# Size-of-Computing Estimates for ngVLA Synthesis Imaging ngVLA Computing Memo #4

S. Bhatnagar, R. Hiriart, M. Pokorny

August 2021

**Abstract**

In this memo we analyze the size-of-computing (SofC) for imaging of representative use-cases for the ngVLA. The primary goals of this exercise are to develop theoretical scaling models for the compute load, establish procedures for measuring the performance of the existing implementation of the required algorithms, and verify the theoretical scaling laws. Based on the scientific requirements for the various science cases, we derive the required imaging algorithms and their parameters, and develop parameterized models for the computing and I/O load. These models are verified against the implementation in the production CASA package. While SofC estimates are also made, since full scale tests were not possible in a reasonable time (due to resource limitations), these estimates are by extrapolation and necessarily approximate.

## 1 Introduction

The ngVLA telescope is being designed to improve the collecting area and angular resolution of the existing VLA by orders of magnitude. To achieve this, the ngVLA will have 244 18m-diameter antennas, with the longest baseline of order $10^4$ km. The antennas are arranged in four distinct arrays: the core array, with 94 antennas and 1.3 km maximum baseline; the plains array, with 74 antennas and 36.5 km maximum baseline; a mid-range array, with 46 antennas and $10^3$ km maximum baseline; and a long baseline array, with the rest of the antennas and $10^4$ km maximum baseline. More antennas and longer baseline length increase the raw data rates from the telescope required to achieve the various Key Science Goals (KSG). The size of computing (SofC) for imaging with the ngVLA is therefore significantly higher compared to existing telescopes in terms of the necessary raw computing, I/O load and computer resources.

Estimating the SofC associated with the ngVLA science drivers is necessary to ensure a viable end-to-end telescope design. The primary purpose of this memo is therefore to establish a procedure for estimating the SofC for the ngVLA based on measuring the *single-core* computational efficiency of the necessary algorithms. For this, we establish the computational efficiency of the particular software implementation of the imaging algorithms used here for running on a single core, and measure its scaling with the data volume. These measures are then used to make an order-of-magnitude estimate of the ngVLA SofC.

It is undoubtedly necessary for ngVLA image processing to use parallel processing for performance. Although CASA imaging tasks support the use of multi-threading and/or multi-processing,

the ability to scale to large numbers of CPUs and/or GPUs is strongly implementation dependent. As a follow-up to this memo, we intend to derive scaling curves for the current CASA parallel processing implementation. The scaling laws derived in this memo provide a baseline against which to compare the efficiency of any parallel code implementation.

The SofC estimates require a (theoretical) model for the cost of computing of various algorithms, and a way to verify the model. Ideally this verification should be done via measurements of I/O and computing loads using simulated data to scale. However, full simulation of the relevant ngVLA use-cases is currently out of scope due to resource limitations (in terms of available time and required computing resources). We therefore verify the models via imaging of affordable simulations and make reasonable extrapolations of the parameterized scaling laws to arrive at the SofC estimates. These estimates are therefore necessarily approximate and should be used as only order-of-magnitude estimates. The estimates are based on the particular implementation of the algorithms in the CASA package. For a reliable extrapolation of the *measured* scaling to the full-scale estimate of the SofC, it is necessary to establish that the computational efficiency of the implementation is reasonable. We therefore also estimate the computing efficiency of this implementation as a ratio of the theoretical estimate of the required FLOPs to the measured FLOPs including overheads (Sec. 5). Comparison of the measured code efficiency with the practically expected single-core efficiency allows us to normalize the effects of the particular implementation used to measure the scaling.

## 1.1 Assumptions and Scope

1. It is assumed that the computing load is dominated heavily by the gridding/degridding operations. The cost of the minor cycle depends on a number of hard to quantify parameters, like the complexity of the brightness distribution and the required imaging dynamic range. The costs also typically scales as the number of pixels with significant emission (not as the total number of pixels) and the complexity of emission in these pixels . It is however possible that in some cases, like wide area surveys made from many snapshots, processing time for minor cycle could end up dominating the computational costs. Subsequent memos will examine more closely the validity of this assumption, in relation with the required science cases for the ngVLA.

2. An important aspect of the cost of computing is the amount of computer memory required by the algorithms. This can impose limits on the level of parallelization that it is possible to achieve with specific implementations, as only a limited number of computing cores (either in CPUs or GPUs) can be used in a machine before exhausting the available memory. Memory usage and bandwidth are recognized as critical aspects for ngVLA, but the current memo only performs some preliminary measurements on memory usage. The subject will be dealt with in a subsequent memo.

3. Pulsar modes will be characterized in a subsequent memo. This memo only covers synthesis imaging science cases.

4. The estimation of required computing resources is based on the notional Reference Observing Program[6], which defines a collection of observations based on the project's Key Science Goal (KSG) requirements. The computing load derived from these science cases may not be completely representative of the average load once the telescope is in steady operations and most of the observations are PI-driven. Subsequent documents will refine the estimations, including the Envelope Observing Program[7], which is intended to be an upper envelope on the resources demanded by the facility in a typical year of PI-driven observations.

5. This memo is intended to define a framework to analyze the computing performance requirements, provide initial measurements and estimations, and identify areas that require additional research and which will be treated in subsequent memos. It is recognized that ngVLA will require massive parallelization, but this subject is only touched cursorily in this document, by means of defining the speedup ratio and the parallelization efficiency parameters. Currently there are no implementations that would allow us to measure these parameters at the scale needed, so the

estimations provided in this document should be considered only as order-of-magnitude guesses. An important goal for the next activities in this regard is to prototype parallel implementations of the required algorithms and measure these parameters.

6. This memo only considers the CLEAN algorithm, along with A and W projection algorithms. Other algorithms, like w-stacking and baseline dependent averaging will be treated in subsequent memos.

7. Operational concerns such as whether the processing will be performed in one super-computing center or distributed in multiple collaborating facilities, in the cloud, or in a hybrid system, are not treated in this memo either. This subject—although important—depends on possible collaborations with other organizations, decisions about the location of the ngVLA facilities, economic and technological factors, etc. At this point it is still too early to advance the design in this regard.

8. The intent of this memo is to perform a computing performance analysis of the synthesis imaging algorithms. A cost derivation is not included. This should be part of project's cost model.

## 1.2 The dominant driver for the SofC

For the purpose of SofC estimates, the end-to-end processing of the data can be divided into two distinct steps: (1) data calibration and flagging, and (2) imaging. The cost of data calibration is relatively small, and the overall cost of end-to-end scientific computing is dominated by imaging. We therefore focus on the SofC for imaging only.

The quality of a raw image is typically limited by instrumental artifacts, and deriving an artifact-free model of the sky fundamentally requires iterative algorithms. This process itself can be divided into two broad steps:

1. **Derivative computation**: The required number of computations is the same as computing a residual image, which requires the evaluation of forward and reverse transforms (*i.e.*, gridding and de-gridding[1]) and two FFTs. This combined process is often referred to as the "major cycle". As these algorithms operate on visibility data, their computational complexity necessarily scales with total data volume, but also depends on a specific imaging algorithm and its parameter values.

2. **Model update**: This step (a.k.a. "deconvolution") involves making a wide-band model of the sky emission given the raw residual image and the telescope point spread function (PSF). This step itself is also iterative in nature, and is often referred to as the "minor cycle". The algorithms for this step are image-domain algorithms. The computational complexity of this step scales with the size of the image, and is independent of the raw data volume.

A typical end-to-end imaging requires a number of evaluations of the derivative. For each evaluation of the derivative, iterative sky-model update algorithms are triggered. The final cost of imaging is a function of the number of times the derivative is computed and the cost of the model update algorithms. The computational complexity of the forward and reverse transforms scales with the number of visibilities and the size of the Convolution Functions (CF) used (Sec. 3). The cost of model-update algorithms is relatively low and scales with the image size. For the ngVLA, the raw data volume is typically several orders of magnitude larger than the image size, and as the computational complexity of the model update algorithms is independent of the data volume, the total cost of imaging is dominated by the cost of evaluating the derivative. Therefore, in this memo we focus on the derivative computation step only. We develop scaling laws for the algorithms for forward and reverse transforms parameterized by the data volume and the parameters of the required algorithm (see Section 3). Computational overheads in practical implementations of these algorithms is included *via* coefficients in the

[1] "Gridding" refers to the operation of interpolating the visibilities, which occur at any point in the uv-space, into a regular grid. "De-gridding" refers to the inverse operation.

equations. In order to make SofC estimates, these coefficients are measured by running the imaging implementation in the CASA package using simulated Measurement Sets (MS) for a selected set of KSGs, which together cover the full range of the required algorithms.

