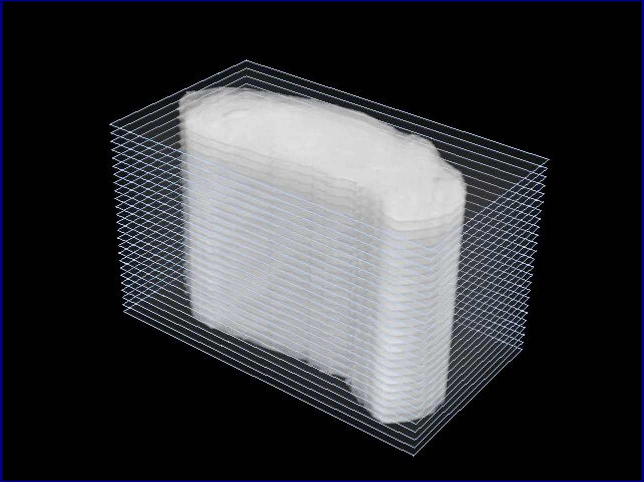
Replenishing the sparse data cubes from the near infrared spectral imager Hyperscout-H of the Hera mission

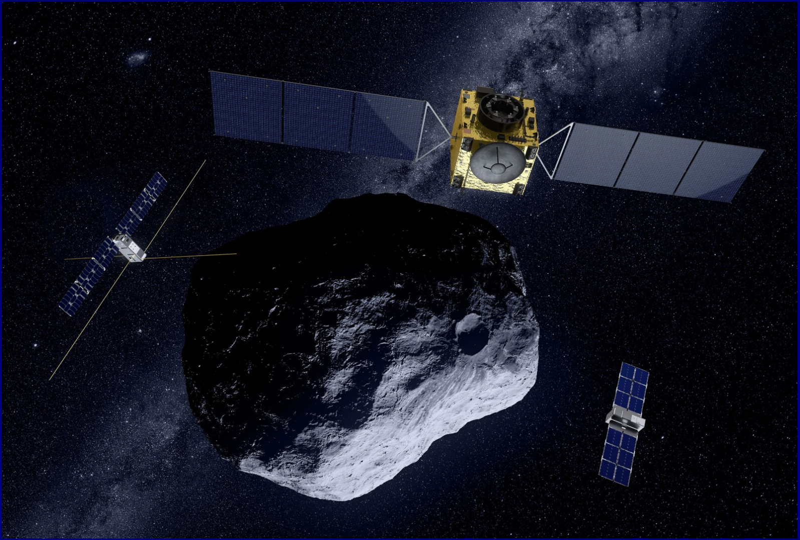
B. Grieger, J. de León, H. Goldberg, T. Kohout, G. Kovács, M. Küppers, B. V. Nagy, M. Popescu



Double Asteroid Redirection Test (DART)

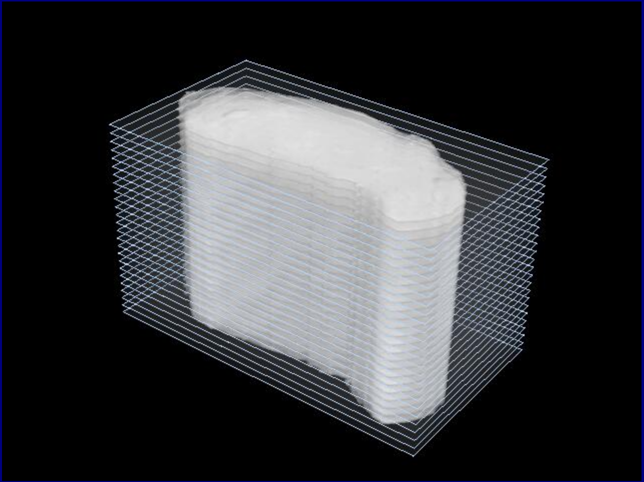
Impacted Dimorphos on 26th September 2022

Hera



Will inspect the Didymos system from late 2026.

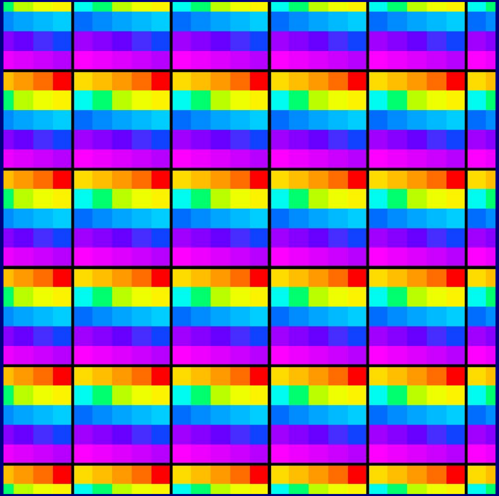
Hyperscout-H hyperspectral imager



25 at one stroke — 25 images at 25 different wavelengths

Well, not quite . . .

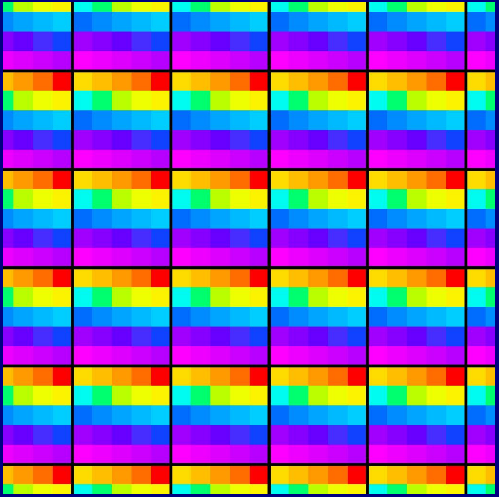
Macro pixels



- The sensor has 2048 $\times$ 1088 pixels.

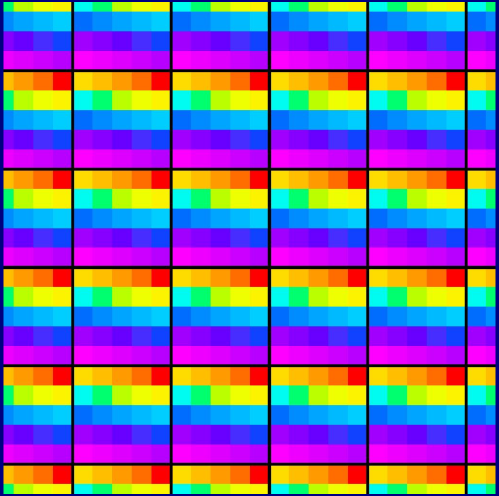
B. Grieger et al.

Macro pixels



- The sensor has 2048 × 1088 pixels.
- These are grouped into macro pixels of 5 × 5 original pixels,

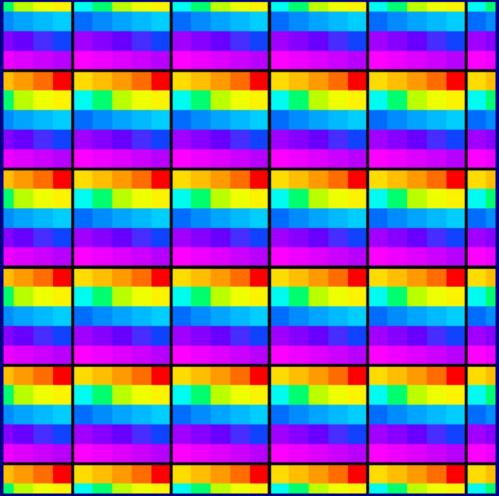
Macro pixels



- The sensor has 2048 × 1088 pixels.
- These are grouped into macro pixels of 5 × 5 original pixels,

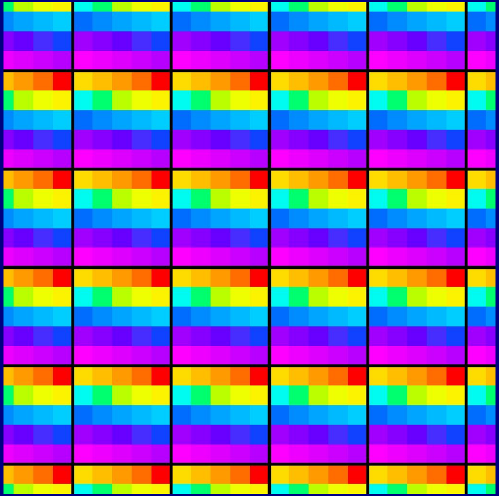
  which have 25 different center wavelengths from 657 to 949 nm

Macro pixels



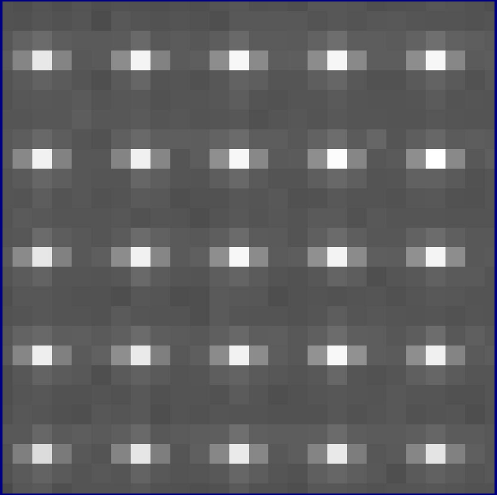
- The sensor has 2048×1088 pixels.
- These are grouped into macro pixels of 5×5 original pixels, which have 25 different center wavelengths from 657 to 949 nm dubbed F1–F25.

Macro pixels



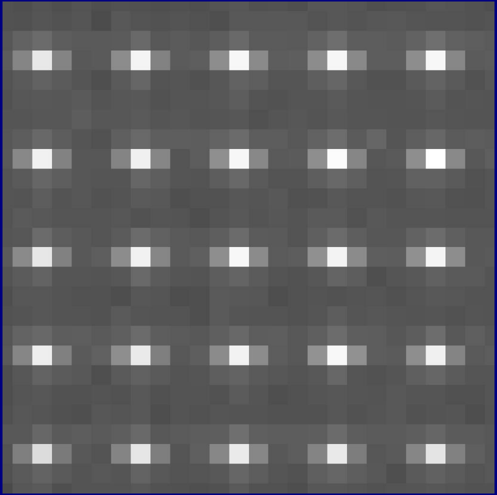
- The sensor has 2048×1088 pixels.
- These are grouped into macro pixels of 5×5 original pixels, which have 25 different center wavelengths from 657 to 949 nm dubbed F1–F25.
- Only $\frac{1}{25}$ of the full $2048 \times 1088 \times 25$ cube pixels are populated, i. e., 4 %.

Monochromator image (25×25 pixels cut out)



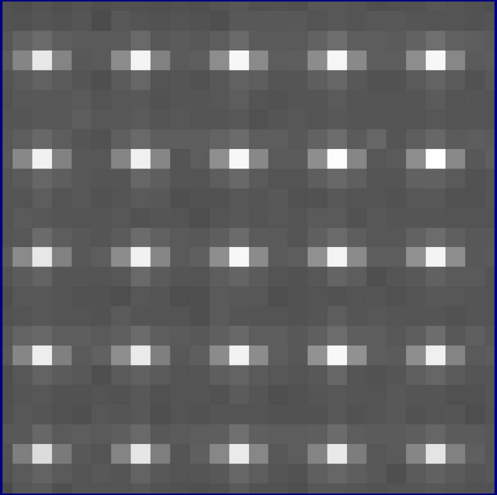
- The monochromator emits light at 700 nm (and – almost – nothing else).

Monochromator image (25×25 pixels cut out)



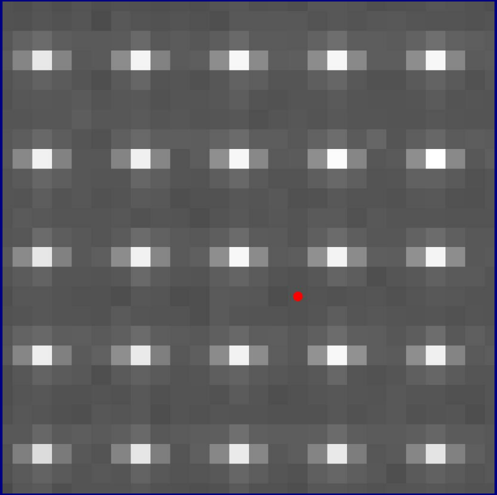
- The monochromator emits light at 700 nm (and – almost – nothing else).
- This wavelength is picked up by F4

Monochromator image (25×25 pixels cut out)



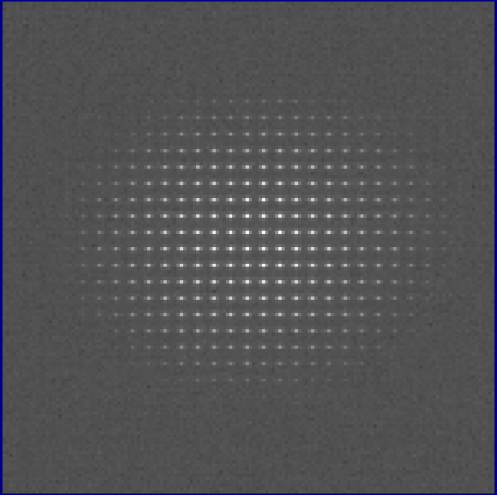
- The monochromator emits light at 700 nm (and – almost – nothing else).
- This wavelength is picked up by F4 and somewhat by its neighbors F3 and F5.

Monochromator image (25×25 pixels cut out)



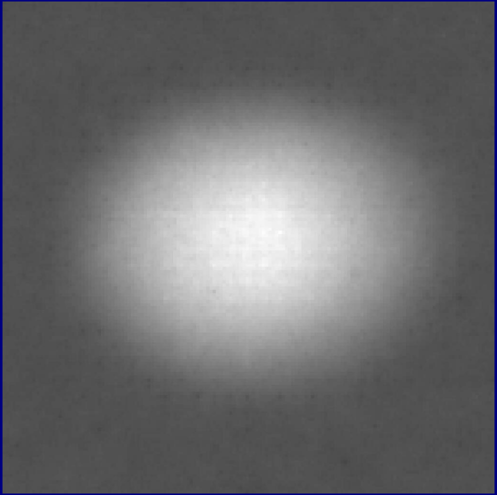
- The monochromator emits light at 700 nm (and – almost – nothing else).
- This wavelength is picked up by F4 and somewhat by its neighbors F3 and F5.
- For later reference, ● marks F17.

