Calibration
Chapter 8

Fundamentals of Radio Interferometry

Trienko Grobler
Rhodes University

NASSP 2016

Outline

- Least Squares Minimization
- Calibration as a least squares problem
- 1GC calibration
- 2GC calibration
- 3GC calibration

Least Squares Minimization

- Given a **model** and **some data**, we want to find the values of a **set of parameters** which **minimize the difference** between our **model** and our **data**.
- We will refer to our **data vector** as ***d*** and our **model vector** as ***m***.
- These vectors contain the **measured values** and those **predicted** by the model respectively.
- We wish to minimize the **Euclidean vector norm of their difference**.
- ***r*** is the **residual vector** and it is a measure of the difference between the values predicted by our model and the observed values.
- It is important to note that in general ***m*** is a function of a number of **parameters**, such as (x_1, x_2, x_3, ...). These parameters form the parameter vector ***x*** which is what we ultimately want to determine.

$$\|\mathbf{r}\| = \|\mathbf{d} - \mathbf{m}\| = \sqrt{\sum_{i=1}^{N} (d_i - m_i)^2}.$$

Gauss-Newton

$$\delta\mathbf{x} = (\mathbf{J}^T\mathbf{J})^{-1}\mathbf{J}^T\mathbf{r}.$$

- δ***x*** is **simply the update** to the current best guess of the **parameter vector**.
- **J** is the **Jacobian** of the problem which we will discuss in detail shortly.
- $(.)^T$ denotes a **matrix transpose**, and $(.)^{-1}$ denotes a **matrix inverse**.
- **Gauss-Newton** is an **iterative algorithm** which starts from some **initial guess** which is the **updated** in accordance with:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \delta\mathbf{x}$$

Jacobian

- The **Jacobian** is simply a **matrix** of the **first derivatives** of the **model term** **relative to** the **parameter vector**.
- This can be written analytically for a model vector of length *M* and a parameter vector of length *N* as:

$$\mathbf{J} = \frac{\delta \mathbf{m}}{\delta \mathbf{x}} = \begin{bmatrix} \frac{\delta m_1}{\delta x_1} & \frac{\delta m_1}{\delta x_2} & \cdots & \frac{\delta m_1}{\delta x_N} \\ \frac{\delta m_2}{\delta x_1} & \frac{\delta m_2}{\delta x_2} & \cdots & \frac{\delta m_2}{\delta x_N} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\delta m_M}{\delta x_1} & \frac{\delta m_M}{\delta x_2} & \cdots & \frac{\delta m_M}{\delta x_N} \end{bmatrix}$$

- This convention is somewhat unique to the radio interferometry problem. The Jacobian is usually defined as the derivative of the residual vector relative to the parameter vector.

Levenberg-Marquardt

$$\delta \mathbf{x} = \left(\mathbf{J}^T\mathbf{J} + \lambda_{LM}\mathbf{D}\right)^{-1}\mathbf{J}^T\mathbf{r}.$$

- It is used **more frequently** as it has **better convergence behaviour** than basic Gauss-Newton.
- There is a degree of **choice** regarding the matrix **D**. However, in practice it is usually the identity matrix, **I**, or a matrix containing the diagonal entries of $\mathbf{J}^{\mathrm{T}}\mathbf{J}$.
- The **lambda factor** is used to **tune the algorithm** and improve its convergence.

scipy.optimize.leastsq

Example: fit d_i to $m_i = x_1 \sin(2\pi x_2 t_i + x_3)$

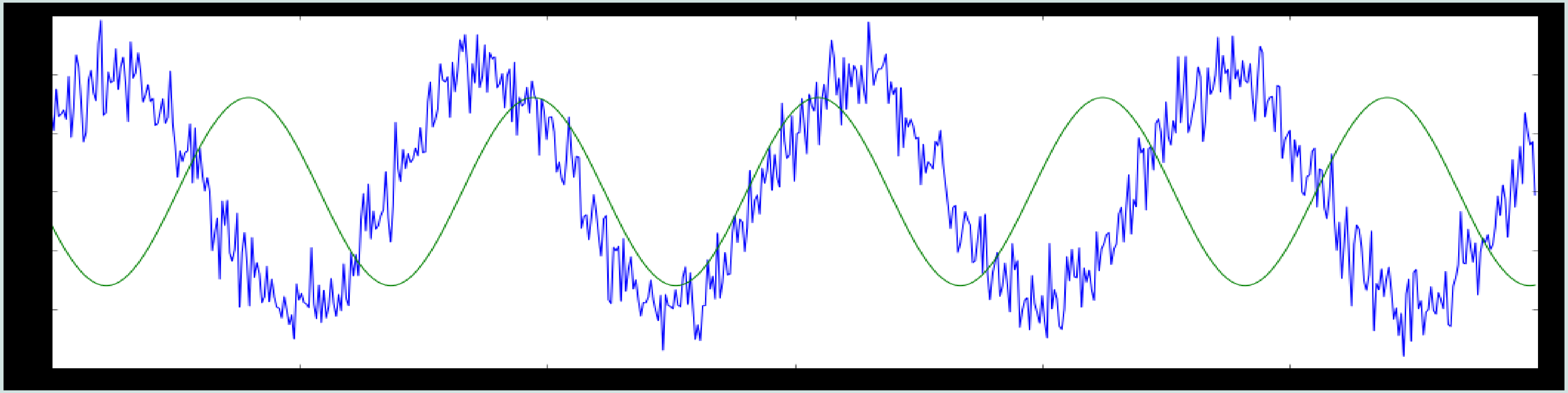
```
x_new = leastsq(residual_func, x_0, args=(t, d))
```

- `scipy.optimize.leastsq` – built in scipy least squares function that implements the Levenberg-Marquardt algorithm.
- `residual_func` – python function which computes $r_i = d_i - m_i$
- `x_0` – starting parameter vector.
- `t` – vector containing the sampling points.
- `d` – vector containing the observed values.
- `x_new[0]` – estimated parameter vector.
- What happened to **J**? It is numerically determined inside `leastsq`.