It is illustrative to plot the number of floating point operations per second in an execution of the CASA imaging task, `tclean` to understand the relative importance of these steps. An example, using the proto-planetary model to simulate visibility data is shown in Figure 1. The proto-planetary model (`ppmodel_image_93GHz.fits.gz`) is one of the examples available in the ngVLA Simulation CASA Guide[1]. The specific model image used in this example is not important for visualizing the profile of the floating point operations versus time – similar profiles are observed with other model images. The high-value spikes correspond to the FFT and inverse FFT. In between these spikes, the model-update step is performed with computations done in single-precision. The length of this step is variable, depending on the stopping criteria for the model update algorithm setup. Typically, imaging starts with a stopping criteria that triggers relatively fewer iterations. As the model improves and the residuals become progressively smaller, it takes more and more model update iterations to converge. This is readily apparent in the example plot. In between the FFT and model-update operations, gridding and de-gridding are performed for each visibility read from the data base. Since the length of this step depends on the number of visibilities, they appear as regular intervals in the plot, with gridding being performed in double precision and de-gridding in single-precision. The first interval in the plot corresponds to the calculation of the PSF, and the last intervals of the plot correspond to saving the model column and restoring the image. This example was executed over a modest data set (0.5 GB, 5.4M rows). As the size of the data set grows to the scales expected for typical ngVLA observations, the task will spend most of the time on the gridding/de-gridding step, as the runtime of these steps is directly proportional to the number of visibilities in the dataset. The runtime of the steps that are performed in image-space, on the other hand, becomes much smaller in comparison for typical image sizes.

The overall computational complexity for imaging varies significantly with the choice of algorithms, and consequently the associated SofC varies from easily affordable to nearly unaffordable. However, not all algorithmic combinations may be necessary, and observing parameters can be adjusted for the particular scientific goals to optimize the data volume and computing load. The ngVLA project defines a list of 24 science cases derived from the KSGs, with associated sensitivity, angular resolution and imaging performance requirements, as shown in Table 1. For the purpose of estimating SofC for the ngVLA, we therefore first determine the observing parameters for each of the science cases, and then determine the required combination of the algorithms for the forward/reverse transform (standard, W-Projection, A-Projection, AW-Projection) and image modeling step (Multi-Scale, Multi-Term, Multi-Scale Multi-Term). The computational efficiency of these algorithms and their scaling with data volume was then measured and used to estimate the SofC for a representative set of science cases.

## 1.3 Comparison with the SKA

The SKA Project has made a similar assessment for the SKA SofC. Given the scale of computing and data volumes involved for the ngVLA, it is instructive to compare with the SKA requirements and understand the salient differences.

While both telescopes can be classified as large-N telescopes (having relatively large number of antennas in the arrays), an important difference that impacts the SofC estimates is the operating frequency range. Broadly, the SKA is a low-frequency telescope operating in the 100s of MHz to several GHz range. The ngVLA, on the other hand, is a high-frequency telescope operating in the few GHz to about 100 GHz range. This frequency difference leads to some important differences in SofC estimates.

The antenna field of view (FoV) for the SKA is larger – primarily due to the lower operating frequency range. The FoV for the SKA-low (array of dipole elements) is even larger compared to SKA-mid (array of antennas). This FoV requires imaging of a larger part of the sky per correlator

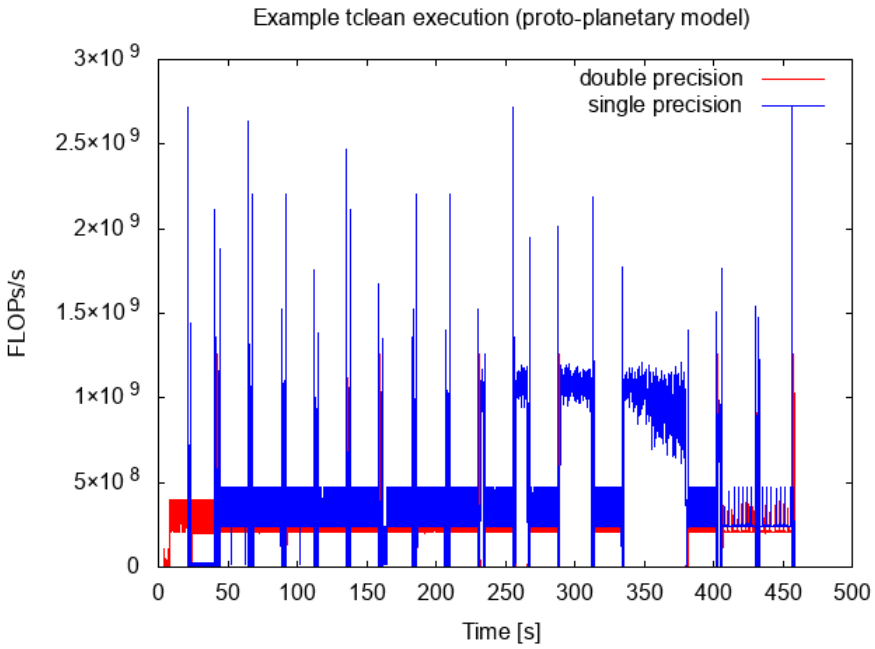


Figure 1: Floating point operations per second for an example `tclean` execution. The data was simulated as described in the ngVLA Simulation CASA Guide[1] and the FLOPs/s was measured using `perf`. The short peaks correspond to the forward and reverse FFT, and the regular intervals below 0.5 GFLOPs/s correspond to gridding/degridding. The operations reaching around 1 GFLOPs/s that start appearing after 250 seconds with progressively longer durations are deconvolution cycles. As the threshold used in the minor cycle to stop cleaning is decreased in each iteration, the algorithm cleans more and more.

phase center. The effects of both the w-term and the antenna PB become stronger with the distance from the phase center. The complexity of the A-Projection algorithms scales with the distance from the center, and the fractional bandwidth of observations. The complexity of the W-Projection algorithm scales strongly with the magnitude of the w coordinate, which in turn scales with the maximum baseline length, the distance from the center, and the imaging FoV. Furthermore, the radio emission from the sky is typically much stronger and more widely spread at low frequencies. High resolution imaging of the entire FoV at high imaging dynamic range becomes necessary to achieve many of the scientific goals of the SKA, which necessarily requires the use of the AW-Projection algorithm. The cost of imaging with the SKA is therefore significantly higher due to a combination of larger FoV, longer baseline, and need for high imaging dynamic range.

The imaging FoV requirement for the ngVLA is smaller for a significant fraction of the KSGs. The KSGs which require relatively larger FoV tend to have lower resolution, lower imaging dynamic range, and smaller expected observing fractions, allowing the conclusion that these KSGs weakly affect the overall SofC. Due to the physics of the emission mechanism, the radio emission from the sky in the ngVLA frequency range is also weaker and not as widely distributed in the FoV. As a result, many of the science goals with the ngVLA can be achieved with a narrower imaging FoV and lower dynamic range. Indeed, for many of the KSGs it is sufficient to image a small fraction of the antenna FoV. In combination with a lower fraction bandwidth, the effects of the w-term and the antenna PB is also significantly lower. Most of the KSGs therefore do not require the use of the W-Projection algorithm, which is more expensive compared to the A-Projection algorithm. The A-Projection algorithm is required for many of the KSGs; although, due to relatively smaller imaging FoV and lower fractional bandwidth, the cost of A-Projection for the ngVLA is also comparatively lower.

Another factor that contributes more strongly to a higher SofC for the SKA is the higher rate of change of the direction dependent (DD) gains for the SKA. The ionospheric plasma interacts with the incoming wave-front and adds significant distortions to it at radio frequencies in the SKA bands. For achieving the desired imaging performance, the effects of the resulting time-dependent DD complex gains must be corrected. SKA-low is also a dipole array with either electronic beam-forming and steering, or a very large FoV, both of which lead to time-varying forward gain. Correction for the effects of time-varying ionospheric gains and PB gains requires re-computing the convolution functions (CF) used by the forward and reverse transforms at a significantly faster rate than the ngVLA. This re-computation of the CFs contributes significantly to the already higher computational load for the SKA.

At higher frequencies, the incident radio wave-front interacts (very) weakly with the ionosphere. This decision is reflected in the column titled "Ionospheric correction" in Table 3. The ngVLA is an antenna array with a much better defined and mechanically steered antenna PB. With a narrower imaging FoV requirement for the ngVLA KSGs, the rate of change of the DD gains is much slower (and the magnitude of the resulting artifacts much smaller). The CFs therefore need not be re-computed often – for most cases, the CFs may even be pre-computed and cached. The additional computational load for the ngVLA is therefore thought to be insignificant. With the ngVLA, CF re-computation may be necessary to correct for the effects of distortions due to the troposphere, but only for those KSGs that require wide-field imaging at high frequencies, which will therefore not significantly affect the SofC.

SKA estimates a computing processing capacity[4] of 100 PFLOPs/s in order to deliver its science cases. However, this figure doesn't incorporate performance efficiencies. They estimate that 10-times more computing power may be needed if a general super-computer is used (measured by its Linpack performance), in opposition to building a custom-made super-computer which *may* deliver higher efficiency, see page 8). This is indeed much higher than our estimates for ngVLA, where (see Section 6) we estimate the need for a 50 PFLOPs/s system for all KSGs (or only 6 PFLOPs/s if the few more stringent KSGs are processed externally or de-scoped).

