A Simple and efficient algorithm to search the Gaia Star Catalogue

Klaas Vantournhout [1], Pedro Santana Camprubi [1] and Mathias Lauer [2]
[1] *CGI Deutschland Ltd. & Co. KG at ESOC/ESA, Darmstadt Germany*
[2] *ESOC/ESA, Darmstadt, Germany*

Abstract

Background: ESA's Euclid mission will make use of Fine Guidance Sensors to aim its telescope at distant galaxies. To ensure enough star coverage for a precise attitude determination, the FGS needs to track higher magnitude stars than conventional star trackers. To this end, Euclid will use the Gaia star catalogue as a reference star catalogue.
Purpose: Due to the vast amount of stars available in the Gaia Star Catalogue, it is inefficient to load the full catalogue into the FGS. Instead, the Gaia Star Catalogue will be processed on-ground providing a small subset of the best candidate guide stars per observation.
Method: This paper describes a simple and efficient algorithm to select stars from the Gaia Catalogue by creating a simple search index based on the *Smallest Enclosed Disk algorithm*. It assumes that the full catalogue, ordered by implemented HEALPix, is split over several equally sized files. By processing the index file, it is quickly determined which files need to be processed to obtain the requested query.
Conclusions: The described method allows to select stars from the Gaia Catalogue efficiently while yielding a nearly constant processing time. Furthermore, it addresses issues related to epoch propagation and the observer's position directly. Moreover, as the implementation is straightforward, it is less error-prone as data remains unchanged, allowing for backwards validation.

Keywords: Gaia Star Catalogue, Fine Guidance Sensor, Precise pointing-determination, Euclid

Introduction

Astronomy space-missions require precise pointing-determination in order to accommodate highly sensitive payload instruments. Commonly, such precision is obtained by combining the payload instrument with a Fine Guidance Sensor (FGS), both sharing their optics. This way, the FGS benefits from the payload's high-quality optics at the sacrifice of a reduced field of view (FOV). This, to use as much as possible of the telescope's capability for the instrument.

ESA's Euclid mission will aim its Korsch-type telescope at distant galaxies. The telescope's $0.79° \times 1.16°$ FOV will be guided by an FGS consisting of four $0.11° \times 0.11°$ FOV CCD detectors. To ensure enough star coverage for a precise attitude determination, the FGS, with its small FOV, needs to track higher-magnitude stars (in the range 10–19) than conventional star-trackers (in the range 2–5). Euclid will use the Gaia catalogue as reference star catalogue. This ensures both star coverage and accurate data. However, because of the catalogue's size, it is inefficient to load the full catalogue into the FGS. Instead, the Gaia catalogue is processed on-ground to provide a small subset of the best candidate guide-stars per observation.

This paper describes a simple and efficient algorithm to select stars from the Gaia catalogue, falling inside the telescope's FOV. The paper discusses the optimization of the algorithm and its implementation using Gaia Data Release 1 (GDR1) [1, 2]. GDR1 contains 1 142 679 769 sources, sorted in a nested HEALPix distribution [3] divided into 5 232 equally sized files (218 453 records each). The HEALPix distribution ensures star-grouping as "neighbouring" stars have their records close. Nonetheless, hops occur whenever there is a change of pixel. In

basic terms, each file contains various "patches" of the sky. The algorithm adapts the Smallest Enclosed Disk algorithm [5] to find the smallest cone containing the stars per "patch" of the sky. This allows the content of each file to be indexed in terms of cones (axis direction and half cone angle). The telescope's FOV can also be parameterised as a cone. It is then simple to quickly determine the files with stars that fall in the telescope's FOV by finding those cones intersecting the FOV's cone. The user then only scans the files intersecting cones to find the stars that fall inside the FOV. This efficiently reduces the amount of data to be processed and optimizes the runtime performance

Searching the Gaia Star Catalogue: the problem

The Gaia Star Catalogue consists of astrometry and photometry for over 1 billion sources brighter than magnitude 20.7 in the white-light photometric band *G* of Gaia. For space missions such as Euclid, the catalogue has to be queried often to select a subset of the catalogue that can be uploaded to the spacecraft. In its most easy form, such a query reads:

Which stars can be found in inertial direction $\vec{d}$ within an angle θ?

