



cnrs

**Centre de Calcul**

de l'Institut National de Physique Nucléaire  
et de Physique des Particules

# Optimisation de la pipeline de traitement des images de Rubin-LSST

**Quentin Le Boulc'h**

Avec J. Bregeon (LPSC), D. Boutigny (LAPP), F. Hernandez (CC-IN2P3),  
D. Parello (Univ. Perpignan), C. Parisel (APC)

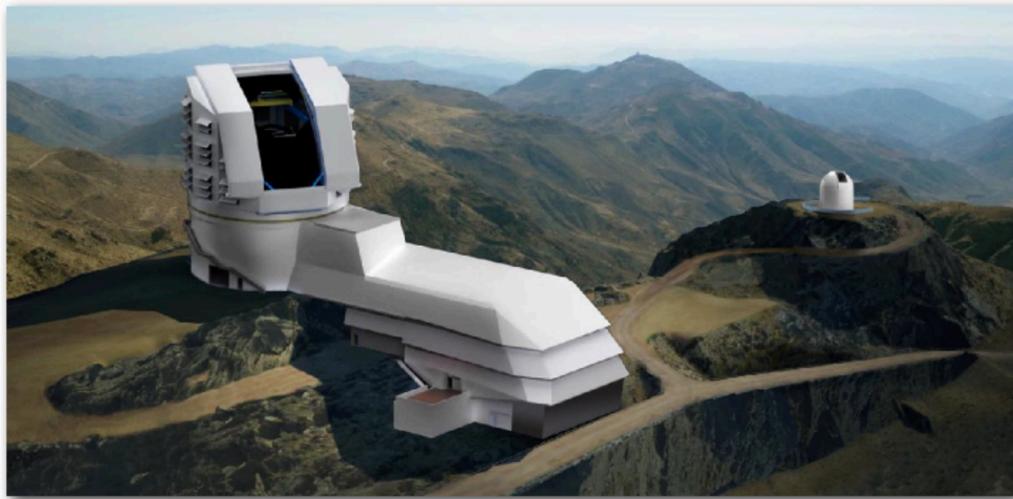


- L'observatoire Vera C. Rubin et le LSST
- Traitement des données
- Profilage de la pipeline
- Résumé et perspectives



# L'observatoire Vera C. Rubin et le LSST

## OBSERVATORY



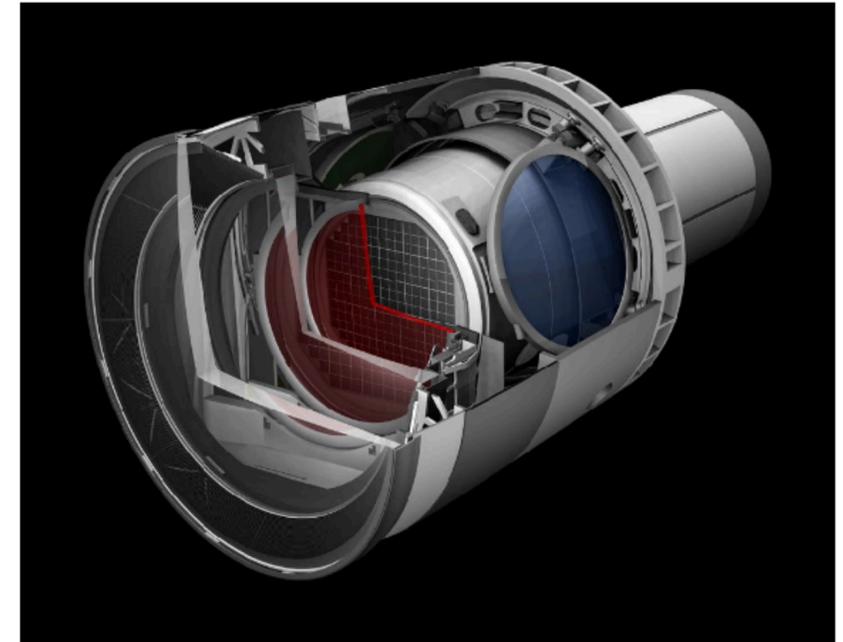
southern hemisphere | 2647m  
a.s.l. | stable air | clear sky |  
dark nights | good infrastructure

## TELESCOPE



main mirror  $\varnothing$  8.4 m (effective  
6.4 m) | large aperture: f/1.234  
| wide field of view | 350 ton |  
compact | to be repositioned  
about 3M times over 10 years  
of operations

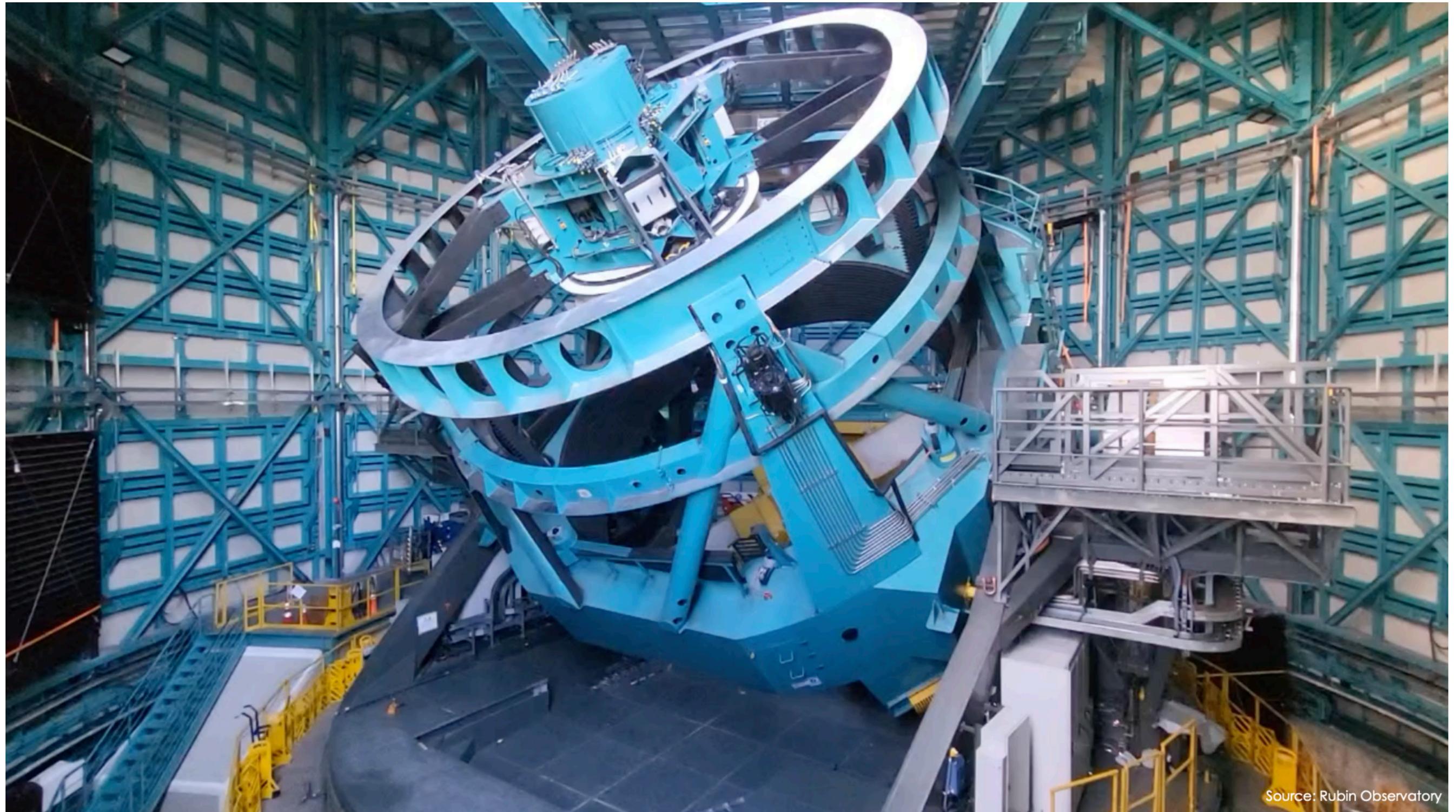
## CAMERA



**3.2 G pixels** |  $\varnothing$  1.65 m |  
3.7 m long | 3 ton | 3  
lenses |  $3.5^\circ$  field of view |  
9.6 deg<sup>2</sup> | 6 filters *ugrizy* |  
320-1050 nm | focal plane  
and electronics in cryostat  
at 173K



