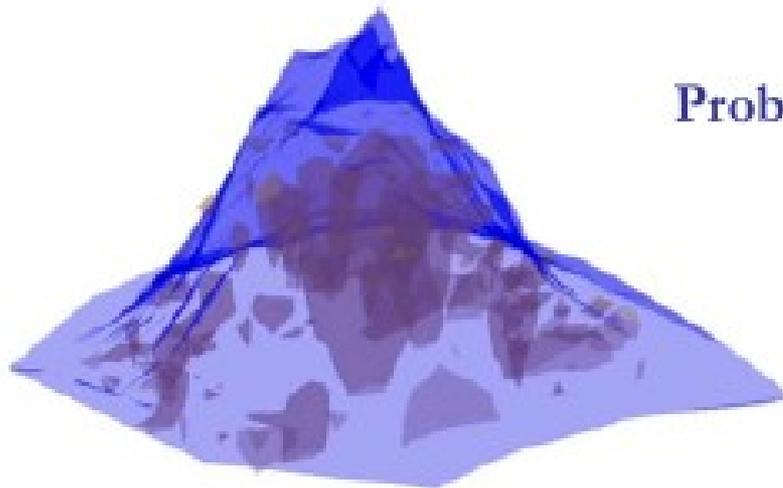
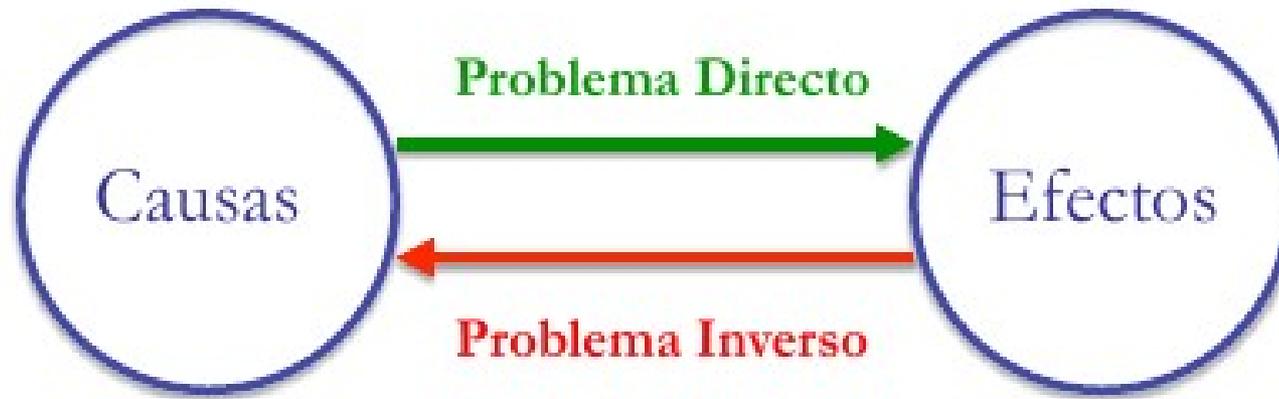


# Problemas Inversos en Procesamiento Digital de Imágenes

Carlos Milovic F. <sup>1,2,3</sup>

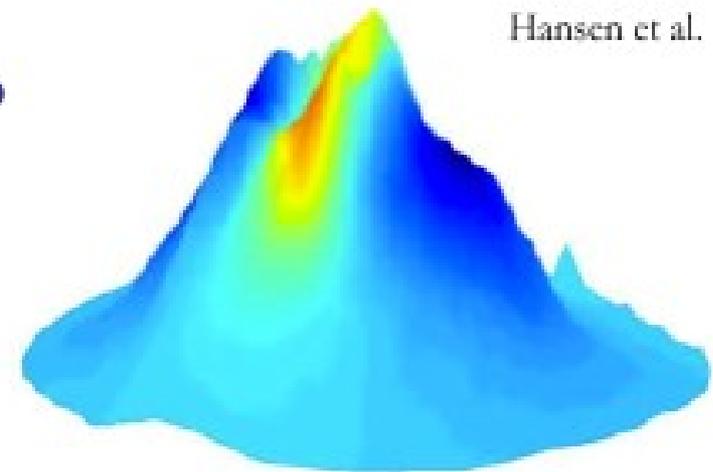
1. PixInsight Project Developer, Pleiades Astrophoto S.L., España
2. Centro de Imágenes Biomédicas - Pontificia Universidad Católica de Chile
3. Departamento de Ingeniería Eléctrica - PUC

*Octubre de 2014*



Reconstrucción al interior del volcán

Problema Inverso



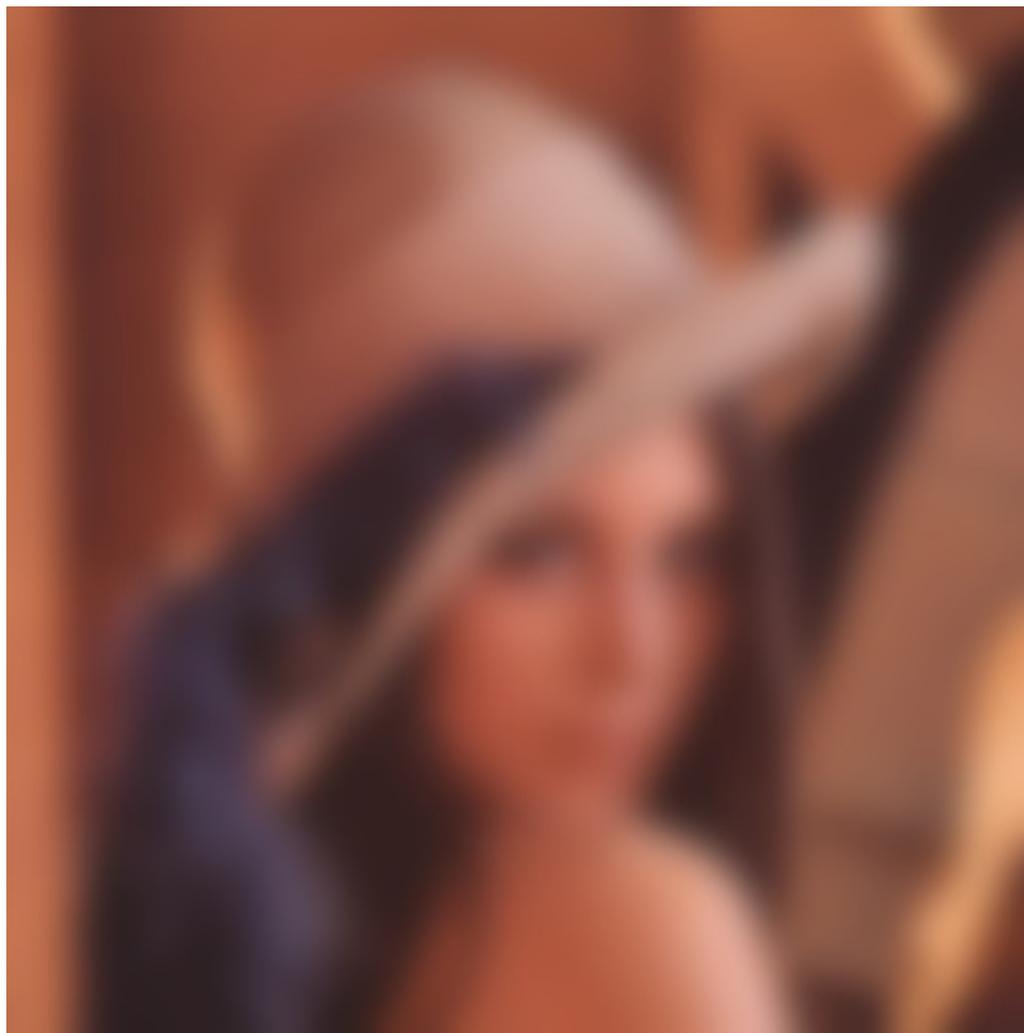
Hansen et al.

Medidas de la superficie del volcán

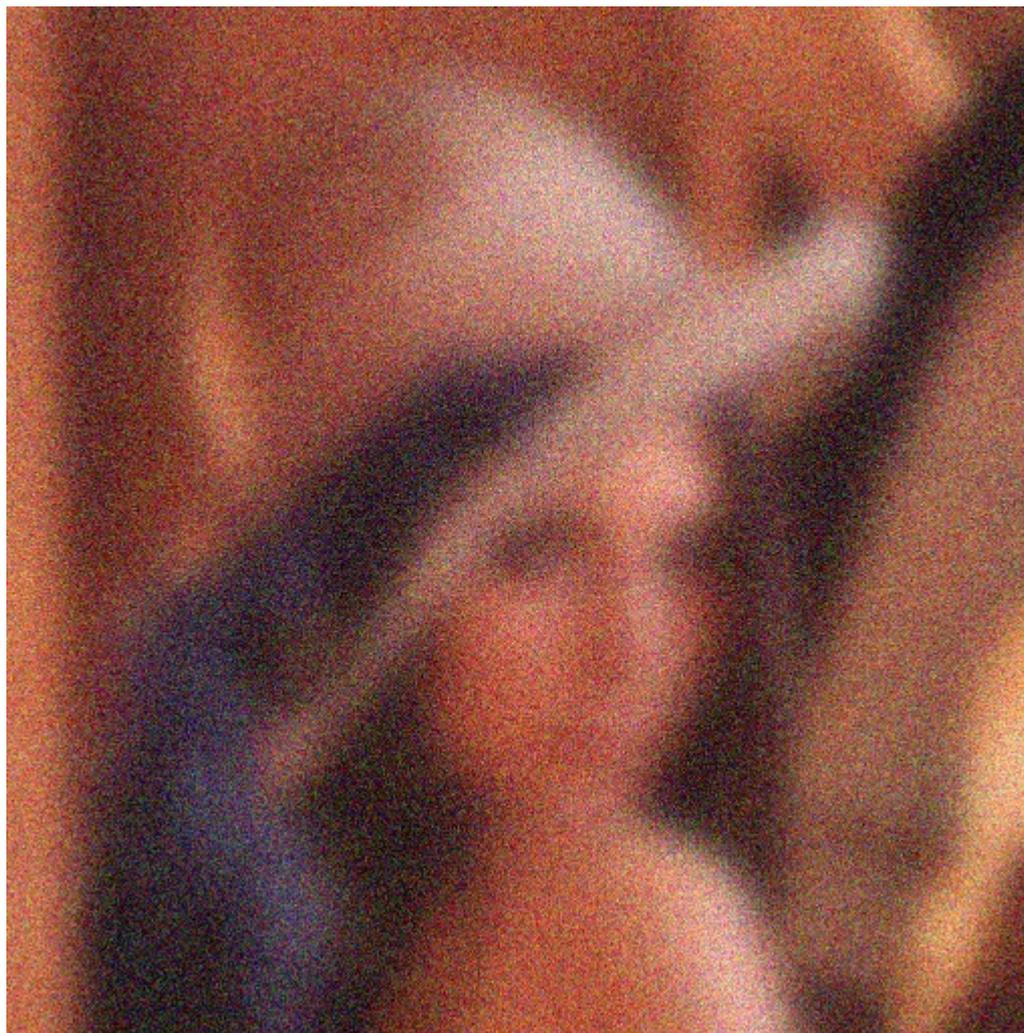
# Expectativas



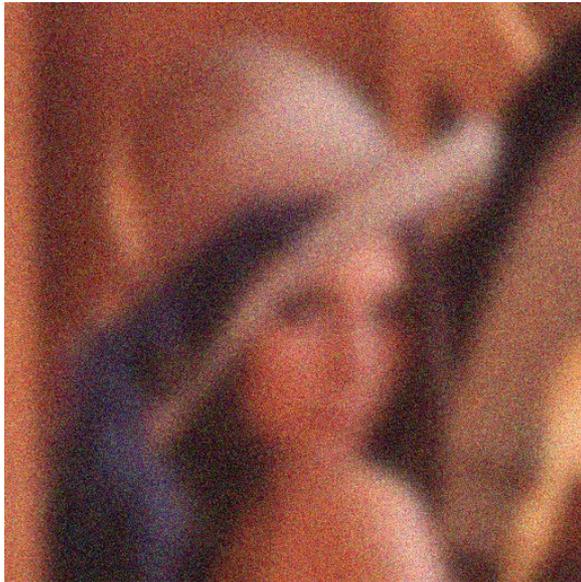
# Realidad



# Realidad



# ¿Podemos resolver el problema inverso?

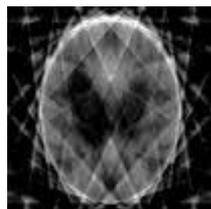
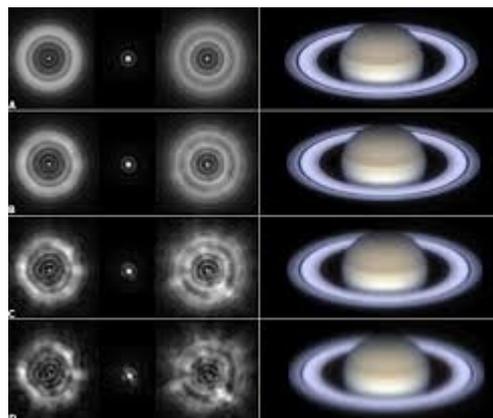
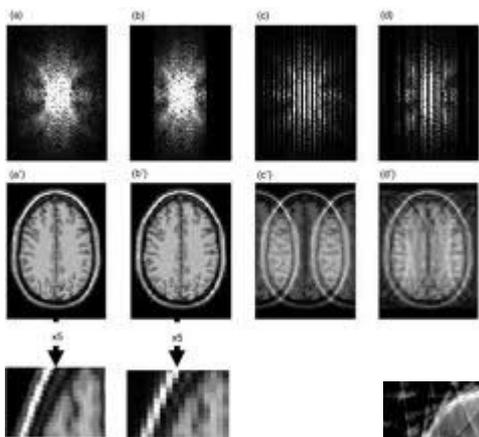


Realidad / Mediciones



Expectativas / Ideal

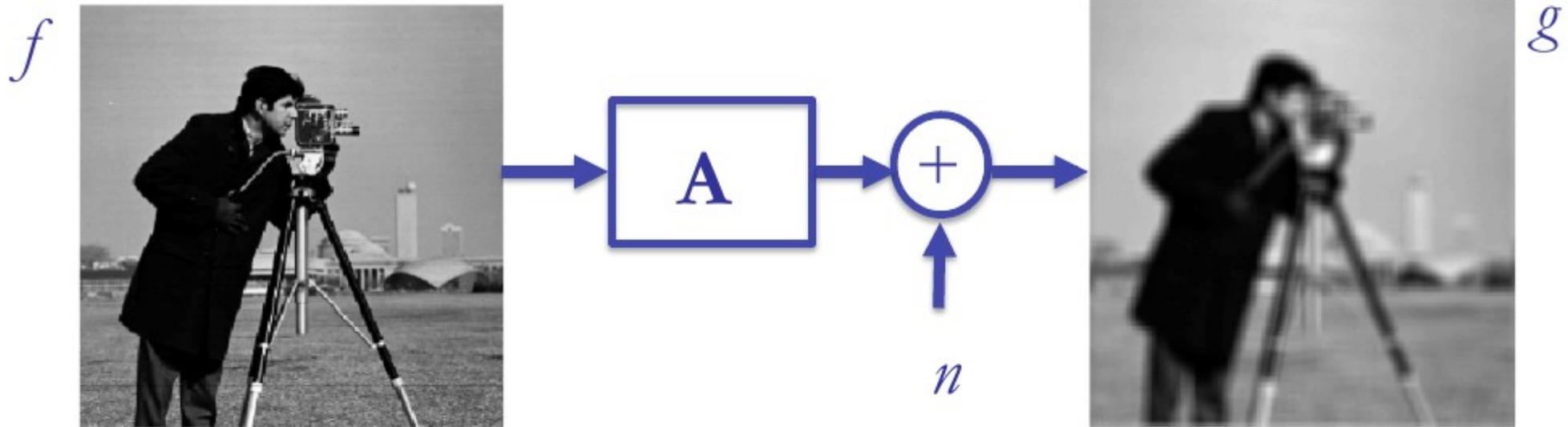
# Degradaciones



# Ruido



# Modelo de Imagen



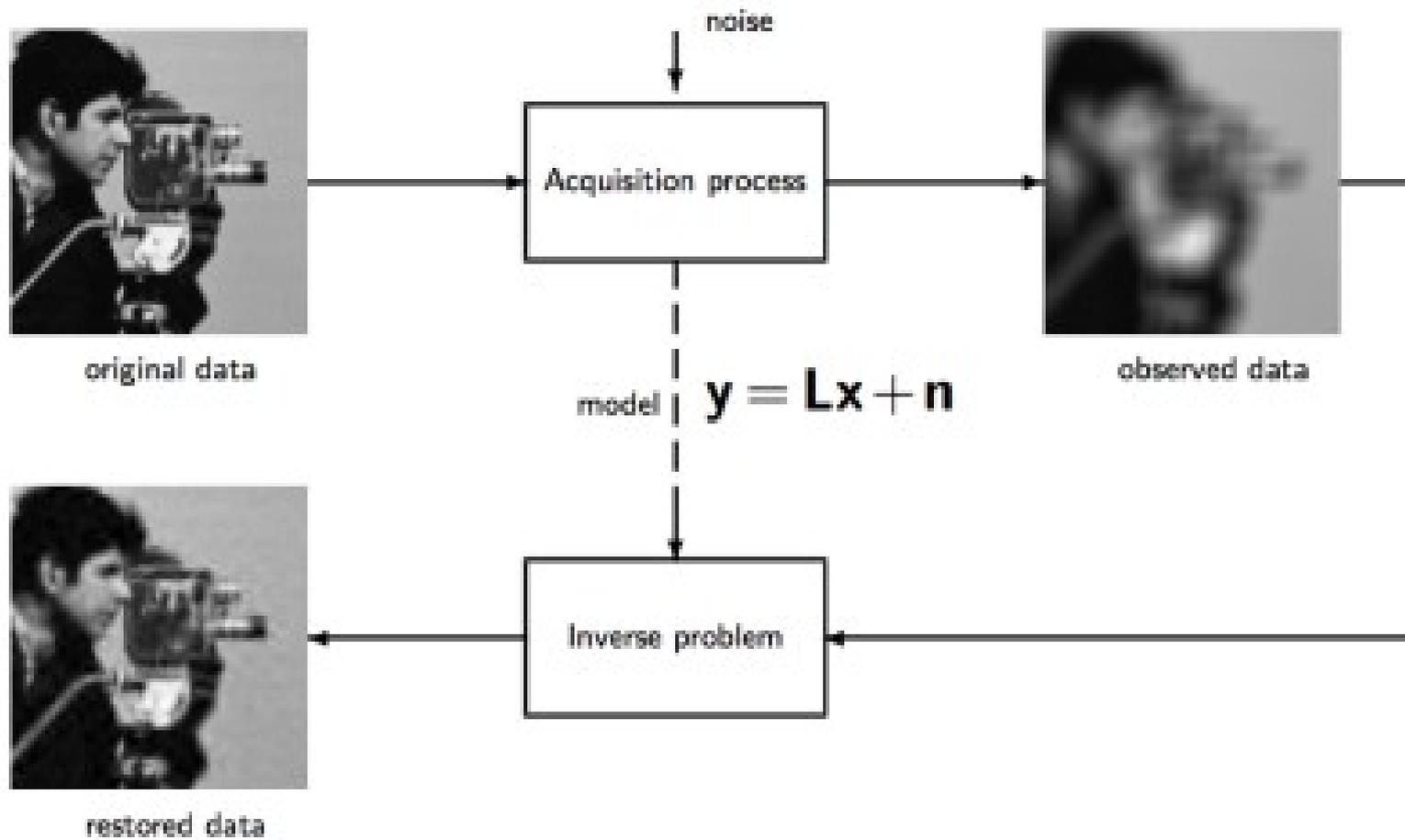
$A$ : operador lineal

$$g = Af + n$$

datos    imagen    ruido

**Meta** (problema inverso): encontrar  $f$  a partir de  $g$ .

# Problema Inverso



## Modelo de Imagen

- Si  $A$  es isotrópico, puede modelarse el problema como una convolución.
  - Problema inverso es una *deconvolución*.
  - $A = h$ , la respuesta al impulso (PSF).
- En el espacio de frecuencia (Fourier), las convoluciones son multiplicaciones punto a punto de matrices.

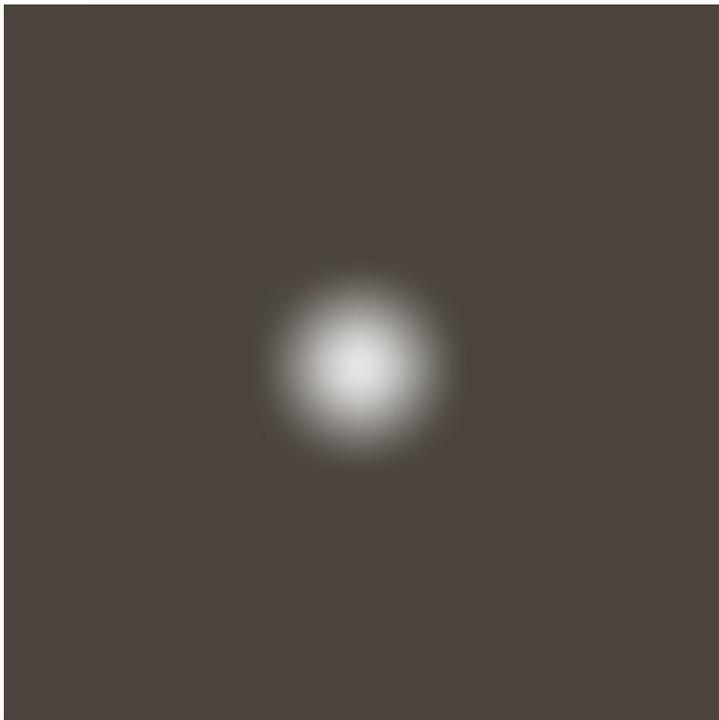
$$FFT \{ g(t) * h(t) \} = G(f) H(f)$$

# Filtro Inverso



$$\hat{F}(u, v) = \frac{G(u, v)}{H(u, v)}$$

# Amplificación de ruido



Ejemplo: modelo de Hufnagel y Stanley [1964]

$$H(u, v) = e^{-k(u^2 + v^2)^{5/6}}$$

Útil para modelos de degradación por turbulencia atmosférica.

