CIPDSS-PST: CIPDSS Portfolio Selection Tool Documentation and User's Guide

Decision and Information Sciences Division

CIPDSS-PST: CIPDSS Portfolio Selection Tool Documentation and User's Guide

by
J.C. VanKuiken, M.J. Jusko, and M.E. Samsa
Decision and Information Sciences Division, Argonne National Laboratory

for
U.S. Department of Homeland Security
Science and Technology Directorate

May 2008

CONTENTS

CONTENTS (Cont.)

FIGURES

FIGURES (Cont.)

TABLES

ACKNOWLEDGMENTS

CIPDSS-PST: CIPDSS PORTFOLIO SELECTION TOOL DOCUMENTATION AND USER'S GUIDE

by

J.C. VanKuiken, M.J. Jusko, and M.E. Samsa

ABSTRACT

The Critical Infrastructure Protection Decision Support System – Portfolio Support Tool (CIPDSS-PST) provides a versatile and powerful tool for selecting, optimizing, and analyzing portfolios. The software introduces a compact interface that facilitates problem definition, constraint specification, and portfolio analysis. The tool also provides a simple screen design for comparing user-preferred choices with optimized selections. CIPDSS-PST uses a portable, efficient, mixed-integer optimization engine (lp_solve) to derive the optimal mix of projects that satisfies the constraints and maximizes the total portfolio utility. The CIPDSS-PST software can be readily applied to other nonportfolio, resource-constrained optimization problems.

SUMMARY

The Critical Infrastructure Protection Decision Support System – Portfolio Support Tool (CIPDSS-PST) is a powerful tool for selecting, optimizing, and analyzing portfolios. The software provides options for making user-preferred selections and comparing them with optimized project selections. A wide range of constraints can be introduced to accommodate various resource limitations or upper and lower limits on the numbers of projects. Conditional constraints can also be accommodated, whereby inclusion of one or more projects can be forced to depend on selections of other projects. The optimized solution will then reflect these additional constraints in the sets of projects selected for the optimal portfolio.

The optimization engine lp_solve (Linderoth and Ralphs 2005; Notebaert et al. 2008) was selected on the basis of a number of criteria that included portability, cost, licensing considerations, and problem size capability. It performed very well in all test cases and continues to be maintained, updated, and verified by a large user community.

CIPDSS-PST can be applied to other nonportfolio resource allocation problems. The framework is very flexible, allowing the user to define the problem objectives and scales for rating each of the alternatives to be considered.

Significant attention was devoted to simplifying the problem construction process and streamlining the data input interfaces. This effort resulted in a single spreadsheet-like form for entering all the information related to defining the critical problem (objective definitions and project evaluations). Likewise, a single screen covers all the data entry needs for defining

constraints and viewing outcomes. The end result is a convenient and compact interface that facilitates problem construction and analysis.

1 INTRODUCTION

The Critical Infrastructure Protection Decision Support System – Portfolio Selection Tool (CIPDSS-PST) provides a structured environment to help users select specific projects or tasks from a choice of candidates that may be competing for limited resources. The overall goal is for the software to identify optimal project selections in the context of user-defined objectives and constraints. Additional functions of the tool yield insights into alternative, nearly optimal sets of projects and the incremental benefits associated with constraint relaxations.

The core functionality of CIPDSS-PST was patterned after the Portfolio Analysis Support System (PASS), an earlier decision support tool (Jusko et al. 2006). CIPDSS-PST extended the decision-analytic logic of the PASS precursor by adding new operational and display features that not only added to its analytical capabilities but also made it easier to use and enhanced the interpretive power of the results.

A primary goal of the CIPDSS-PST software tool is to maximize the value of selected projects (or tasks) on the basis of a set of user-defined objectives and constraints. The objectives are defined to capture various aspects of project performance or impacts that affect the overall perceived value of each project. Some examples of objectives are those that specify a project's completion time, scope of benefits, technical difficulty, or any other criterion that is measurable and of value to the decision maker. Each of the objectives is characterized by the user in terms of its relative importance.

After the objectives are established, each project or task is evaluated in terms of how well it achieves each objective. From the specific project objective "scores," an overall project value is calculated. CIPDSS-PST determines a priority ranking of all projects on the basis of how well each one has achieved its objectives. If no other considerations were active, this priority ranking would represent the natural order for selecting projects. However, usually there are other constraints associated with a portfolio of projects, which, if accounted for, would alter the selection of projects. For example, budgetary constraints are common, and other objectives (e.g., spreading project investments over different categories of activities like research and development [R&D], new ventures, or overhead) might represent additional competing factors to be considered in selecting the optimal projects.

CIPDSS-PST recognizes simple constraints, such as maximum total funding limitations or minimum funding for specific projects. It also treats more complex constraints, such as project prerequisites or mutual exclusions. Considerations for these complicating factors can significantly affect the selection of optimal projects. CIPDSS-PST provides a tool that explicitly recognizes such constraints and captures their effects on the selection of projects for an optimal portfolio.

Although CIPDSS-PST was designed with a focus on project portfolio evaluations, it can be used to analyze virtually any set of alternatives that compete for limited resources, whether monetary, human, material, or others.

This report serves as a user-guide and describes the fundamental decision-analytic problem representation. It is intended to describe how (1) a problem is formulated, (2) project values are derived, (3) the optimization engine is invoked, and (4) the user can apply visualization tools to analyze and probe the results for sensitivities.

Section 2 provides an overview of basic steps used in CIPDSS-PST to obtain an optimal portfolio. Sections 3 and 4 contain examples to illustrate the use of CIPDSS-PST and derivations to show how the fundamental project values are obtained. Section 5 documents the optimization engine selection process; the criteria that were considered, the engines that were examined, and the final choice that was made. Section 6 summarizes the report.

2 APPROACH

2.1 OVERALL DESCRIPTION

CIPDSS-PST treats the primary goal of portfolio selection as a mathematical maximization problem — as a problem of how to maximize the achievement of a set of objectives. Each project contributes to the achievement of the objectives to a different degree, and assessing how well a project achieves the objectives determines that project's relative value or importance. CIPDSS-PST provides a structured framework for (1) identifying and characterizing the objectives, (2) identifying candidate projects and assessing how effectively each one achieves the objectives, (3) characterizing resource and other types of limitations, and (4) selecting from the list of candidate projects the optimal set of projects that would maximize the overall achievement of the objectives.

If there were no resource or other limits in effect, then simple priority rankings and assessments for each project would result in a natural order for selecting projects. However, there are usually other constraints associated with a portfolio of projects that, if satisfied, would alter the optimal selection of projects. Budgetary constraints are common, but other considerations (e.g., spreading resource investments over several critical programmatic areas) are also typical.