A simple and direct way to answer this query is to process all billion sources and check if they satisfy the requested condition, i.e. $\vec{d} \cdot \vec{s_i} > \cos\theta$ with $\vec{s_i}$ the inertial direction of source i. While this might be easy and quickly implemented, this solution becomes unmanageable when time constraints are an issue, especially when multiple queries have to be performed and might have to be repeated due to planning issues in the commanding cycle of the spacecraft.

Searching through such a vast amount of records in an efficient way requires a structured ordering of these records. The Gaia Data Processing and Analysis Consortium (DPAC) decided to arrange the detections by a spatial index known as HEALPix [2]. The surface of a sphere is partitioned in 12 equally sized areas, referred to as the 12 base pixels. The resolution of pixels is increased by splitting each pixel into 4 new ones and so on. This way the resolution increase by three steps from base level 0 into 12, 48, 192 and 768 pixels.

When retrieving Gaia Data Release 1 (GDR1) of the Gaia Star Catalogue for offline analyses, it is obtained as 5 322 equally sized files with 218 453 records each. The files are ordered, such that, when concatenating them, one first finds the records belonging to base pixel 0, followed by those of base pixel 1 and so on up to base pixel 12. The ordering per base pixel is done per resolution level. That is to say, first one finds the subpixel 0 stars, followed by subpixel 1, 2 and 3 with each subpixel ordered accordingly. Figure 1 shows the ordering up to resolution level 2.

This spatial distribution ensures that close stars are grouped together and allows to quickly detect which arranged groups should be processed. However, it presents some disadvantages [4]:

- Complicated treatment of sources close to the region boundaries of the HEALPix.
- Handling of detection of high proper motion stars which cannot be easily bounded to any fixed region
- Repeated data-queries over spatially distributed FOV.

The software provided by the Gaia consortium, allows a quick search, given a direction and FOV which pixels contain stars of interest. It feels therefore natural to split the catalogue into files where each file contains all stars of a given pixel. The approach to query the catalogue is then straightforward:

1. Given an inertial direction $\vec{d}$ and half-cone angle θ, determine which pixels are of interest
2. Open only the files corresponding to those particular pixels and scan all stars in those files to check if they match your query.

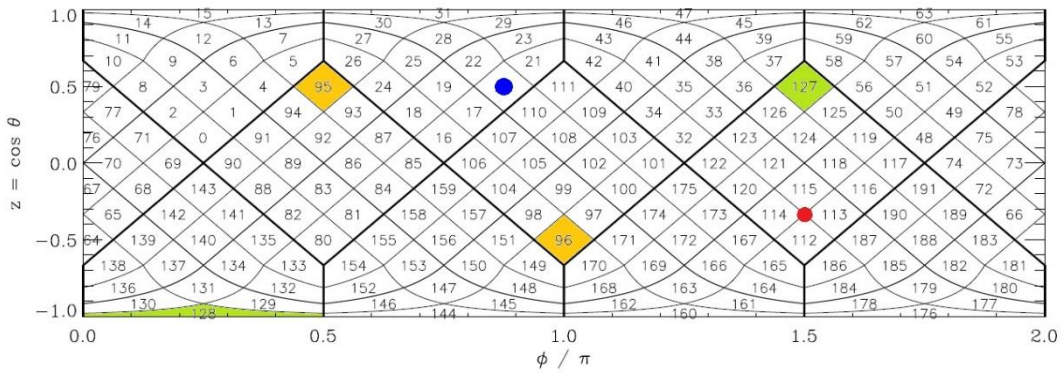
A downside to this approach is that the stars are not uniformly distributed over the sky. While, at a given resolution level, the pixels are equally sized, they do not contain the same amount of stars. Table 1 shows that some pixels contain a large number of records, while others barely any. The drawback of this is that your processing time largely depends on the area you process, especially when you take into account that you might have to process more than a single file. An extreme example of this is the area marked red in Figure 1. For any field in the direction of $[\text{ra,dec}] = [270°, -19.47°]$, you have to process 154 577 618 records.

