Reconstrucción de imágenes tomográficas

Huaier, Marenco, Salvia

21/12/2018

El problema

- Tenemos un cuerpo bidimensional con densidad variable en cada punto, y queremos reconstruir esas densidades.

El problema

- Tenemos un cuerpo bidimensional con densidad variable en cada punto, y queremos reconstruir esas densidades.
- Vamos a discretizar el cuerpo en celdas y pensarlo como una imagen en escala de grises, donde la intensidad del color representa la densidad del cuerpo en cada celda.

El problema

- Tenemos un cuerpo bidimensional con densidad variable en cada punto, y queremos reconstruir esas densidades.
- Vamos a discretizar el cuerpo en celdas y pensarlo como una imagen en escala de grises, donde la intensidad del color representa la densidad del cuerpo en cada celda.
- Vamos a generar rayos que atraviesen la imagen para calcular el tiempo que se tarda en atravesar una sección del cuerpo.

El problema

- Tenemos un cuerpo bidimensional con densidad variable en cada punto, y queremos reconstruir esas densidades.
- Vamos a discretizar el cuerpo en celdas y pensarlo como una imagen en escala de grises, donde la intensidad del color representa la densidad del cuerpo en cada celda.
- Vamos a generar rayos que atraviesen la imagen para calcular el tiempo que se tarda en atravesar una sección del cuerpo.
- Si generamos suficientes rayos, tendremos información suficiente para calcular la densidad en cada celda. La información viene dada por las ecuaciones:

$$t_k = \sum_{i=1}^{n} \sum_{j=1}^{n} d_{ij}^{(k)} v_{ij}^{-1}$$

o por la ecuación matricial $Ds = t$.

El problema del problema

- Las mediciones de los tiempos no son exactas, por lo que no vamos a poder encontrar una solución exacta del problema.

El problema del problema

- Las mediciones de los tiempos no son exactas, por lo que no vamos a poder encontrar una solución exacta del problema.
- Entonces resolveremos el sistema $Ds = t$ de manera aproximada por *cuadrados mínimos*.

El problema del problema

- Las mediciones de los tiempos no son exactas, por lo que no vamos a poder encontrar una solución exacta del problema.
- Entonces resolveremos el sistema $Ds = t$ de manera aproximada por *cuadrados mínimos*.
- Para ello utilizaremos la descomposición SVD de D, que será obtenida calculando los autovalores y autovectores de $D^t D$ con el método de la potencia con deflación.

Manejo de imágenes

- El programa sólo acepta imágenes `csv` cuadradas. Estas representan el cuerpo a estudiar; el valor de cada pixel es la densidad en un punto.

Manejo de imágenes

- El programa sólo acepta imágenes `csv` cuadradas. Estas representan el cuerpo a estudiar; el valor de cada pixel es la densidad en un punto.
- Recibe además un tamaño de celda y una cantidad arbitraria de niveles de ruido.

Manejo de imágenes

- El programa sólo acepta imágenes `csv` cuadradas. Estas representan el cuerpo a estudiar; el valor de cada pixel es la densidad en un punto.
- Recibe además un tamaño de celda y una cantidad arbitraria de niveles de ruido.
- La imagen reconstruida se agranda dividiendo cada pixel/celda en la cantidad de píxeles que tenían esa celda, para compararla con la original.

Simulación de rayos

- Cada $d_{ij}^{(k)}$ se calcula como la cantidad de píxeles atravesados por el rayo k en la celda ij.

Simulación de rayos

- Cada $d_{ij}^{(k)}$ se calcula como la cantidad de píxeles atravesados por el rayo k en la celda ij.
- Cada t_k se calcula como la suma de las intensidades de todos los píxeles atravesados por el rayo k.

Simulación de rayos

- Cada $d_{ij}^{(k)}$ se calcula como la cantidad de píxeles atravesados por el rayo k en la celda ij.
- Cada t_k se calcula como la suma de las intensidades de todos los píxeles atravesados por el rayo k.
- A cada t_k se le suma un valor aleatorio entre $-R$ y R, donde R es el nivel de ruido.

Representación de matrices

- La cantidad de celdas que atraviesa un rayo es escasa en comparación con la cantidad de celdas totales.

Representación de matrices

- La cantidad de celdas que atraviesa un rayo es escasa en comparación con la cantidad de celdas totales.
- Por eso, la matriz D será un arreglo de vectores ralos, donde cada vector representa una columna de la matriz.