# 2 ngVLA Reference Observing Program

The ngVLA Reference Observing Program (ROP) quantifies the technical and observational needs of the driving use cases of the KSGs identified in the ngVLA Science Requirements [6]. The requirements for each of the use cases are shown in Table 1. The table shows both the driving and supporting use cases, although at this time only the driving use cases have been assigned a non-zero fraction of observation time. Both interferometric and pulsar search/timing use cases are included, although only the interferometric cases are considered for estimating the imaging computational load. The ROP cases, along with their observing fractions, are taken as representative of the types of observations that will be scheduled for observation in the first years of the ngVLA. A separate collection of science cases that should be closer to the typical use of the telescope (the Expected Observing Program, or EOP) is being prepared and will be used to update the SofC estimations when it becomes available.

Table 2 shows the subset of use cases considered in this memo, along with several derived parameters. Bandwidth and time averaging smearing constrain the maximum channel width and dump duration that can be used to observe without unacceptable loss of sensitivity. These constraints are taken into account to calculate the "effective" channel widths and dump duration [2]. To avoid a prohibitively large number of channels, the angular resolution used in these calculations is the required resolution from Table 1, not the resolution obtained by using the maximum baseline in the sub-array ($\lambda/b_{\rm max}$). It is assumed that the synthesized beam can be "sculpted" to the required resolution [5]. It is readily apparent that KSG2-type cases (UC 3-5) are significantly more expensive than the rest, given their high Vis/Hour rate. These cases require a very large number of spectral channels, given the need to perform blind searches for molecular gas with high spectral resolution.

## 2.1 Algorithmic requirements

As mentioned earlier, the SofC for ngVLA is dominated by the derivative computation step (major cycle), which is itself dominated by the gridding and de-gridding operations. The ngVLA KSGs collectively require one the following gridding/de-gridding algorithms:

1. **Standard gridding**: This is the traditional gridding used when no correction of direction dependent (DD) terms in the measurement equation is necessary. This is typically used for narrow field or shallow imaging (or both), where the effects of the antenna PB and non co-planarity of the array are not significant (compared to the target image-plane noise).

2. **W-Projection**: This algorithm corrects for the DD effects of the non co-planarity of the array during the imaging process. Non co-planarity of the array is measured by $N_w$ given by

$$N_w = \frac{B_{max}\left[FoV\right]^2}{\lambda} \tag{1}$$

   where $B_{max}$ is the longest baseline used, $\lambda$ is the wavelength of the observing setup (both in meters), with the $FoV$ in radians.

   When $N_w > 1$, the W-Projection algorithm is required for imaging. This typically happens when imaging relatively larger fraction of the antenna FoV at high resolution (long baselines).

3. **A-Projection**: This algorithm corrects for the DD effects of the antenna PB in time, frequency and polarization. This is typically required for imaging large FoV (greater than $0.5\lambda/D$ where $D$ is the antenna diameter), including mosaic imaging, at resolution such that $N_w \leq 1$. For large FoV imaging where $N_w$ is also greater than one, AW-Projection is required.

The algorithms required for the various KSGs, determined on the basis of the above considerations, is in Table 3. The two algorithms for the image modeling step that may impact SofC are Multi-Scale algorithm (a.k.a. MSClean), required for narrow-band modeling of morphologically complex

Table 1: Science requirements for the Reference Observing Program science cases. The sum of the observing time fractions (column "Fraction") is higher than 1 (1.23). This is justified by the use of sub-arrays, which allow concurrent observations. The use of "FULL" in the FoV and Bandwidth columns is interpreted as requiring the highest possible values for these quantities. Science cases with zero fraction time and pulsar cases are not shown.

| UC | Name | Fraction | FoV (arcsec) | FWHM (mas) | Dynamic Range | Center Freq. | Bandwidth | Channel Width | Max. Dump Time | Sub-array |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | KSG1 Driving Cont Band 6 eg Taurus disk | 0.09 | 5.0 | 10 | 1.00E+03 | 100.0 GHz | FULL | 120.0 MHz | 1.0 | Main |
| 2 | KSG1 Driving Cont Band 4 eg Taurus disk | 0.04 | 5.0 | 10 | 1.00E+03 | 27.3 GHz | FULL | 120.0 MHz | 1.0 | Main |
| 3 | KSG2 Driving Line Band 5 eg Sgr B2(N) | 0.04 | 60.0 | 100 | 1.00E+03 | 40.5 GHz | 4.0 GHz | 13.5 kHz | 1.0 | Main |
| 4 | KSG2 Driving Line Band 4 eg Sgr B2(N) | 0.01 | 60.0 | 100 | 1.00E+03 | 27.3 GHz | 4.0 GHz | 9.1 kHz | 2.0 | Main |
| 5 | KSG2 Driving Line Band 3 eg Sgr B2(N) | 0.01 | 60.0 | 100 | 1.00E+03 | 16.4 GHz | 4.0 GHz | 5.5 kHz | 2.0 | Main |
| 6 | KSG3 Driving Line Band 5 eg COSMOS | 0.04 | FULL | 1000 | 1.00E+02 | 40.5 GHz | FULL | 675.0 kHz | 1.0 | Plains+Core |
| 7 | KSG3 Driving Line Band 4 eg COSMOS | 0.01 | FULL | 1000 | 1.00E+02 | 27.3 GHz | FULL | 455.0 kHz | 2.0 | Plains+Core |
| 8 | KSG3 Driving Line Band 3 eg COSMOS | 0.01 | FULL | 1000 | 1.00E+02 | 16.4 GHz | FULL | 273.3 kHz | 2.0 | Plains+Core |
| 9 | KSG3 Driving Line Band 6 eg Spiderweb galaxy | 0.02 | 5.0 | 100 | 1.00E+03 | 72.0 GHz | 240.0 MHz | 7.2 MHz | 1.0 | Main |
| 10 | KSG3 Driving Line Band 5 eg Spiderweb galaxy | 0.01 | 5.0 | 100 | 1.00E+03 | 36.0 GHz | 120.0 MHz | 3.6 MHz | 1.0 | Main |
| 11 | KSG3 Driving Line Band 4 eg Spiderweb galaxy | 0.01 | 5.0 | 100 | 1.00E+03 | 27.7 GHz | 92.3 MHz | 2.8 MHz | 2.0 | Main |
| 12 | KSG3 Driving Line Band 6 eg Virgo Cluster | 0.07 | FULL | 100 | 1.00E+03 | 112.5 GHz | 6.0 GHz | 375.0 kHz | 1.0 | Plains+Core |
| 13 | KSG3 Driving Line Band 1 eg M81 Group | 0.11 | FULL | 1000 | 1.00E+03 | 1.4 GHz | 7.0 MHz | 4.7 kHz | 2.0 | Plains+Core |
| 14 | KSG3 Driving Line Band 1 eg M81 Group | 0.13 | FULL | 60000 | 1.00E+03 | 1.4 GHz | 7.0 MHz | 47.3 kHz | 2.0 | Core |
| 15 | KSG5 Driving Cont Band 1 OTF Find LIGO event | 0.07 | FULL | 1000 | 5.00E+03 | 2.4 GHz | FULL | 2.0 MHz | 0.5 | Main |
| 16 | KSG5 Driving Cont Band 4 OTF Find LISA event | 0.07 | FULL | 1000 | 5.00E+03 | 27.3 GHz | FULL | 5.0 MHz | 0.5 | Plains+Core |
| 17 | KSG5+4 Driving Cont Band 2 OTF Find BHs+PossiblePulsars | 0.04 | FULL | 1000 | 5.00E+03 | 7.9 GHz | FULL | 5.0 MHz | 0.5 | Plains+Core |
| 19 | KSG5 Driving Cont Band 3 Gw170817@200Mpc | 0.24 | 1.0 | 1 | 1.00E+02 | 16.4 GHz | FULL | 120.0 MHz | 2.0 | LBA |

Table 2: Derived parameters for the use cases to be considered in the computing sizing estimation for interferometric imaging.

