Taller 2 - Cuadrados Mínimos

Métodos Numéricos

Departamento de Computación, FCEyN,Universidad de Buenos Aires.

14 de Octubre de 2016

Hasta ahora...

A esta altura se podría decir que la tenemos bastante clara resolviendo sistemas $Ax = b$ con $A \in \mathbb{R}^{m \times n}$ y $m = n$.

Como venimos viendo en la materia, muchos problemas y aplicaciones reales se pueden resolver planteando sistemas de esta forma, y si A es inversible sabemos que existe una única solución y tenemos diversos métodos para calcularla o aproximarla.

Todo muy lindo con los sistemas cuadrados, pero ¿qué pasa si $m \neq n$? Claramente el sistema podría no tener solución (o tener muchas). ¿Tiene sentido plantearse esto?

Hasta ahora...

A esta altura se podría decir que la tenemos bastante clara resolviendo sistemas $Ax = b$ con $A \in \mathbb{R}^{m \times n}$ y $m = n$.

Como venimos viendo en la materia, muchos problemas y aplicaciones reales se pueden resolver planteando sistemas de esta forma, y si A es inversible sabemos que existe una única solución y tenemos diversos métodos para calcularla o aproximarla.

Todo muy lindo con los sistemas cuadrados, pero ¿qué pasa si $m \neq n$? Claramente el sistema podría no tener solución (o tener muchas). ¿Tiene sentido plantearse esto?

Motivación

- A Gauss se le ocurrió calcular las órbitas de cuerpos celestes como planetas o cometas.
- Entonces pensó, fácil, una órbita elíptica la puedo determinar con una ecuación de 5 parámetros, asi que puedo tomar 5 observaciones de la posición de mi planeta, despejar y así saber su órbita.

Será tan fácil?...

- Hagámoslo nosotros..
- Las observaciones que tomamos podrían tener ruido (sobre todo tratándose de órbitas de planetas), por las dudas tomemos 5 más y resolvamos el sistema sobredeterminado $Ax \approx b$ (más adelante vemos como resolverlo).
- ¿Son iguales?
- Mientras más observaciones tomemos, más cerca vamos a estar de la órbita real.

Será tan fácil?...

- Hagámoslo nosotros..
- Las observaciones que tomamos podrían tener ruido (sobre todo tratándose de órbitas de planetas), por las dudas tomemos 5 más y resolvamos el sistema sobredeterminado $Ax \approx b$ (más adelante vemos como resolverlo).
- ¿Son iguales?
- Mientras más observaciones tomemos, más cerca vamos a estar de la órbita real.

Será tan fácil?...

- Hagámoslo nosotros..
- Las observaciones que tomamos podrían tener ruido (sobre todo tratándose de órbitas de planetas), por las dudas tomemos 5 más y resolvamos el sistema sobredeterminado $Ax \approx b$ (más adelante vemos como resolverlo).
- ¿Son iguales?
- Mientras más observaciones tomemos, más cerca vamos a estar de la órbita real.

Será tan fácil?...

- Hagámoslo nosotros..
- Las observaciones que tomamos podrían tener ruido (sobre todo tratándose de órbitas de planetas), por las dudas tomemos 5 más y resolvamos el sistema sobredeterminado $Ax \approx b$ (más adelante vemos como resolverlo).
- ¿Son iguales?
- Mientras más observaciones tomemos, más cerca vamos a estar de la órbita real.

Será tan fácil?...

- Hagámoslo nosotros..
- Las observaciones que tomamos podrían tener ruido (sobre todo tratándose de órbitas de planetas), por las dudas tomemos 5 más y resolvamos el sistema sobredeterminado $Ax \approx b$ (más adelante vemos como resolverlo).
- ¿Son iguales?
- Mientras más observaciones tomemos, más cerca vamos a estar de la órbita real.

Recapitulando

- Como vimos, resolver este tipo de sistemas nos sirve por ejemplo para tener mayor robustez cuando los datos que disponemos tienen ruido, cosa que sucede a menudo a la hora de experimentar.
- Sea $A \in \mathbb{R}^{m \times n}$, $m \neq n$. Nos vamos a concentrar en el caso más típico con $m \gg n$, es decir, sistemas sobredeterminados. En particular, la situación más comun que da origen a este tipo de problemas es *data fitting*, donde lo que buscamos es aproximar m muestras a partir de n variables o features.
- El método que vamos a ver hoy para resolver estos sistemas es cuadrados mínimos.