As this is highly unfavourable, we opted for a different approach. Since GDR1 is delivered as 5 322 equally sized files with 218 453 records each it would be useful if one can determine which of those files contain stars of interest.

Table 1: Some basic statistics of the HEALPix pixels per resolution level. The total amount of pixels is presented in combination with the star count of the pixels with most and least stars. An average star count per pixel is also given.

Resolution Level	Number of pixels	least populated pixel		most populated pixel		average star count
		pixel ID	star count	pixel ID	star count	
0	12	4	10 772 679	7	297 879 390	95 223 314
1	48	5	1 766 829	28	154 577 618	23 805 828
2	192	20	336 513	112	56 101 710	5 951 457
3	768	82	78 293	448	17 290 804	1 487 864
4	3 072	328	15 980	2 634	5 058 117	371 966
5	12 288	1 119	2 583	8 274	1 554 462	92 991

Figure 1: The layout of the HEALPix pixels on the sphere arrange hierarchically for the resolution level 2. The 12 base pixels are clearly visible in thick lines and its resolution is twice refined. Each pixel is numbered from 0 till 191. The Gaia Star Catalogue for offline retrieval is sorted according to this numbering leading to a spatial ordering. Note that large jumps can occur between consecutive stars when changing pixel. A jump of 116°23′1.35" occurs when switching from pixel 95 to 96 (yellow) and a jump of 131°48′27.15" when switching between 127 and 128. For a pixel-based query, the two coloured dots mark the smallest and biggest processing cases. For a direction and field of view full located in within pixel 20 (blue) one requires to process 336 513 records, while in the direction of $[\text{ra,dec}] = [270°, -19.47°]$ (red), one needs to process 154 577 618 records (pix. 112: 56 101 710 records, pix. 113: 39 772 791 records, pix. 114: 36 571 559 records and pix. 115: 22 131 558 records). More details can be found in Ref. [3]. Image adapted from Ref. [3].



Searching the Gaia Star Catalogue: a possible solution

As mentioned in the previous section, GDR1 can be obtained as a set of equally sized files, which, when concatenated in order, create a fully HEALPix sorted data set. The idea is to keep the original set of files and create an index which allows one to quickly determine which files to open. This index should only be generated a single time.

A simple version of such an index file would keep track of the pixels available per file. Unfortunately, this will not resolve the problems which exist for stars with high proper motion or large parallax as mentioned earlier (See Ref. [4]). To this end, we abandon the HEALPix concept and use a simple mathematical object, the cone. Defined by a direction $\vec{n}$ and half cone angle φ, a cone $\mathcal{C}$ can represent a group of stars. All these stars satisfy the condition $\vec{s} \cdot \vec{n} \geq \cos\varphi$ with $\vec{s}$ the star's inertial direction. By determining, per catalogue file f, a set of cones $\boldsymbol{\mathcal{C}_f} = \{\mathcal{C}_{f,k}\}$ of which we know that each star of f lays in at least one cone of $\boldsymbol{\mathcal{C}_f}$, a low-resolution version of f is created. Furthermore, by additionally adding a proper motion μ and parallax ϖ to each cone, we can solve the problems for stars with large proper motions and parallax and compensate for epoch propagation and the observer's barycentre position. The proper motion μ of cone $\mathcal{C}$ is nothing more than the largest proper motion of the stars which are used to create the respective cone. The same holds for the parallax ϖ.

Determining which files to query is now easily done by first scanning the index file. If you want to select every star in a field of view with inertial direction $\vec{d}$ and half-cone angle θ, then select the catalogue files f for which there exists a cone $\mathcal{C}_{f,k}$ with:

$$\vec{d} \cdot \vec{n}_k \geq \cos\left(\theta + \varphi_k + \mu_k(t - t_0) + \varpi_k\left|\vec{b}\right|\right).$$

Here, $\vec{b}$ is the position of the observer with respect to the barycentre at epoch t_0. An algorithm to search the Gaia Star Catalogue could now look like:

Algorithm QUERYCATALOGUEWITHINDEX $\left(\vec{d}, \theta, t, \vec{b}, \mathit{IndexFile}\right)$

Input: An inertial direction $\vec{d}$ and angle θ for an observer in barycentric position $\vec{b}$ at time t. An index file of the Gaia Star Catalogue

Output: The stars matching the search request.