# L'observatoire Vera C. Rubin et le LSST



# L'observatoire Vera C. Rubin et le LSST



90% du temps d'observation dédié au Legacy Survey of Space and Time (LSST) :

- 43% du ciel
- Ciel de l'hémisphère sud observé en totalité toutes les 4 nuits
- Chaque zone du ciel observée sera visitée environ 800 fois dans 6 bandes de longueur d'onde différentes
- A partir de 2025 et pendant 10 ans

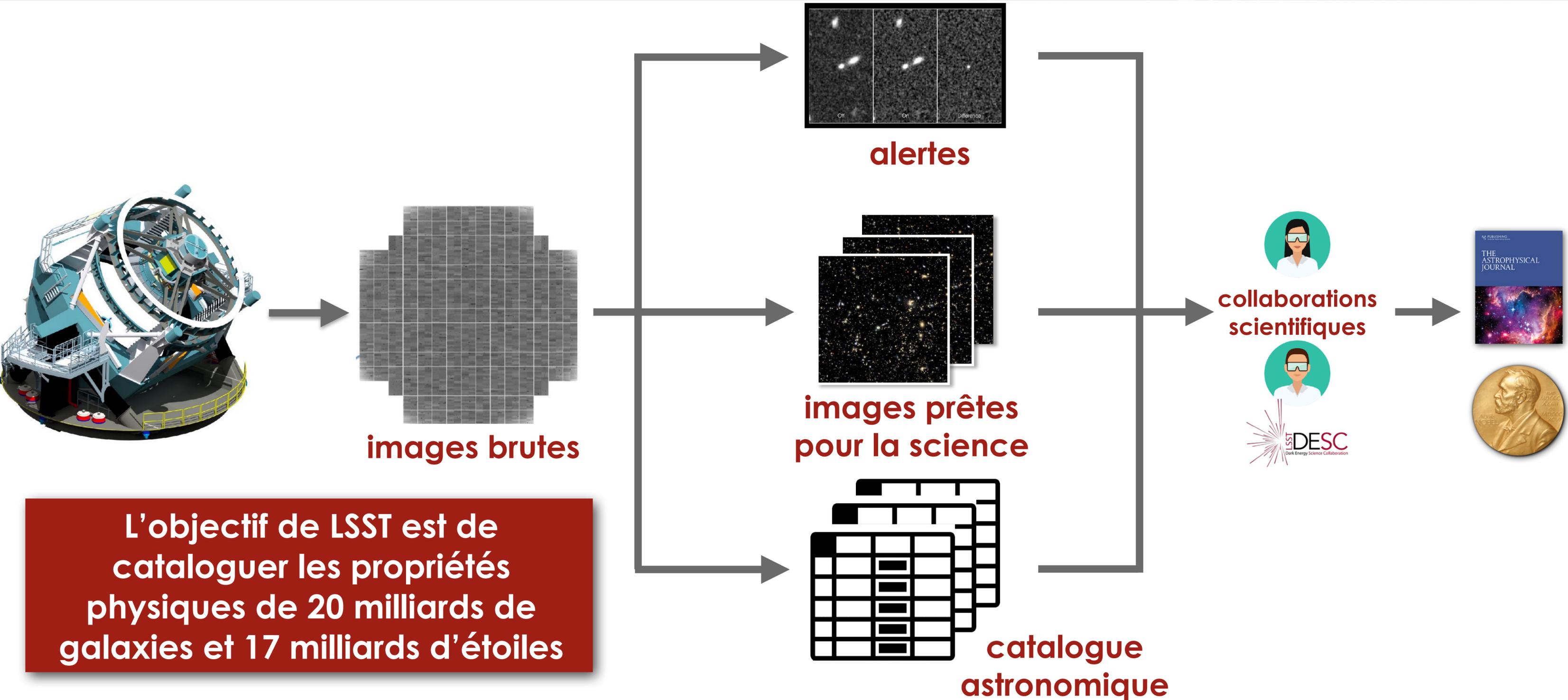
Objectifs scientifiques :

- Explorer la nature de la matière noire et de l'énergie sombre
- Etude du système solaire et de la voie lactée
- Etude des objets transients

Contributions de l'IN2P3 :

- 10 laboratoires IN2P3 : APC, CC-IN2P3, CPPM, IJCLab, IP2I, LAPP, LPC, LPNHE, LPSC, LUPM
- Plus de 100 personnes (chercheurs, ingénieurs, techniciens)
- Contributions à la caméra, changeur de filtre, traitement des données, analyse scientifique

# L'observatoire Vera C. Rubin et le LSST



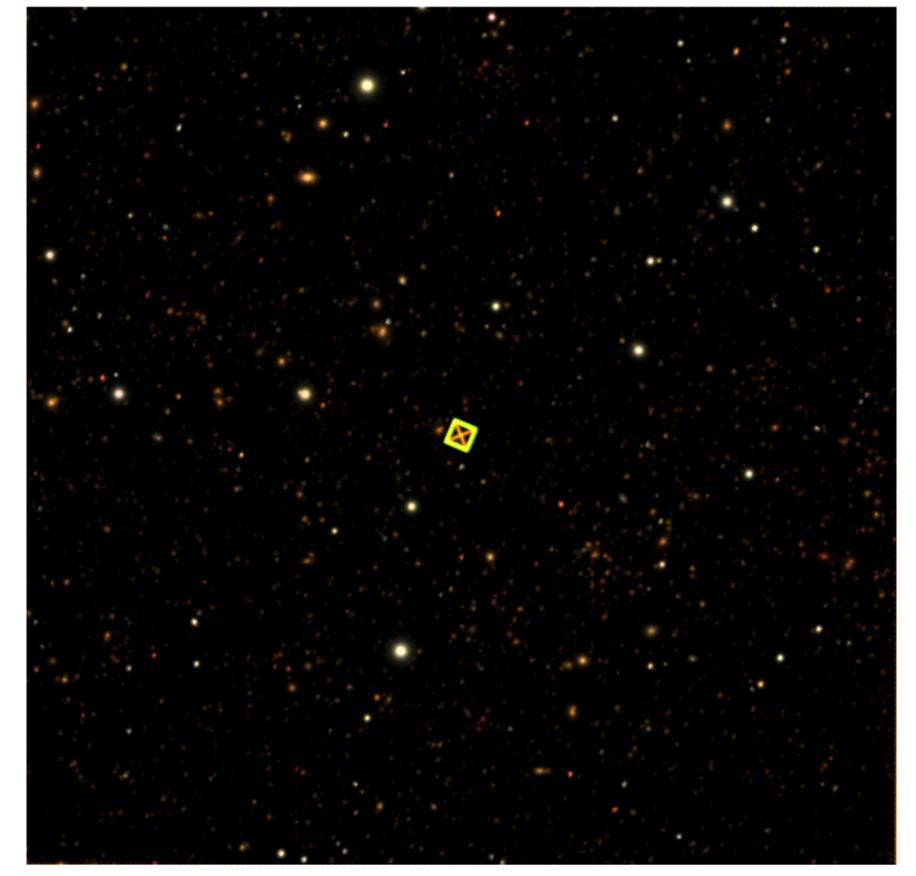
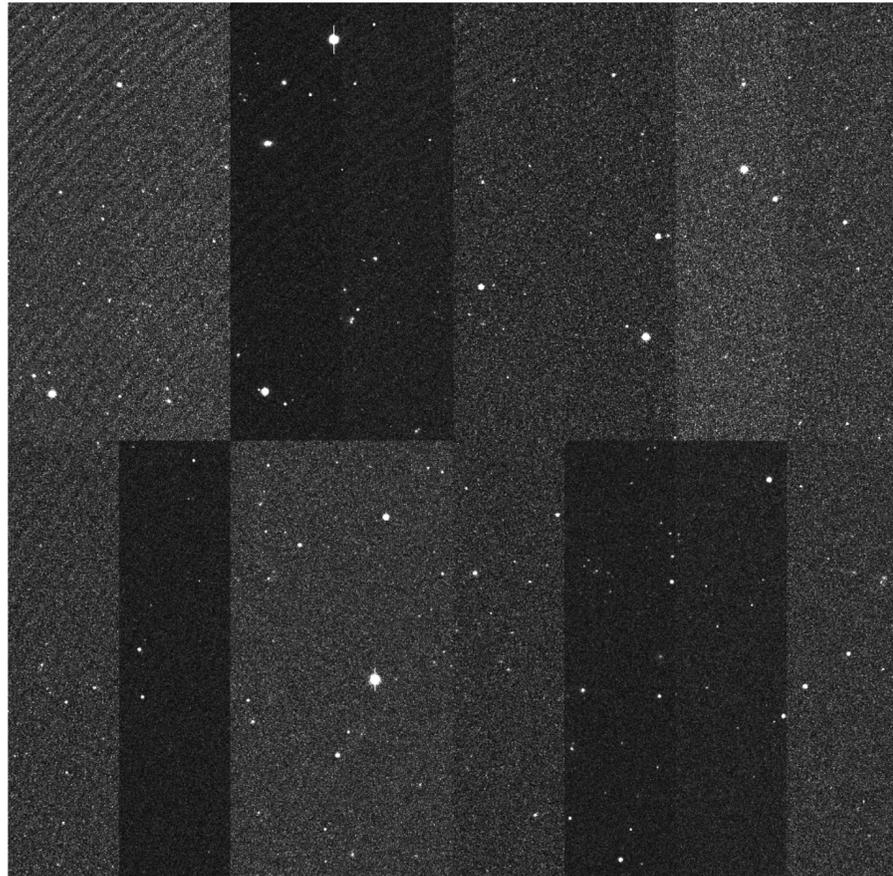
L'objectif de LSST est de cataloguer les propriétés physiques de 20 milliards de galaxies et 17 milliards d'étoiles