Monochromator image (150×150 pixels)



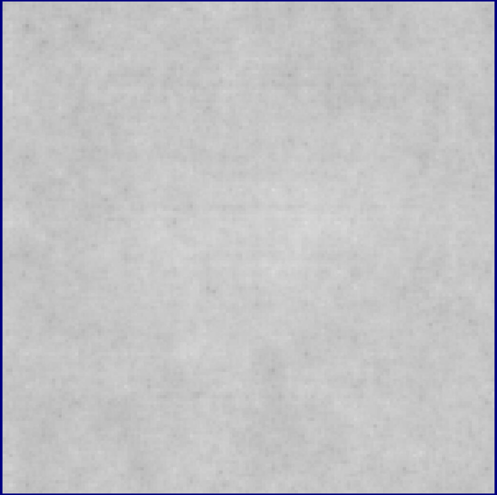
- Raw image

Monochromator image (150×150 pixels)



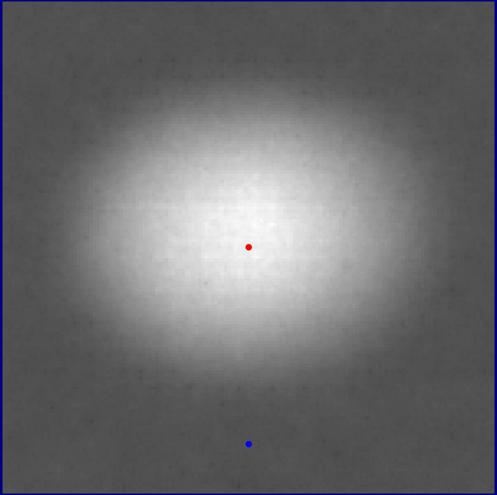
- F4 replenished image

Monochromator image (150×150 pixels)



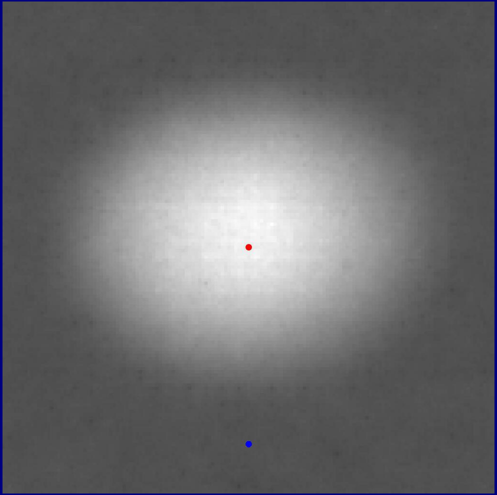
- F17 replenished image

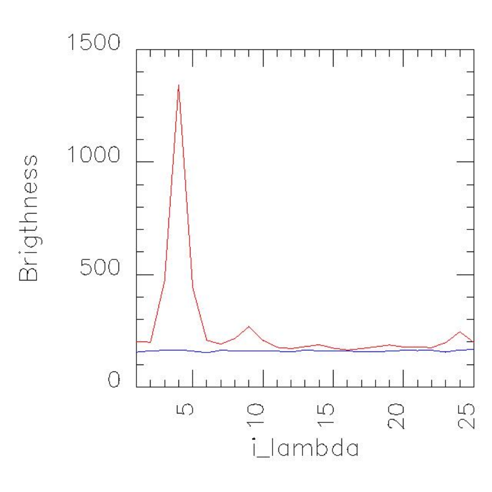
Monochromator image and spectra



- F4 replenished image
- We look at spectra from the replenished cube at ● and ○,
- which are in F17 pixels.

Monochromator image and spectra

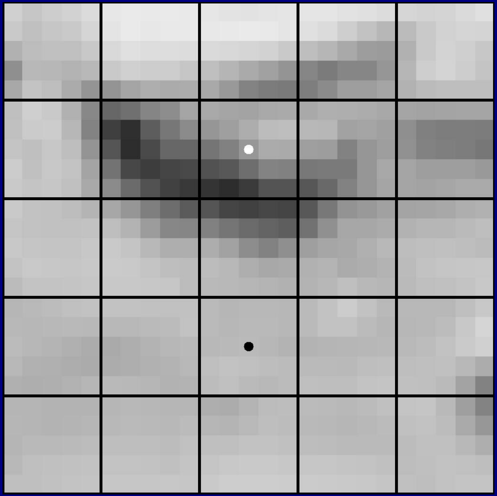




Simulated cube (F1, 450×450 pixels cut out)

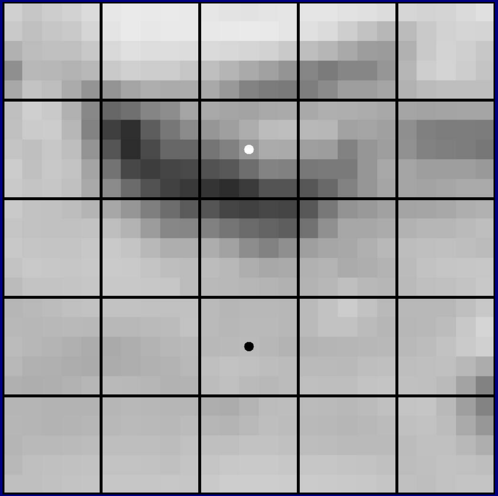
- Easy case: spectrum is the same everywhere.
- But there are brightness variations, mostly due to shading.

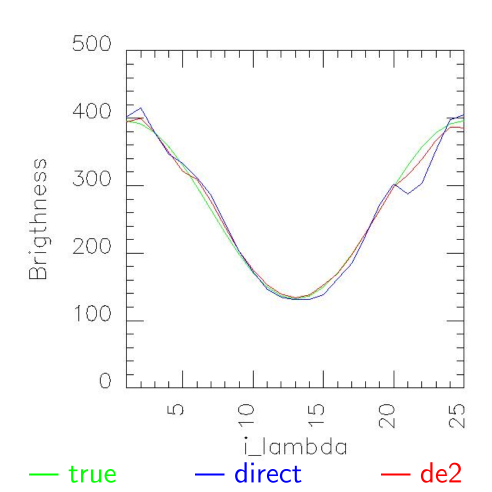
Simulated cube (F1, 25×25 pixels cut out)



- Easy case: spectrum is the same everywhere.
- But there are brightness variations, mostly due to shading.

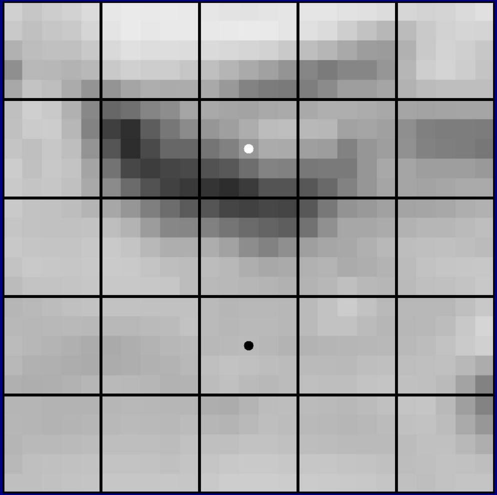
Simulated cube (F1, 25×25 pixels cut out)

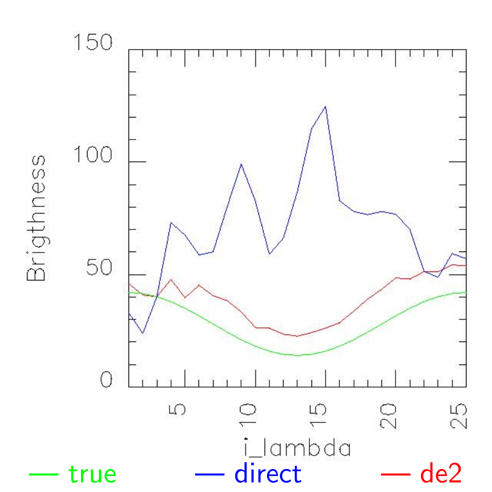




- smooth brightness

Simulated cube (F1, 25×25 pixels cut out)





- uneven brightness

The de2 approach 

We only have a single measurement $f'(i,j,k_{i,j})$ at each spatial pixel (i,j).

- Separate the normalized spectrum $\hat{f}(i,j,k)$ and the brightness scaling factor $b(i,j)$, so that the retrieved spectrum is $f(i,j,k) = b(i,j)\,\hat{f}(i,j,k)$.
- We use only *ratios* of measured values from two adjacent pixels, e. g., $\frac{f'(i,j,k_{i,j})}{f'(i,j+1,k_{i,j+1})}$.
- This allows to compute $\hat{f}(i,j,k_{i,j})$ from its nearest neighbors:

$$\hat{f}(i,j,k_{i,j}) = \frac{1}{4}\,\hat{f}(i,j+1,k_{i,j+1})\,\frac{f'(i,j,k_{i,j})}{f'(i,j+1,k_{i,j+1})} + \cdots$$

- To compute $\hat{f}(i,j,k)$ for $k \neq k_{i,j}$, we assume that $\hat{f}$ is spatially smooth:

$$\hat{f}(i,j,k_{i,j}) = \frac{1}{4}\,\hat{f}(i,j+1,k_{i,j}) + \cdots$$

- When $\hat{f}$ has converged, we set $b(i,j) = \frac{f'(i,j,k_{i,j})}{\hat{f}(i,j,k_{i,j})}$

The de2 approach 

We only have a single measurement $f'(i,j,k_{i,j})$ at each spatial pixel (i,j).

- Separate the normalized spectrum $\hat{f}(i,j,k)$ and the brightness scaling factor $b(i,j)$, so that the retrieved spectrum is $f(i,j,k) = b(i,j)\,\hat{f}(i,j,k)$.
- We use only *ratios* of measured values from two adjacent pixels, e. g., $\frac{f'(i,j,k_{i,j})}{f'(i,j+1,k_{i,j+1})}$.
- This allows to compute $\hat{f}(i,j,k_{i,j})$ from its nearest neighbors:

$$\hat{f}(i,j,k_{i,j}) = \frac{1}{4}\,\hat{f}(i,j+1,k_{i,j+1})\,\frac{f'(i,j,k_{i,j})}{f'(i,j+1,k_{i,j+1})} + \cdots$$

- To compute $\hat{f}(i,j,k)$ for $k \neq k_{i,j}$, we assume that $\hat{f}$ is spatially smooth:

$$\hat{f}(i,j,k_{i,j}) = \frac{1}{4}\,\hat{f}(i,j+1,k_{i,j}) + \cdots$$

- When $\hat{f}$ has converged, we set $b(i,j) = \frac{f'(i,j,k_{i,j})}{\hat{f}(i,j,k_{i,j})}$

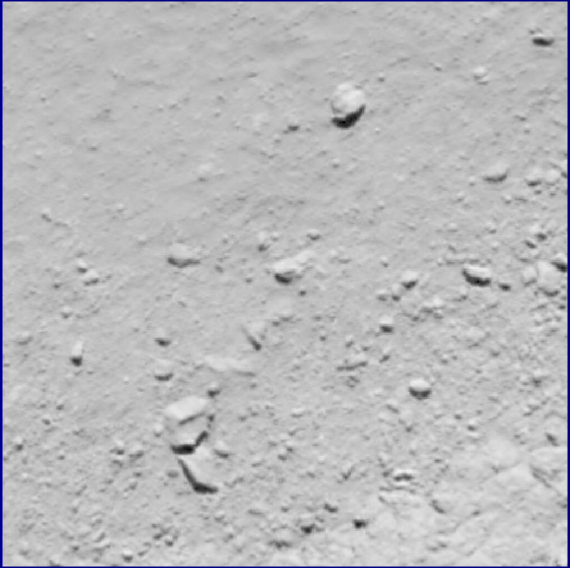
Simulated cube (F1, 450×450), still the easy case

Simulation

Simulated cube (F1, 450×450), still the easy case

Direct replenishing (nearest micro pixel)

