugada) coincide con el método de descenso por coordenadas para funciones cuadráticas cuya hessiana A es diagonal y positiva definida.

- En seguida describimos el método de **descenso cíclico** por coordenadas para funciones cuadráticas cuya hessiana es positiva definida (no necesariamente diagonal).

Algoritmo: Descenso cíclico (para cuadráticas)

Dados $A \in \mathbb{R}^{n\times n}$ simétrica positiva definida y $\vec{\boldsymbol{b}}, \vec{\boldsymbol{x}}_0 \in \mathbb{R}^n$.

Sean $\left\{\vec{\boldsymbol{e}}_j\right\}_j$ los vectores canónicos.

Iniciar $\vec{\boldsymbol{r}}_0 \leftarrow A\vec{\boldsymbol{x}}_0 - \vec{\boldsymbol{b}}\,, \quad k \leftarrow 0\,, \quad j \leftarrow 1$

Mientras $\|\vec{\boldsymbol{r}}_k\| > tol$

$$\alpha_k \leftarrow -\frac{\vec{\boldsymbol{r}}_k^T \vec{\boldsymbol{e}}_j}{\vec{\boldsymbol{e}}_j^T A \vec{\boldsymbol{e}}_j}$$

$$\vec{\boldsymbol{x}}_{k+1} \leftarrow \vec{\boldsymbol{x}}_k + \alpha_k \vec{\boldsymbol{e}}_j$$

$$\vec{\boldsymbol{r}}_{k+1} \leftarrow A\vec{\boldsymbol{x}}_{k+1} - \vec{\boldsymbol{b}}$$

$$j \leftarrow 1 + \operatorname{mod}(j, n)$$

$$k \leftarrow k + 1$$

fin (mientras)

Ejercicio:

- Implementar el algoritmo *Descenso cíclico*, tal que, no se usan operaciones con vectores para calcular α_k y $\vec{\boldsymbol{x}}_{k+1}$.

Algoritmo 3 (CG versión 1). Ver notas en la página 64.

Algoritmo 3 (CG versión práctica). La versión 1 con las nuevas formulas para α_k y β_{k+1}, ver la página 65.

Experimentos.

1) El algoritmo del descenso cíclico termina en n iteraciones cuando A es diagonal.

Haga un experimento.

2) Use las dos versiones del algoritmo CG (versión 1 y versión práctica) y para resolver el sistema $A\vec{\boldsymbol{x}} = \vec{\boldsymbol{b}}$.

- verifique si los algoritmos llegan en n iteraciones.
- compare los residuos y las soluciones.
- Supuestamente, la versión práctica es más precisa. ¿Lo puedes ver en los resultados?

Para ese experimento tome la matriz (tri-diagonal) $A \in \mathbb{R}^{50\times 50}$ del spline cúbico (sobre una malla equidistante), es decir

$$A = \begin{pmatrix} 2 & 1 & & \\ 1 & 2 & \ddots & \\ & \ddots & \ddots & 1 \\ & & 1 & 2 \end{pmatrix}$$

útil: `np.diag`

y $\vec{\boldsymbol{b}} \in \mathbb{R}^{50}$ el vector de unos.

3) En el primer paso el método CG es el método de mayor descenso. Vimos que este (último) método llega en un paso si los eigenvalores de A son iguales, es decir, los conjuntos de nivel de ϕ en $\mathbb{R}^n$ son bolas.

Si se restringe ϕ a un eigenespacio $\mathrm{Eig}_\lambda(A) = \{\vec{\boldsymbol{v}} \colon A\vec{\boldsymbol{v}} = \lambda\vec{\boldsymbol{v}}\}$ sus conjuntos de nivel son bolas en $\mathbb{R}^{\dim(\mathrm{Eig}_\lambda(A))}$. Eso da la intuición, que para cada eigenvalor alcanza un paso para encontrar el mínimo en esta bola.

Verifique esta intuición:

a) Usar `np.random.rand` para construir una matriz aleatoria W. Use python para encontrar la factorización QR de W. Luego define A simétrica pos. def. por

$$A \coloneqq QDQ^T \qquad \text{para alguna diagonal } D.$$

b) Escoger D tal que tenga un eigenvalor, 2, 5, 10, n eigenvalores distintos pero con dimensión más grande.

c) ver cuantas iteraciones requiere el algoritmo CG.

4) Un ejemplo extremadamente mal condicionado para $n > 7$.

Usar `A = LA.pascal(n);  b = np.ones(n);` (Entonces $\vec{\boldsymbol{x}}^\star = \vec{\boldsymbol{e}}_1$ para todo n).

El descenso ciclico que empieza en $\vec{\boldsymbol{x}}_0 = \vec{\boldsymbol{0}}$ encuentra la solución en un solo paso. Pero, debido a errores de redondeo, los algoritmos CG no siempre llegan en n iteraciones a la solución. Hacer dos plots

$$\text{residuos vs. paso}(k) \qquad \text{y} \qquad \text{error en norma máxima vs. paso}(k)\,.$$