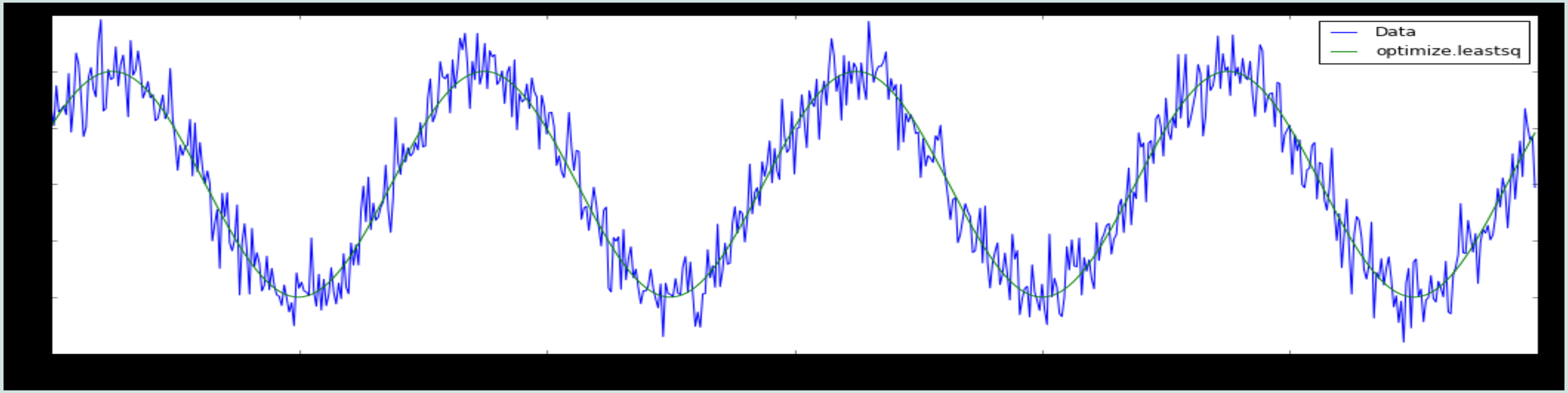
$$\mathrm{J} = [\sin(2\pi x_2 t_i + x_3)\ 2\pi x_1 t_i \cos(2\pi x_2 t_i + x_3)\ x_1 \cos(2\pi x_2 t_i + x_3)$$

Example

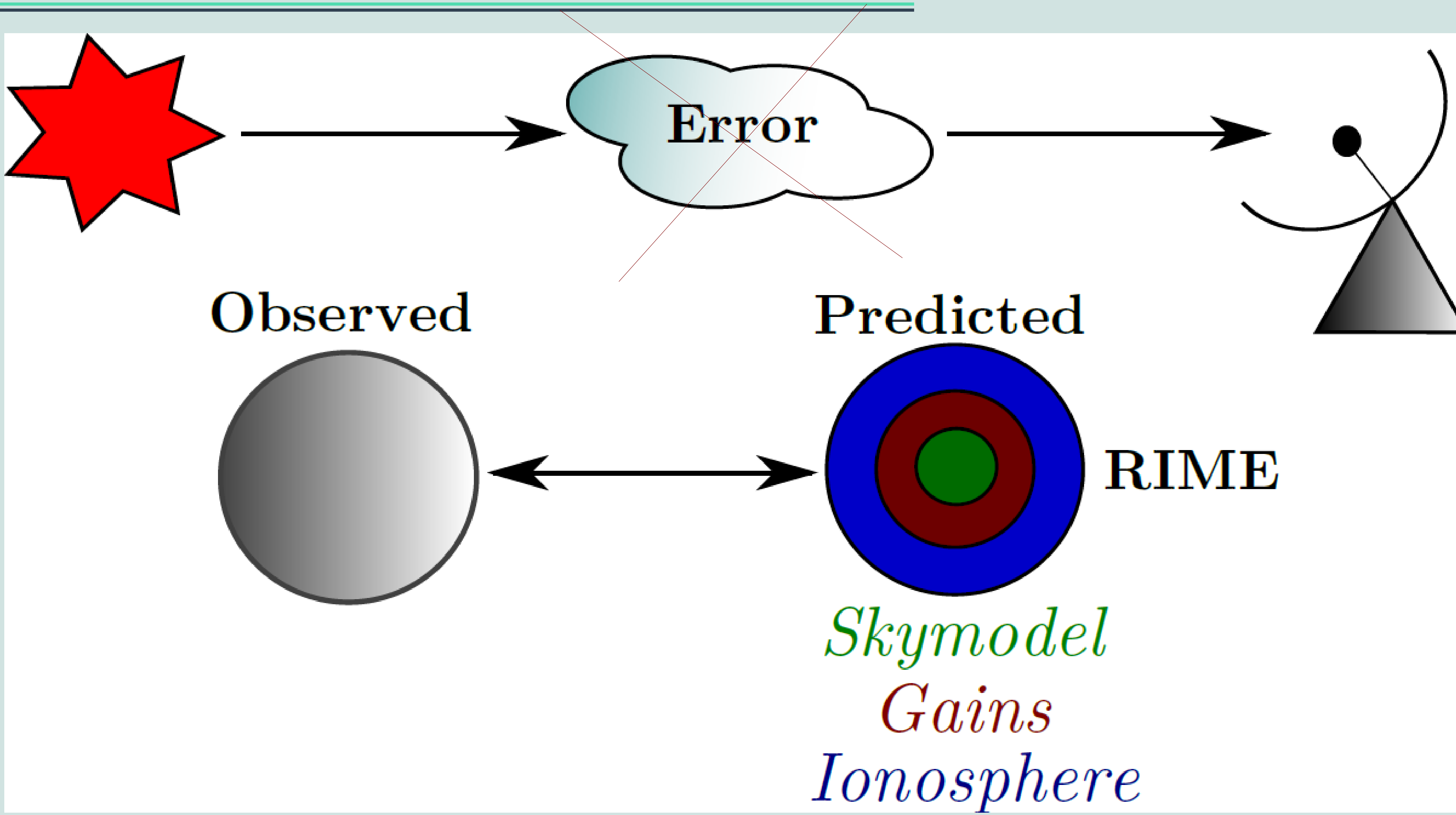
Blue – observed data; **Green** – model evaluated at `x_0`



Blue – observed data; **Green** – model evaluated at `x_new[0]`



Calibration



Interferometric data gets **corrupted** by **environmental and instrumental effects**. **Calibration** is the procedure by which we try to **eliminate** the errors induced by the aforementioned effects.

