Enfoque Bayesiano para Interferometría: Formalización

Autor: Lerko Araya Hernández
Profesores: Jorge Silva
Felipe Tobar
Axel Osses
Fecha: 14 de noviembre de 2016

Facultad de Ciencias Físicas y Matemáticas
Departamento de Ingeniería Eléctrica
Universidad de Chile

1. Antecedentes

La interferometría sienta sus bases en el Teorema de Van Cittert - Zernike, el cuál indica: *Si se denomina V a la función de visibilidades de un frente de ondas, I a su intensidad y A el área de recepción efectiva del instrumento utilizado para captar la señal, entonces:* [1]

$$V(u,v) = \int\int A(x,y)I(x,y)\exp\{-2\pi j(ux+vy)\}dxdy \tag{1.1}$$

Es decir, el teorema indica que, salvo una constante asociada al instrumento de medición, la función de visibilidades de una onda y su intensidad son un par transformada/antitransformada de Fourier. [2]
El problema radica en que debido a que limitaciones físicas de las antenas, no es posible obtener todas las mediciones del espectro de Fourier. Por esto, solo se cuenta con un set de mediciones.

2. Formalización

Sea una imagen en el dominio de los píxeles $I : \mathbb{N}\times\mathbb{N} \to \mathbb{Z}_{[0,255]}$. Se asume gaussianidad en este dominio, esto es, se tiene un modelo para I:

$$I^{mod}(x,y) = \sum_{i=1}^{N_B} \hat{\alpha}_i\hat{\phi}_i(x,y) \qquad \forall i = 1,...,N_B\ \hat{\alpha}_i \in \mathbb{R}; \hat{\phi}_i : \mathbb{R}\times\mathbb{R}\to\mathbb{R}$$

Donde $\hat{\phi}(x,y)$ es una función base, en este caso, al asumir gaussianidad, se tiene que $\hat{\phi}(x,y)$ tiene la siguiente forma de la ecuación 2.1.

$$\hat{\phi}_i(x,y) = \frac{1}{\sqrt{2\pi}l} exp\left\{-\frac{1}{2}\left(\frac{\left\|\begin{bmatrix}x\\y\end{bmatrix} - \begin{bmatrix}C_x^i\\C_y^i\end{bmatrix}\right\|}{l}\right)^2\right\} \tag{2.1}$$

Gracias al Teorema de Van Cittert - Zernike, las mediciones que se obtienen al utilizar la técnica de interferometría son la transformada de Fourier2D de la imagen. Como la transformada de Fourier es lineal, se tiene que la transformada de $I^{mod}(x,y)$ también es una suma de gaussianas, más específicamente la expresión está dada por la ecuación 2.3.

$$\begin{aligned}\hat{V}(u,v) &= \frac{1}{2\pi}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\sum_{i=1}^{N_B}\hat{\alpha}_i\hat{\phi}_i(x,y)exp\left\{-jux\right\}exp\left\{-jvy\right\}dxdy \\ &= \frac{1}{2\pi}\sum_{i=1}^{N_B}\hat{\alpha}_i\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\hat{\phi}_i(x,y)exp\left\{-jux\right\}exp\left\{-jvy\right\}dxdy \\ &= \sum_{i=1}^{N_B}\hat{\alpha}_i\frac{1}{2\pi 1/l}\exp\left\{-\frac{1}{2}\frac{(u^2+v^2)-j(C_x^iu+C_y^iv)}{1/l^2}\right\}\end{aligned} \tag{2.2}$$

En donde se usó norma L2. Luego, renombrando:

$$\alpha_i = \hat{\alpha}_i \exp\left\{ j\frac{uC_x^i}{1/l^2} \right\} \exp\left\{ j\frac{vC_y^i}{1/l^2} \right\} \qquad \wedge \qquad \phi(x,y) = \frac{1}{\sqrt{2\pi}1/l} \exp\left\{ -\frac{1}{2} \left(\frac{\left\| \begin{bmatrix} u \\ v \end{bmatrix} \right\|}{1/l} \right)^2 \right\} \tag{2.3}$$

Con lo anterior, y notando que las variables (u, v) son variables mudas, se tiene que el modelo de imagen en Fourier es:

$$\hat{V}(x,y) = \sum_{i=1}^{N_B} \alpha_i \phi(x,y) \qquad \forall i = 1, ..., N_B \ \alpha_i \in \mathbb{C}; \phi_i : \mathbb{R} \times \mathbb{R} \to \mathbb{R} \tag{2.4}$$

En este trabajo, se asume un modelo de ruido en las mediciones, es decir, se a la ecuación 2.4 se le debe agregar un ruido gaussiano complejo.