Simulated cube (F1, 450×450), still the easy case

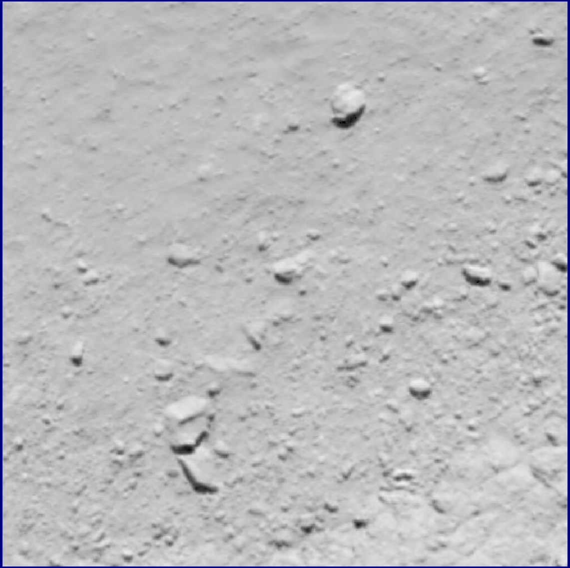


de2 replenishing

Simulated cube (F1, 450×450), still the easy case

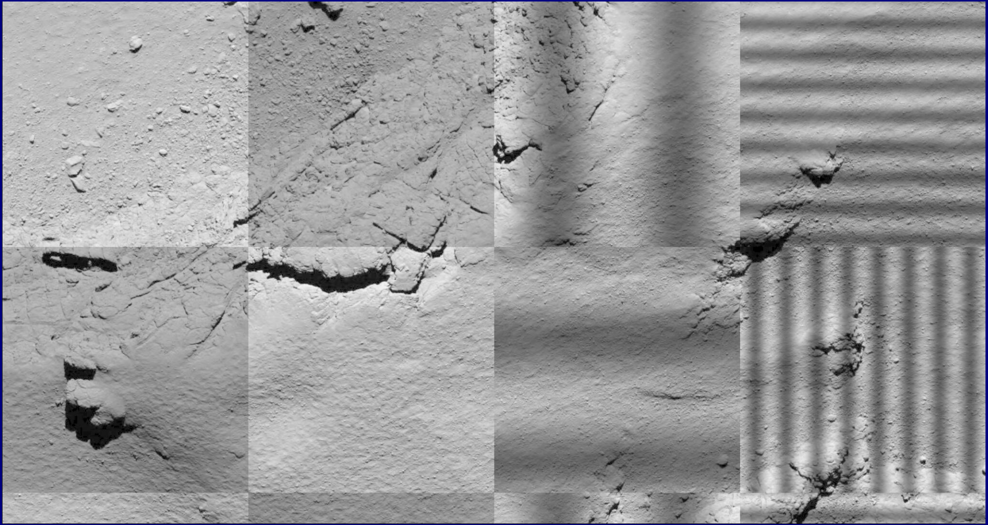
Simulation

Simulated cube (F1, 450×450), still the easy case



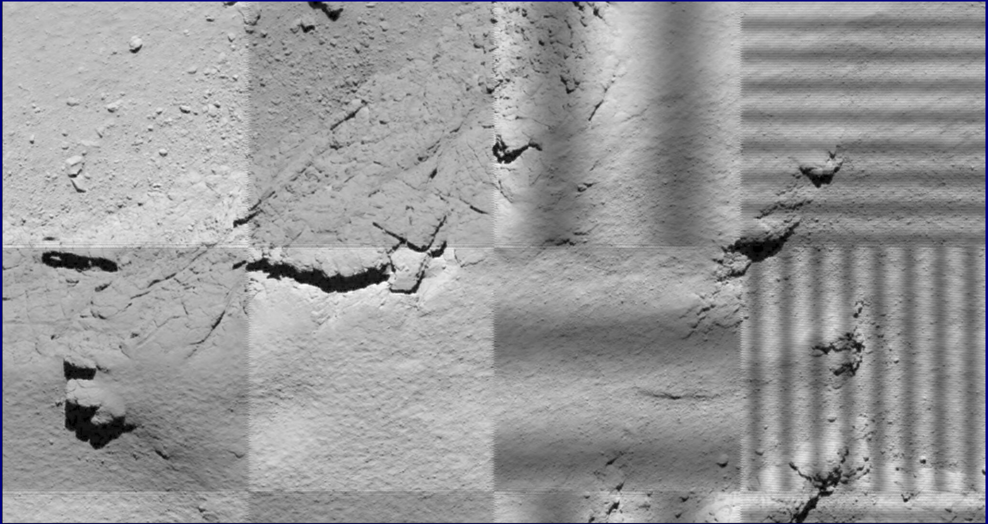
de2 replenishing

Simulated cube (F1), full 2048×1088



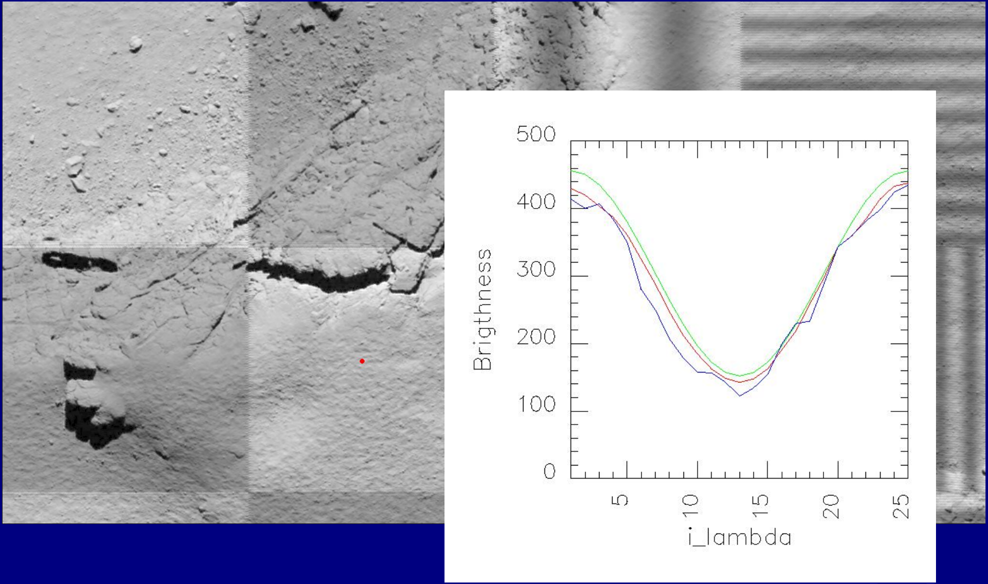
Simulation

Simulated cube (F1), full 2048×1088

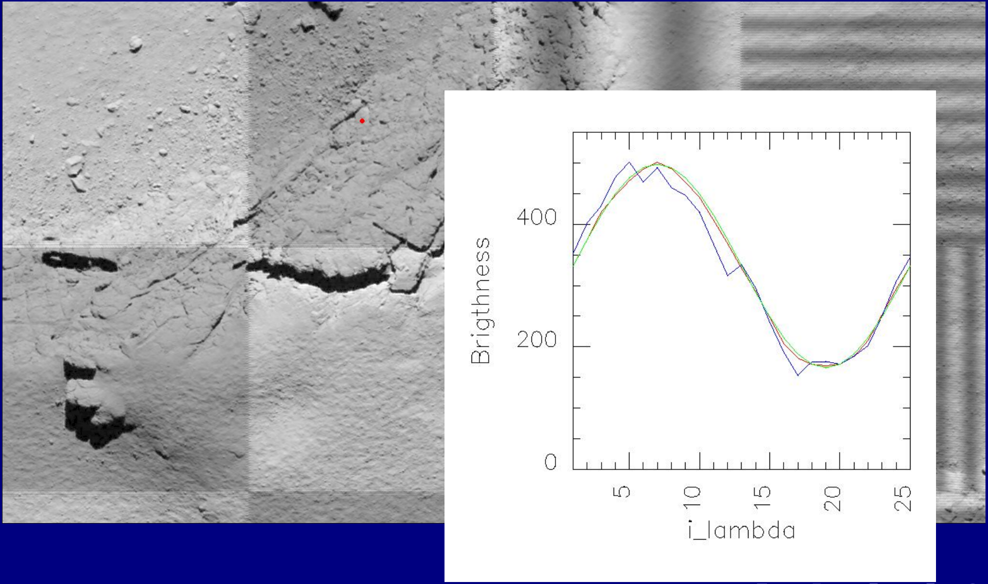


de2 replenishing

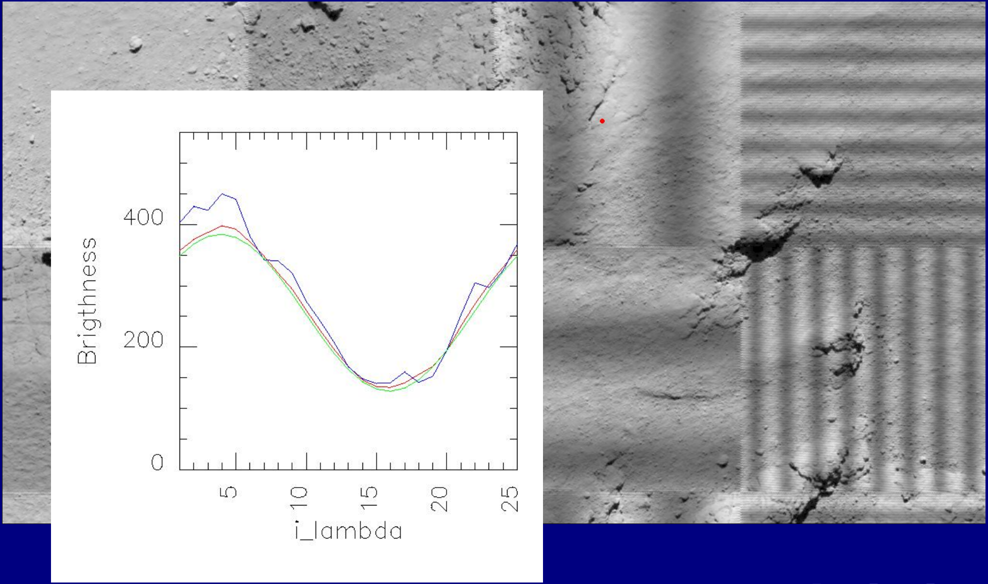
Simulated cube (F1, full 2048×1088)



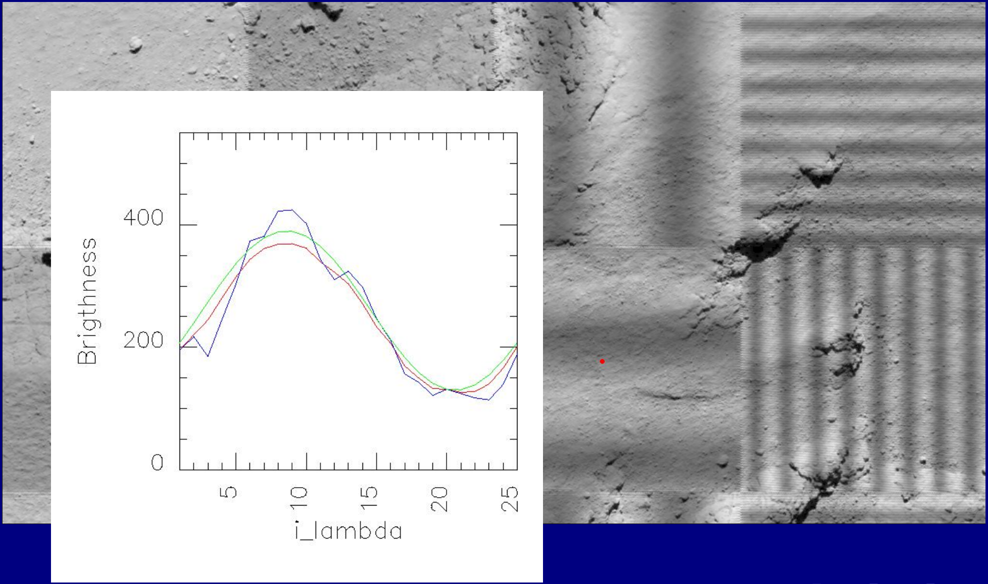
Simulated cube (F1, full 2048×1088)



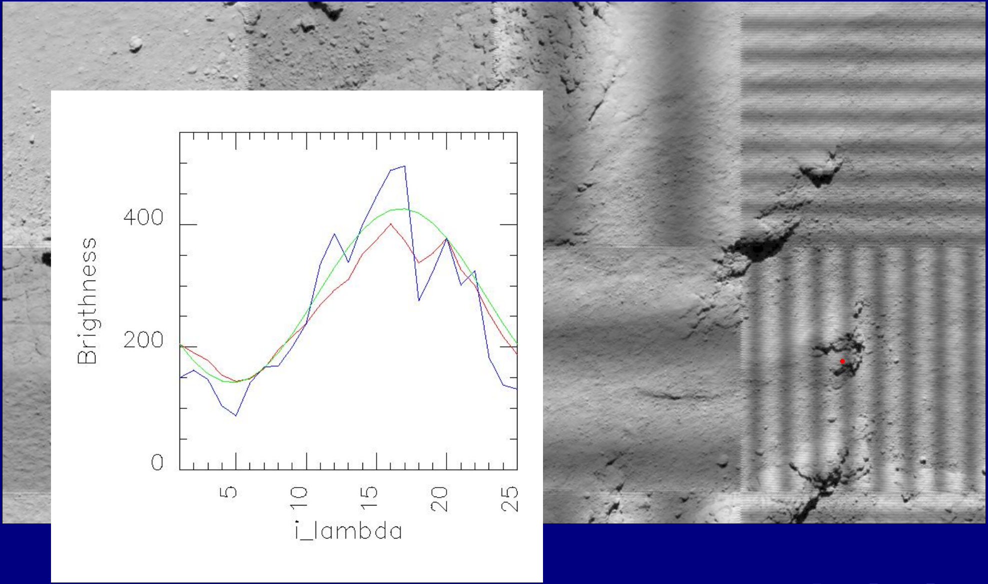
Simulated cube (F1, full 2048×1088)



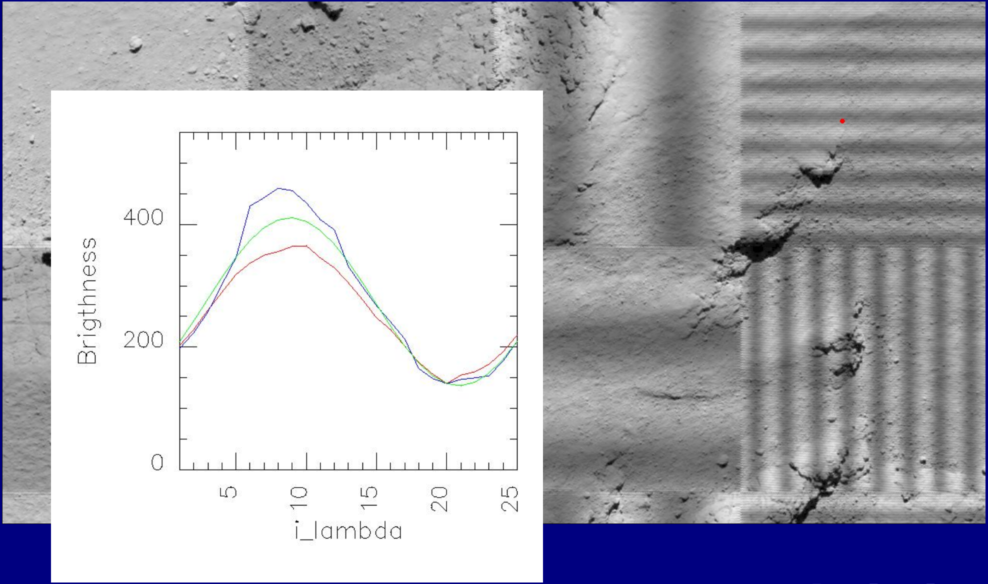
Simulated cube (F1, full 2048×1088)



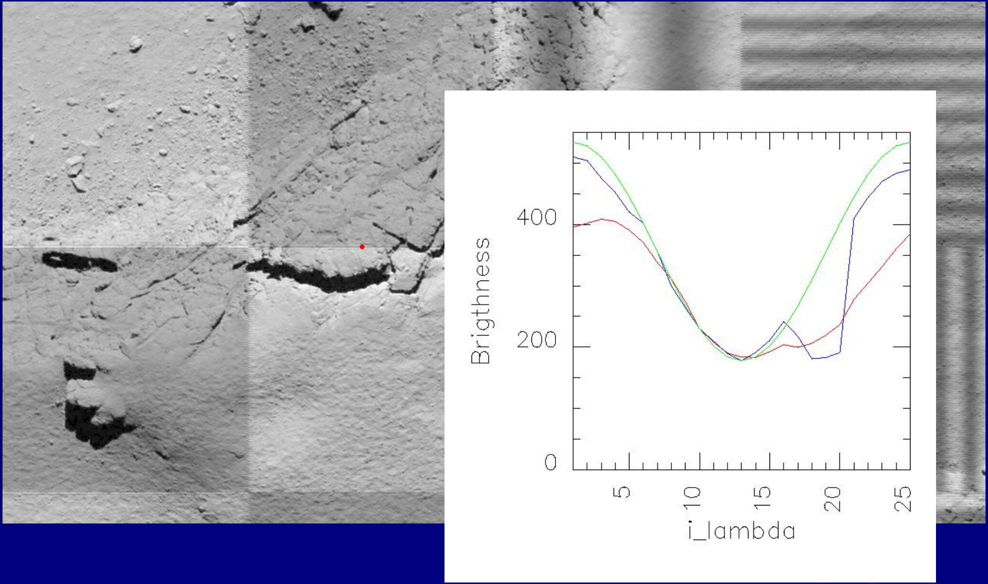
Simulated cube (F1, full 2048×1088)



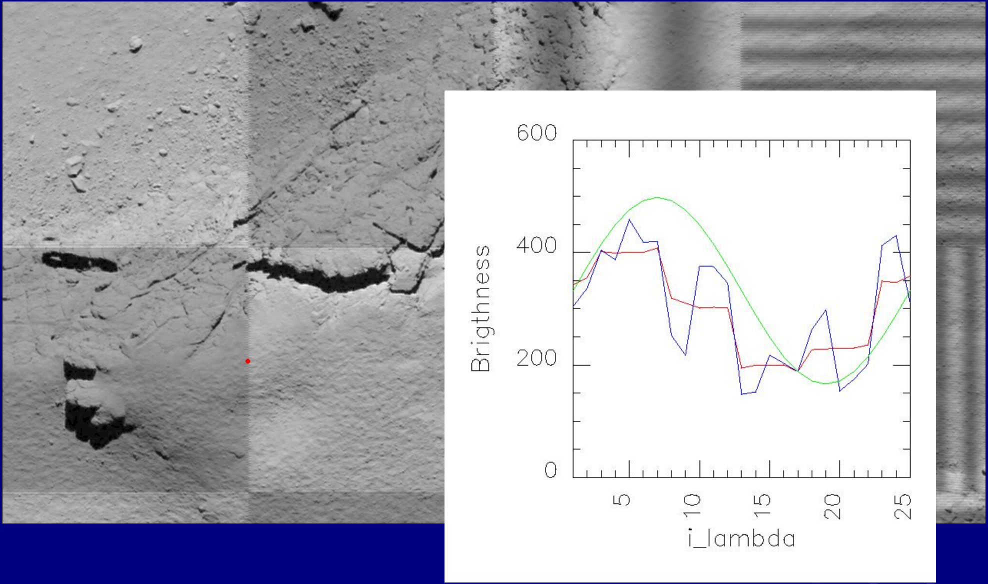
Simulated cube (F1, full 2048×1088)



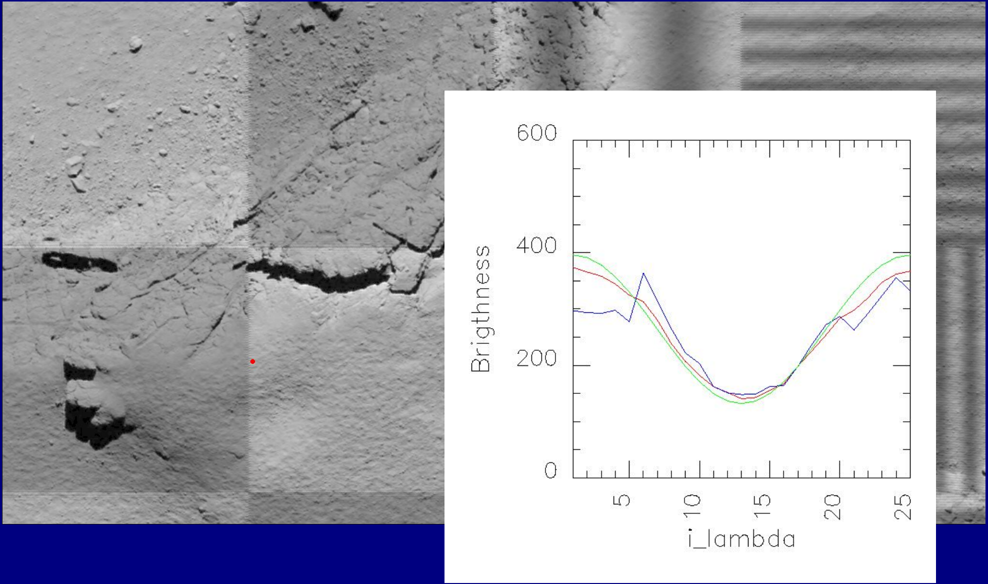
Simulated cube (F1, full 2048×1088)