Unpolarized Calibration

$$d_{pq}(t) = g_p(t)g_q^*(t)m_{pq}(t) + \epsilon_{pq}(t)$$

- $d_{pq}(t)$ and $m_{pq}(t)$ denote the **corrupted observed** and model **visibility** at time t associated with baseline pq.
- the factors g_p and g_q denote the complex gain of antenna p and q.
- the term ε_{pq} is a zero mean (Gaussian) noise term, representing thermal noise

$$\min_{\mathbf{g}} \sqrt{\sum_{pq} |d_{pq} - g_p g_q^* m_{pq}|^2}$$

- **Calibration** entails finding the **antenna gains** which **minimizes** the **difference** between our **observed** and **predicted visibilities**.

An equivalent matrix formulation

$$\min_{\boldsymbol{G}} \left\| \boldsymbol{D} - \boldsymbol{G}\boldsymbol{M}\boldsymbol{G}^{H} \right\|$$

- **D** is the **observed visibility matrix**. Each entry, which we denote by d_{pq}, represents the visibility measured by the baseline formed by antennas *p* and *q*.
- **M** is the **model visibility matrix**. The entry m_{pq} of **M** denotes a true or model visibility which was created with the calibration sky model and a *uv*-point on the *uv*-track associated with baseline *pq*.
- **G** =diag(**g**) is the **antenna gain matrix**, where $\mathbf{g}=[g_1,g_2,\ldots,g_N]^T$ denotes the **antenna gain vector**. The vector **g** represents the instrumental response of the antennas, i.e. the complex antenna gains.
- $\mathbf{GMG}^H$ denotes the **predicted visibilities**.

Calibration: least squares problem

Problem

$$\min_{\breve{\mathbf{g}}} \|\mathbf{r}(\mathbf{m},\mathbf{d},\breve{\mathbf{g}})\|$$

$$\mathbf{r}(\mathbf{m},\mathbf{d},\breve{\mathbf{g}}) = \mathbf{d} - f(\mathbf{m},\breve{\mathbf{g}})$$

Vectorization

$$\mathbf{d} = [\mathrm{vec}(\Re\{\boldsymbol{D}\})^T, \mathrm{vec}(\Im\{\boldsymbol{D}\})^T]^T$$

$$\mathbf{m} = [\mathrm{vec}(\Re\{\boldsymbol{M}\})^T, \mathrm{vec}(\Im\{\boldsymbol{M}\})^T]^T$$

$$\breve{\mathbf{g}} = [\Re\{\mathbf{g}^T\}, \Im\{\mathbf{g}^T\}]^T$$

$$f(\mathbf{m},\breve{\mathbf{g}}) = [\mathrm{vec}(\Re\{\boldsymbol{G}\boldsymbol{M}\boldsymbol{G}^H\})^T, \mathrm{vec}(\Im\{\boldsymbol{G}\boldsymbol{M}\boldsymbol{G}^H\})^T]^T$$

Matricization

$$\boldsymbol{G}(\breve{\mathbf{g}}) = \mathrm{diag}(\breve{\mathbf{g}}_U) + i\,\mathrm{diag}(\breve{\mathbf{g}}_L)$$

$$\boldsymbol{M}(\mathbf{m}) = \mathrm{vec}^{-1}(\mathbf{m}_U) + i\,\mathrm{vec}^{-1}(\mathbf{m}_L)$$