Representación de matrices

- La cantidad de celdas que atraviesa un rayo es escasa en comparación con la cantidad de celdas totales.
- Por eso, la matriz D será un arreglo de vectores ralos, donde cada vector representa una columna de la matriz.
  - ¿Por qué las columnas y no las filas?
    $(D^t D)_{ij} = < \mathrm{col}_i(D), \mathrm{col}_j(D) >$

Representación de matrices

- La cantidad de celdas que atraviesa un rayo es escasa en comparación con la cantidad de celdas totales.
- Por eso, la matriz D será un arreglo de vectores ralos, donde cada vector representa una columna de la matriz.
    - ¿Por qué las columnas y no las filas?
      $(D^t D)_{ij} = < \mathrm{col}_i(D), \mathrm{col}_j(D) >$
- $D^t D$ se representa con un simple arreglo de arreglos de `double`'s.

Autovalores y autovectores

Utilizamos el método de la potencia con deflación.

$$v_k = \frac{D^t D v_{k-1}}{||D^t D v_{k-1}||}$$

$$\lambda_k = v_k^t D^t D v_k$$

$$D^t D \leftarrow D^t D - \lambda v v^t$$

- Este método puede llegar a ser muy inestable numéricamente.

Criterio de convergencia

La iteración termina cuando el vector v_k está cerca de ser un autovector de $D^t D$.

$$E_k = \frac{||D^t D v_k - \lambda_k v_k||}{||\lambda_k v_k||}$$

- Si $E_k \leq \varepsilon_1$ se devuelven el autovalor y autovector actuales.

Criterio de convergencia

La iteración termina cuando el vector v_k está cerca de ser un autovector de $D^t D$.

$$E_k = \frac{||D^t D v_k - \lambda_k v_k||}{||\lambda_k v_k||}$$

- Si $E_k \leq \varepsilon_1$ se devuelven el autovalor y autovector actuales.
- Si el error no mejoró luego de una iteración ($E_k = E_{k-1}$) entonces...

Criterio de convergencia

La iteración termina cuando el vector v_k está cerca de ser un autovector de $D^t D$.

$$E_k = \frac{||D^t D v_k - \lambda_k v_k||}{||\lambda_k v_k||}$$

- Si $E_k \leq \varepsilon_1$ se devuelven el autovalor y autovector actuales.
- Si el error no mejoró luego de una iteración $(E_k = E_{k-1})$ entonces...
  - Si $E_k > \varepsilon_2$, el autovalor es negativo o el autovalor es mucho más grande al anteriormente calculado se frenará el cálculo de autovalores y autovectores.

Criterio de convergencia

La iteración termina cuando el vector v_k está cerca de ser un autovector de $D^t D$.

$$E_k = \frac{||D^t D v_k - \lambda_k v_k||}{||\lambda_k v_k||}$$

- Si $E_k \leq \varepsilon_1$ se devuelven el autovalor y autovector actuales.
- Si el error no mejoró luego de una iteración $(E_k = E_{k-1})$ entonces...
  - Si $E_k > \varepsilon_2$, el autovalor es negativo o el autovalor es mucho más grande al anteriormente calculado se frenará el cálculo de autovalores y autovectores.
  - En caso contrario se devuelven el autovalor y autovector actuales.

Criterio de convergencia

La iteración termina cuando el vector v_k está cerca de ser un autovector de $D^t D$.

$$E_k = \frac{||D^t D v_k - \lambda_k v_k||}{||\lambda_k v_k||}$$

- Si $E_k \leq \varepsilon_1$ se devuelven el autovalor y autovector actuales.
- Si el error no mejoró luego de una iteración $(E_k = E_{k-1})$ entonces...
  - Si $E_k > \varepsilon_2$, el autovalor es negativo o el autovalor es mucho más grande al anteriormente calculado se frenará el cálculo de autovalores y autovectores.
  - En caso contrario se devuelven el autovalor y autovector actuales.
- Fijamos $\varepsilon_1 = 0{,}01$, $\varepsilon_2 = 0{,}1$.

SVD y cuadrados mínimos

- Para obtener esta descomposición se calcula $D^t D$ en vez de DD^t ya que la primera será de tamaño $n^2 \times n^2$ y la segunda de $m \times m$, sabiendo que en general $m > n$.