Es muy similar a un filtro gaussiano pasabajos

$$\hat{F}(u, v) = F(u, v) + \frac{N(u, v)}{H(u, v)}$$

# Recortar acción de H



$f_{\text{corte}} = 40$

# Recortar acción de H



$f_{\text{corte}} = 70$

## Recortar acción de H

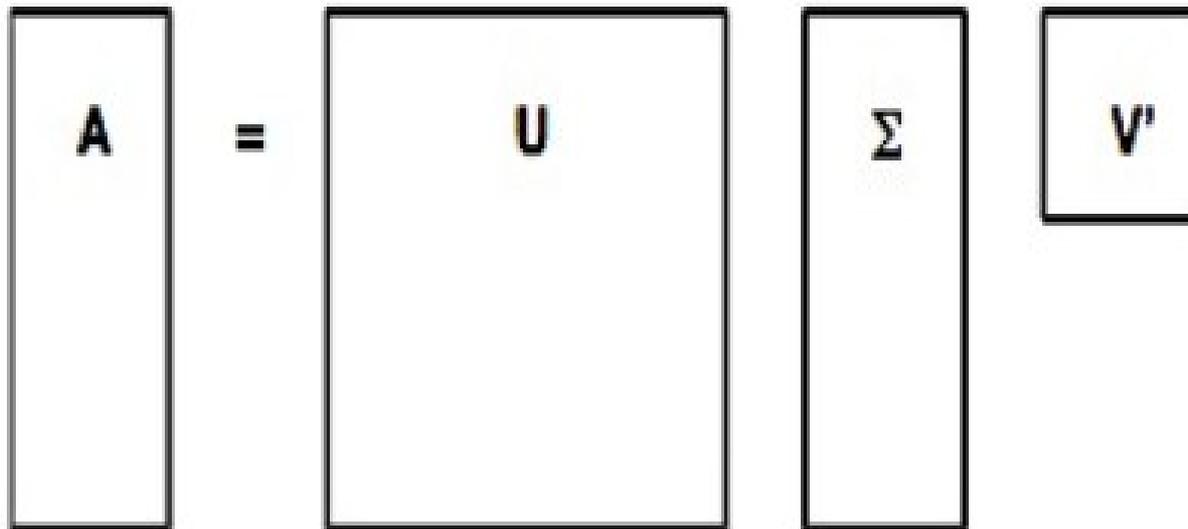


f\_corte = 85

# Otra alternativa: SVD o Valores propios

$$A = USV^T = \sum_{i=1}^n u_i \sigma_i v_i^T$$

$$S = \text{diag}(\sigma_1, \dots, \sigma_n) \quad \sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_n \geq 0$$



# SVD

- Sea la matriz  $A$ :

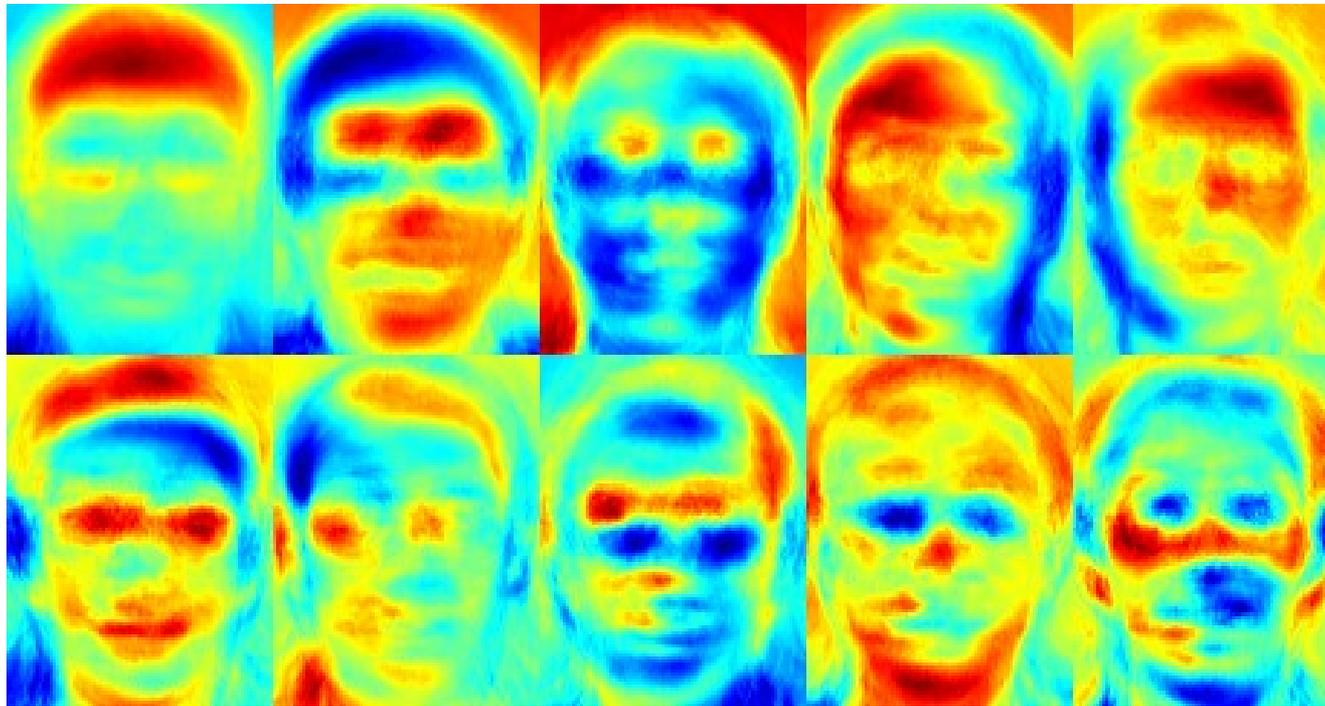
$$A = \begin{pmatrix} 1 & 0 & 3 \\ 0 & 2 & 4 \\ 3 & 4 & 1 \end{pmatrix}$$

- Su descomposición en valores singulares viene dada por:

$$U = \begin{pmatrix} -0.38 & -0.46 & 0.80 \\ -0.62 & -0.50 & -0.59 \\ -0.68 & 0.72 & 0.09 \end{pmatrix} \quad S = \begin{pmatrix} 6.35 & 0 & 0 \\ 0 & 3.71 & 0 \\ 0 & 0 & 1.35 \end{pmatrix} \quad V = \begin{pmatrix} -0.38 & 0.46 & 0.80 \\ -0.62 & 0.50 & -0.59 \\ -0.68 & -0.72 & 0.09 \end{pmatrix}$$

$$A = USV^T$$

# Ejemplo SVD en reconocimiento de caras.



# Ejemplo SVD en reconocimiento de caras.



## SVD

$$\hat{f} = \operatorname{argmin} \|g - Af\|_2 \quad \longrightarrow \quad \hat{f} = (A^T A)^{-1} A^T g$$

$$A^T A = \left( \sum_{k=1}^r \sigma_k v_k u_k^T \right) \left( \sum_{l=1}^r \sigma_l v_l u_l^T \right) = \sum_{k=1}^r \sum_{l=1}^r \sigma_k \sigma_l v_k (u_k^T u_l) v_l^T = \sum_{k=1}^r \sigma_k^2 v_k v_k^T$$

Si  $A^T A$  es invertible ( $n = r$ ):

$$(A^T A)^{-1} = \sum_{k=1}^r \frac{1}{\sigma_k^2} v_k v_k^T$$

Luego:

$$(A^T A)^{-1} A^T = \left( \sum_{k=1}^r \frac{1}{\sigma_k^2} v_k v_k^T \right) \left( \sum_{l=1}^r \sigma_l v_l u_l^T \right) = \sum_{k=1}^r \frac{1}{\sigma_k} v_k u_k^T$$

# SVD

Sustituyendo obtenemos:

$$\hat{f} = \sum_{k=1}^n \left( v_k^T f + \frac{u_k^T n}{\sigma_k} \right) v_k = f + \sum_{k=1}^n \frac{(u_k^T n)}{\sigma_k} v_k$$

Por lo tanto la reconstrucción corresponde a la suma de la verdadera imagen y términos debido al ruido.

- Los valores singulares decaen gradualmente a cero.
- Los vectores singulares presentan mas oscilaciones a medida que los valores singulares disminuyen.

**¡Podemos truncar la sumatoria!**

## Otras formas de estabilizar con SVD

$$\sum_{i=1}^n \phi_i^{[\lambda]} \frac{u_i^T g}{\sigma_i} v_i$$

Damped SVD:

$$\phi_i^{[\lambda]} = \frac{\sigma_i}{\sigma_i + \lambda}, \quad \lambda \geq 0$$

Filtro exponencial:

$$\phi_i^{[\lambda]} = 1 - \exp(-\beta \sigma_i^2), \quad \beta \geq 0$$

# Filtro de Wiener

Es equivalente a utilizar el espacio de Fourier como vectores propios y estabilizar la solución.



$$\left[ \frac{1}{H(u, v)} \frac{|H(u, v)|^2}{|H(u, v)|^2 + S_{\eta}(u, v)/S_f(u, v)} \right] G(u, v)$$

# Filtro de Wiener

$$\left[ \frac{1}{H(u, v)} \frac{|H(u, v)|^2}{|H(u, v)|^2 + S_\eta(u, v)/S_f(u, v)} \right] G(u, v)$$

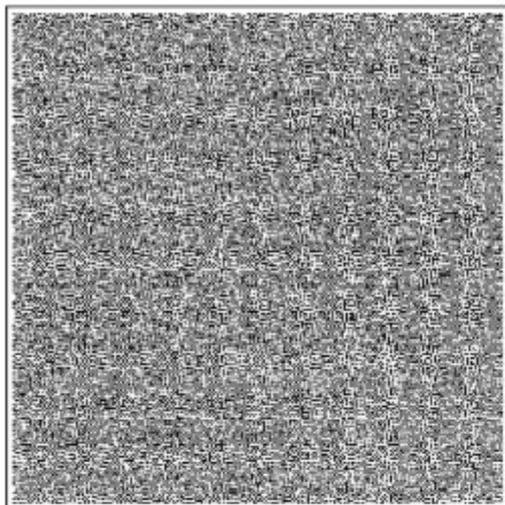
$S_\eta(u, v) = |N(u, v)|^2$  = espectro de potencia del ruido

$S_f(u, v) = |F(u, v)|^2$  = espectro de potencia de la imagen no degradada

$$SNR = \frac{\sum_{u=0}^{M-1} \sum_{v=0}^{N-1} |F(u, v)|^2}{\sum_{u=0}^{M-1} \sum_{v=0}^{N-1} |N(u, v)|^2}$$

$$MSE = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [f(x, y) - \hat{f}(x, y)]^2$$

# Filtro de Wiener



$\lambda=0.1$

The ill-conditioning of a problem does not mean that a meaningful approximate solution cannot be computed. Rather the ill-conditioning implies that standard methods in numerical linear algebra cannot be used in a straightforward way to compute such a solution. Instead, more sophisticated methods must be applied in order to ensure the computation of a meaningful solution.

This is the essential goal of regularization methods.

$\lambda=1$

The ill-conditioning of a problem does not mean that a meaningful approximate solution cannot be computed. Rather the ill-conditioning implies that standard methods in numerical linear algebra cannot be used in a straightforward way to compute such a solution. Instead, more sophisticated methods must be applied in order to ensure the computation of a meaningful solution.

This is the essential goal of regularization methods.

$\lambda=10$

The ill-conditioning of a problem does not mean that a meaningful approximate solution cannot be computed. Rather the ill-conditioning implies that standard methods in numerical linear algebra cannot be used in a straightforward way to compute such a solution. Instead, more sophisticated methods must be applied in order to ensure the computation of a meaningful solution.

This is the essential goal of regularization methods.

$\lambda=100$

The ill-conditioning of a problem does not mean that a meaningful approximate solution cannot be computed. Rather the ill-conditioning implies that standard methods in numerical linear algebra cannot be used in a straightforward way to compute such a solution. Instead, more sophisticated methods must be applied in order to ensure the computation of a meaningful solution.

This is the essential goal of regularization methods.

$\lambda=1000$

# Cambio paradigma: Optimización

$$\min_f \|Af - g\|_2$$

$$\hat{f} = (A^T A)^{-1} A^T g$$

# Cambio paradigma: Optimización

$$\min_f \|Af - g\|_2$$

$$\textit{sujeto a: } \Omega(f) < \epsilon$$

## Ejemplo: Tikhonov

$$\begin{aligned} \min \quad & \|Q\hat{f}\|^2 \\ \text{sujeto a} \quad & \|g - H\hat{f}\|^2 = \|n\|^2 \end{aligned}$$

$$\hat{f} = (H^T H + \gamma Q^T Q)^{-1} H^T g$$

Si  $Q = \text{Id}$ , entonces esto es el filtro de Wiener.

# Ejemplo: Constrained Least Squares

$$\begin{aligned} \min \quad & \|\nabla^2 f(x, y)\|^2 \\ \text{sujeto a} \quad & \|g - H\hat{f}\|^2 = \|n\|^2 \end{aligned}$$

$$\hat{F}(u, v) = \left[ \frac{H^*(u, v)}{|H(u, v)|^2 + \gamma|P(u, v)|^2} \right] G(u, v)$$

donde  $P(u, v)$  es la transformada de Fourier de

$$p(x, y) = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

# Ejemplo: Constrained Least Squares



Filtro de Wiener



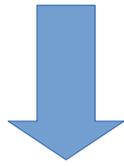
LS Restringido

# Cambio paradigma: Optimización

$$\min C(f) + \lambda \Omega(f)$$

# Cambio paradigma: Optimización

$$\min C(f) + \lambda \Omega(f)$$

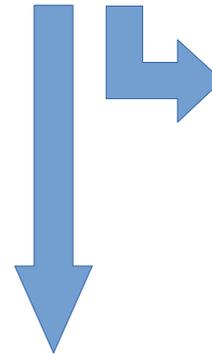


Consistencia de datos.

Ej. Norma 2 (RMS),

Suma error absoluto, etc.

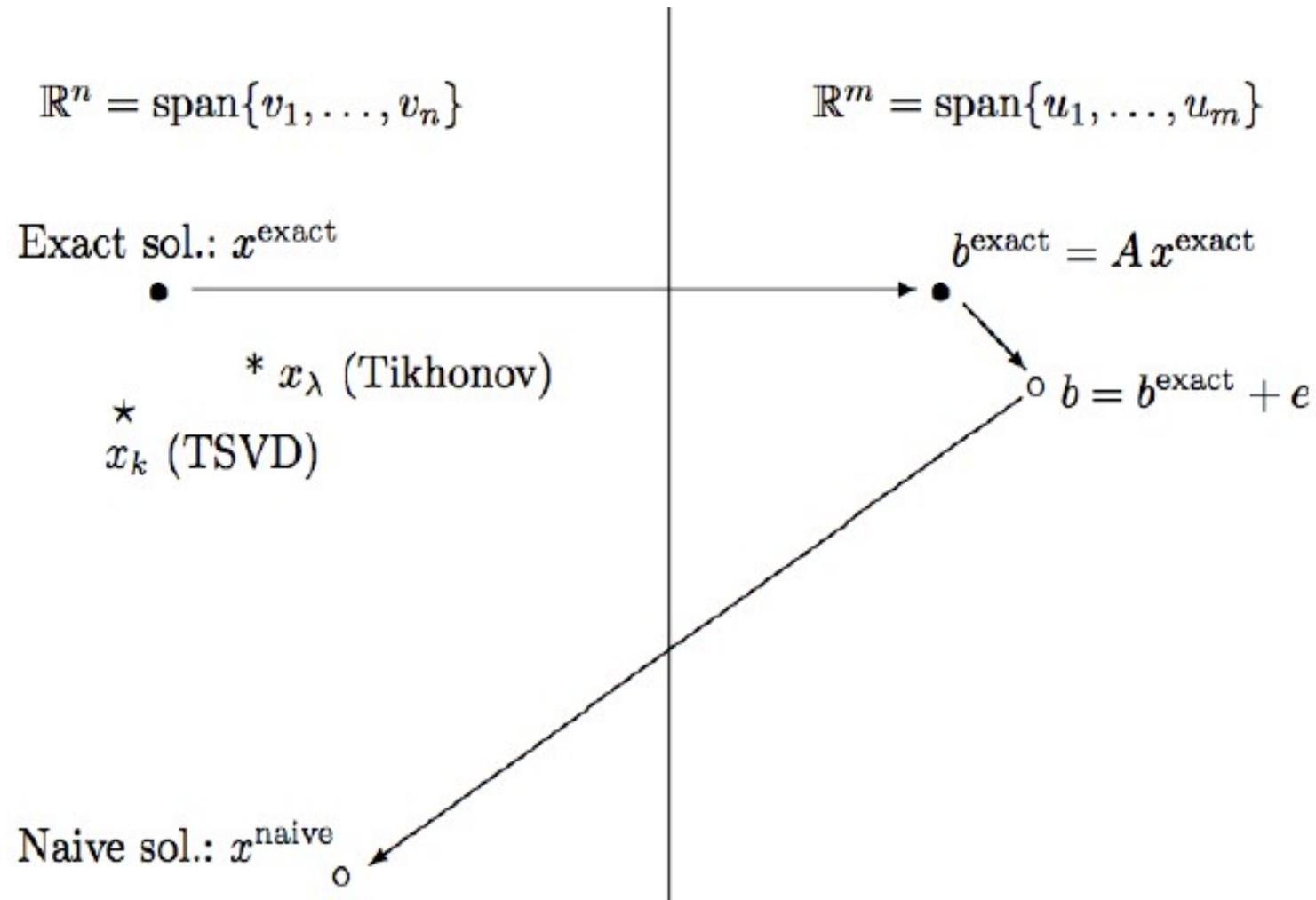
Depende del modelo de ruido.



Regularizador  
o conocimiento  
previo.

Multiplicador de Lagrange.  
Balance entre términos.

# Regularizador



# Enfoque Bayesiano

- Queremos encontrar la imagen ideal con mayor probabilidad dada una imagen capturada

$$\operatorname{argmax}_f p(f|g) = \operatorname{argmax}_f \frac{p(g|f)p(f)}{p(g)}$$

## Enfoque bayesiano

- Queremos encontrar la imagen ideal con mayor probabilidad dada una imagen capturada

$$\operatorname{argmax}_f p(f|g) = \operatorname{argmax}_f \frac{p(g|f)p(f)}{p(g)}$$

$$\begin{aligned} \operatorname{argmin}_f [-\log(p(f|g))] &\propto \operatorname{argmin}_f [-\log(p(g|f)) - \log(p(f))] \\ &= \operatorname{argmin}_f C(f) + R(f) \end{aligned}$$

# Ruido Gaussiano

- Sabemos que:

$$p(g|f) = \prod_i^n \left( \frac{1}{\sigma \sqrt{2\pi}} \right) e^{-\frac{1}{2\sigma^2}(Af_i - g_i)^2}$$

$$-\log(p(g|f)) \propto \sum_i^n (Af_i - g_i)^2 = \|Af - g\|_2^2$$

## Regularizadores más utilizados

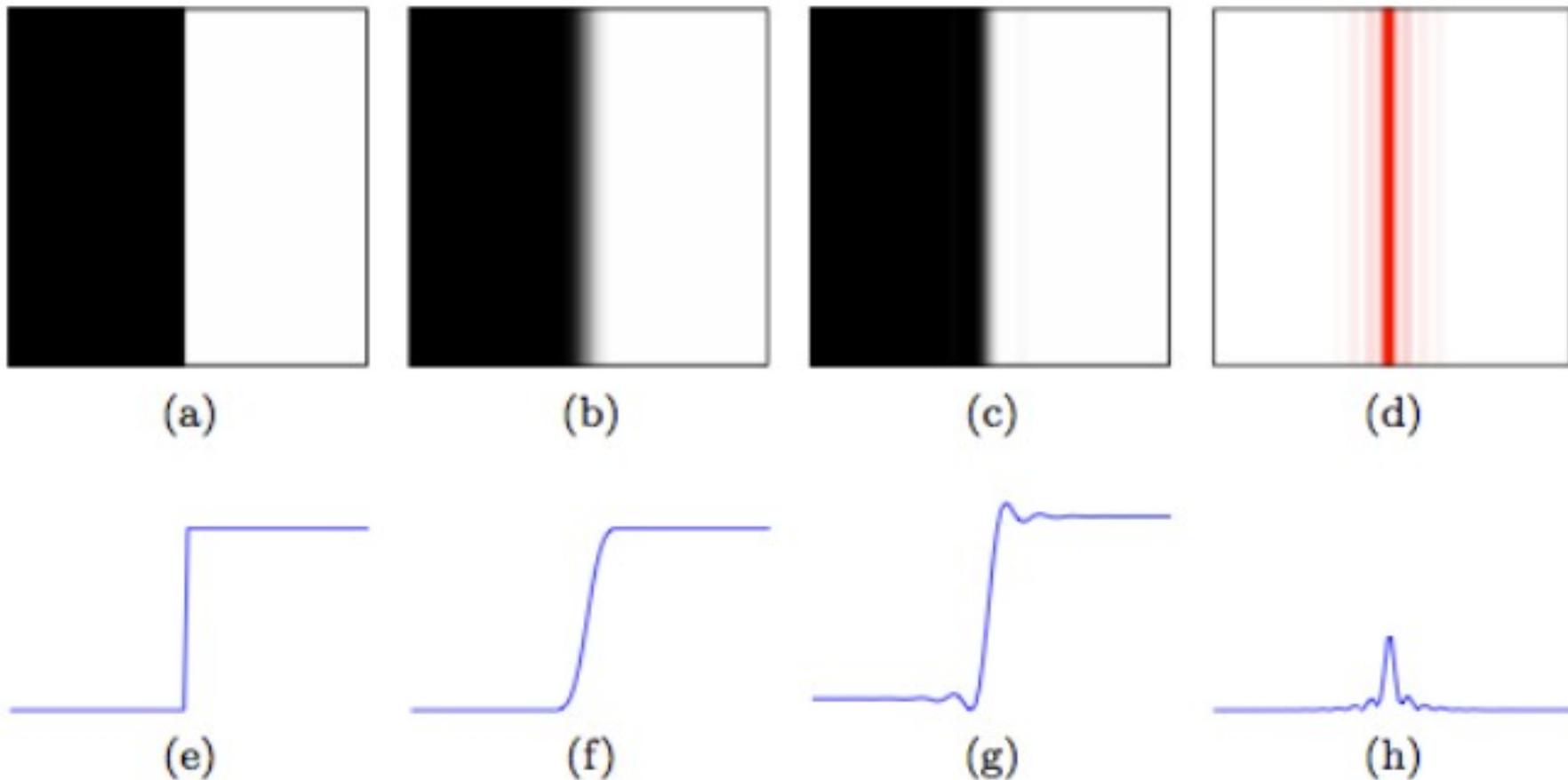
- Tikhonov:

$$\|L(f - f_0)\|^2 = (f - f_0)^T L^T L (f - f_0) \quad \|f\|_2^2 \quad \|f - f_0\|^2$$

- Variación total (TV):

$$\|\nabla f\|_1 = \int_{\Omega} |\nabla f| dx$$

# Tikhonov genera ringing



Problema con las bases (Fourier) al ser eliminadas (aliasing) y el uso de la norma 2.

## Tikhonov genera ringing

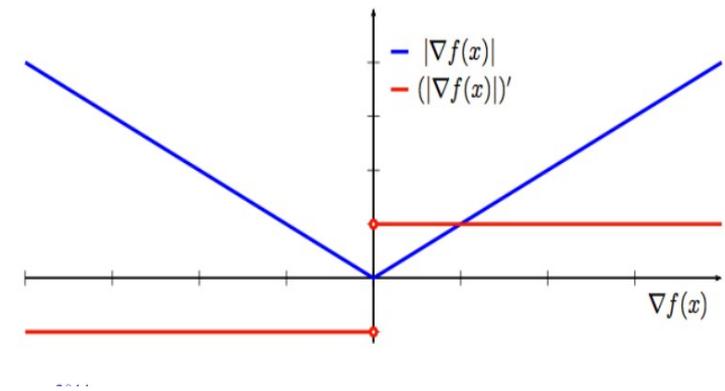
- Tikhonov y TSVD asumen que los datos son suaves y continuos.
- TV solo asume discontinuidades.
  - Converge a soluciones constantes a pedazos.

$$TV(f(x, y)) = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

$$TV_{L_1}(f) = \sum_{i,j} \left| \nabla f_x \right| + \left| \nabla f_y \right|$$

# Variación Total

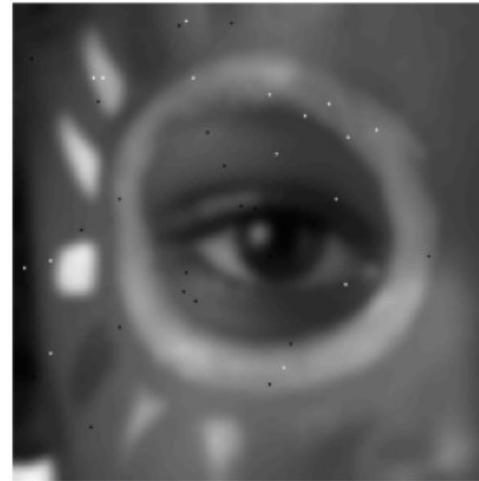
- Ventajas:
  - Preserva bien bordes.
  - No presenta ringing.
- Desventajas:
  - Mucho más difícil de resolver. (Usar algoritmos iterativos como el Gradiente Conjugado).
  - Efecto “escalera” o “cartoon”.
  - Cambios suaves se pierden.
  - Funcional NO es diferenciable.