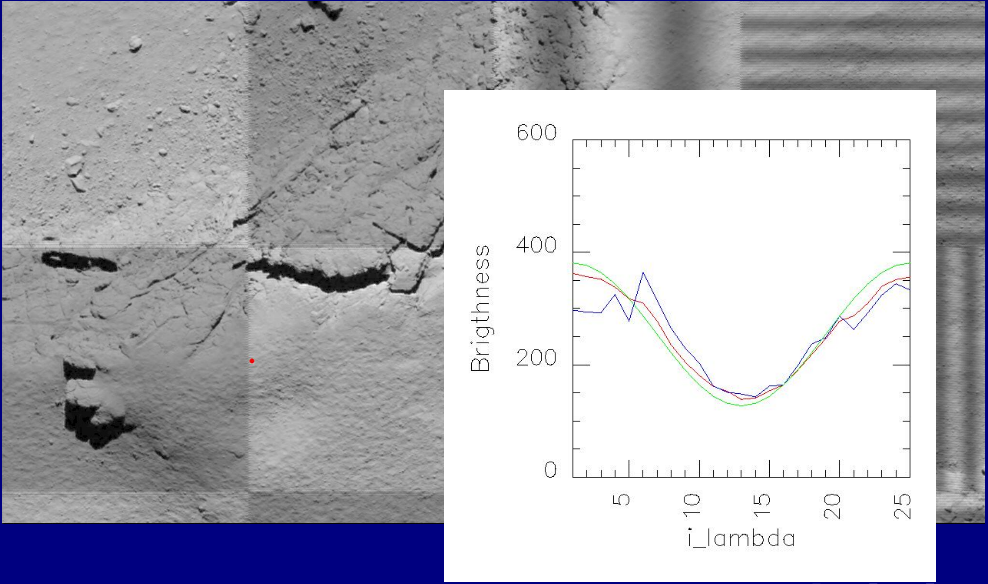
Simulated cube (F1, full 2048×1088)



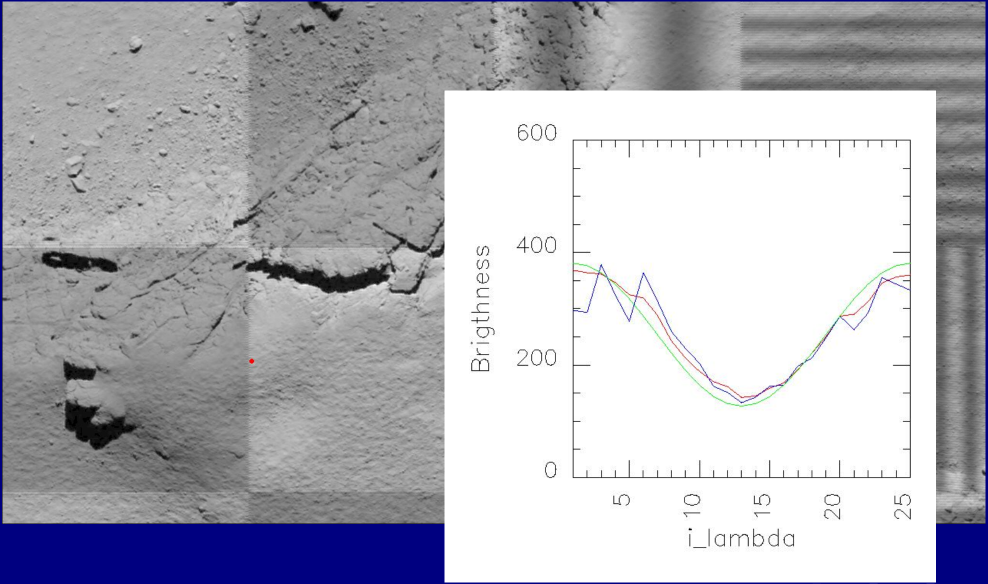
Simulated cube (F1, full 2048×1088)



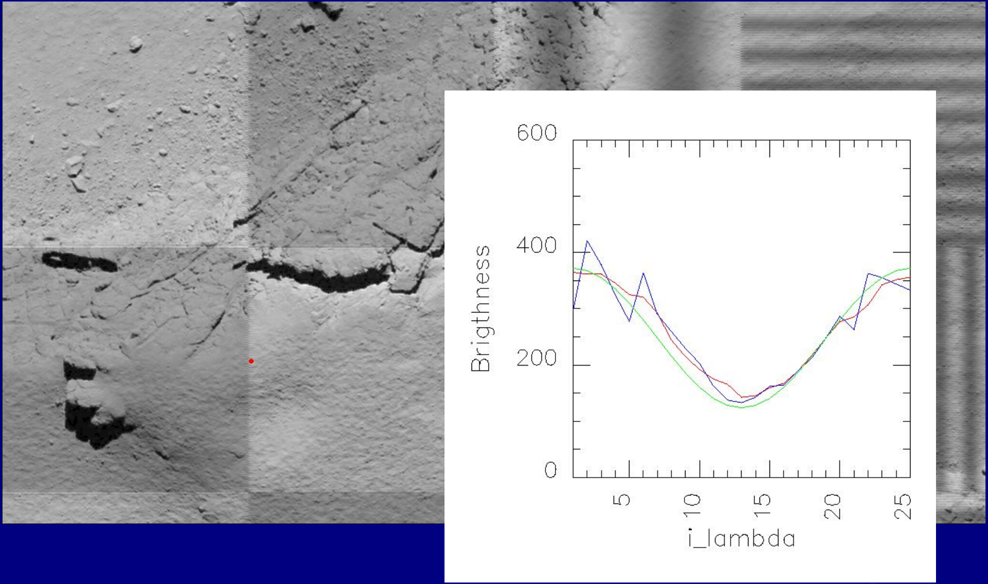
Simulated cube (F1, full 2048×1088)



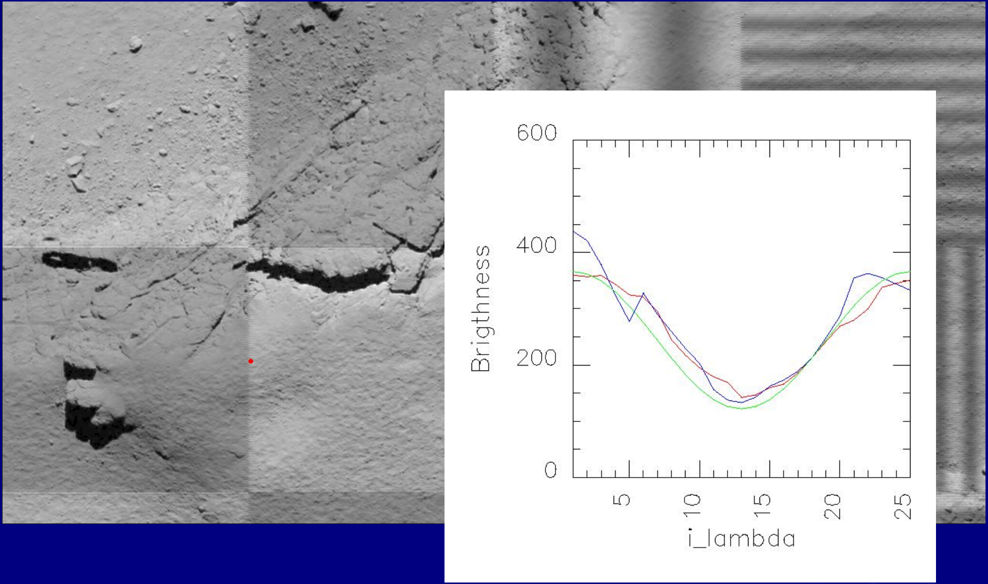
Simulated cube (F1, full 2048×1088)



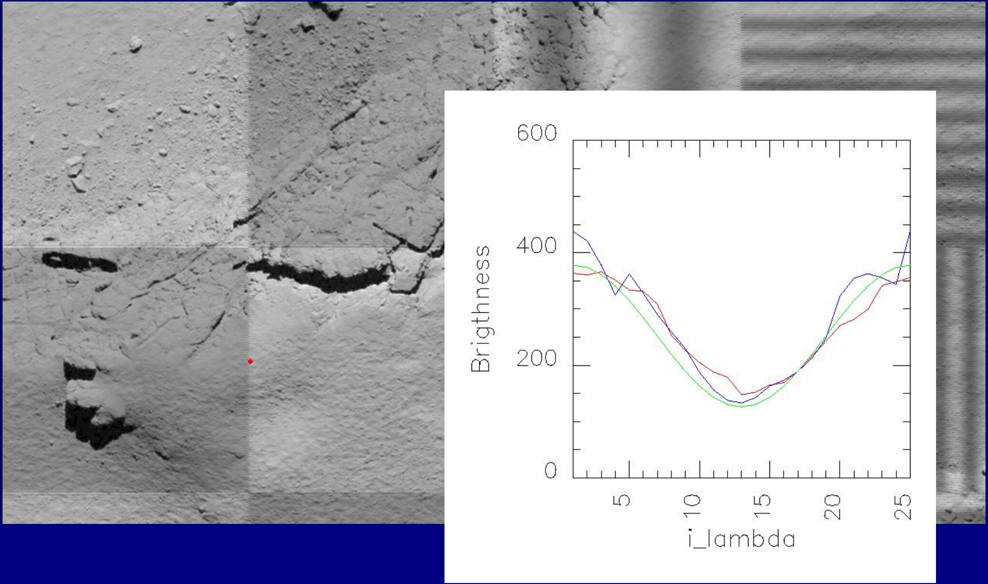
Simulated cube (F1, full 2048×1088)



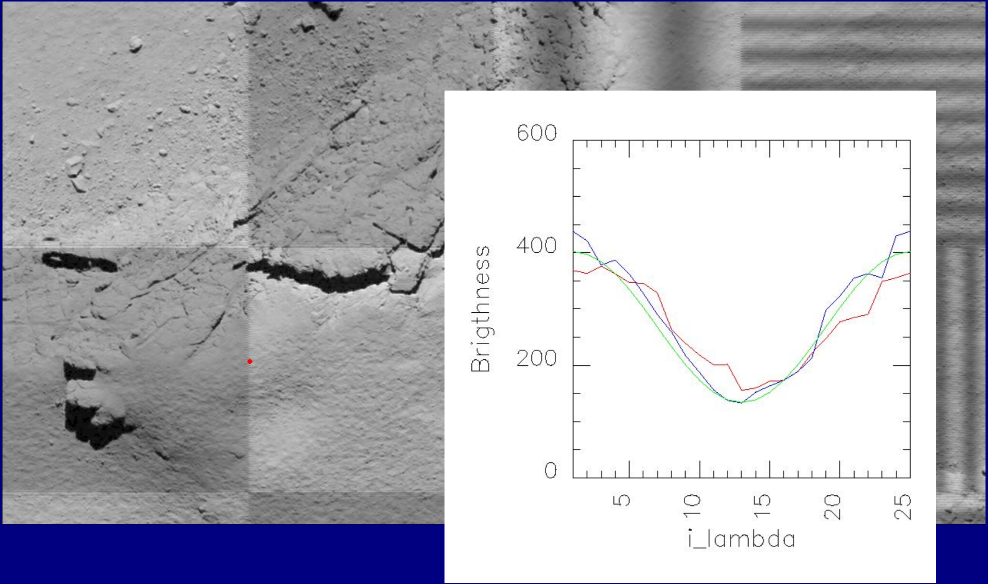
Simulated cube (F1, full 2048×1088)



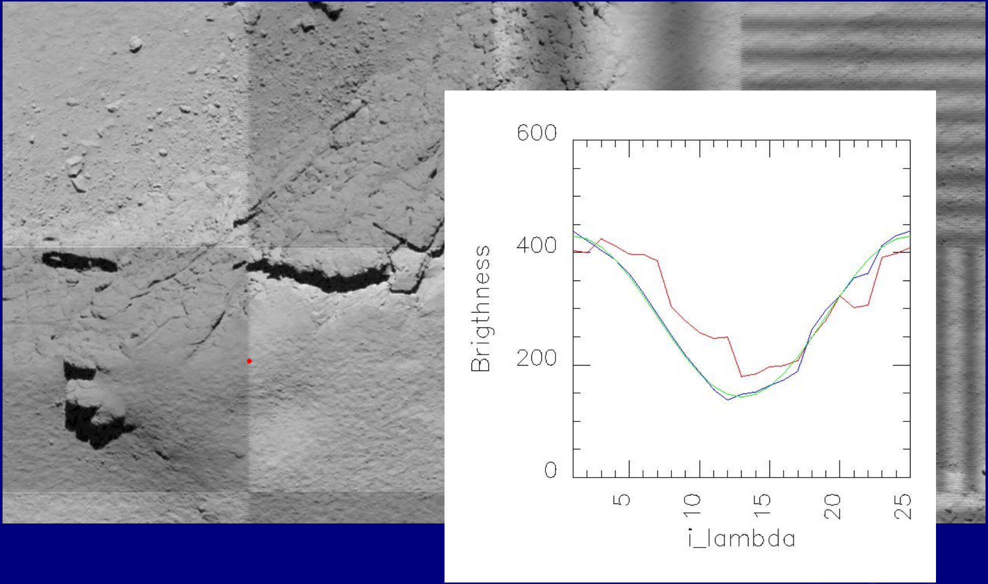
Simulated cube (F1, full 2048×1088)



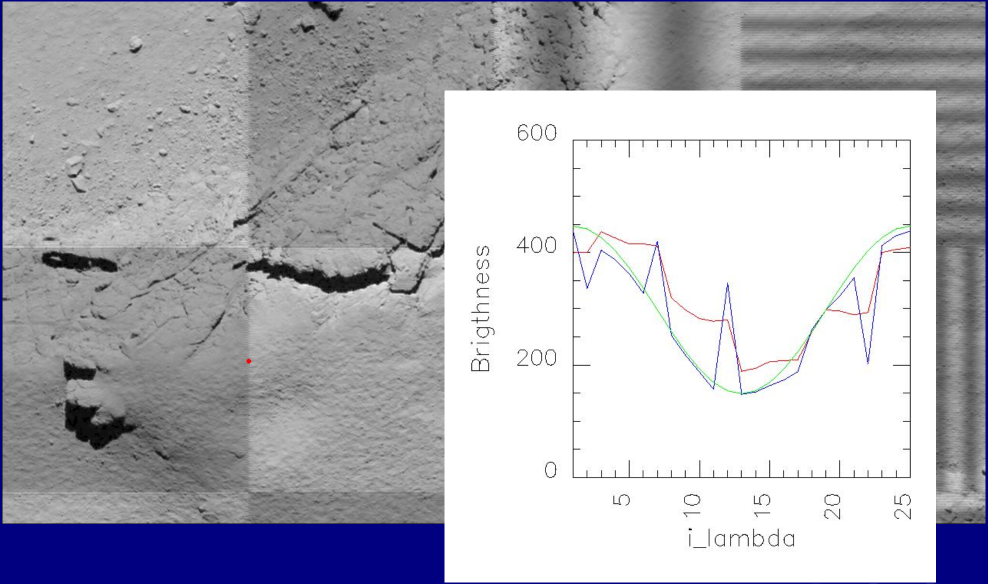
Simulated cube (F1, full 2048×1088)