$$\boxed{V^{mod}(x,y) = \sum_{i=1}^{N_B} \alpha_i \phi(x,y) + \eta(0, M, K)} \tag{2.5}$$

En donde M es la covarianza del ruido y K es la pseudo-covarianza, las cuales están definidas por las ecuaciones 2.6. Cabe destacar que en este caso particular se asume media $\mu = 0$ para el ruido.

$$M = \mathbb{E}[(z-\mu)(z-\mu)^T] \qquad K = \mathbb{E}[(z-\mu)\overline{(z-\mu)}^T] \tag{2.6}$$

La función de densidad para esta v.a. está dada por:

$$f(z) = \frac{1}{\pi|\Sigma|} exp\left\{ -\frac{1}{2} \begin{bmatrix} z-\mu \\ \bar{z}-\bar{\mu} \end{bmatrix} \underbrace{\begin{bmatrix} M & K \\ \overline{K} & \overline{M} \end{bmatrix}^{-1}}_{\Sigma^{-1}} \begin{bmatrix} z-\mu \\ \bar{z}-\bar{\mu} \end{bmatrix}^T \right\} \tag{2.7}$$

2.1. Optimización

Para encontrar poder estimar las mediciones que no tenemos a partir de los que tenemos, se utilizará *máximum likelihood*. De este modo, sea $U = \{u_k, v_k\}_{k=1}^{N_{sample}}$ el conjunto de mediciones y sea $V = \{V_k\}_{k=1}^{N_{sample}}$ sus respectivas visibilidades, entonces la probabilidad a priori está dada por la expresión 2.8. Donde θ representa el modelo de parámetros $\{M, K, l, \alpha_i, C_x^i, C_y^i\}$.

$$\begin{aligned} p(V,U|\theta) &= \prod_{k=1}^{N_{sample}} p(V_k, u_k, v_k|\theta) \\ &= \prod_{i=1}^{N_{sample}} \frac{1}{\pi|\Sigma|} exp\left\{ -\frac{1}{2} \begin{bmatrix} V_k - \hat{V}(u_k,v_k) \\ \overline{V_k} - \hat{V}(u_k,v_k) \end{bmatrix} \begin{bmatrix} M & K \\ \overline{K} & \overline{M} \end{bmatrix}^{-1} \begin{bmatrix} V_k - \hat{V}(u_k,v_k) \\ \overline{V_k} - \hat{V}(u_k,v_k) \end{bmatrix}^T \right\} \end{aligned} \tag{2.8}$$

Luego, máximizar la función de *likelihood* es lo mismo que minimizar la *Negative Log Likelihood*, por ende, la función objetivo está dada por la ecuación 2.9.

$$NLL = \sum_{k=1}^{N_{sample}} log(\pi|\Sigma|) + \frac{1}{2} \begin{bmatrix} V_k - \sum_{i=1}^{N_B} \alpha_i \phi_i(x,y) \\ \overline{V_k} - \sum_{i=1}^{N_B} \alpha_i \phi_i(x,y) \end{bmatrix} \begin{bmatrix} M & K \\ \overline{K} & \overline{M} \end{bmatrix}^{-1} \begin{bmatrix} V_k - \sum_{i=1}^{N_B} \alpha_i \phi_i(x,y) \\ \overline{V_k} - \sum_{i=1}^{N_B} \alpha_i \phi_i(x,y) \end{bmatrix}^T \tag{2.9}$$