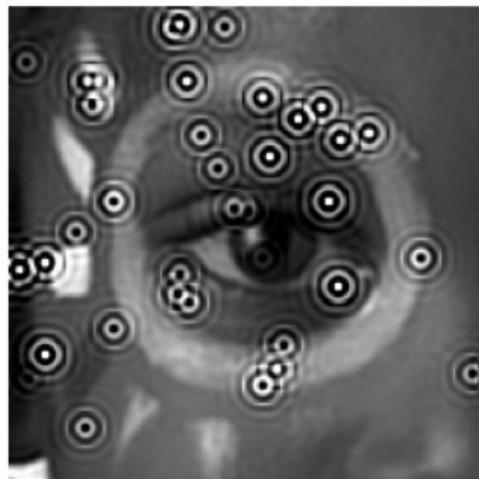
# Regularizadores más utilizados



(a) Original image



(b) Blurred image, with added salt & pepper noise



(c) Tikhonov restoration of salt & pepper noise image



(d) TV solution to the same problem

# Variación Total

Original clean image



Noisy image



TV denoised image,  $\tau = 0.85$

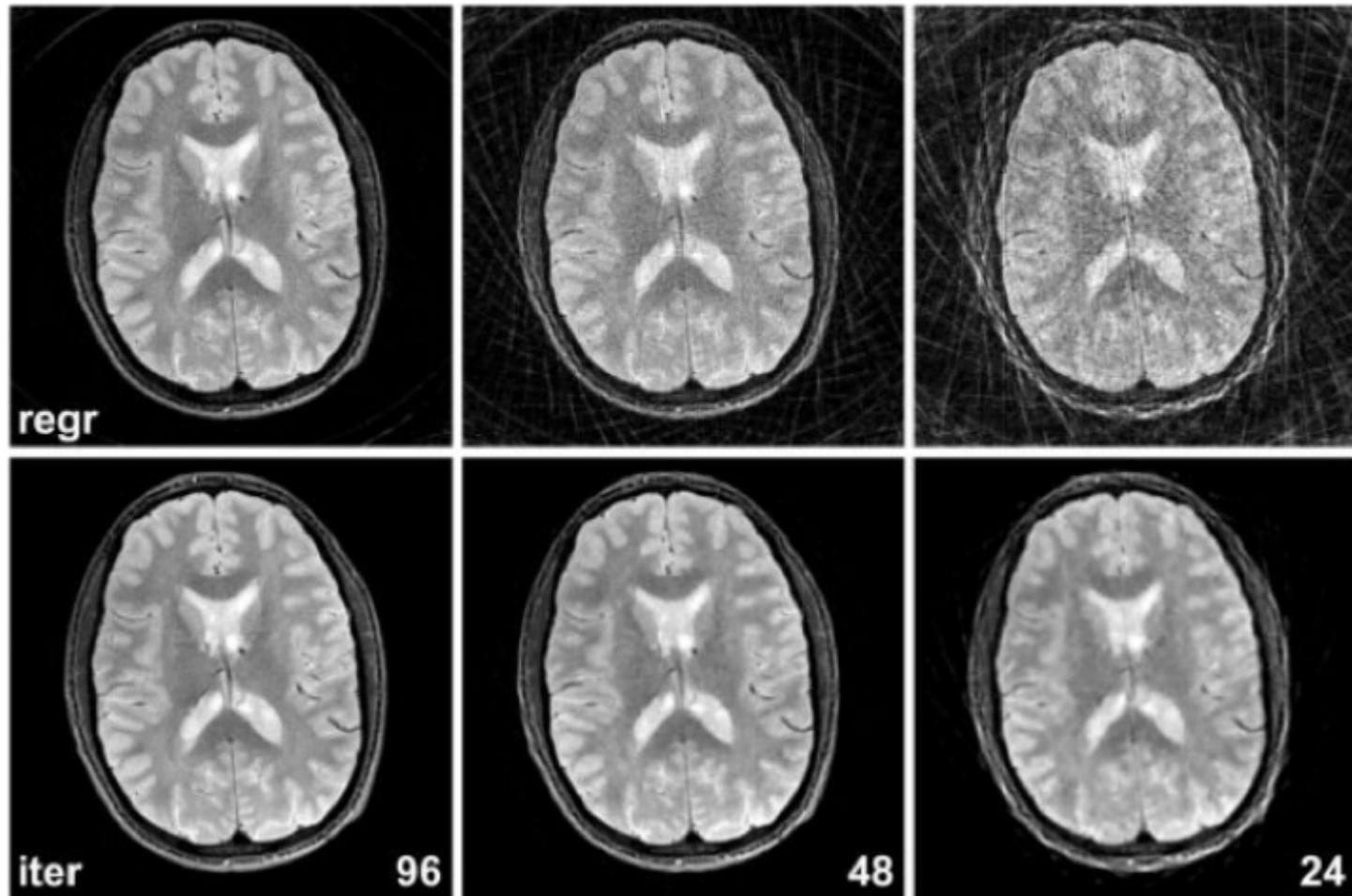


TV denoised image,  $\tau = 1.2$



# Variación Total

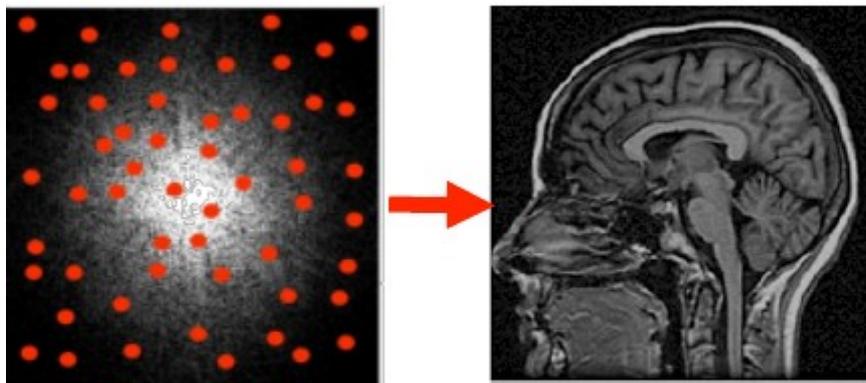
- Ejemplo reconstrucción submuestreada (radial):



# Paréntesis: Compressed Sensing

$$\min_f \left\| Af - g \right\|_2 + \lambda \left\| f \right\|_0$$

- Candès demostró que una señal **rala o compresible** se puede recuperar de manera exacta con alta probabilidad usando minimización con norma  $l_1$  si:



$$M > k\mu^2 S \log^4 N$$

M: número de medidas

N: tamaño de la señal

$\mu$ : grado de coherencia

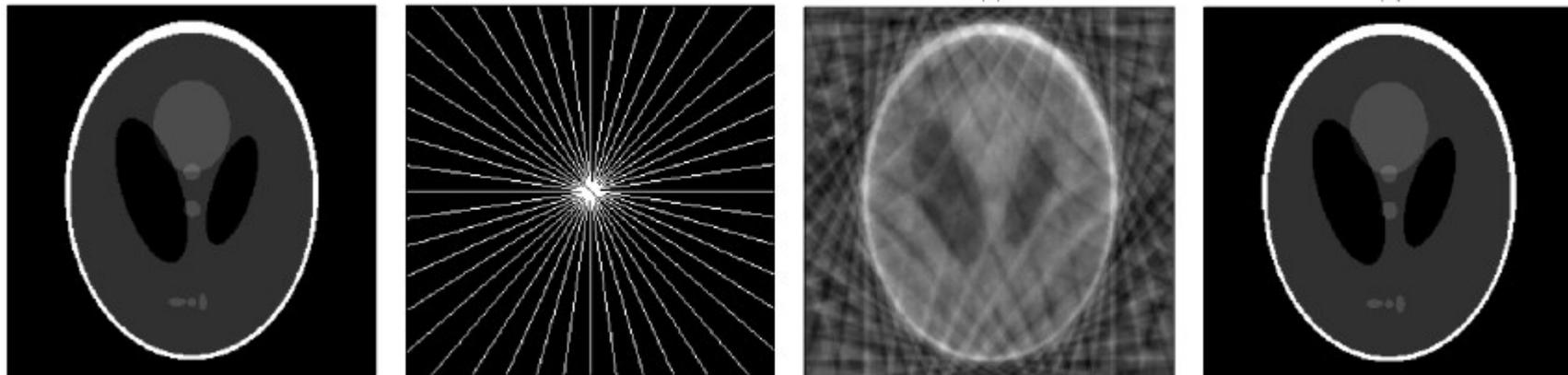
S: tamaño del soporte

$$\min_f \left\| f \right\|_1$$

s.a.  $Ef = g$

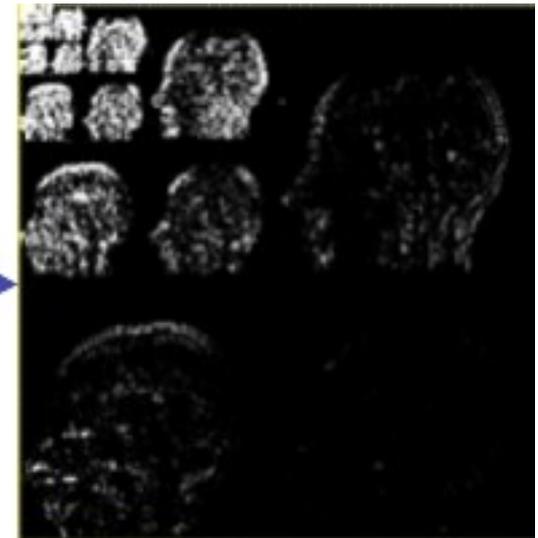
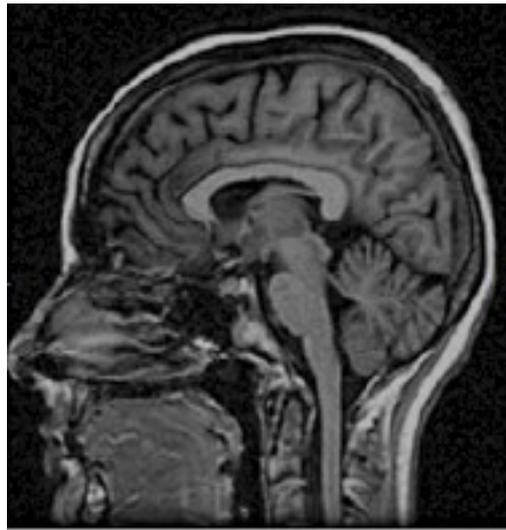
$$M \sim 3S - 5S$$

# Paréntesis: Compressed Sensing



# Paréntesis: Compressed Sensing

- Elegir bien el espacio raro.



- TV es el espacio raro del gradiente.

# Variación Total

- Ejemplo inpainting:

Noisy and corrupted image



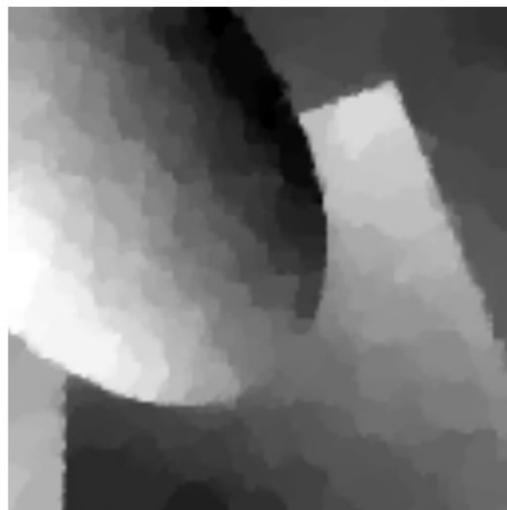
TV inpainted image,  $\tau = 0.85$



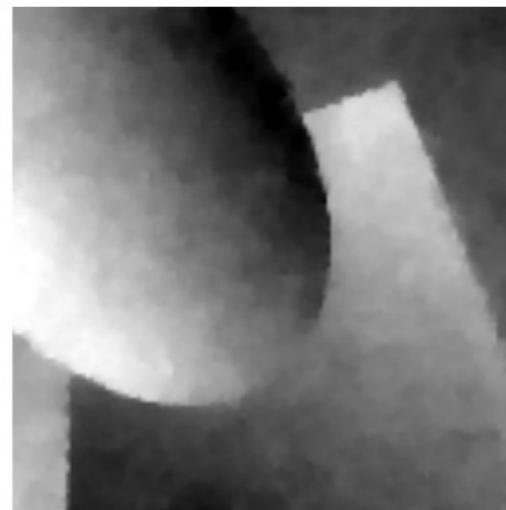
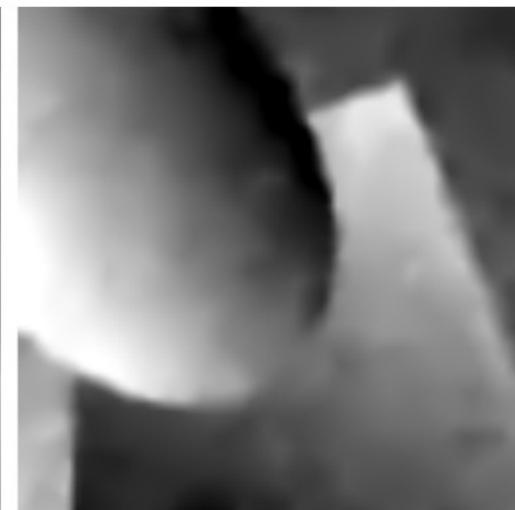
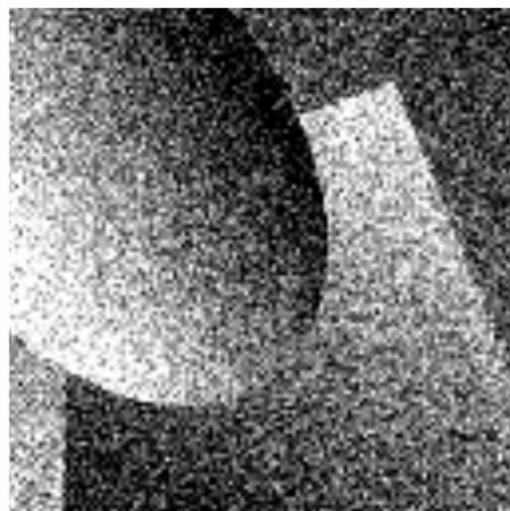
# Artefactos tipo escalera



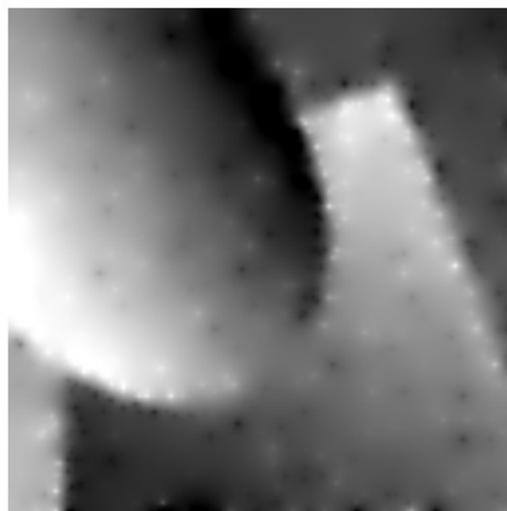
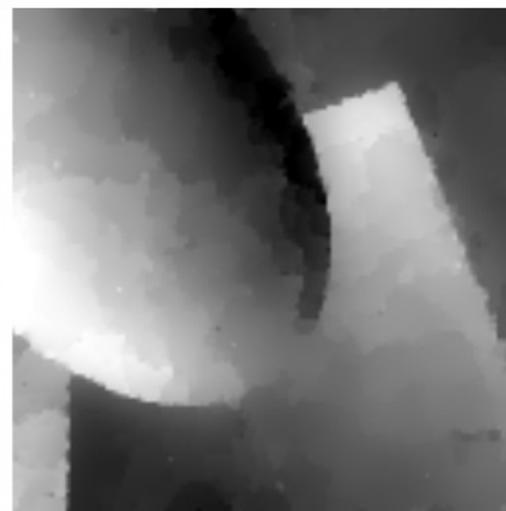
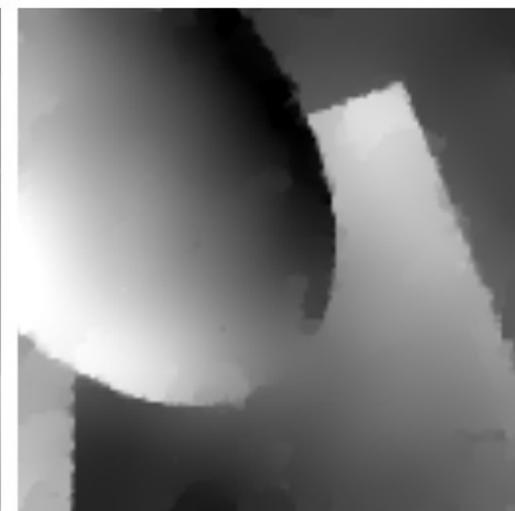
original image



TV

 $TV_\epsilon$  $TV^2$ 

noisy image

 $\|\Delta \cdot\|_{\mathcal{M}}$  $TV \Delta \beta TV^2$  $TGV_\alpha^2$

# Variación Total Generalizada

$$TGV_{\alpha}^2(f) = \min_v \alpha_1 \int_{\Omega} |\nabla f - v| dx + \alpha_0 \int_{\Omega} |\xi(v)| dx$$

donde:

$$\xi(v) = \frac{1}{2} (\nabla v + \nabla v^T)$$

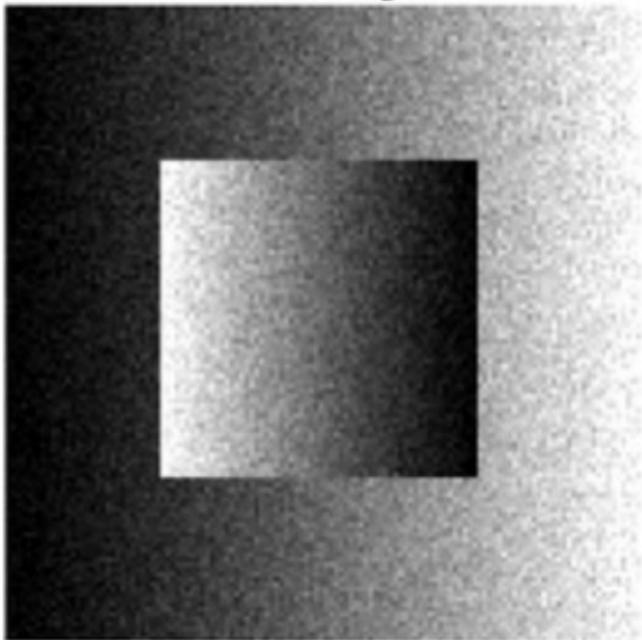
# Variación Total Generalizada

$$TGV_{\alpha}^2(f) = \min_v \alpha_1 \int_{\Omega} |\nabla f - v| dx + \alpha_0 \int_{\Omega} |\xi(v)| dx$$

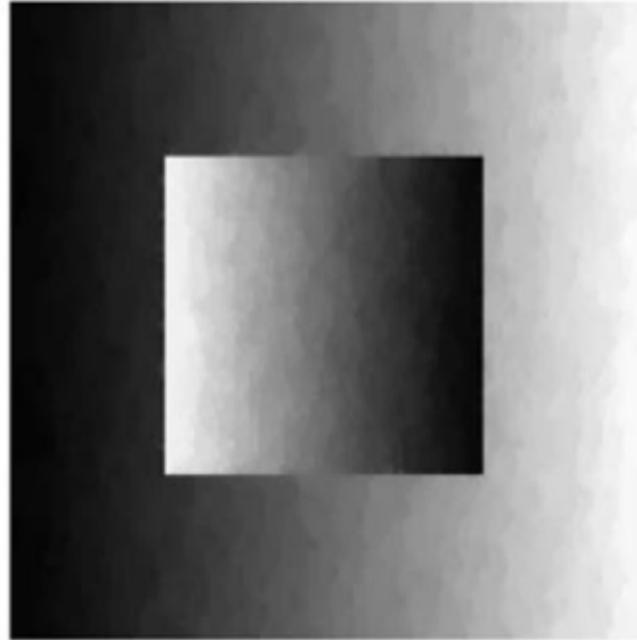
donde:

$$\xi(v) = \frac{1}{2} (\nabla v + \nabla v^T)$$

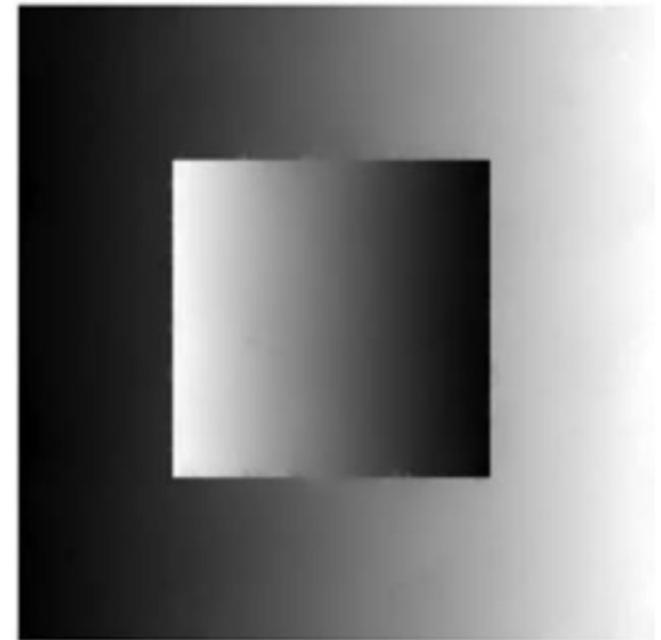
Noisy



TV



TGV

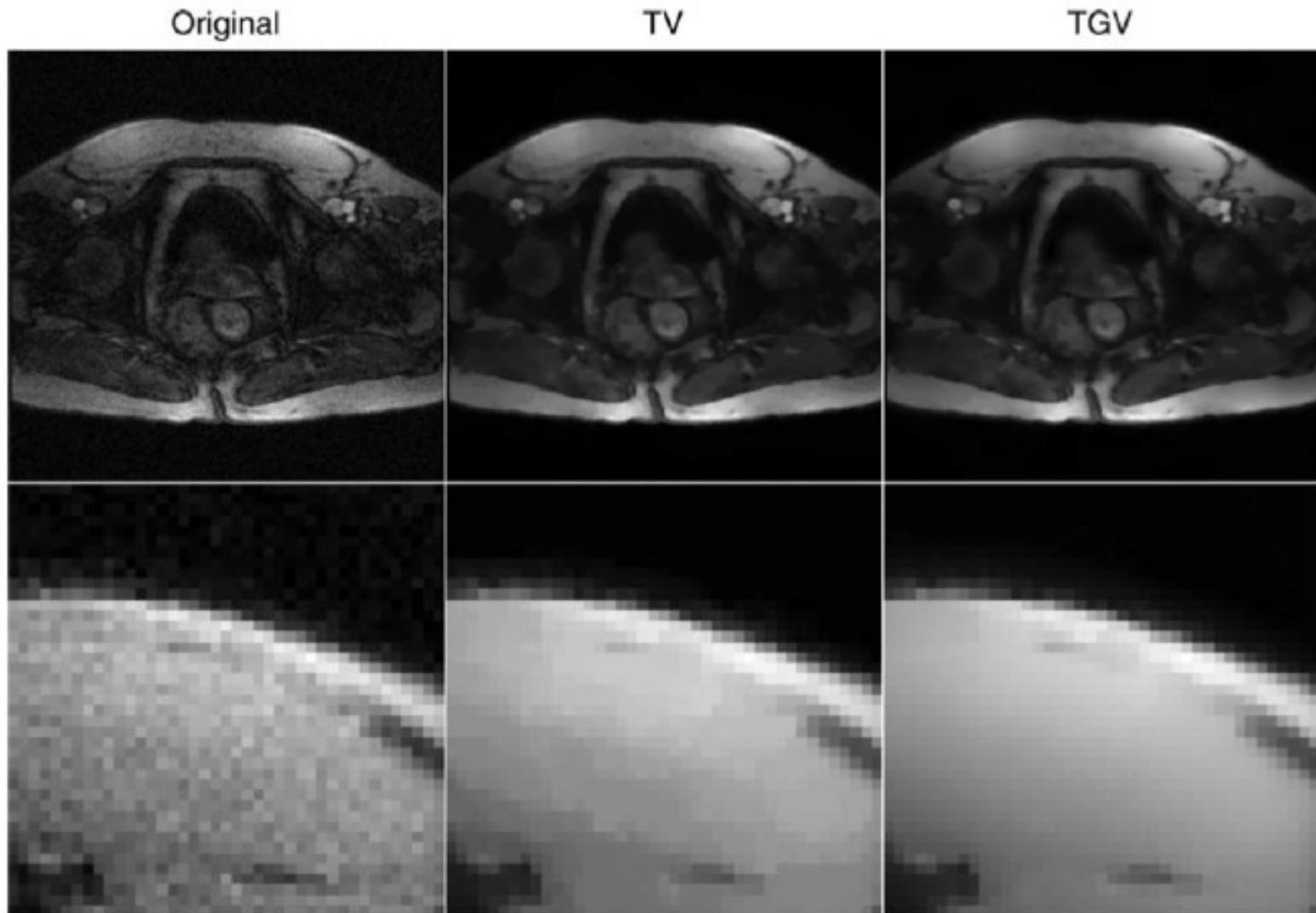


# Variación Total Generalizada

- Intuitivamente:
  - En las regiones suaves de una imagen dada  $f$ , la segunda derivada  $\nabla^2 f$  es localmente “pequeña”.
  - Por lo tanto para la minimización es bueno escoger  $\nu = \nabla f$  en estas regiones, resultando localmente en la penalización de  $\nabla^2 f$
  - Por otro lado en una vecindad de los bordes,  $\nabla^2 f$  va a ser considerablemente mayor que  $\nabla f$ , por lo tanto es bueno para la minimización escoger  $\nu = 0$  en estas áreas.
  - En estas áreas TGV mide la primera derivada que corresponde al “salto” en el borde.