# Traitement des données

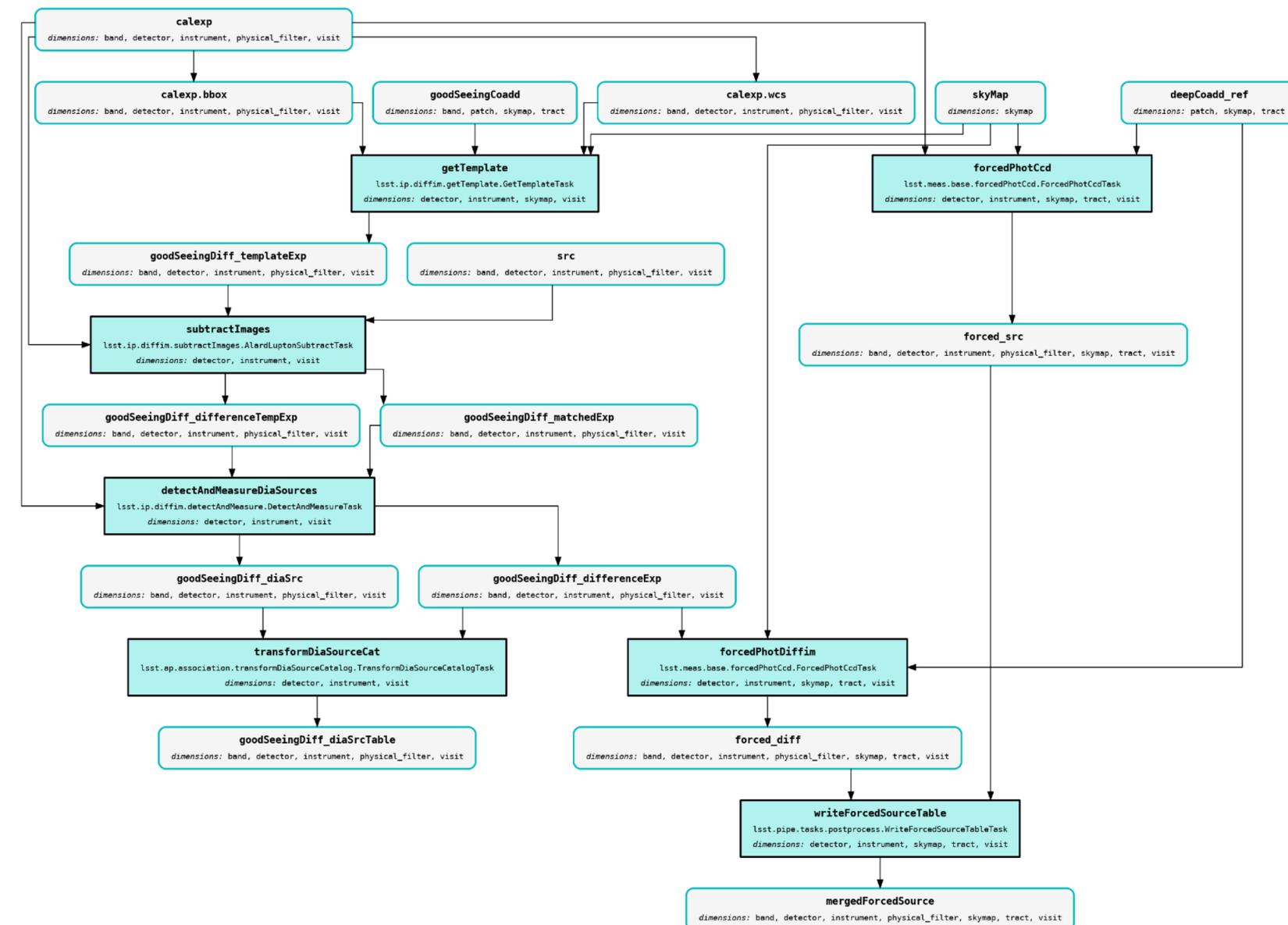
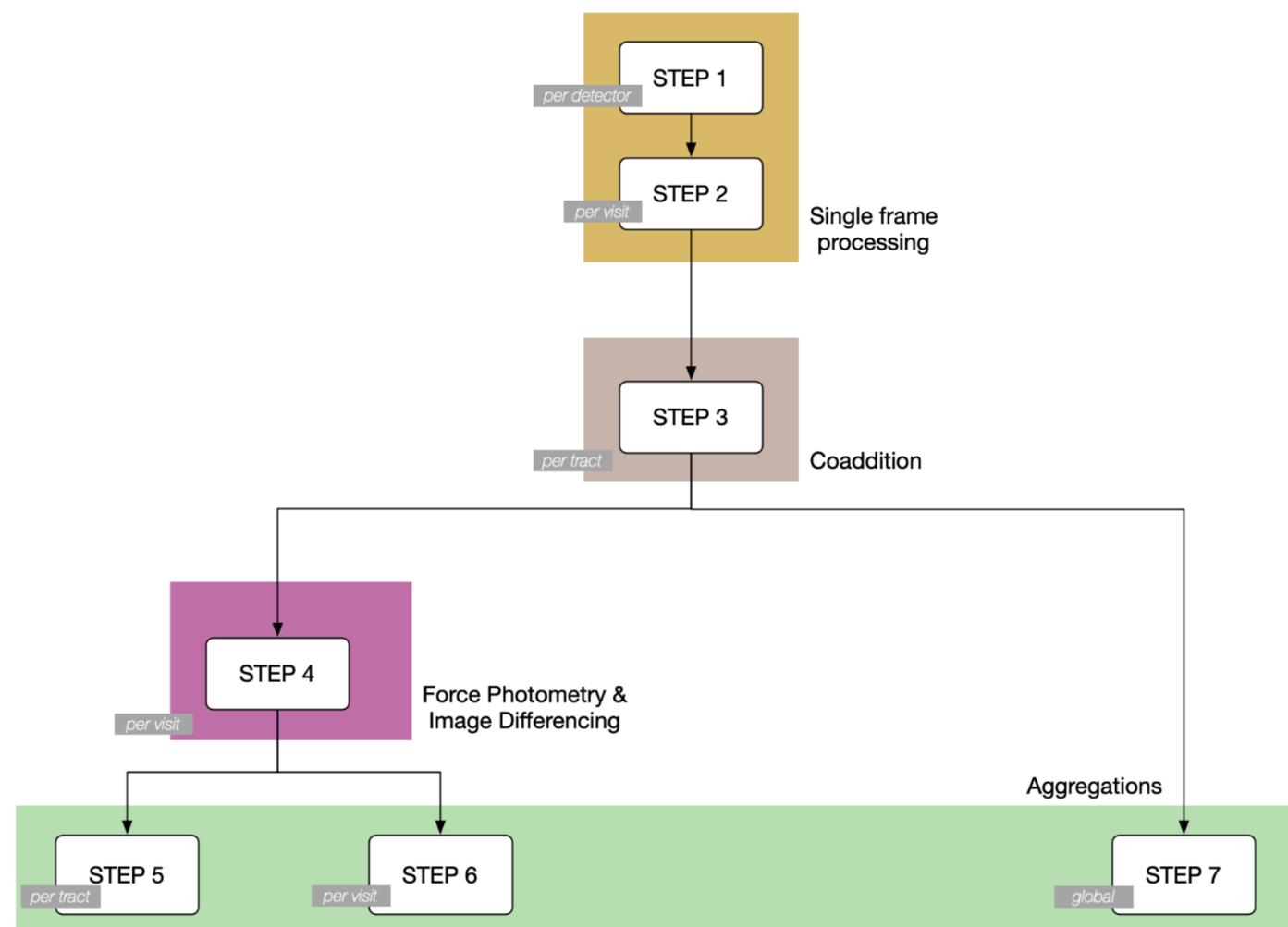
Objectifs de la pipeline de traitement des images :