Recapitulando

- Como vimos, resolver este tipo de sistemas nos sirve por ejemplo para tener mayor robustez cuando los datos que disponemos tienen ruido, cosa que sucede a menudo a la hora de experimentar.
- Sea $A \in \mathbb{R}^{m \times n}$, $m \neq n$. Nos vamos a concentrar en el caso más típico con $m \gg n$, es decir, sistemas sobredeterminados. En particular, la situación más comun que da origen a este tipo de problemas es *data fitting*, donde lo que buscamos es aproximar m muestras a partir de n variables o features.
- El método que vamos a ver hoy para resolver estos sistemas es cuadrados mínimos.

Recapitulando

- Como vimos, resolver este tipo de sistemas nos sirve por ejemplo para tener mayor robustez cuando los datos que disponemos tienen ruido, cosa que sucede a menudo a la hora de experimentar.
- Sea $A \in \mathbb{R}^{m \times n}$, $m \neq n$. Nos vamos a concentrar en el caso más típico con $m \gg n$, es decir, sistemas sobredeterminados. En particular, la situación más comun que da origen a este tipo de problemas es *data fitting*, donde lo que buscamos es aproximar m muestras a partir de n variables o features.
- El método que vamos a ver hoy para resolver estos sistemas es cuadrados mínimos.

Recapitulando

- Como vimos, resolver este tipo de sistemas nos sirve por ejemplo para tener mayor robustez cuando los datos que disponemos tienen ruido, cosa que sucede a menudo a la hora de experimentar.
- Sea $A \in \mathbb{R}^{m \times n}$, $m \neq n$. Nos vamos a concentrar en el caso más típico con $m \gg n$, es decir, sistemas sobredeterminados. En particular, la situación más comun que da origen a este tipo de problemas es *data fitting*, donde lo que buscamos es aproximar m muestras a partir de n variables o features.
- El método que vamos a ver hoy para resolver estos sistemas es cuadrados mínimos.

Data fitting

- Supongamos que tenemos m puntos (t_i, y_i). Queremos obtener una función que ajuste lo "mejor" posible esos datos.
- Para nosotros mejor ajuste va a significar minimizar la suma de los cuadrados de la diferencia entre los datos reales y los que nos da nuestra función modelo, de ahí el nombre del método.
- Lo que queremos entonces es encontrar una función f tal que garantice:

$$\min_f \sum_{i=1}^{m} (y_i - f(t_i))^2 \tag{1}$$

- Para modelar f vamos a usar n parámetros, es decir que encontrar nuestra función modelo es encontrar el vector $x \in \mathbb{R}^n$ de parámetros que minimice (1).
- Nosotros vamos a trabajar con funciones que sean lineales en los componentes del vector de parámetros x, en particular polinomios: $f(t) = f(t, x) = x_1 + x_2 t + x_3 t^2 + \ldots + x_n t^{n-1}$

Data fitting

- Supongamos que tenemos m puntos (t_i, y_i). Queremos obtener una función que ajuste lo "mejor" posible esos datos.
- Para nosotros mejor ajuste va a significar minimizar la suma de los cuadrados de la diferencia entre los datos reales y los que nos da nuestra función modelo, de ahí el nombre del método.
- Lo que queremos entonces es encontrar una función f tal que garantice:

$$\min_f \sum_{i=1}^{m} (y_i - f(t_i))^2 \tag{1}$$

- Para modelar f vamos a usar n parámetros, es decir que encontrar nuestra función modelo es encontrar el vector $x \in \mathbb{R}^n$ de parámetros que minimice (1).
- Nosotros vamos a trabajar con funciones que sean lineales en los componentes del vector de parámetros x, en particular polinomios: $f(t) = f(t, x) = x_1 + x_2 t + x_3 t^2 + \ldots + x_n t^{n-1}$

Data fitting

- Supongamos que tenemos m puntos (t_i, y_i). Queremos obtener una función que ajuste lo "mejor" posible esos datos.
- Para nosotros mejor ajuste va a significar minimizar la suma de los cuadrados de la diferencia entre los datos reales y los que nos da nuestra función modelo, de ahí el nombre del método.
- Lo que queremos entonces es encontrar una función f tal que garantice:

$$\min_f \sum_{i=1}^{m} (y_i - f(t_i))^2 \tag{1}$$

- Para modelar f vamos a usar n parámetros, es decir que encontrar nuestra función modelo es encontrar el vector $x \in \mathbb{R}^n$ de parámetros que minimice (1).
- Nosotros vamos a trabajar con funciones que sean lineales en los componentes del vector de parámetros x, en particular polinomios: $f(t) = f(t, x) = x_1 + x_2 t + x_3 t^2 + \ldots + x_n t^{n-1}$

