Jean Ballet, CEA Saclay SLAC, 23 May 2005

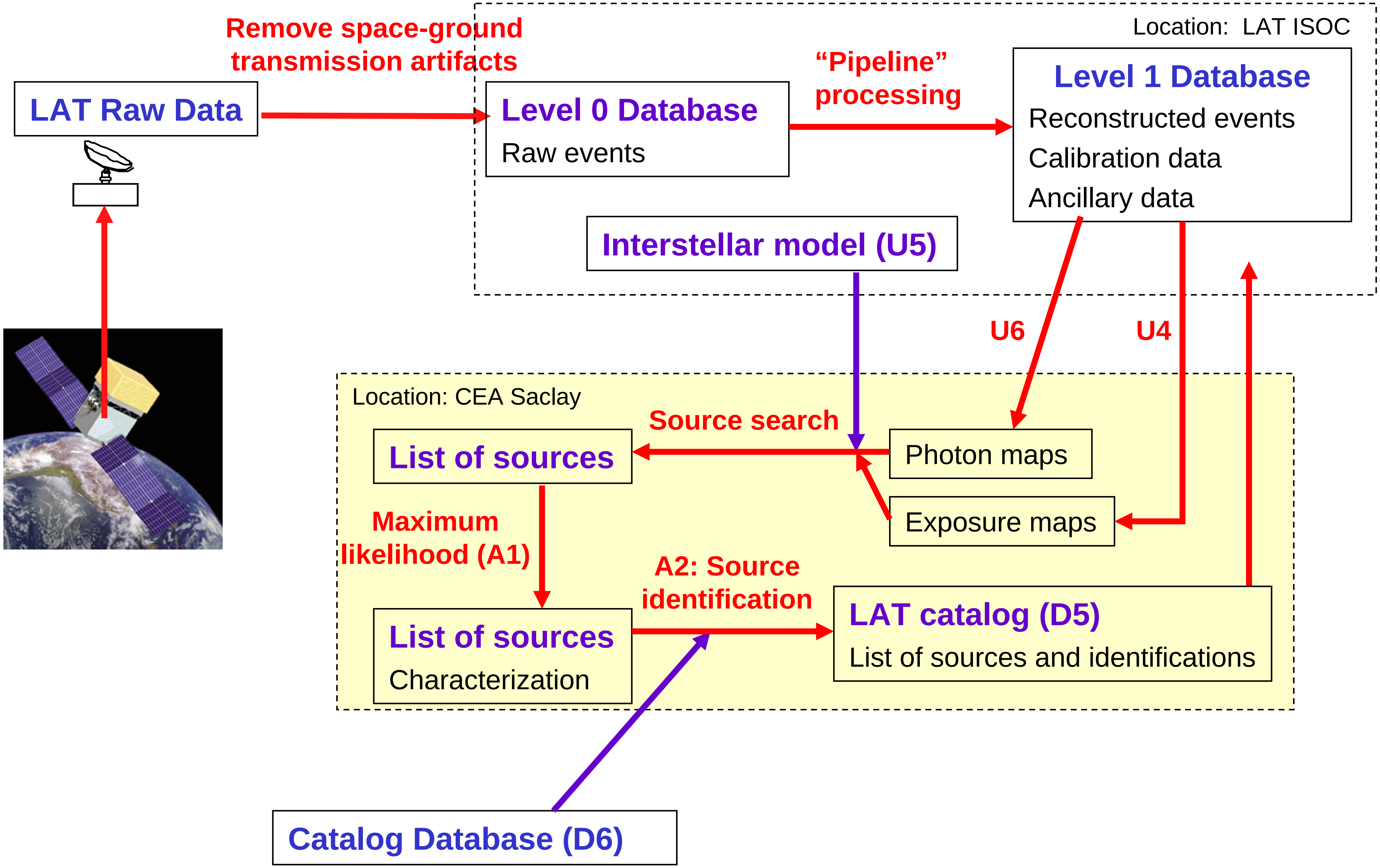
Source catalog generation

Aim: Build the LAT source catalog (1, 3, 5 years)

Four main functions:

- ✓ Find unknown sources over the whole sky (output: list of positions). This is the purpose of the next three presentations.
- ✓ Localize sources. Output: list of precise positions (and uncertainties)
- ✓ Characterize sources (significance, flux, spectrum, variability). This is no different from studying already known sources, and can be done using the likelihood method.
- ✓ Identify sources (find counterparts in existing external catalogs). This is the purpose of J. Knödlseder’s presentation.