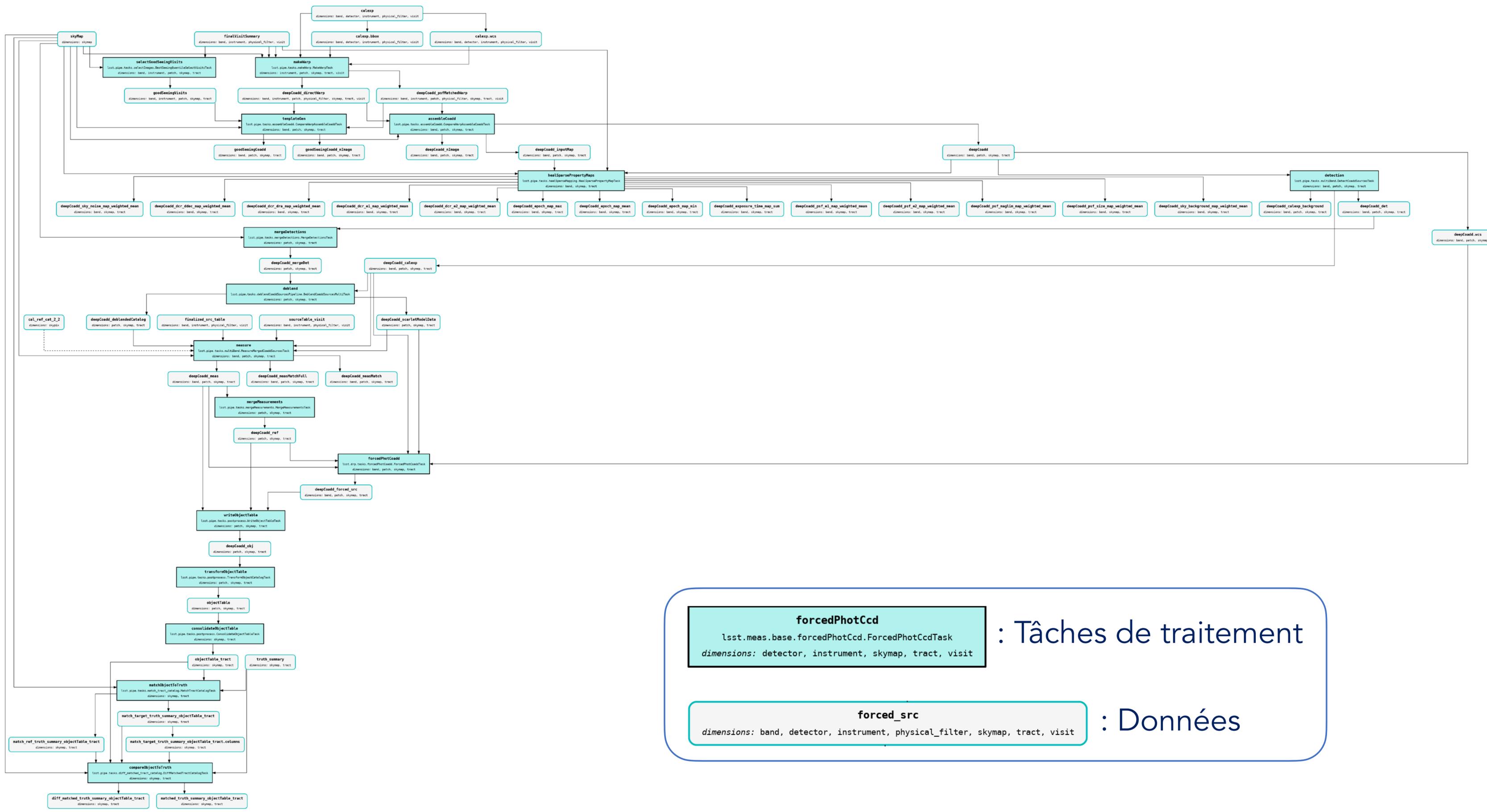
- **Traitement des images brutes** et calibration (suppression des effets instrumentaux, ...)
- Empilement d'images (**coaddition**) pour améliorer le rapport signal/bruit
- **Mesure de différences** entre images pour la détection d'objets transients
- Détection d'objets et mesure de leurs propriétés, création de catalogues



# Traitement des données

- Ensemble d'environ 80 tâches de traitement, regroupées en 7 étapes
- Chaque correspond à un workflow (dépendances entre les tâches via les données)





**forcedPhotCcd**  
 lsst.meas.base.forcedPhotCcd.ForcedPhotCcdTask  
 dimensions: detector, instrument, skymap, tract, visit

: Tâches de traitement

**forced\_src**  
 dimensions: band, detector, instrument, physical\_filter, skymap, tract, visit

: Données

Exigences fortes sur les performances au vu des volumes en jeu :