Data fitting

- Supongamos que tenemos m puntos (t_i, y_i). Queremos obtener una función que ajuste lo "mejor" posible esos datos.
- Para nosotros mejor ajuste va a significar minimizar la suma de los cuadrados de la diferencia entre los datos reales y los que nos da nuestra función modelo, de ahí el nombre del método.
- Lo que queremos entonces es encontrar una función f tal que garantice:

$$\min_f \sum_{i=1}^{m} (y_i - f(t_i))^2 \tag{1}$$

- Para modelar f vamos a usar n parámetros, es decir que encontrar nuestra función modelo es encontrar el vector $x \in \mathbb{R}^n$ de parámetros que minimice (1).
- Nosotros vamos a trabajar con funciones que sean lineales en los componentes del vector de parámetros x, en particular polinomios: $f(t) = f(t, x) = x_1 + x_2 t + x_3 t^2 + \ldots + x_n t^{n-1}$

Data fitting

- Supongamos que tenemos m puntos (t_i, y_i). Queremos obtener una función que ajuste lo "mejor" posible esos datos.
- Para nosotros mejor ajuste va a significar minimizar la suma de los cuadrados de la diferencia entre los datos reales y los que nos da nuestra función modelo, de ahí el nombre del método.
- Lo que queremos entonces es encontrar una función f tal que garantice:

$$\min_f \sum_{i=1}^{m} (y_i - f(t_i))^2 \tag{1}$$

- Para modelar f vamos a usar n parámetros, es decir que encontrar nuestra función modelo es encontrar el vector $x \in \mathbb{R}^n$ de parámetros que minimice (1).
- Nosotros vamos a trabajar con funciones que sean lineales en los componentes del vector de parámetros x, en particular polinomios: $f(t) = f(t, x) = x_1 + x_2 t + x_3 t^2 + \ldots + x_n t^{n-1}$

Data fitting

- Supongamos que tenemos m puntos (t_i, y_i). Queremos obtener una función que ajuste lo "mejor" posible esos datos.
- Para nosotros mejor ajuste va a significar minimizar la suma de los cuadrados de la diferencia entre los datos reales y los que nos da nuestra función modelo, de ahí el nombre del método.
- Lo que queremos entonces es encontrar una función f tal que garantice:

$$\min_f \sum_{i=1}^{m} (y_i - f(t_i))^2 \tag{1}$$

- Para modelar f vamos a usar n parámetros, es decir que encontrar nuestra función modelo es encontrar el vector $x \in \mathbb{R}^n$ de parámetros que minimice (1).
- Nosotros vamos a trabajar con funciones que sean lineales en los componentes del vector de parámetros x, en particular polinomios: $f(t) = f(t, x) = x_1 + x_2 t + x_3 t^2 + ... + x_n t^{n-1}$

Pasando a forma matricial

Supongamos que queremos aproximar m puntos (t_i, y_i) por un polinomio de grado n. Lo podemos representar matricialmente como:

$$Ax = \begin{pmatrix} 1 & t_1 & t_1^2 & \dots & t_1^{n-1} \\ 1 & t_2 & t_2^2 & \dots & t_2^{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & t_m & t_m^2 & \dots & t_m^{n-1} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} \approx \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix} = b \quad (2)$$

Como podemos resolver este sistema? Seguramente ni tenga solución exacta porque tenemos más ecuaciones que incógnitas.

Pasando a forma matricial

Supongamos que queremos aproximar m puntos (t_i, y_i) por un polinomio de grado n. Lo podemos representar matricialmente como:

$$Ax = \begin{pmatrix} 1 & t_1 & t_1^2 & \dots & t_1^{n-1} \\ 1 & t_2 & t_2^2 & \dots & t_2^{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & t_m & t_m^2 & \dots & t_m^{n-1} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} \approx \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix} = b \quad (2)$$

Como podemos resolver este sistema? Seguramente ni tenga solución exacta porque tenemos más ecuaciones que incógnitas.