SVD y cuadrados mínimos

- Para obtener esta descomposición se calcula $D^t D$ en vez de DD^t ya que la primera será de tamaño $n^2 \times n^2$ y la segunda de $m \times m$, sabiendo que en general $m > n$.
- Calculamos U aprovechando que:

$$Dv_i = \sigma_i u_i$$

$$\implies u_i = Dv_i / \sigma_i$$

SVD y cuadrados mínimos

- $||Ds - t||^2 = ||\Sigma V^t s - U^t t||^2$

SVD y cuadrados mínimos

- $||Ds - t||^2 = ||\Sigma V^t s - U^t t||^2$
- Minimizar $||\Sigma V^t s - U^t t||^2$ es equivalente a minimizar $||\Sigma y - U^t t||^2$ y hacer $s = Vy$.

SVD y cuadrados mínimos

- $||Ds - t||^2 = ||\Sigma V^t s - U^t t||^2$
- Minimizar $||\Sigma V^t s - U^t t||^2$ es equivalente a minimizar $||\Sigma y - U^t t||^2$ y hacer $s = Vy$.
- Escribimos:
$$U^t t = \begin{pmatrix} c \\ d \end{pmatrix}$$

SVD y cuadrados mínimos

- $||Ds - t||^2 = ||\Sigma V^t s - U^t t||^2$
- Minimizar $||\Sigma V^t s - U^t t||^2$ es equivalente a minimizar $||\Sigma y - U^t t||^2$ y hacer $s = Vy$.
- Escribimos:
$$U^t t = \begin{pmatrix} c \\ d \end{pmatrix}$$
-
$$||\Sigma y - U^t t||^2 = \left\| \begin{pmatrix} \sigma_1 y_1 - c_1 \\ \vdots \\ \sigma_r y_r - c_r \end{pmatrix} \right\|^2 + ||d||^2$$

SVD y cuadrados mínimos

- $||Ds - t||^2 = ||\Sigma V^t s - U^t t||^2$
- Minimizar $||\Sigma V^t s - U^t t||^2$ es equivalente a minimizar $||\Sigma y - U^t t||^2$ y hacer $s = Vy$.
- Escribimos:
$$U^t t = \begin{pmatrix} c \\ d \end{pmatrix}$$
-
$$||\Sigma y - U^t t||^2 = \left\| \begin{pmatrix} \sigma_1 y_1 - c_1 \\ \vdots \\ \sigma_r y_r - c_r \end{pmatrix} \right\|^2 + ||d||^2$$
- Elegimos $y_i = c_i/\sigma_i$ para $i = 1 \ldots r$, $y_i = 0$ para $i = r + 1 \ldots n^2$.

SVD y cuadrados mínimos

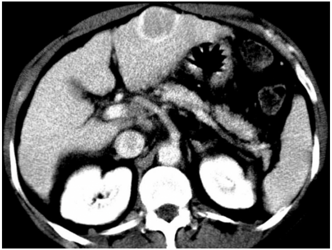
- $||Ds - t||^2 = ||\Sigma V^t s - U^t t||^2$
- Minimizar $||\Sigma V^t s - U^t t||^2$ es equivalente a minimizar $||\Sigma y - U^t t||^2$ y hacer $s = Vy$.
- Escribimos:
$$U^t t = \begin{pmatrix} c \\ d \end{pmatrix}$$
-
$$||\Sigma y - U^t t||^2 = \left\| \begin{pmatrix} \sigma_1 y_1 - c_1 \\ \vdots \\ \sigma_r y_r - c_r \end{pmatrix} \right\|^2 + ||d||^2$$
- Elegimos $y_i = c_i/\sigma_i$ para $i = 1 \dots r$, $y_i = 0$ para $i = r + 1 \dots n^2$.
- El resultado final $s = Vy$ es una combinación lineal de las primeras r columnas de V.

Número de condición de $D^t D$

- $D^t D$ es simétrica semi-definida positiva, entonces sus valores singulares son sus autovalores.
- El número de condición de $D^t D$ se calcula como $\kappa(D^t D) = \lambda_{\text{max}}/\lambda_{\text{min}}$, suponiendo que obtuvimos todos los autovalores.

Experimentación: tamaño de celda

Por un lado, se experimentó generando 20000 rayos aleatorios con un nivel de ruido 0.0 para observar cómo cambia la calidad visual, el PSNR y el número de condición de $D^t D$ según el tamaño de las celdas. Utilizamos las imágenes `tomo2.csv` y `tomo3.csv`.



