Gaia data simulations: a powerful tool to prepare for the Gaia scientific exploitation

X. Luri & C. Babusiaux
On behalf of the DPAC CU2

Index

- Context: history, of the simulator. The CU2 and the DPAC
- Overview of the simulator: present status of the GaiaSimu library and the data generators
- Future: development after launch
- The simulator and the scientific exploitation of Gaia. The CU9
- Conclusions

A bit of history: 10 years of Gaia simulations

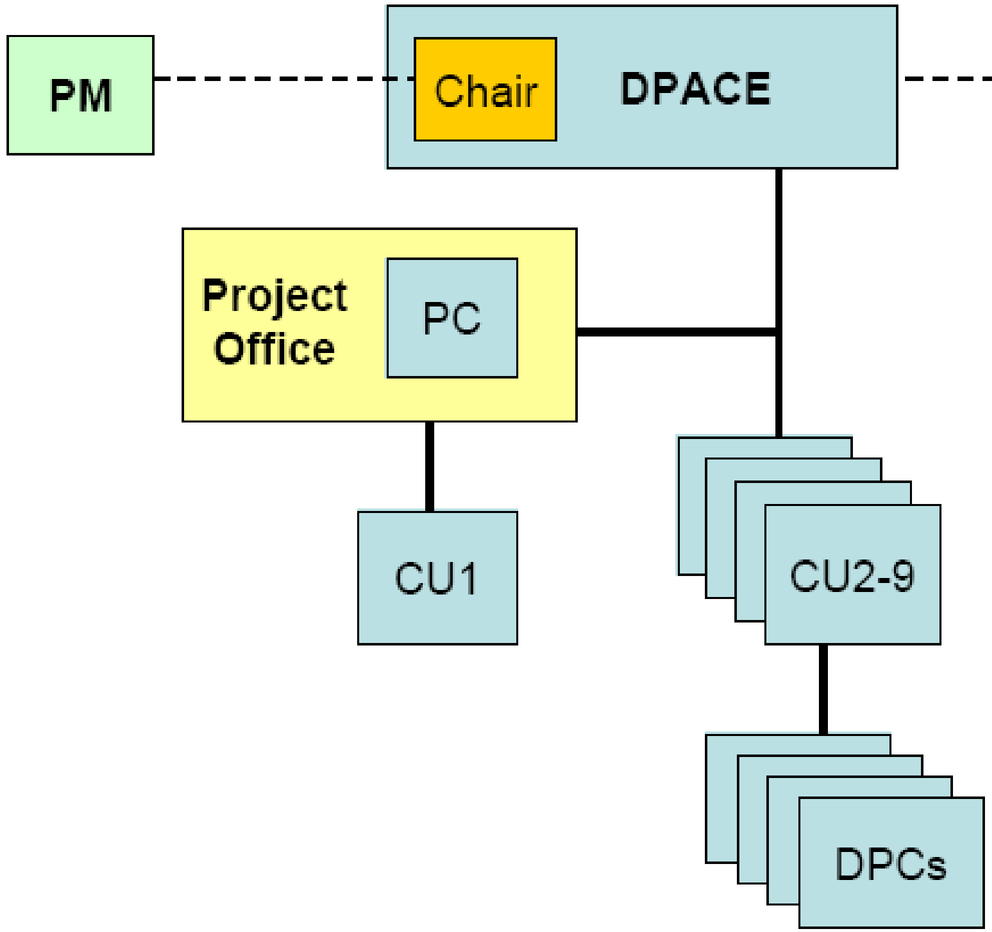
The development of the Gaia simulator started in 2000 at the University of Barcelona; these early versions were created to feed test data to the first prototypes of the data reduction software (the GDAAS system) and were the seed for the present system.

Later on, in 2001, the Gaia Simulation Working Group (SWG) was created in order to develop a system with a wider scope, providing simulated data for the test and validation of the data reduction system.

In 2006 the Gaia Data Processing and Analysis Consortium (DPAC) was created, and the responsibility of the development of the simulator was transferred to its Coordination Unit 2 (CU2). Since then the CU2 has been refining and upgrading the system.