CIPDSS-PST provides the user with convenient mechanisms for implementing constraints and optimizing a portfolio when virtually any number of constraints are present. When constraints are included in the analysis, CIPDSS-PST identifies the set of projects that would maximize the portfolio's value in terms of meeting the objectives while satisfying the constraint requirements. If a problem becomes over-constrained (i.e., meaning there is no valid solution that would satisfy all of the constraints), a warning message is displayed so the user knows that adjustments are needed.

2.2 OVERVIEW OF STEPS IN THE PORTFOLIO ANALYSIS PROCESS

There are seven steps in the portfolio optimization process. CIPDSS-PST uses spreadsheet-like entry forms that are designed to help the user complete these steps. Constraints can be simple or complex, and multiple sets of constraints can be implemented. Unconstrained cases can also be evaluated. An overview of the steps follows:

1. Establish the objectives;

2. Assign relative weights to each objective;

3. Set scale ranges for rating each project according to the objectives;

4. Identify, cost, and categorize the projects;

5. Rate projects according to the objectives;

6. Define portfolio constraints; and

7. Optimize portfolio selections and perform sensitivity analyses.

These steps are briefly described below and are presented with examples in Sections 3 and 4 of this report.

2.2.1 Establishing Objectives

This first step is perhaps the most critical one in the CIPDSS-PST process. Three sources — Keeney (1992), Keeney and Raiffa (1993), and Hammond et al. (1998) — describe how the objectives and the scales to measure their achievement are established. Some examples of objectives are the project cost, scope of benefits, degree of risk, and completion time. For those unfamiliar with establishing objectives and scales, the three sources serve as useful tutorials for setting up a new problem. If the objectives are not established properly, then the portfolio analysis that follows may be misleading. Carefully setting the objectives will simplify and clarify the portfolio analysis process.

2.2.2 Establishing the Relative Importance (Weights) of the Objectives

Objectives typically vary in terms of their relative importance or value. The user must establish the relative importance of each of the objectives defined in Step 1. In CIPDSS-PST, the user selects a "Relative Weight of Objective" between 0 and 100 to represent the importance of each objective relative to other defined objectives. The least desirable value is 0, and the most desirable value is 100. Objective weights between 0 and 100 have intermediate values that are proportional to the value of the objective (i.e., an objective valued at 50 points is worth half of an objective valued at 100 points in terms of relative importance). Each objective should be assigned a value greater than zero, unless the user is conducting a sensitivity analysis to determine the effect that a given objective has on the portfolio choices. Only the *relative* values are important. This means that if all objectives are assigned 50 points, then all objectives are treated as being of equal value. CIPDSS-PST performs the necessary value normalizations after the user has specified relative weights.

2.1.3 Setting Scale Ranges for Rating Objectives

An objective can be satisfied by various projects at different levels, and the level of achievement has an associated importance or value to the user. CIPDSS-PST offers two general types of scale ranges for rating how projects satisfy objectives: continuous or discrete. Within the two categories of scale ranges, the program provides eight types of continuous distributions (e.g., triangular, normal, and lognormal are three of them) and an unlimited number of user-defined discrete rating scales. After determining the preferred choices of scale range types, the user can specify the parameters that define best and worst values in the scale ranges, as well as points between the best and worst outcomes.

Figure 2.1 shows a preview of the primary input screen, which is used to define the objectives as described above. The objectives are defined in the area between the "Objectives" and "Projects" headings (headings are highlighted in yellow). Green and blue boxes represent user inputs to define the portfolio selection problem. User inputs for objectives include names, descriptions, scale type, scale units (for continuous scales), scale range descriptors, relative value of scale range, and relative weight of objective. More comprehensive descriptions of these input parameters are provided in Section 3.

2.2.4 Identifying, Costing, and Categorizing Projects

The same spreadsheet-like form used for entering objective information (Figure 2.1) is also used for entering information about the candidate projects that are to be considered for possible inclusion in the optimal portfolio. The user enters abbreviated project titles (e.g., P1, P2, P3, …) and can also include longer descriptions of the projects. The longer titles are optional, but the short identifying descriptors are necessary.

FIGURE 2.1 Preview of Basic CIPDSS-PST User Input Screen (Objective and Project Definitions)

The CIPDSS-PST data entry fields also include an opportunity to provide point estimates for the costs of each project (next-to-last column on right side of the screen). These cost data are optional but can be used to generate useful rank-ordering metrics for projects (according to utility-per-unit-cost rankings). In addition to these point estimates for costs, ranges of uncertainty for the cost estimates can also be explicitly captured through user-defined cost objectives. These can be characterized by probability distributions or discrete ranges.

CIPDSS-PST provides a mechanism for categorizing projects according to user-defined categories (e.g., as construction versus R&D versus overhead, or as military versus civilian, or by corporate division). A simple matrix with inclusion/exclusion buttons allows the user to define which projects belong to which categories. These categorizations provide powerful tools for implementing constraints (e.g., for specifying that at least 20% of the budget must be allocated to projects that qualify as R&D activities). Examples are given in Section 4 to illustrate how these categorizations and constraints can be used to represent relatively complex conditional inclusion requirements. An example matrix for project categorizations is shown in the upper right portion of Figure 2.2.

FIGURE 2.2 Preview of Sample CIPDSS-PST Constraints & Analysis Screen

2.2.5 Rating Projects According to Objective Scales

Figure 2.1 illustrates the CIPDSS-PST screen that is used to rate or score each project in terms of how it is expected to fulfill each of the objectives. These ratings are entered next to the name and description information. For objectives with continuous-scale ranges, the user specifies distribution parameters (e.g., for triangular distributions: minimum, maximum, and mode parameters would be specified). For objectives with discrete-scale ranges, the user estimates the relative likelihood of the project falling in each of the scale ranges. To simplify the entry process, the ratings (0–100) are interpreted as relative, and the program performs the necessary normalizations to complete the calculations.

2.2.6 Defining Portfolio Constraints

Portfolio constraints can be easily entered in the screen shown in Figure 2.2. For the total portfolio, constraints can be implemented as upper or lower bounds on costs or total numbers of projects to be selected. In addition, for each of the user-defined project categorizations (e.g., construction, R&D, or overhead), the user can specify upper and lower bounds for the total number or cost of projects selected from a given category. The constraints and categorization tools can be used creatively to represent conditional selection criteria, such as project prerequisites (e.g., the portfolio *must* include Project A *if* Project B is to be included). Section 4 provides examples of conditional constraint implementations.