(a) `tomo2`

(b) `tomo3`

Figura: Imágenes de prueba `tomo2` y `tomo3`.

Experimentación: tamaño de celda

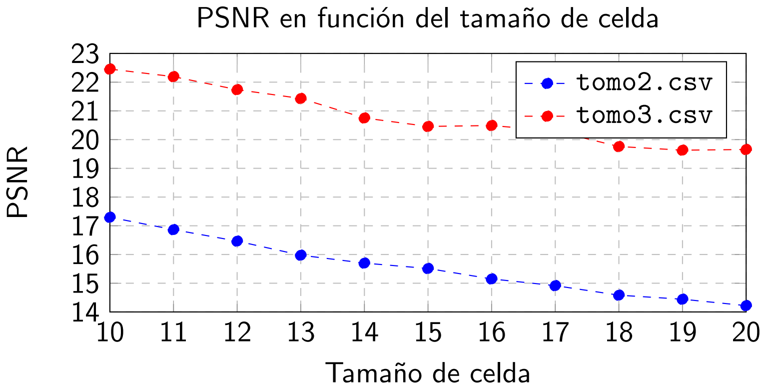


Figura: PSNR en función del tamaño de celda, para las imágenes `tomo2.csv` y `tomo3.csv`. Nivel de ruido 0.0, 20000 rayos aleatorios.

Experimentación: tamaño de celda

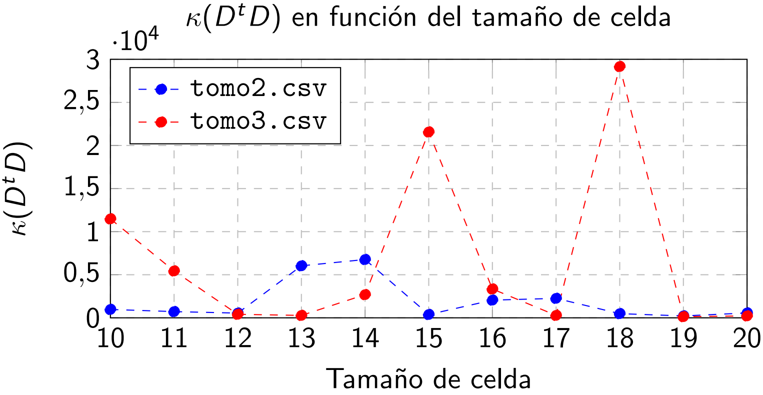
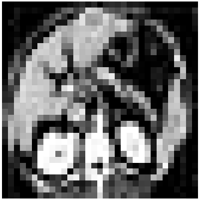


Figura: $\kappa(D^tD)$ en función del tamaño de celda, para las imágenes `tomo2.csv` y `tomo3.csv`. Nivel de ruido 0.0, 20000 rayos aleatorios.

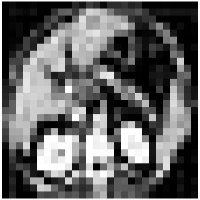
Experimentación: tamaño de celda

(a) Tamaño de celda 10

(b) Tamaño de celda 13



(c) Tamaño de celda 17

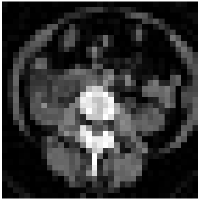


(d) Tamaño de celda 20

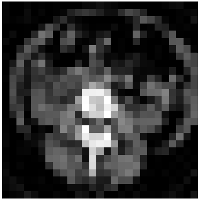
Figura: Imágen `tomo2.csv` reconstruida con diferentes tamaños de celda, nivel de ruido 0.0 y 20000 rayos aleatorios.

Experimentación: tamaño de celda

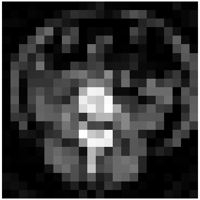
(a) Tamaño de celda 10



(b) Tamaño de celda 13



(c) Tamaño de celda 17

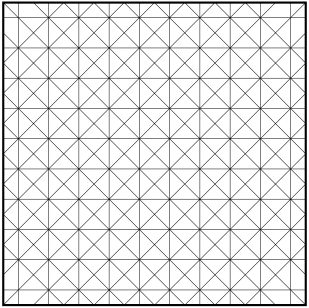


(d) Tamaño de celda 20

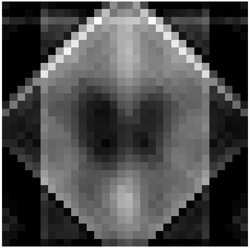
Figura: Imágen `tomo3.csv` reconstruida con diferentes tamaños de celda, nivel de ruido 0.0 y 20000 rayos aleatorios.