We can therefore celebrate today the 10th anniversary of the Gaia Simulator

The DPAC and the CU2



Coordination units

Data processing centres

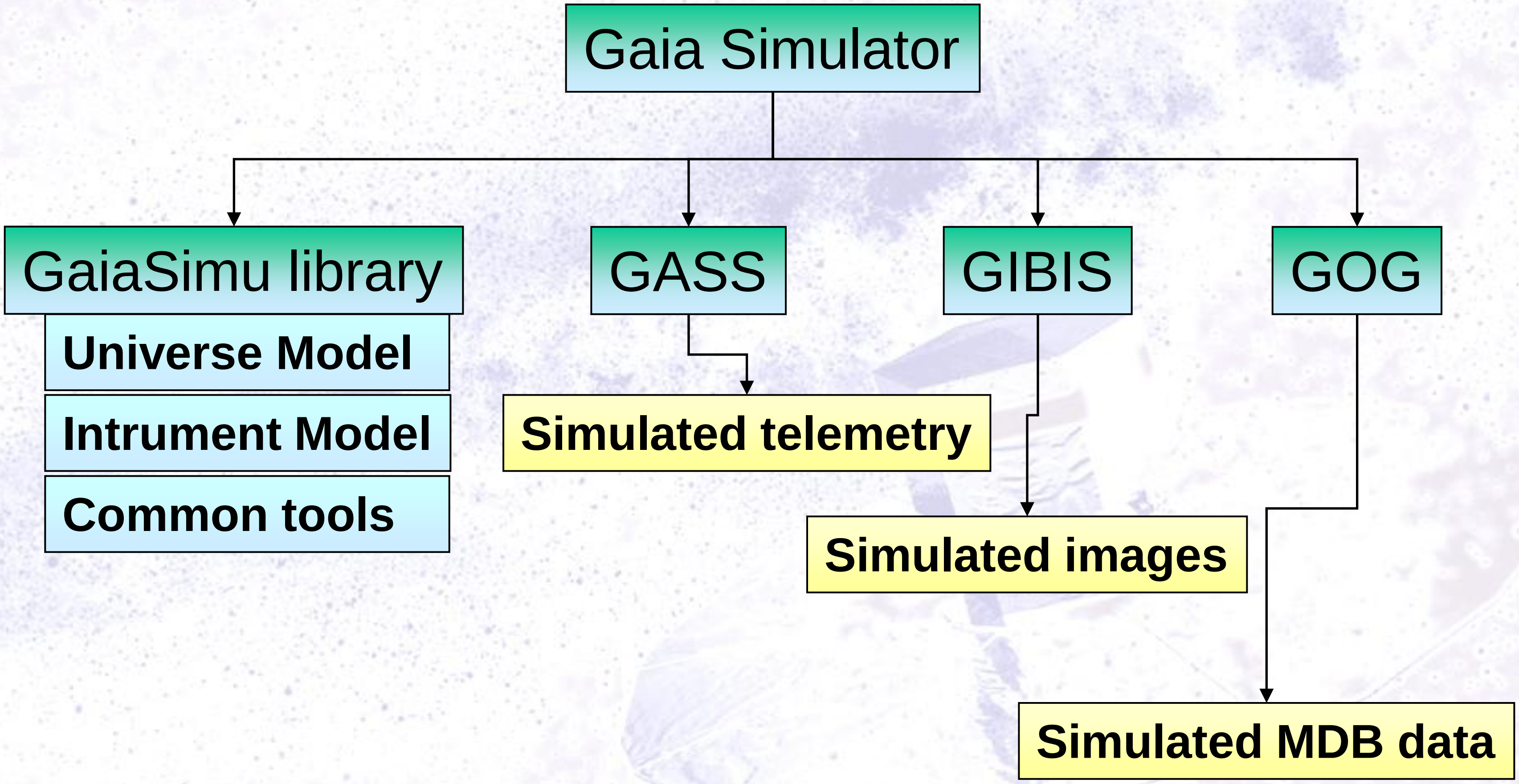
The CU2 operates in close coordination with the rest of the DPAC:

- Requests for simulated data are sent in each cycle by the different CUs in DPAC
- The development of the different modules of the simulator is jointly agreed, taking into account each CU's needs, their global priorities within the overall consortium, the availability of the simulation models (which can depend on industrial tests or other CU accuracy models) and the available CU2 manpower.

Therefore, the CU2 task is to cover the simulation needs for the work of other CUs, ensuring that reliable data simulations are available for the development and testing of the various stages of the data processing development.

Thus, presently, the purpose of the simulator is closely tied to the needs of the preparation of the Gaia data reduction. For the moment, the scientific issues of the Gaia data exploitation are not in the forefront.