1. Initialize f to an empty string
2. **for all** $\mathcal{C}$ **in** *IndexFile* **do**
3. **if** f **equals** GETCONEFILE$(\mathcal{C})$ **then** process the next cone
4. $\vec{n}, \varphi, \mu, \varpi \leftarrow$ GETCONEPROPERTIES$(\mathcal{C})$
5. **if** $\vec{d} \cdot \vec{n}_k \geq \cos\left(\theta + \varphi + \mu(t - t_0) + \varpi\left|\vec{b}\right|\right)$ **then**
6. $f \leftarrow$ GETFILECORRESPONDINGTOCONE$(\mathcal{C})$
7. QUERYCATALOGUEFILE(f)
8. **end if**
9. **done**

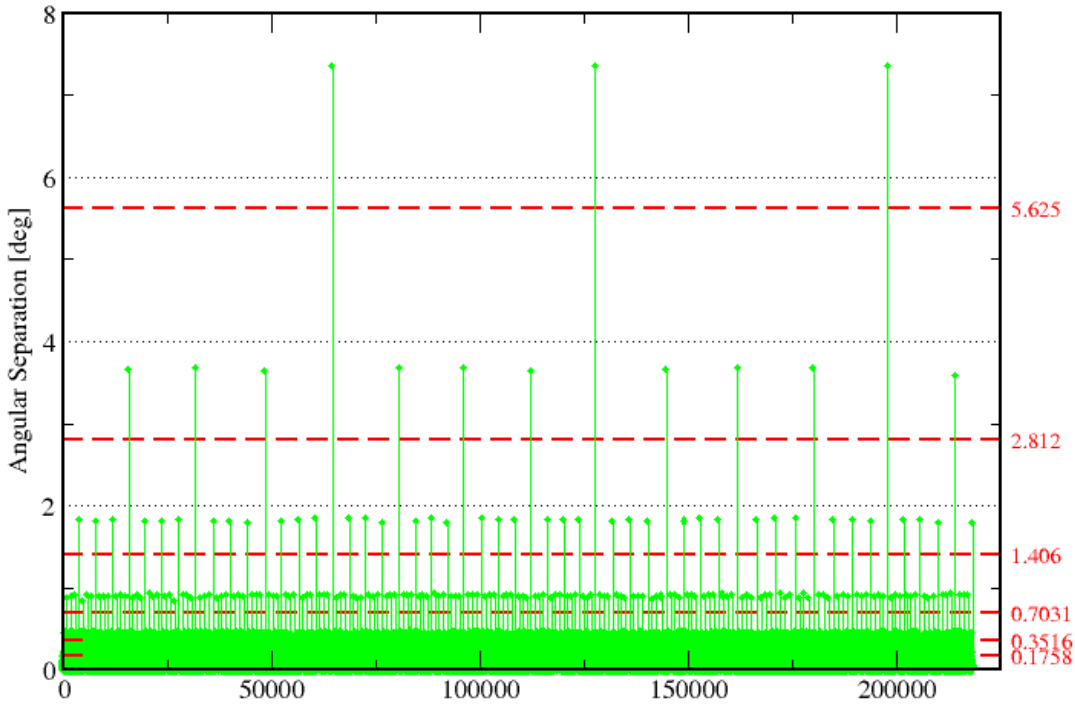
In this algorithm, we assume that QUERYCATALOGUEFILE(f) is a personalized query command.

Building the index file for the Gaia Star Catalogue

The biggest question remaining is how to build the index file. As explained earlier, the Gaia Star Catalogue is spatially ordered and consist of multiple files with the same amount of records per file. Furthermore, consecutive files can contains stars from the same HEALPix. As an example, pixel 1 of resolution level 2 is distributed over the first 3 files: `GaiaSource_000-000-000.dat`, `GaiaSource_000-000-001.dat`, and `GaiaSource_000-000-002.dat`. But a single file can also contain data from different pixels. This implies that consecutive stars do not necessarily need to lay spatially close to one another. While most stars will have a small angular separation as they are members of the same pixel, large separations can occur when the pixel changes. Examples can be found in `GaiaSource_000-011-057.dat` having a separation angle of 131°48′27.15" when switching from level 2 pixels 127 to 128 (See Figure 1).

