# Recommendations for the AIPS++ Telescope Model

T.J. Cornwell, NRAO

January 24, 1993

## 1 Introduction

In a recent AIPS++ design note, I described and advocated an approach to the Imaging Model which was based upon linear algebra (Cornwell, 1992). The use of linear algebra concepts leads quite naturally to a simple, clear and concise conceptual definition of Imaging Model, together with the services it provides. Given the apparent attractiveness (at least to me) of this mathematical approach, it is natural to ask whether something similar can be done for calibration. If the answer is no, then we still stand to gain by understanding why not.

Before proceeding, I want to emphasize that I am attempting to develop a *conceptual* framework for calibration, not a computational framework. In the Imaging Model, this meant that although linear algebra was invoked as the concept behind Imaging Model, it would only be in rare simple cases that straightforward linear algebra (such as matrix multiply or matrix inverse) would actually be used for the computations. In general, one would use the short cuts (such as the circulant approximation for convolution) for most computations that we use now. Similarly in this discussion, the emphasis is on concepts rather than computations.

## 2 Summary of the recommended Imaging Model

The logic in the first memo leads to the idea that the Imaging Model be regarded as an class representing a linear equation $AI = D$ where $I$ is the sky brightness, $D$ is the data and $A$ is an operator (usually a matrix) which converts sky to data. The only services required of the Imaging Model are:

**invertable** returns true if the linear equation is invertable,

**invert** returns $A^{-1}I$ for given $I$,

**predict** returns $AI$ for given $I$,

**solve** returns $A^T wD$ for given $D$, optionally normalized by the diagonal elements of $A^T wA$ (if $A^T wA$ is diagonal, the normalized version is the solution to the normal equation),

**residual** returns $(AI - D)^T w(AI - D)$ and $A^T w(AI - D)$ for given $I$ and $D$, optionally normalized by the diagonal elements of $A^T wA$,

**observe** returns $A^T wAI$ for given $I$, optionally normalized by the diagonal elements of $A^T wA$,

**Hdiagonal** returns the diagonal elements of $A^T wA$.

Bob Hjellming pointed out the utility of the first two services for simple well-conditioned imaging problems like single dish processing. In many cases, however, $A$ is singular and so a non-linear algorithm must estimate the sky $I$ using some extra information. In doing so it calls upon the services of the Imaging Model to interact with the data.

One could imagine defining a special Imaging Model such that the matrix $A$ is supplied explicitly in the constructor. These various services would then be realized by actually calling the relevant Math routines. In view of some misunderstanding of the first memo, I should emphasize that *only in the rare simple cases will any of these operations be implemented using simple matrix algebra.* In general, special properties of the matrix $A$ must be used to make the computation feasible. A particular example which should reassure those who are nervous of this abstract formulation is that programs like APCLN will have nearly all the same basic operations but arranged in a slightly different way. There should be no efficiency loss in adopting my proposed definition of Imaging Model.

In current AIPS++ terms, $D$ is a YegSet and $I$ is an Image.

At this point it is worth restating just why we have a need for the Imaging Model. The answer is that we want to decouple some aspects of deconvolution algorithms from the details of the data. In this proposal, deconvolution algorithms interact with data only through the services provided by the Imaging Model. Hence new deconvolution algorithms can be introduced easily without having to explicitly write a version for each type of Telescope. If we accept this goal for deconvolution, the next obvious step is to see whether it can be extended to calibration. This really leads to two quite separate questions: first, can we abstract the calibration of Telescopes into an equation part (analogous to Imaging Model) and a solver part (analogous to the deconvolution algorithm)? and second, can we abstract the calibration of a Telescope into a class which can be called upon in selfcalibration? I address these two questions in turn.

# 3 Splitting Calibration

My discussion of Imaging Model could be applied to any linear system of equations. Is calibration linear? I can think of three levels of difficulty in calibration:

**Linear** Single dish calibration often involves simple ratios of averaged data points and often can reasonably be formulated as linear algebra.

**Mildly non-linear** The ANTSOL approach to interferometer calibration is formally non-linear since it solves an equation like:

$$g_i g_j^* = X_{ij} \tag{1}$$

However, in practice, this can be solved quite easily using simple gradient search techniques since the gains $g_i$ are often roughly the same scale.

**Strongly non-linear** A nice example arises in global fringe fitting in VLBI where strange heuristics are needed to get an initial guess for the least-squares algorithm which is used to solve for antenna-based phases and phase derivatives. While the least-squares algorithm is mildly non-linear, obtaining the initial guess is tough and highly non-linear. Another example from VLBI is the fully general polarization case.