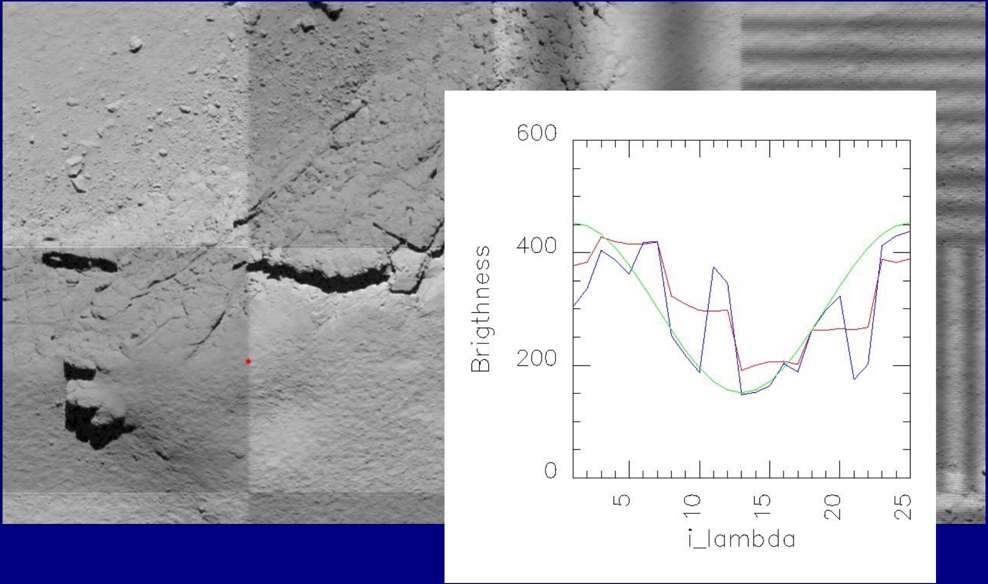
Simulated cube (F1, full 2048×1088)



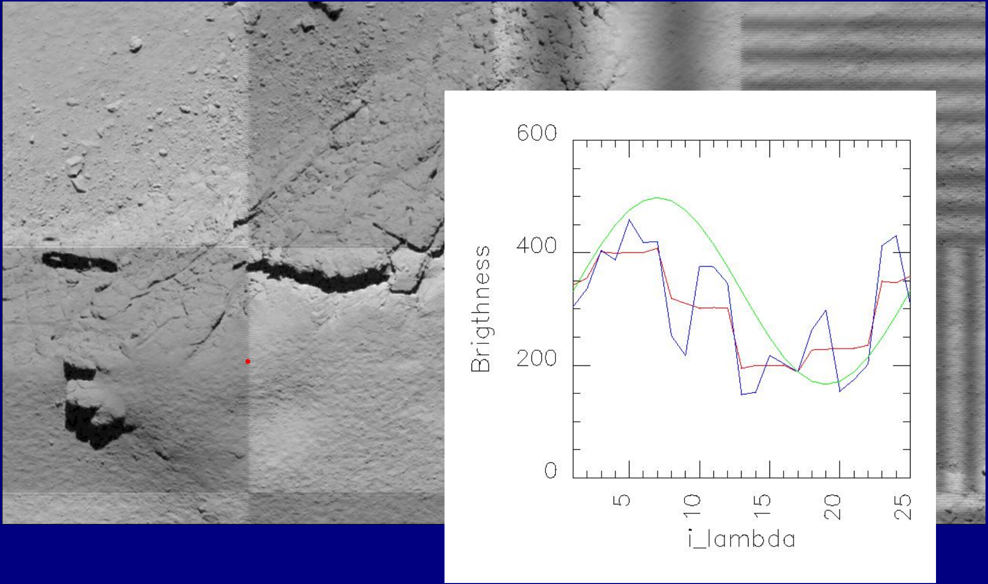
Simulated cube (F1, full 2048×1088)



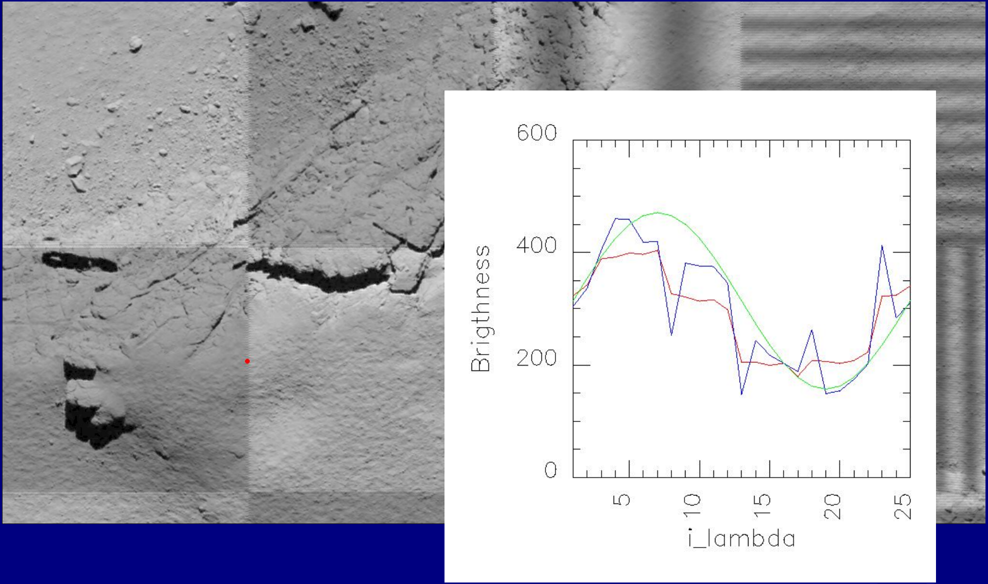
Simulated cube (F1, full 2048×1088)



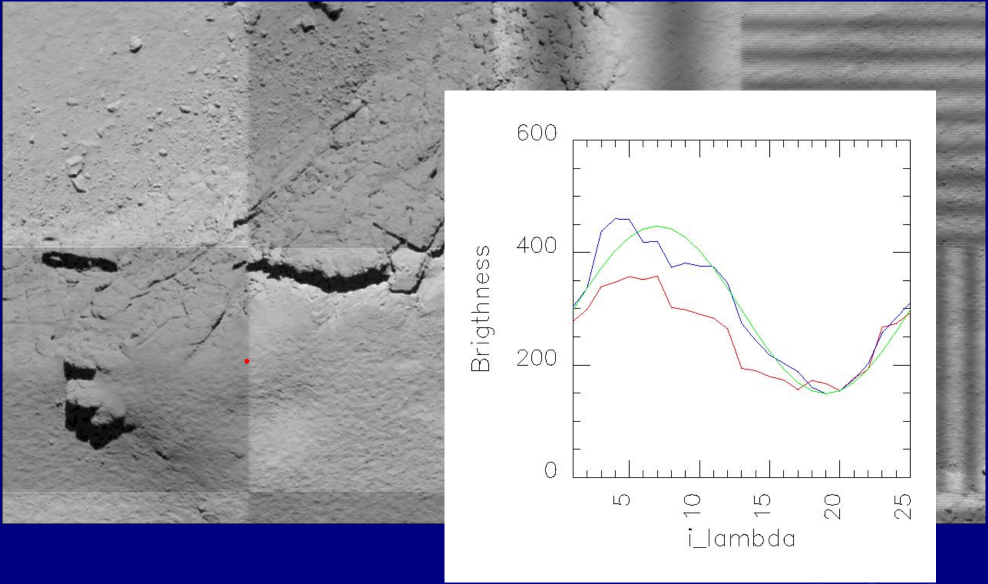
Simulated cube (F1, full 2048×1088)



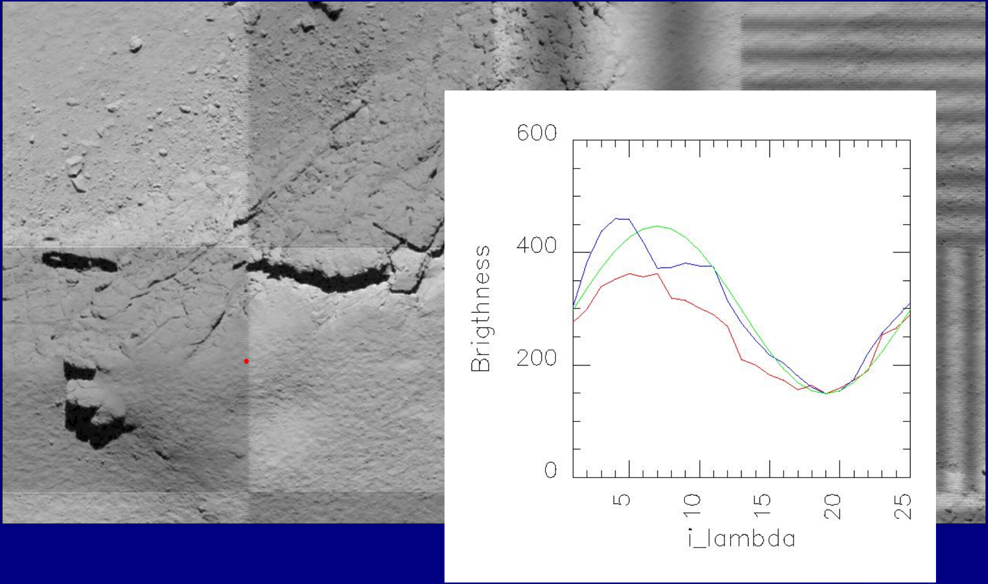
Simulated cube (F1, full 2048×1088)



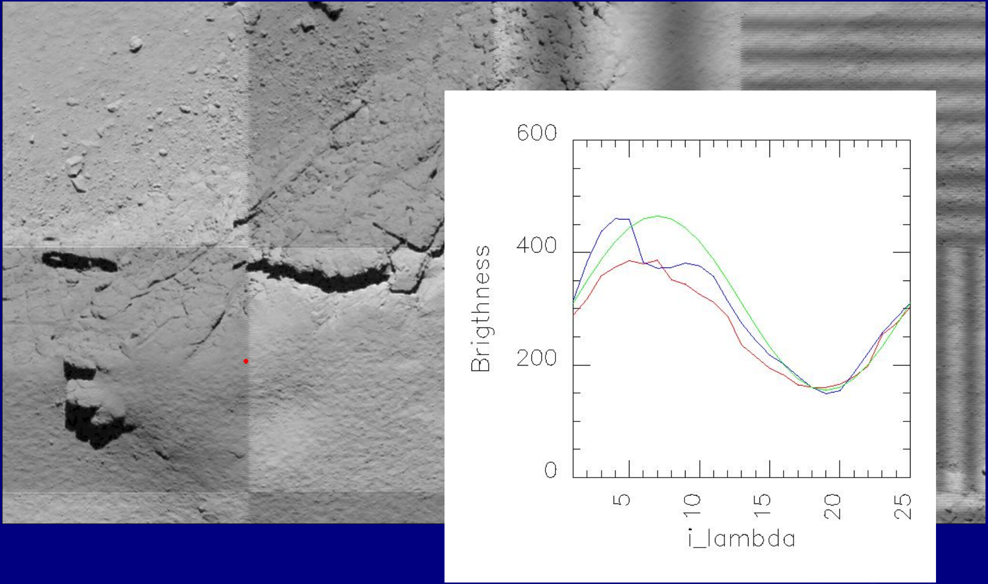
Simulated cube (F1, full 2048×1088)



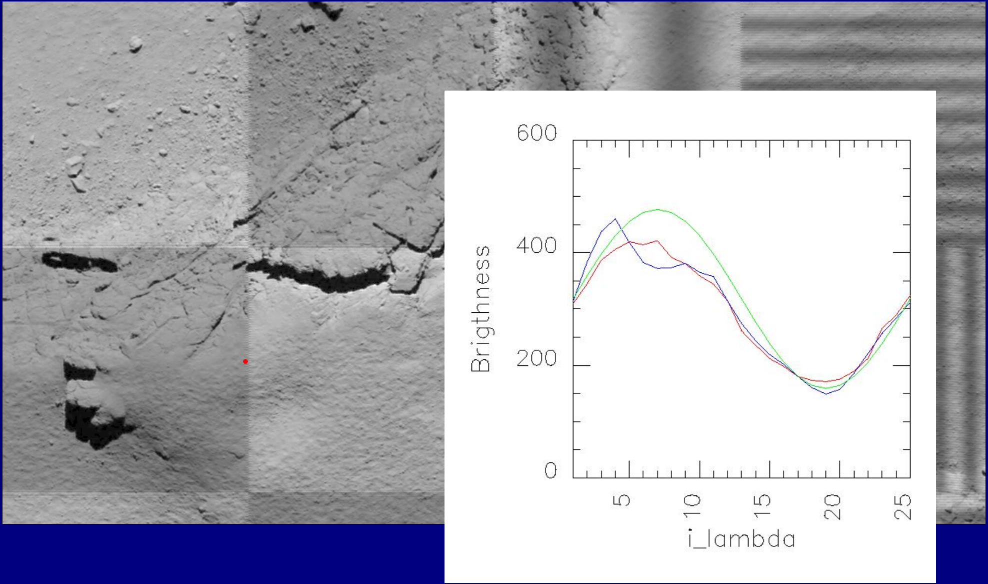
Simulated cube (F1, full 2048×1088)



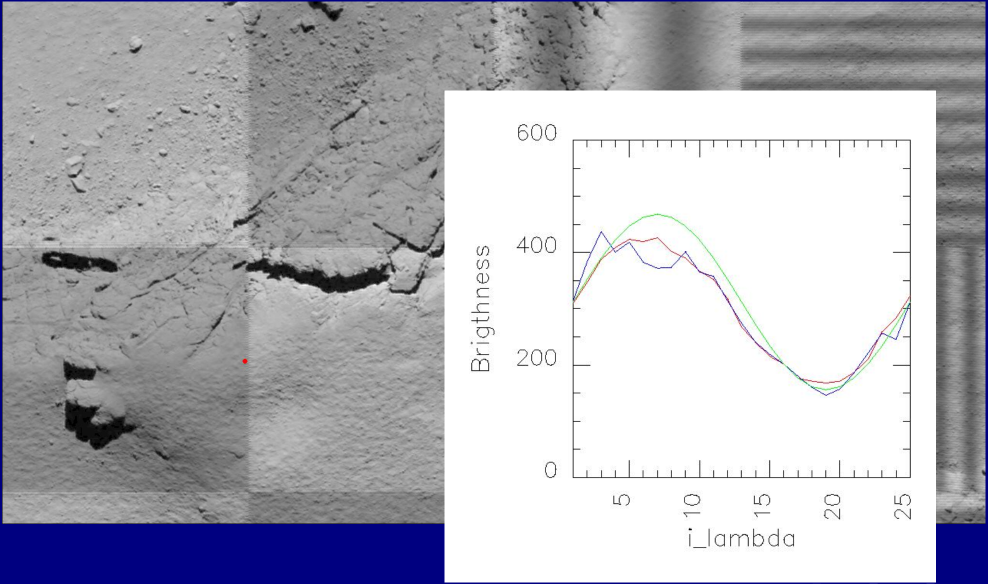
Simulated cube (F1, full 2048×1088)