The Gaia simulator: overview

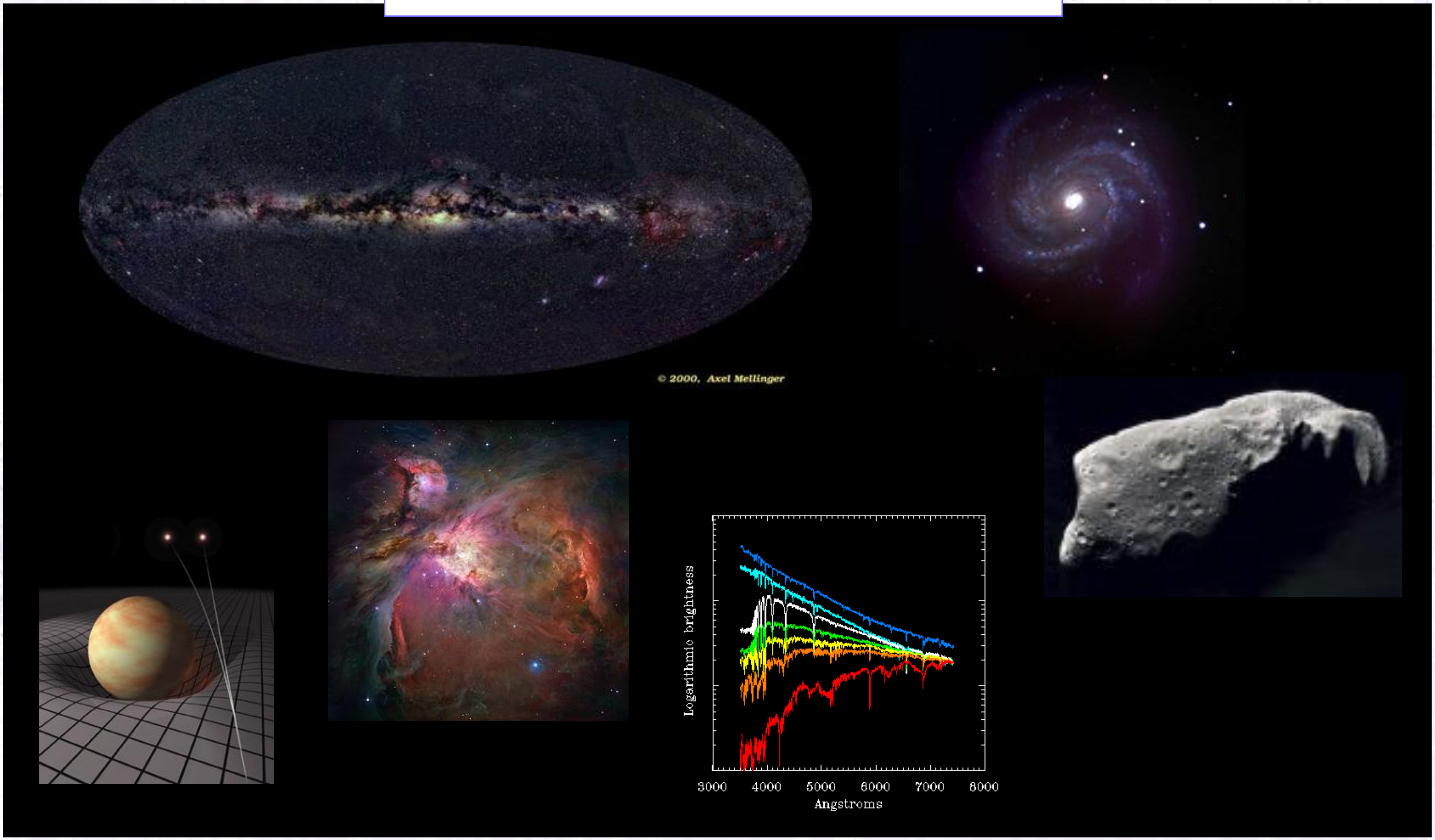


The GaiaSimu library

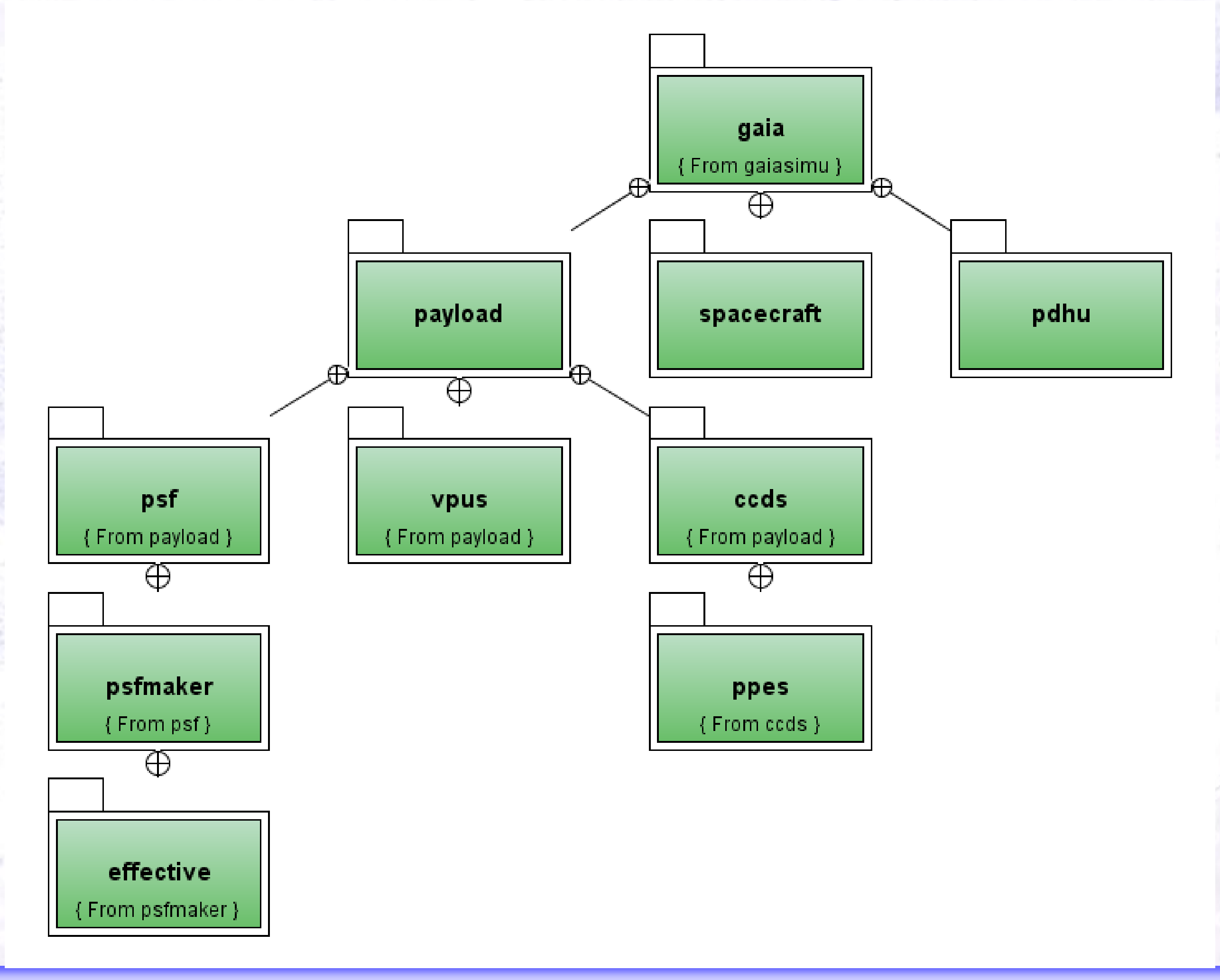
This library is the basis for the development of the Gaia simulator. It contains the common models used by the data generators to produce simulations of the Gaia observations. It is composed of three parts:

1. The instrument model: models of the Gaia instruments and elements of the spacecraft
2. The universe model: model of the objects in the sky that Gaia will observe, with all its physical characteristics
3. Common tools: a toolbox for use throughout the simulator

The Universe Model



The Instrument Model



The data generators

The simulator comprises three *data generators*, software components that use the GaiaSimu library to produce specific types of data for the DPAC. The three data generators are:

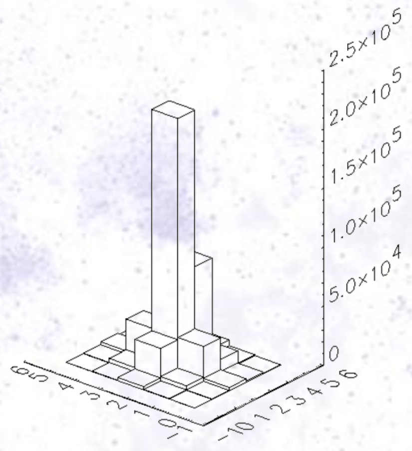
- The GAia System Simulator (GASS)
- The Gaia Instrument and Basic Image Simulator (GIBIS)
- The Gaia Object Generator (GOG)

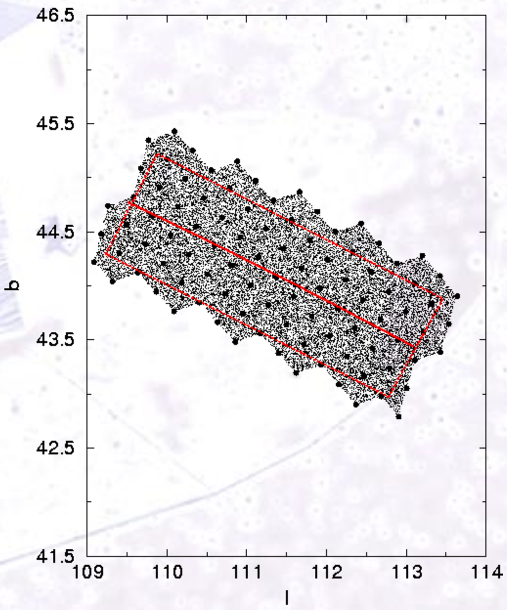
Data generators: GASS

This data generator provides simulations of the telemetry stream of the mission based on some simplifications of the instrument and Universe models allowing a large amount of data to be simulated over a significant period of time.

GASS provides realistic data for:

- Predictions to be used for mission design.
- Filling of test databases
- Testing of core reduction algorithms
- Evaluation of mission performances, in particular for peculiar objects (binary stars, NEO's, extrasolar planets,...)