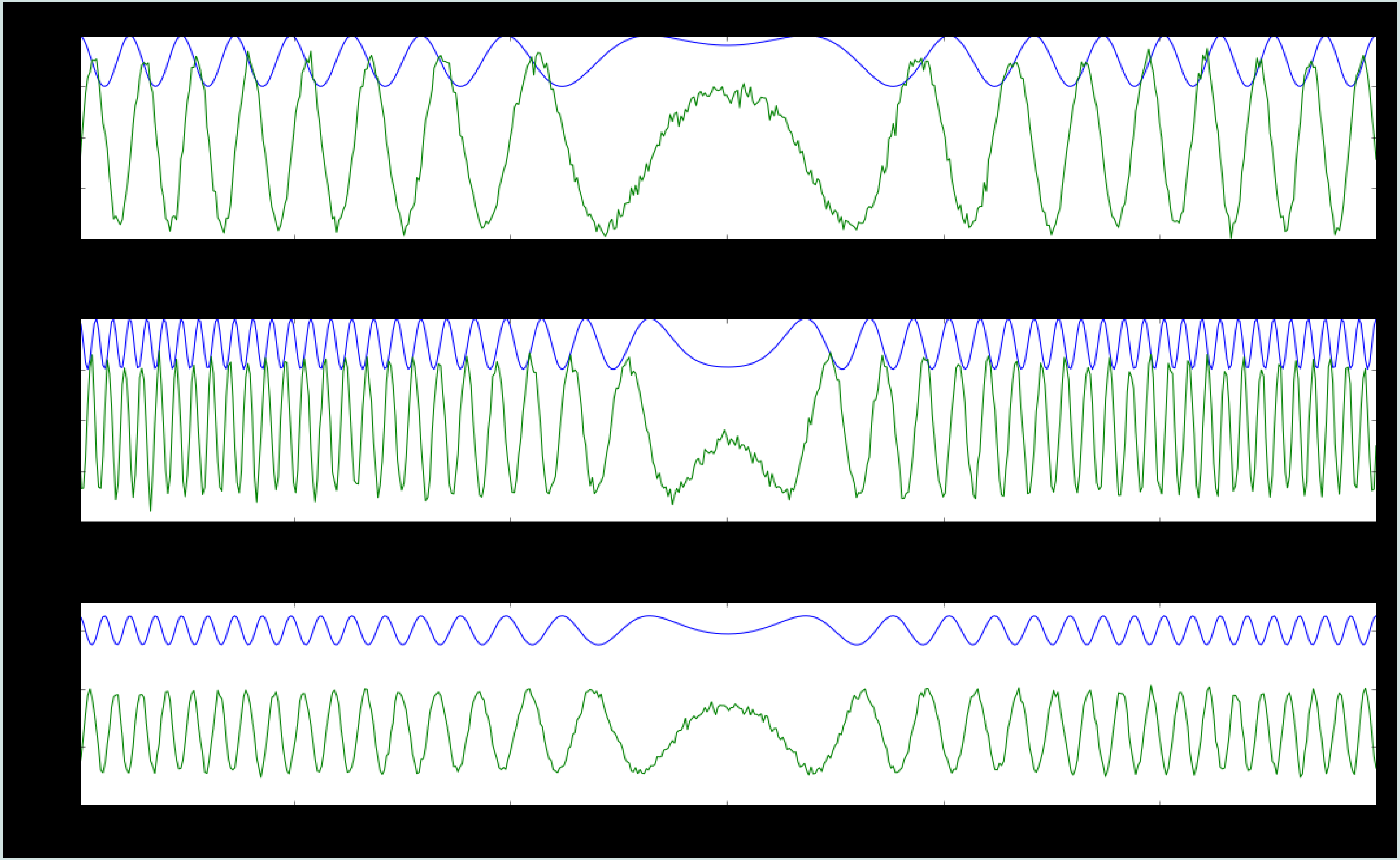
Calibration: `optimize.leastsq`

- First vectorize the **real and imaginary** part of **D**, i.e. construct **d**.
- The vector **m** is generated in a similar manner.
- We then need to create a function `err_func` which calculates $\mathbf{r} = \mathbf{d} - f(\mathbf{m}, \breve{\mathbf{g}})$.
- Initialize $\breve{\mathbf{g}}_0 = [\mathbf{1},\mathbf{0}]^T$.
- `b_g = optimize.leastsq(err_func, b_g_0, args=(d, m))`.
- Construct $\mathbf{g} = \breve{\mathbf{g}}_U + i\breve{\mathbf{g}}_L$.
- We repeat the above for each time-slot.

NB Calibration is a complex problem:

- During calibration we **split** the problem into a **real and imaginary** part and then we **optimize** the real and imaginary parts of the parameter vector simultaneously.
- The **reason** for this is that **differentiation** (needed to calculate the Jacobian) is in general **not defined** in complex space.

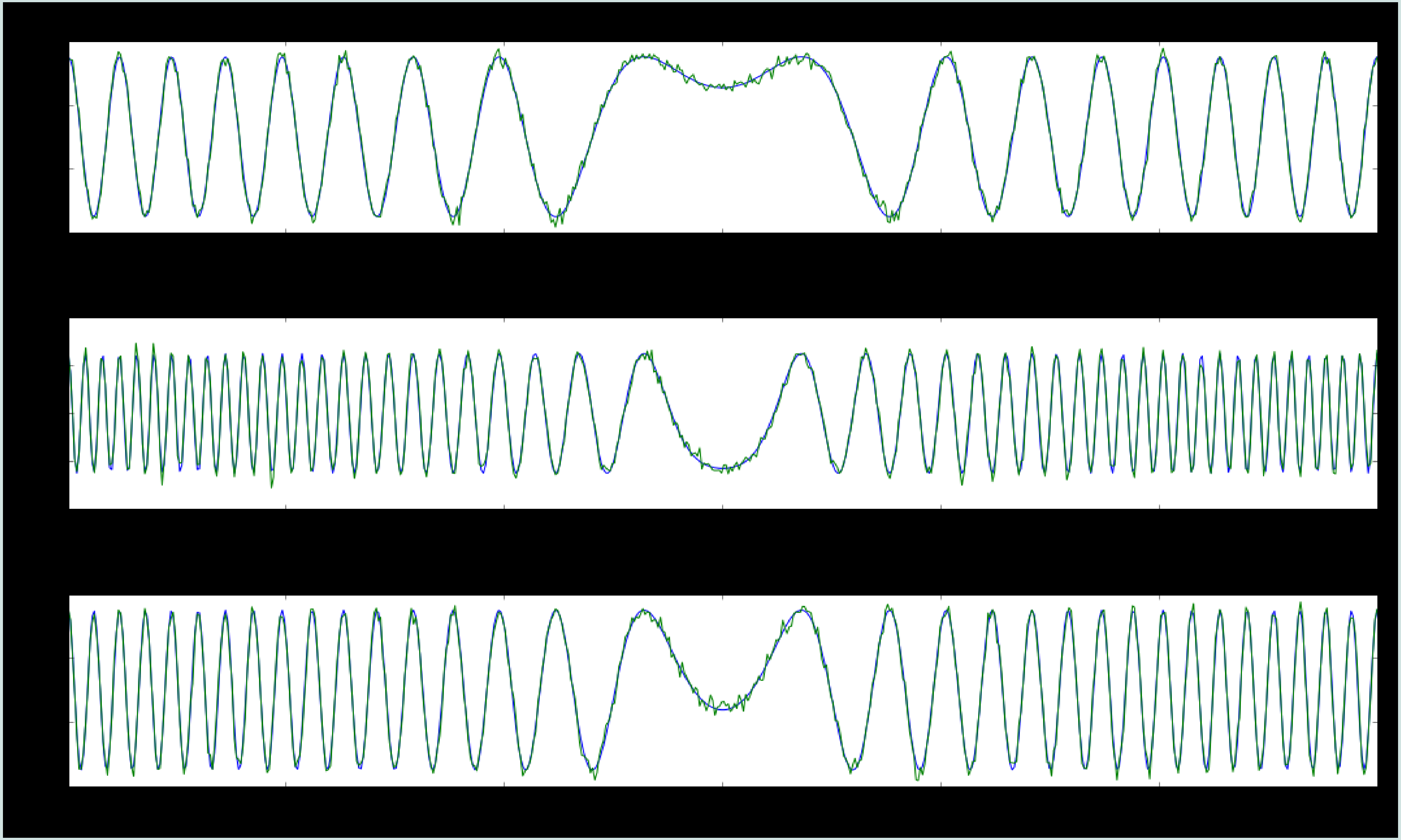
Example: corrupted visibilities



Blue – true visibilities, **Green** – corrupted visibilities

Example: corrected visibilities

$$\boldsymbol{D}^{(\mathrm{c})} = \boldsymbol{G}^{-1}\boldsymbol{D}\boldsymbol{G}^{-H}$$



Blue – true visibilities, **Green** – corrected visibilities

StEFCal

- **StEFCal** is an **alternating direction implicit method**. It works by first solving $\mathbf{G}^H$ with $\mathbf{G}$ held constant and then solving $\mathbf{G}$ with $\mathbf{G}^H$ held constant. This is tantamount to **linearising the calibration problem**.
- StEFCal lowers the **execution time** from order N^3 to N^2.
- As $\mathbf{D}$-$\mathbf{G}\mathbf{M}\mathbf{G}^H$ is Hermitian, the two steps are equivalent, which ultimately leads to the following update step:

$$g_p^{[i]} = \frac{\boldsymbol{D}_{:,p}^H \boldsymbol{Z}_{:,p}^{[i-1]}}{\left(\boldsymbol{Z}_{:,p}^{[i-1]}\right)^H \boldsymbol{Z}_{:,p}^{[i-1]}} \qquad \boldsymbol{Z}^{[i]} = \boldsymbol{G}^{[i]}\boldsymbol{M}$$