| UC | Fraction | Eff. Dump Time | Eff. Max. Channel | Nchan | Img. Linear Size | N Baseline | Vis/Hour |
|---|---|---|---|---|---|---|---|
| 1 | 0.09 | 1.0 | 90.0 MHz | 223 | 1500 | 22791 | 73.19 GVis |
| 2 | 0.04 | 1.0 | 20.5 MHz | 659 | 1500 | 22791 | 216.28 GVis |
| 3 | 0.04 | 1.0 | 13.5 kHz | 296297 | 1800 | 22791 | 97241.83 GVis |
| 4 | 0.01 | 2.0 | 9.1 kHz | 439561 | 1800 | 22791 | 72129.85 GVis |
| 5 | 0.01 | 2.0 | 5.5 kHz | 727273 | 1800 | 22791 | 119342.01 GVis |
| 6 | 0.04 | 1.0 | 675.0 kHz | 29630 | 345 | 14028 | 5985.35 GVis |
| 7 | 0.01 | 2.0 | 455.0 kHz | 29671 | 512 | 14028 | 2996.82 GVis |
| 8 | 0.01 | 2.0 | 273.3 kHz | 30004 | 855 | 14028 | 3030.45 GVis |
| 9 | 0.02 | 1.0 | 7.2 MHz | 34 | 150 | 22791 | 11.16 GVis |
| 10 | 0.01 | 1.0 | 3.6 MHz | 34 | 150 | 22791 | 11.16 GVis |
| 11 | 0.01 | 2.0 | 2.8 MHz | 34 | 150 | 22791 | 5.58 GVis |
| 12 | 0.07 | 1.0 | 375.0 kHz | 16000 | 960 | 14028 | 3232.05 GVis |
| 13 | 0.11 | 2.0 | 4.7 kHz | 1480 | 7421 | 14028 | 149.48 GVis |
| 14 | 0.13 | 2.0 | 47.3 kHz | 148 | 124 | 4371 | 4.66 GVis |
| 15 | 0.07 | 0.5 | 205.5 kHz | 11194 | 8760 | 22791 | 7347.53 GVis |
| 16 | 0.07 | 0.5 | 5.0 MHz | 2700 | 512 | 14028 | 1090.82 GVis |
| 17 | 0.04 | 0.5 | 1.7 MHz | 5035 | 3003 | 14028 | 2034.17 GVis |
| 19 | 0.24 | 2.0 | 6.2 MHz | 1334 | 3000 | 435 | 4.18 GVis |

Table 3: Table of decisions for the imaging and image-modeling algorithmic requirements. The science cases requiring AW-projection are highlighted.

| UC | Name | W correction | PB correction | Ionospheric correction | Multi-scale | Multi-term |
|---|---|---|---|---|---|---|
| 1 | KSG1 Driving Cont Band 6 eg Taurus disk | No | No | No | Yes | Yes |
| 2 | KSG1 Driving Cont Band 4 eg Taurus disk | No | No | No | Yes | No |
| 3 | KSG2 Driving Line Band 5 eg Sgr B2(N) | No | Yes | No | Yes | No |
| 4 | KSG2 Driving Line Band 4 eg Sgr B2(N) | No | Yes | No | Yes | No |
| 5 | KSG2 Driving Line Band 3 eg Sgr B2(N) | No | Yes | No | Yes | No |
| 6 | KSG3 Driving Line Band 5 eg COSMOS | No | Yes | No | Yes | No |
| 7 | KSG3 Driving Line Band 4 eg COSMOS | No | Yes | No | Yes | No |
| 8 | KSG3 Driving Line Band 3 eg COSMOS | No | Yes | No | Yes | No |
| 9 | KSG3 Driving Line Band 6 eg Spiderweb galaxy | No | Yes | No | Yes | No |
| 10 | KSG3 Driving Line Band 5 eg Spiderweb galaxy | No | Yes | No | Yes | No |
| 11 | KSG3 Driving Line Band 4 eg Spiderweb galaxy | No | Yes | No | Yes | No |
| 12 | KSG3 Driving Line Band 6 eg Virgo Cluster | No | Yes | No | Yes | No |
| 13 | KSG3 Driving Line Band 1 eg M81 Group | Yes | Yes | Yes | Yes | No |
| 14 | KSG3 Driving Line Band 1 eg M81 Group | No | Yes | Yes | Yes | No |
| 15 | KSG5 Driving Cont Band 1 OTF Find LIGO event (QL) | No(Yes) | No(Yes) | No(Yes) | Yes | Yes |
| 16 | KSG5 Driving Cont Band 4 OTF Find LISA event (QL) | No | No(Yes) | No | Yes | Yes |
| 17 | KSG5+4 Driving Cont Band 2 OTF Find BHs +PossiblePulsars (QL?) | Yes | Yes | No | Yes | Yes |
| 18 | KSG5 Driving Cont eg Band 2 Followup from OTF | No | No | No | Yes | Yes |
| 20 | KSG3 Supporting Cont Band 6 eg Virgo Cluster | No | Yes | No | Yes | Yes |
| 21 | KSG3 Supporting Cont Band 5 eg Virgo Cluster | No | Yes | No | Yes | Yes |
| 22 | KSG3 Supporting Cont Band 4 eg Virgo Cluster | No | Yes | No | Yes | Yes |
| 23 | KSG3 Supporting Cont Band 3 eg Virgo Cluster | No | Yes | No | Yes | Yes |
| 24 | KSG3 Supporting Cont Band 2 eg Virgo Cluster | Yes | Yes | No | Yes | Yes |

brightness distributions, and Multi-Term Multi-Scale algorithm (a.k.a. MTMFS), required for wide-band modeling of morphologically complex brightness distributions. These decisions are listed in the last two columns named "Multi-scale" and "Multi-term", respectively, in Table 3. As a rule, fractional imaging bandwidth greater than 30% requires wide-band imaging. Ionospheric corrections are required at lower frequencies when imaging wide fields of view at relatively high dynamic ranges (in the order of 5000 or more). This binary decision is listed in the column named "Ionospheric correction".

Decisions on the use of A-Projection to correct for the effects of the antenna PB, determined by the required FoV, are in the column named "PB correction". KSGs requiring FoV greater than about $0.5\lambda/D$ are listed as requiring A-Projection.

Decisions on the use of algorithms to correct for the effects of the w-term are listed in the column named "W correction". KSGs where $N_w > 1$ (Eq. 1) require corrections for the w-term. The severity of the effects of the w-term can be gauged by the magnitude of $N_w$. `UC 13` (KSG3 Driving Line Band 1 e.g. M81 Group) and `UC 15` (KSG5 Driving Cont Band 1 OTF Find LIGO event), with $N_w = 25$ and 10 respectively, will have the most severe w-term effects. `UC 14, 17` and `24` each have $N_w \approx 2$. `UC 15` (LIGO events) requires only Quick Look (QL) imaging, used for triggering data recording for imaging at a later time. QL imaging can therefore be done without use of expensive algorithms like A- and W-Projection, as is indicated in Table 3. The decision for algorithms required for production (non-QL) imaging is indicated within brackets.

The algorithm of choice for w-term correction is the W-Projection algorithm. As discussed in Sec. 3, the computing cost of the W-Projection algorithm scales with the size of the CFs (which are binned in w-values), and the distribution of the number of visibilities per w-bin. We used a representative data set that measures the compute load for the most severely affected KSG (see Sec. 4). The W-Projection algorithm was set up (within resource constraints) to measure the performance for what we expect to be required by the relevant KSGs.

# 3 Algorithms for forward/reverse transform and scaling laws

The SoC for various imaging algorithms vary by large factors in general. Standard imaging that does not require correction for direction-dependent (DD) effects has a relatively small size of computing than algorithms that have DD corrections, such as W-Projection (for correcting w-term effects only), A-Projection (for correction of primary beam (PB) effects only) or AW-Projection (for correction of both the w-term and PB effects). In order to determine the optimal combination of imaging and deconvolution algorithms needed for each KSG, based on the required imaging dynamic range and FoV, the imaging requirements were broadly classified as

1. narrow-field imaging,
2. wide-field imaging including single pointing imaging of full antenna FoV and mosaic imaging, and
3. spectral cube imaging.

Algorithmic requirements for these categories further differ depending on the frequency of observations. For example, a single pointing wide-field (or mosaic) image at frequencies where the array can be considered to be co-planar will require a PB correction (the A-Projection algorithm) but not a correction for the w-term. Similarly, wide-field imaging at lower frequencies may need corrections for the w-term *and* PB effects, requiring the use of the more expensive AW-Projection algorithm.

For the purpose of sizing the computing load, the cost of the derivative computation step (major cycle) is dominated by, and proportional to, the support size of the convolution functions (CF) (the number of pixels of the CF used during gridding). For standard imaging (no DD corrections), the CF support size is $7 \times 7$ pixels and remains constant as a function of time, frequency and polarization. The CF support for A-Projection starts at about $10 \times 10$ pixels at the lower-frequency end of the band

and increases approximately linearly with frequency. The CF size for W-Projection increases as $w^2$, starting with $7 \times 7$ pixels for $w = 0$. For AW-Projection, the CF size increases both with frequency and $w$.

The scaling law for standard imaging where the CF size does not change is simply

$$C_{Standard} = [12\ FLOP] \times N^{\circ}_{vis} \times S^2 \tag{2}$$

where $N^{\circ}_{vis}$ is the total number of visibilities gridded, $S$ is the size along one axis of the CF. The CF for standard gridding is real-valued. Each visibility is multiplied by a weight (2 multiplications), multiplied by the CF (2 multiplications as the CF is real), and accumulated (2 adds). For Stokes-I imaging it is necessary to grid 2 visibilities, so the number of operations is $2 \times (2+2+2) = 12$. When the CF is complex multiplying it by the weighted visibility takes 6 operations (4 multiplications and 2 additions), so the number of operations is $2 \times (2+6+2) = 20$.

