Software Section

RANDMOD: A System for Randomizing Modifications to an Instance of a Linear Program

HARVEY J. GREENBERG  Mathematics Department, Campus Box 170, University of Colorado at Denver, P.O. Box 173364, Denver, CO 80217-3364. BITNET: hgreenberg@cudaer

(Received: August 1990; accepted: October 1990)

RANDMOD is an interactive system to perform controlled randomization on a linear programming model. Its purpose is to enable statistical analysis of algorithm performance or model robustness while preserving structural realism of the underlying model. The basic operations are augmentation, perturbation, scaling and sign reversal. In addition, RANDMOD provides interactive query, session control and interfaces with other systems.

One of the difficulties with empirical testing in linear programming is how to design meaningful experiments. One purpose of experiments in linear programming is to measure performance of an algorithm, or comparisons among algorithms, in order to evaluate their performance. A common approach today is with a library of test problems, like NETLIB.[2, 7] This has limitations, however, in giving representative results that can be extrapolated over populations of real, unsolved linear programs.

A more scientific approach is statistical, but structural realism can be lost without some control over the randomization. The purpose of RANDMOD is to enable statistical experimentation while preserving much of the structural realism of a family of linear programs (see Greenberg[3] for some elaboration). One such study was given by O'Neill.[8]

1. Computing Environment

RANDMOD is written in FORTRAN/77 and has been tested under DOS 3.3, VAX/VMS, IBM/CMS, IBM/TSO, PRIME/9950, DEC/XENIX and SUN4/UNIX. In principle, it can be compiled to run in any computing environment. The default installation for a microcomputer under DOS uses about 435K memory and 0.66 mb hard disk space.

RANDMOD is available as a collection of DOS files, so it must first be loaded under DOS (3.0 or higher) before it can be uploaded to another environment. Installation on other computers requires only compilation and linking, but also preparation of the source code. This has been automated with a DOS batch file. The various installation procedures and how to change dimensions, if desired, are described in the RANDMOD Primer.[4]

2. Basic Operations

There are four basic operations performed by RANDMOD: augmentation, perturbation, scaling and reversals. In addition, RANDMOD has basic query, input/output, and session control commands. Here we elaborate only on the basic operations and how randomization is performed.

Once a linear program is loaded into memory, any submatrix can be constructed to control the portions of the linear program that are affected. Different submatrices can be constructed during the same session, allowing many operations to be applied before the new linear program is generated for exportation to an optimizer. For example, if the base matrix is a regional model, each submatrix can correspond to a region's activities and associated constraints, like capacity limits; or, the submatrix can be composed of inter-regional links and associated balance equations.

The AUGMENT command is used to augment rows to columns with randomly generated coefficients and limits derived from aggregating rows of a submatrix. The types of augmentations are degenerate, redundant and infeasible. A degenerate equation is generated by aggregating (submatrix) equations to form a weakly redundant constraint. A (strongly) redundant constraint is generated by a similar aggregation, followed by a shift in the right-hand side value. An infeasible constraint is generated similar to a redundant one, except the inequality is reversed, and the right-hand side shift is away from the feasibility region.

Augmentations can test algorithm correctness and its sensitivity to the types of rows augmented. For analysis

Subject classifications: Programming: linear: large scale systems. Computers/computer science: software; system design operation.
Other key words: Computational testing.
support, augmentations test cognitive elements, such as the quality of a diagnosis of infeasibility.

The PERTURB command perturbs the bounds of rows or columns, or it perturbs distinct coefficient values. The value perturbation applies to distinct magnitudes only, keeping the sign pattern of the body of the matrix unchanged. It does not, however, change unity elements \((\pm 1\)'s\), because they are generally not dependent on data; rather, they are generally part of the model’s logic, as in the case of a pure network.