2.2.7 Choosing Portfolio Selections and Performing Sensitivity Studies

At this stage (after Steps 1 through 6 are completed), CIPDSS-PST has all the information it needs to run the portfolio analysis and provide the user with results. The user can choose to manually select projects for the portfolio or use the optimization button " ★ " to invoke the integrated optimization engine. If the "optimize" option is invoked, the optimized portfolio is displayed side-by-side with the user's preferred selections, revealing any improvements that would result from the optimization and allowing for an easy comparison of project selections.

Sections 3 and 4 contain examples to illustrate the use of CIPDSS-PST. The internal derivations of project values are also documented, so the user can better understand how the input parameters affect the outcomes. In addition, for the screen depicted in Figure 2.1, the user can activate the rows expand button " ↕☰ " to reveal intermediate results in the value calculations. The intermediate results can be very helpful to the user in understanding relative project rankings and in verifying user inputs, intuition, and insights with regard to portfolio dynamics.

The opening screen for CIPDSS-PST (Figure 2.3) provides a quick-start guide. The user preferences button " 🖼 " provides several options for customizing how the CIPDSS-PST display appears and how the screens respond to scrolling.
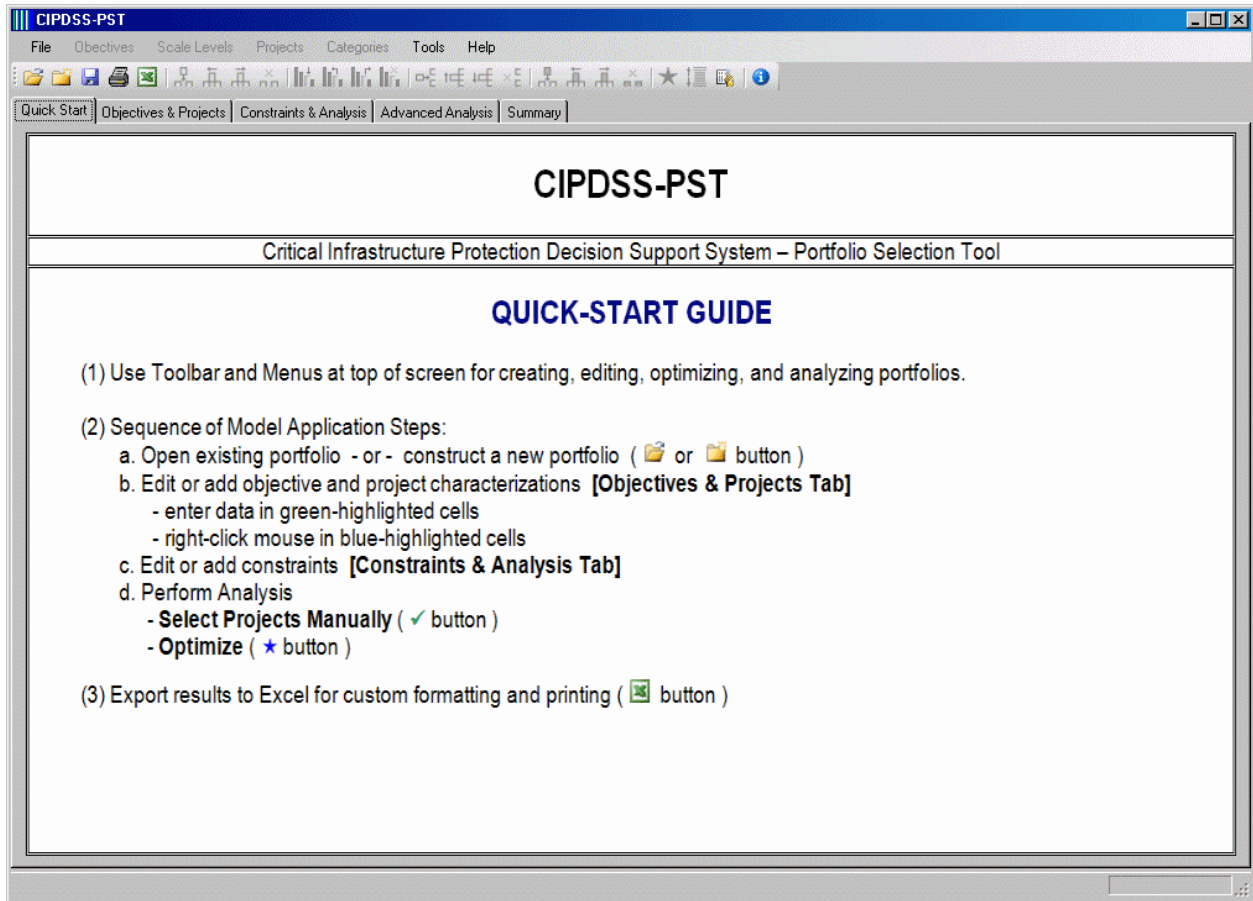
FIGURE 2.3 Opening Screen for CIPDSS-PST

3 USER ENTRIES AND VALUE/UTILITY DERIVATIONS

This section describes the data entries and calculations that ultimately drive the overall portfolio optimization process. CIPDSS-PST provides a basic framework for defining and weighting the relative values of objectives. Then each project is evaluated in the context of those objectives to determine relative project values and utilities. In this context, "utility" represents a single measure of merit that combines all of the estimated outcomes and relative importance factors as assessed by the user. The CIPDSS-PST framework is designed to be flexible to accommodate a comprehensive range of objectives and project characterization metrics.

As indicated in Section 2, the preparation and analysis steps consist of the following:

1. Establish the objectives;

2. Assign relative weights to each objective;

3. Set scale ranges for rating the projects according to the objectives;

4. Identify, cost, and categorize the projects;

5. Rate projects according to the objectives;

6. Define portfolio constraints; and

7. Optimize portfolio selections and perform sensitivity analyses.

The procedures and assumptions for quantifying inputs are described below, along with an example problem that shows sample inputs for solving the problem with CIPDSS-PST.

3.1 NOTATION FOR THE SAMPLE PROBLEM

Mathematical notation is introduced in this report for the reader's convenience in tracking how the data entries are used in value/utility calculations. This notation is not displayed on the CIPDSS-PST screens, but it is possible for the user to view intermediate results of the calculations by pressing the rows expand button " ▤ " when viewing the screen shown in Figure 2.1. Section 3.3 describes the detailed internal CIPDSS-PST calculations.

3.2 OBJECTIVES & PROJECTS TAB

The user can start by using the open folder button " 📂 " to open an existing sample portfolio case. Selecting a predefined case, such as "sample-case-1.xml," can help the reader follow the instructions for constructing a new problem, as described in this section. The sample case can serve as a template that can be edited and customized to fit any new problem.