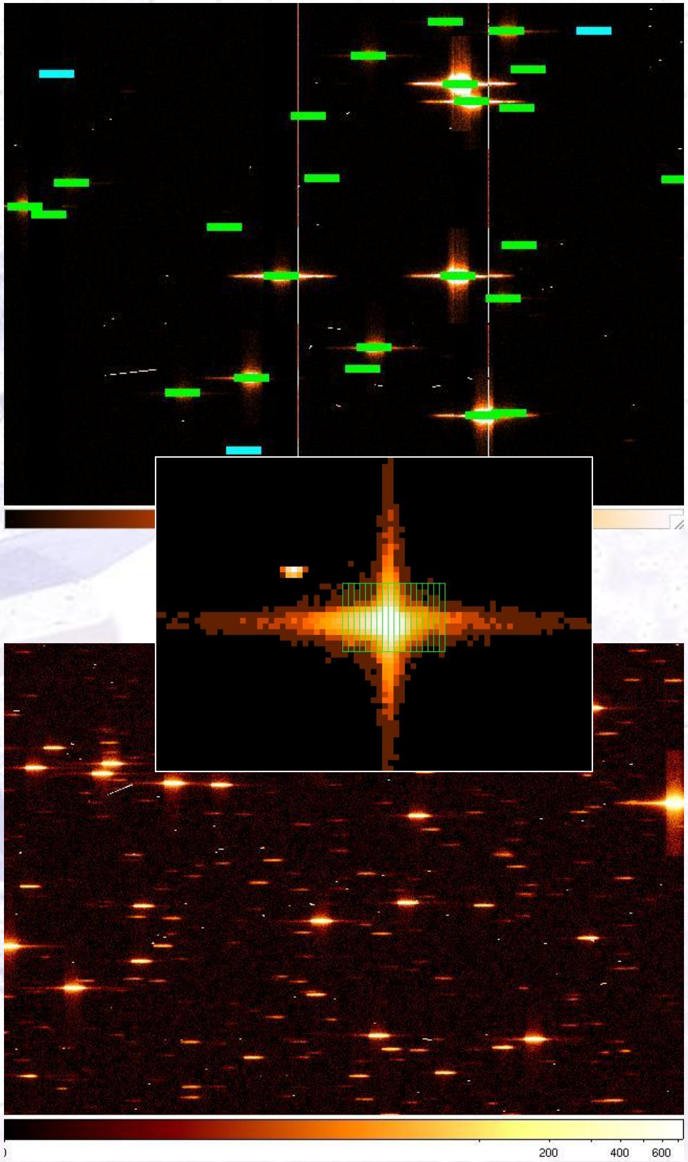
Data generators: GIBIS

This data generator provides simulations of the data at the pixel level. The resulting simulations are as realistic as possible and restricted, due to computing time limitations, to a region of a sky over a short period of time.

GIBIS provides realistic CCD images for:

- Instrument design
- PDHE design
- Detection & selection algorithm development
- Scientific mission design
- Detailed analysis of reduction algorithms

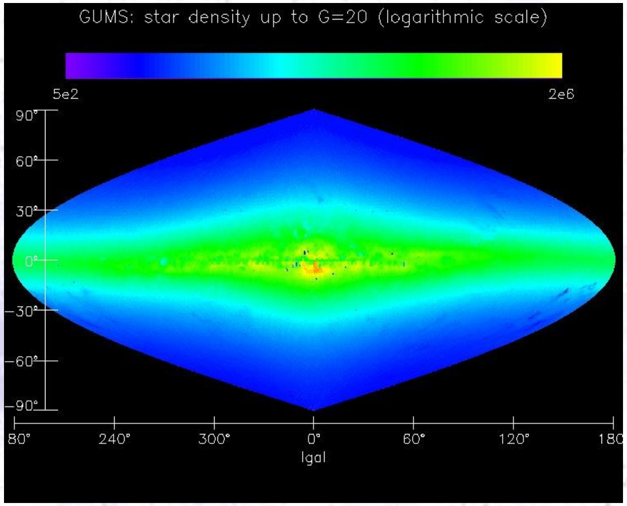
Specially for the study of crowding, background and extended objects