Pasando a forma matricial

Supongamos que queremos aproximar m puntos (t_i, y_i) por un polinomio de grado n. Lo podemos representar matricialmente como:

$$Ax = \begin{pmatrix} 1 & t_1 & t_1^2 & \dots & t_1^{n-1} \\ 1 & t_2 & t_2^2 & \dots & t_2^{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & t_m & t_m^2 & \dots & t_m^{n-1} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} \approx \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix} = b \quad (2)$$

Como podemos resolver este sistema? Seguramente ni tenga solución exacta porque tenemos más ecuaciones que incógnitas.

Intuición

Lo que estamos buscando nosotros no es una solución exacta sino minimizar la norma cuadrada del vector residual $r = b - Ax$.

Quién es el vector x que minimiza la norma de r? Qué vector x hace que Ax sea lo más parecido posible a b?

Intuición

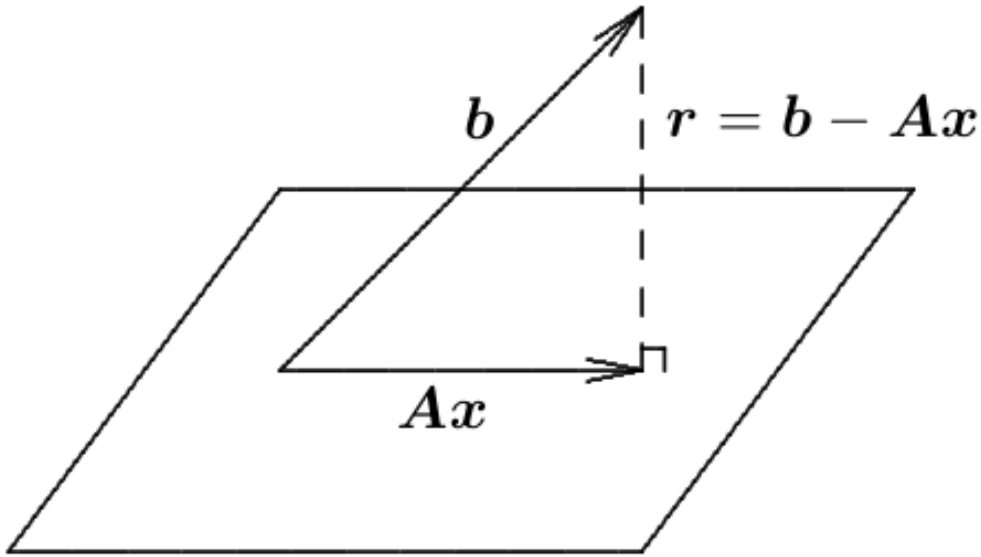
Lo que estamos buscando nosotros no es una solución exacta sino minimizar la norma cuadrada del vector residual $r = b - Ax$.

Quién es el vector x que minimiza la norma de r? Qué vector x hace que Ax sea lo más parecido posible a b?

Intuición

Lo que estamos buscando nosotros no es una solución exacta sino minimizar la norma cuadrada del vector residual $r = b - Ax$.

Quién es el vector x que minimiza la norma de r? Qué vector x hace que Ax sea lo más parecido posible a b?



Lo que estamos buscando es la proyección ortogonal de b en la imagen de A.

Ecuaciones normales

En base a lo anterior se puede ver que la solución a nuestro problema de cuadrados mínimos está dada por:

$$A^T Ax = A^T b \tag{3}$$

Además si A tiene rango columna completo:

- $A^T A$ es no singular. De esta manera resolviendo (3) obtenemos la solución al problema de cuadrados mínimos.
- $A^T A$ es simétrica y definida positiva, por lo que podemos resolver por Cholesky.

Ecuaciones normales

En base a lo anterior se puede ver que la solución a nuestro problema de cuadrados mínimos está dada por:

$$A^T A x = A^T b \tag{3}$$

Además si A tiene rango columna completo:

- $A^T A$ es no singular. De esta manera resolviendo (3) obtenemos la solución al problema de cuadrados mínimos.
- $A^T A$ es simétrica y definida positiva, por lo que podemos resolver por Cholesky.

Ecuaciones normales

En base a lo anterior se puede ver que la solución a nuestro problema de cuadrados mínimos está dada por:

$$A^T Ax = A^T b \tag{3}$$

Además si A tiene rango columna completo:

- $A^T A$ es no singular. De esta manera resolviendo (3) obtenemos la solución al problema de cuadrados mínimos.
- $A^T A$ es simétrica y definida positiva, por lo que podemos resolver por Cholesky.