Catalogue production pipeline



Catalog pipeline. Sequence

Aim: Implement automatic loop to find and characterize the sources

Minimal features:

1. Detect sources over current diffuse model
2. Get a precise position
3. Run Likelihood on all sources to get precise flux, spectrum and significance
4. Split into time pieces to get light curve
5. Run sourceIdentify to get counterparts

- ✓ Task scheduling tool (like OPUS) for distributing work over CPUs
- ✓ Simple database for bookkeeping and for the source lists

Associated product: sensitivity maps in several energy bands, or tool to provide minimum detectable flux as a function of spectral index and duration

Catalog pipeline. Schedule

1. Identify candidate source search algorithms. Done (next talks)
2. Define evaluation criteria. Started. *November 2004.* Not yet concluded.
3. Build pipeline prototype. *By end 2004*. Good progress.
4. Evaluate candidate algorithms. *Beginning of 2005*. Done only on DC1.
5. First selection of source search algorithm. *Mid 2005 (before DC2).* Not done yet. Can wait until DC2.
6. Define processing database. *By end 2005*
7. Integrate pipeline elements (including flux history, identification). *2006*.
8. Ready: *end 2006*

Source localization

- ✓ Done locally (for each source in turn)
- ✓ Typical algorithm (like SExtractor) uses a smoothed map as input, and interpolates to find the maximum.
- ✓ Another possibility would be to use a different algorithm (like the multichromatic wavelet) to localize sources once they are detected.
- ✓ We need to provide a precision on source position

- ✓ Building TSmaps for all sources is certainly VERY CPU intensive.
- ✓ The precision depends mostly on the source spectrum (estimated by likelihood) and the source significance. This can be computed once and for all, and simulations can tell whether secondary parameters (like background shape) are important.

Pipeline prototype

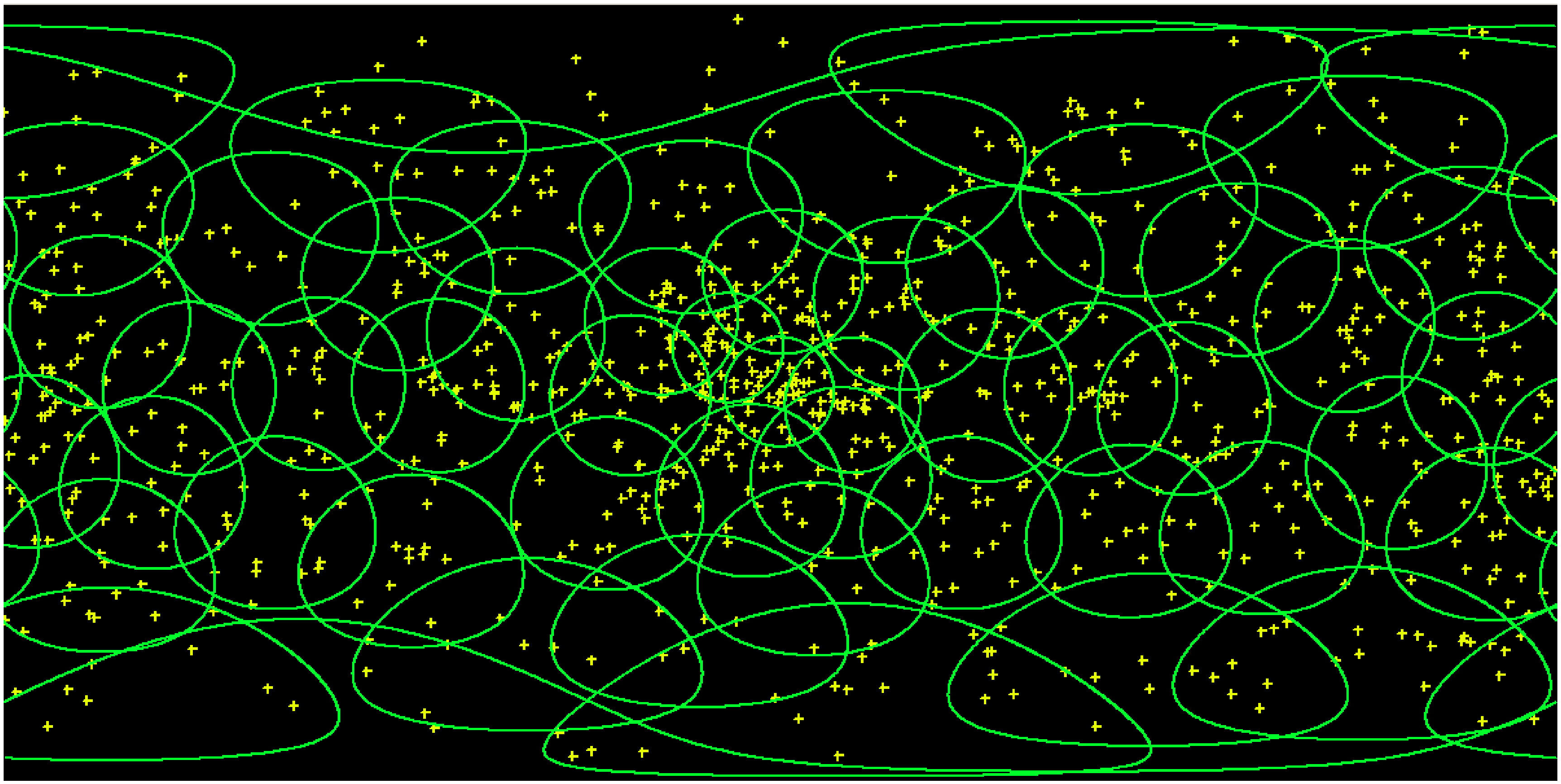
- ✓ We have started with the likelihood step (most time consuming)
- ✓ Basis provided by J. Chiang (catalogAnalysis package, sourceAnalysis Python script). Chains event selection, exposure map generation and likelihood run.
- ✓ Use the OPUS task scheduler
- ✓ Try using OPUS in a minimal way (do not decompose too much) in order to facilitate portability.