# Variación Total Generalizada

- Ejemplo de denoising:



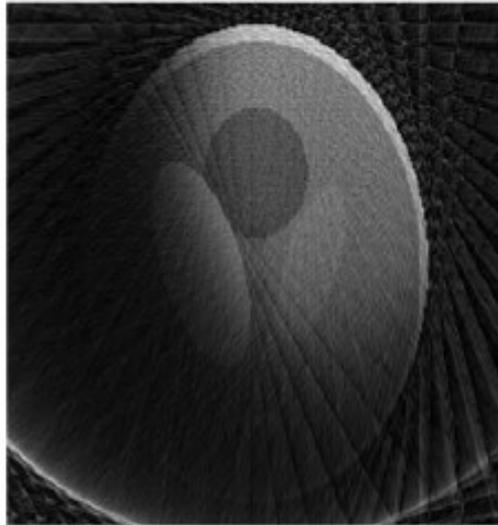
# Variación Total Generalizada

- Ejemplo de reconstrucción submuestreada (radial):

Original



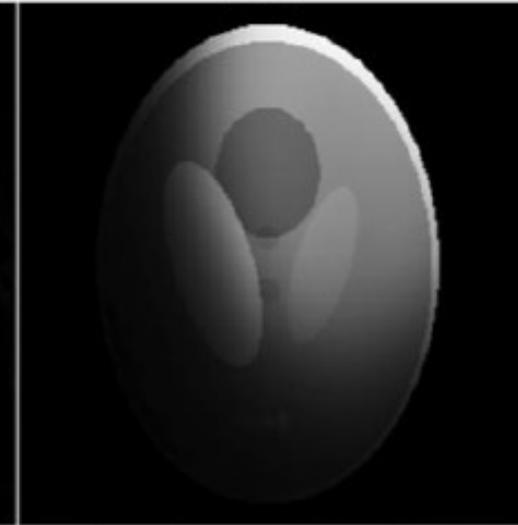
NUFFT



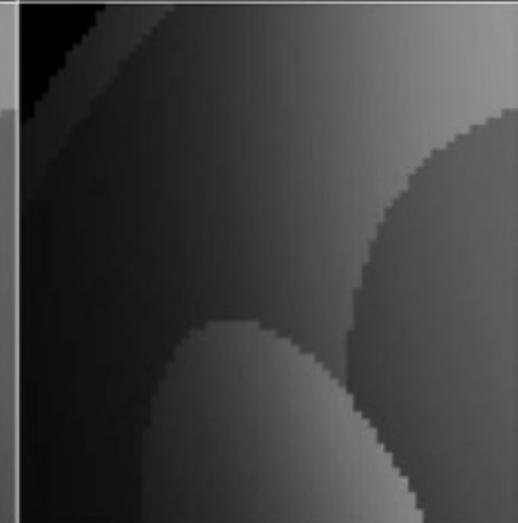
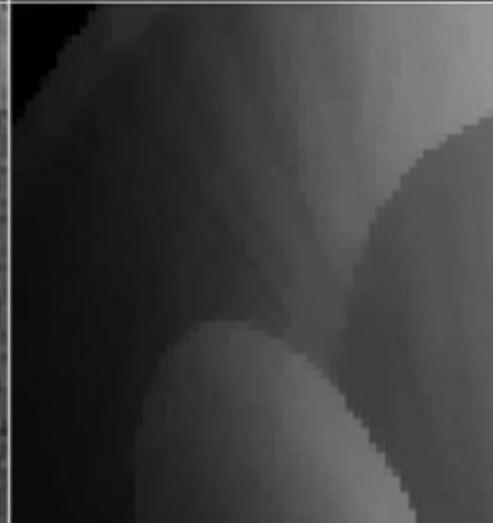
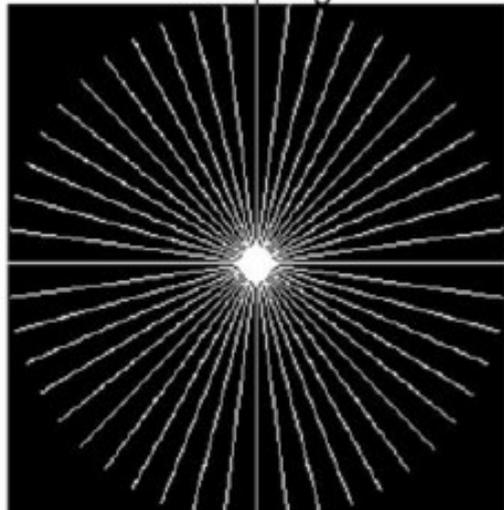
TV



TGV

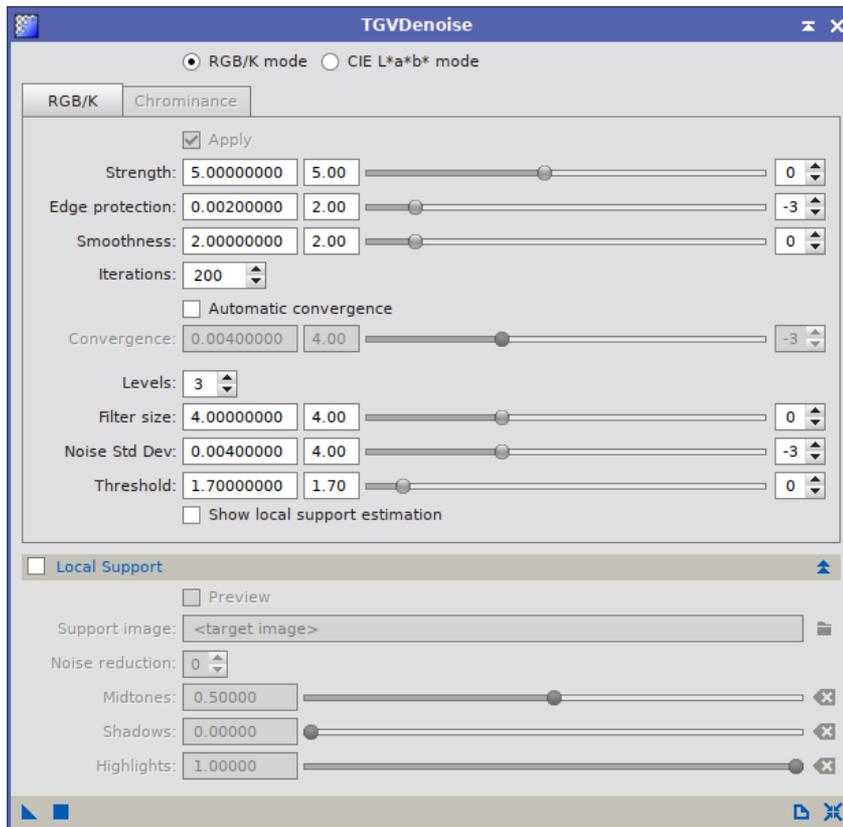


Sampling



# Ejemplos Denoising con TGV

- Performance:
  - Imagen 256x256,  
100 iteraciones
    - C++: 0.945s
    - Matlab: 7.062s
  - Imagen 1024x1024,  
100 iteraciones
    - C++: 2.233s
    - Matlab: 317.9s



# Ejemplos Denoising con TGV

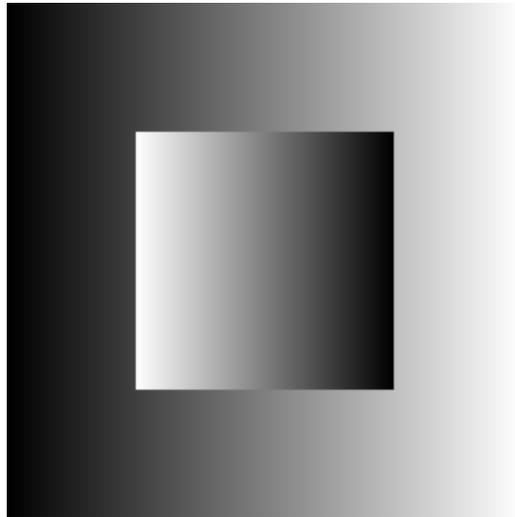
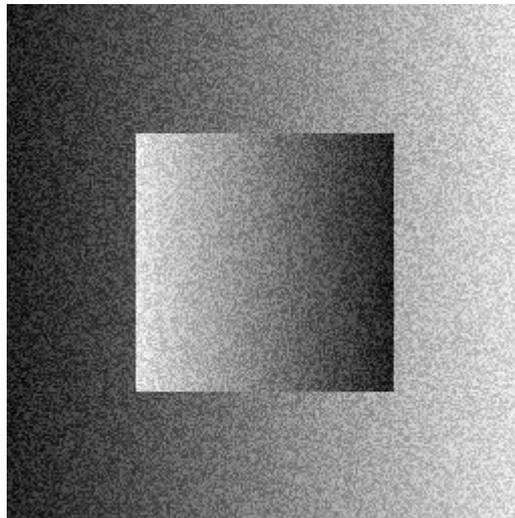
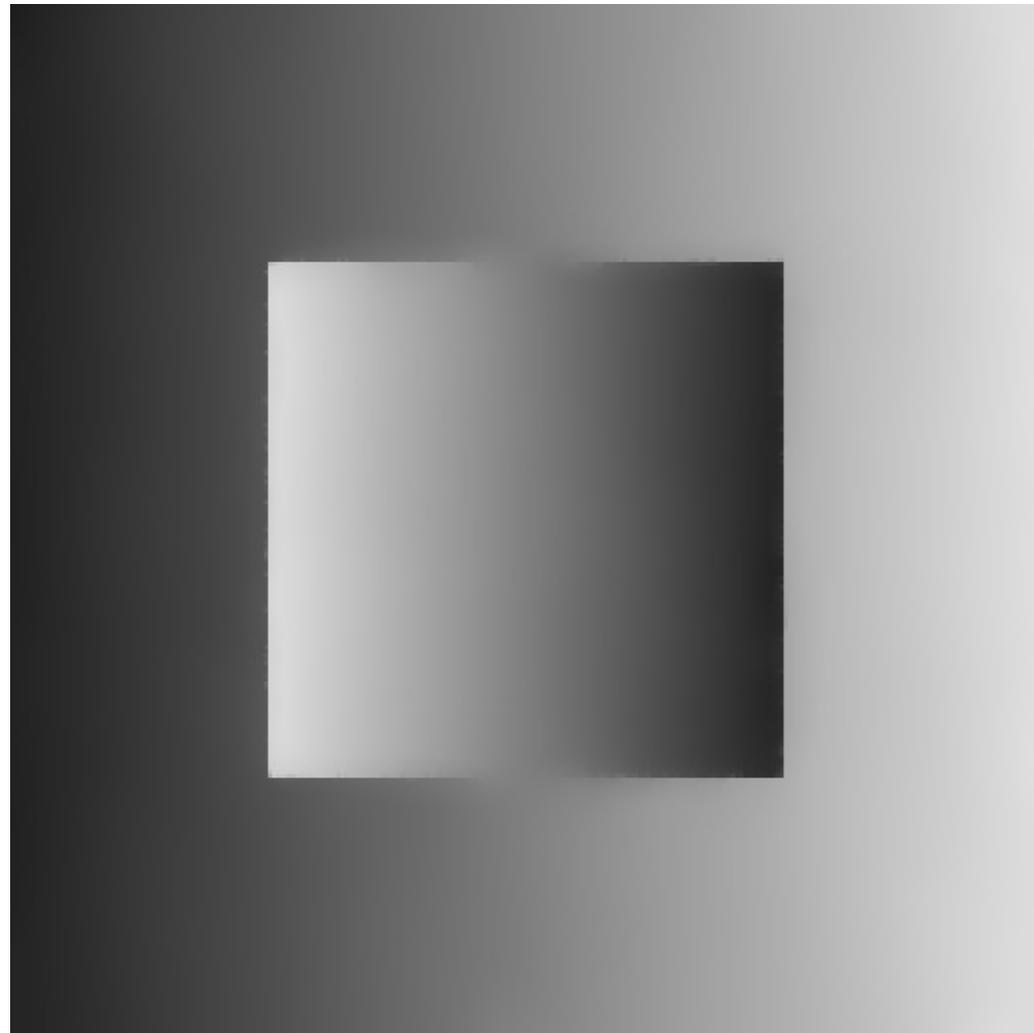