Ecuaciones normales

Ventajas de resolver CM usando ecuaciones normales:

- Si $m \gg n$ (cosa muy común), resolvemos un sistema en $n \times n$
- Fácil de implementar, calcular $A^T A$ y resolver por Cholesky

Sin embargo, al resolver de esta manera nos podríamos encontrar con algunos problemas:

- Perdida de precisión al calcular $A^T A$ y $A^T b$
- $cond(A^T A) = cond(A)^2$

Que otra opción tenemos para evitar estos problemas?

Ecuaciones normales

Ventajas de resolver CM usando ecuaciones normales:

- Si $m \gg n$ (cosa muy común), resolvemos un sistema en $n \times n$
- Fácil de implementar, calcular $A^T A$ y resolver por Cholesky

Sin embargo, al resolver de esta manera nos podríamos encontrar con algunos problemas:

- Perdida de precisión al calcular $A^T A$ y $A^T b$
- $cond(A^T A) = cond(A)^2$

Que otra opción tenemos para evitar estos problemas?

Ecuaciones normales

Ventajas de resolver CM usando ecuaciones normales:

- Si $m \gg n$ (cosa muy común), resolvemos un sistema en $n \times n$
- Fácil de implementar, calcular $A^T A$ y resolver por Cholesky

Sin embargo, al resolver de esta manera nos podríamos encontrar con algunos problemas:

- Perdida de precisión al calcular $A^T A$ y $A^T b$
- $cond(A^T A) = cond(A)^2$

Que otra opción tenemos para evitar estos problemas?

Resolución usando QR

- Si factorizamos A con QR tenemos que:

$$A = Q \begin{bmatrix} R \\ 0 \end{bmatrix} \text{ con } R \in \mathbb{R}^{n \times n}$$

- Además, como Q es ortogonal sabemos que mantiene la norma y entonces

$$\|r\|_2 = \left\|Q^T r\right\|_2 = \left\|Q^T(b - Ax)\right\|_2 = \left\|Q^T b - \begin{bmatrix} R \\ 0 \end{bmatrix} x\right\|_2$$

- Si llamamos $Q^T b = b' = \begin{bmatrix} b'_1 \\ b'_2 \end{bmatrix}$, con $b'_1 \in \mathbb{R}^n$
- Nuestro problema de cuadrados mínimos se reduce a minimizar $\|r\|_2^2 = \|b'_1 - Rx\|_2^2 + \|b'_2\|_2^2$
- Luego la solución del problema de cuadrados mínimos es $Rx = b'_1$ y el error de nuestra aproximación $\|r\|_2^2 = \|b'_2\|_2^2$

Resolución usando QR

- Si factorizamos A con QR tenemos que:

$$A = Q \begin{bmatrix} R \\ 0 \end{bmatrix} \text{ con } R \in \mathbb{R}^{n \times n}$$

- Además, como Q es ortogonal sabemos que mantiene la norma y entonces

$$\|r\|_2 = \|Q^T r\|_2 = \|Q^T(b - Ax)\|_2 = \left\| Q^T b - \begin{bmatrix} R \\ 0 \end{bmatrix} x \right\|_2$$

- Si llamamos $Q^T b = b' = \begin{bmatrix} b'_1 \\ b'_2 \end{bmatrix}$, con $b'_1 \in \mathbb{R}^n$
- Nuestro problema de cuadrados mínimos se reduce a minimizar $\|r\|_2^2 = \|b'_1 - Rx\|_2^2 + \|b'_2\|_2^2$
- Luego la solución del problema de cuadrados mínimos es $Rx = b'_1$ y el error de nuestra aproximación $\|r\|_2^2 = \|b'_2\|_2^2$

Resolución usando QR

- Si factorizamos A con QR tenemos que:

$$A = Q \begin{bmatrix} R \\ 0 \end{bmatrix} \text{ con } R \in \mathbb{R}^{n \times n}$$

- Además, como Q es ortogonal sabemos que mantiene la norma y entonces

$$\|r\|_2 = \left\|Q^T r\right\|_2 = \left\|Q^T (b - Ax)\right\|_2 = \left\| Q^T b - \begin{bmatrix} R \\ 0 \end{bmatrix} x \right\|_2$$