Rayos verticales, horizontales y diagonales

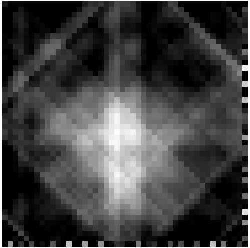
Se generan todos los rayos posibles que sean horizontales, verticales y diagonales con pendiente 1.



Experimentando: rayos verticales, horizontales y diagonales



(a) `phantom.csv`, con tamaño de celda 8



(b) `tomo3.csv`, con tamaño de celda 10

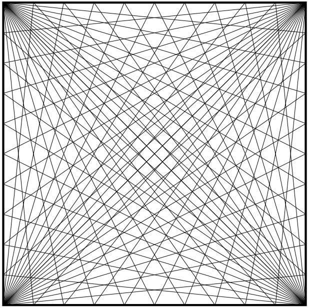
Figura: Imágenes `phantom.csv` y `tomo3.csv` reconstruidas con rayos horizontales, verticales y diagonales. Nivel de ruido 0.0.

Experimentando: rayos verticales, horizontales y diagonales

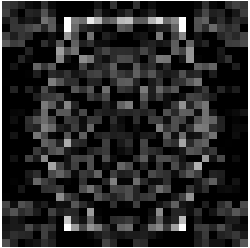
- `phantom.csv`:
  - PSNR: 15.2316
  - 183 de 1024 autovalores de $D^t D$ calculados.
  - Chequeado con Numpy: $D^t D$ singular.
- `tomo3.csv`:
  - PSNR: 17.0948
  - 283 de 2116 autovalores de $D^t D$ calculados.
  - Chequeado con Numpy: $D^t D$ singular.

Rayos que barren la imagen

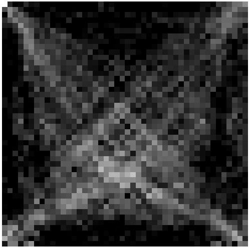
Consiste en pararse en cada una de las esquinas y generar rayos que vayan a todos los píxeles de los bordes no adyacentes.



Experimentando: rayos que barren la imagen



(a) `phantom.csv`, con tamaño de celda 8



(b) `tomo3.csv`, con tamaño de celda 10

Figura: Imágenes `phantom.csv` y `tomo3.csv` reconstruidas con rayos que barren la imagen desde las cuatro esquinas. Nivel de ruido 0.0.

Experimentando: rayos que barren la imagen

- `phantom.csv`:
  - PSNR: 12.4005
  - 790 de 1024 autovalores de $D^t D$ calculados.
  - Chequeado con Numpy: $\kappa(D^t D) = 1578918$.
- `tomo3.csv`:
  - PSNR: 13.9602
  - 1088 de 2116 autovalores de $D^t D$ calculados.
  - Chequeado con Numpy: $\kappa(D^t D) = 6470105$.

Rayos aleatorios

Dada una cantidad de rayos deseada m. Se genera cada rayo de la siguiente manera:

1. Se elige la trayectoria del rayo en términos de en qué borde partirá el rayo y hasta cuál llegará.

Rayos aleatorios

Dada una cantidad de rayos deseada m. Se genera cada rayo de la siguiente manera:

1. Se elige la trayectoria del rayo en términos de en qué borde partirá el rayo y hasta cuál llegará.
2. Se elige un punto de cada borde elegido, los cuales determinarán el rayo.

Experimentando: Cantidad de rayos aleatorios

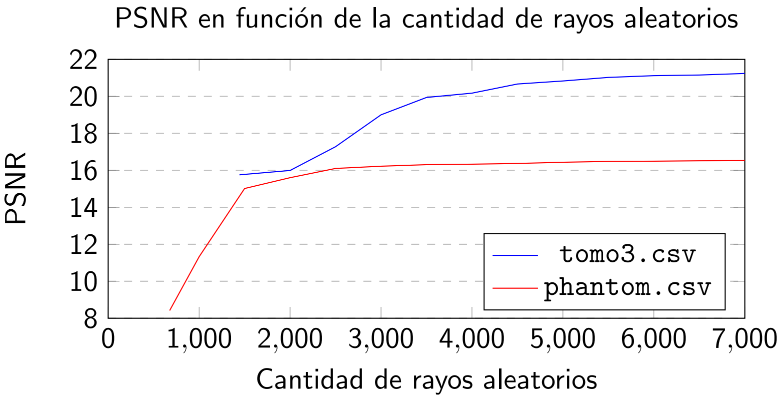


Figura: PSNR en función de cantidad de rayos aleatorios, para las imágenes `tomo3.csv` y `phantom.csv`, tamaños de celda 12 y 10 respectivamente, y nivel de ruido 0.0.