With this in mind, it is evident that a single cone per catalogue file is inefficient as this would lead to some cones with a very large cone angle. When querying the index file with this approach, a lot of false positive matches would occur. Also, creating too many small cones is not efficient. This would, in its extreme case of one-cone per star, be equivalent to processing the full catalogue. Finding the optimum is not straightforward. However, if one has a group of stars which are close together, then the cone matching these stars should be the one with the smallest cone-angle. This minimizes the possibility of having false positives in the index query. Determining the smallest cone encompassing a set of directions is easily done by adopting the *Smallest enclosed disk* algorithm presented in Ref. [5] and converting it compute cones.

Figure 2: The above figure shows the angular separation between consecutive stars of the catalogue file `GaiaSource_000-000-000`. The band-structure visible in the file is a clear consequence of the ordering of the stars in the catalogue file. The red lines represent the angles marking the "size" of the pixel for a given resolution level n ($90/2^n$). When the angular separation between two stars lays above such line, it is most likely that the two stars are in separate pixels.



Due to the natural spatial ordering of the Gaia Star Catalogue, it is easy to determine which stars need to be grouped together. We know that most consecutive stars are members of the same pixel. This implies that the angular separation is small. But when two consecutive stars are in different pixels, the angular separation is larger than nominal. This can be seen in Figure 2. The band structure visible in the angular separation of consecutive stars is correlated with the values $\{90/2^n : n = 0,1,2,\dots,12\}$. This is because 90° is the maximum separation in right ascension of the base pixels (See Figure 1). From Figure 2 we can now decide to group consecutive stars together when the angular separation is smaller than a value β. The algorithm to create the catalogue file can now be written as:

Algorithm CREATECATALOGUEINDEX $(\beta, \boldsymbol{\mathcal{G}})$
Input: The initial list of Gaia Star Catalogue files $\boldsymbol{\mathcal{G}} = \{f_1, \dots, f_n\}$ and a separation angle β.
Output: The index of the catalogue $\boldsymbol{\mathcal{G}}$.

1. **for all** f **in** $\boldsymbol{\mathcal{G}}$ **do**
2. Split the list of stars in f into sub-lists $\ell_1, \ell_2, \dots, \ell_m$ such that two adjacent stars of list $\mathcal{L}$ are part of the same sublist ℓ_k if their angular separation is smaller β. Each of these sublists will create an entry in the index file.
3. **for** $k \leftarrow 1$ **to** m **do**
4. Let $\mathcal{C}_k$ be the cone with the smallest cone angle encompassing the directions of all stars in ℓ_k.
5. Set μ_k to be the maximum proper motion of the stars in ℓ_k
   $\mu_k \leftarrow \max\left(\sqrt{\mu_{ra,s}^2 + \mu_{dec,s}^2} : \forall s \in \ell_k\right)$
6. Set ϖ_k to be the maximum parallax of the stars in ℓ_k
   $\varpi\_k \leftarrow \max(\varpi_s : \forall s \in \ell_k)$
7. **done**
8. **done**

A natural choice for the separation angle β is any of the values $\{90/2^n : n = 0,1,2,\dots,12\}$. Furthermore, since the stars creating a cone are consecutive in the catalogue file f, it is possible to introduce two more values, `RN_START` and `RN_END`, in the index file. These two values indicate which records of catalogue file f to query when this cone matches. This creates an extra resolution in the system and could be used to speed up the routine QUERYCATALOGUEFILE(f) in the algorithm QUERYCATALOGUEWITHINDEX. The record markers `RN_START` and `RN_END` mimic an extra splitting of the catalogue files f. As if each file is split into multiple files representing the stars per cone.