- Si llamamos $Q^T b = b' = \begin{bmatrix} b'_1 \\ b'_2 \end{bmatrix}$, con $b'_1 \in \mathbb{R}^n$
- Nuestro problema de cuadrados mínimos se reduce a minimizar $\|r\|_2^2 = \|b'_1 - Rx\|_2^2 + \|b'_2\|_2^2$
- Luego la solución del problema de cuadrados mínimos es $Rx = b'_1$ y el error de nuestra aproximación $\|r\|_2^2 = \|b'_2\|_2^2$

Resolución usando QR

- Si factorizamos A con QR tenemos que:
$$A = Q \begin{bmatrix} R \\ 0 \end{bmatrix} \text{ con } R \in \mathbb{R}^{n \times n}$$
- Además, como Q es ortogonal sabemos que mantiene la norma y entonces
$$\|r\|_2 = \left\|Q^T r\right\|_2 = \left\|Q^T(b - Ax)\right\|_2 = \left\|Q^T b - \begin{bmatrix} R \\ 0 \end{bmatrix} x\right\|_2$$
- Si llamamos $Q^T b = b' = \begin{bmatrix} b'_1 \\ b'_2 \end{bmatrix}$, con $b'_1 \in \mathbb{R}^n$
- Nuestro problema de cuadrados mínimos se reduce a minimizar $\|r\|_2^2 = \|b'_1 - Rx\|_2^2 + \|b'_2\|_2^2$
- Luego la solución del problema de cuadrados mínimos es $Rx = b'_1$ y el error de nuestra aproximación $\|r\|_2^2 = \|b'_2\|_2^2$

Resolución usando QR

- Si factorizamos A con QR tenemos que:
$$A = Q \begin{bmatrix} R \\ 0 \end{bmatrix} \text{ con } R \in \mathbb{R}^{n \times n}$$
- Además, como Q es ortogonal sabemos que mantiene la norma y entonces
$$\|r\|_2 = \left\|Q^T r\right\|_2 = \left\|Q^T (b - Ax)\right\|_2 = \left\| Q^T b - \begin{bmatrix} R \\ 0 \end{bmatrix} x \right\|_2$$
- Si llamamos $Q^T b = b' = \begin{bmatrix} b'_1 \\ b'_2 \end{bmatrix}$, con $b'_1 \in \mathbb{R}^n$
- Nuestro problema de cuadrados mínimos se reduce a minimizar $\|r\|_2^2 = \|b'_1 - Rx\|_2^2 + \|b'_2\|_2^2$
- Luego la solución del problema de cuadrados mínimos es $Rx = b'_1$ y el error de nuestra aproximación $\|r\|_2^2 = \|b'_2\|_2^2$

Resolución usando QR

- Si factorizamos A con QR tenemos que:
$$A = Q \begin{bmatrix} R \\ 0 \end{bmatrix} \text{ con } R \in \mathbb{R}^{n \times n}$$
- Además, como Q es ortogonal sabemos que mantiene la norma y entonces
$$\|r\|_2 = \left\|Q^T r\right\|_2 = \left\|Q^T (b - Ax)\right\|_2 = \left\|Q^T b - \begin{bmatrix} R \\ 0 \end{bmatrix} x\right\|_2$$
- Si llamamos $Q^T b = b' = \begin{bmatrix} b'_1 \\ b'_2 \end{bmatrix}$, con $b'_1 \in \mathbb{R}^n$
- Nuestro problema de cuadrados mínimos se reduce a minimizar $\|r\|_2^2 = \|b'_1 - Rx\|_2^2 + \|b'_2\|_2^2$
- Luego la solución del problema de cuadrados mínimos es $Rx = b'_1$ y el error de nuestra aproximación $\|r\|_2^2 = \|b'_2\|_2^2$

Resolución usando QR

Ventajas de resolver CM usando QR:

- Evitamos calcular $A^T A$ y $A^T b$, evitando posibles errores numéricos.
- Error relativo de la solución: $cond(A) + \|r\|_2 \, cond(A)^2$

Por otro lado, si $m \gg n$, resolver por QR requiere aproximadamente el doble de trabajo computacional (si lo resolvemos de manera inteligente).

En general, la elección del método de resolución a utilizar dependerá las características del problema particular que se quiera resolver.