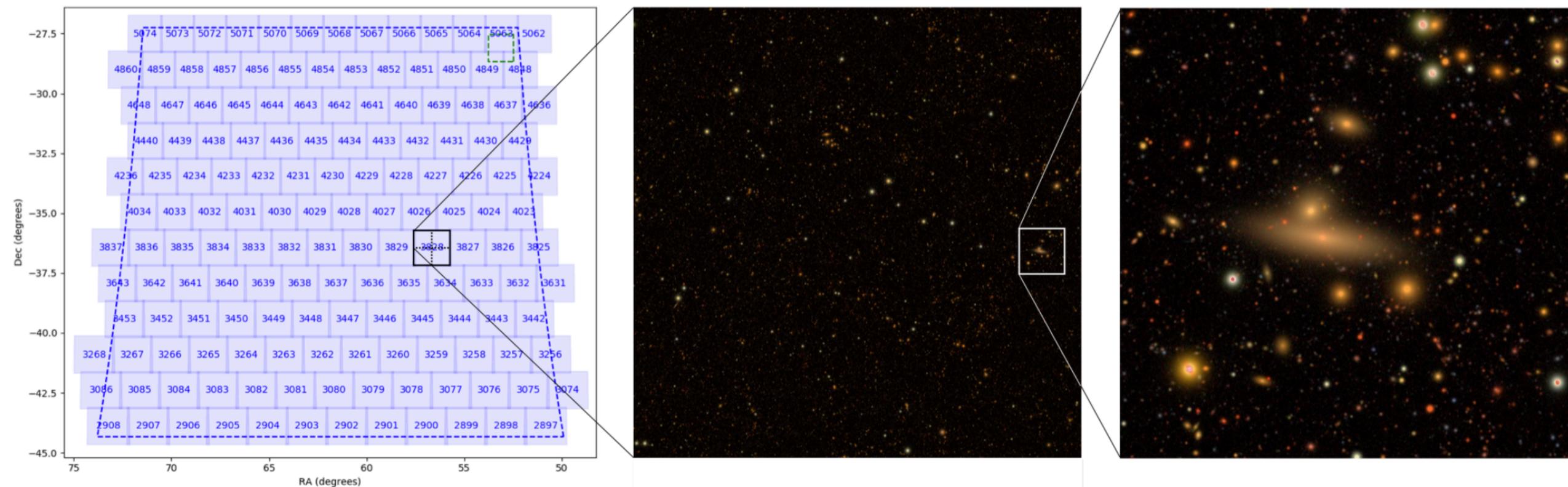
- 6.4 GB par exposition (images brutes)
- 2500 images (science + calibration) par nuit soit 16 TB (5 PB par an)
- 0.5 EB de données générées au total après 10 ans
- 15 PB de base de données après 10 ans (catalogues)

Contributions aux traitements :

- USA : 35%
- UK : 25%
- France (CC-IN2P3) : 40%

Exemple de la campagne de traitements DP0.2 (2022) :

- Traitement de données simulées produites par la Dark Energy Science Collaboration
- Equivalent à 0.5% du relevé, ou 10 nuits de prise de données
- 58 M de tâches exécutées, pour plus de 2 M d'heures CPU
- 3 PB de données générées



Fort enjeu sur la réduction de l'utilisation des ressources (temps CPU, mémoire, stockage) :

- Financier
- Environnemental
- Performances
  
- Code open source développé essentiellement par le projet Rubin (US)
- Intérêt commun sur l'optimisation

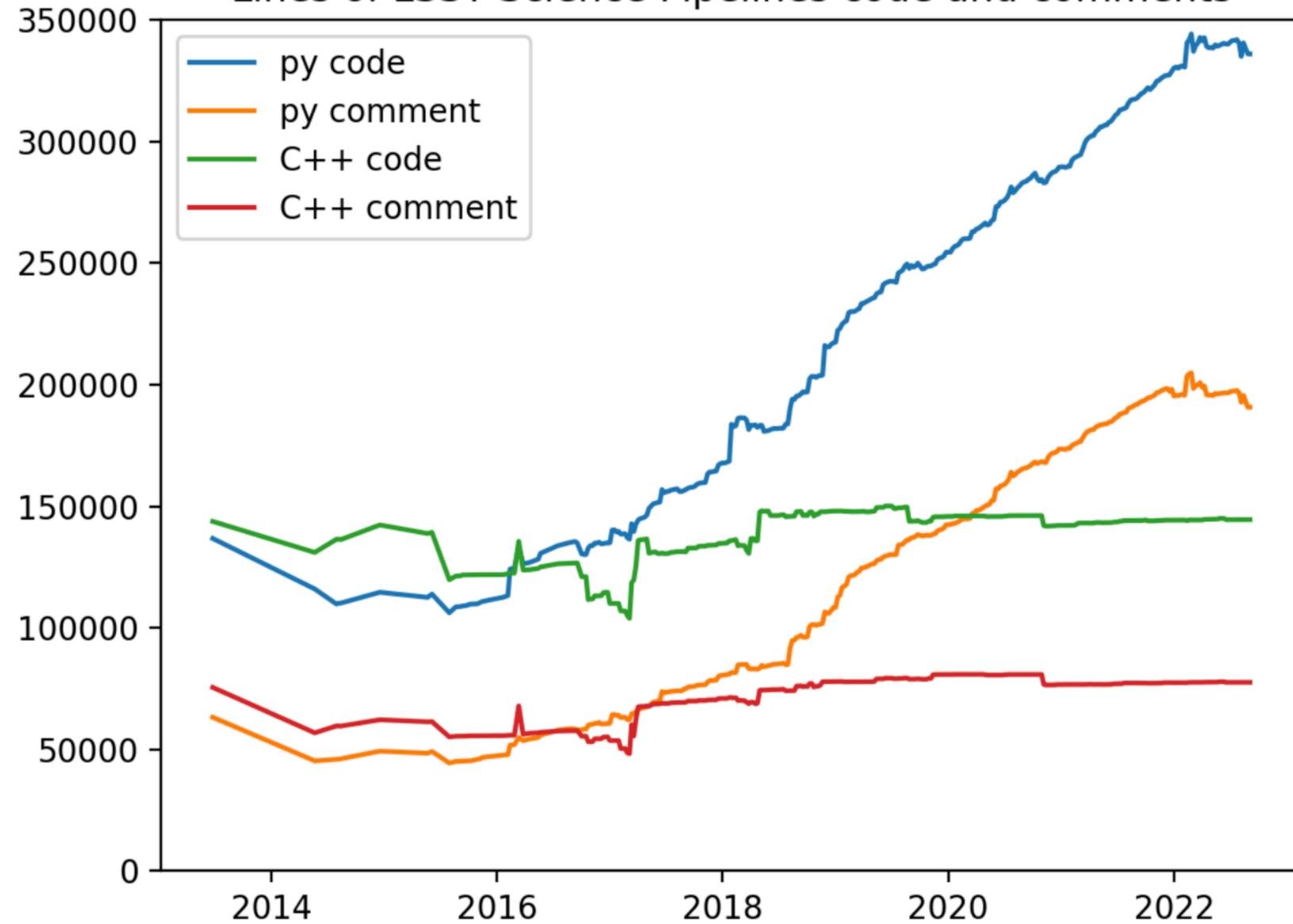
=> Projet d'optimisation de la pipeline de traitement des données porté à l'IN2P3 (projet RubinOP de la Mission pour les Initiatives Transverses et Interdisciplinaires)

Avant d'optimiser : besoin de mesures fiables pour comprendre ce qui doit (et peut) être optimisé