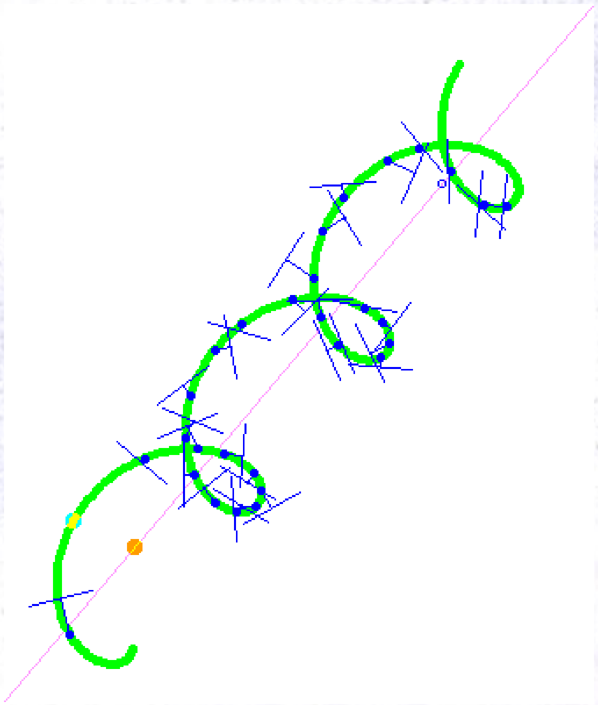


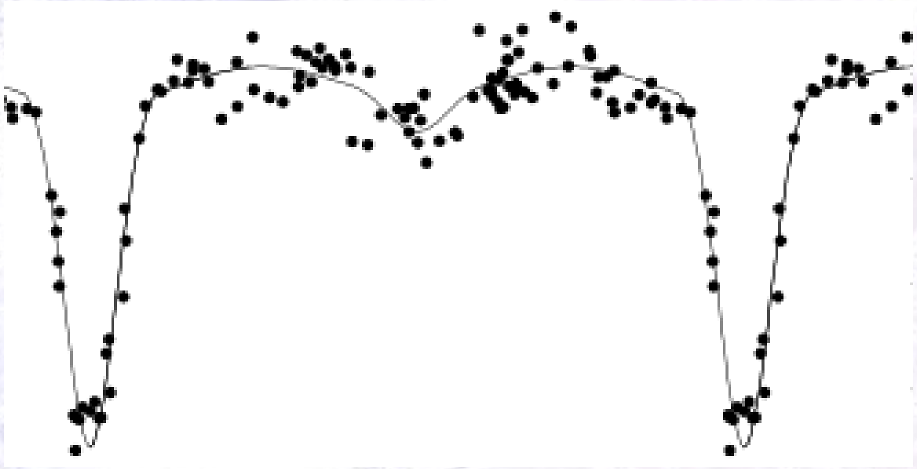
Data generators: GOG

This data generator provides simulations of number counts and lists of observable objects from the Universe Model and, for a given source or a collection of sources, simulations of intermediate and end-of-mission Gaia data.

[GOG aims to simulate the contents of the MDB at any stage of the data processing.]







Summary

The Gaia Simulator code amounts today to more than **100.000 lines of code** and has produced **several terabytes of simulated data** in the last years that have been used for mission design and development and testing of the initial versions of some reduction algorithms.

What next: is there life after launch?

Some DPAC CUs will start working with real data shortly after launch: CU3, CU5 & CU6.

→ These CUs may need simulations to understand the real data

Other DPAC CUs will not receive real (processed) data until about 1.5 years after launch.

→ These CUs will still need simulations during this time, and later on to check their results

Furthermore, CU9 will start working on the preparation of the Gaia catalogue after launch

→ Simulations will be needed to test the catalogue generation, the database system and the validation procedures

But more important, there will be releases of intermediate data during the mission

→ Can we use the simulator for the scientific exploitation of these data

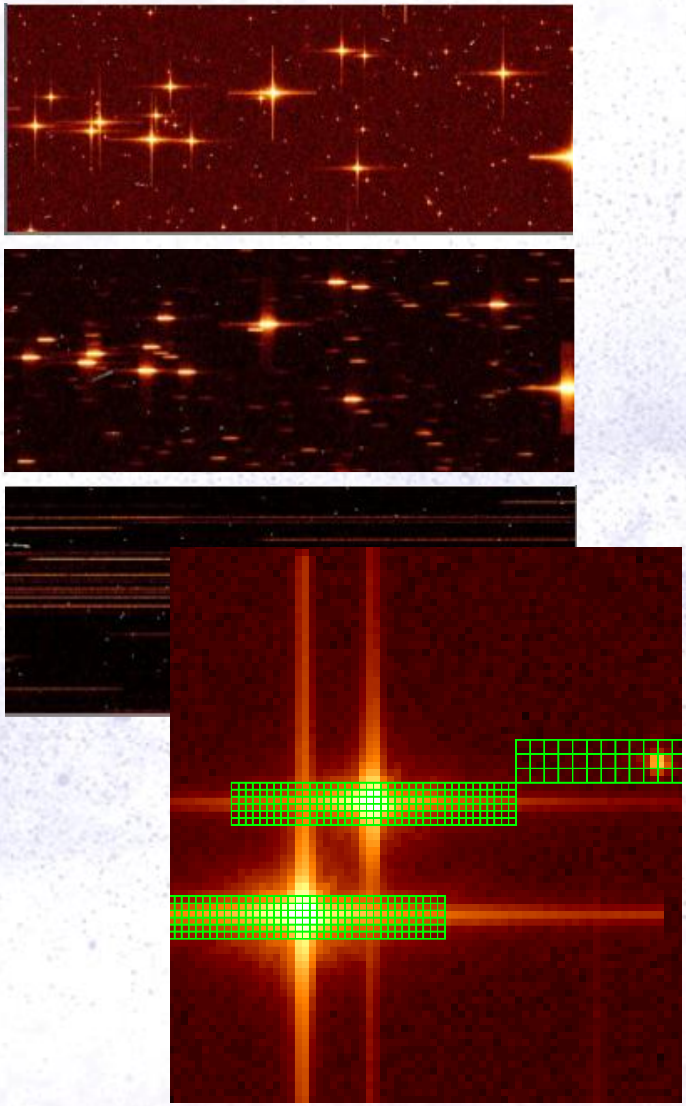
Scientific use of the simulator: overview