- ✓ Standard region of interest (20° radius) contains many sources (several tens)
- ✓ The optimization algorithm has trouble converging with so many parameters (2 per source + diffuse emission)
- ✓ Likelihood behaves very strangely with the DC1 PSF calibration, particularly close to 0 flux. Used the TEST PSF (simple representation provided by J. Chiang, forcing the parameters to vary smoothly with energy).
- ✓ The MINUIT optimizer sometimes ends in error. Used DRMNGB optimizer for the full runs.

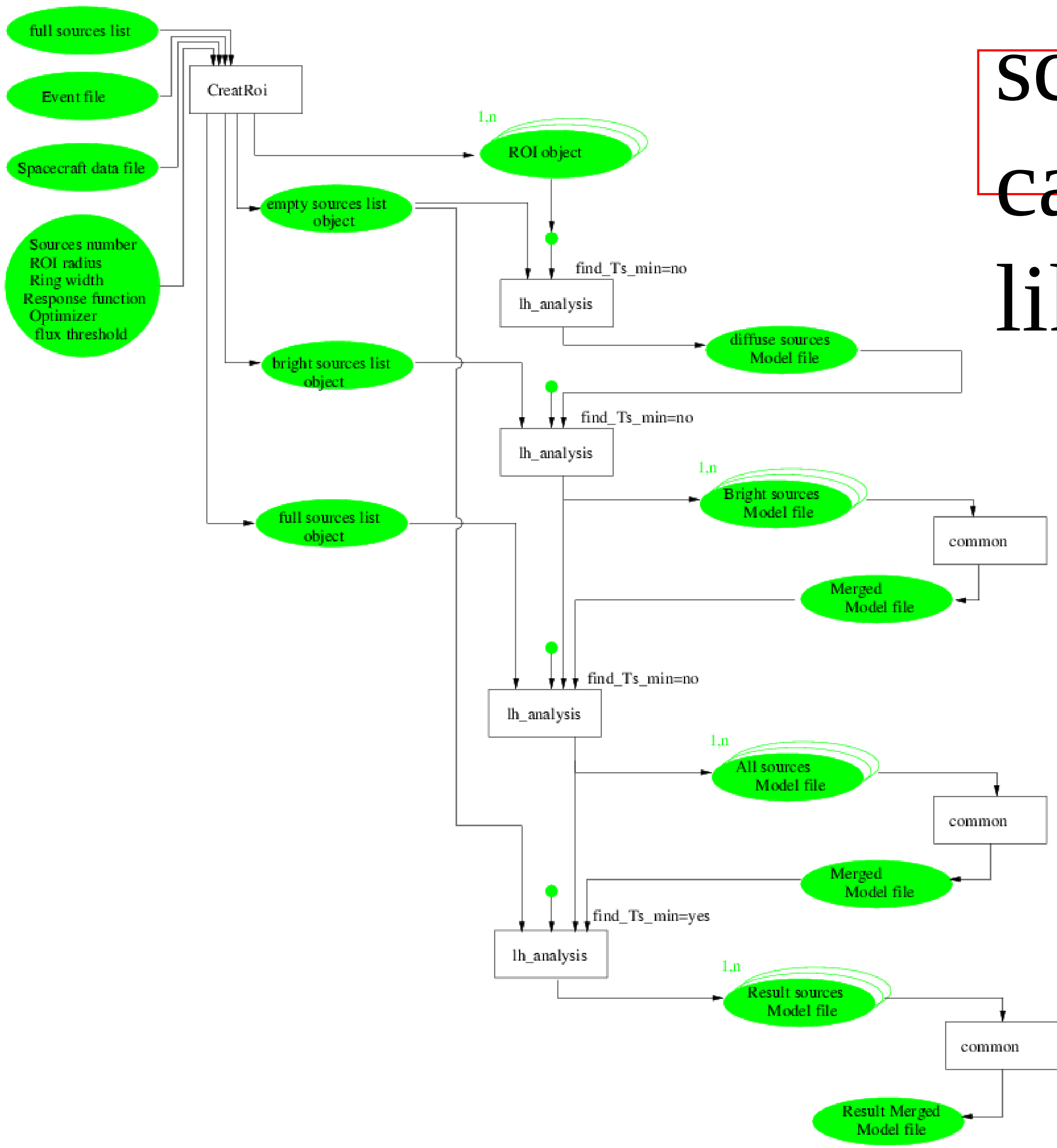
Regions of interest

Input: list of sources (from all-sky source search, here all DC1 sources for illustration)

- ✓ Radius < 20°
- ✓ Number of sources < 40 (actually used 20 or 10)
- ✓ Start with RoIs centered on bright sources
- ✓ Try moving around to encompass more sources
- ✓ End here in 53 RoIs