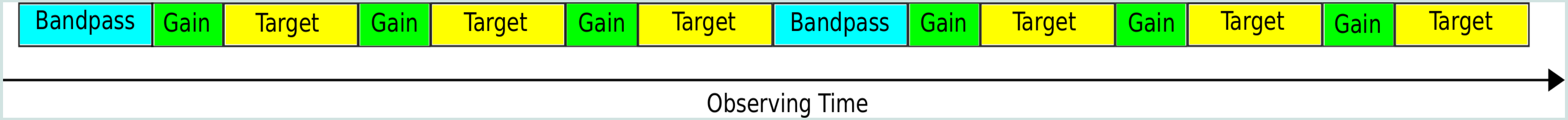
- $\mathbf{A}_{:,p}$ denotes the *p*th column of **A.**
- In practice we replace the **gain solution** of **each even iteration** by the **average** of the **current gain solution** and the **gain solution** of the **previous odd iteration**.
- $\mathbf{G}^0 = \mathbf{I}$

1GC Calibration

- 1GC is performed using **calibrator observations**.
- These are observations of a **source with known parameters** such as flux, shape and spectrum.
- Observations of calibrators are **interspersed** with observations of the target field.
- This is done so that the calibrator observations **track changes** in the observational parameters.
- Thus, it is possible to solve for **calibrator gain solutions** which can then be **transferred to the target field**.
- It is also possible to do more complicated **solution transfers** by **interpolating between values** or fitting curves across the solutions.
- This is usually an effective method of **removing large-scale errors** in the visibilities.
- 1GC can be performed by using the **least-squares approach already discussed** (or a simplified version of it if the calibrator field contains only one source).

Calibration Quantities

- **Absolute flux calibration** is used to determine the true flux of sources in the field.
- **Bandpass calibration** is used to correct for errors along the frequency axis of the observation.
- **Delay calibration** is used to remove the phase delay error which manifests as a linear ramp in the bandpass.
- **Gain calibration** is used to determine the complex valued gains.
- In practice, **absolute flux, delay, and bandpass calibration** can all be performed using the same calibrator. Gain calibration could also be performed using this calibrator, but only if it was **sufficiently close to the target**. This is not usually the case and a **unique calibrator** is required for determining the complex gains.



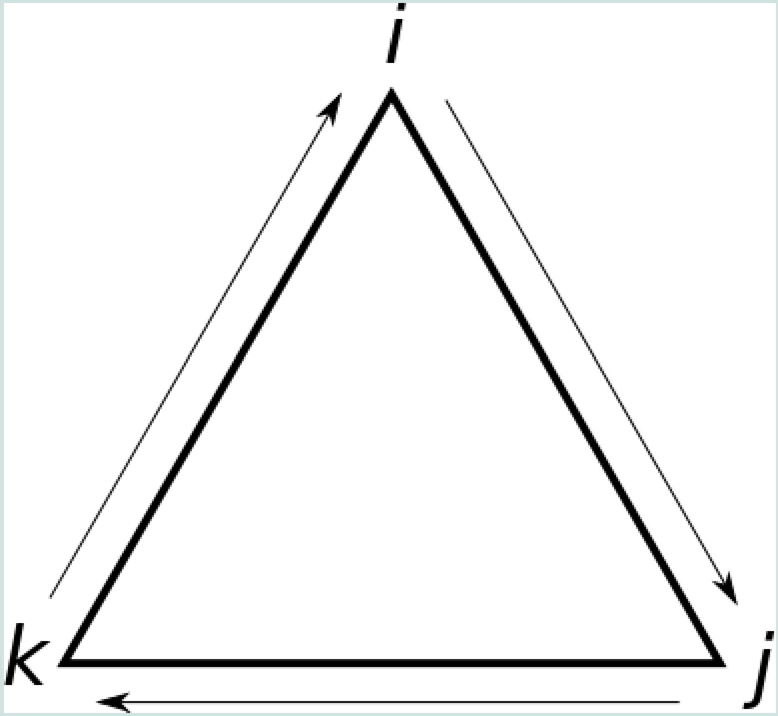
Closure Quantities

Precursor of Calibration

$$\tilde{v}_{pq} = g_p v_{pq} g_q^*$$

$$\begin{aligned}\tilde{v}_{pq} &= A_p e^{-\imath\phi_p} A_{pq} e^{-\imath\phi_{pq}} A_q e^{\imath\phi_q} \\ &= A_p A_{pq} A_q e^{\imath(-\phi_p - \phi_{pq} + \phi_q)}\end{aligned}$$

$$\begin{aligned}\tilde{\phi}_{ij} &= arg(\tilde{v}_{ij}) = -\phi_i - \phi_{ij} + \phi_j \\ \tilde{\phi}_{jk} &= arg(\tilde{v}_{jk}) = -\phi_j - \phi_{jk} + \phi_k \\ \tilde{\phi}_{ki} &= arg(\tilde{v}_{ki}) = -\phi_k - \phi_{ki} + \phi_i\end{aligned}$$



Amplitude Closure: Smith (1952)

$$\frac{|\tilde{v}_{ij}||\tilde{v}_{kl}|}{|\tilde{v}_{ik}||\tilde{v}_{jl}|} = \frac{|A_{ij}||A_{kl}|}{|A_{ik}||A_{jl}|}$$

