Abstract

# Diagnostic Solvers for Linear Systems with Constraints

Rondall E. Jones, Ph.D.
Sandia National Labs, Retired; Ktech Corporation, Retired
www.rejonesconsulting.com
rejones7@msn.com

**Introduction.** The author's web site offers several software products, including a suite of diagnostic solvers for linear systems of equations including optional constraints. Typically these programs are used for ill-conditioned problems. But even after the best solver has done its job, there are questions: How do the residuals look? Is the model good? Are the equations all meaningful? Might there be accidental data or model errors? How accurate is the solution? Was the decomposition really accurate? Etc. In this presentation we will discuss the types of diagnostics which are automatically produced by each of these solvers as an attempt to help to user understand the most possible about the system that has been solved. While none of these diagnostics are in any sense new, some are rather unusual, and the easy availability of them is worthy of notice. Note that not every diagnostic type is possible for every problem. For example, some diagnostics only apply to over-determined systems.

**Background.** The suite of solvers we will be discussing is based largely on the rejtrix.h library which was discussed at a previous IP conference. See [1] and [2]. Each solver accepts a problem and associated constraints in a simple Excel-compatible *.csv file format (except the commas are optional). There are several choices of solver based on desired weighting and the exact form of regularization. These solvers all have names of the form solve*.exe, where "*" is replaced by some meaningful abbreviation. See [3] for documentation of the current set of solvers, and usage details. This suite is still under development, currently for certain bio-tech applications. All these solvers routinely handle under-determined, square, and over-determined systems; simple singularity; and both general inequality constraints and equality constraints, as supported by rejtrix.h. Some handle arbitrary levels of ill-conditioneding.

**Linear Problems.** We will label an arbitrary user-provided linear system as follows:

| | | |
|---|---|---|
| Ax=b | where A is m by n (ordinary equations) | (1) |
| Ex==f | (equality constraints) | (2) |
| Gx>=h | (Inequality constraints) | (3) |

The number of rows in A, E, and G is entirely up to the user, subject only to certain reasonable limits for software reasons. Each can have zero or more rows. But the number of columns in A, E, and G must of course agree. We also allow "less than" constraints, but these are converted internally to "greater than" form in (3).

**Solution by SVD**. To facilitate the discussion below, we note that if there were no constraints and the system (1) were accurate and not ill-conditioned, then (1) could have been solved using the straightforward SVD (that is, $A=USV^T$) and we would have had

$$x = VS^{-1}U^Tb \qquad (4)$$

It is useful to look at $V^T x$, which equal to $S^{-1}U^T b$. Here, $V^T x$ is the vector of projections of x onto the columns of V, and $P= S^{-1}U^T b$ is the so-called discrete Picard condition vector. P is hoped to be a well behaved, converging sequence of values. But as all IP investigators are well aware, noise, singularity, ill-conditioning, etc, ruin the hoped-for convergence of P, and truncation, regularization, constraints, etc, are added to return P to a better converging shape. It is sometimes useful to examine the final form of $V^T x$ in one way or another.

**Diagnostics.** Now we come to the various diagnostic outputs provided as appropriate by the various solvers, for various input. The first output is an echo of the whole problem. (In some cases this version of the problem will look at little different from the user's input, since all inequality constraints are shown as "greater than" constraints, and a request for a "NONNEG" solution expands to m inequalities.) Then the computed solution is shown.

**Residuals.** Every mathematician knows to check the residual after a system of equations is solved, so that is the next thing the solve* programs output. The residual is simple to show, but we also detect and note any obvious outliers, using heuristics to determine when to note a residual as outstanding. We also show some simple statistics, such as the RMS of the residuals.

**Constraint conflicts.** All the solve* programs are quite robust in the presence of various pathologies, including conflicting constraints. But they will note such difficulties in the diagnostic output. For the equality constraints, (2), we note how many of the constraints were satisfied exactly, and list specifically any which could not be satisfied. For inequality constraints, we classify the constraints as either "satisfied exactly" (usually because the inequality was upgraded to an equality constraint in order to invoke the constraint), or "easily satisfied" (because the constraint never had to invoked) or "not satisfied".