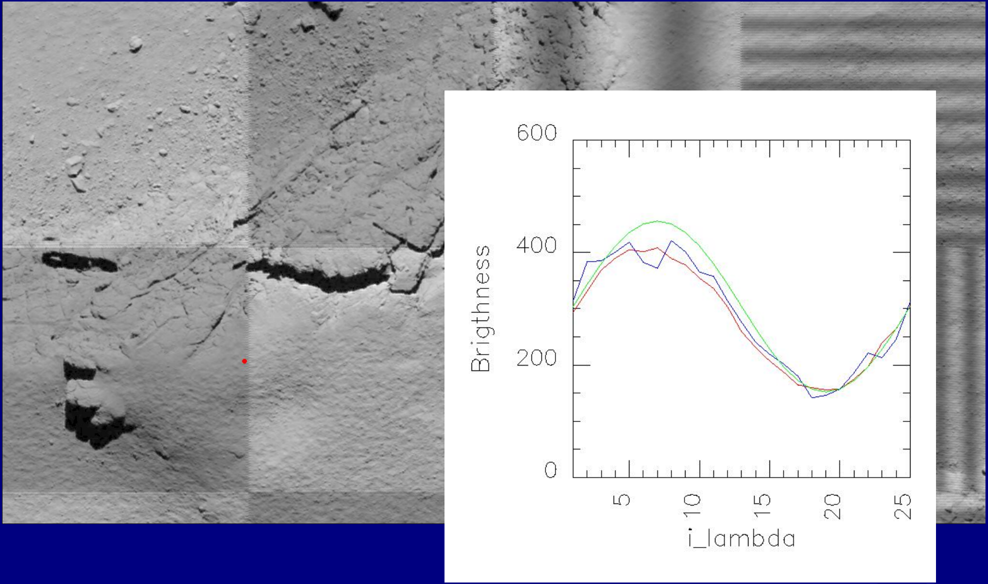
Simulated cube (F1, full 2048×1088)



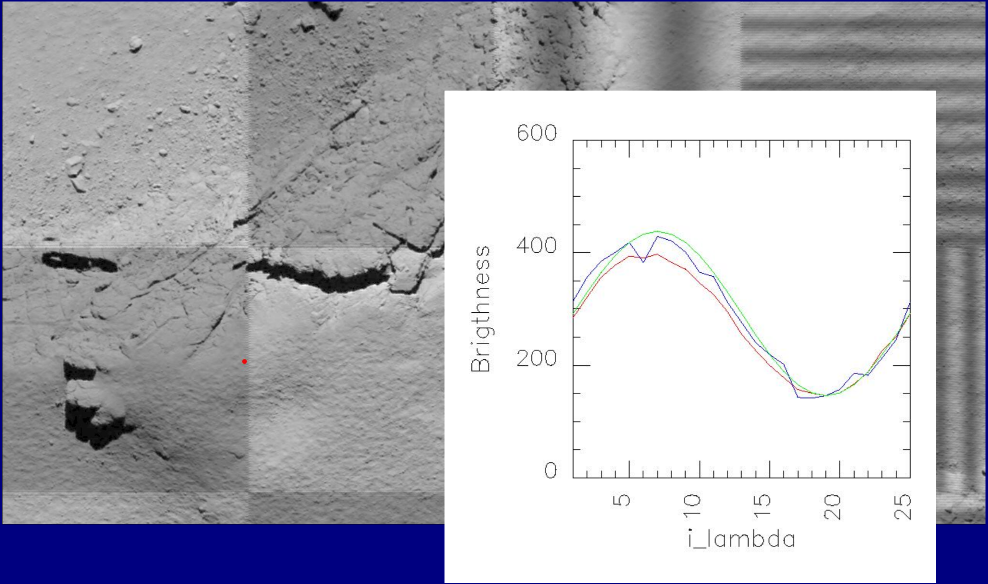
Simulated cube (F1, full 2048×1088)



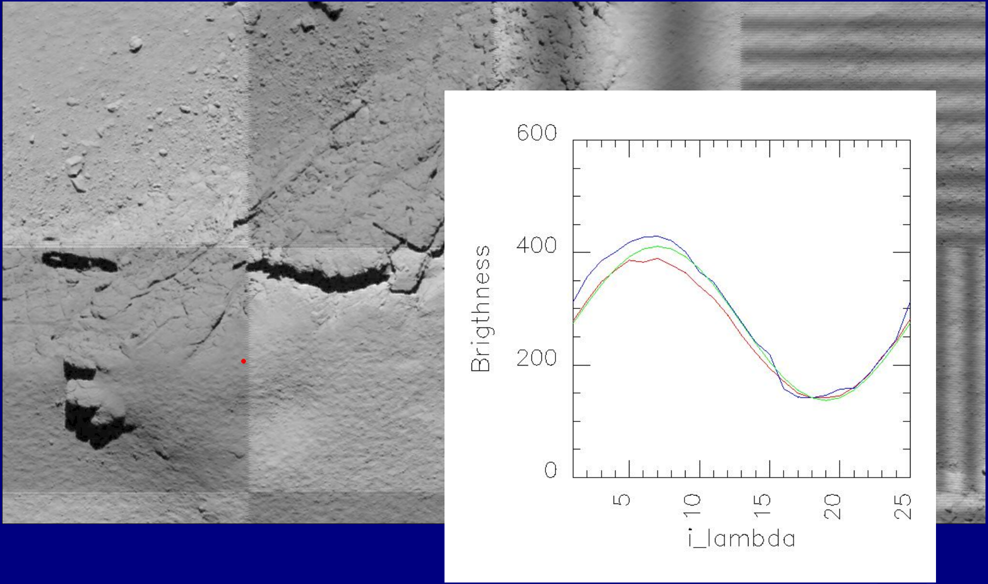
Simulated cube (F1, full 2048×1088)



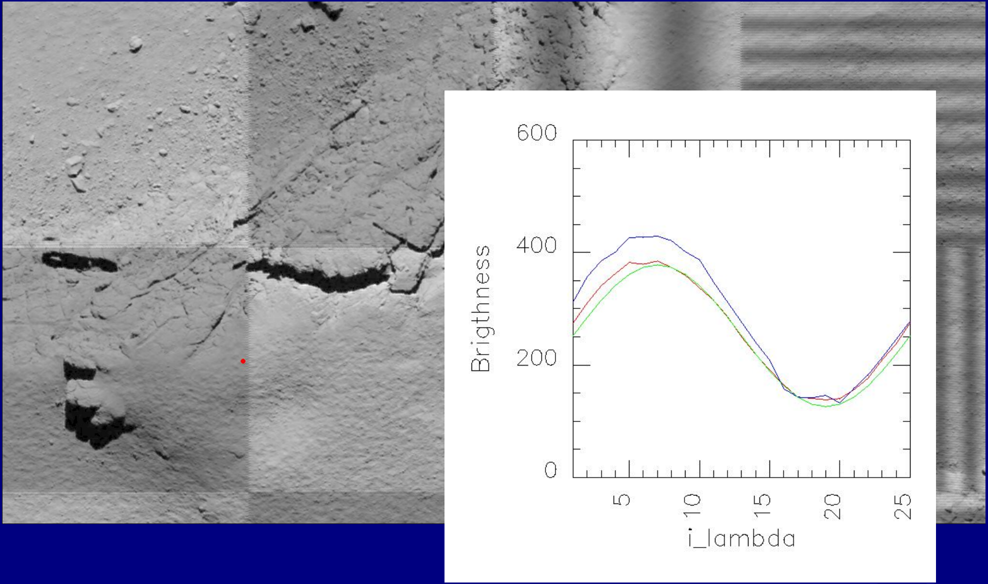
Simulated cube (F1, full 2048×1088)



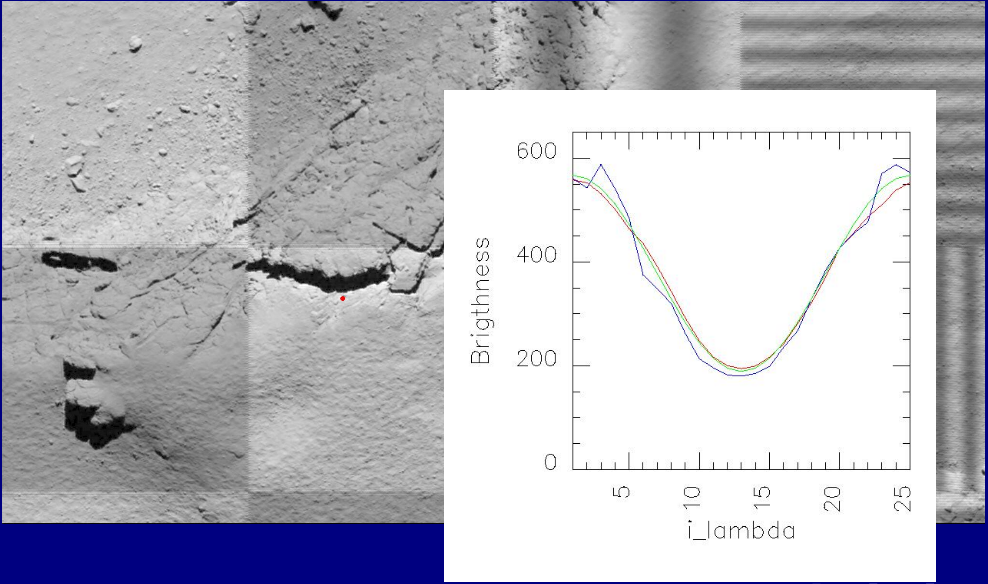
Simulated cube (F1, full 2048×1088)



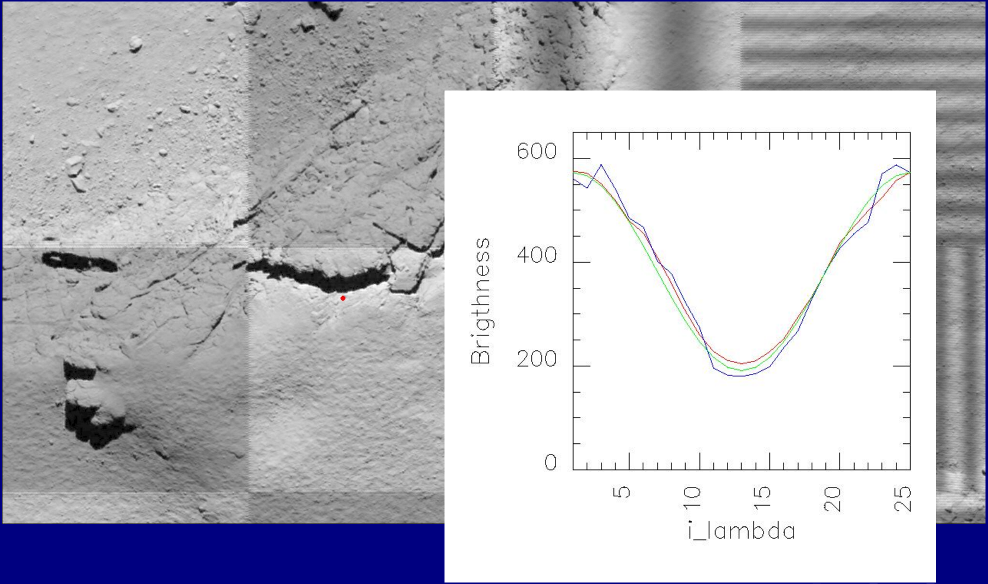
Simulated cube (F1, full 2048×1088)



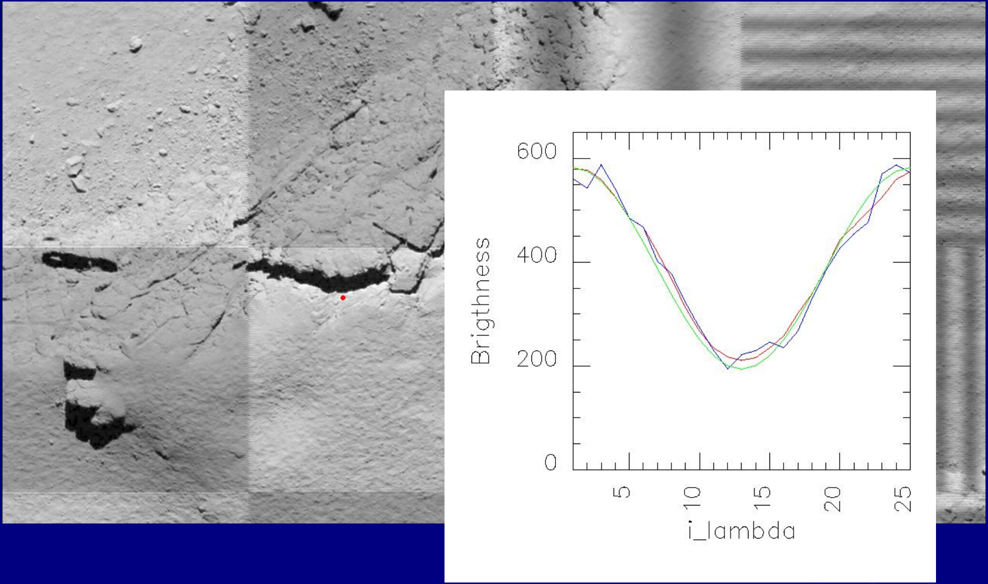
Simulated cube (F1, full 2048×1088)



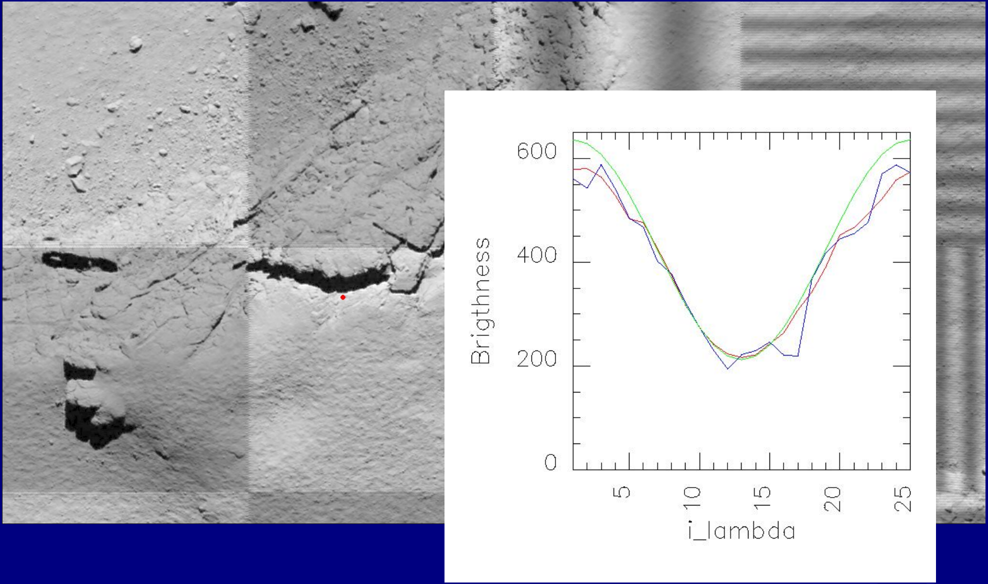
Simulated cube (F1, full 2048×1088)



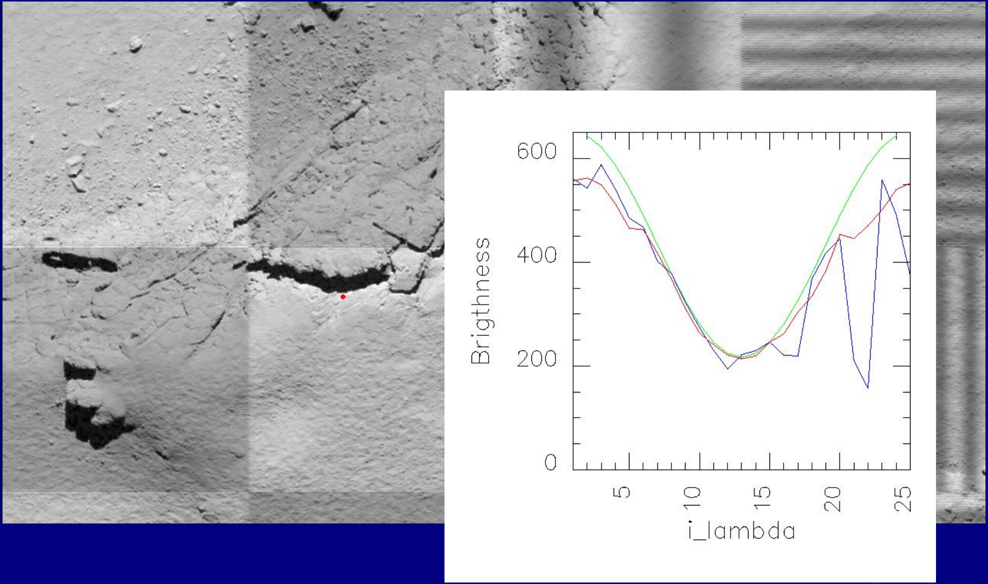
Simulated cube (F1, full 2048×1088)



