Astronomy in the SKA era: SKA-low mini project

Xinyu Zhong
Queens' College

Word count: 1863

1   Calibration Equation

The calibration equation is given by:

$$\mathbf{R} = \mathbf{GMG}^H \tag{1}$$

where $\mathbf{R}$ is the measured visibility, $\mathbf{G}$ is the calibration matrix, $\mathbf{M}$ is the true visibility, and $\mathbf{G}^H$ is the Hermitian transpose of $\mathbf{G}$.

In this equation, $\mathbf{R}$ is a Hermitian matrix. The reasoning is as follows:

$\mathbf{R_{ij}}$ measures the time-cross correlation between measured voltages v_i and v_j from two antennas i and j. Given that the phase difference from i to j is ϕ_{ij}, the phase difference from j to i is $\phi_{ji} = -\phi_{ij}$. Therefore, $\mathbf{R}$ is a Hermitian matrix, i.e. $\mathbf{R}_{ij} = \mathbf{R}_{ji}^*$

Inverse the calibration equation we get:

$$\mathbf{M} = \mathbf{G}^{-1}\mathbf{R}(\mathbf{G}^H)^{-1} \tag{2}$$

Taking the Hermitian transpose of the equation above, we get:

$$\mathbf{M}^H = (\mathbf{G}^{-1}\mathbf{R}(\mathbf{G}^H)^{-1})^H \tag{3}$$
$$= \mathbf{G}^{-1}\mathbf{R}^H(\mathbf{G}^{-1})^H \tag{4}$$
$$= \mathbf{M} \tag{5}$$

where we used the fact that $(\mathbf{G}^H)^{-1} = (\mathbf{G}^{-1})^H$ and $\mathbf{R}^H = \mathbf{R}$.

We obtain the fact that the model $\mathbf{M}$ should also be a Hermitian matrix.

Given that the autocorrelation matrix $\mathbf{R}_{ii}$ are strongly contaminated by the noise, we can then set the diagonal elements of $\mathbf{R}$ to zero. Consequently, the model $\mathbf{M}$ should also have zero diagonal elements.

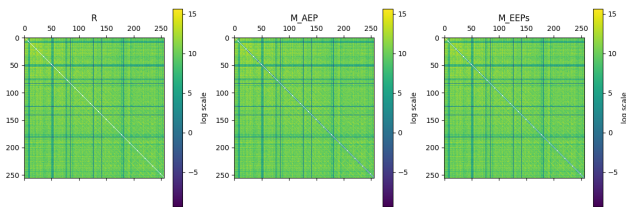Figure 1 shows the plot of log scale of absolute value of $\mathbf{R}$, $\mathbf{M}_{AEPs}$, and $\mathbf{M}_{EEPs}$.

Fig. 1: The plot of log scale of absolute value of $\mathbf{R}$, $\mathbf{M}_{AEP}$, and $\mathbf{M}_{EEPs}$. The horizontal and vertical axis are the antenna indices. The logarithmic value of the diagonal terms are shown as the empty diagonal, as their log scale is negative infinity.

From the plot in Fig. 1, we see that the diagonal elements are zero, hence the empty diagonal as their log scale is negative infinity. The absolute value are symmetric about the diagonal, which is consistent with the fact that $\mathbf{R}$, $\mathbf{M}_{AEP}$, and $\mathbf{M}_{EEPs}$ are Hermitian matrices.

When applying a phase factor of $e^{j\phi}$ to the calibration term, i.e. $\mathbf{G}' = \mathbf{G}e^{j\phi}$ we get:

$$\mathbf{G}'\mathbf{MG}'^H = \mathbf{G}e^{j\phi}\mathbf{M}(\mathbf{G}e^{j\phi})^H$$
$$= e^{j\phi}\mathbf{GMG}^He^{-j\phi}$$
$$= \mathbf{GMG}^H$$

Therefore the residual error

$$||\mathbf{R} - \mathbf{GMG}^H||_F \tag{6}$$

is invariant under multiplication of a phase factor. This means that we would find infinitely many solutions of gains to the calibration equation, each differing by a global phase factor.