Resolución usando QR

Ventajas de resolver CM usando QR:

- Evitamos calcular $A^T A$ y $A^T b$, evitando posibles errores numéricos.
- Error relativo de la solución: $cond(A) + \|r\|_2 \, cond(A)^2$

Por otro lado, si $m \gg n$, resolver por QR requiere aproximadamente el doble de trabajo computacional (si lo resolvemos de manera inteligente).

En general, la elección del método de resolución a utilizar dependerá las características del problema particular que se quiera resolver.

Resolución usando QR

Ventajas de resolver CM usando QR:

- Evitamos calcular $A^T A$ y $A^T b$, evitando posibles errores numéricos.
- Error relativo de la solución: $cond(A) + \|r\|_2 \, cond(A)^2$

Por otro lado, si $m \gg n$, resolver por QR requiere aproximadamente el doble de trabajo computacional (si lo resolvemos de manera inteligente).

En general, la elección del método de resolución a utilizar dependera las características del problema particular que se quiera resolver.

Veamos una aplicación real

La tomografía computada es una tecnología que utiliza rayos X para producir una imagen de un corte o area específica del objeto de estudio, permitiendonos "ver" lo que hay adentro.

Algunos datos

- Las imágenes por TC son de mucha exactitud, no son invasivas, no provocan dolor y se obtienen en tiempo real.
- Se generan imágenes de huesos, tejidos blandos y vasos sanguíneos al mismo tiempo.
- En comparación con una RMN, es menos sensible al movimiento de pacientes y la pueden realizar pacientes con implantes.

No todas son buenas

- Dependiendo del estudio y la tecnología utilizada, el paciente se expone a dosis de radiación no menores que podrían causar enfermedades severas como cancer, sobretodo en niños y adolescentes.
- En muchas TC, es necesario una inyección intravenosa de material de contraste que podría causar reacciones alérgicas.

Cómo funciona

El aparato de TC emite un haz de rayos X que incide sobre el objeto a estudiar. La radiación que no haya sido absorbida por el objeto es captada por los detectores que se encuentran del otro lado.

El aparato va rotando de manera que el tubo emisor de rayos X emita en distintos ángulos para poder obtener una imágen confiable del corte.

Si se necesitan más cortes, la mesa donde reposa el objeto avanza o retrocede y el proceso se repite.

Nuestro experimento

- Vamos a experimentar simulando una tomografía computada de un corte del craneo de un paciente.
- Nuestro "cerebro" a estudiar será la imagen de una tomografía ya realizada, con el objetivo de poder reconstruirla lo mejor posible, sabiendo que en la realidad no contaríamos con dicha imagen.
- Vamos a simular la emisión de rayos X, de manera que si un rayo atraviesa una zona densa de la imagen (las zonas más densas son las zonas blancas de la imagen) tardará más tiempo en atravesarla que si atraviesa una zona opaca.

Nuestro experimento

- Vamos a experimentar simulando una tomografía computada de un corte del craneo de un paciente.
- Nuestro "cerebro" a estudiar será la imagen de una tomografía ya realizada, con el objetivo de poder reconstruirla lo mejor posible, sabiendo que en la realidad no contaríamos con dicha imagen.
- Vamos a simular la emisión de rayos X, de manera que si un rayo atraviesa una zona densa de la imagen (las zonas más densas son las zonas blancas de la imagen) tardará más tiempo en atravesarla que si atraviesa una zona opaca.

Nuestro experimento

- Vamos a experimentar simulando una tomografía computada de un corte del craneo de un paciente.
- Nuestro "cerebro" a estudiar será la imagen de una tomografía ya realizada, con el objetivo de poder reconstruirla lo mejor posible, sabiendo que en la realidad no contaríamos con dicha imagen.
- Vamos a simular la emisión de rayos X, de manera que si un rayo atraviesa una zona densa de la imagen (las zonas más densas son las zonas blancas de la imagen) tardará más tiempo en atravesarla que si atraviesa una zona opaca.

Nuestro modelo

Para reconstruir la imagen, vamos a discretizar la misma en n filas y m columnas, y llamaremos d_{ij} a la densidad de la imagen reconstruida en la fila i y columna j.

Emitiremos un total de k rayos y para cada uno definiremos:

- t_r el tiempo que tarda el rayo r en atravesar la imagen. El tiempo estará dado por la suma del valor de cada pixel por el que pasa el rayo.
- p_{ij}^r la cantidad de pixeles que atraviesa el rayo r en la fila i y columna j de la discretización.