For those unfamiliar with establishing objectives and scales, Keeney (1992), Keeney and Raiffa (1993), and Hammond et al. (1998) serve as useful tutorials for setting up a new problem. The following example illustrates the mechanics of representing the objectives in CIPDSS-PST.

3.2.1 Entering Objective Names and Descriptions

This example uses project completion time, scope of benefits, and technical difficulty as the three objectives to be optimized. (Some other examples of objectives are costs, technical or financial uncertainties, expansion of production capability, uniqueness, or likelihood of attracting funding from outside sources.) It is important to recognize that costs can be entered and treated in two ways in CIPDSS-PST. One way is to include costs explicitly as an objective and trade them off against other objectives. This approach is recommended for problems in which cost is a valid consideration. In addition, to facilitate the construction of simple project rank-order listings in CIPDSS-PST screens, the expected costs for each project can be entered in the "cost" data field (shown later in this section). Even if cost estimates are entered explicitly as an objective, with the associated option of using probability distributions to capture potential uncertainties, the user is encouraged to also specify *expected* costs as an estimate in the cost data field.

Characterizing costs through objective definitions allows costs to be treated explicitly in the optimization process. In contrast, user entries in the simple cost data field are not recognized in the optimization process. They are only used in summary screens for reference and sorting purposes. As a guideline for choosing whether or not to create a separate objective to treat costs, the user should consider cases where two projects are otherwise equally valued according to the objectives. If the preferred portfolio choices favor the lower-cost project over the higher-cost project, the user should explicitly include costs in the objectives (in addition to entering the expected cost data). And if the preference for the lower-cost option is only slight, the relative weight for costs should be small in the objective function.

Figure 3.1 shows the objective names under the "Objectives" heading. The green background indicates which of the table fields are to be filled in by the user (either alphanumeric or numeric entries). Fields that have a blue background indicate that the user needs to make a multiple-choice selection. In addition, the screen provides a field for more lengthy descriptions of each objective. This extended description can be very useful for documenting the interpretation of objective definitions that are not obvious.

3.2.2 Defining Objective Scales: Scale Types and Units of Measure

For any objective, the user has a choice of using a continuous scale of measurement or a discrete "constructed" scale. The choice appears when the user right-clicks on the blue-highlighted scale-type field (Figure 3.2).
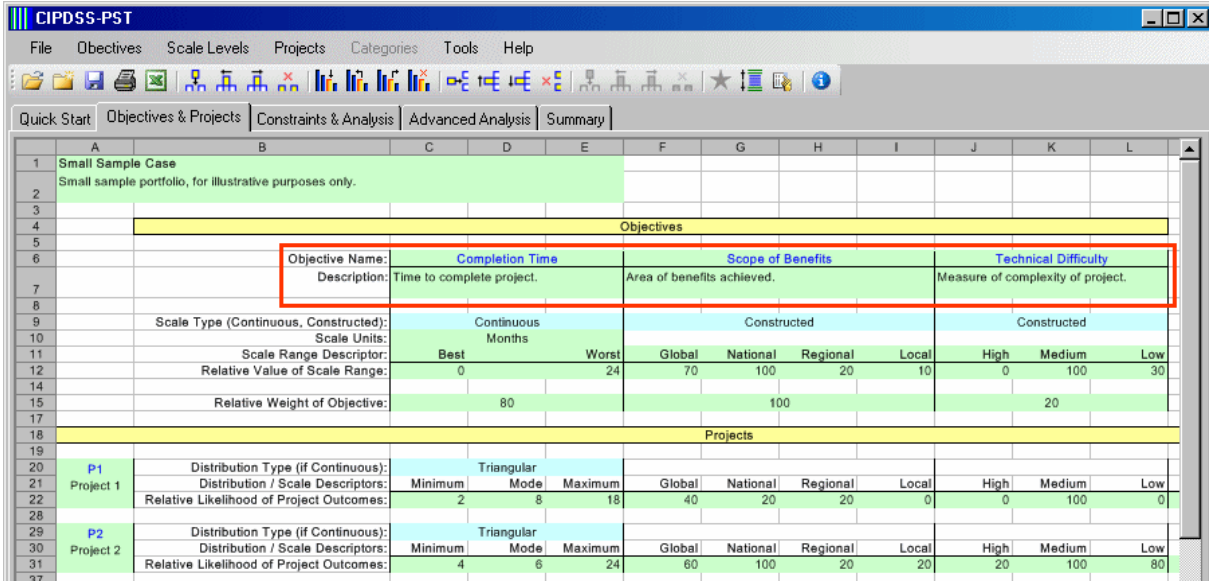
FIGURE 3.1 Objective Name and Description Fields
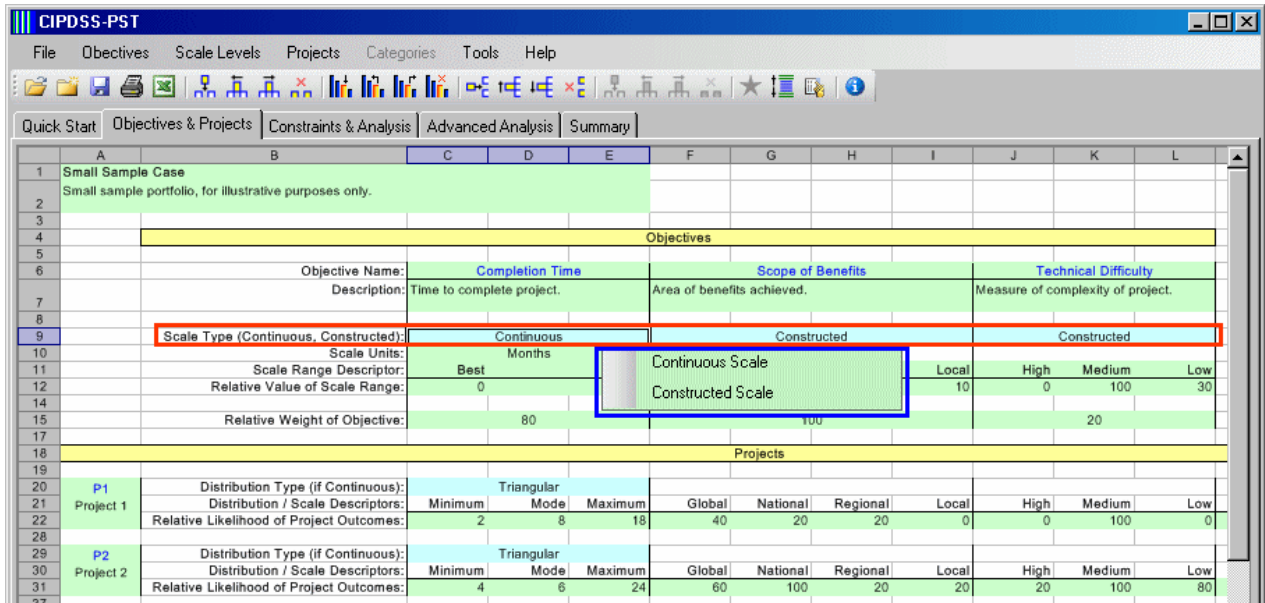
FIGURE 3.2 Objective Scale Types