**Solution Error Estimates.** If the user has provided non-zero error estimates for the elements of b, then we next perform a Monte Carlo error analysis of the solution. Typically we do 101 solutions, randomly modifying the elements of b by Gaussian errors with means equal to each given error estimate. The constraints are invoked for all solutions exactly as for the first solution. This results in 101 samples for each x[i]. Currently, because more detail confused users, we output only the upper and lower limits of the middle 95% of samples for each x[i]. We do not use traditional theoretical error estimations because of the difficulty of incorporating the effect of constraints.

**Consistency.** When the system is over-determined we assess the "consistency" of the coefficients in A and b. The idea here is simple: we perform a Total Least Squares solution, which in general results in a change to each element of A and each element of b. Then we look for unusually large instances of changes to an element of A required by the TLS, and list these instances. The b vector is treated similarly. This analysis can be very revealing.

**Correlations.** In the case of square or underdetermined systems (1), the (minimum norm) solution is easily seen to be a linear combination of the rows of A. But that is actually always true. So it is sometimes interesting to users to see how similar the final solution is to each row of A. We output this information in the form of "angles" between the solution and each row of A. (That is, the arc-cosine of

the unitized dot product of each row of A with x.) Secondly, the right hand side, b, should be a linear combination of the columns of A (in the absence of error) so we output the "angle" between b and each individual column of A. Thirdly, we depict the projections of the solution x onto the columns of V by showing the "angles" between x and the columns of V.

**Details of pseudoinverse solution of (1).** It is sometimes instructive to look at the details of the build-up of the pseudo-inverse solution to the ordinary equations. (that is, (4) with zero singular values correctly handled.) So we output these details: the singular values; two assessments of the condition number; the vector $U^T b$; the vector $P= S^+U^T b$; the pseudo-inverse solution; the vector $V^T x$ (not as angles this time), and three measures of the numerical quality of the SVD of A.

**Linear dependency.** It is useful to know of any cases of the presence of linear dependencies among any small set of rows of A. This can be accomplished by examining the columns of U looking for cases of columns with only a few non-zero coefficients. An analysis of these cases is output.

**Relative Importance of equations.** In some contexts, the "covariance matrix" is considered of great significance. In this context, covariance matrix means $C = (A^T A)^{-1}$. The diagonal elements of C are sometimes taken as (relative) estimates of the variance of the solution values. This approach ignores important aspects of the problem, but can give meaningful information about the structure of A. For over-determined systems, we successively remove each row from A and recompute the covariance matrix. A substantial rise in the sum of the diagonal values signals that the removed equation is important to the solution. No change at all means the equation may be irrelevant or redundant.

**Other.** If the solution has been regularized, whether manually or automatically, we report the assessed rank of the system (1), and the value of the Tikhonov regularization parameter, lambda.

**Note:** Much of the algorithmic code that supports the solve* solvers is provided in the rejtrix.h library. But the code which supports the input and output to Excel compatible files and some of the diagnostics is not currently considered to be appropriate to release generally, due to need for extensive documentation and other considerations. It is available for private release as needed. We hope that attendees may have other interesting ideas for diagnostics which we might add to these solvers.

[1] The earlier paper is available at http://www.rejonesconsulting.com/autoregIPES_2006.pdf.

[2] The rejtrix.h library and documentation are available at http://www.rejonesconsulting.com/indexlibrary.htm.

[3] The current suite of Excel-format solvers is presented at http://www.rejonesconsulting.com/indexdensealgorithm.htm

[4] *Matrix Computations*, by Gene H. Golub and Charles Van Loan, Johns Hopkins Press.

[5] For an annotated example (not comprehensive) of a diagnostic output file, see http://www.rejonesconsulting.com/ExplainOutput.xls