In general, the CF for A-Projection is complex valued. For efficiency the data is partition into blocks of frequency ranges across which a single CF is valid. In the implementation used here this block of frequency range is referred to as the Spectral Window (SPW). A single CF is used per SPW, with the linear size of the CF scaling approximately linearly with frequency. The number of visibilities per SPW is also the same. With $N_{vis}(i)$ and $S(i)$ as the number of visibilities and linear size of the CF for the $i^{th}$ SPW, the scaling law for A-Projection becomes

$$\begin{aligned} C_{AP} &= [20\ FLOP] \times \sum_{i=0}^{N_{spw}-1} N_{vis}(i) \times S^2(i) &(3)\\ &= [20\ FLOP] \times \sum_{i=0}^{N_{spw}-1} \frac{N^{\circ}_{vis}}{N_{spw}} \times S^2_{\circ} \left[\frac{\nu_i}{\nu_{\circ}}\right]^2 &(4) \end{aligned}$$

where $\nu_i$ is the reference frequency of the $i^{th}$ SPW at which the corresponding CF is computed, and $\nu_{\circ}$ and $S_{\circ}$ refer to the frequency and linear size of the CF for the reference SPW respectively. The total bandwidth used for imaging is divided equally into $N_{spw}$ SPWs and $N^{\circ}_{vis}$ corresponds to the total (un-partitioned) visibilities.

The scaling law for W-Projection is similar to Eq. 3:

$$\begin{aligned} C_{WP} &= [20\ FLOP] \times \sum_{w=0}^{W_{Max}-1} N_{vis}(w) \times S^2(w) &(5)\\ &= [20\ FLOP] \times \sum_{w=0}^{W_{Max}-1} N_{vis}(w) \times S^2(w=0) \left[\alpha w^2 + 1\right]^2 \end{aligned}$$

where $N_{vis}(w)$ correspond to the fraction of the total visibilities where the CF with a support size of $S(w)$ is valid. $W_{Max}$ is the largest w-value in the problem for which an independent CF is required. $\alpha$ depends on the distribution of the w-coordinate. This in turn depends on the array geometry, which is best measured from the data used. Figure 2 shows the distribution of the number of visibilities as a function of the magnitude of the W co-ordinate, the resulting FLOP and the cumulative FLOP ($\int_0^w N_{vis}(w) S^2(w) dw$) as a function W for KSG5. The ngVLA has a centrally condensed uv-coverage. The longer baselines for which the FLOPs are higher are significantly fewer in number. As a result, the required FLOPs increase by an order of magnitude across about two orders of magnitude in W values.

Note that the distribution of the w-coordinate as a function of w-bins will be different for other KSGs. However, as mentioned before, while the run-time for full simulation and imaging is resource-limited, we believe that the SofC estimates based on the measured scaling with number of visibilities and w-value is reliable at the accuracy required at this stage. The total number of FLOPs can be seen as the total number of visibilities multiplied by the average of the squared support function ($C = N^o_{vis} \cdot \int_0^w \frac{N_{vis}(w)}{N^{\circ}_{vis}} S^2(w) dw = N^o_{vis} \cdot \langle S^2(w) \rangle$ for W-Projection, and similarly for A-Projection

although in this case the distribution of visibilities is over the frequency). The average depends on the specific distribution of the visibilities over $w$ (or over the frequency in case of A-Projection), an example of which is shown in Figure 2. We assume that the values measured are representative of all KSGs. A more careful measurement would need to use a representative model for each KSG.

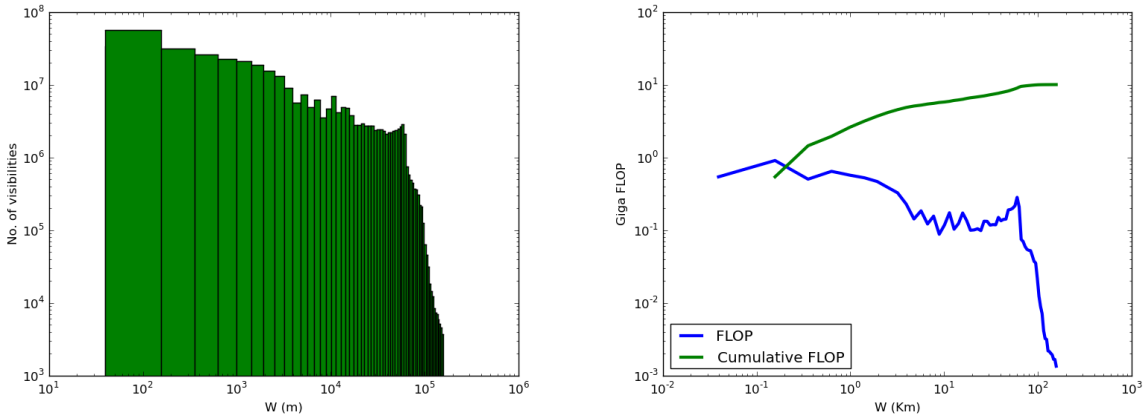


Figure 2: **Left:** Distribution of the number of visibilities as a function of W bins in meters for KSG5. **Right:** The blue curve shows the FLOP as a function of W and the green curve shows the cumulative FLOPs as a function of W ($\int_0^w N_{vis}(w)S^2(w)dw$). The fluctuations in the blue curve follow the fluctuations in the distribution of visibilities in each W-bin. This in turn depends on the antenna configuration in the array.

## 3.1 Extrapolation Methodology

This section explains the methodology by which measurements of a single gridding cycle can be extrapolated to estimate the required system computing size.

The computing load for one gridding cycle can be written as

$$CL_{gridding} = \frac{\text{Visibilities}}{\text{second}} \cdot \frac{\text{FLOPs}}{\text{Visibility}} \tag{6}$$

As explained above, the number of FLOPs per visibility depends on the size of the convolution kernel used during gridding, which has been measured for the standard gridding, A-projection and W-projection.

The total computing load generated by imaging a dataset is proportional to the load generated by the calibration, gridding and deconvolution steps, and can be expressed as

$$CL = k \cdot (CL_{gridding} + CL_{deconv}) + l \cdot CL_{calibration} \tag{7}$$

The $CL_{calibration}$ term is proportional to the number of visibilities acquired during the calibration scans (and target source scans in the case of self-calibration), while $CL_{deconv}$ is roughly dependent on the size of the image. It is assumed that the calibration and deconvolution terms are much smaller than the gridding term and therefore as a first approximation the computing load is only proportional to $CL_{gridding}$:

$$CL = k \cdot CL_{gridding} \tag{8}$$

The use of multi-term imaging is effectively the repetition of standard imaging multiple times, so it can be modeled by an additional factor for the cases where its required. The overhead introduced by the multi-scale algorithm affects only deconvolution term, to they are ignored in this memo.

In order to avoid an unbounded system buffer capacity for accumulating visibilities, on average the system's throughput needs to match the load.

We define the observed computing performance of a single core as:

$$CP^{(obs,sc)} = \frac{\text{FLOPs}}{\text{Exec. time for a single core}}, \tag{9}$$

measured by executing test runs. Similarly, the observed computing performance of a parallel execution is

$$\begin{aligned} CP^{(obs,par)} &= \frac{\text{FLOPs}}{\text{Exec. time for parallel system}} \\ &= CP^{(obs,sc)} \cdot \frac{\text{Exec. time for a single core}}{\text{Exec time for parallel system}} \\ &= CP^{(obs,sc)} \cdot Sp(N_c) \end{aligned} \tag{10}$$

where $Sp(N_c)$ is the speedup.

The observed core performance differs from the nominal core performance by the core efficiency $\epsilon_c$ factor

$$CP^{(sc,obs)} = \epsilon_c CP^{(sc)}, \tag{11}$$

The nominal performance $CP^{(sc)}$ is what gets reported in the core specifications. Achieving that performance would require only doing floating point operations (without accessing memory) using SIMD instructions, and with an insignificant level of branch prediction failures. This parameter is the maximum performance shown in the roofline model in Figure 3. The core efficiency can be estimated from Figure 1 and Figure 3. For this example, the observed core performance is $\approx 0.675$ GFLOPs/s and the nominal performance is 13 GFLOPs/s, which gives a core efficiency of 0.05.

Matching the load with the parallel system throughput:

$$CL = \epsilon_c CP^{(sc)} Sp(N_c) \tag{12}$$

Multiplying and dividing by the number of cores $N_c$, and defining the parallelization efficiency $\epsilon_p = Sp(N_c)/N_c$, the ratio of the real speedup and the ideal speedup:

$$CL = CP^{(sc)} N_c \epsilon_c \epsilon_p \tag{13}$$

The nominal single-core performance times the number of cores is what gets usually reported as the "system performance" of an HPC system. Let's name this $SP = CP^{(sc)} \cdot N_c$.

With this, it is possible to estimate the required system performance as

$$SP = \frac{k \cdot CL_{gridding}}{\epsilon_c \epsilon_p} [\text{FLOPs/s}] \tag{14}$$

The following list summarizes the parameters used in these estimations. Parameters $k$ and $\epsilon_p$ are educated guesses and could be off by a factor of many. It is our intention to refine these estimates either by direct measurement or by parsing through existing execution logs. The multi-term factor is determined by the algorithm.