Hence my conclusion is that calibration can range from cases where linear algebra works, to cases where full non-linear optimization is required.

Extending the linear equations used in Imaging, we could say that the observed data $D$ is a function of a set of unknown parameters $P$ and some "correct" data $\widehat{D}$. We then have that:

$$D = C(\widehat{D}, P) \tag{2}$$

where $C$ is now a general non-linear function. Given the parameters $P$, we can predict the data perfectly. The two interesting problems are, first, the inverse problem of deriving $P$ from knowledge of the functional form of $C$, measurements of the data vector $D$ and a prediction of $\widehat{D}$ perhaps from the Imaging Model $\widehat{D} = AI$, and, second, deriving $\widehat{D}$ for real observations.

Note that we could introduce the Imaging Model explicitly by writing, instead, a relation involving the image $I$ and the matrix $A$:

$$D = C(AI, P) \tag{3}$$

However, I think that this is not worthwhile at this point and so for the moment I will continue with writing $\widehat{D}$ for $AI$. I will return to this point later when I discuss algorithms in more detail.

Turning to the issue of how to solve equations of this type, we can define a $\chi^2$ term to be minimized in order to derive $P$:

$$\chi^2 = (C(\hat{D}, P) - D)^T w(C(\hat{D}, P) - D) \tag{4}$$

where $w$ is inverse of the covariance matrix of errors. If the errors are independent between data points, then $w$ is diagonal with elements:

$$w_{i,i} = \frac{1}{\sigma_i^2} \tag{5}$$

We consider an approach to solving for the $P$ parameters based upon the idea of optimization: many iterative algorithms update an estimate of $P$ based upon $\chi^2$ and its gradient with respect to $P$:

$$\frac{\partial \chi^2}{\partial P} = 2 \frac{\partial C(\hat{D}, P)}{\partial P}^T w(C(\hat{D}, P) - D) \tag{6}$$

As before, we can require the services of the Telescope model to calculate this term:

1. Use a service **predict** to find $C(\hat{D}, P)^T$,
2. Subtract $D$ to get $C(\hat{D}, P) - D$,
3. Use a service **solve** to get:

$$\frac{\partial C(\hat{D}, P)}{\partial I}^T w(C(\hat{D}, P) - D) \tag{7}$$

So far, this is all quite obvious but is it helpful? One could imagine feeding this Telescope Model to a non-linear least-squares Solver in the same way that Imaging Model was fed to an *Imager*. I can think of several objections to this scheme.

1. Non-linear optimization is much trickier than linear optimization and so it often pays to use heuristics to help the solution along. Another way of seeing that this must be true is to note that for a general function $C(\hat{D}, P)$ all the derivatives will be important whereas for the linear functions used in the Imaging Model, the terms beyond the second derivative $A^T A$ are irrelevant (actually zero).
2. Non-linear least-squares problems are often feasible only if the initial guess is sufficiently close to the true global minimum (since then derivatives higher than the second can be neglected). Often, however, getting an initial guess is the hardest step. An example in VLBI global fringe fitting was described above.

3. Another difference from the Imaging Model is in the dimensionality of $P$. In the Imaging Model, there may be $10^5 - 10^7$ pixels and so $A$ could have up to $10^{14}$ elements. In calibration, the number of free parameters could be very small (even as small as one for an amplitude scale) and so one could imagine using second derivative information, something that was assumed to be impossible for Imaging and may be vital in some particularly difficult cases of calibration.

The overwhelming conclusion is that the scheme I proposed for the Imaging Model cannot be easily extended to non-linear functions such as that found in calibration. The answer to the first question ("can we abstract the calibration of Telescopes into an equation part and a solver part?") is No.

# 4 Calibration as a service

I now turn to the second question: "can we abstract the calibration of a Telescope into a class which can be called upon in selfcalibration?".

For Imaging we need calibrated data. For this we require a service **apply** of Telescope Model which returns $\widehat{D} = C^{-1}(D, P)$. The instrumental parameters $P$ are internal to a Telescope Model and are estimated using a service **solve** which solves $D_{\text{cal}} = C(\widehat{D}, P)$ where $D_{\text{cal}}$ is the data set observed for calibration. Presumably gain numbers are stored in things like tables inside the Telescope Model, but that does not concern us here.

In summary, the Telescope Model appears to be a non-linear equation $D = C(\widehat{D}, P)$ in the same way that Imaging Model is a linear function $D = AI$. The services of Telescope Model are:

**apply** returns the calibrated data $\widehat{D} = C^{-1}(D, P)$,

**solve** changes the internal state to $P$ found by solving (in some way) $D_{\text{cal}} = C(D, P)$.

The inverse in the **apply** need not be a formal inverse.