2   EEP and AEP

In this section, we compare the power of embedded element pattern (EEP) and average element pattern (AEP) for the 256-element array. The EEP is the power pattern of a single element, while the AEP is the average power pattern of the array.

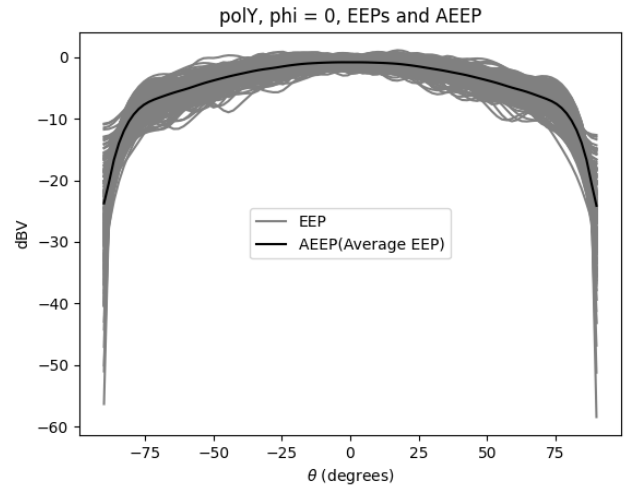The plot of 256 EEPs and the AEP is shown in Fig. 2.

Fig. 2: The plot of 256 EEPs and the AEP. The EEP is the power pattern of a single element, while the AEP is the average power pattern of the array.

The plot in Fig. 2 shows the following results: The variability of the EEPs is small at low absolute θ value (5dbV between the maximum and minimum value at $\theta = 0$) while large at high absolute θ value (about 50dbV between the maximum and minimum value at $\theta = 0$)

3   StEFCal Algorithm

The *StEFCal* algorithm is implemented using python, using the algorithm described in Algo. 14.

One practical issue with the algorithm is that the algorithm may not always converge as the gain matrix could oscillate between two values. The issue was resolved in the original paper by updating the gain with an average of the gain matrix from the previous iteration and the current iteration[1].

Algorithm 1 Algorithm StEFCal

1: Initiate $G^{[0]}$; $G^{[0]} = I$ is adequate in most cases
2: **for** $i = 1, 2, \ldots, i_{\max}$ **do**
3:     **for** $p = 1, 2, \ldots, P$ **do**
4:         $z \leftarrow G^{[i-1]} \cdot M_{:,p} = g^{[i-1]} \odot M_{:,p}$
5:         $g_p \leftarrow (R_{:,p}^H \cdot z)/(z^H \cdot z)$
6:     **end for**
7:     **if** $\mathrm{mod}_2(i) = 0$ **then**
8:         **if** $\|g^{[i]} - g^{[i-1]}\|_F / \|g^{[i]}\|_F \leq \tau$ **then**
9:             Convergence reached
10:         **else**
11:             $G^{[i]} \leftarrow (G^{[i]} + G^{[i-1]})/2$
12:         **end if**
13:     **end if**
14: **end for**

At every even iteration, the algorithm checks for convergence. The algorithm converges if the percentage change in the gain matrix is less than the threshold τ. If the algorithm does not converge, the gain matrix is updated by taking the average of the gain matrix from the previous iteration and the current iteration [1].

The threshold τ in the *StEFCal* algorithm is set to 10^{-5}. It takes 230 iterations to converge if we use M_{AEP} and 192 iterations to converge if we use M_{EEPs}.