- $\boldsymbol{k = 100}$: This is a measure of the *effective* number of evaluations of the derivative (major cycles) required for end-to-end processing.

  It typically takes about 10 major cycles for imaging iterations to reach convergence. Manual processing often requires re-imaging several times before a satisfactory image is produced. The heuristics (*e.g.* to refine imaging parameters) in imaging pipelines also require multiple full imaging cycles (*e.g.* VLASS imaging pipeline). Either way, we estimate that about 10 full imaging runs will be necessary, each requiring about 10 major cycles. The value of $k$ is therefore set to 100.

- **Multi-term factor = 3**: Multi-term imaging is necessary when the fractional bandwidth for continuum imaging is greater than 30%. *nterm* is a parameter of the Multi-term algorithm, and the number of gridding/de-gridding operations necessary for Multi-term imaging is given by $\frac{nterm \cdot (nterm+1)}{2}$. For Multi-term imaging we use $nterm = 2$. The value the Multi-term factor is therefore 3.

- $\boldsymbol{\epsilon_c = 5\%}$: This is the efficiency of the gridding/de-gridding code on a single core and is the ratio of the measured FLOPs/s per visibility and the maximum possible FLOPs/s rating for the CPU used. (SKA uses 6.9% for this parameter, see [4].)

- $\boldsymbol{\epsilon_p = 0.8}$: This is the parallelization efficiency — a ratio of the real and the ideal speedup. The SofC estimate has strong dependency on this parameter and its scalability behavior. For the purposes of this memo, we assume that the system exhibits *weak scalability*, meaning that it is possible to maintain the given efficiency by increasing the problem size (the number of visibilities) in proportion to the number of parallel processes. This assumption, which will be verified in a follow-up memo, is supported by the fact that the SofC is dominated by a few KSGs which require spectral cube imaging with a large number of spectral channels. In this kind of imaging, each channel (or a combination of a few channels) can be imaged independently, without the need to synchronize the parallel processes. In general, we can assume that the problem size will grow with the number of processes and the time spent in serial operations will be maintained relatively constant or grow slowly with the problem size. These are required conditions for the speedup to be characterized by Gustafson's Law [3] instead of Amdahl's Law. Gustafson's model doesn't suffer from a saturation of the speedup with the number of processes like Amdahl's. While we haven't yet measured this parameter (also a goal of a follow-up memo), given the characteristics of cube imaging, we expect that a relatively high efficiency value should be possible.

# 4 Measurements

Given the large size of data sets for ngVLA KSGs, the current CASA software, the available NRAO hardware, and the time available to us for completing these measurements, we perform trials using smaller size data sets that are nevertheless representative for ngVLA KSGs. The measurements obtained from trials on smaller size data sets are then used to obtain the scaling laws that will allow estimation of the SoC for ngVLA-sized data sets. The main goal of measuring the performance of current CASA imaging code is to determine the parameters of the scaling laws we use to estimate the SofC for ngVLA imaging.

The three primary classes of metrics we measure are floating point operations (FLOPs), I/O usage and memory usage, each of which may consist of a variety of specific metrics. For example, FLOPs may be measured for single precision and double precision values separately or together. Possible metrics in the I/O class are bytes read and bytes written per second, either at the application level or the system level (which may differ according to the efficiency of page cache access). In order to eliminate effects of multi-processing on the metrics, and to establish a baseline for the future evaluation of parallelization efficiency, the imaging trials have been run exclusively in serial (i.e., not parallel) processing mode.

The execution time of the trials is also an important metric, which can be used to estimate the compute efficiency of current CASA imaging codes. The compute efficiency is a measure of the fraction of the theoretical performance available on a CPU the current CASA codes are able to use. The execution time can also be used to estimate the rate at which visibilities are processed by the imaging codes running on a single CPU. The visibility processing rate can be a useful measure of the overall performance of CASA imaging tasks performed either serially or in parallel.

It is undoubtedly necessary for ngVLA image processing to use parallel processing for efficiency. Although CASA imaging tasks support the use of multi-threading and/or multi-processing, the ability to scale to large numbers of CPUs and/or GPUs is strongly implementation dependent. As a follow-up to this memo, we intend to derive weak and strong scaling curves for the current CASA parallel

processing implementation. The scaling laws derived in this memo provide a baseline against which to compare the efficiency of any parallel code implementation.

## 4.1 Test methods

We use two primary for gathering test metrics of trial imaging runs: native CASA instrumentation, and the `Score-P` infrastructure. The metrics provided by these two tools in combination include all those we require to develop the scaling laws.

### 4.1.1 Native CASA instrumentation

Limited code instrumentation is currently built into CASA, the usage of which is controlled by a `casarc` variable. As indicated by the name of the `casarc` variable, *synthesis.imager.memprofile.enable*, the instrumentation is primarily geared toward profiling memory usage of the imager codes, and it is for the measurement of memory usage that we have employed this CASA capability. The native instrumentation is adequate to the purposes of the present study. Whereas the available alternative of using the `Score-P` instrumentation in tracing mode would provide more detailed metrics, that approach was determined to require too many computer resources to complete all the trials efficiently.

### 4.1.2 `Score-P` instrumentation

`Score-P` bills itself as the "Scalable Performance Measurement Infrastructure for Parallel Codes". The `Score-P` home page, https://www.vi-hps.org/projects/score-p, states

> The `Score-P` measurement infrastructure is a highly scalable and easy-to-use tool suite for profiling, event tracing, and online analysis of HPC applications.

The `Score-P` infrastructure supports the collection of code execution metrics by sampling, profiling or tracing. To gather code metrics at run-time, the user code must be instrumented by one or more varieties of methods. The mode of operation, as well as the selection of metrics, can be specified by the user at run-time, subject to constraints imposed by the code instrumentation methods.

Of all the classes of metrics that are needed for the present study, measures of FLOPs are the most challenging to obtain. Modern processors include built-in performance monitoring units, or *PMU*s, which may provide the only direct measurement of FLOPs available. PMUs promise access to various CPU execution metrics, including, for example, L1 cache hits and misses, or floating point instructions. However, there are various practical issues that often limit the usability or effectiveness of PMU counters. The foremost practical issue is that PMUs are implemented inconsistently by manufacturers, and are often largely undocumented features of CPUs. Another significant practical issue is that available PMU counters often do not quite measure what is most desired by application developers: for example, CPU speculative execution and branch prediction may render a count of floating point operations difficult to interpret unless counters distinguish issued *vs* retired floating point operations. To address some of the problems of using PMU counters directly, the `PAPI` library is available. `PAPI` presents a higher-level interface to PMU counters, and it attempts to provide a better API for programmers by abstracting away some architectural differences in PMUs. However, these features of `PAPI` may make the task of interpreting counter values even more difficult!

In light of the aforementioned difficulties of obtaining FLOPs metrics, the following method was used for the present study. First, a toy program was developed for which the expected number of single and double precision FLOPs could be obtained by calculation. Next, on machines at NRAO of various CPU micro-architectures, different choices of `PAPI` metrics, including direct references through `PAPI` to PMU counters, were selected through `Score-P` at run-time, and the results compared to the expected FLOPs measures. For each of the CPU micro-architectures, for both single and double precision, the most accurate selection or combination of `PAPI` metrics was determined, and then applied through

Table 4: Total floating point operations (FLOPs) for one gridding cycle as a function of the number of visibilities for standard gridding, A-projection, and W-projection, measured using `ptsi`. One gridding cycle incorporates both gridding and de-gridding operations.

| **Standard Gridding** | |
|---|---|
| Number of Visibilities | FLOPS |
| 1.4E+7 (14,586,240) | 5.41E+10 |
| 8.8E+7 (87,517,440) | 1.494E+11 |
| 8.8E+8 (875,174,400) | 1.1796E+12 |
| 3.5E+9 (3,500,697,600) | 4.628E+12 |
| 1.1E+10 (10,502,092,800) | 1.3678E+13 |
| **A-projection** | |
| Number of Visibilities | FLOPS |
| 5.8E+7 (58,344,960) | 4.41E+11 |
| 1.2E+8 (116,689,920) | 8.77E+11 |
| 2.3E+8 (233,379,840) | 1.804E+12 |
| **W-projection** | |
| Number of Visibilities | FLOPS |
| 5.8E+7 (58,344,960) | 1.26E+12 |
| 1.2E+8 (116,689,920) | 2.51E+12 |
| 2.3E+8 (233,379,840) | 5.02E+12 |

`Score-P` at run-time for each of the trials. Based on this method, it is estimated that FLOPs values are accurate to about 10–20%.

# 5 Data analysis

The results of measuring the total number of FLOPs performed by the `ptsi` test program, for data sets of different sizes and for different types of gridding algorithms are shown in Table 4. The number of I/O operations as a function of number of visibilities is presented in Table 5. The parameters derived from the measurements, which are used to calculate the SofC are summarized in Table 6. The rest of this section explains how these parameters were derived.