The size of an index file is given in Table 2 as a function of β where each record contains the following information:

FILENAME	f	base name of the GaiaSource file
RA	α (deg)	barycentric right ascension of the direction of the cone
DEC	δ (deg)	barycentric declination of the direction of the cone
DIR_{X,Y,Z}	$\vec{n}$	barycentric direction of the cone. Unit vector
HALFCONE	φ (rad)	half-cone angle of the cone
MAX_PM	μ (mas/year)	maximal proper motion of the stars that are used to construct the cone
MAX_PARALLAX	ϖ (mas)	maximal parallax of the stars that are used to construct the cone
RN_START		record number of the first star used to construct the cone
RN_END		record number of the last star used to construct the cone

Results and Conclusion

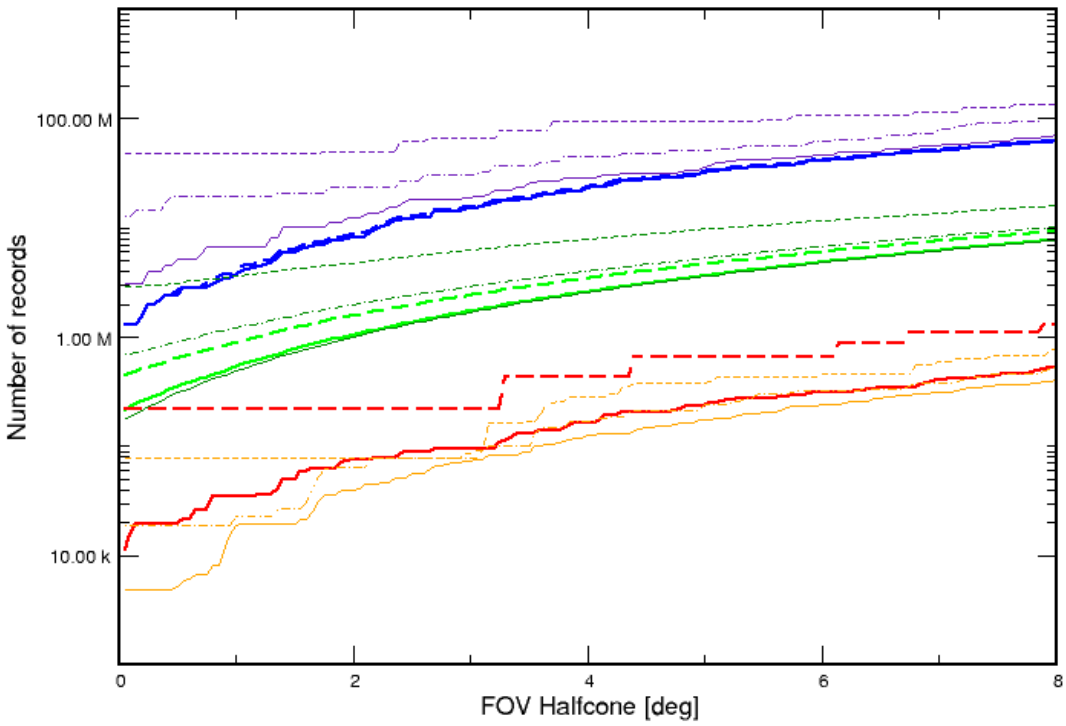
To demonstrate the effectiveness of this method, we looked at some basic statistics. Table 2 shows the number of cones per index file for various separation angles β. It is clear that the overhead of processing an index file is negligible as the total amount of cones per index file is small in comparison to the total amount of stars per catalogue file.

Table 2: Some basic statistics of a small set of index files.

Separation Angle β	Number of Cones	Maximum Number of Cones per File	Average Number of Cones per file
$90°/8 = 11.25°$	5 614	3	1
$90°/16 = 5.63°$	6 766	6	1
$90°/32 = 2.81°$	11 375	22	2

An analysis was made to compare the computational effort of a HEALPix-based search versus an index-based search. A full scan of the sky was made over 768 uniformly distributed directions for a full range of fields of view with half-cone angles ranging from 0.05° up to 8°. Figure 3 shows the minimum, maximum and average amount of records to be processed per half cone angle. When comparing the results of the index-based searches (separation angle $\beta = 90°/2^5$) with and without the usage of the record markers `RN_START` and `RN_END` (thick dashed versus full), the results show that, in average, using the record markers reduces the computational effort by half for small fields of view. For larger fields of view, however, the computational effort becomes equivalent. For a field of view of 8° half cone, the effect is reduced to 0.17% (1 500 000 records less to process with respect to the 9 200 000). The computational performance can be further improved when reducing the size of the separation