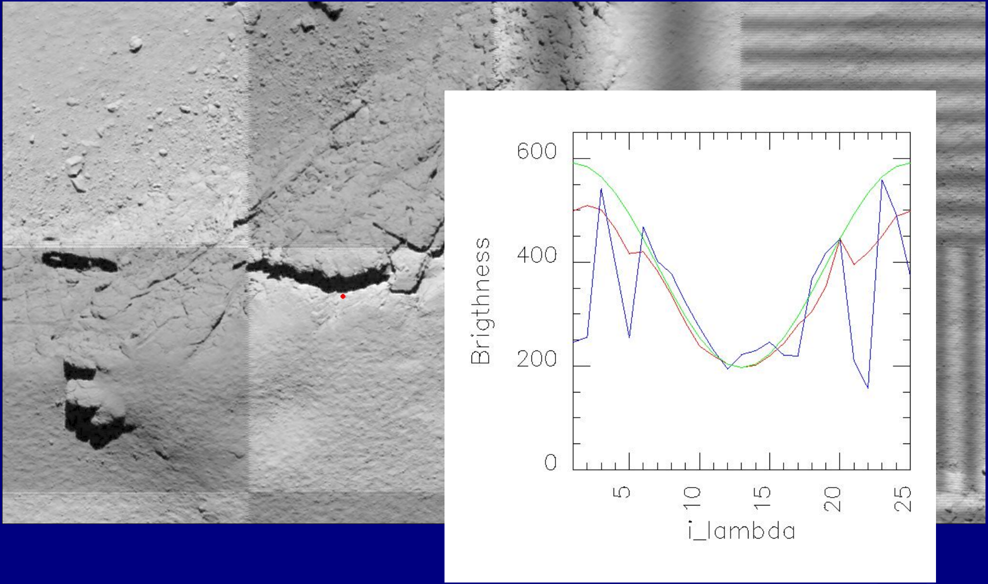
Simulated cube (F1, full 2048×1088)



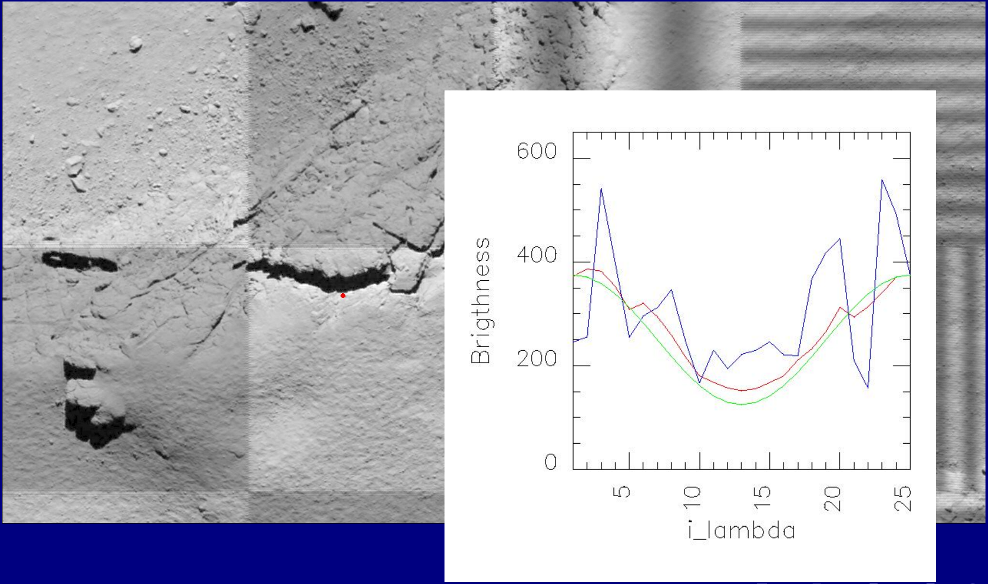
Simulated cube (F1, full 2048×1088)



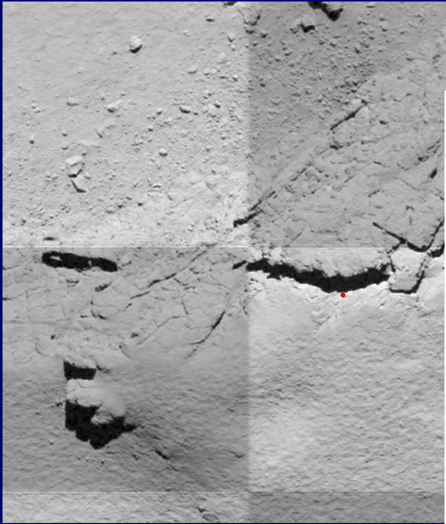
Simulated cube (F1, full 2048×1088)

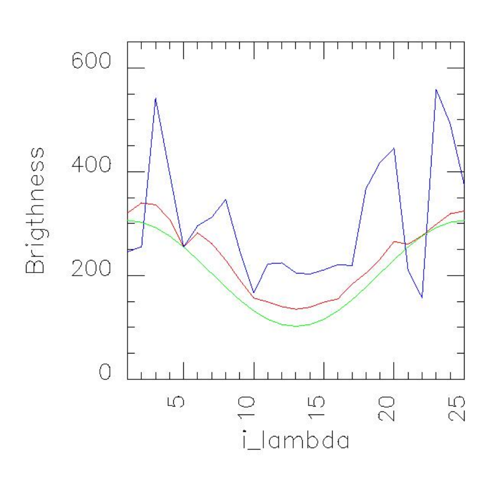


Simulated cube (F1, full 2048×1088)

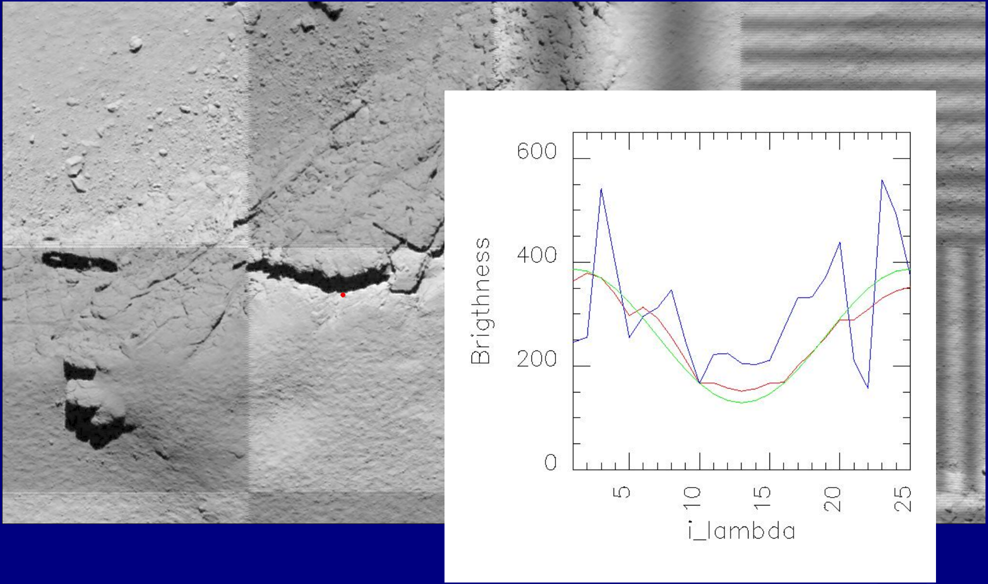


Simulated cube (F1, full 2048×1088)

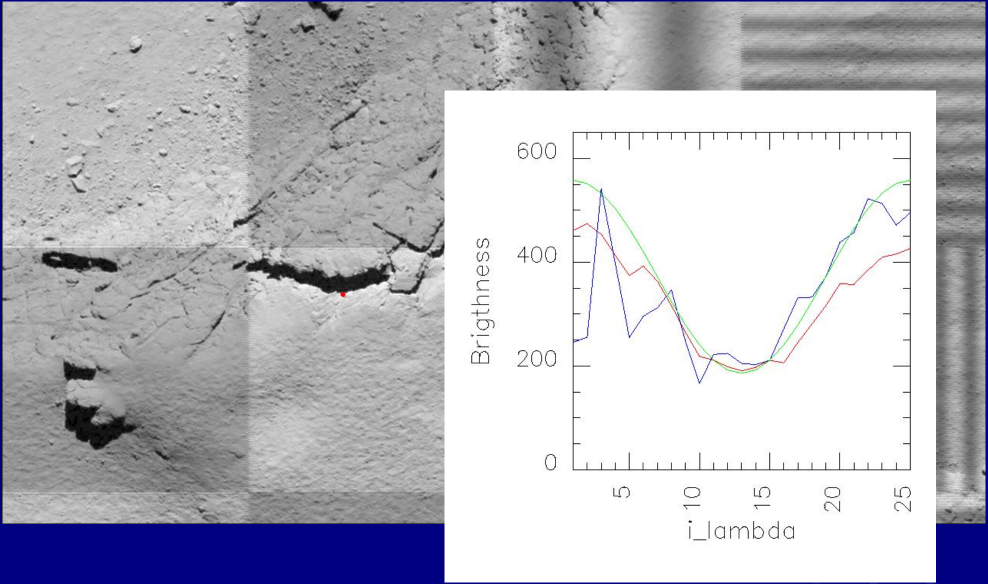




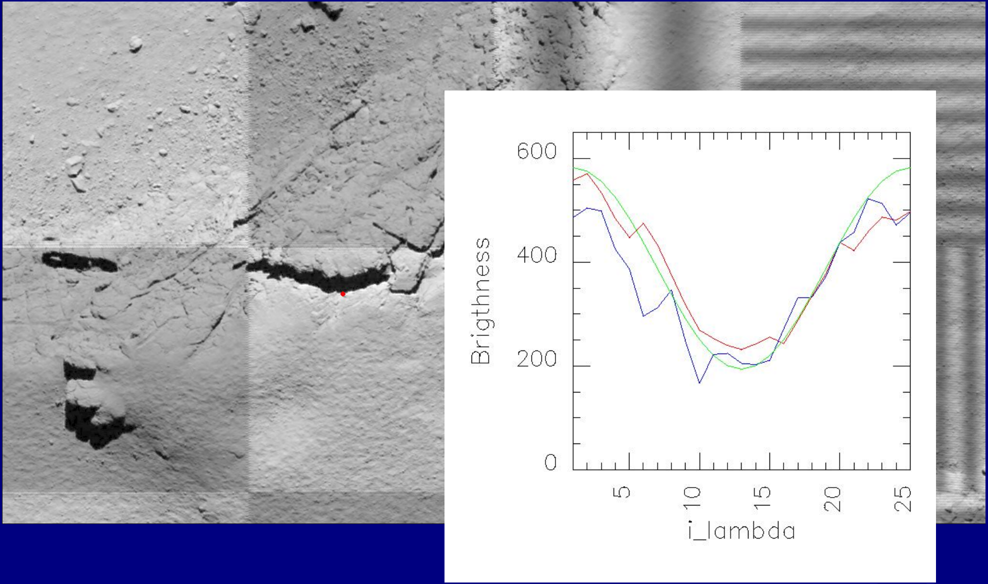
Simulated cube (F1, full 2048×1088)



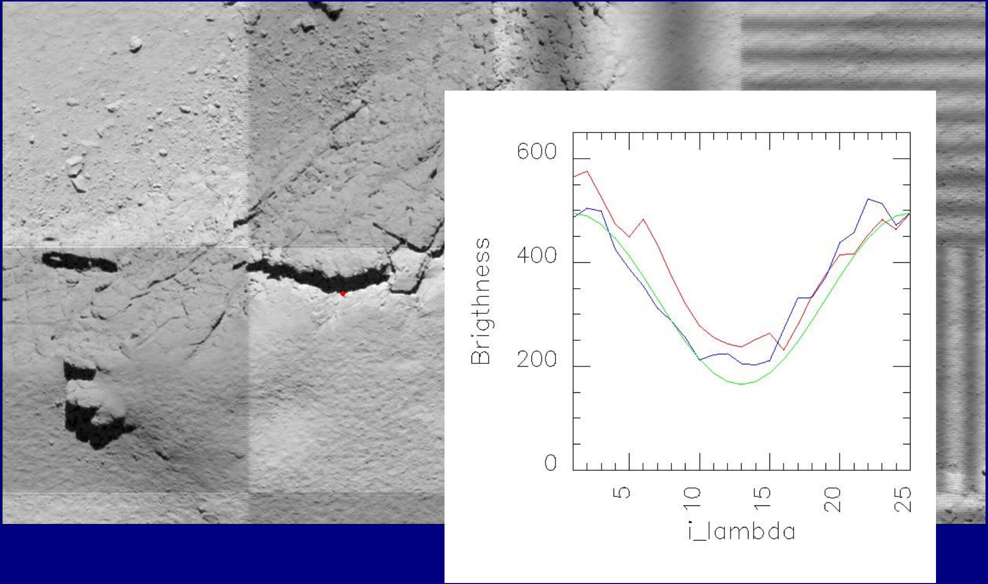
Simulated cube (F1, full 2048×1088)



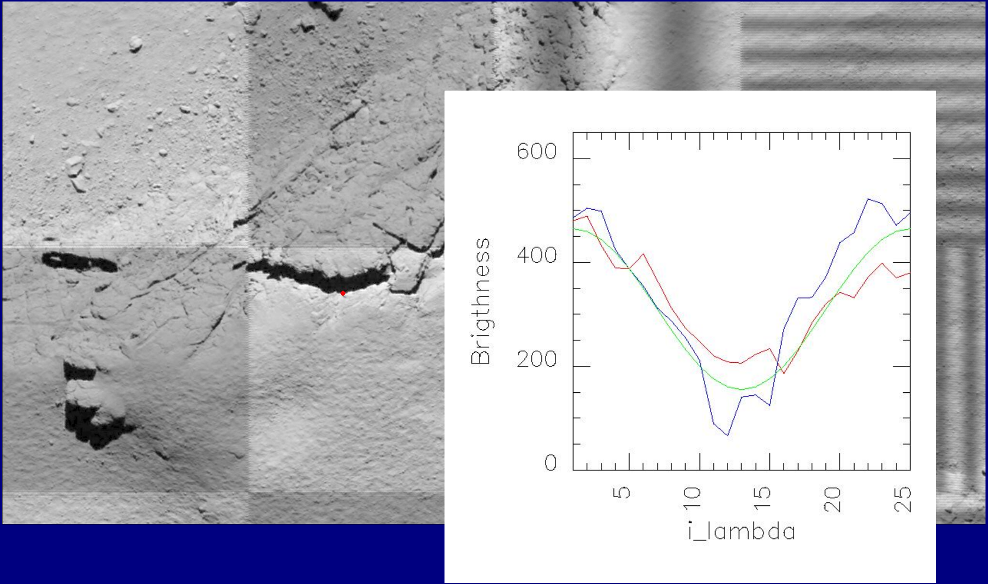
Simulated cube (F1, full 2048×1088)