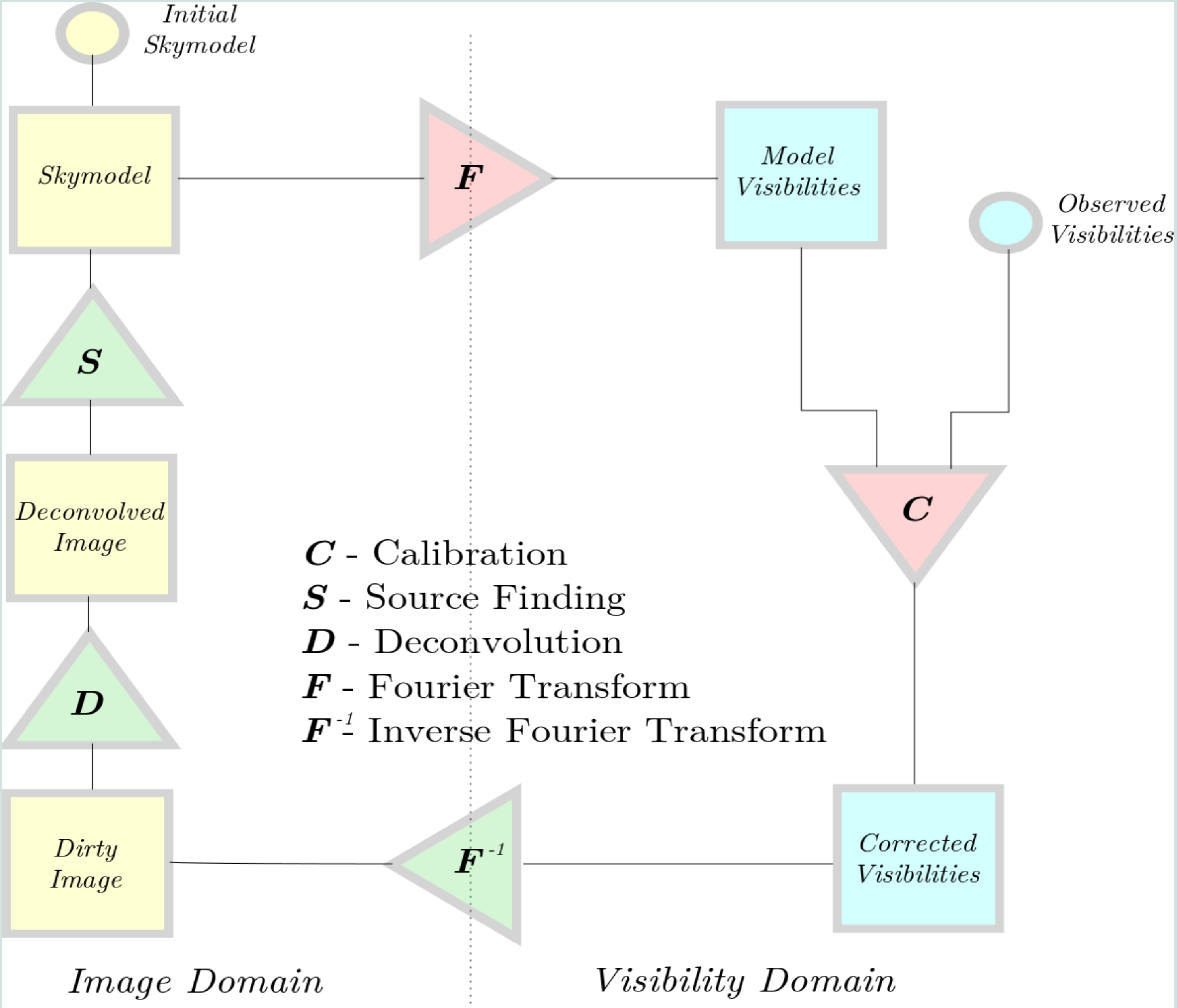
Phase Closure: Jennison (1958)

$$\tilde{\phi}_{ij} + \tilde{\phi}_{jk} + \tilde{\phi}_{ki} = -\phi_{ij} - \phi_{jk} - \phi_{ki}$$

2GC Calibration

- **After** performing **1GC** (applying the antenna gains from the calibrator to the target field) we should be able to make a **decent image** of our **target field**.
- The **dynamic range** of this image can be improved even further by using the **self-calibration framework**.
- **Self-calibration** can be regarded as a variant of the **Gerchberg-Saxton** algorithm.
- Self-calibration makes use of the **observed field** to **calibrate** the visibilities.
- We continuously switch between two domains; the **image domain** and the **visibility domain**.
- In the **image domain** we perform **deconvolution** and **source finding**, while **calibration** takes place in the **visibility domain**.

Self-Calibration Framework

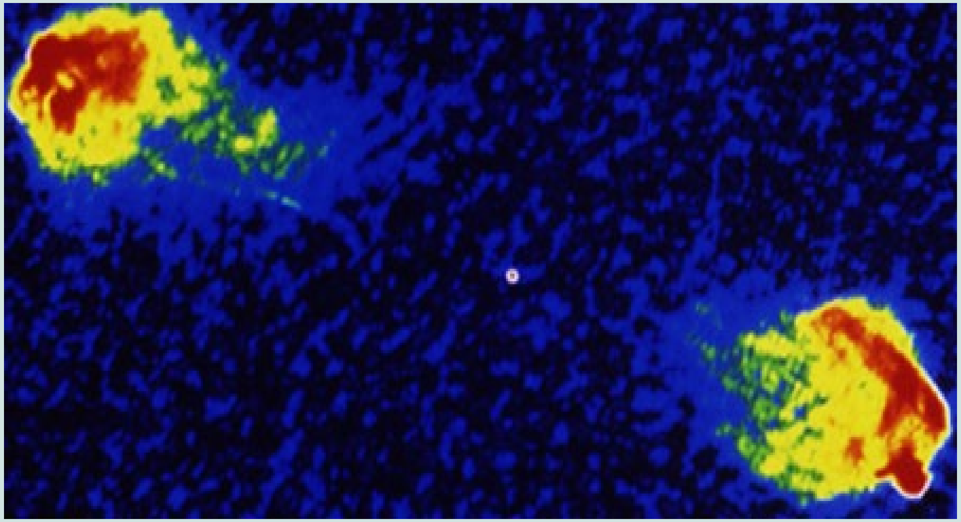


SelfCal Steps

1. We start by creating an incomplete **initial sky-model** of our target field (using a post-1GC image).
2. We use the **initial/improved sky-model** to **calibrate** our observed visibilities which are subsequently imaged.
3. We **deconvolve** the resulting image.
4. We run a **source finder** on the deconvolved image to construct a more **accurate sky model**.
5. We return to **step 2**, or **terminate** the algorithm if the we have reached the **target dynamic range** or if further improvement is not possible.