The curves show good agreement with the expected theoretical results. For standard gridding the expected slope is given by Equation 2, multiplied by 2 as the test program performs gridding twice (one to compute the residual image and one to calculate the PSF): $(7 \times 7) \times 2 \times 12 = 1176$. The measured slope in the plot is 1280.8, the difference explained by a few additional unaccounted operations in the implementation (e.g., FP comparisons). The slopes for A-projection and W-projection are 7472.8 and 21468.4, respectively, introducing factors of 5.8 and 16.76 with respect to standard gridding. They imply *average* support sizes for A-projection and W-projection of 16.9 and 28.7 respectively for these executions.

When two or more convolution kernels need to be combined, the final convolution kernel is the result of convolving the original kernels. If $N_1$ and $N_2$ are the support function sizes of two original kernels, the support size of the convolved function is $N_1 + N_2 - 1$. Given that the measurements above correspond to the application of A and W-projection *combined* with the standard Prolate spheroidal function (which has a support size of 7), the support sizes for the A-projection and W-projection kernels are 11 and 23, respectively. For the cases that require AW-projection, the convolution of the three kernels results in a support size of 39 $(7+11+23-2)$. This corresponds to a factor of 31 $(39^2/7^2)$ in the number of operations with respect to the standard gridding case.

The results for the input/output patterns (Table 5 are in good agreement with theory as well. Writes are relatively constant, corresponding mostly to writing images, with sizes that are independent of the number of visibilities. The fitted slope for the read curve is 32.16, a good match for 2 complex numbers for the visibilities (16 bytes) and 2 floats for the weights (8 bytes). This gives arithmetic

Table 5: Number of of read/write operations as a function of the number of visibilities in the dataset, for one gridding/de-gridding cycle using the `ptsi` test program.

| Number of Visibilities | Read Ops. | Write Ops |
|---|---|---|
| 1.5E+7 (14,586,240) | 1.75E+10 | 1.55E+10 |
| 5.8E+7 (58,344,960) | 1.54E+10 | 7.74E+9 |
| 8.8E+7 (87,517,440) | 2.07E+10 | 1.55E+10 |
| 1.2E+8 (116,689,920) | 2.44E+10 | 1.04E+10 |
| 2.3E+8 (233,379,840) | 4.21E+10 | 1.58E+10 |
| 8.8E+8 (875,174,400) | 4.32E+10 | 1.55E+10 |
| 3.5E+9 (3,500,697,600) | 1.22E+11 | 1.55E+10 |
| 1.1E+10 (10,502,092,800) | 3.59E+11 | 1.55E+10 |

Table 6: Measured parameter summary.

| Algorithm | Operations per Visibility (FLOPs/Vis) | Arithmetic Intensity (FLOPs/Byte) |
|---|---|---|
| Standard Gridding | 1280.8 | 40 |
| A-projection | 7472.8 | 233 |
| W-projection | 21768.4 | 670 |
| AW-projection | 39704.8 | 1240 |

intensities (ratio of number of operations per byte read) of 40 FLOPs/byte for standard gridding, 233 FLOPs/byte for A-projection, and 670 FLOPs/byte for W-projection. The projected arithmetic intensity for AW-Project is 1240 FLOPs/byte. These intensities are shown superimposed in the roofline model graph in Figure 3. The execution is CPU-bound (*i.e.*, not I/O limited) for the different levels of hierarchical memory in the machine. The required file system bandwidth for the system to continue to be CPU-bound is 325 MB/s (13 GFLOPs/s / 40 FLOPs/byte). Note that this is the I/O bandwidth required per core. In order to be able to sustain the high aggregated bandwidth required, strategies like copying data to fast NVME-based local storage can be applied.

# 6 Results and discussion

The size of computing estimation for each ROP use case is shown in Table 7. We emphasize the main result from this table:

> **ngVLA will require a system capable of sustaining a throughput of 50 PFLOPs/second in order to process an ensemble of observations that generates a similar load than the ROP. This places the ngVLA data processing center in the super-computer class.**

The most demanding cases are high data rate spectral line observations with a large number of

Table 7: Require size of computing for each ROP use case.

| UC | Fraction | Data Rate | Required System Perf. |
|---|---|---|---|
| 1 | 9% | 0.081 GB/s | 0.234 PFLOPs/s |
| 2 | 4% | 0.240 GB/s | 0.231 PFLOPs/s |
| 3 | 4% | 108.046 GB/s | 611.764 PFLOPs/s |
| 4 | 1% | 80.144 GB/s | 453.780 PFLOPs/s |
| 5 | 1% | 132.602 GB/s | 750.799 PFLOPs/s |
| 6 | 4% | 6.650 GB/s | 37.655 PFLOPs/s |
| 7 | 1% | 3.330 GB/s | 18.853 PFLOPs/s |
| 8 | 1% | 3.367 GB/s | 19.065 PFLOPs/s |
| 9 | 2% | 0.012 GB/s | 0.070 PFLOPs/s |
| 10 | 1% | 0.012 GB/s | 0.070 PFLOPs/s |
| 11 | 1% | 0.006 GB/s | 0.035 PFLOPs/s |
| 12 | 7% | 3.591 GB/s | 20.333 PFLOPs/s |
| 13 | 11% | 0.166 GB/s | 6.886 PFLOPs/s |
| 14 | 13% | 0.005 GB/s | 0.058 PFLOPs/s |
| 15 | 7% | 8.164 GB/s | 23.512 PFLOPs/s |
| 16 | 7% | 1.212 GB/s | 3.491 PFLOPs/s |
| 17 | 4% | 2.260 GB/s | 202.056 PFLOPs/s |
| 19 | 24% | 0.005 GB/s | 0.013 PFLOPs/s |
| **Average** | | **7.665 GB/s** | **49.606 PFLOPs/s** |

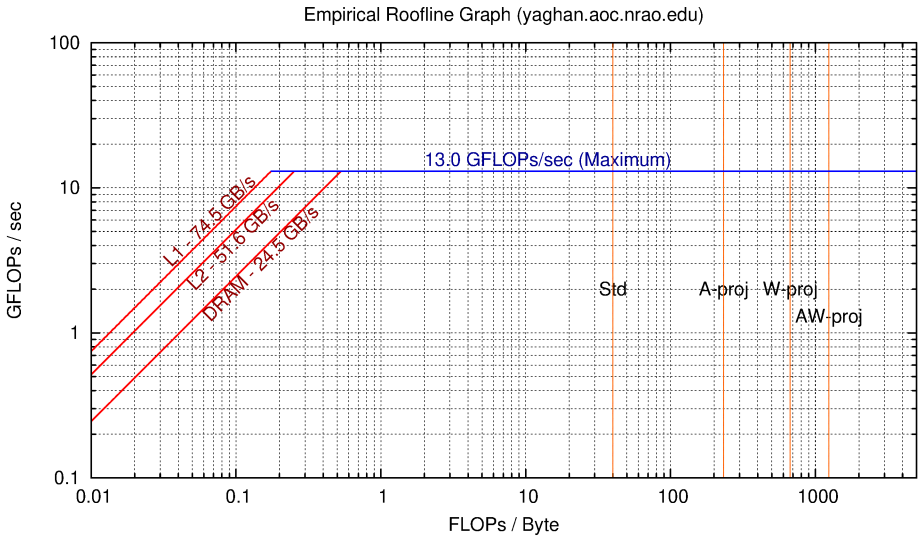


Figure 3: Roofline model with superimposed arithmetic intensities. The roofline model was obtained by using the CS Roofline toolkit (https://bitbucket.org/berkeleylab/cs-roofline-toolkit/src/master/) over the same machine where the example imaging run from Figure 1 was executed. This plot shows how gridding is CPU-bound when running in this machine, i.e., the arithmetic intensities are large enough to not intersect the red lines representing the bandwidth of the hierarchical memory levels. A file-system bandwidth of 325 MB/s would be necessary for the system to continue being CPU-bound. Note that if the maximum performance of the system is increased (the blue horizontal line), then there is a point where the system becomes memory bandwidth constrained. This case is common in GPU implementations, which have both high processing power in the cores, and relatively small memory sizes.

channels (KSG2 cases 3-5). Parallelization for these cases is highly efficient because each channel can be imaged independently. The next most demanding use cases are wide-field high dynamic range imaging cases which are not generally time sensitive (e.g. KSG4+5 case 17), and the rest of the cases demand lower post-processing system performance.

A mitigation strategy for the most demanding low frequency wide fields would be to reduce the effective imaging dynamic range delivered by the project. This mitigation can be removed later, with the expectation that the necessary resources will become more affordable during operations or additional resources can be added to the project (like on-demand Cloud computing). Improvements in algorithms, which sometimes have a dramatic positive effect on the required computing cannot be ruled out.

The time sensitive cases (e.g., LIGO follow-ups) should not demand high computational loads. A rapid imaging mode has been added to the science requirements to address these cases, allowing for prioritization of post processing resources for time-sensitive observations. Even with low parallelization efficiency, this ensures that there should not be a loss of significant follow up science or scientific viability.