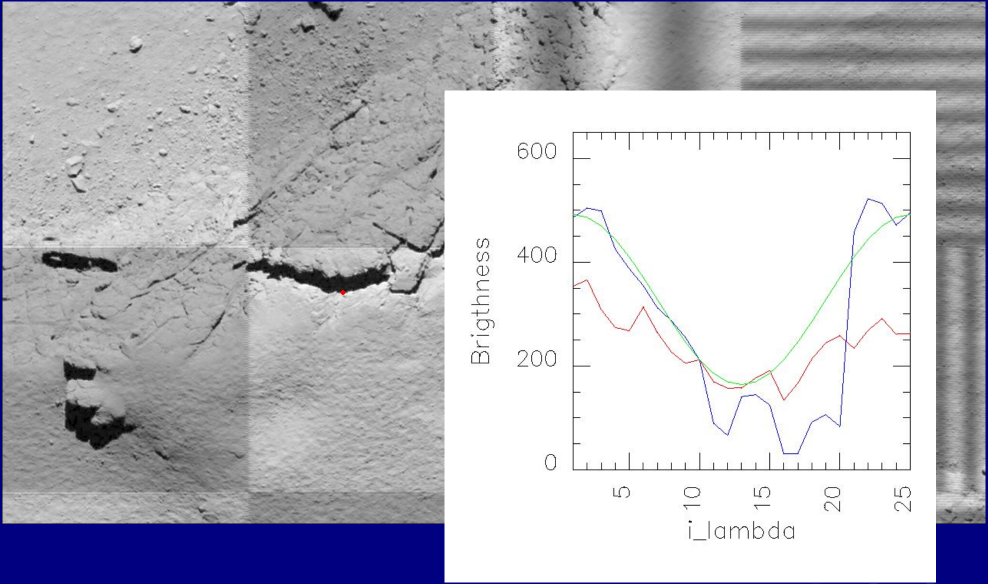
Simulated cube (F1, full 2048×1088)



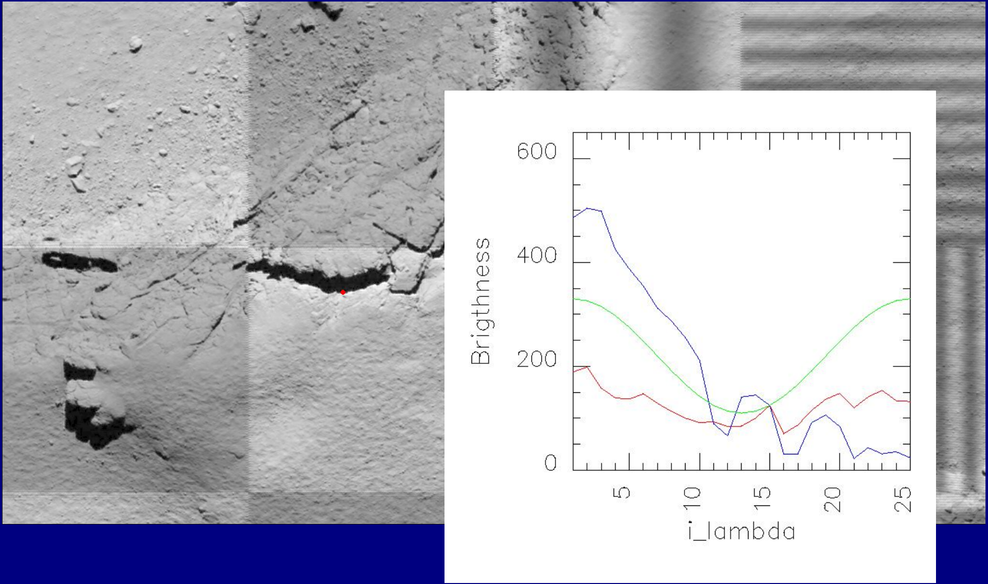
Simulated cube (F1, full 2048×1088)



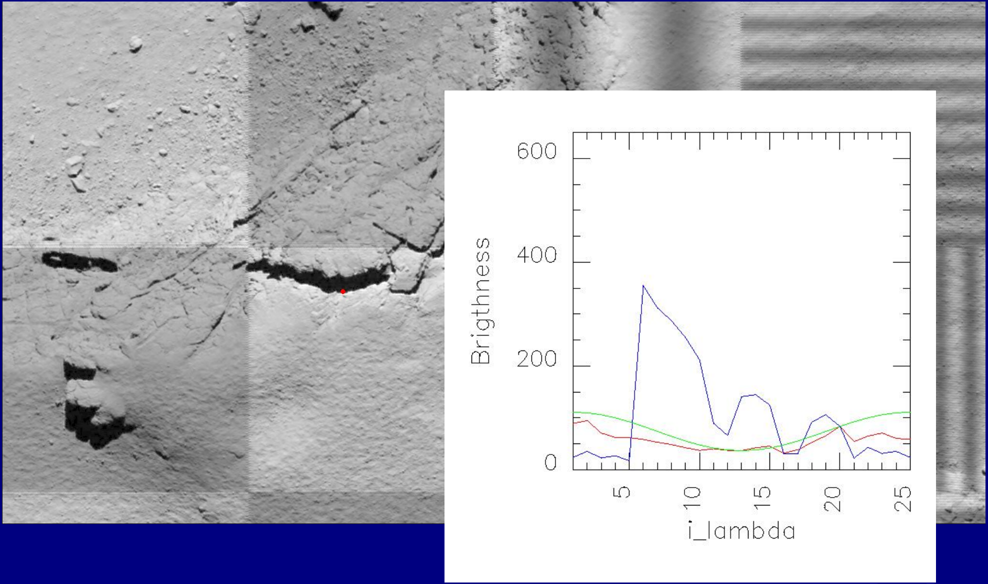
Simulated cube (F1, full 2048×1088)



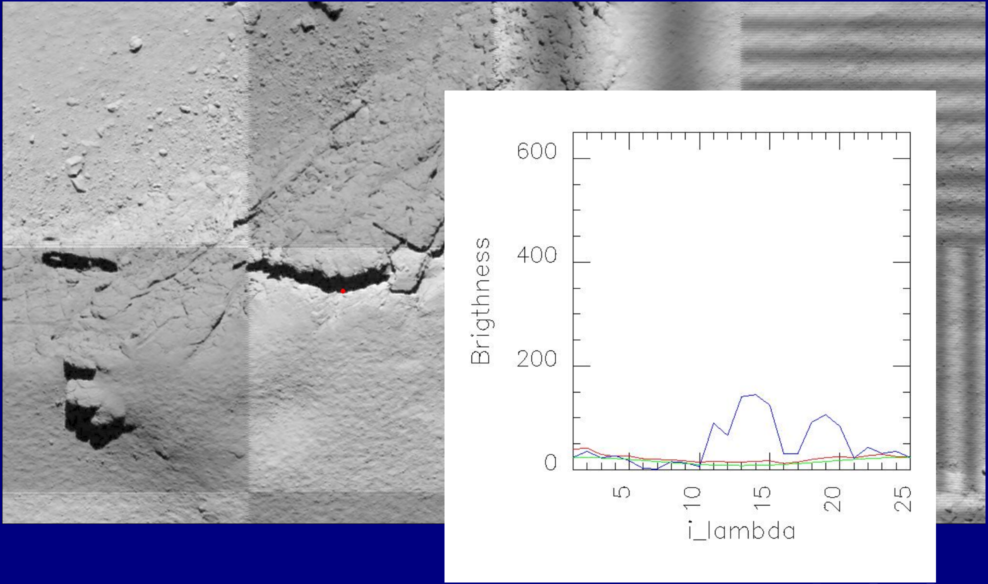
Simulated cube (F1, full 2048×1088)



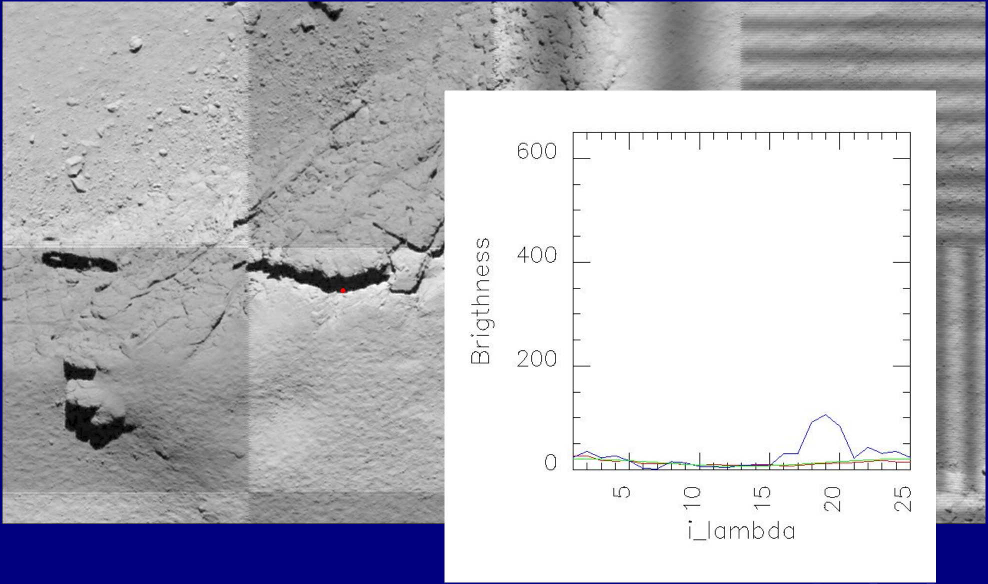
Simulated cube (F1, full 2048×1088)



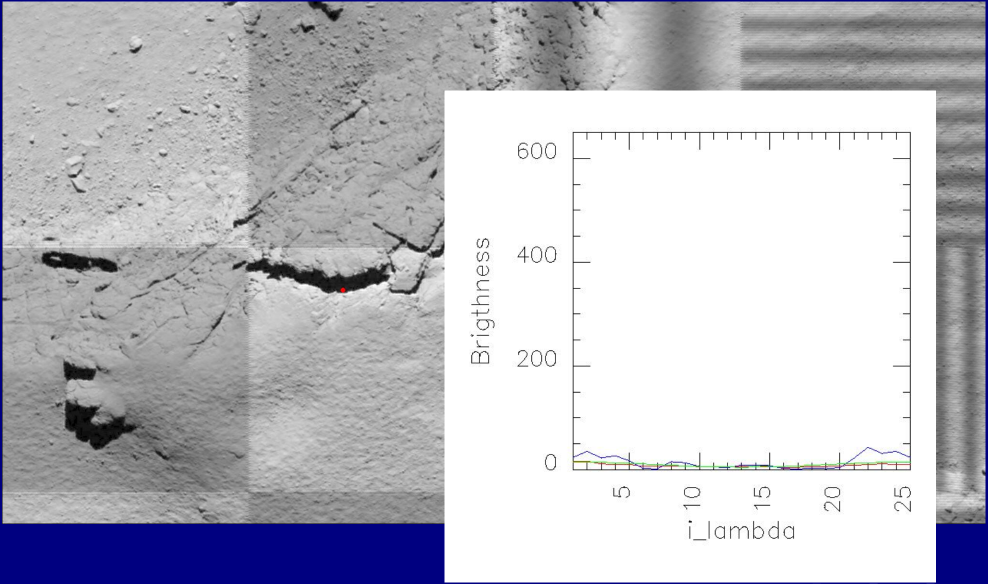
Simulated cube (F1, full 2048×1088)



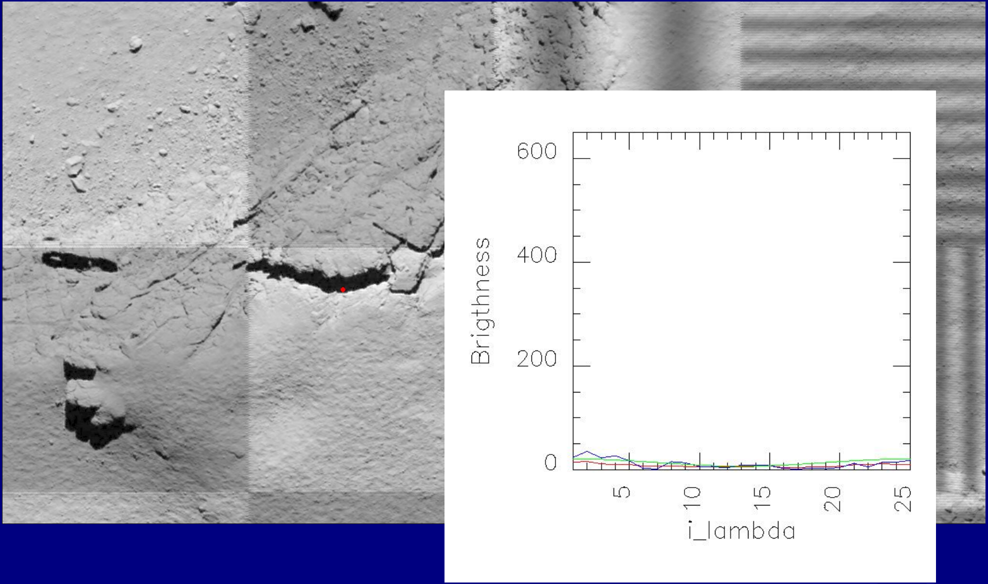
Simulated cube (F1, full 2048×1088)



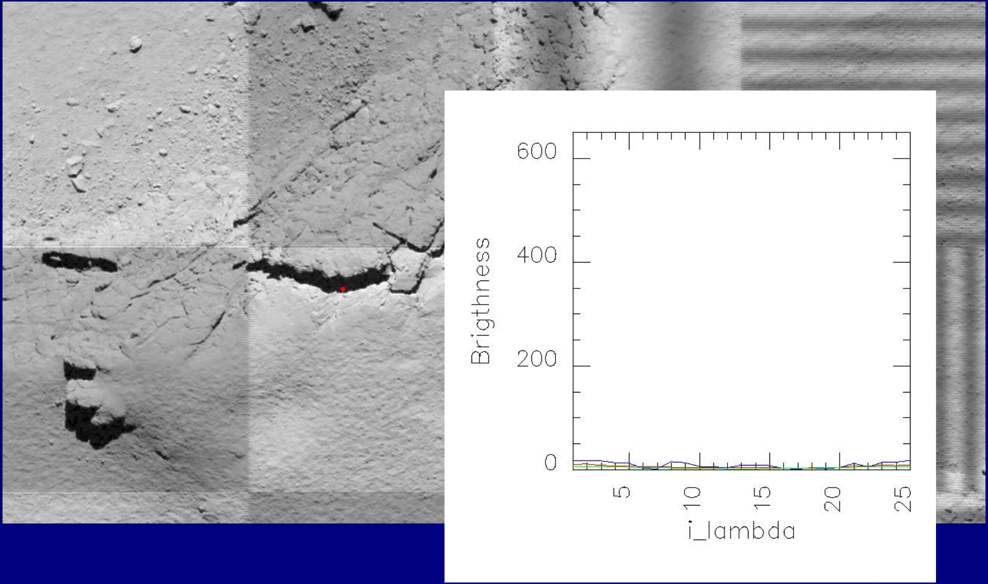
Simulated cube (F1, full 2048×1088)



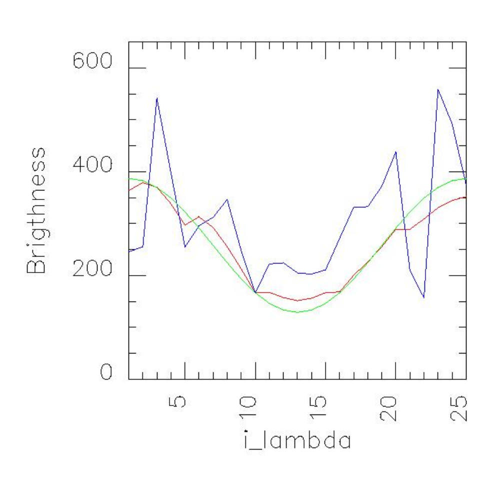
Simulated cube (F1, full 2048×1088)



Simulated cube (F1, full 2048×1088)



Conclusions



- We have replenished a simulated data cube with the simple direct approach and with our homegrown approach de2.
- Both approaches have problems with strong spatial gradients in the (normalized) spectrum.
- de2 is much better than the direct approach in dealing with strong spatial brightness gradients.
- We will do more tests with more realistic test data and also try other approaches.