SelfCal Example: Cygnus A

Before SelfCal



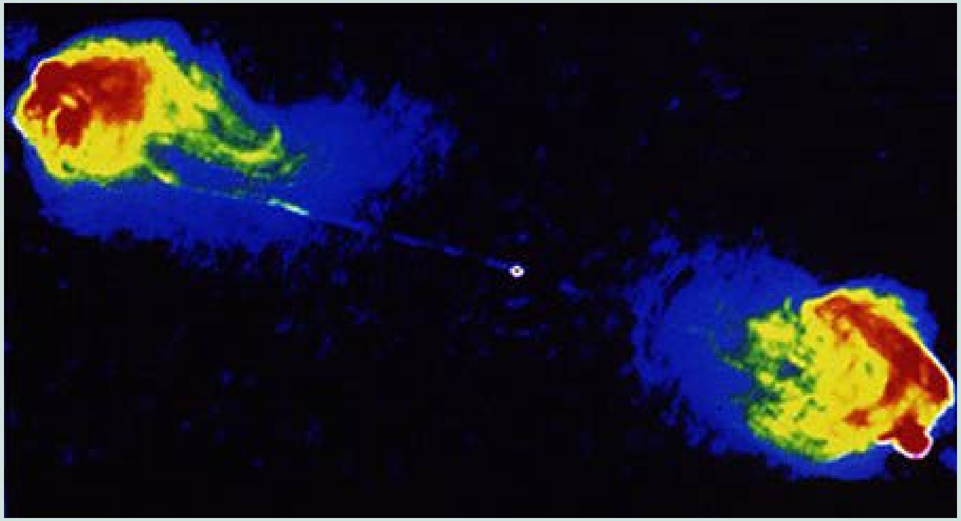
Data for this image of Cygnus A obtained at the VLA, Socorro, NM.

Observers: R.A. Perley, J. W. Dreher

Courtesy National Radio Astronomy Observatory/Associated Universities, Inc.

NRAO/AUI Information Services, Charlottesville, VA 22903; 804-296-0211

After SelfCal



Calibration algorithm?

- Up until this point we have **not described** the actual **calibration algorithm** one needs to **use** in the **self-calibration framework**.
- This is because the algorithm used is **irrelevant** in the context of the framework.
- However, it is useful to note that the **(currently)** most used algorithm is the **least-squares approach** we presented earlier.
- In the **past** self-calibration actually employed **closure quantities**.
- Back when **closure quantities were used** to implement the **calibration sub-block** in the self-calibration framework diagram; self-calibration was known by a **different name**: **hybrid-mapping**.
- Using a **least-squares solver** to calibrate was first proposed in **Cornwill(1981)**.
- Another interesting concomitant of the **least-squares approach** is that it allows us to solve for **individual antenna gains** instead of baseline-based gains.

Hybrid Mapping

- The best known hybrid-mapping approach is discussed in **Readhead and Wilkinson(1978)**:

1. If we have an *N*-**element array**, we obtain *N*-1 **baseline phases** from our **initial/updated model visibilities**.
2. The baseline phases are determined such that the **closure-phases are minimised**.
3. After **imaging** the corrected visibilities, **deconvolution is performed**.
4. We **update our sky-model** based on the deconvolved image.
5. Return to **step 1 or terminate** if convergence has been reached.

3GC Calibration

- The **increased field-of-view** of modern telescopes causes **direction-dependent effects**, such as the **primary beam** and **pointing error**, to become **apparent**.
- Therefore, we **cannot** only **rely** on using **direction-independent self-calibration**.
- There are, in principle, **many approaches** one can use to perform direction-dependent calibration.
- We will highlight a specific approach: **differential gains**.

RIME

WARNING: POLARIZED

$$\mathbf{V}_{pq} = \mathbf{G}_p \mathbf{X}_{pq} \mathbf{G}_q^H \qquad \mathbf{X}_{pq} = \sum_s \mathbf{X}_{spq}$$

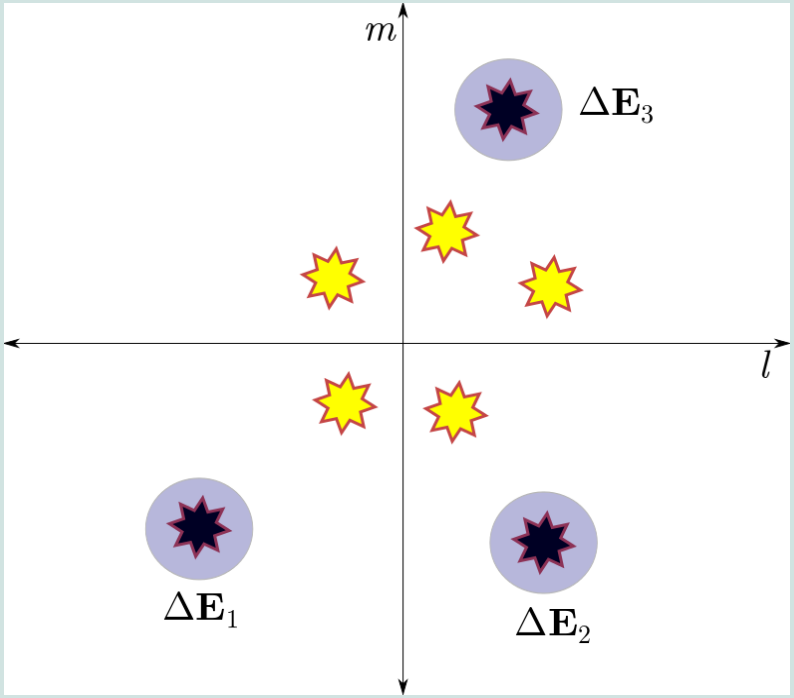
- This equation is known as the **all-sky RIME**, where $\mathbf{V}_{pq}$ is the 2 x 2 **correlation matrix** measured by the interferometer and $\mathbf{X}_{pq}$ is the 2 x 2 **coherency matrix**.
- Moreover, $\mathbf{G}_p$ and $\mathbf{G}_q$ are **G-Jones antenna matrices**. During **calibration we estimate** $\mathbf{G}_p$ and $\mathbf{G}_q$ which we subsequently use to **correct** the correlation matrix $\mathbf{V}_{pq}$.
- The subscripts *p* and *q* denote the **antennas** that were used to make the measurement.
- The **right most equation** implies that in the all-sky RIME we assume that the error that corrupts our visibilities is **independent of the sources' positions**.

Differential Gains

- Example: the **primary beam** of an instrument **varies** significantly over a **large field-of-view** (generally in time and frequency).
- In the **case of the primary beam**, we could try to model the **direction dependent effect** by adding an **a-priori E-Jones** matrix to our Jones' chain.
- However, if we **do not** have **any information** about the **physical source** that is responsible for a **direction dependent effect** then we could use the idea of **differential gains instead**.
- In addition to to the **direction-independent gain** we add a **differential gain** which can be different for each source.
- $\mathbf{\Delta E}_{sp}$ and $\mathbf{\Delta E}_{sq}$ are the **differential gains** associated with source *s* and antenna *p* and *q* respectively. Which can be obtained using least squares.