# 5 Discussion

Is this a useful abstraction of calibration? Imaging Model is useful principally because we know that we will call different types of Imaging Model from various types of Imagers. Similarly, we will call Telescope Model from self-calibrating Imagers so having a concise description of calibration helps considerably. Perhaps more important however is the conceptualization of calibration as a non-linear machine. Solving for calibration changes the internal state of the machine, and applying calibration invokes the machine on some data.

I argued above that calibration could be split into linear, mildly non-linear and fully non-linear cases. This means that it would be a good idea to develop a facility in AIPS++ for these forms of optimization. It would be best to do this now, ahead of any significant work in implementing calibration.

To summarize the net effect of Imaging Model and Telescope Model, I can now describe the general structure of the deconvolution and self-calibration procedures. For deconvolution alone, algorithms look like this:

1. Select an initial model $I$ for the sky,
2. Use the **residual** service of Imaging Model to find a residual image $A^T w(AI - D)$,
3. Use the residual image in some algorithm-dependent way to update the current estimate of $I$,
4. Return to step 2 unless convergence is obtained.

For deconvolution/self-calibration, algorithms will follow the following general pattern:

1. Select an initial model $I$ for the sky,
2. Use the **predict** service of Imaging Model to predict the data $\widehat{D}$,
3. Use the **solve** service of Telescope Model to find the parameters $P$,
4. Use the **apply** service of Telescope Model to correct the data: $\widehat{D} = C^{-1}(D, P)$,
5. Use the **residual** service of Imaging Model to find a residual image $A^T w(AI - \widehat{D})$,
6. Use the residual image in some algorithm-dependent way to update the current estimate of $I$,
7. Return to step 2 unless convergence is obtained.

Variants on this exist. For example, the initial model may be generated using an approximately calibrated dataset. Furthermore, steps 3 and 4 may involve many cycles of a deconvolution procedure.

In the terminology introduced by Holdaway and Bhatnagar, both deconvolution and deconvolution/self-calibration algorithms are of the class of *Imagers*.

If AIPS++ decides to use this formulation of Telescope Model (which is more or less that selected at Green Bank: Cornwell and Shone, 1992) the next concern is to build the internals for many different types of Telescope Models. The interesting questions then become ones of sharing class designs and concepts amongst different Telescopes. It is this step which has consumed much time

and threatens to lead one into the old mistake of trying to design a very general model which can then be specialized to particular cases. I think that the answer to this temptation is just to design fairly simple Telescope Models specialized to particular Telescopes. Too much agonizing about a putative common calibration scheme will lead to paralysis.

Finally, I would like to discuss how this can be applied to very difficult imaging problems in which the calibration intimately affects the way the imaging is performed. An example of such a problem is non-isoplanatic selfcalibration/deconvolution. In such cases we can assign the Imaging Model calibratable parameters $P$ so that we have the matrix equation:

$$D = A(P)I \tag{8}$$

The Telescope Model is therefore responsible for creating an Imaging Model with estimates for some parameters. In the case of non-isoplanatic imaging, these parameters would be the phases of various parts of the sky at various times. The imaging proceeds thus:

1. Select an initial model $I$ for the sky,
2. Select an initial Imaging Model $A(P)$,
3. Use the **predict** service of Imaging Model $A(P)$ to predict the data $\widehat{D}$,
4. Use the **solve** service of Telescope Model to find the parameters $P$,
5. Generate a new Imaging Model $A(P)$,
6. Use the **apply** service of Telescope Model to correct the data: $\widehat{D} = C^{-1}(D, P)$,
7. Use the **residual** service of Imaging Model $A(P)$ to find a residual image $A(P)^T w(A(P)I - \widehat{D})$,
8. Use the residual image in some algorithm-dependent way to update the current estimate of $I$,
9. Return to step 2 unless convergence is obtained.

In this approach calibration yields two types of parameters, those independent of imaging (e.g. flux scale, overall positions, etc.) and those involved in imaging (phases of patches of the sky as a function of time).

# 6 References

Cornwell, T. J. and Shone, D. L., 1992, ed. "Calibration, Imaging and Datasystems for AIPS++ - Report of the meeting held at Green Bank, West Virginia, 3-14 February 1992", AIPS++ Memo series.

Cornwell, T. J., "Recommendations for the AIPS++ Imaging Model", AIPS++ Implementation Memo 147.