Experimentando: Cantidad de rayos aleatorios

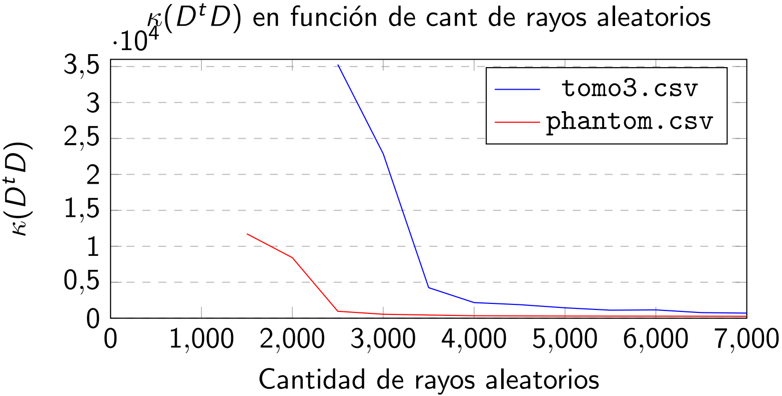
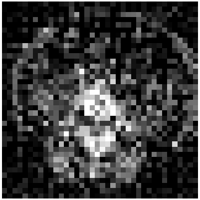
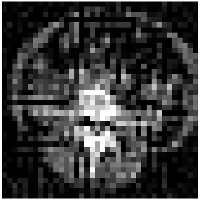


Figura: $\kappa(D^t D)$ en función de cantidad de rayos aleatorios, para las imágenes `tomo3.csv` y `phantom.csv`, tamaños de celda 12 y 10 respectivamente, y nivel de ruido 0.0.

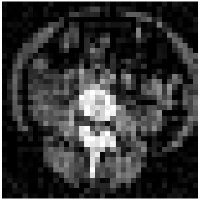
Experimentando: Cantidad de rayos aleatorios



(a) 1444 rayos aleatorios (D resulta cuadrada).



(b) 2500 rayos aleatorios.

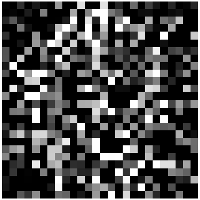


(c) 4000 rayos aleatorios.

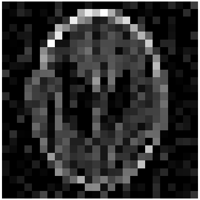
(d) 7000 rayos aleatorios.

Figura: Imagen `tomo3.csv` reconstruida con tamaño de celda 12, nivel de ruido 0.0. Se muestran resultados para diferentes cantidades de rayos aleatorios.

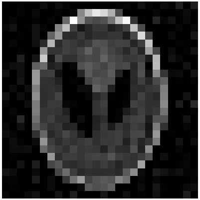
Experimentando: Cantidad de rayos aleatorios



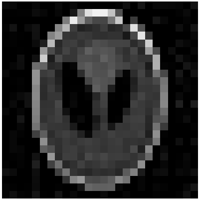
(a) 676 rayos aleatorios (D resulta cuadrada).



(b) 1500 rayos aleatorios.



(c) 4000 rayos aleatorios.

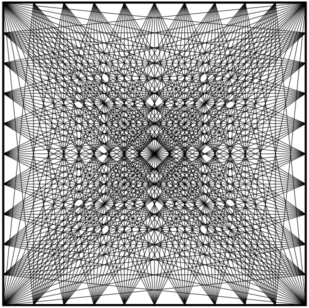


(d) 7000 rayos aleatorios.

Figura: Imagen `phantom.csv` reconstruida con tamaño de celda 10, nivel de ruido 0.0. Se muestran resultados para diferentes cantidades de rayos aleatorios.