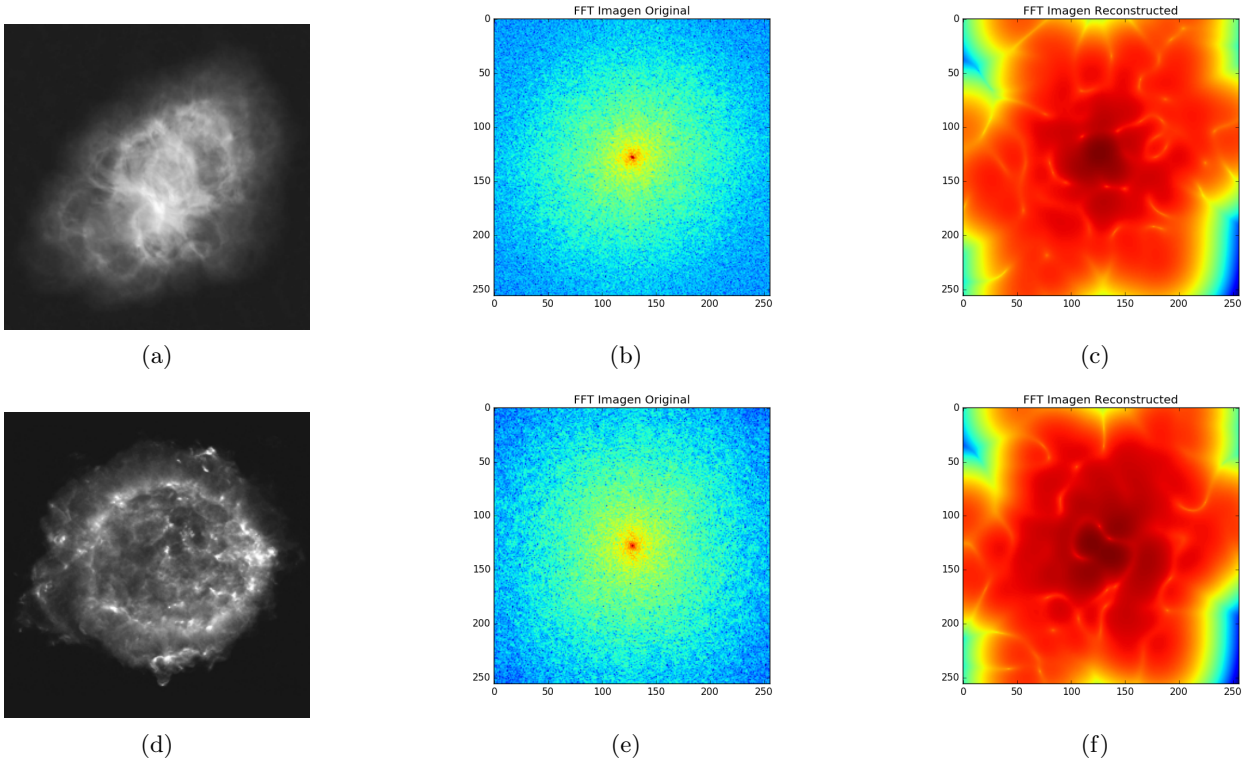
Finalmente, se debe plantear el problema de optimización:

$$\min_{M,K,l,\alpha_i,C_x^i,C_y^i} \quad NLL(M, K, l, \alpha_i, C_x^i, C_y^i)$$

3. Implementación

La implementación consta de la utilización de la librería `scipy.optimize`. Para esto primero se implementó la función `NLL(x,U,V)` la cual recibe los argumentos `x`, que es el vector de variables a optimizar, es decir, $\{M, K, l, \alpha_i, C_x^i, C_y^i\}$, también, recibe los conjuntos $\{U, V\}$ que son los conjuntos de mediciones.
Como la librería solo permite optimizar funciones de la forma $f(x)$, se implementó la función `auxNLL(x)`, la cual largo los datos de los conjuntos de mediciones $\{U, V\}$.

4. Resultados



(a) (b) (c)

(d) (e) (f)

Figura 1: En la primera columna: Dominio de los píxeles; en la segunda columna: Dominio de Fourier (Magnitud); en la tercera columna: Resultados de la implementación (Magnitud).

Los resultados expuestos en las figuras 1c y 1f distan bastante de la transformada de Fourier de las figuras 1b y 1e esto puede ser atribuible a que la formulación aún está con las funciones bases dependiendo de los centros. Por otro lado también se observa que las frecuencias cercanas al cero son las que tienen mayor potencia. Lo cual concuerda con el modelo, dado que la función gaussiana de la ecuación 2.4 está centrada en cero.

5. Análisis y Discusión

5.1. Modelo

La realizar el desglose matemático del modelo propuesto, se observa que es sutílmente distinto a lo discutido en las reuniones. Esto puesto que al aplicar transformada de Fourier, la función base (al menos considerando gaussiana), no depende de los centros, si no que los centros son absorbidos por el ponderador α_i, además, el parámetro α_i comienza a depender de la posición (u, v). [1]

5.2. Implementación

Respecto a la implementación, el principal problema encontrado es a la hora de optimizar. Como se dijo anteriormente, se utilizó la librería: `scipy.optimize`. La hipótesis respecto a el por qué los resultados no son buenos, es que esta

[1] En ese sentido, me gustaría que discutamos un poco y veamos como proseguir.