An example of a continuous scale — as applied to the first objective — would be time expressed in some unit of measure, such as days or months. For this example, the time scale of months is used, and the range for that scale is 0–24 months. Alternatively, the completion time could have been represented in discrete "bins" (constructed scales). For example, four bins of 6 months each could cover the same time frame as that defined in the 0–24 month continuous scale. And when the constructed scale was used, the scale range descriptors would be changed to 0–6 months, 6–12 months, 12–18 months, and 18–24 months.

The choice between continuous and constructed scales is largely determined by user preference and how the objective is to be measured. If the user chooses a continuous scale for rating a particular objective, then a continuous probability distribution format will automatically be displayed under the corresponding project rating fields, and the user can then choose among eight different types of distributions. If the user chooses to use a constructed scale for rating an objective, then the proper number of bins will automatically be displayed under the corresponding project rating fields for that objective.

3.2.3 Defining Objective Scale Ranges

Once the scale type (and units of measure for continuous scales) is determined, as highlighted in Figure 3.2, the user enters the scale range descriptors (Figure 3.3). This entry is simply a label that serves as a reminder of what units of measure have been adopted. For continuous scales, CIPDSS-PST inserts this descriptor as the orientation for which end of the continuous range is considered best and which end of the range is considered worst. For constructed scales, the descriptors convey the range of each scale bin. For the example cited above for time (measured in months and separated into four discrete bins), the scale range

FIGURE 3.3 Objective Scale Units of Measure, Scale Ranges, and Relative Values

descriptors are 0–6 months, 6–12 months, 12–18 months, and 18–24 months. Or, as shown in Figure 3.3, the scale descriptors for the objective defined as "Scope of Benefits" are "Global," "National," "Regional," and "Local." A constructed scale can be divided into any number of bins.

3.2.4 Assigning Relative Values to Objective Scale Ranges

The final step for defining objective scales is to assign the relative values for each scale range. For continuous scales, the relative values are simply the endpoints of the distributions. Relative values are assumed to be linear and continuous over the range of the objective scale. Internally, CIPDSS-PST substitutes values in the range of 0–100 to complete the required normalizations and value calculations.

However, for constructed scales, the user must determine the values of each scale range *relative to the other scale ranges*. For example, under the "Scope of Benefits" objective, the user needs to assign relative values to each category (global, national, regional, or local). In the example, the categories have been "assessed" and reflect the highest value for national benefits (100%), second-highest value for global benefits (70%), third-highest value for regional benefits (20%), and lowest value for local benefits (10%).

As shown by this example, the value ratings do not need to total 100%. And while any value (between 0 and 100%) can be entered for any of the constructed scales, *it is highly recommended that at least one of the scale values be defined by the user as 100%* (for the scale range that represents the best outcome). CIPDSS-PST performs the necessary normalizations internally before the final utility assessments or optimization is made (see Section 3.3). For convenience, a great deal of flexibility in assigning the relative values of scale ranges is allowed, but to avoid potential confusion, it is recommended that the user assign the value of 100% to the highest-valued scale range. This convention improves the intuition for assigning relative values to the other scale ranges. Also, as shown in Figure 3.3 under the objective category of "Technical Difficulty," relative values of zero may be assigned to cases in which those outcomes are assessed to contribute no value to the overall project utility.

3.2.5 Assigning Relative Weights to Each Objective

Once the objective scales have been defined (in terms of types, units, ranges, and values), the user is able to assign relative weights for each of the objectives (outlined with blue box in Figure 3.3). The user can assign any number between 0 and 100 for any of the objectives, and the entries do not need to total 100%. However, *it is highly recommended that at least one of the objectives (the most valued) should be assigned a value of 100%.* CIPDSS-PST performs the necessary normalizations for later calculations. The convention of assigning 100% to the most-valued objective (as it did in case for assigning scale range values) improves the intuition for assigning relative weights for the other objectives.

The data entries described above are the final user inputs needed to characterize the problem objectives. The final stage of problem preparation involves rating each of the projects according to each of the objective scales. Those steps are described below in Sections 3.2.6 and 3.2.7.