# Profilage de la pipeline

# Profilage de la pipeline

Lines of LSST Science Pipelines code and comments

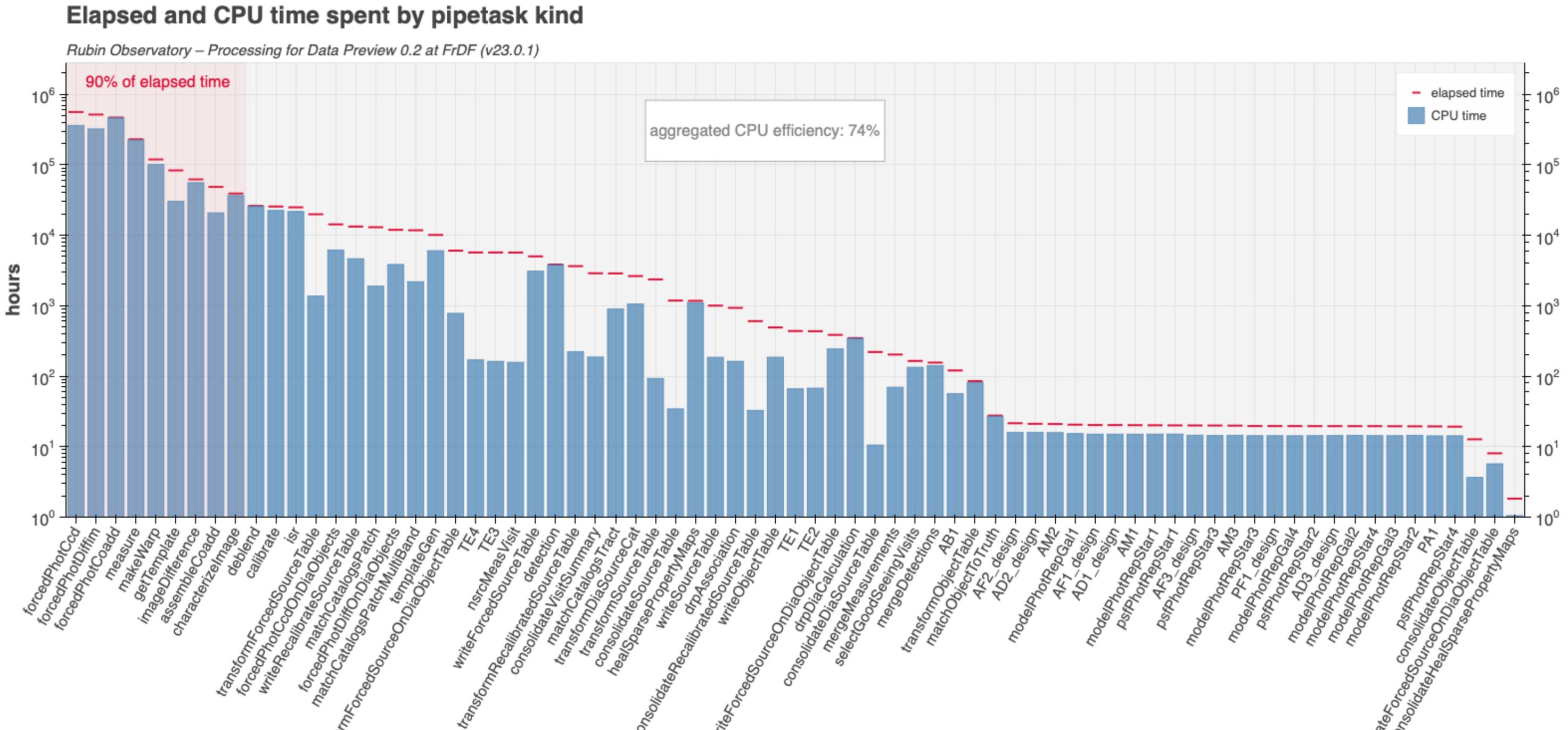


- Coeur du code en C++
- Surcouche Python
- Code disponible sur <https://github.com/lsst>
- Documentation : <https://pipelines.lsst.io>

Source: T. Jenness, Rubin Observatory

# Profilage de la pipeline

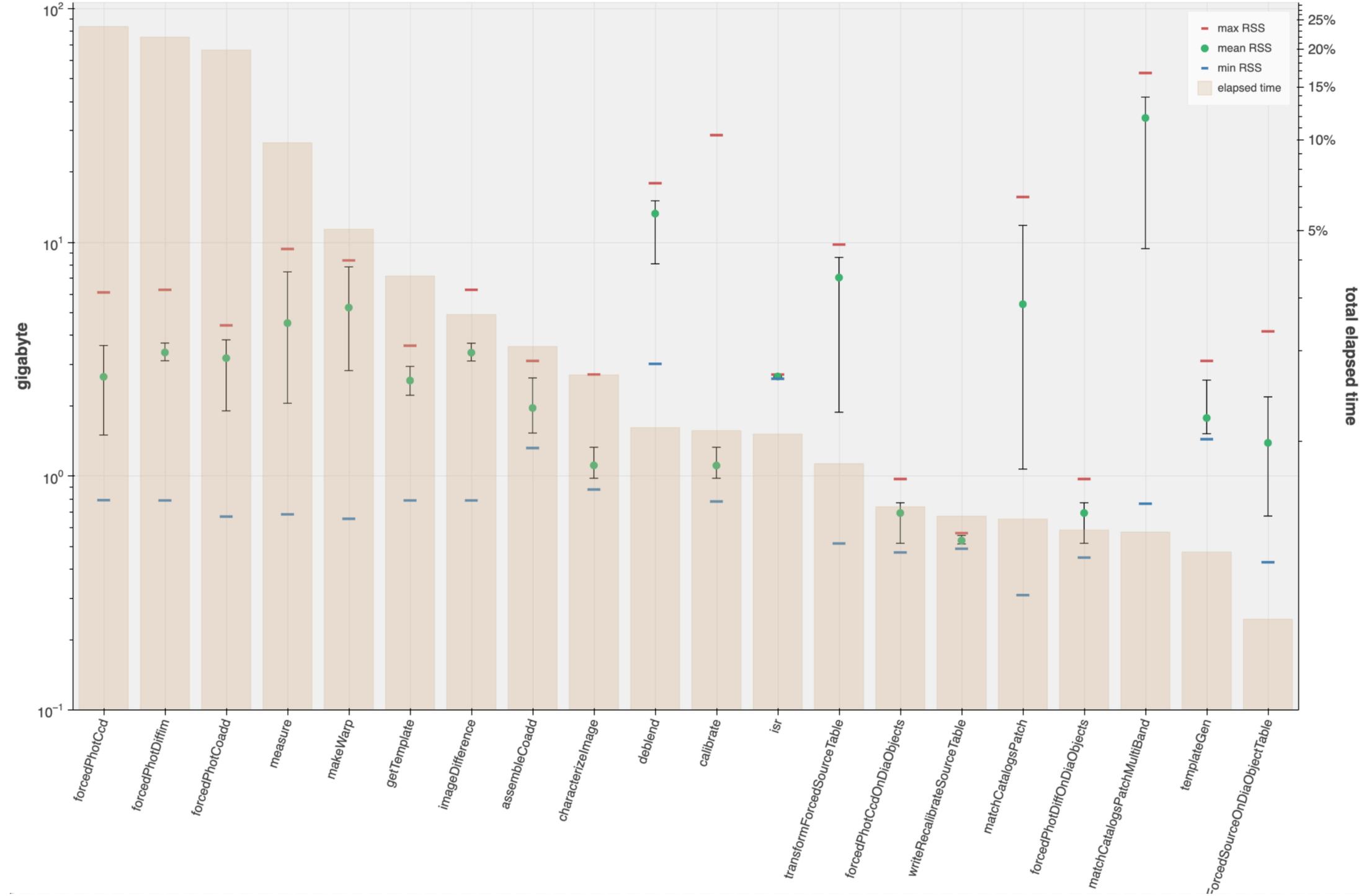
- Le framework d'exécution LSST fournit des mesures de temps de calcul et utilisation mémoire
- Déjà utilisé pour analyser l'utilisation des ressources lors de la campagne DP0.2 :



# Profilage de la pipeline

### Memory used by the most compute-intensive pipetasks

Rubin Observatory French Data Facility – processing for Data Preview 0.2 (v23)



Les limites de cette analyse :

- Uniquement une vue globale, pas de profilage fin sur chaque tâche
  - Besoin de mesurer l'utilisation CPU et mémoire au sein des tâches (par fonction ou ligne de code)
- Exécution à grande échelle, complexe et longue à reproduire
  - Besoin d'une pipeline de référence simple et rapide à exécuter, reproductible et représentative
  - [https://github.com/lsst/ci\\_hsc\\_gen3](https://github.com/lsst/ci_hsc_gen3) : petit jeu de données de test de la camera HSC (Subaru)
  - Traitement complet avec la pipeline LSST en quelques heures sur un coeur CPU

# Profilage de la pipeline

- [cProfile](#) : profilage CPU intégré à Python: `python -m cProfile myscript.py`
  - Nombre d'appels de chaque fonction et temps d'exécution
  - La sortie peut être exploitée par SnakeViz:

Reset Root

Reset Zoom

Style: **Icicle** ▾

Depth: **15** ▾

Cutoff: **1 / 1000** ▾

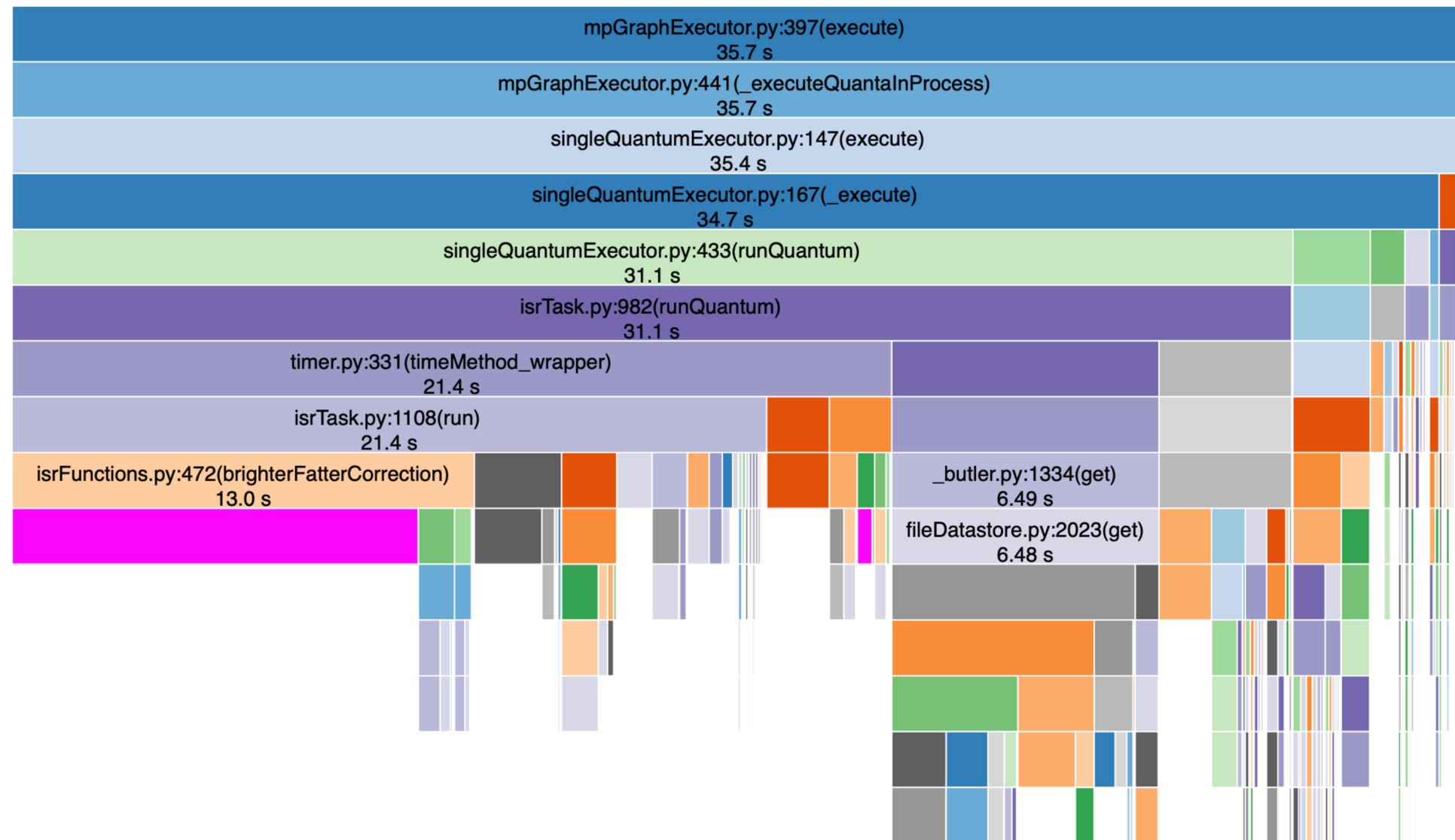
**Name:**  
<built-in method  
lsst.afw.math.\_math.convolve>

**Cumulative Time:**  
11.0 s (30.78 %)

**File:**  
~

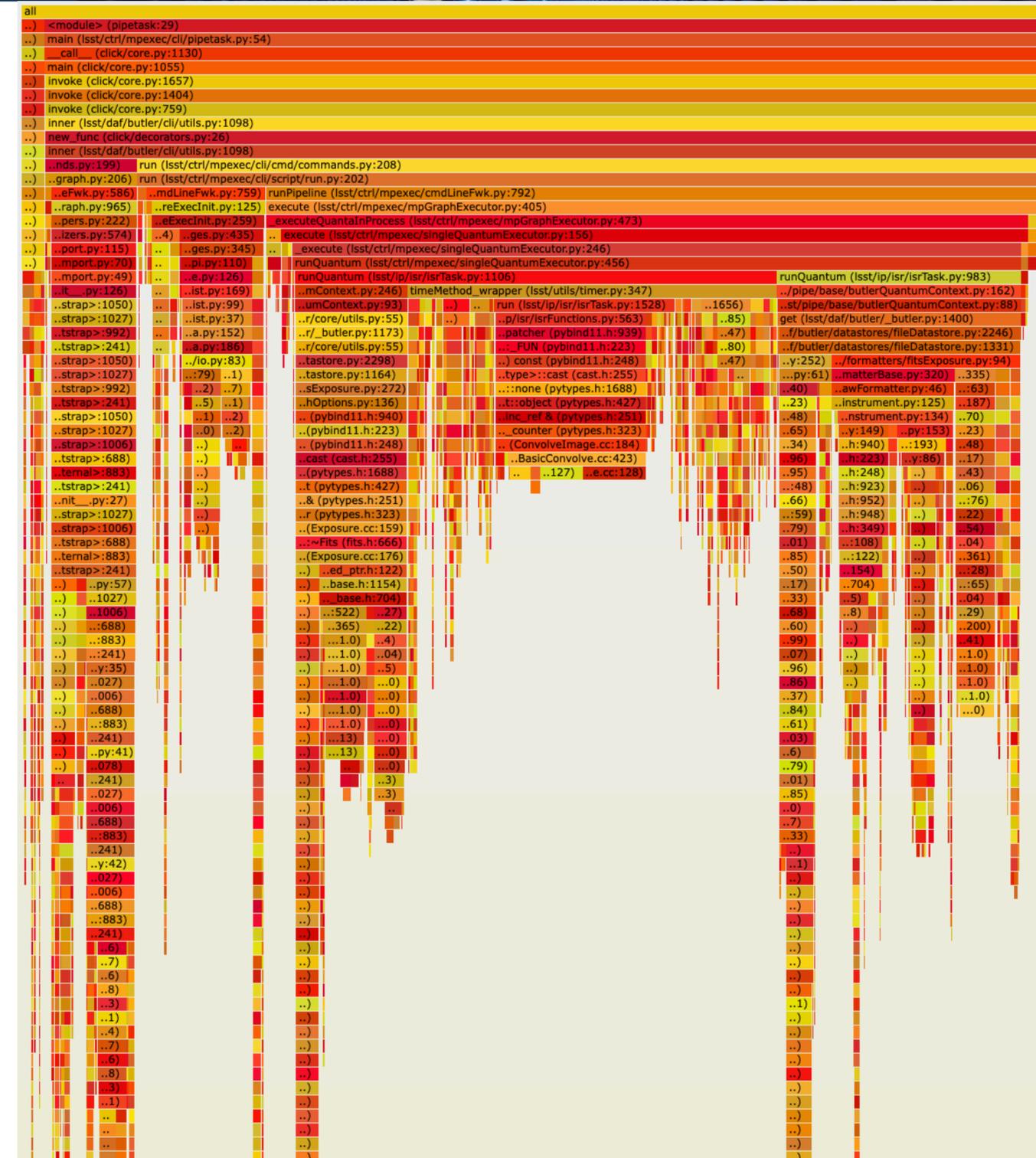
**Line:**  
0

**Directory:**



# Profilage de la pipeline

- [py-spy](#) : `py-spy record -- python myscrip.py`
  - Profilage CPU (fonctions)
  - Profilage des extensions C / C++
  - Génère un SVG de type "flamegraph"
- De nombreux autres outils disponibles :
  - [Scalene](#)
  - [Austin](#)
  - [Pyinstrument](#)
  - ...