Imagen original

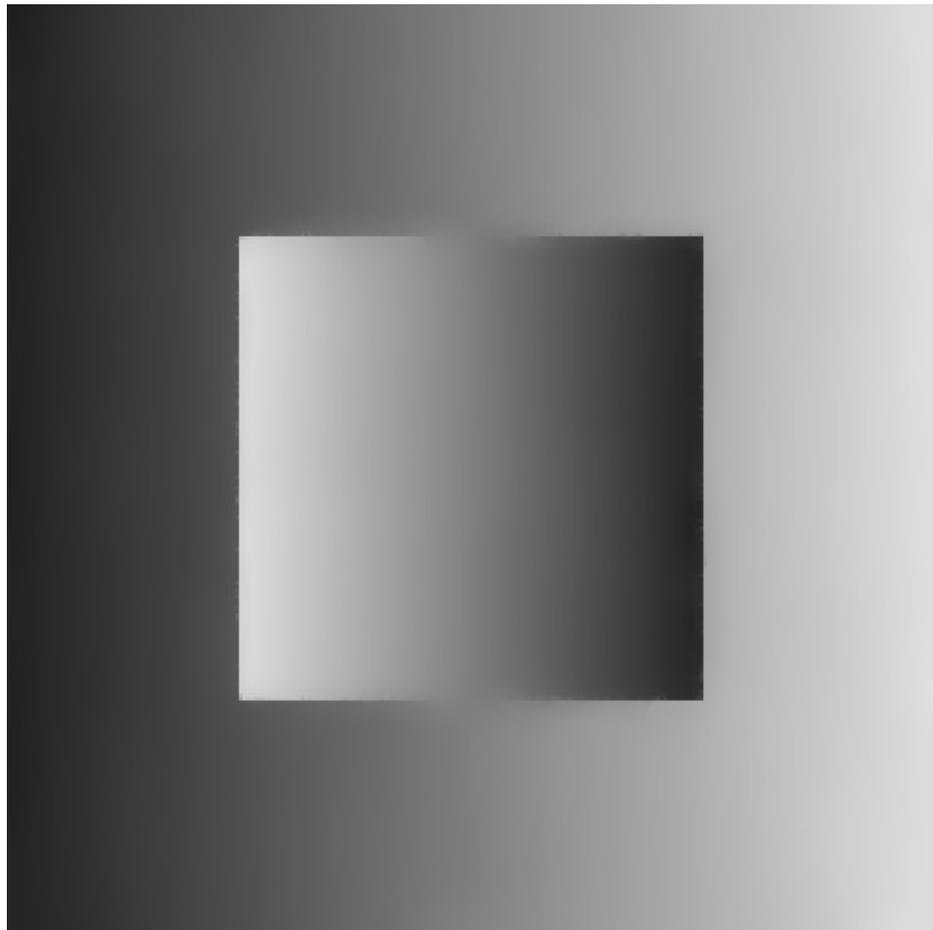


Ruido uniforme 25%

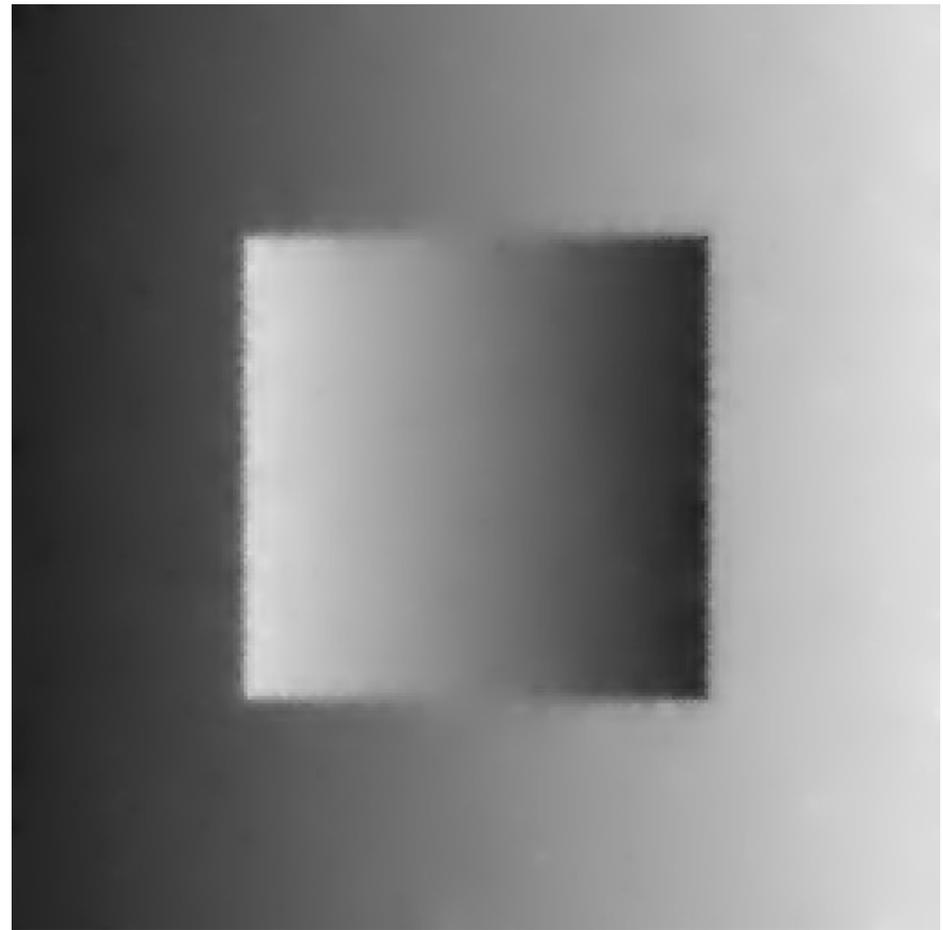


Resultado TGVDenoise

# Ejemplos Denoising con TGV

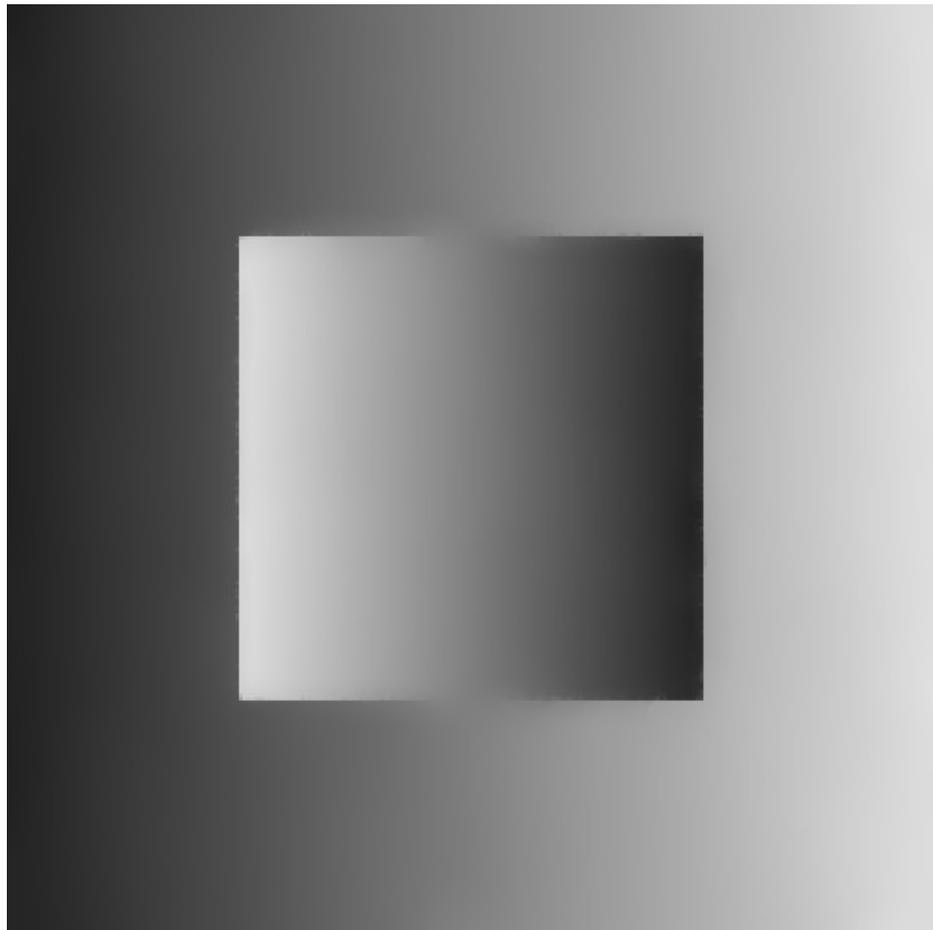


Resultado TGVDenoise

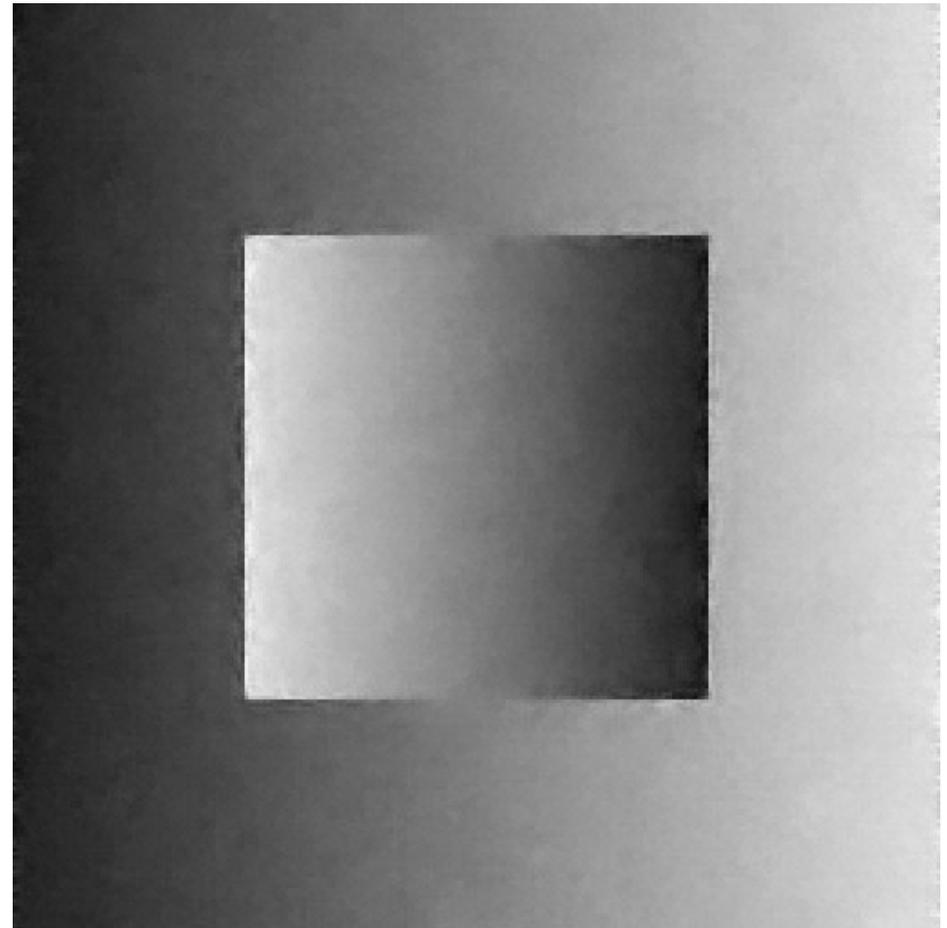


Resultado ATrousWavelet

# Ejemplos Denoising con TGV

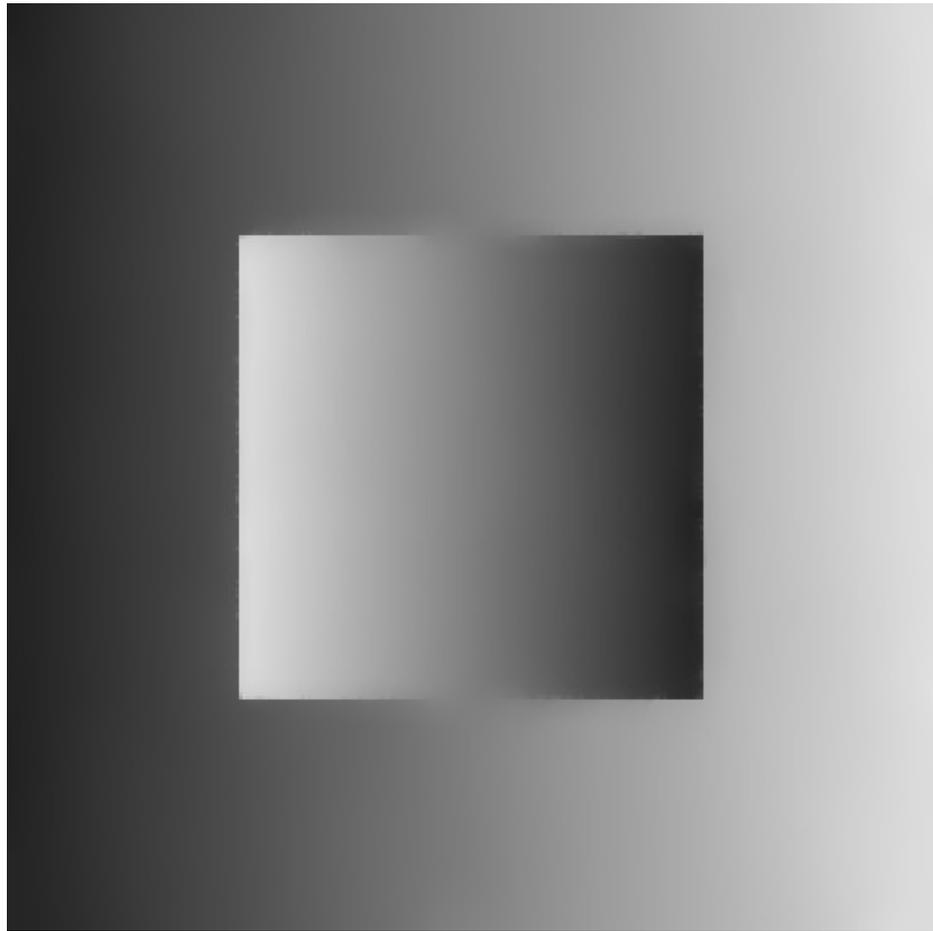


Resultado TGVDenoise

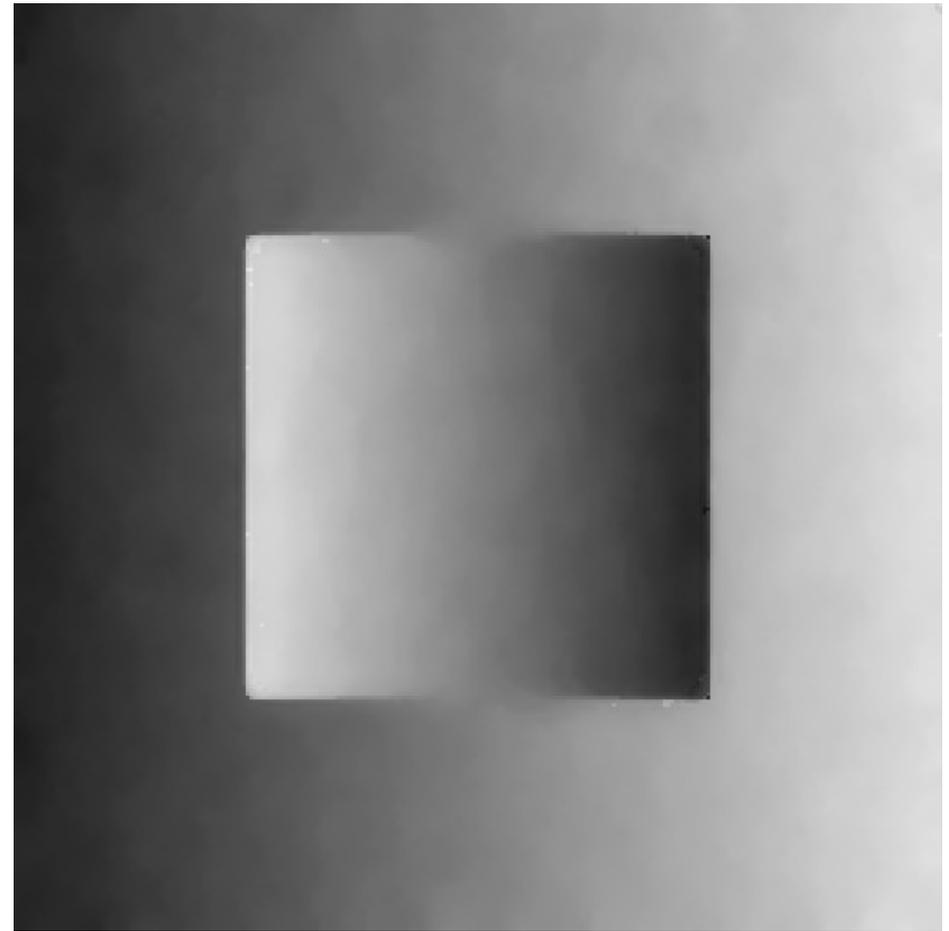


Resultado GREYCstoration

# Ejemplos Denoising con TGV

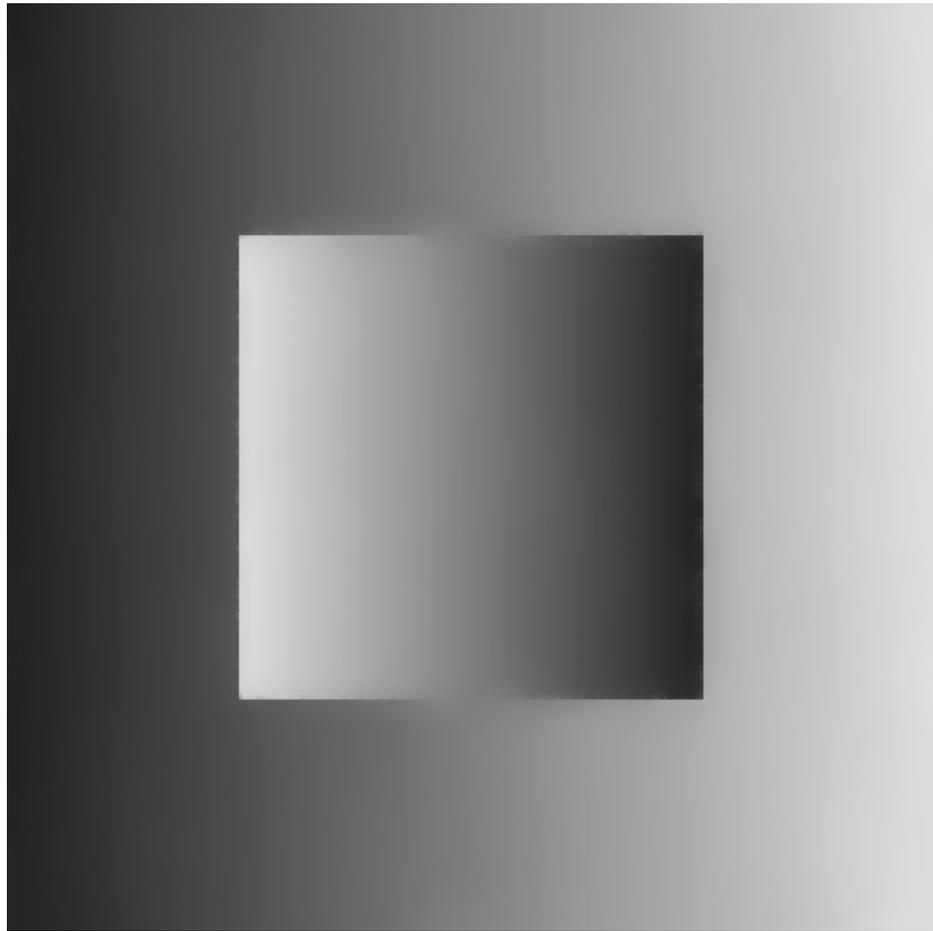


Resultado TGV Denoise

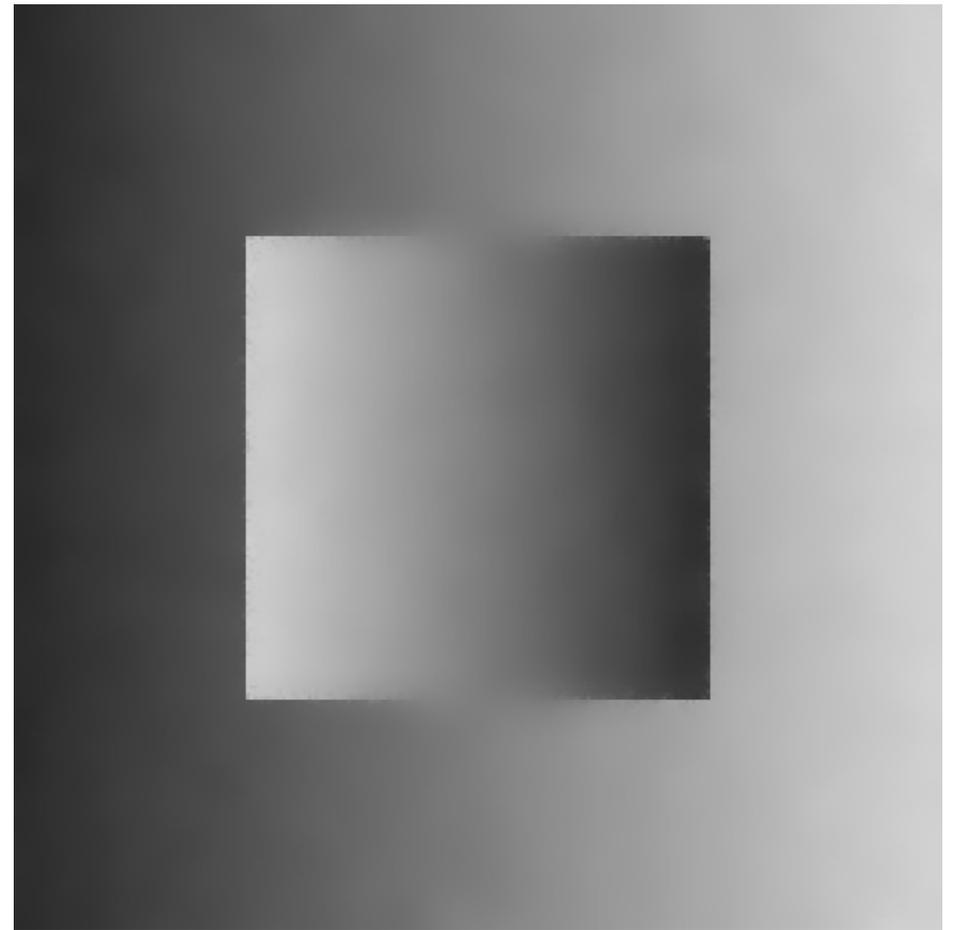


Resultado Multiscale  
Median Transform

# Ejemplos Denoising con TGV



Resultado TGVDenoise



Resultado ACDNR

# Ejemplos Denoising con TGV



Original



Resultado TGV Denoise

# Ejemplos Denoising con TGV



Original

Resultado TGVDenoise

# Ejemplos Denoising con TGV

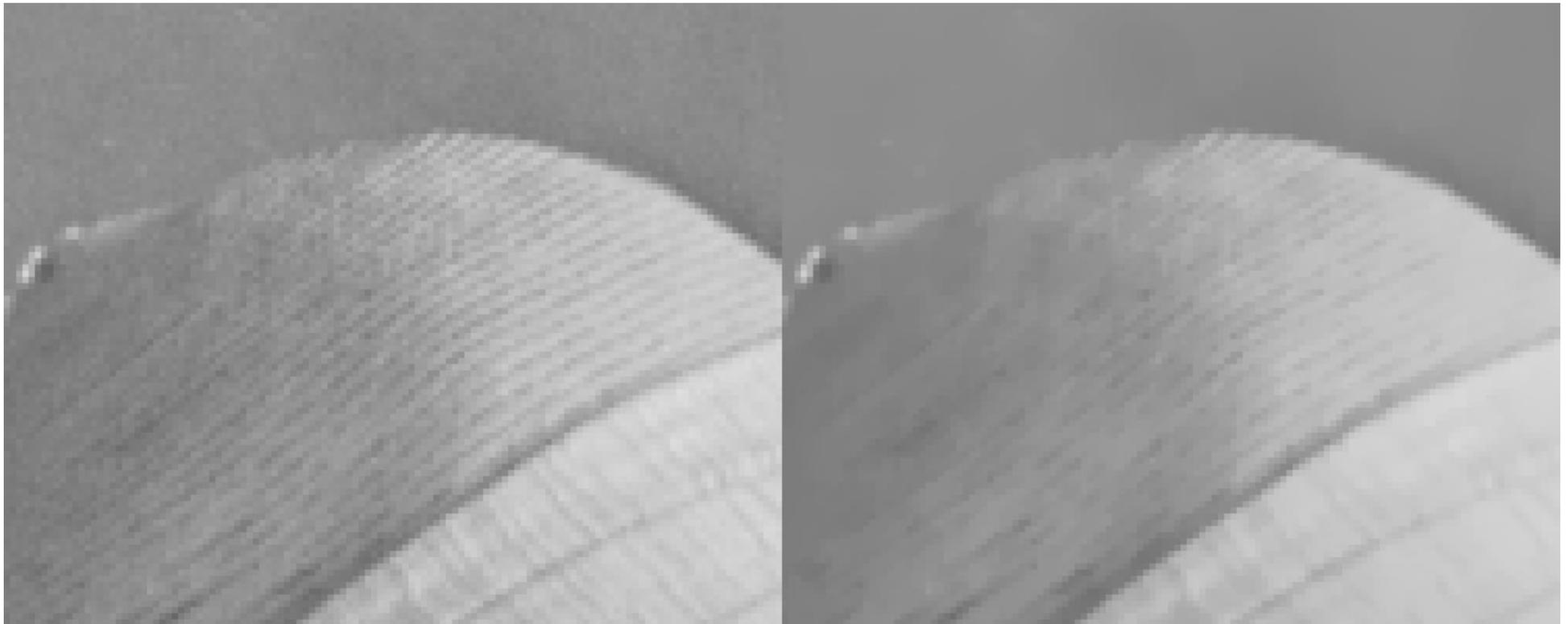


Original



Resultado TGVDenoise

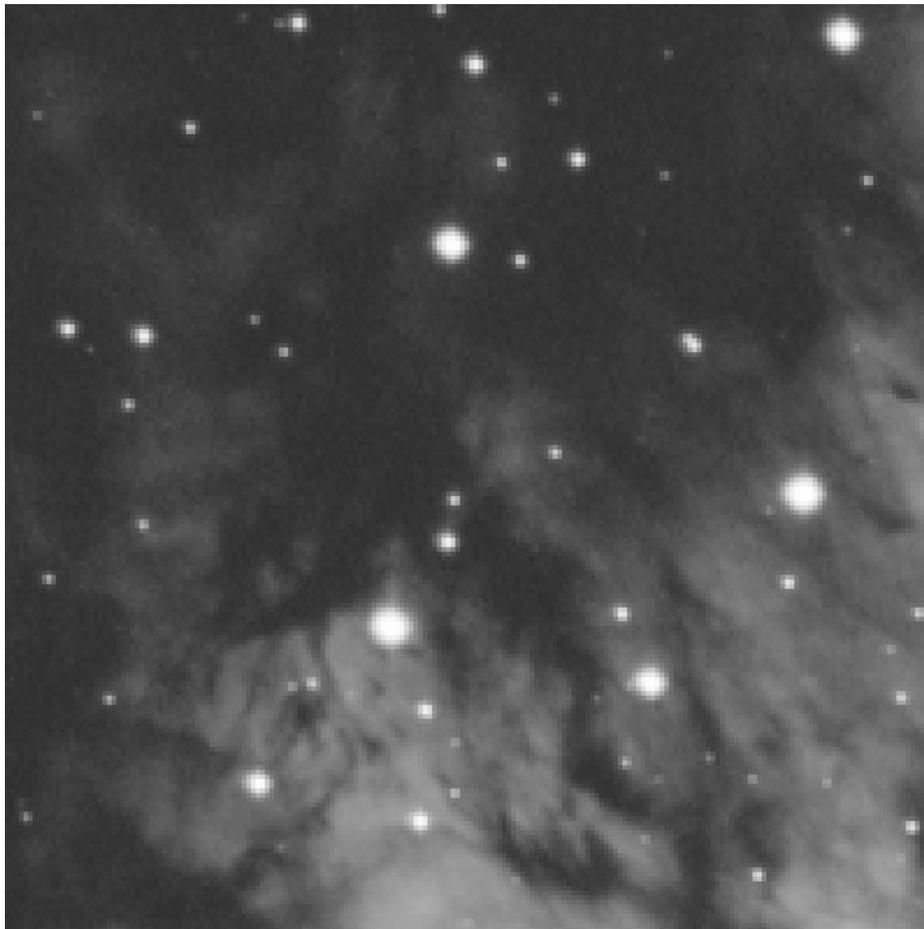