The convergence is shown in Fig. 3. The Frobenius norm of the change in the gain matrix compared to the previous iteration is reduced exponentially to 10^{-5}. In the first 50 iterations, the change in the gain matrix is siginificant: The percentage change in the gain matrix is reduced to 10^{-3}. After the first 50 iterations, the change in the gain matrix is reduced to 10^{-5}, and the algorithm converges. We see a linear behaviour in the convergence plot with y-axis set to log scale. It shows that the algorithm converges exponentially as the percentage change reduces exponentially.
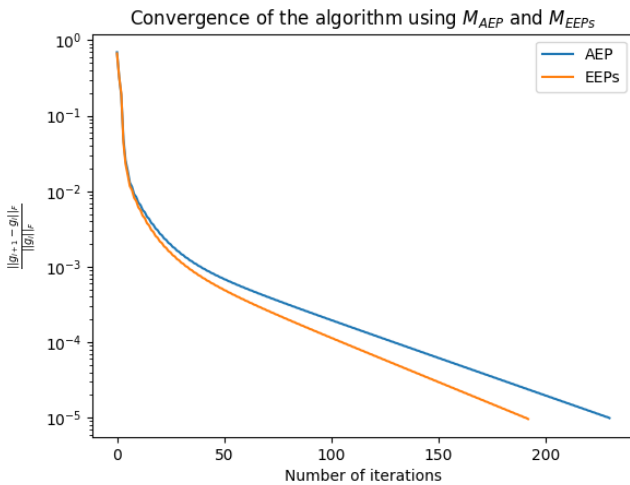

Fig. 3: Convergence of the *StEFCal* algorithm. The y-value is the forbenius norm of the change in the gain matrix compared to the previous iteration.

3.1 Model Optimisation

At every even iteration, if the algorithm does not converge, the gain matrix is updated by taking the average of the gain

matrix from the previous iteration, i.e.

$$
\begin{aligned}
G_i &= \frac{1}{2}(G_i + G_{i-1}) \\
&= G_i - \frac{1}{2}\Delta G_i
\end{aligned}
\tag{7}
$$

where $\Delta G_i = G_i - G_{i-1}$.

This step in the algorithm prevents G from oscillating between two values instead of converging. However, it sets back the optimisation, creating a stepping back effect, evidently seen in the jagged convergence in Fig 3 Therefore, to speed up the optimisation. We could reduce the coefficient in front of ΔG in Eq 7, we call it *correction coefficient*, to reduce the step back effect. The effect on the convergence when estimating using M_{EEPs}, using various *correction coefficient* is shown in Fig. 4.
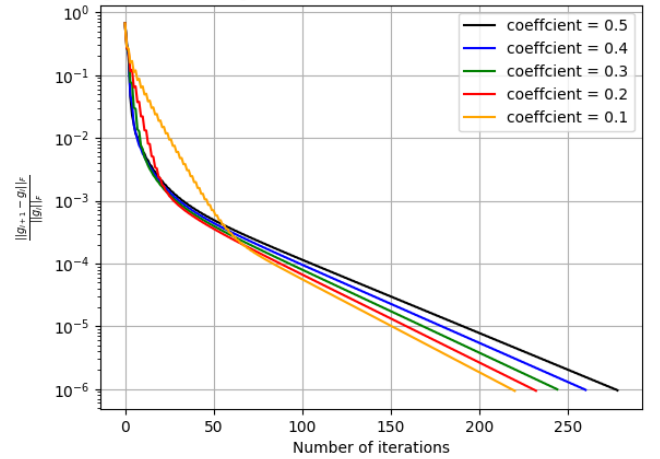

Fig. 4: Optimisation of the *StEFCal* algorithm. The y-value is the forbenius norm of the change in the gain matrix compared to the previous iteration. Coefficient = 0.5 is the original algorithm shown in Alg. 14. The convergence rate to reach $\tau = 10^6$ increases with decreasing coefficient. i.e. update g_i with a lesser contribution from the previous iteration, g_{i-1}.

The plot in Fig. 4 shows that the convergence rate to reach $\tau = 10^{-6}$ increases with decreasing coefficient. The number of steps required reduces from about 280 iterations to 230 iterations when the coefficient is reduced from 0.5 to 0.1. Therefore, the performance of the algorithm can be improved by reducing the coefficient in front of ΔG in Eq 7 to 0.1.