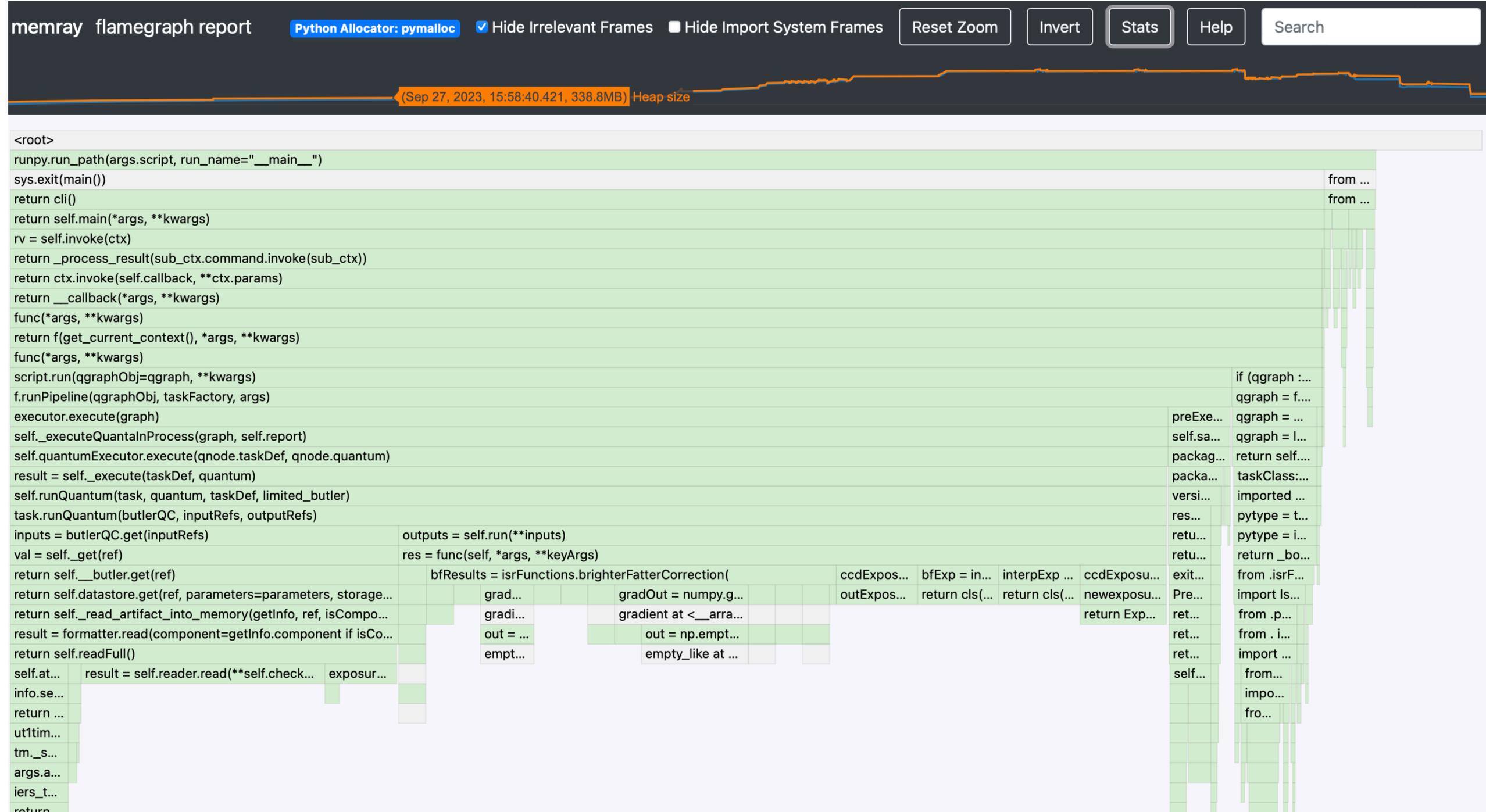
# Profilage de la pipeline

- [Memray](#) : profilage mémoire
  - Peut être exécuté directement en ligne de commande : `memray run myscript.py`
  - Ou directement dans le code Python :

```
import memray
with memray.Tracker("output.bin") :
```
  - Nombre d'allocations et quantité de mémoire pour chaque fonction
  - Profilage des extensions C / C++

# Profilage de la pipeline

- Sortie "flamegraph" :



# Profilage de la pipeline

- Sortie "stats" :

```
→ memray stats caracImage_oct27_full_nonative.bin
└─ Total allocations:
    61840023

└─ Total memory allocated:
    17.855GB

└─ Histogram of allocation size:
    min: 0.000B

    < 6.000B : 56584
    < 36.000B : 15149095
    < 224.000B : 41648132
    < 1.329KB : 3470307
    < 8.077KB : 1289892
    < 49.068KB : 193616
    < 298.075KB : 31804
    < 1.768MB : 479
    < 10.741MB : 56
    ≤ 65.250MB : 58

    max: 65.250MB

└─ Allocator type distribution:
    MALLOC: 60200868
    POSIX_MEMALIGN: 726010
    CALLOC: 642234
    REALLOC: 200092
    ALIGNED_ALLOC: 70819

└─ Top 5 largest allocating locations (by size):
    - measure: ... path ... /DarwinX86/meas_modelfit/g93c4d6e787+0bdc8f6279/python/lsst/meas/modelfit/cmodel/cmodelContinued.py:63 → 11.449GB
    - measure: ... path ... /DarwinX86/meas_base/g67924a670a+6c888dcbf9/python/lsst/meas/base/wrappers.py:43 → 1.369GB
    - _solve: ... path ... /DarwinX86/meas_extensions_gaap/g96f6979c0c+f64f51c0d6/python/lsst/meas/extensions/gaap/_gaussianizePsf.py:314 → 442.305MB
    - getNoiseGenerator: ... path ... /DarwinX86/meas_base/g67924a670a+6c888dcbf9/python/lsst/meas/base/noiseReplacer.py:366 → 384.006MB
    - cosmicRay: ... git ... /pipe_tasks/python/lsst/pipe/tasks/repair.py:183 → 384.006MB

└─ Top 5 largest allocating locations (by number of allocations):
    - measure: ... path ... /DarwinX86/meas_modelfit/g93c4d6e787+0bdc8f6279/python/lsst/meas/modelfit/cmodel/cmodelContinued.py:63 → 50443948
    - measure: ... path ... /DarwinX86/meas_base/g67924a670a+6c888dcbf9/python/lsst/meas/base/wrappers.py:43 → 1802230
    - _gaussianizeAndMeasure: ... path ... /DarwinX86/meas_extensions_gaap/g96f6979c0c+f64f51c0d6/python/lsst/meas/extensions/gaap/_gaap.py:605 → 1074060
    - convolve: ... path ... /DarwinX86/meas_extensions_convolved/g42fff21dfb+02b235c229/python/lsst/meas/extensions/convolved/convolved.py:459 → 784602
    - selectSources: ... path ... /DarwinX86/meas_algorithms/g8494ff85c9+4cea07f1a2/python/lsst/meas/algorithms/objectSizeStarSelector.py:380 → 752856
```

# Profilage de la pipeline

- Profilage ligne par ligne :
  - [line\\_profiler](#) (CPU) et [memory\\_profiler](#) (mémoire)
  - Plus précis mais plus complexe
  - Pour plus tard ?



# Résumé et perspectives

- CC-IN2P3 : 40% des traitements d'images de l'observatoire Vera C. Rubin
- Des besoins importants en ressources de calcul et stockage
- Une pipeline de traitement complexe
- Début d'un projet d'optimisation :
  - Prise en main d'une pipeline de test
  - Exploration des outils de profilage
- Perspectives :
  - Mise en place d'outils de profilage et d'analyse (sélection de métriques, dashboards)
  - Etude complète CPU + mémoire de quelques tâches
  - Analyse des parties du code consommatrices
  - Travail d'optimisation avec les équipes de développement