The distribution of the computing processing loads generated by each use case is shown in Figure 4. The most demanding cases (KSG2 cases 3-5 and KSG4+5 case 17), although comprising only 10% of all observations, generate roughly a 44 PFLOPs/s load, while all the other observations generate 6 PFLOPs/s. In order to preserve the quality of service for less demanding observations, the computing resources can be partitioned into a 44 PFLOPs/s cluster that is used exclusively to serve them (or similar high processing load observations), and a 6 PFLOPs/s cluster that will process the rest of the observations. Otherwise, computationally simpler observations that have latencies on the order of minutes or hours would queue behind long-running observations that have latencies in the order of days. This partitioning is not necessarily static, it can be made dynamic depending on the characteristics of the pending observations in the processing queue. This is usually taken care by cluster management middleware.

Of course, most individual days will not match the ROP distribution exactly. While the NRAO resources will keep up on average, the latency for individual observations will vary. The computing latency (or processing time) distribution for the reference observing program observations assuming 4-hour duration for each observation is shown in Figure 5. As the computational resources have been sized to sustain the throughput, the distribution of processing latencies is centered around 4 hours. However, given the high variance of computing loads in the specific science cases in the ROP (5 orders of magnitude), the latencies are also widely spread.

# 7 Conclusions

The primary goal of the work described in this memo is to develop a reliable estimate of the ngVLA size-of-computing (SofC). The necessary pre-requisite for this are: (a) identify the algorithms required and develop scaling laws, (b) identify the computing hot-spots for an end-to-end data reduction, and (c) establish a verifiable procedure for measuring the efficiency of the code.

The computing required for imaging contributes dominantly to the end-to-end ngVLA SofC. The cost of imaging is in turn dominated by the computation of the derivative (a.k.a. the "major cycle"; Sec. 1.2). We used simulated data with the data volume in the $10^{7-10}$ visibilities range. While the actual ngVLA data size will be significantly larger, due to resource limitations we cannot use full scale data sets. Instead, we developed theoretical parametric scaling laws (Sec. 3) for the required algorithms (Sec. 2.1) and fitted them to the measured metric to determine the parameters. We performed our tests on a single CPU core, which provides a baseline for the performance of parallel execution tests in the future. For measuring performance, we used a standalone application for the necessary imaging algorithms implemented in the CASA C++ libraries and instrumented it with `Score-P` (Secs. 4,4.1.2) to record various performance metrics. These were then extrapolated to determine the ngVLA SofC for various representative ngVLA KSGs (Sec. 5). We estimate an error of $10-20\%$ on these measurements.

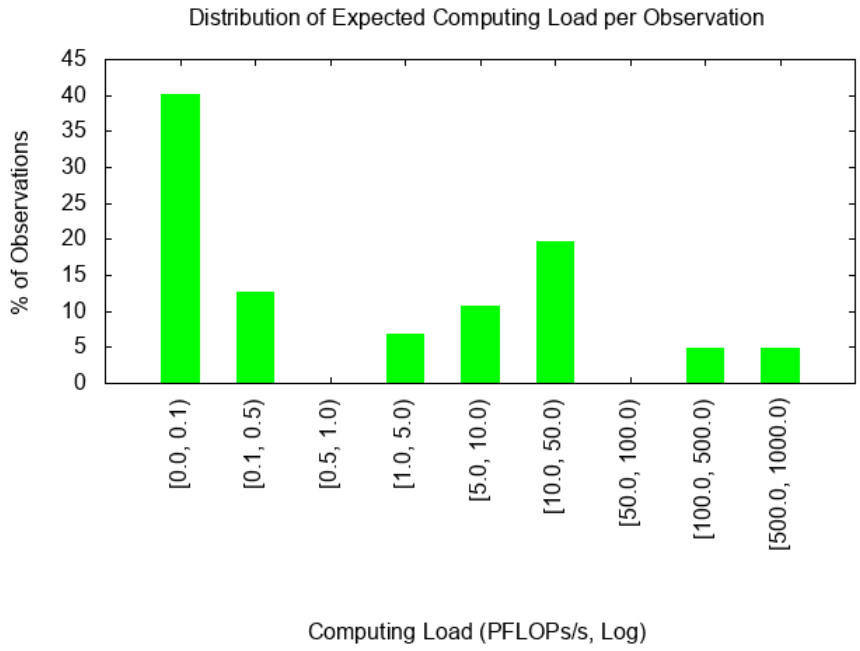


Figure 4: Computing load distribution to sustain throughput for the observations in the ROP.

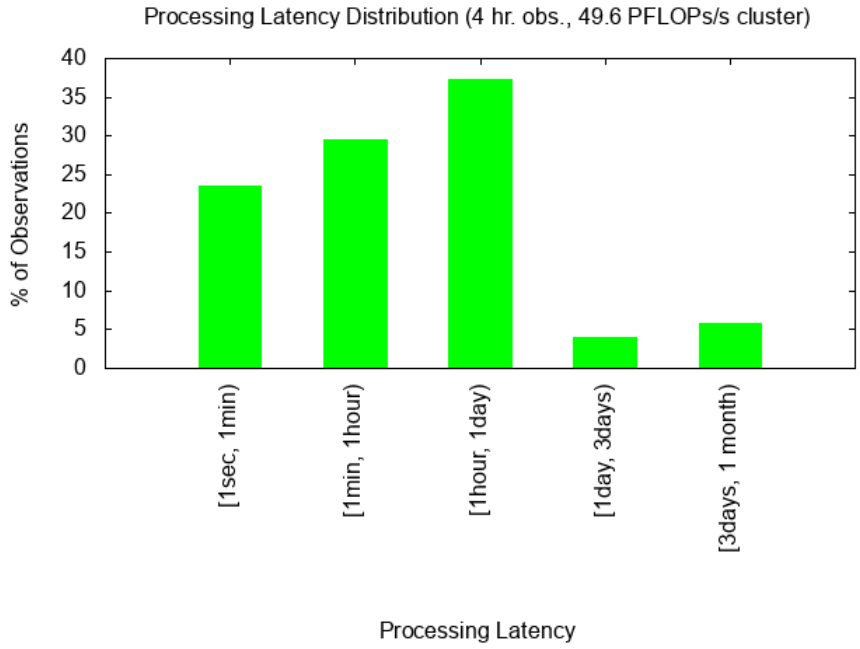


Figure 5: Estimated post-processing latency distribution for the science cases defined in the ROP, assuming that each observation has a duration of 4 hours, and the data is processed by a 50 PFLOPs/s cluster partitioned in 6 PFLOPs/s to serve less-demanding cases, and 44 PFLOPs/s to serve KSG2 and KSG4+5.

Due to measurement errors and need to extrapolate, the SofC estimates are necessarily approximate and should be treated as estimates within a factor of a few. Based on these measurements, we estimate that the ngVLA will require computing system rated for about 50 PFLOPs/s throughput. A few KSGs (specifically, KSG2s) dominate this overall SofC estimate.

Implementations based on GPUs and FPGAs have the potential to significantly reducing the cost of the required system. Our research for exploring implementations on GPUs and FPGAs is on-going. The SofC estimates could also be lower if higher throughput (higher efficiency) implementation is demonstrated on these massively parallel systems.

From these estimates, our conclusion is that the ngVLA processing will require parallel processing at a relatively massive scale. Parallelization efficiency and it's scaling with parallelization breadth are key inputs for designing such a system. Developing a procedure and measuring these parameters is therefore important. Imaging spectral cubes with a large number of spectral channels dominates the overall SofC estimates. Since this requires independent imaging of each spectral channel (or a combination of a few channels), the parallelization efficiency for it can be high. For estimates in this memo, we used a value of 80% for cube-imaging parallelization efficiency. Actual measurement of this parameter is the primary goal of a follow-up memo.

# References


[1] *Simulating ngVLA Data-CASA5.4.1.* URL: https://casaguides.nrao.edu/index.php?title=Simulating_ngVLA_Data-CASA5.4.1.

[2] A. H. Bridle and F. R. Schwab. Bandwidth and time-average smearing. In *ASP Conf. Ser. 180: Synthesis Imaging in Radio Astronomy II*, pages 371–381, 1999. URL: http://adsabs.harvard.edu/cgi-bin/nph-bib_query?bibcode=1999sira.conf....1C&db_key=AST.

[3] John L. Gustafson. Reevaluating amdahl's law. *Commun. ACM*, 1988.

[4] Peter Quinn, Tim Axelrod, Ian Bird, Richard Dodson, Alex Szalay, and Andreas Wicenec. Delivering ska science, 2015. arXiv:1501.05367.

[5] V. Rosero. Sculpting of the synthesized beam and image fidelity study of ksg 1. Technical report, May 2019. URL: "http://library.nrao.edu/public/memos/ngvla/NGVLA_65.pdf".

[6] J. M. Wrobel. A notional reference observing program. Technical report, 020.10.15.05.10-0001-REP-B, Jan. 2020. URL: "https://ngvla.nrao.edu/page/projdoc".

[7] J. M. Wrobel, B. S. Mason, and E. Murphy. A notional envelope observing program. Technical report, 020.10.15.05.10-0002-REP, Aug. 2020. URL: "https://ngvla.nrao.edu/page/projdoc".