4 Error analysis

In this section we take the average error of the gain matrix over 256 elements. The absolute error is the defined as the average of the difference between the true gain matrix and the estimated gain matrix. The absolute error in amplitude is the average difference between the amplitude of each element in the true gain matrix and the estimated gain matrix. The absolute error in phase is the average difference between the phase of each element in the true gain matrix and the estimated gain matrix.

The absolute error, absolute error in amplitude, and absolute error in phase, for both G estimated from M_{EEPs} and $M_A EP$ are shown in Fig. 5.
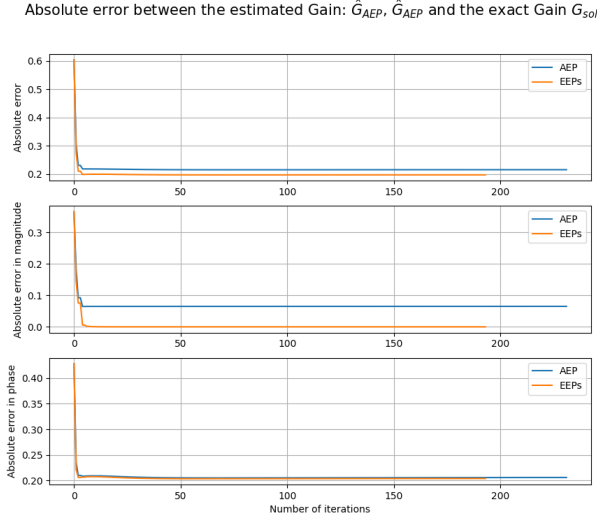

Fig. 5: The absolute error of the *StEFCal* algorithm. Top: The absolute error between G_{sol} and G. Middle: The absolute error of magnitude of the gain. Bottom: the absolute error of the phase of the gain. Orange lines are error of using M_{EEP}, and blue lines are error of using M_{AEPs}.

Fig. 5 shows the following results:

1. Using M_{EEPs} gives a lower error in the gain matrix compared to using M_{AEPs}. The average error is reduced to 0.20 compared to 0.22 when using M_{AEPs}.

2. The algorithm is performs well in reducing the error of the amplitude of the gain matrix compared to the phase of the gain matrix.

3. The error converges to in the first 5 iterations of the algorithm.

$\mathbf{M_{EEPs}}$ give lower error: Using M_{EEPs} gives a lower error in the gain matrix compared to using M_{AEPs}. The average error is reduced to 0.20 compared to 0.22 when using M_{AEPs}. The error in the amplitude of the gain matrix is reduced to 0.01 compared to 0.06 when using M_{AEPs}. The error in the phase of the gain matrix is reduced to 0.21 compared to 0.22 when using M_{AEPs}. Therefore, the usage of M_{EEPs} if more accurate than using M_{AEPs} by a significant margin.

Phase error converges to non-zero value: The error in amplitude converges to 0 for M_{EEPs} while the error in amplitude converges to 0.06 for M_{AEPs}. On the other hand, the error in the phase converges to 0.21 for M_{EEPs} and 0.22 for M_{AEPs}. It shows that the algorithm is more effective in reducing the error in the amplitude of the gain matrix compared to the phase of the gain matrix. This is because that the Frobenius norm in Eqn. 6 is invariant under multiplication of a phase factor. Therefore, the algorithm would converge to gain that has a global phase factor compared to the true gain matrix.

Convergence in the first 5 iterations: Most of the error is reduced in the first 5 iterations of the algorithm. This is also seen in the convergence plot in Fig. 3, where the percentage change Frobenius norm reduces to less than 0.01 after about 5 iterations.

5   Calibration of beamformed voltages

The beamformed voltages are given by:

$$b_i = \sum_i^{N_{ant}} \hat{w}_i v_i e^{-ik \sin\theta \cos\phi x_i + ik \sin\theta \sin\phi y_i} \tag{8}$$

where b is the beamformed voltages, $\hat{w}_i$ is the weight of the i-th antenna, and v_i is the voltage of the i-th antenna. The weight of the antennas is given by:

$$\hat{w}_i = e^{jk(\sin\theta_0 \cos\phi_0 x_i + \sin\theta_0 \sin\phi_0 y_i)} \tag{9}$$

where k is the wave number, θ_0 is the zenith angle, ϕ_0 is the azimuth angle, and x_i and y_i are the x and y coordinates of the i-th antenna.