# Ejemplos Denoising con TGV



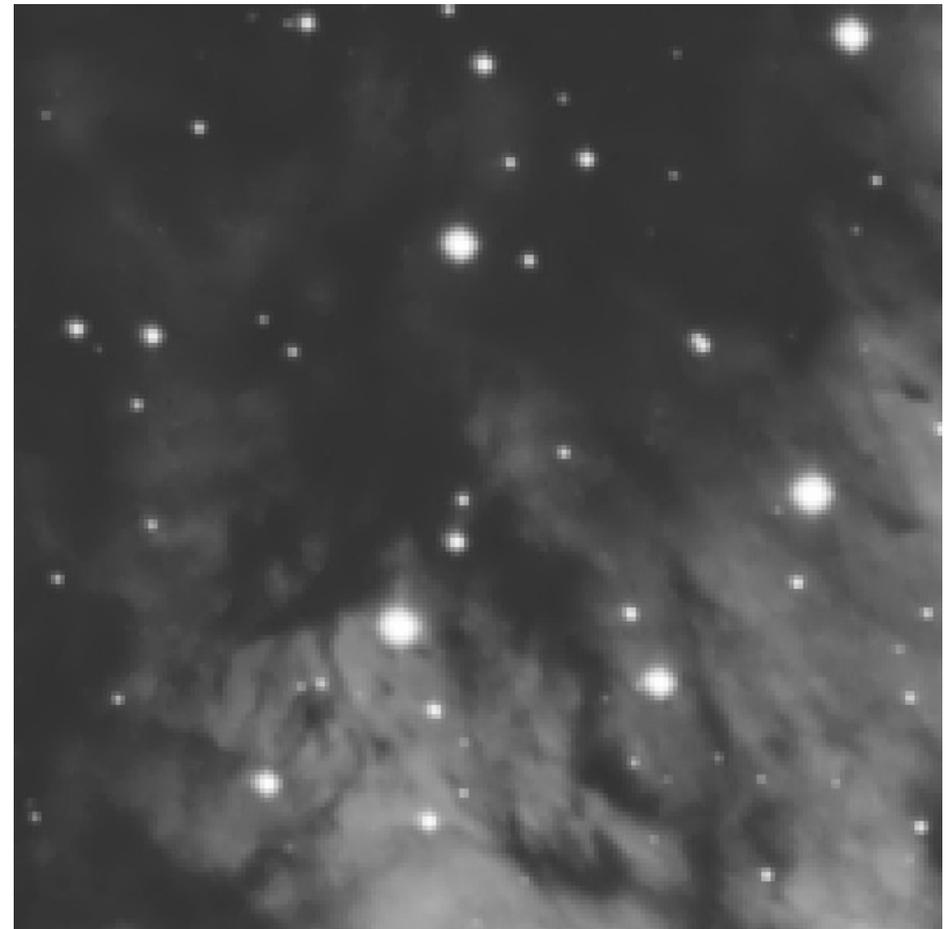
Original

Resultado TGVDenoise

# Imágenes astronómicas lineales



Original

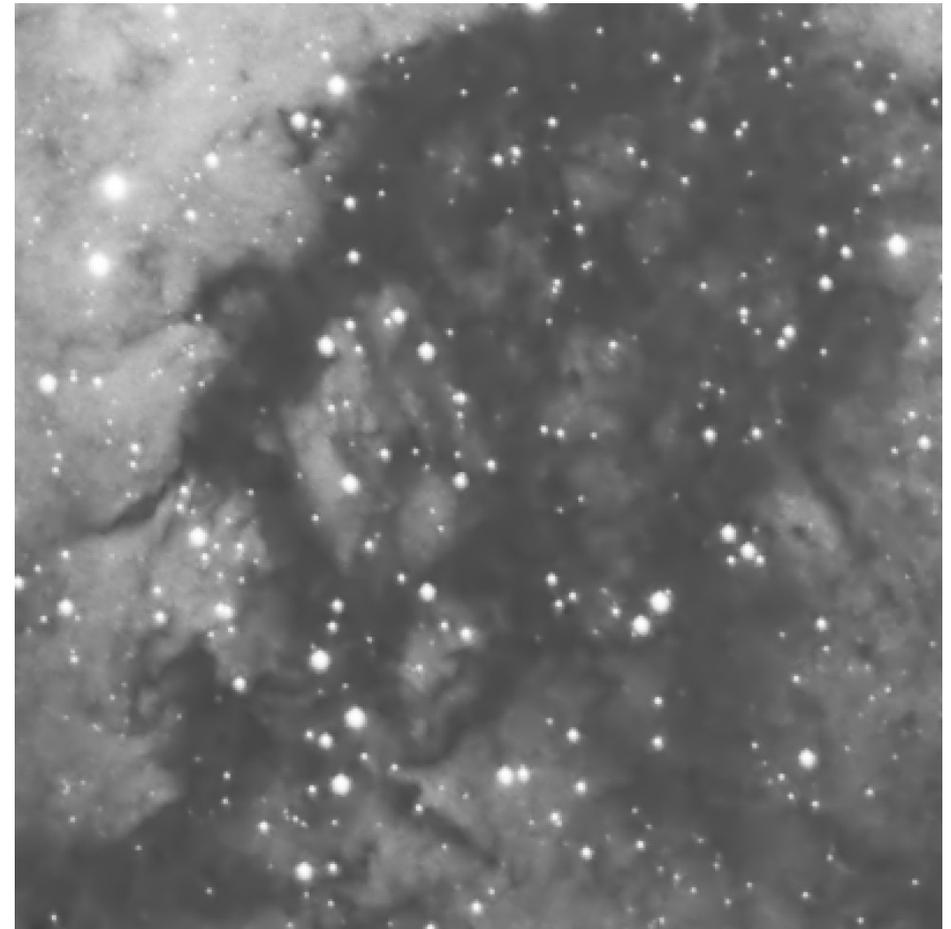


Resultado TGVDenoise

# Utilizando consistencia de datos espacialmente dependiente

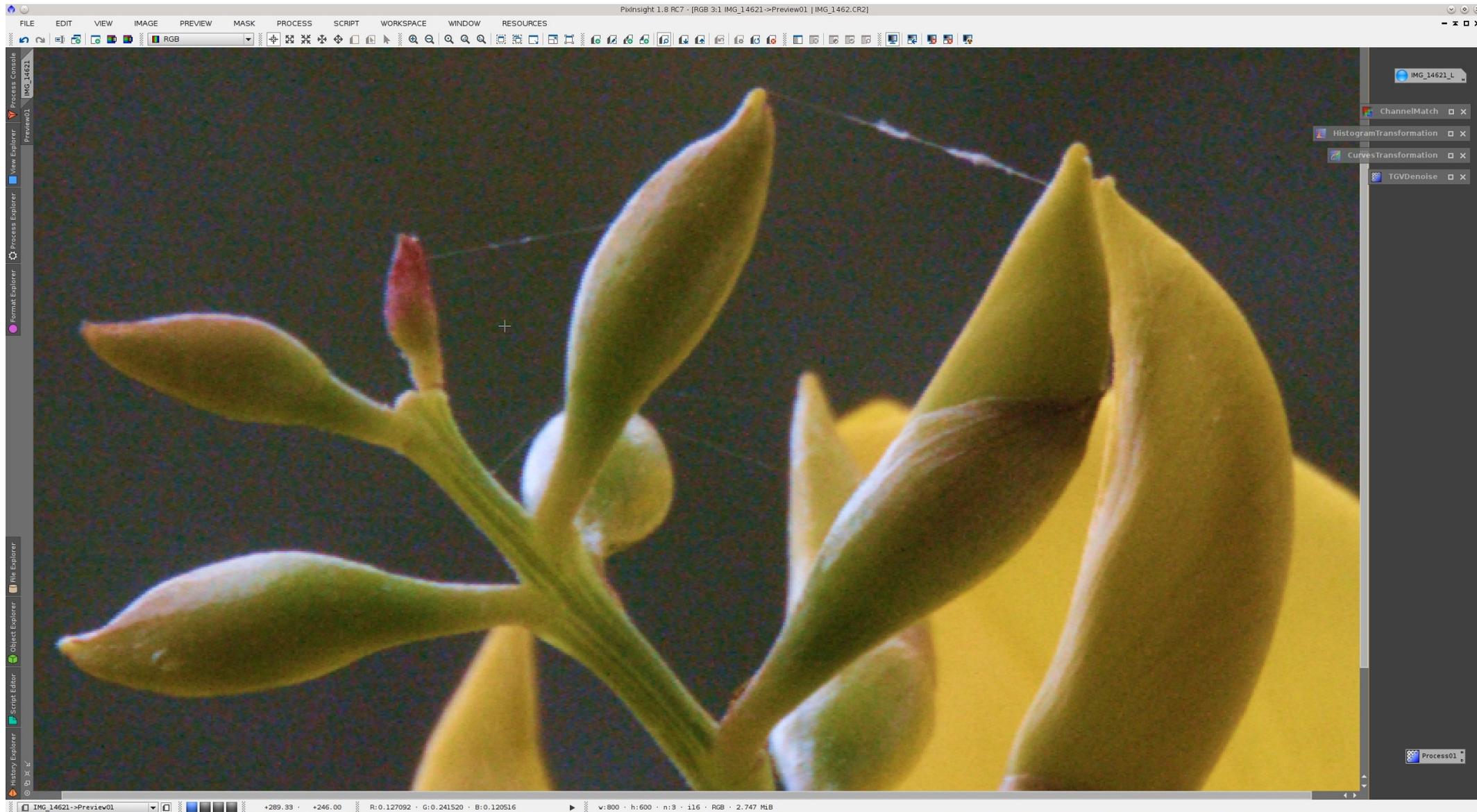


Original



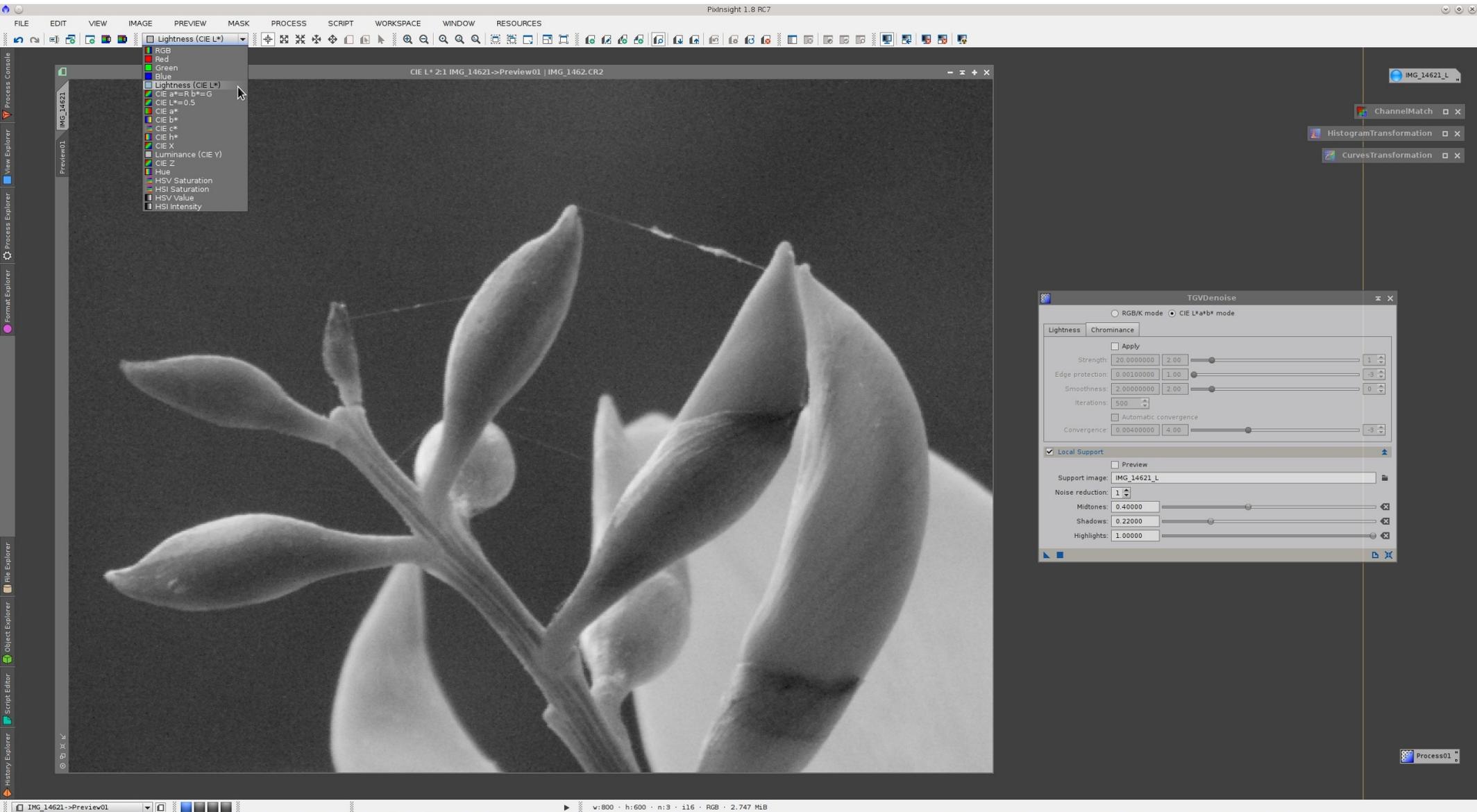
Resultado TGVDnoise

# Color (CIE Lab) + LS



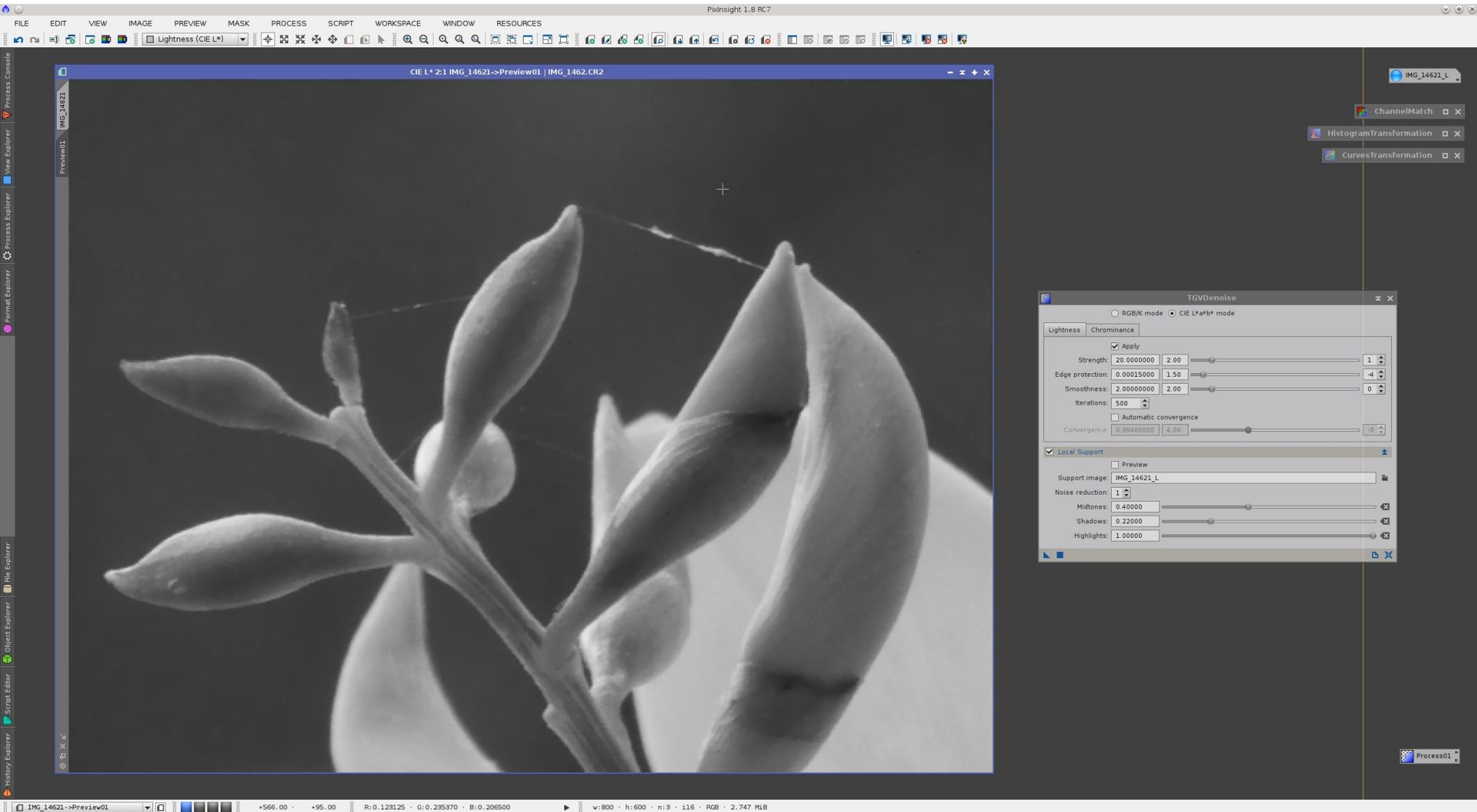
Original

# Color (CIE Lab) + LS



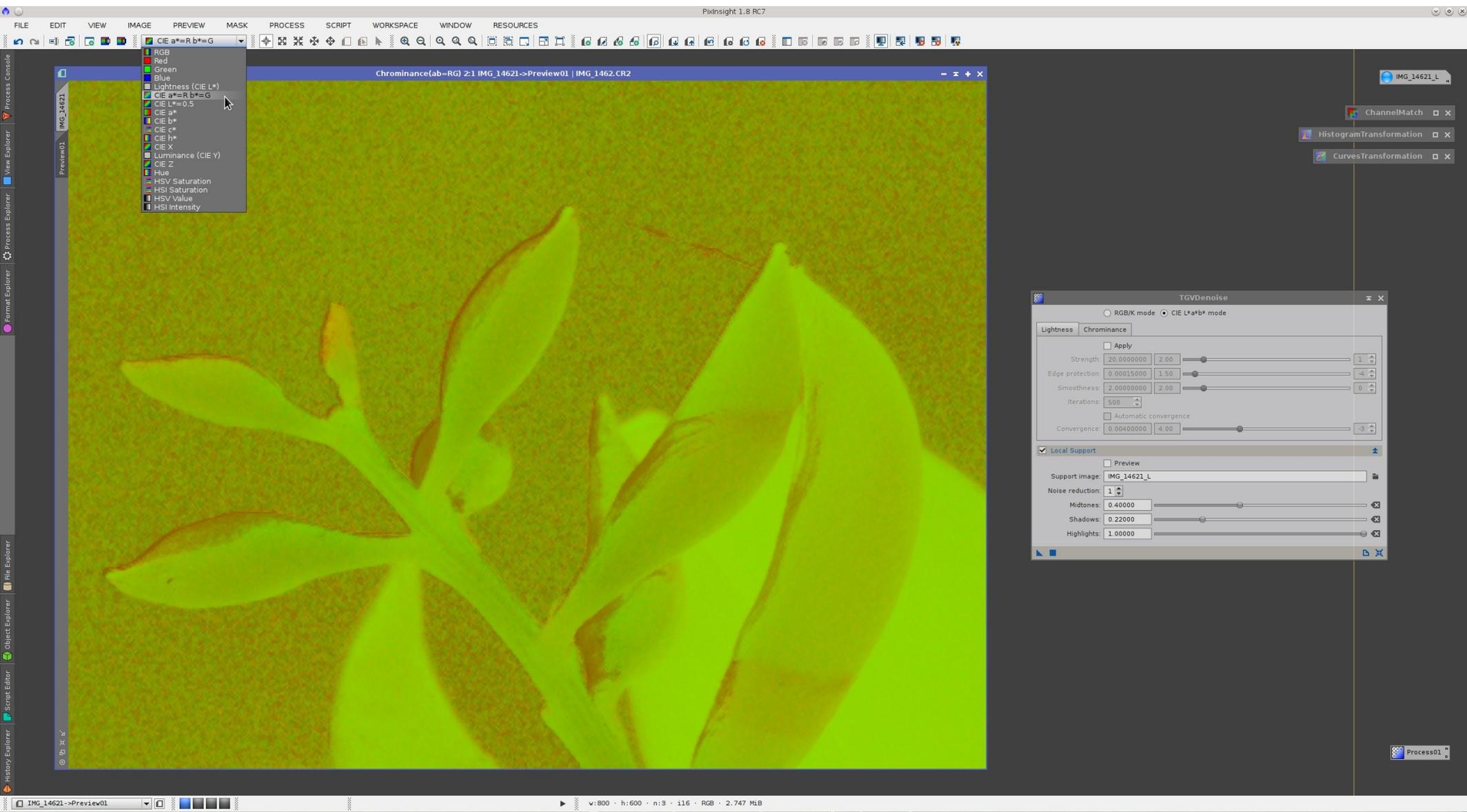
Luminancia original

# Color (CIE Lab) + LS



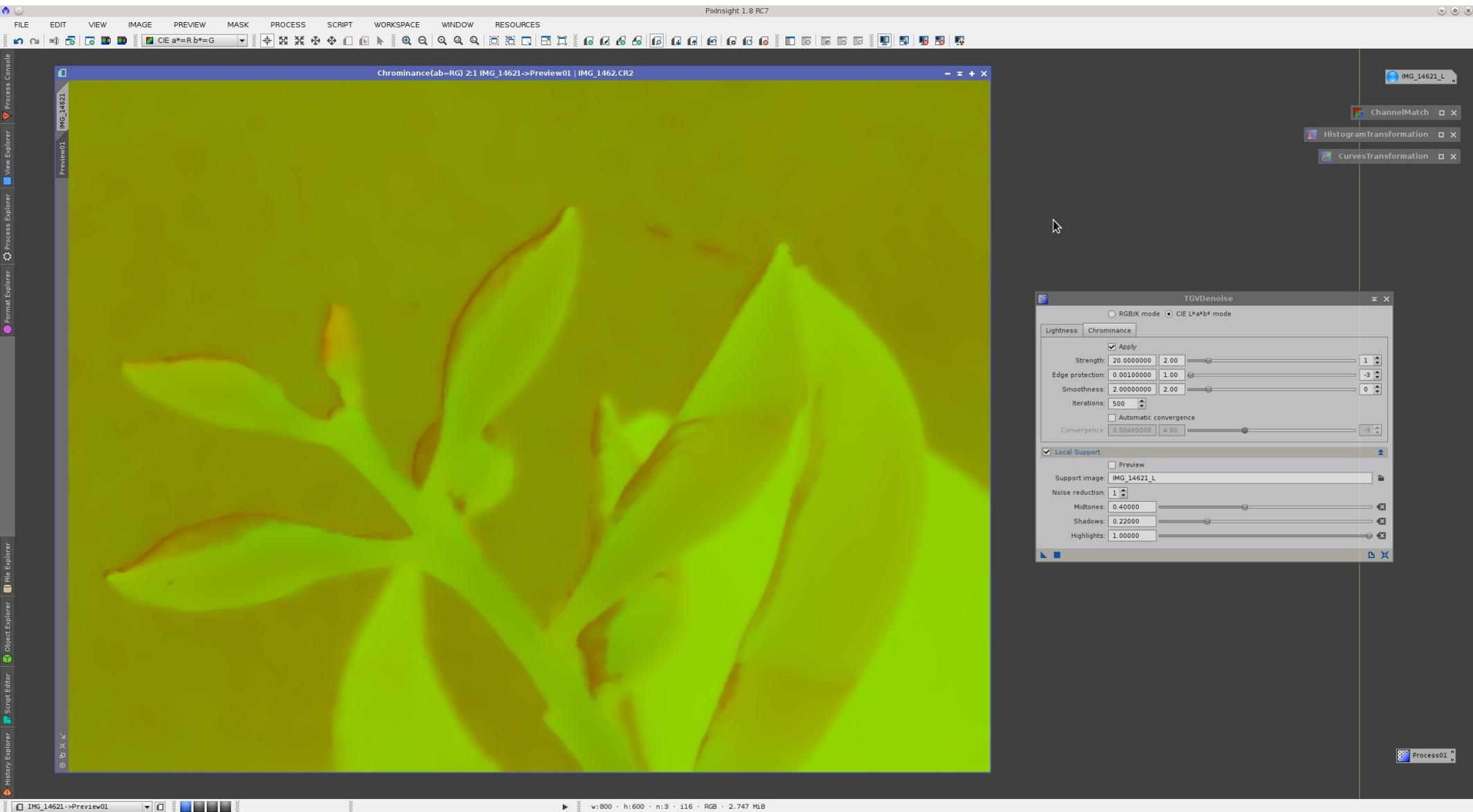
Luminancia procesada

# Color (CIE Lab) + LS



Crominancia original

# Color (CIE Lab) + LS



Crominancia procesada

# Color (CIE Lab) + LS

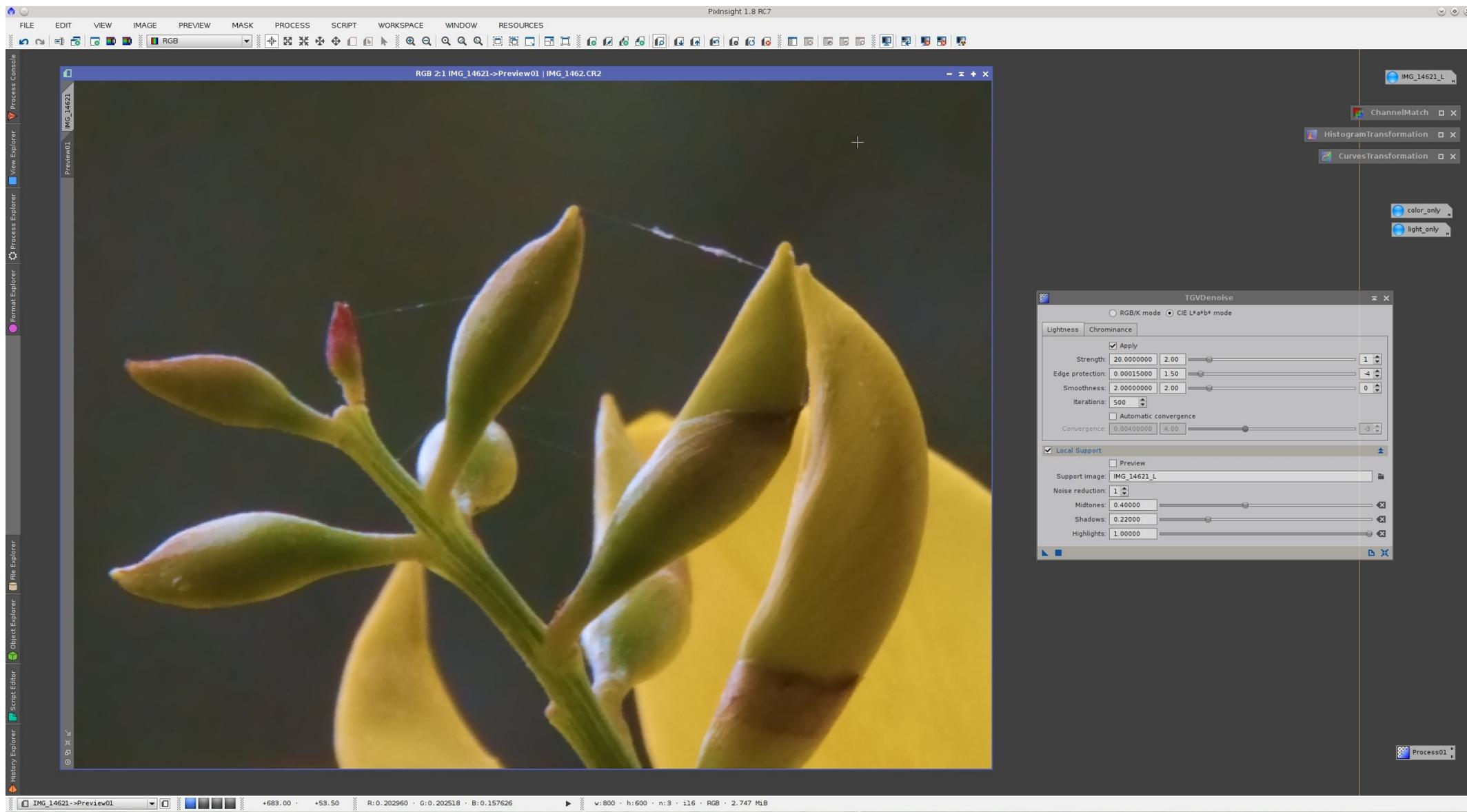


Imagen procesada

# Color (CIE Lab) + LS



Original

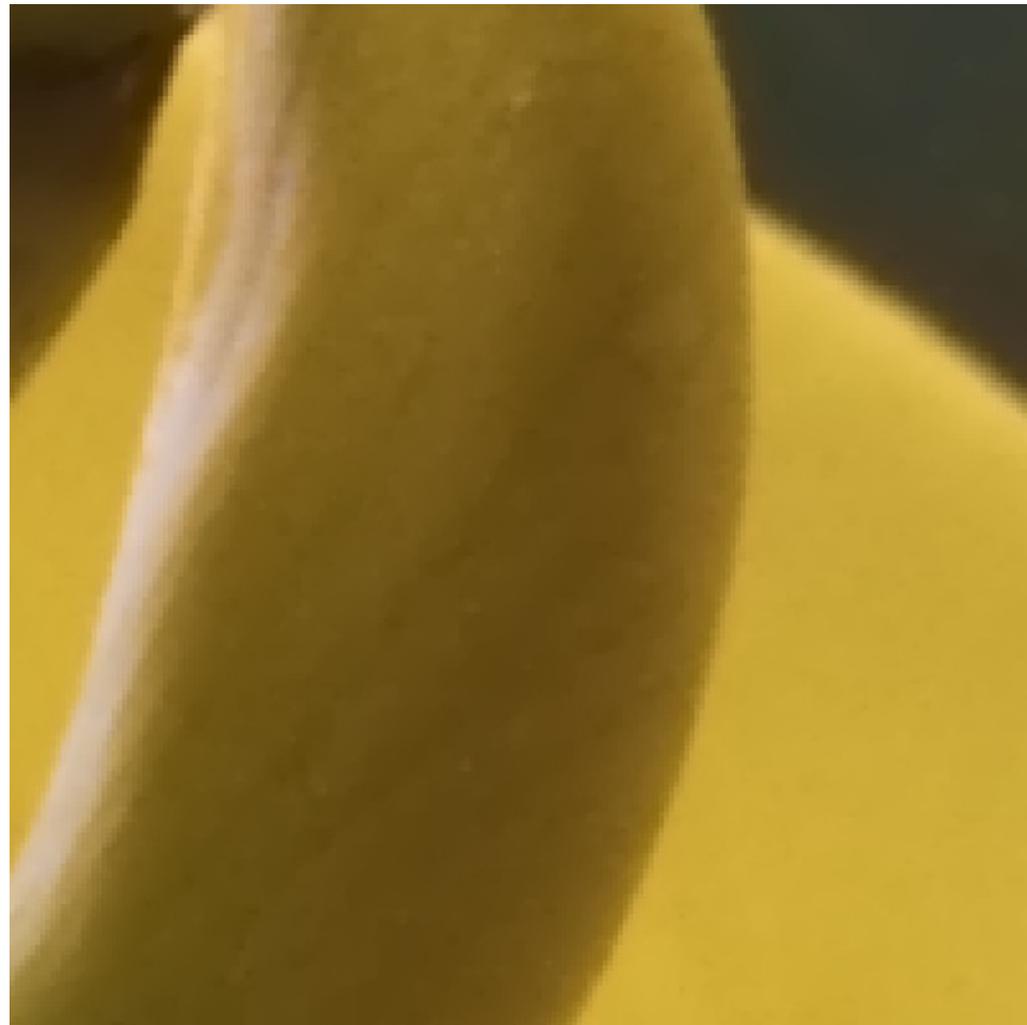


Resultado TGVDenoise

# Color (CIE Lab) + LS

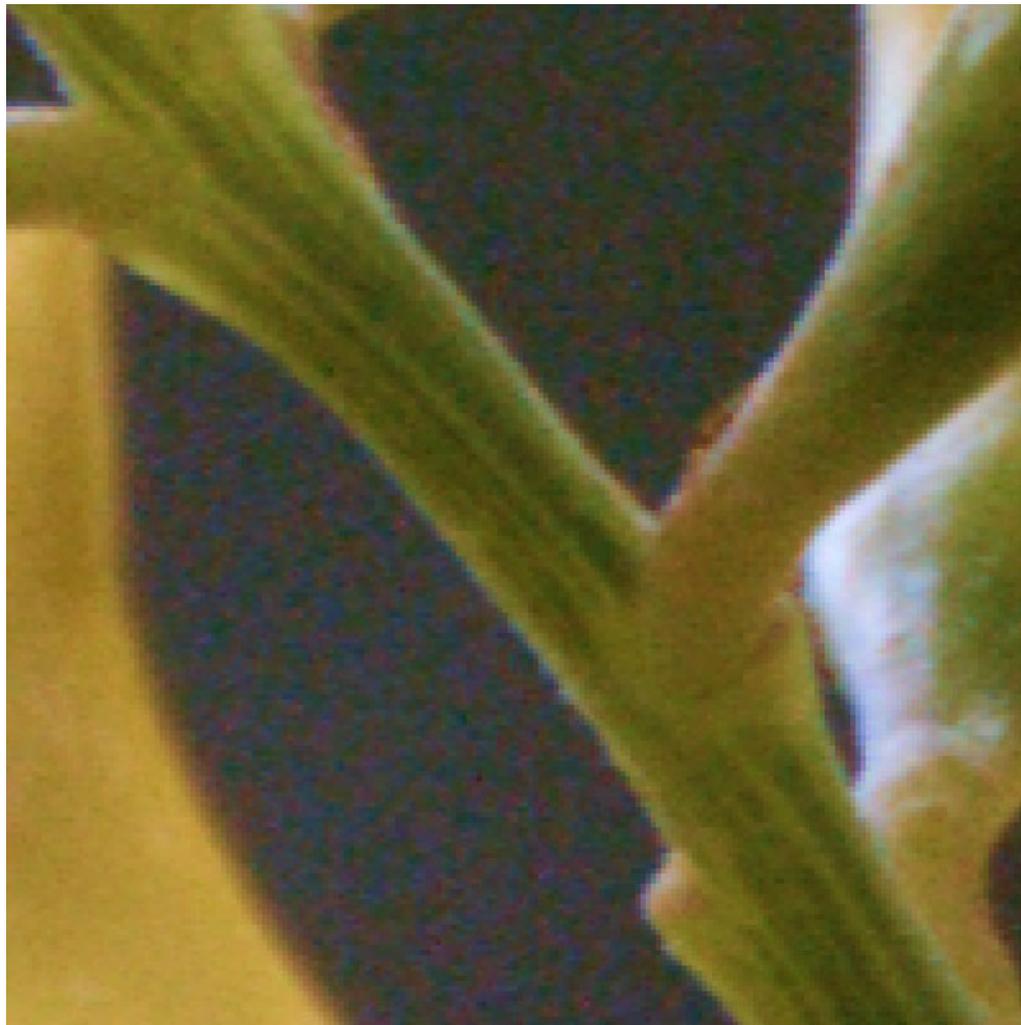