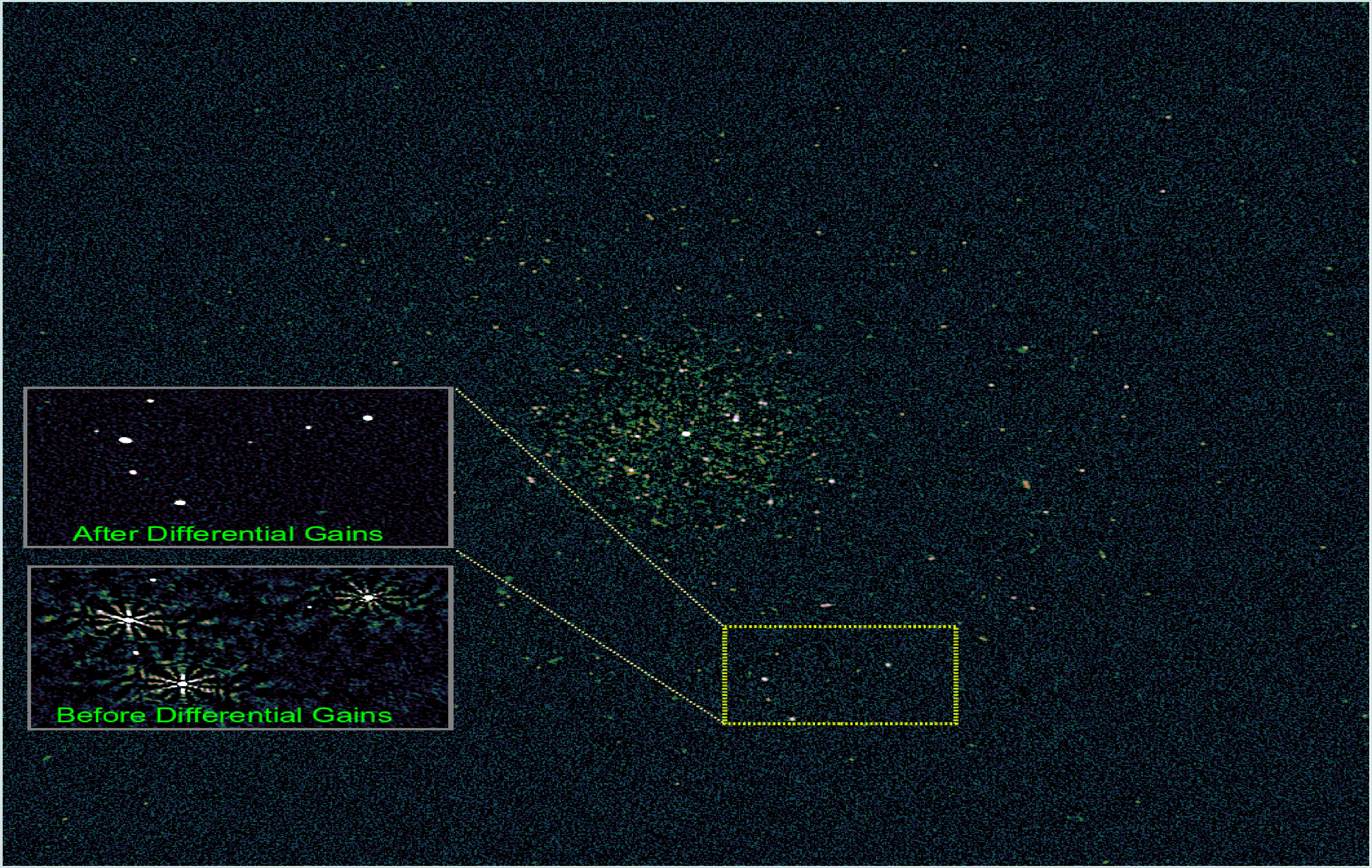
$$\mathbf{V}_{pq} = \mathbf{G}_p \left(\sum_s \Delta\mathbf{E}_{sp} \mathbf{X}_{spq} \Delta\mathbf{E}_{sq}^H \right) \mathbf{G}_q^H$$

Which sources need Δ gains?



- The sources which require a differential gain are usually **surrounded by imaging artefacts** (shown by the purple regions around the black sources in the figure). The yellow sources do not require a differential gain factor (no imaging artefacts around them).
- The **further a source is from the field center**, the more likely it is to be affected by a direction dependent effect.

Δ Gains Example: 3C 147



Physics-based and Heuristic-only

- 3GC can, in general, be divided into **physics-based** and **heuristic-only** approaches.
- If we **know** the **underlying physical phenomenon** which is responsible for a specific direction-dependent effect, we may employ a physics-based calibration approach.
- This is usually accomplished by **constructing a parametrized model** based on the **underlying physical phenomenon**. The aim of this approach is to **estimate the parameters** of this model and use the results to **correct our observed visibilities**.
- In some cases, the **direction-dependent phenomenon** is **known a-priori** and we simply need to **correctly incorporate it** whilst calibrating.
- On the other end of the spectrum we have the **heuristic-only approaches**. In an heuristic approach we **do not know the physical source** of a specific direction-dependent effect. Instead, we **introduce** a number of **free-parameters** which we try to optimize based on some **user-defined heuristic**.

Examples of each

Heuristic-only

- Peeling: **Noordam(2004)**
- Differential Gains: **Smirnov(2011)**

Physics-based

- Pointing-selfcal: **Bhatnagar(2004)**
- Kalman filter: **Tasse(2014)**
- Primary beam: **Mitra(2015)**

Other solvers

- Eigendecomposition: **Boonstra(2003)**
- SAGEcal: **Kazemi(2011)**
- Robust calibration: **Kazemi(2013)**
- StEFCal: **Salvini(2014)**
- Riemann-Manifold: **Yatawatta(2013)**
- Blind Calibration: **Kazemi(2015)**
- Complex Optimization: **Smirnov(2015)**
- Kalman filter: **Tasse(2014)**

Summary

- We reviewed least squares minimization.
- We showed how calibration can be posed as a least squares problem.
- 1GC: Initial calibration with a calibrator.
- 2GC: Use the field which is being observed to calibrate the observation – also known as SelfCal.
- 3GC: Mitigate direction dependent effects.