The beamformed voltages in the plane $\phi = 0$ is calculated and calibrated using three different gain solutions: G_{sol}, G_{EEPs}, and G_{AEPs}, which is the true gain matrix, the gain matrix using M_{EEPs}, and the gain matrix using M_{AEPs}, respectively. The beamformed voltages are shown in Fig. 6.
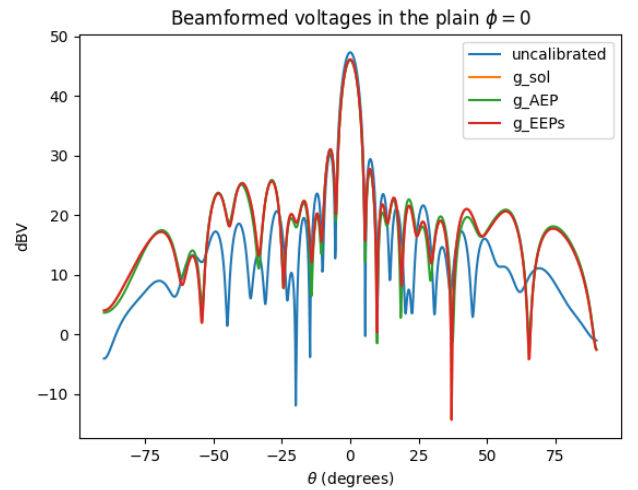

Fig. 6: The beamformed voltages in the plane $\phi = 0$. The beamformed voltages are calibrated using three different gain solutions: G_{sol}, G_{EEPs}, and G_{AEPs}, which is the true gain matrix, the gain matrix using M_{EEPs}, and the gain matrix using M_{AEPs}, respectively. Note that the calibrated beamformed voltages using G_{sol} overlap with the calibrated beamformed voltages using G_{EEPs}.

The residual error of the beamformed voltages is shown in the Fig. 7. The residual error is calculated as the difference between the beamformed voltages using the true gain matrix and the beamformed voltages using the estimated gain matrix.
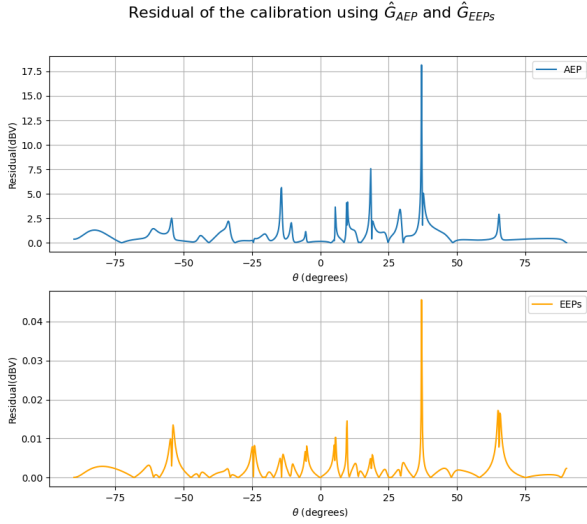
Fig. 8: Uncalibrated station beam pointed at $\theta_0 = 40(degree)$ and $\phi_0 = 80(degree)$.

Fig. 7: The residual error of the beamformed voltages in the plane $\phi = 0$. The residual error is calculated as the difference between the beamformed voltages using the true gain matrix and the beamformed voltages using the estimated gain matrix, G_{EEPs} and G_{AEPs}.

The residual plot in Fig. 7 shows the following results:

1. The residual error varies across the angle θ. However, the distribution of the error is similar for G_{EEPs} and G_{AEPs}.