Figure 3: *The minimum (red/orange), maximum (blue/indigo) and average (green/dark green) amount of records processed for 768 uniformly distributed directions and a range of half cone angles as fields of view. The thick lines show the results when using an index file with $\beta = 90/2^5$ with (full) and without (dashed) usage of `RN_START` and `RN_END`. The thin lines show the results for a HEALPix based search of resolution level 3 (dashed), 4 (dash-dot) and 5 (full)*



angle β. This, however, will only have an effect on the index-based search using the record markers. In the other case, the effect will be minimal as the file-size cannot be reduced in contrast to the artificial reduction when using the record markers.

When comparing the index-based search with respect to the HEALPix-based search, it becomes evident that the index-based search is more optimal up to a resolution level of 5. At this level, both methods are in average equivalent, however, the worst case scenarios are still better to be processed using the search index. Moreover, when comparing the search with resolution level 5 with respect to the cone-based search, the variability of the HEALPix-based search is much higher. This alludes to constant-time processing for a single field of view when using the index-based search. It is understandable that both methods will slowly converge to one another as increasing the resolution size HEALPix-based search becomes equivalent to increasing the resolution by decreasing the separation angle $\beta = 90°/2^n$. This is also why, for the presented results, the HEALPix-based search with resolution level 5 is equivalent to the index-based search with separation angle $\beta = 90°/2^5$. The higher the resolution level, the more cones/files one starts to have and at a given resolution level, you lose all computational benefits.

While the index-based search becomes equivalent HEALPix based search for higher resolution levels, it has various advantages:

- The algorithmic implementation is straightforward and less error-prone
- It allows epoch propagation for the proper motion as well as compensation of the observer's barycentre position.
- The catalogue downloaded for offline retrieval can be used as is. No reordering or redistribution of records is needed. This reduces the possibility of corrupt data and makes backwards validation possible.
- There is no need for third-party software.

In an operational environment, the method has shown to be reliable and efficient enough to ensure that the operational constraints are fulfilled. Nonetheless, further investigation might help to improve the index file. That is to say, can we find an algorithm that reduces the number of cones and minimizes the ratio of solid angle covered by the stars in the catalogue file with respect to the solid angle covered by the entries in the index file?

Acknowledgement

This work has made use of data from the European Space Agency (ESA) mission Gaia (https://www.cosmos.esa.int/gaia), processed by the Gaia Data Processing and Analysis Consortium (DPAC, https://www.cosmos.esa.int/web/gaia/dpac/consortium). Funding for the DPAC has been provided by national institutions, in particular the institutions participating in the Gaia Multilateral Agreement.

References

1. Gaia Collaboration, A. G. A. Brown, A. Vallenari, T. Prusti, J. H. J. de Bruijne, F. Mignard, R. Drimmel, C. Babusiaux, C. A. L. Bailer-Jones, U. Bastian and et al. *Gaia Data Release 1. Summary of the astrometric, photometric, and survey properties*. A&A **595** (2016)
2. Gaia Collaboration, T. Prusti, J. H. J. de Bruijne, A. G. A. Brown, A. Vallenari, C. Babusiaux, C. A. L. Bailer-Jones, U. Bastian, M. Biermann, D. W. Evans and et al. *The Gaia mission*. A&A **595** (2016)
3. K.M. Gorski, E. Hivon, A.J. Banday, B.D. Wandelt, F.K. Hansen, M. Reinecke and M. Bartelman, *HEALpix – a Framework for High Resolution Discretization, and Fast Analysis of Data Distributed on the Sphere,* Astrophys. J. **622** (2005)
4. Gaia Collaboration *Gaia Data Release 1: Documentation*
5. Mark de Berg, et al, *Computational Geometry Algorithms and Applications*, (Springer Verlag, 2008)