3.2.6 Identifying and Costing the Projects
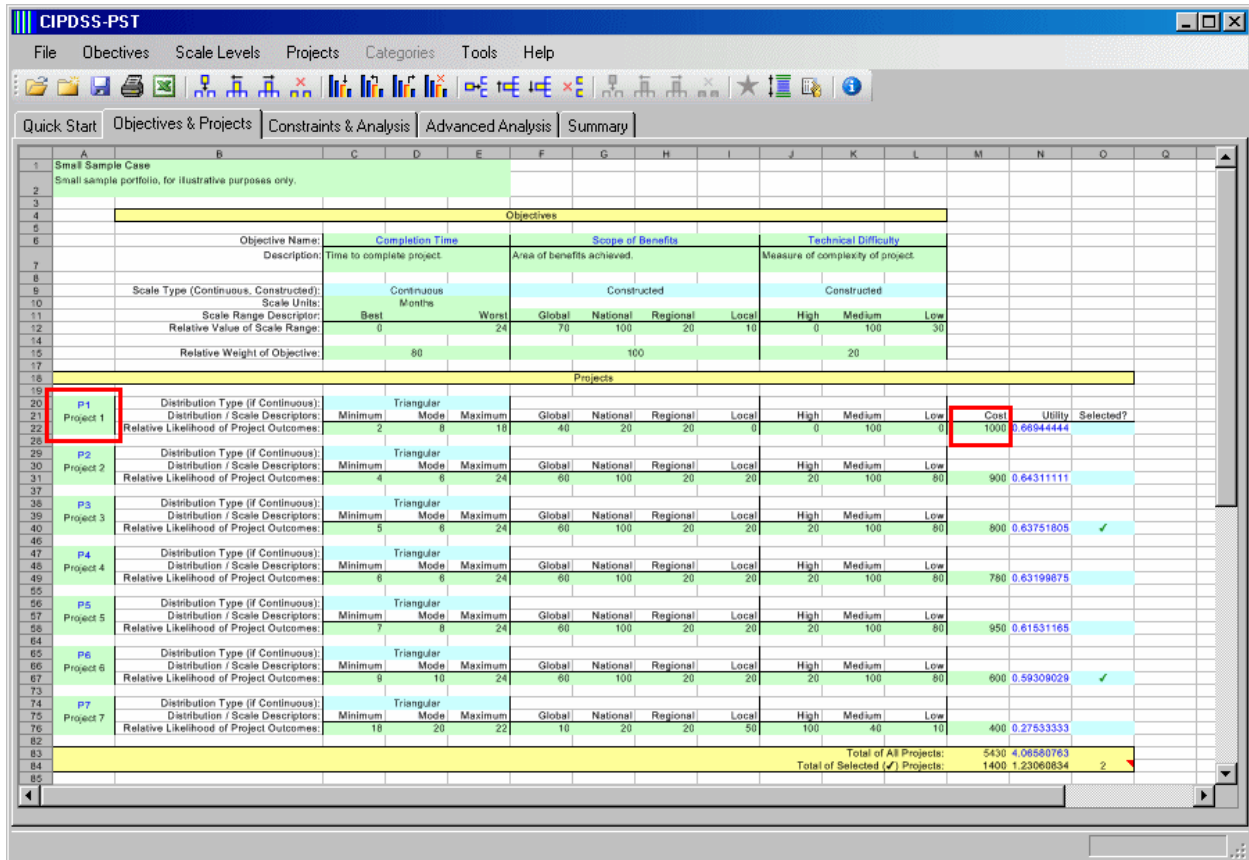
Figure 3.4 illustrates where the user enters a brief title/label and a description for each project. The title should be short enough (e.g., eight characters or fewer) to allow the compact notation to be readable without expanding the data cell and occupying a large portion of the viewable screen. The description field can be as long as needed but will collapse to a few lines under normal use. The purpose of these fields is to provide unique identifiers for each project and critical reference information for occasional reviews and reminders. The add button " ⊞ ", move up button " ⬆ ", move down button " ⬇ ", and remove button " ✕ " are used to insert, move, or delete projects from the list of candidates.

Figure 3.4 also highlights the area where a user is allowed to enter estimated cost information for each project. As noted in Section 3.1, this cost entry is a point estimate and used

FIGURE 3.4 Entering Project Names and Descriptions

only for summary purposes. It can be used in screens for sorting, but it is not the same as introducing cost explicitly as an objective.

3.2.7 Rating Projects According to the Objectives

Figure 3.5 highlights the data entry fields used to evaluate each project with respect to the objectives. The three categories of inputs are (1) distribution type (if the objective has been defined by a continuous measurement scale), (2) distribution or scale descriptor, and (3) relative likelihood of outcomes for this project. These inputs provide all the information needed to complete the calculation of project utility, which is then used to optimize the selection of projects for a given portfolio.

The distribution type is selected from a dropdown menu of eight options (Figure 3.6): triangular, normal, lognormal, uniform, single value (fixed), geometric, exponential, and binomial distributions. Once the user selects one of the distribution options for a continuous scale objective, the CIPDSS-PST program prompts for the critical parameters that define the distribution. For example, as shown in Figure 3.5, after selecting the triangular distribution to represent "Completion Time" for Project 1, the user would be prompted for three parameters: minimum, mode, and maximum. The CIPDSS-PST program automatically inserts the headings (labels) for each parameter. For this example, the user would enter the number 2 for minimum, 8 for mode, and 18 for maximum. And the interpretation for these entries would be that the completion time for Project 1 would be expected to fall between 2 and 18 months, with a most likely outcome of 8 months.

FIGURE 3.5 Project Ratings: Distribution Types, Scales, and Relative Likelihoods
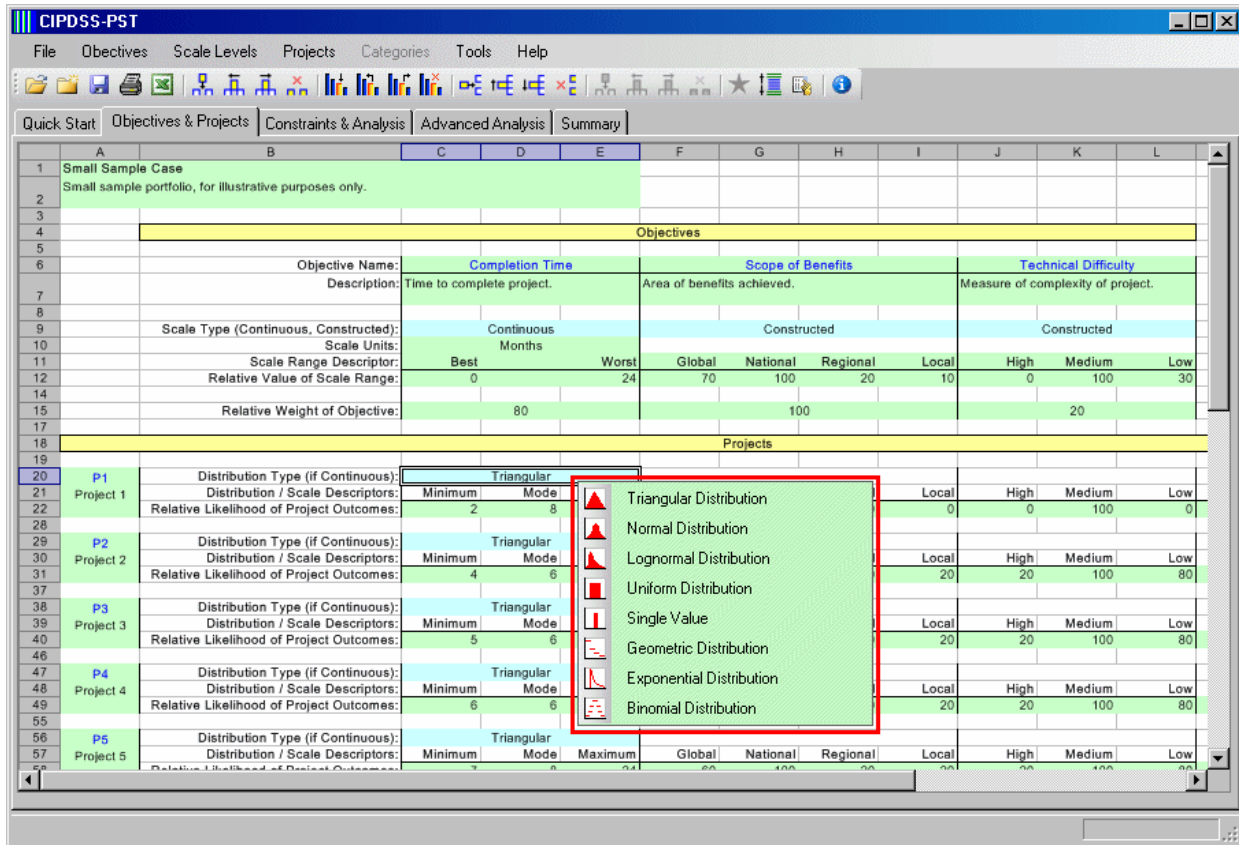
FIGURE 3.6 Distribution Types for Objectives with Continuous-scale Types

The type of distribution can be different for each project, even for the same objective. For example, the user can choose to characterize the completion time for Project 1 with a triangular distribution, that for Project 2 with a uniform distribution, and that for Project 3 with a binomial distribution. This degree of freedom can help accommodate different types of information that might be available for different projects. Similarly, different objectives for the same project can be characterized with different types of distributions.