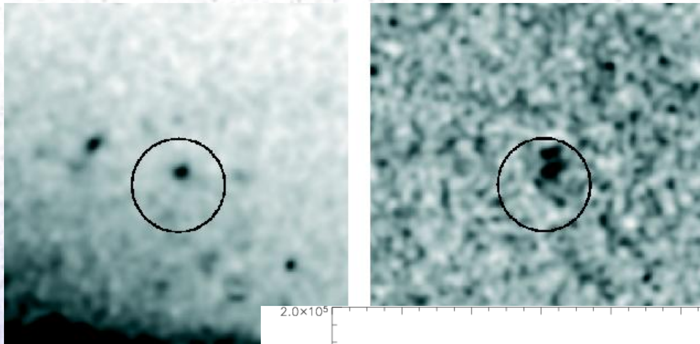
The Gaia simulator is now a mature and powerful tool that can be useful beyond the preparation of the data reduction software in DPAC; here we list some key areas where it can be of use for the preparation of the scientific exploitation of the Gaia data, and the exploitation itself.

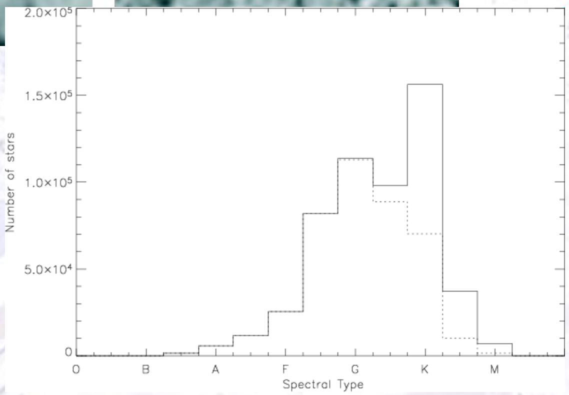
Detailed simulation of Gaia observations

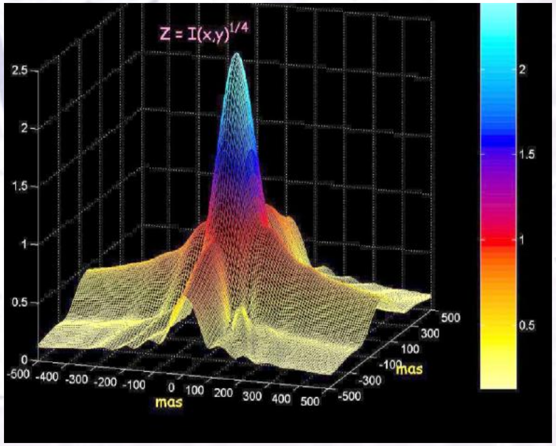
- Improve analysis of key observations (e.g. QSO for reference frame, Solar System Objects,…)
- Testbed for image reconstruction algorithms (e.g. for shape reconstruction of extended objects or multiple systems)
- Testbed for improved centroiding algorithms for special types of objects (e.g. close binaries)
- Etc.

GIBIS + the instrument model






2.0×10^5
1.5×10^5
1.0×10^5
5.0×10^4
0
Number of stars
O
B
A
F
G
K
M
Spectral Type


Z = I(x,y)^1/4
mas
mas

Generation of Gaia catalogue mock-ups

- Development and test of data extraction and data mining software
- Test and improvement of scientific analysis algorithms for specific types of objects
- Test and improvement of scientific models (e.g. galaxy models)
- Tool for the assessment of the presence of systematic errors
- Tool for statistical checks of the Gaia data (e.g. completeness, biases, etc.).
- Etc.

GOG + the universe model

The Universe Model

Solar System

- Sun, Earth, Moon
  - (not for observation)
- Planets and satellites
- Minor bodies
  - Asteroids
  - Comets
  - Kuiper belt
- Other components
  - Zodiacal light
  - Solar wind
  - Etc.