Perturbations and augmentations cannot be made together, because the perturbations are likely to corrupt the meaning of any augmentation. For example, a redundant constraint can become non-redundant once coefficient values are perturbed. In general, perturbation is used to measure robustness of an algorithm or of the model underlying the base matrix. Coefficient values, for example, can become widely dispersed, and the polyhedral structure could change with facet rotations and translations caused by the perturbations.

The SCALE command defines a multiplier for each row or column, called its scale, which is applied when the new matrix is generated. Each scale is a positive value, which, unlike perturbation, preserves the solution structure, at least in theory. It can be used as another test of an algorithm’s robustness.

The REVERSE command is used to reverse signs of nonzeros in the submatrix. This allows tests of diagnosing modeling errors, such as infeasibilities created by the inadvertent reversals of arcs in a network.

Each operation is controlled by randomization parameters, which govern the number of operations and their extent. For example, the number of augmentations, or the percent of elements perturbed, are determined by independent (pseudo-) random variables, whose distributions are controlled by the RANDOMIZ command. Weights used in augmentation and ranges used in perturbation are controlled by other, independent random variables.

All operations are logged and can be reported with a file output that could be used for statistical collection. In addition, algorithm controls that are specified in a file, such as the spec files of MINOS and OB1, can also be set with the .KEYWORD command. In this way, one can collect statistics about both the linear program and the algorithm settings.

3. Interfaces

RANDMOD files are compatible with most optimizers and have been tested with MINOS and OB1. A base matrix can be obtained either from a library of test problems, or from modeling languages that generate standard (MPS) matrix files, such as MODLER.\[5\]

In addition, any submatrix of a linear program coefficient matrix can be written for deeper analysis. One interface is with Alvarado’s Sparse Matrix Manipulation System (SMMS).\[1\] This is a DOS system that provides (among other things) graphic visualization of matrices. One example of its use is the exportation, from RANDMOD, of the active constraint rows and basic structural variables for agenda analysis. This is done with the SUBMAT WRITE command by specifying the SMMS format option.

4. Structures

RANDMOD is highly modularized. Figure 1 illustrates its 4 primary modules: (1) User Interface (FLIP); (2) Matrix/Solution Data Access (GETMAT); (3) Procedures that execute the commands; and, (4) Query, which is taken from ANALYZE\[5\] (a computer-assisted analysis system for linear programming models and solutions). These modules can be used for other applications. Each contains some document files, and the source listings contain necessary details.

Another view of the RANDMOD structure is functional. Figure 2 illustrates a typical loop. A base matrix is

![Figure 1. Calling sequence of RANDMOD modules.](image)

![Figure 2. Job flow for experimental analysis.](image)
obtained either from a library, like NETLIB,\textsuperscript{[2]} or from a modeling language, like MODLER.\textsuperscript{[6]} Then, a solver is applied, like OB1, which generates a solution file plus a log file that are used for collecting statistics. The matrix file re-enters RANDMOD for additional randomization, such as more perturbations of nonzero coefficients (preserving signs), and the process repeats. After sufficient statistics are collected, the results can be entered into a statistical package for analysis (not part of RANDMOD).

5. Availability

RANDMOD is available for nonprofit research and education. A primer plus disk is available from the author at nominal cost.

ACKNOWLEDGMENTS

This research was partially supported by a consortium of industries: Amoco Oil Company, General Research Corporation, Shell Research, Ketron Management Science, MathPro, Inc., and US West. Partial support was also provided by the Office of Naval Research, Contract N-00014-88-K-0104.

REFERENCES

4. H.J. Greenberg, 1990. \textit{A Primer for RANDMOD}, University of Colorado at Denver, Mathematics Department, Denver, CO.
5. H.J. Greenberg, 1990. \textit{A Primer for ANALYZE}, University of Colorado at Denver, Mathematics Department, Denver, CO.
6. H.J. Greenberg, 1990. \textit{A Primer for MODLER}, University of Colorado at Denver, Mathematics Department, Denver, CO.