2. The residual error is about 10 times lower when using G_{EEPs}(about 0.01 dBV)compared to G_{AEPs}(about 1.0 dBV).

6 Calibrated Station Beam

The station beam is then calibrated by the most accurate gain solution, G_{EEPs}. The calibrated station beam is given by:

$$b = \sum_{i}^{N_{ant}} \hat{w}_i G_{EEPs} v_i e^{-ik\sin\theta\cos\phi x_i + ik\sin\theta\sin\phi y_i} \quad (10)$$

where the weight of the antennas is given by Eq. 9 with $\theta_0 = 40(degree)$ and $\phi_0 = 80(dergee)$. The uncalibrated station beam and calibrated beam is shown in Fig. 8 and Fig. 9, respectively.

Fig 8 and Fig 9 shows the contour plot of the uncalibrated station beam and the calibrated station beam, respectively. The image has double peaks at about $(\theta = 15°, \phi = 90°)$,and $(\theta = 15°, \phi = 270°)$, and a saddle point at the origin. The calibrated station beam is more focused on the direction of the source compared to the uncalibrated station beam.
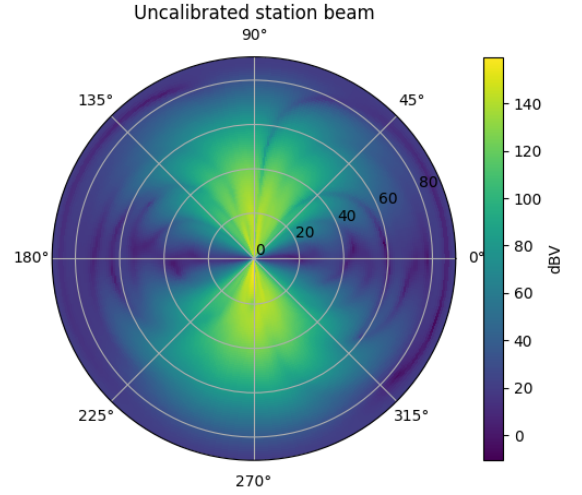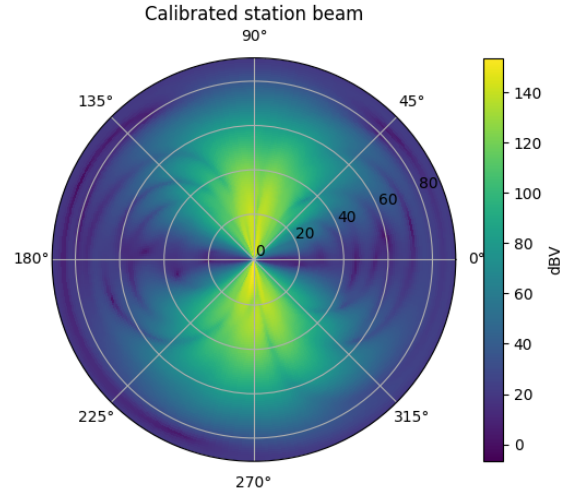

Fig. 9: The calibrated station beam using the most accurate gain solution, G_{EEPs}, pointed at $\theta_0 = 40(degree)$ and $\phi_0 = 80(degree)$.

A   Use of generative AI

Chat-GPT 3.5, accessible for free at https://chat.openai.com/, was utilized for certain small sections of the source code. These instances, where using Chat-GPT provided a more efficient solution than other sources could offer, are clearly marked in the code with comments. Additionally, Chat-GPT helped in formatting some of the tables in this report using LATEX. The concepts and execution of this project are entirely my own, built upon the knowledge gained from this course's materials, my own experiences, and the cited references.

References

[1]   Stefano Salvini and Stefan J. Wijnholds. "Fast gain calibration in radio astronomy using alternating direction implicit methods: Analysis and applications". In: 571 (Nov. 2014), A97. ISSN: 1432-0746. DOI: 10.1051/0004-6361/201424487. URL: http://dx.doi.org/10.1051/0004-6361/201424487.