Metodo "completo"

Se generan todos los rayos posibles que pasen por dos píxeles que estén en bordes opuestos de la imagen.



Experimentando: método completo

Figura: Imagen `tomo.csv` reconstruida con el método “completo”, tamaño de celda 2 y nivel de ruido 0.0.

Experimentando: nivel de ruido

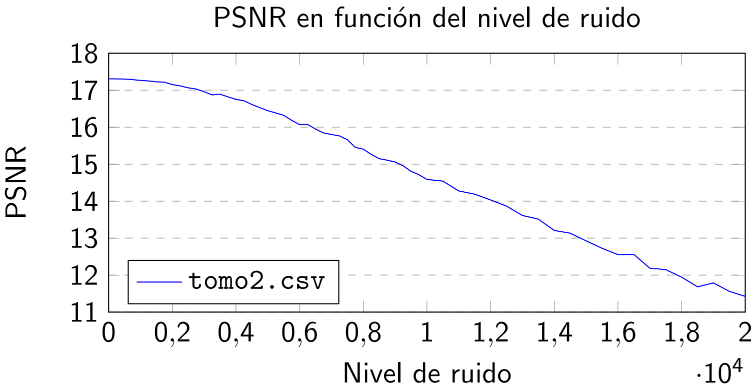


Figura: PSNR en función del nivel de ruido agregado a los tiempos de los rayos, para la imagen `tomo2.csv`, tamaño de celda 10 con 20000 rayos aleatorios.

Experimentando: nivel de ruido

(a) Nivel de ruido: 0.

(b) Nivel de ruido: 4000.

Figura: Imagen `tomo2.csv` reconstruida con tamaño de celda 10 y 20000 rayos aleatorios. Se muestran resultados para diferentes niveles de ruido.

Experimentando: nivel de ruido

(a) Nivel de ruido: 8000.

(b) Nivel de ruido: 12000.

Figura: Imagen `tomo2.csv` reconstruida con tamaño de celda 10 y 20000 rayos aleatorios. Se muestran resultados para diferentes niveles de ruido.

Experimentando: nivel de ruido

(a) Nivel de ruido: 16000.

(b) Nivel de ruido: 20000.

Figura: Imagen `tomo2.csv` reconstruida con tamaño de celda 10 y 20000 rayos aleatorios. Se muestran resultados para diferentes niveles de ruido.

Conclusiones

- La manera en la que se emiten los rayos es decisiva a la hora de evaluar qué tan buena fue la reconstrucción.

Conclusiones

- La manera en la que se emiten los rayos es decisiva a la hora de evaluar qué tan buena fue la reconstrucción.
- Un tamaño de celda más chico implica mejor calidad (pero es más costoso en tiempo).

Conclusiones

- La manera en la que se emiten los rayos es decisiva a la hora de evaluar qué tan buena fue la reconstrucción.
- Un tamaño de celda más chico implica mejor calidad (pero es más costoso en tiempo).
- La descomposición SVD es apropiada para hallar la solución de cuadrados mínimos. Pero es necesario contar con un método para obtener autovalores y autovectores eficazmente.

Conclusiones

- La manera en la que se emiten los rayos es decisiva a la hora de evaluar qué tan buena fue la reconstrucción.
- Un tamaño de celda más chico implica mejor calidad (pero es más costoso en tiempo).
- La descomposición SVD es apropiada para hallar la solución de cuadrados mínimos. Pero es necesario contar con un método para obtener autovalores y autovectores eficazmente.
- La aproximación de la solución mediante cuadrados mínimos es un método efectivo a la hora de reconstruir la imagen original.

Posibles extensiones

Algunos cambios que podrían hacerse son:

- Usar otro algoritmo para obtener autovalores y autovectores.

Posibles extensiones

Algunos cambios que podrían hacerse son:

- Usar otro algoritmo para obtener autovalores y autovectores.
- Usar las factorizaciones QR y Cholesky de $D^t D$, los métodos iterativos de Jacobi, Gauss-Seidel o la eliminación Gaussiana para resolver las ecuaciones normales.

Posibles extensiones

Algunos cambios que podrían hacerse son:

- Usar otro algoritmo para obtener autovalores y autovectores.
- Usar las factorizaciones QR y Cholesky de $D^t D$, los métodos iterativos de Jacobi, Gauss-Seidel o la eliminación Gaussiana para resolver las ecuaciones normales.
- Nuevos métodos de generación de rayos.