For objectives rated with constructed (discrete) scales, the data entries are flexible and straightforward. The user first determines how many scale levels are needed to characterize the objective. For example, the "Scope of Benefits" objective has been characterized by four scale levels (bins) corresponding to global, national, regional, and local benefits. If the existing template has too few columns to accommodate the number of scale levels needed, the user can apply the add column button " ![add](.) " to add columns. Alternatively, if there are too many columns, the user can apply the remove column button " ![remove](.) " to remove columns. The move column left button " ![left](.) " and move column right button " ![right](.) " can be used to move existing scale levels to alternate locations.

Once the desired number of columns is arranged, the user provides labels to describe the scale level. So for "Scope of Benefits," the labels are "Global," "National," "Regional," and

"Local." And for "Technical Difficulty," the labels are "High," "Medium," and "Low." To rate each project according to these objective scales, the user provides numerical inputs that characterize the anticipated outcomes in the "Relative Likelihood of Project Outcomes" row. The relative likelihoods of project outcomes can be input as any number between 0 and 100, and they do not need to total 100. The CIPDSS-PST program provides normalizations for the relative likelihood inputs, so they can be properly weighted in the utility calculations.

Figure 3.7 shows where the user has an opportunity to manually select specific projects to be included in a portfolio (outlined with red box). As projects are selected, the sum of overall utility (as derived in Section 3.3) and costs are displayed at the bottom of the screen (outlined with blue box). As noted earlier, utility represents a single measure of merit that combines all of the estimated outcomes and relative importance factors as assessed by the user.

The total utility and costs for *all* projects are also displayed for reference. More sophisticated manipulations and displays of utilities, costs, and constraints are shown on the next tab (Constraints & Analysis) and described in Section 4. But for a simple one-page review and summary of the candidate projects, the screen shown in Figure 2.1 provides a useful display of characteristic information. As noted earlier, more detailed summary information can be viewed on the basic input screen by pressing the reveal button " ⁞☰ " to reveal the intermediate parameter calculations and final utility results (Figure 3.8).

FIGURE 3.7 Manual Selection of Project for a Portfolio

FIGURE 3.8 Summary Screen with Intermediate Calculation Results Displayed

3.3 PROJECT VALUE AND UTILITY DERIVATIONS FOR THE SAMPLE PROBLEM

This section shows an overview of how project values and utilities are derived from user inputs, and it also provides the detailed mathematical derivations for these values (Section 3.3.2). For this section, the following notation and conventions are used:

N = number of projects,

n = project number index,

J = number of objectives,

j = objective number index,

I_j = number of objective scale ranges for Objective j (j = 1, 2, … J), and

i = scale range index.

The column furthest to the left in Table 3.1 contains a label used for reference purposes. That same label is used for shorthand notation in the column furthest to the right, showing the source of derivations and intermediate values.

3.3.1 Overview of Sample Problem Derivations

Table 3.1 is visually different from the screen display shown in Figure 3.8, but the two are parallel in terms of the example portrayed and the user-defined inputs. Table 3.1 can be used to trace the intermediate calculations used to derive project and portfolio utility values.

For example, to derive the values used in row X6 (Normalized Relative Value of Scale Range: $\overline{RVSR}_{ij}$), Table 3.1 shows that those values equal the entries in row X5 (Relative Value of Scale Range: $RVSR_{ij}$) divided by 100. And to derive the values in row X8, each entry in row X7 is divided by the sum of all entries in row X7 ($X8 = X7 \div \sum X7$).

All of the entries in Table 3.1 are similarly derived, and the end result is a translation of user inputs, first expressed and input as *estimated outcomes for each project* (i.e., scale-level project values), into *total project utilities*. In this context, utility represents a single measure of merit that combines all of the estimated outcomes and relative importance factors as assessed by the user. The total project utilities are used to compare, rank, and select projects on a uniform and comparable basis.

Table 3.1 includes entries for three hypothetical projects, and the derivation of results can be followed to the completion of total project utility results (X14) for each of the projects. Table 3.2 summarizes the results from Table 3.1, and the entries in this table serve to initiate the portfolio optimization process. Alternatively, these values can be used to rank and guide manual selections to construct a user-preferred portfolio.

3.3.2 Detailed CIPDSS-PST Derivations

The mathematical derivations of the results for Tables 3.1 and 3.2 are shown here in greater detail.

N = number of projects in portfolio (user-defined),

n = project number index, and

PN_n = project name (for project n) (user-defined).