Global scheme for calling likelihood

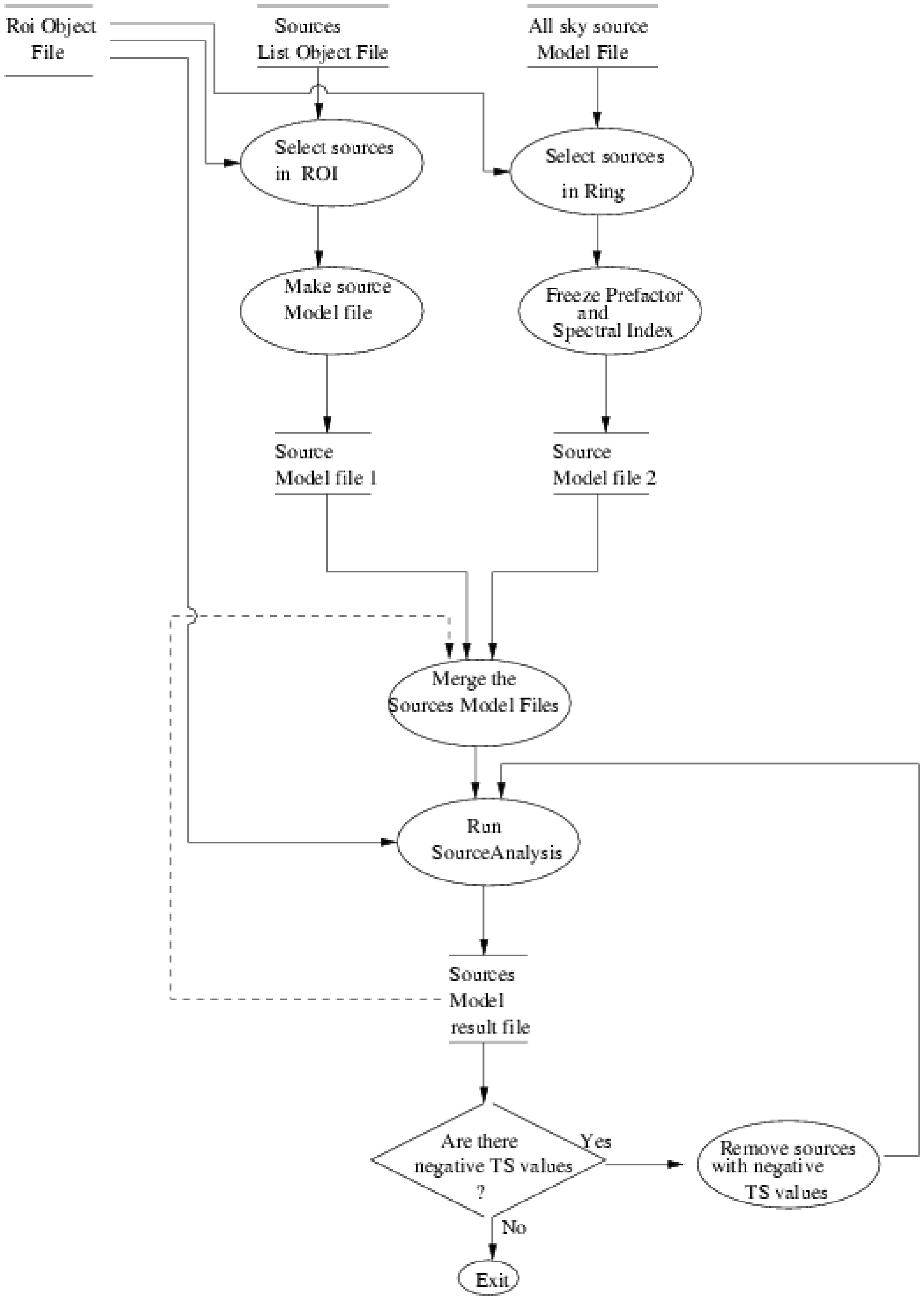


Idea: Run likelihood in several steps to facilitate convergence

Input: full list of sources, split into bright/faint

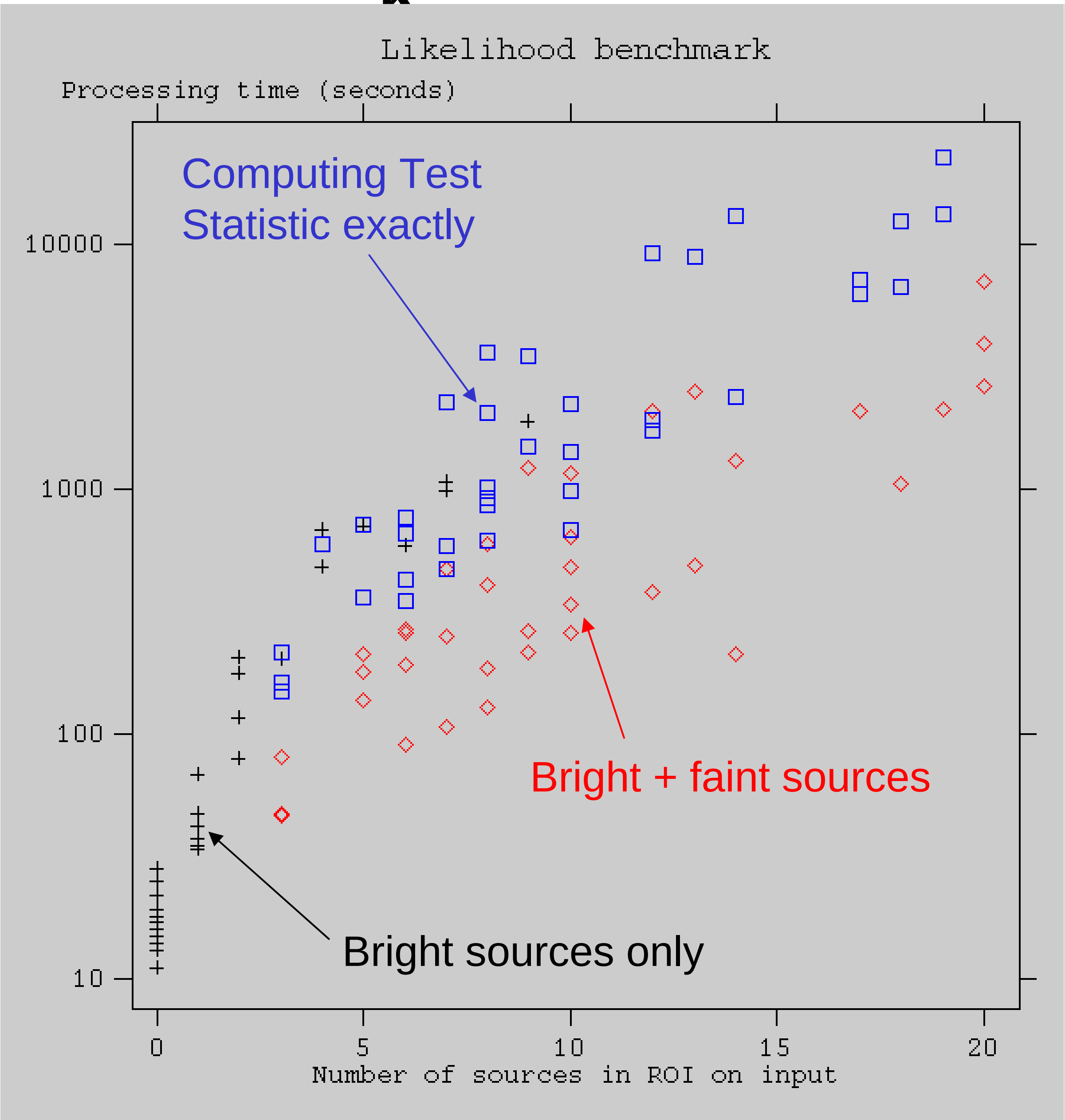
- ✓ At each step, use the same RoIs, events lists, exposure maps, but a different XML source file
- ✓ After each step, merge XML source files into a global (all-sky) one, keeping parameters from RoI to which source is closest (if several)

Elementary brick for OPUS



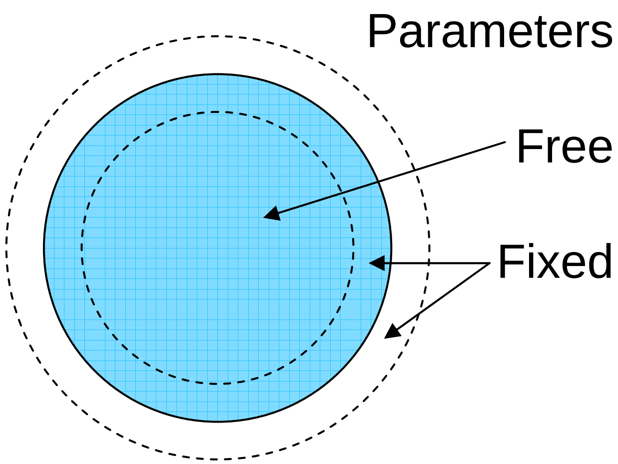
- ✓ Start with XML file from previous run (if any)
- ✓ Add sources within 10° of RoI border (from global XML file), leaving their parameters fixed
- ✓ Add other sources within the RoI from input source list, setting initial flux to 0
- ✓ Remove sources with negative TS values in output (could cut a little higher than 0)

Benchmark



- ✓ Tested on DC1, 6 days with 3EG source list
- ✓ Computing time increases very fast (nearly exponentially) with number of sources in RoI (here limited to 20)

Possible improvement: use smaller radius for free sources than for events (so that each free source is entirely contained within RoI



Tools required

Either used in pipeline itself, or as a help to adjust parameters

- ✓ Interface to the diffuse emission model (cube), allowing to get an image in a given energy band.
- ✓ Script to generate a light curve for known sources.
- ✓ Sensitivity provider ?

Source catalog generation

Catalog generation is on the way !

Several open points:

1. Should an adjustement of the diffuse emission model be foreseen inside the catalog production process, in the new scheme (cube) ?
2. Do we need a separate step for source localization and position error ?
3. Should we implement additional cuts on the data (e.g. on off-axis angle) ?