Our Galaxy

- Field stars
  - **“Normal”**
  - **Multiple systems**
  - **Variable stars**
- Stellar clusters
  - Open clusters
  - Globular clusters
  - OB associations
  - Stellar streams
- Extended objects
  - Planetary nebula
  - HII regions
  - Reflection nebula
- Other components
  - Galactic diffuse light
  - **Unresolved background stars**
  - Extrasolar planets

Extragalactic objects

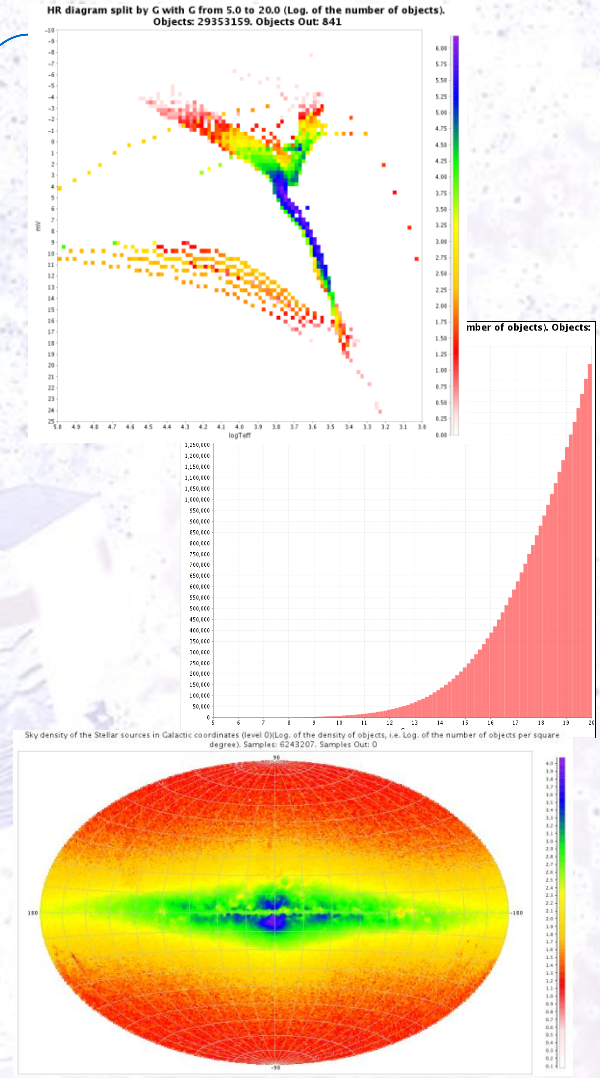
- Galaxies with resolved structure (List)
  - Field stars
  - Stellar clusters
  - Surface brightness
  - Supernovae
- Galaxies with unresolved structure
  - Surface brightness
  - Supernovae
- QSO
- Other components
  - Diffuse extragalactic light

Universe Model

+

GOG Analysis Tool (GAT)

→ Statistics of tailored catalogues!



Scientific use of the simulator: GASS

GASS will probably useful only in the context of DPAC, but if raw telemetry simulation is useful for some scientific purpose please let us know.

The CU9 and the simulator

The CU9 (to become operative in the near future) will be in charge of producing the Gaia catalogue and making it available to the community.

Thus, it could benefit of or even require simulations for its activities, specially from GOG.

One could even consider distributing simulations through CU9 as preparation for the data delivery, or even as a complement to it.

Concluding remarks

We are still in the middle of the fray to get the data reduction system up and running in time for the arrival of the Gaia data.

However, the DPAC members should also get ready for the scientific use these data, reaping the benefits of years of experience in the preparation of the mission. The simulator can play a role in this process.

Thank you!

```
// New time intervals for the NFIs
t1= NFI*(time.getGMTime()/NFI);
t2= t1 + NFI;
t3= t2 + NFI;

// Arrays to build the splines
double[] t= new double[nPoints/4 +1];
double[] n1= new double[nPoints/4 +1];
double[] n2= new double[nPoints/4 +1];
double[] n3= new double[nPoints/4 +1];


//
// First spline
//

// Generate the three Markov chains
double[] c1 = generateNFIChain(activeNFI.NFI1, NoiseChain.x);
double[] c2 = generateNFIChain(activeNFI.NFI1, NoiseChain.y);
double[] c3 = generateNFIChain(activeNFI.NFI1, NoiseChain.z);

// Select one point in four to fill the splines
for(int i=0; i<nPoints+1; i=i+4){
    t[i/4] = (double)( t1 + i*tStep );
    n1[i/4] = 0.5 * c1[i];
    n2[i/4] = 0.5 * c2[i];
    n3[i/4] = 0.5 * c3[i];
//System.out.println(t[i/4]/1.e9 + " " + 2.*n1[i/4]
}

s1n1.setSample(t,n1);
s1n2.setSample(t,n2);
s1n3.setSample(t,n3);

//
// Second spline
//

// Generate the three Markov chains
c1 = generateNFIChain(activeNFI.NFI2, NoiseChain.x);
c2 = generateNFIChain(activeNFI.NFI2, NoiseChain.y);
c3 = generateNFIChain(activeNFI.NFI2, NoiseChain.z);

// Select one point in four to fill the splines
for(int i=0; i<nPoints+1; i=i+4){
    t[i/4] = (double)( t2 + i*tStep );
    n1[i/4] = 0.5 * c1[i];
    n2[i/4] = 0.5 * c2[i];
    n3[i/4] = 0.5 * c3[i];
}

s2n1.setSample(t,n1);
s2n2.setSample(t,n2);
s2n3.setSample(t,n3);
```