Original

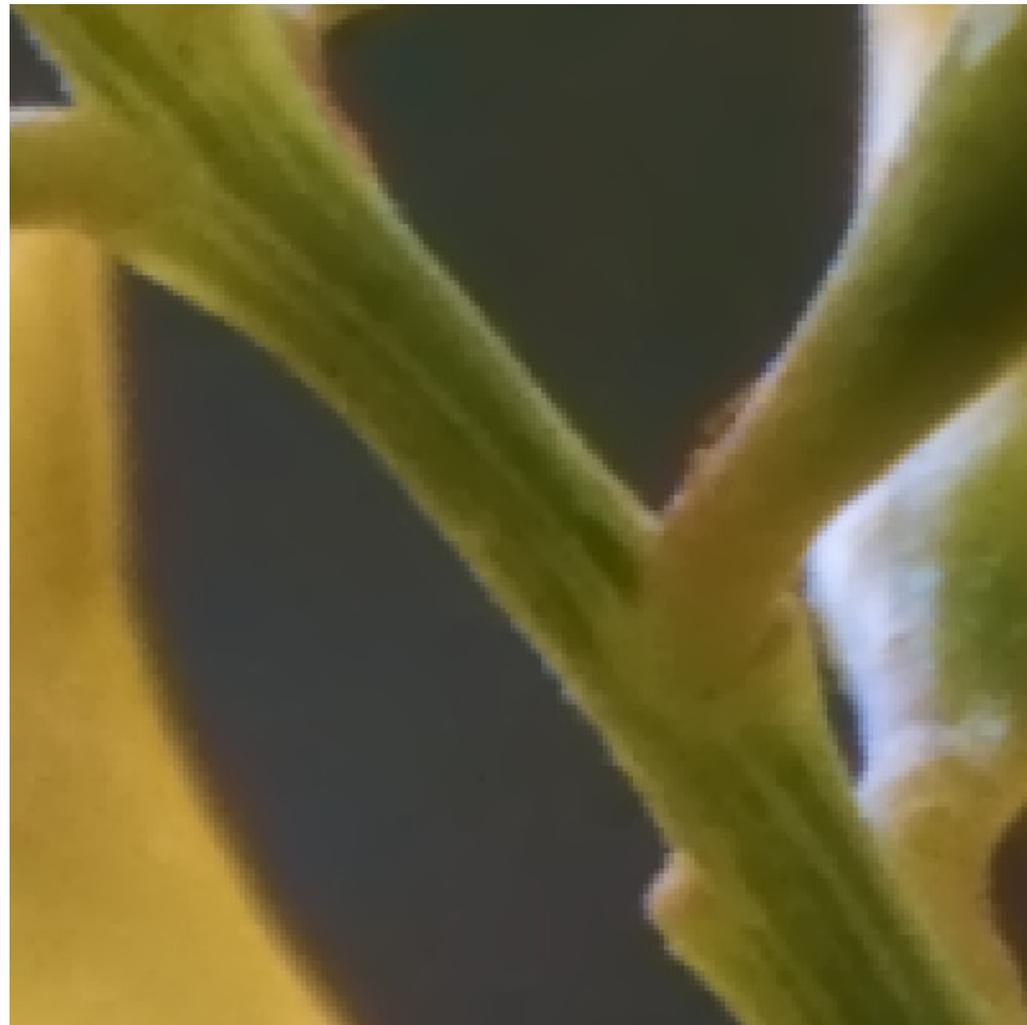


Resultado TGVDnoise

# Color (CIE Lab) + LS



Original



Resultado TGVDenoise

# En MRI Compressed Sensing

- $K = U \cdot W$

$W$  es la transformada de Fourier (FFT2)

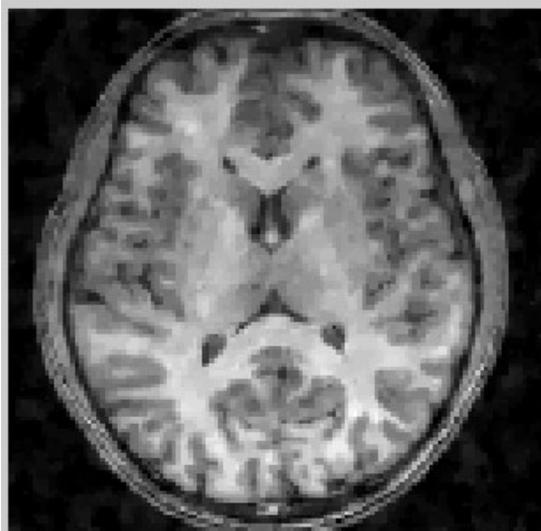
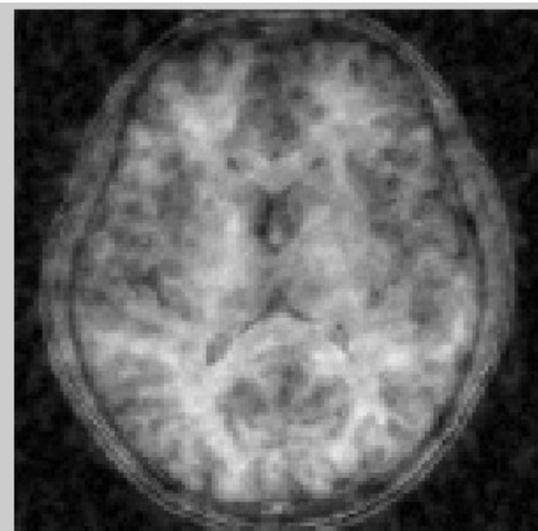
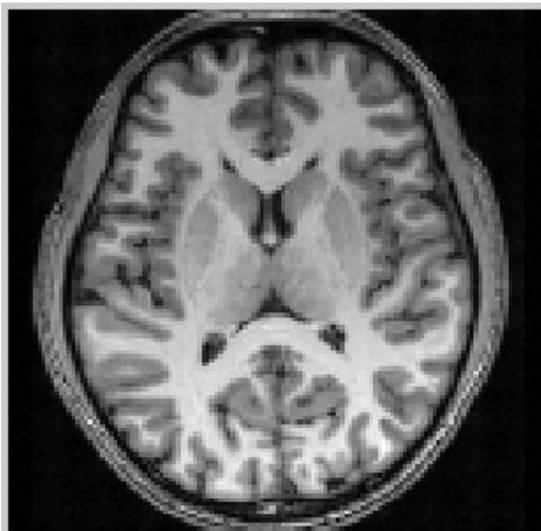
$U$  es el patrón de submuestreo

- $K^* = \text{real}(iW)$

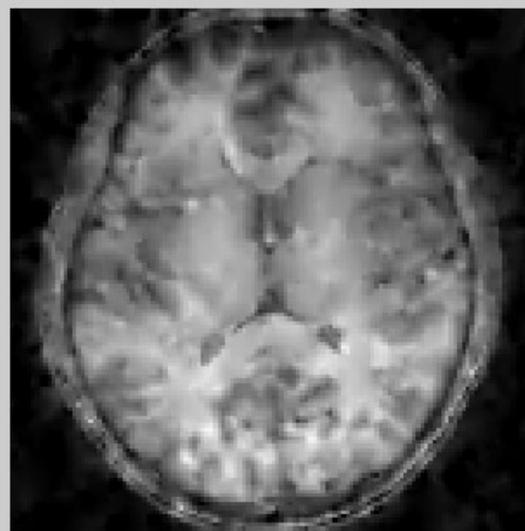
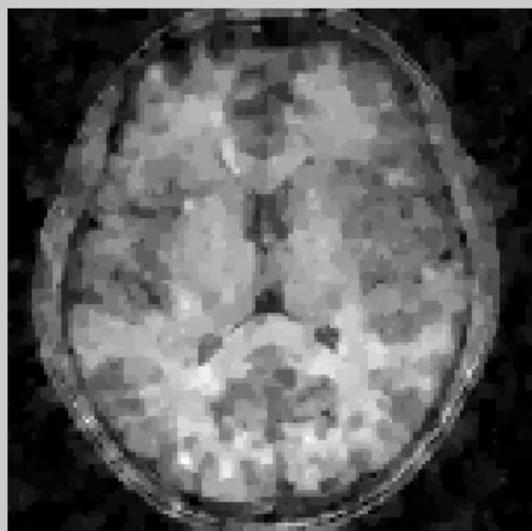
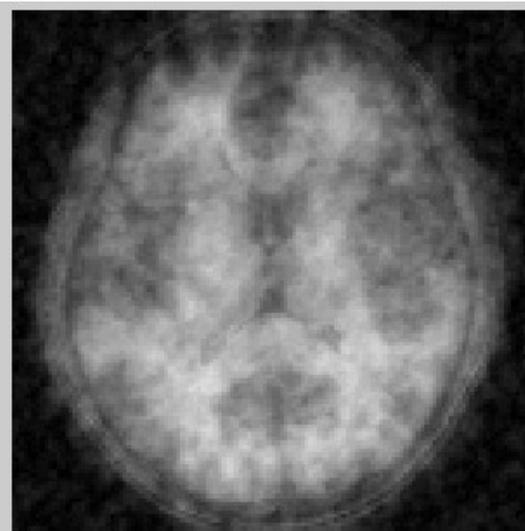
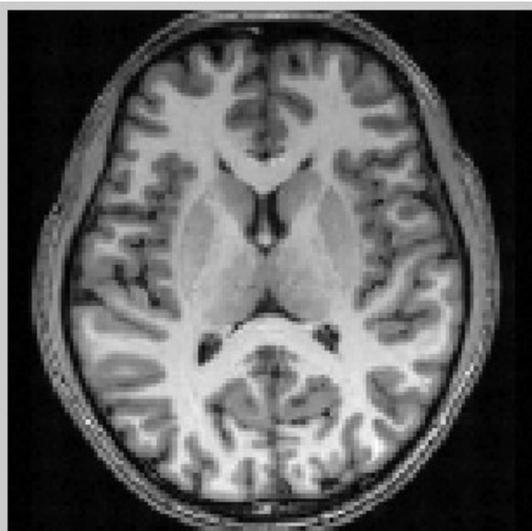
$iW$  es la transformada de Fourier inversa.

Estamos reconstruyendo imagen anatómica. Hay que forzar que sea real.

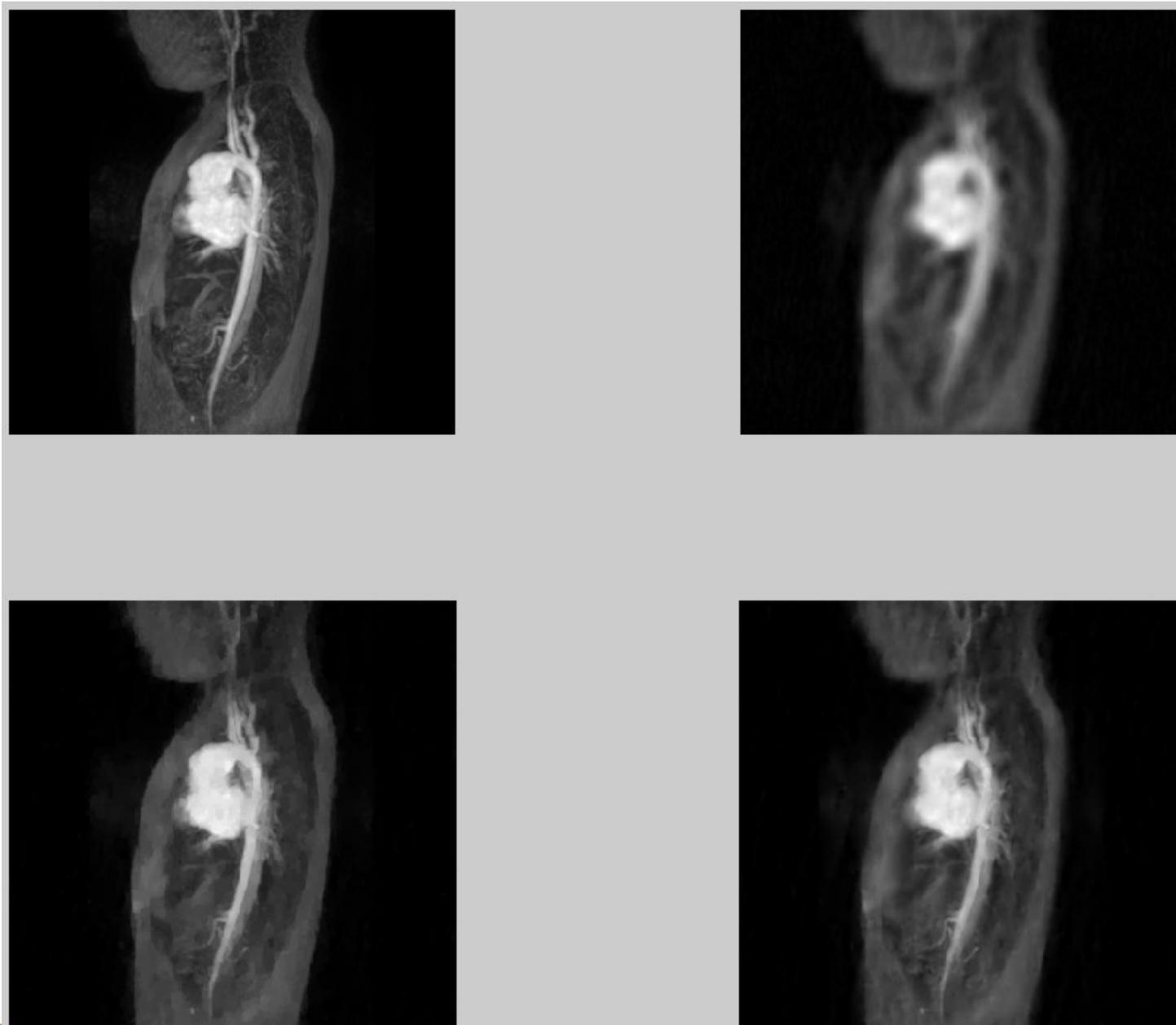
## Cerebro - factor 2



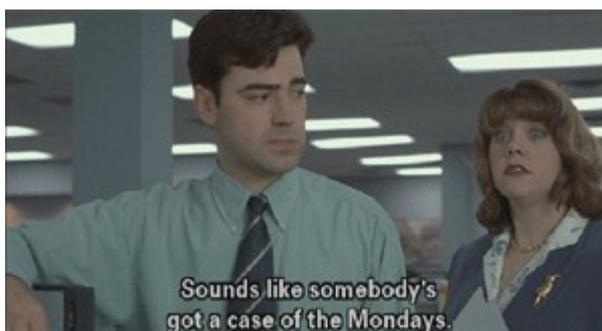
## Cerebro - factor 4



# Angiografía – factor 8

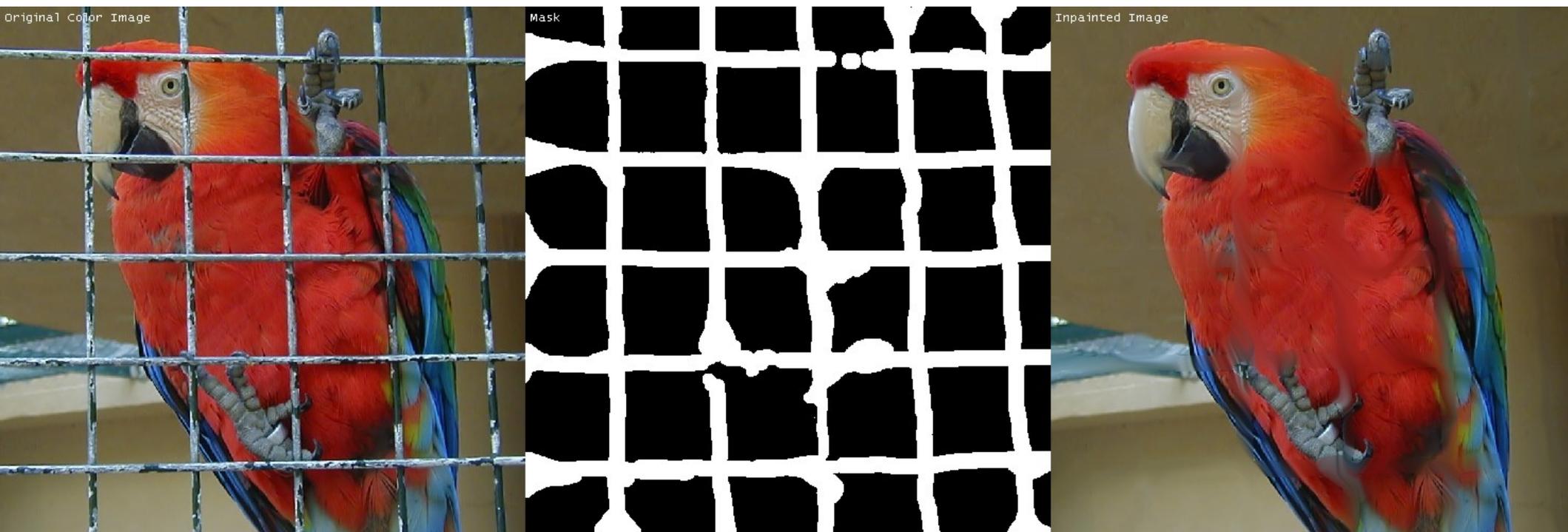


# Inpainting

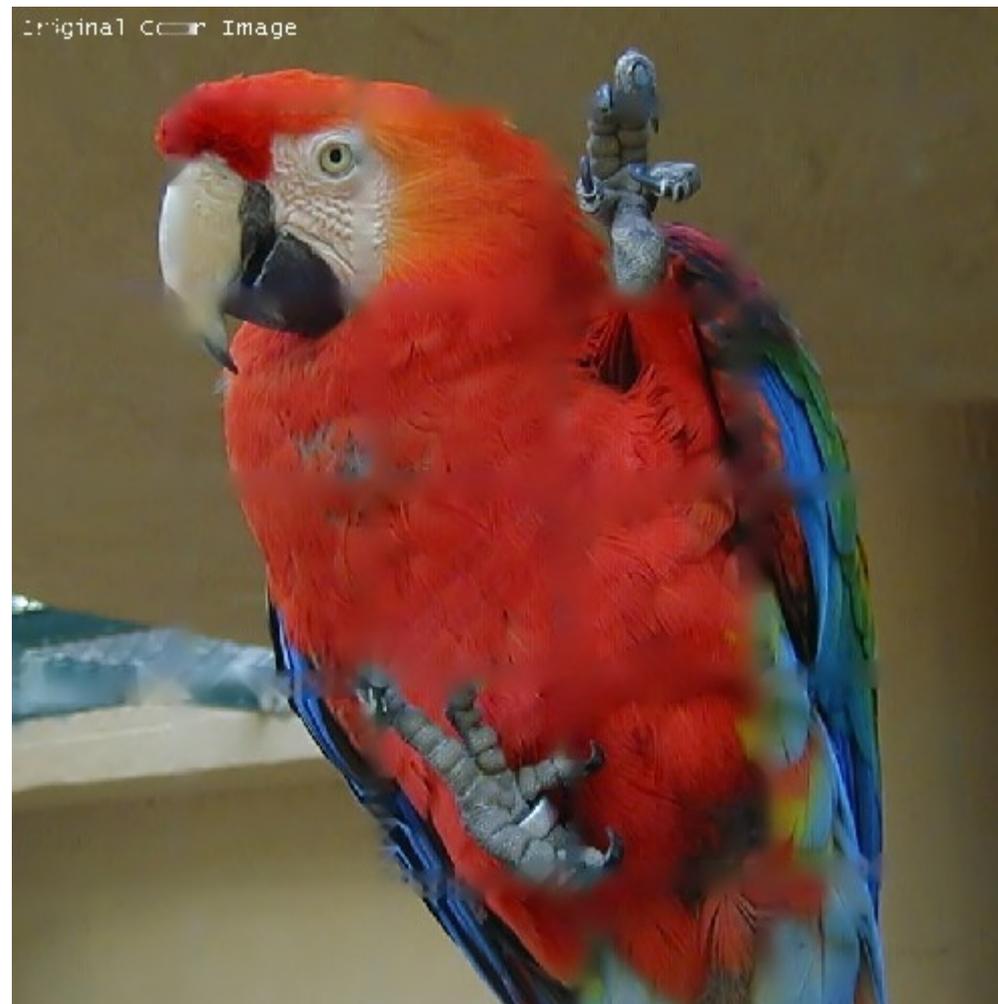


Sounds like somebody's  
got a case of the Mondays.