Intuitivamente, si un rayo atraviesa pixeles de ciertas regiones de la discretizacion y tarda mucho tiempo en atravesar la imagen, es porque dichas regiones son densas.

Nuestro modelo

Para reconstruir la imagen, vamos a discretizar la misma en n filas y m columnas, y llamaremos d_{ij} a la densidad de la imagen reconstruida en la fila i y columna j.

Emitiremos un total de k rayos y para cada uno definiremos:

- t_r el tiempo que tarda el rayo r en atravesar la imagen. El tiempo estará dado por la suma del valor de cada pixel por el que pasa el rayo.
- p_{ij}^r la cantidad de pixeles que atraviesa el rayo r en la fila i y columna j de la discretización.

Intuitivamente, si un rayo atraviesa pixeles de ciertas regiones de la discretizacion y tarda mucho tiempo en atravesar la imagen, es porque dichas regiones son densas.

Nuestro modelo

Para reconstruir la imagen, vamos a discretizar la misma en n filas y m columnas, y llamaremos d_{ij} a la densidad de la imagen reconstruida en la fila i y columna j.

Emitiremos un total de k rayos y para cada uno definiremos:

- t_r el tiempo que tarda el rayo r en atravesar la imagen. El tiempo estará dado por la suma del valor de cada pixel por el que pasa el rayo.
  - p_{ij}^r la cantidad de pixeles que atraviesa el rayo r en la fila i y columna j de la discretización.

Intuitivamente, si un rayo atraviesa pixeles de ciertas regiones de la discretizacion y tarda mucho tiempo en atravesar la imagen, es porque dichas regiones son densas.

Nuestro modelo

Para reconstruir la imagen, vamos a discretizar la misma en n filas y m columnas, y llamaremos d_{ij} a la densidad de la imagen reconstruida en la fila i y columna j.

Emitiremos un total de k rayos y para cada uno definiremos:

- t_r el tiempo que tarda el rayo r en atravesar la imagen. El tiempo estará dado por la suma del valor de cada pixel por el que pasa el rayo.
- p_{ij}^r la cantidad de pixeles que atraviesa el rayo r en la fila i y columna j de la discretización.

Intuitivamente, si un rayo atraviesa pixeles de ciertas regiones de la discretizacion y tarda mucho tiempo en atravesar la imagen, es porque dichas regiones son densas.

Nuestro modelo

Para reconstruir la imagen, vamos a discretizar la misma en n filas y m columnas, y llamaremos d_{ij} a la densidad de la imagen reconstruida en la fila i y columna j.

Emitiremos un total de k rayos y para cada uno definiremos:

- t_r el tiempo que tarda el rayo r en atravesar la imagen. El tiempo estará dado por la suma del valor de cada pixel por el que pasa el rayo.
- p_{ij}^r la cantidad de pixeles que atraviesa el rayo r en la fila i y columna j de la discretización.

Intuitivamente, si un rayo atraviesa pixeles de ciertas regiones de la discretizacion y tarda mucho tiempo en atravesar la imagen, es porque dichas regiones son densas.

Nuestro modelo

Usando las variables definidas anteriormente, nuestro sistema a resolver queda definido como:

$$\begin{pmatrix} p_{11}^1 & p_{12}^1 & \cdots & p_{21}^1 & \cdots & p_{nm}^1 \\ p_{11}^2 & p_{12}^2 & \cdots & p_{21}^2 & \cdots & p_{nm}^2 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ p_{11}^k & p_{12}^k & \cdots & p_{21}^k & \cdots & p_{nm}^k \end{pmatrix} \begin{pmatrix} d_{11} \\ d_{12} \\ \vdots \\ d_{21} \\ \vdots \\ d_{nm} \end{pmatrix} \approx \begin{pmatrix} t_1 \\ t_2 \\ \vdots \\ t_k \end{pmatrix} \tag{4}$$

Una vez que sepamos los valores de densidad d_{ij} podremos reconstruir nuestra imagen.

Probemos como funciona

Cosas a tener en cuenta a la hora de la experimentación:

- ¿Qué tamaño debe tener una buena discretización?
- ¿Cuántos rayos necesito emitir para que la aproximación sea buena?
- ¿Qué metodo de CM nos coviene usar? ¿Cuál aproxima mejor? ¿Cuánto tardan en correr?

Pasemos al enunciado.