librería utiliza las funciones como caja negra, por ende, para si queremos darle el gradiente a la función debía calcularlo a mano, lo cual es complicado dado que son casi 1000 (varianzas, α_i, centros) variables de optimización. A pesar de esto, la librería es capaz de estimar el gradiente numéricamente. El problema de esto es que hace que el tiempo de ejecución aumente exponencialmente y la estimación puede acarrear errores numéricos.
Considerando lo anterior, se decidió utilizar librerías que me permitieran utilizar álgebra simbólica, de manera que el computador pueda calcular el gradiente automáticamente. De este modo, se llegó a `theano` y `tensorflow`.
Se descartó `theano` por que no permite trabajar con números complejos. `TensorFlow` tiene soporte para números complejos, pero solo en su versión escalar, es decir, no hay soporte para matrices complejas. A pesar de esto, se realizó una implementación con `TensorFlow`, pero se ha hecho muy complicado invertir matrices y hacer asignaciones a números complejos en matrices.
En resumen, se ha hecho muy difícil implementar la optimización, principalmente por que el dominio es complejo. A lo cual se propone abordar el problema en $\mathbb{R}^2$, es decir, en vez de ver la imagen como una función con codominio complejo, verla con codominio $\mathbb{R}^2$ (ver anexo 7.1). Otro factor, es lo complicado que me ha sido calcular el gradiente.

6. Referencias

[1] J. Moran, A. Thompson, and G. S. Jr, *Interferometry and Synthesis in Radio Astronomy*, 2nd ed. Weinheim: Wiley-VCH, 2001.

[2] G. Liberona, "Aplicaciones del procesamiento de imágenes digitales a astronomía," Bachelor's thesis, Universidad de Chile, Santiago, 2015.

7. Anexos

7.1. Formalización en $\mathbb{R}^2$

Asumiendo que la función de visibilidad $V(x, y)$ vive en $\mathbb{R}^2$, el ruido debe ser multivariable.

$$\boxed{V^{mod}(x,y) = \sum_{i=1}^{N_B} \alpha_i \phi(x,y) + \eta(0,K)} \tag{7.1}$$

En donde K es la matriz de covarianza del ruido. La función de densidad para esta v.a. está dada por:

$$f(z) = \frac{1}{2\pi\sqrt{|K|}} exp\left\{ -\frac{1}{2} \begin{bmatrix} z_1-\mu_1 \\ z_2-\mu_2 \end{bmatrix}^T \underbrace{\begin{bmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{bmatrix}^{-1}}_{K^{-1}} \begin{bmatrix} z_1-\mu_1 \\ z_2-\mu_2 \end{bmatrix} \right\} \tag{7.2}$$

7.2. Optimización

Para encontrar poder estimar las mediciones que no tenemos a partir de los que tenemos, se utilizará *máximum likelihood*. De este modo, sea $U = \{u_k, v_k\}_{k=1}^{N_{sample}}$ el conjunto de mediciones y sea $V = \{V_k\}_{k=1}^{N_{sample}}$ sus respectivas visibilidades, entonces la probabilidad a priori está dada por la expresión 7.3. Donde θ representa el modelo de parámetros $\{K, l, \alpha_i, C_x^i, C_y^i\}$ [2].

$$\begin{aligned} p(V,U|\theta) &= \prod_{k=1}^{N_{sample}} p(V_k, u_k, v_k|\theta) \\ &= \prod_{i=1}^{N_{sample}} \frac{1}{2\pi\sqrt{|K|}} exp\left\{ -\frac{1}{2} \begin{bmatrix} V_k^1 - \hat{V}^1(u_k,v_k) \\ V_k^2 - \hat{V}^2(u_k,v_k) \end{bmatrix}^T \begin{bmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{bmatrix}^{-1} \begin{bmatrix} V_k^1 - \hat{V}^1(u_k,v_k) \\ V_k^2 - \hat{V}^2(u_k,v_k) \end{bmatrix} \right\} \end{aligned} \tag{7.3}$$

Luego, maximizar la función de *likelihood* es lo mismo que minimizar la *Negative Log Likelihood*, por ende, la función objetivo está dada por la ecuación 7.4.

$$NLL = \sum_{k=1}^{N_{sample}} log\left(2\pi\sqrt{|K|}\right) + \frac{1}{2} \begin{bmatrix} V_k^1 - \hat{V}^1(u_k,v_k) \\ V_k^2 - \hat{V}^2(u_k,v_k) \end{bmatrix}^T \begin{bmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{bmatrix}^{-1} \begin{bmatrix} V_k^1 - \hat{V}^1(u_k,v_k) \\ V_k^2 - \hat{V}^2(u_k,v_k) \end{bmatrix} \tag{7.4}$$

Finalmente, se debe plantear el problema de optimización:

$$\min_{K,l,\alpha_i,C_x^i,C_y^i} \quad NLL(K, l, \alpha_i, C_x^i, C_y^i)$$

[2] Algo que me gustaría conversar es si, dado que estamos optimizando α ¿Es necesario encontrar los centro?