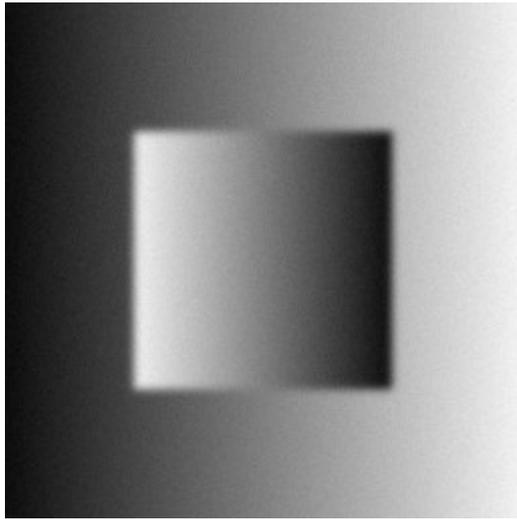
# Inpainting (GREYCstoration)



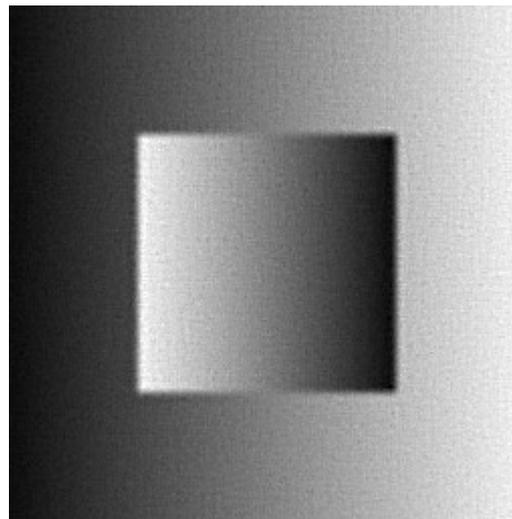
# Inpainting (GREYCstoration vs TGV)



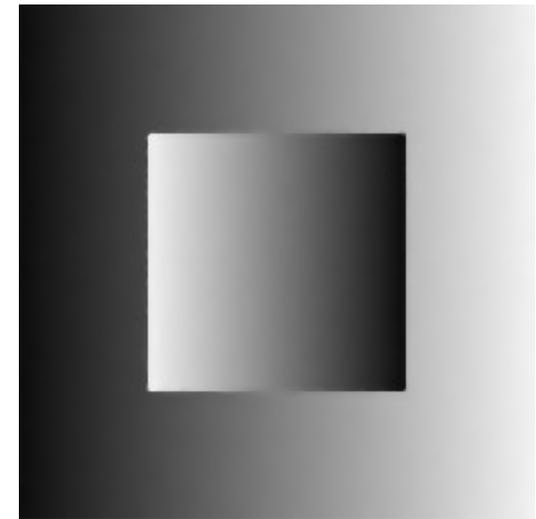
# Deconvoluciones



Desenfoque y ruido gaussianos

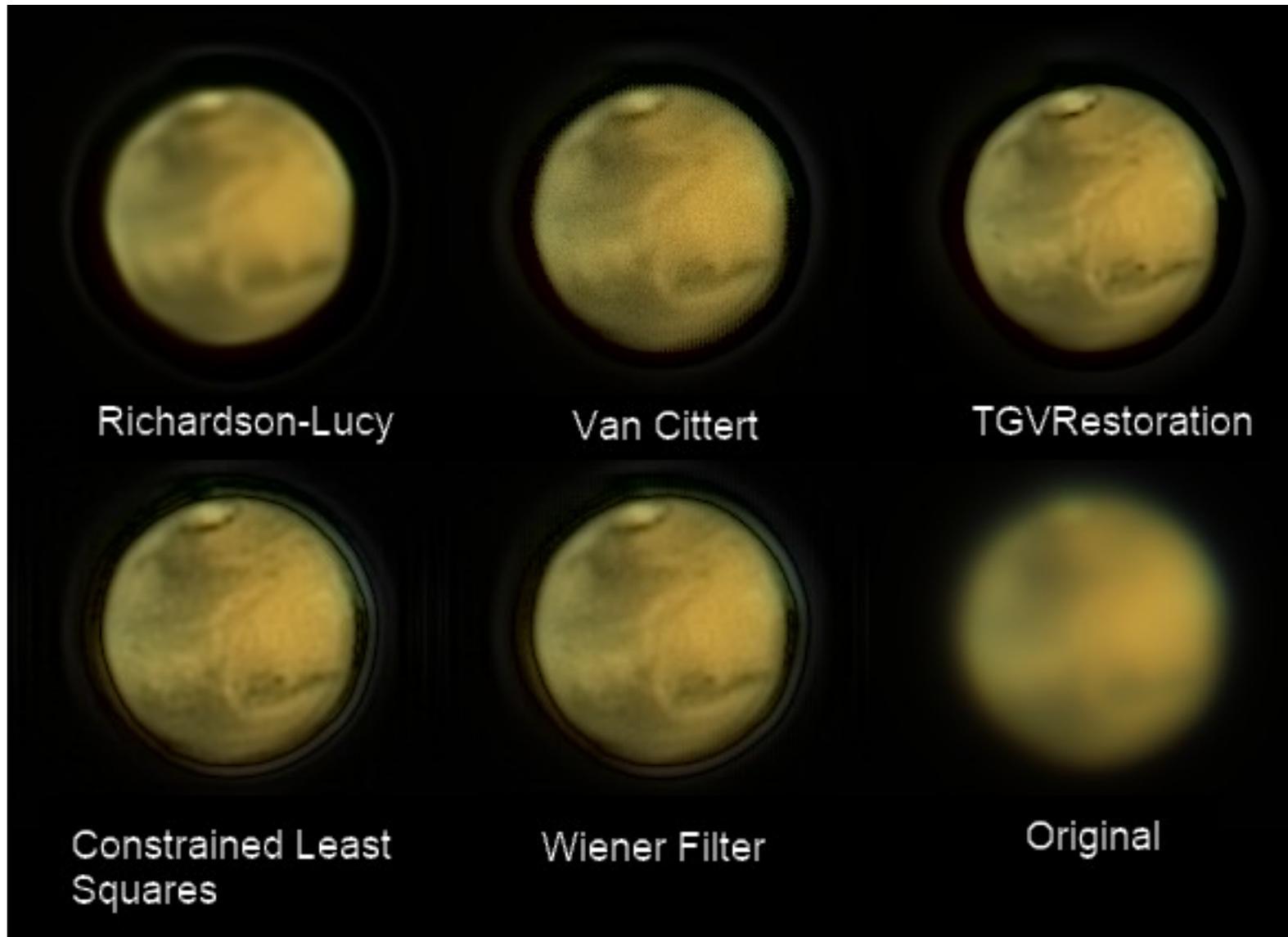


Resultado con VanCittert regularizado con wavelets

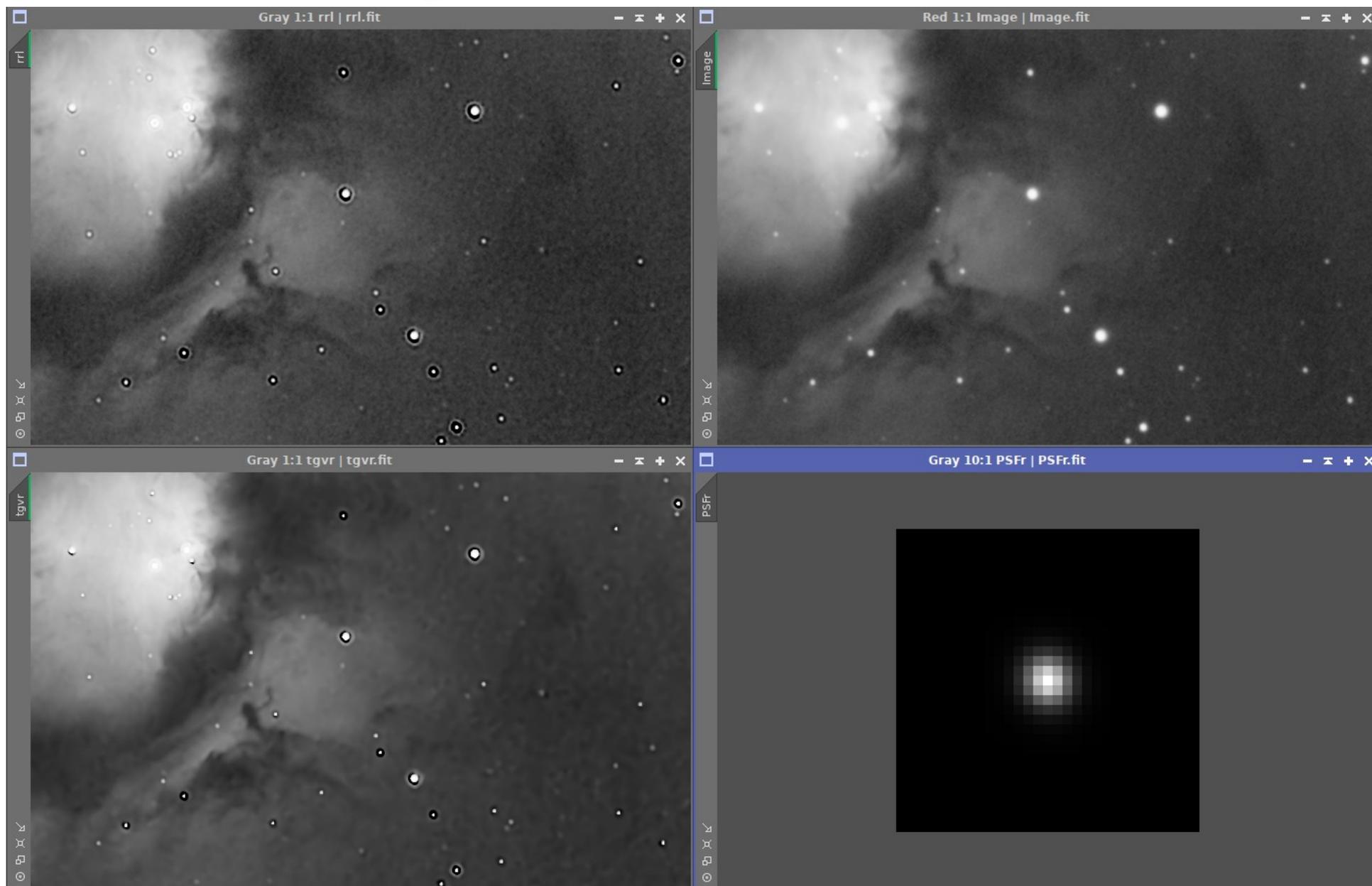


Resultado con TGV

# Marte con telescopio aficionado



# Imagen de Cielo Profundo



# Ringing

Original Signal



Blurred Signal



Unsharp Mask



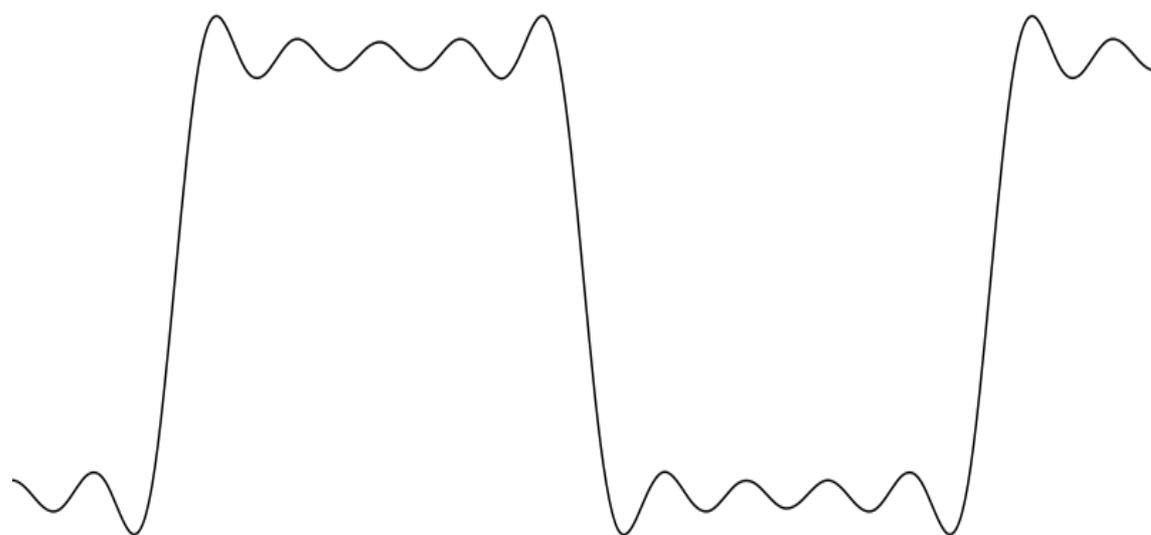
Sharpened Signal



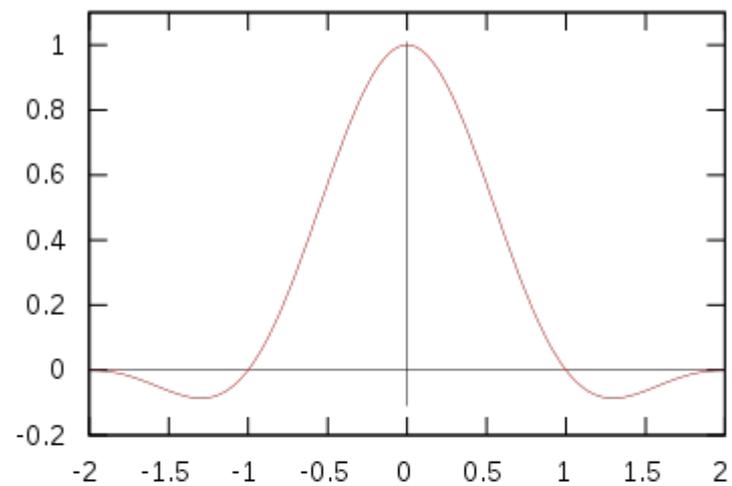
# Ringing



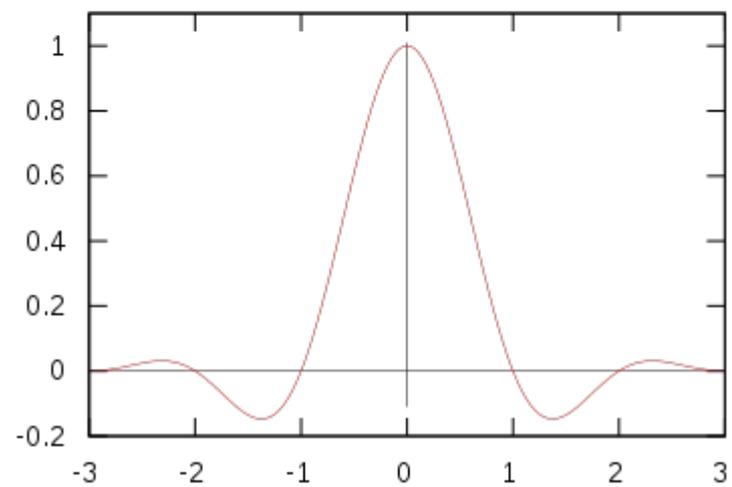
# Ringing



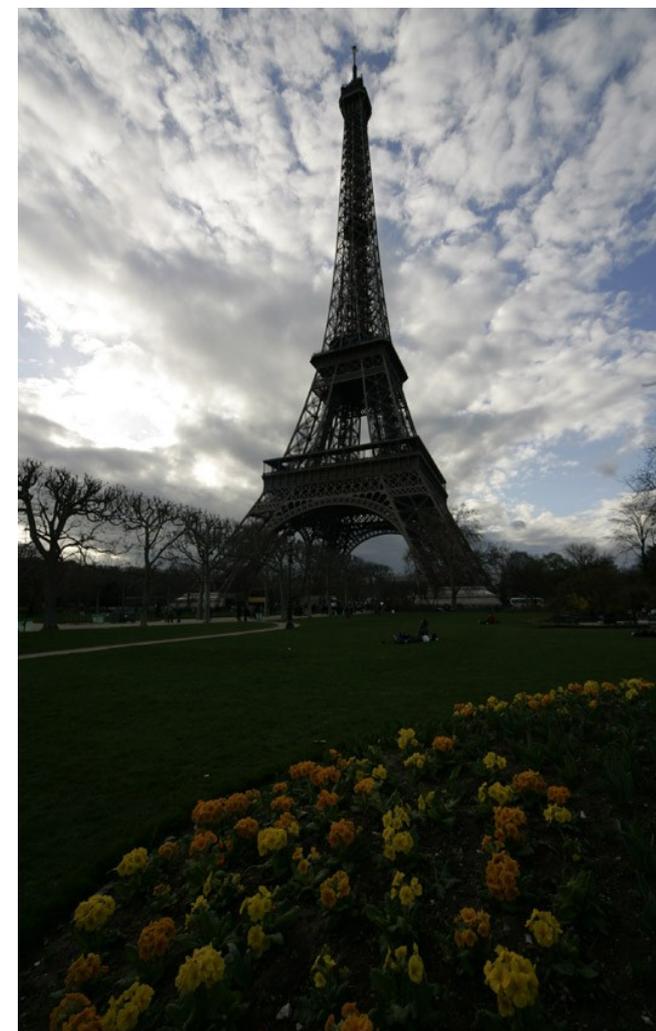
Lanczos kernel for  $a=2$



Lanczos kernel for  $a=3$



# Otro problema inverso: HDR



# Otro problema inverso: HDR



*Ejemplo: secuencia de  
1/1000s a 1/4s*

Consejo:  
**Usar formato RAW**



- Mayor rango dinámico.
- Datos lineales (intensidad proporcional al tiempo de captura).

# Gradient Domain HDR Compression



- A la imagen se le calcula el gradiente.
- Se determina qué bordes atenuar, gracias a un análisis multiescala.
- ***Se recalcula la imagen a partir del gradiente modificado.***

# Gradient Domain HDR Compression

- Cálculo del gradiente multiescala (gaussianas sucesivas, o con wavelets):

$$\nabla H_k = \left( \frac{H_k(x+1, y) - H_k(x-1, y)}{2^{k+1}}, \frac{H_k(x, y+1) - H_k(x, y-1)}{2^{k+1}} \right)$$

- Factores de atenuación dependiente de cada gradiente:

$$\varphi_k(x, y) = \frac{\alpha}{\|\nabla H_k(x, y)\|} \left( \frac{\|\nabla H_k(x, y)\|}{\alpha} \right)^\beta$$

- Acumulación multiplicativa de factores:

$$\Phi_d(x, y) = \varphi_d(x, y)$$

$$\Phi_k(x, y) = L(\Phi_{k+1})(x, y) \varphi_k(x, y)$$

$$\Phi(x, y) = \Phi_0(x, y)$$

# ¿Cómo devolverse al espacio de imagen?

## -> Problema inverso

- No es posible integrar directamente. No es un campo conservativo. Usamos criterio de minimizar el error cuadrático medio para encontrar una solución óptima:

$$\iint F(\nabla I, G) dx dy, \quad F(\nabla I, G) = \|\nabla I - G\|^2 = \left(\frac{\partial I}{\partial x} - G_x\right)^2 + \left(\frac{\partial I}{\partial y} - G_y\right)^2.$$

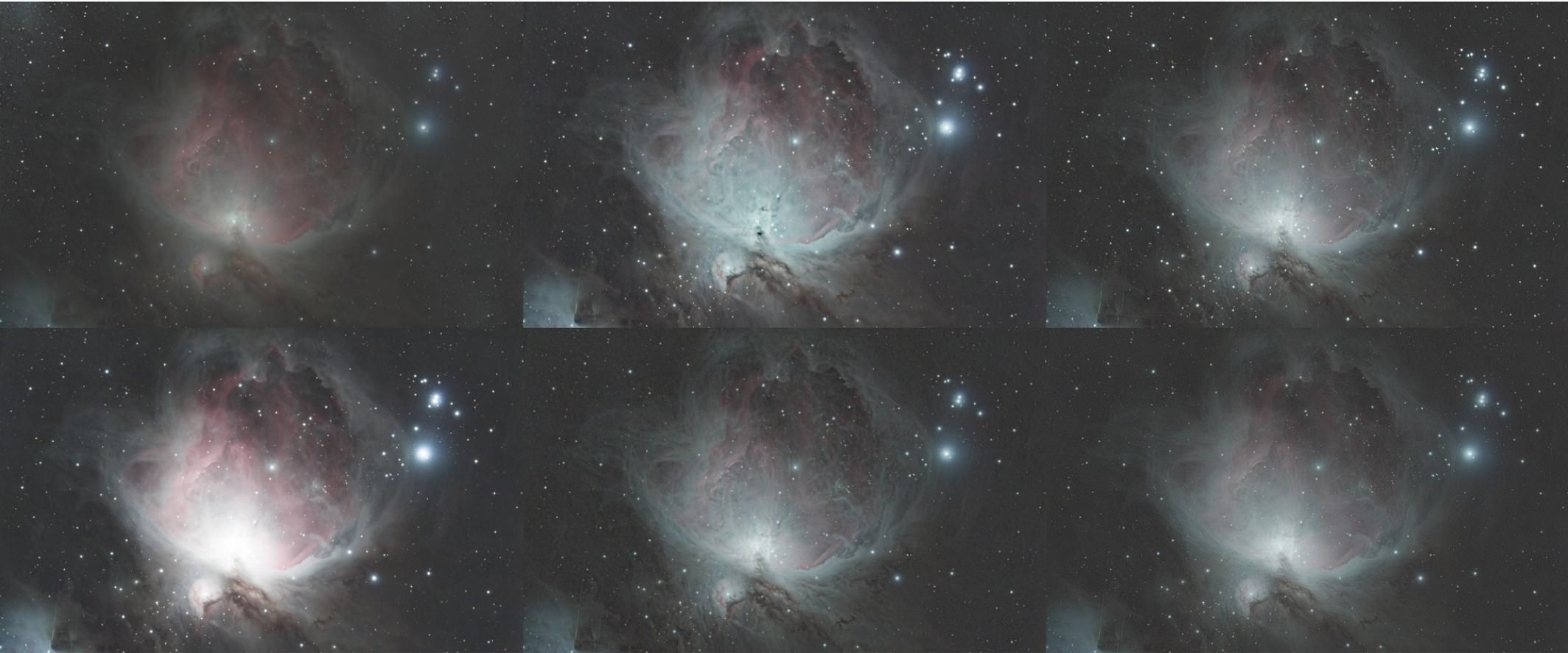
- Esto tiene como solución la ecuación de Poisson:

$$\nabla^2 I = \text{div } G$$

- Que se puede resolver usando varios métodos, como el multigrilla o los solvers rápidos (FFT o DCT).

$$\mathcal{F}_x [f^{(n)}(x)](k) = (2\pi i k)^n \mathcal{F}_x [f(x)](k).$$

# Ejemplos



# Bibliografía

- Total Generalized Variation. K. Bredies, K. Kunisch & T. Pock. 2009
- Second Order Total Generalized Variation (TGV) for MRI. F. Knoll, K. Bredies, T. Poll & R. Stollberger. 2010
- **Recovering piecewise smooth multichannel images by minimization of convex functionals with total generalized variation penalty. K. Bredies. Apr 2012**
- *Spatially dependent regularization parameter selection in total generalized variation models for image restoration. K. Bredies, Y. Dong & M. Hintermüller. Feb 2012*

Fin.