TABLE 3.1 Sample Problem and Internal CIPDSS-PST Calculations

Label	Variable	Values										Source
		A. Define/Name Objectives (example below with three objectives [J=3])										
X1	Objective Number and Name (j); (ONj)	1. Completion Time		2. Scope of Benefits				...	3(=J). Technical Difficulty			User
		B. Define Scale Ranges and Types (for Measuring Objective "j") and Assess Relative Value of Each Scale Range										
X2	Scale Type	Continuous/"Natural"		Discrete/"Constructed"				...	Discrete/"Constructed"			User
X3	Scale Range Index (i)	1	$2 (=I_1)$	1	2	3	$4 (=I_2)$	...	1	2	$3 (=I_J)$	User
X4	Scale Range Descriptor (SRD_{ij})	0 months ($SRmin_1$)	24 months ($SRmax_1$)	Global	National	Regional	Local	...	high	med	low	User
X5	Relative Value of Scale Range ($RVSR_{ij}$)	100 (best)	0 (worst)	70	100 (best)	20	10 (worst)	...	0 (worst)	100 (best)	30	User (≥0 & ≤100 & must include 100)
X6	Normalized Relative Value of Scale Range ($\overline{RVSR}_{ij}$)	1.0	0.0	0.700	1.000	0.200	0.100	...	0.000	1.000	0.300	Eq. 1 (≥0 & ≤1.0) X5÷100
		C. Assign Relative Weights for Each Objective										
X7	Relative Weight of Objective (RWO_j)	80		100				...	20			User (≥0 & ≤100 & should include 100)
X8	Normalized Relative Weight of Objective ($\overline{RWO}_j$)	0.400		0.500				...	0.100			Eq. 2 sum to 1 X7÷(∑X7)
		D.1 Assess Relative Likelihood of Outcomes (for Project "1")										
X9	Relative Likelihood of Project Outcomes ($RLPO_{nij}$)	2 mo. [min.]	8 mo. [mode]	18 mo. [max.]	40	20	20	0	...	0	100	0	User (≥0 & ≤100)
X10	Normalized Relative Likelihood of Project Outcomes ($\overline{RLPO}_{nij}$)	0.000	0.125	0.000	0.500	0.250	0.250	0.000	...	0.000	1.000	0.000	Eq. 3a/b sum to 1 for discrete X9÷(∑X9)
		(triangular probability distribution)		(discrete probability distribution)					(discrete probability distribution)			
		E.1 Determine Values and Utilities for Project "1" (Scale-level, Objective-level, and Total)										
X11	Scale-level Project Value ($SLPV_{nij}$)	N/A	N/A	N/A	0.350	0.250	0.050	0.000	...	0.000	1.000	0.000	Eq. 4 X6•X10 for Discrete
X12	Objective-level Project Value ($OLPV_{nj}$)	0.611 (area under (X6 • X10))		0.650 (∑i (X10))				...	1.000 (∑i (X10))			Eq. 5
X13	Objective-level Project Utility ($OLPU_{nj}$)	0.244		0.325				...	0.100			Eq. 6 X8•X12
X14	Total Project Utility (TPU_n)	0.669										Eq. 7 ∑j (X13)

TABLE 3.1 (Cont.)

Label	Variable	Values											Source
		D.2 Assess Relative Likelihood of Outcomes (for Project "2")											
X9	Relative Likelihood of Project Outcomes ($RLPO_{nij}$)	4 mo. [min.]	6 mo. [mode]	24 mo. [max.]	60	100	20	20	...	20	100	80	**User** (≥0 & ≤100)
X10	Normalized Relative Likelihood of Project Outcomes ($\overline{RLPO}_{nij}$)	0.000	0.100	0.000	0.300	0.500	0.100	0.100	...	0.100	0.500	0.400	Eq. 3a/b sum to 1 for discrete X9÷(∑X9)
		(triangular probability distribution)			(discrete probability distribution)					(discrete probability distribution)			
		E.2 Determine Values and Utilities for Project "2" (Scale-level, Objective-level, and Total)											
X11	Scale-level Project Value ($SLPV_{nij}$)	N/A	N/A	N/A	0.210	0.500	0.020	0.010	...	0.000	0.500	0.120	**Eq. 4** X6•X10 for discrete
X12	Objective-level Project Value ($OLPV_{nij}$)	0.528 (area under (X6 • X10))			0.740 (∑ᵢ (X10))				...	0.620 (∑ᵢ (X10))			**Eq. 5**
X13	Objective-level Project Utility ($OLPU_{nj}$)	0.211			0.370				...	0.062			**Eq. 6** X8•X12
X14	Total Project Utility (TPU_n)	0.643											**Eq. 7** ∑ⱼ (X13)
		... (assessments for other projects 3, 4, ... [N - 1]) ...											
		D.N Assess Relative Likelihood of Outcomes (for Project "N")											
X9	Relative Likelihood of Project Outcomes ($RLPO_{nij}$)	18 mo. [min.]	20 mo. [mode]	22 mo. [max.]	10	20	20	50	...	100	40	10	**User** (≥0 & ≤100)
X10	Normalized Relative Likelihood of Project Outcomes ($\overline{RLPO}_{nij}$)	0.000	0.500	0.000	0.100	0.200	0.200	0.500	...	0.667	0.267	0.067	Eq. 3a/b sum to 1 for discrete X9÷(∑X9)
		(triangular probability distribution)			(discrete probability distribution)					(discrete probability distribution)			
		E.N Determine Values and Utilities for Project "N" (Scale-level, Objective-level, and Total)											
X11	Scale-level Project Value ($SLPV_{nij}$)	N/A	N/A	N/A	0.070	0.200	0.040	0.050	...	0.000	0.267	0.020	**Eq. 4** X6•X10 for discrete
X12	Objective-level Project Value ($OLPV_{nij}$)	0.167 (area under (X6 • X10))			0.360 (∑ᵢ (X10))				...	0.287 (∑ᵢ (X10))			**Eq. 5**
X13	Objective-level Project Utility ($OLPU_{nj}$)	0.067			0.180				...	0.029			**Eq. 6** X8•X12
X14	Total Project Utility (TPU_n)	0.276											**Eq. 7** ∑ⱼ (X13)

TABLE 3.2 Evaluation and Optimization of Portfolio

Project No.	Project Name	Risk Neutral $(r = 0.0)$	Source
1	ABC	0.669	Eq. 7 (X14)
2	DEF	0.643	Eq. 7 (X14)
NSP[a]	XYZ	0.276	Eq. 7 (X14)
Total utility (TU) of portfolio	(X15)	1.588	Eq. 8 $\sum_n$ (X14)

[a]   Number of selected projects (selected manually or through optimization for inclusion in portfolio). Note: NSP = N if all projects are selected, and NSP = 3 in this simple example.

Row X1

J = number of objectives (user-defined),

j = objective number index, and

ON_j = objective name for Objective j (user-defined).

Row X2

ST_j = scale type for Objective j (j = 1, 2, … J) (continuous or discrete) (user-defined).

Definition: This is the type of scale used for evaluating each project with respect to Objective j. Continuous scales are typically used for natural scales, such as costs (e.g., $) or time (e.g., days/months). Discrete scales are typically applied to constructed scales, such as a technical challenge (e.g., high/medium/low) or a benefit (high/medium-high/average/medium-low/low).

Row X3

I_j = number of scale ranges for Objective j (j = 1, 2, … J) (user-defined) and

i = scale-range index.

Row X4

SRD_{ij} = scale-range descriptor for Scale Range i of Objective j (j = 1, 2, … J;
  i = 1, 2, … I_j) (user-defined).

For continuous/natural scales:

$SRmin_j$ = scale range minimum for Objective j (j = 1, 2, … J) (user-defined) and

$SRmax_j$ = scale range maximum for Objective j (j = 1, 2, … J) (user-defined).

Row X5

$RVSR_{ij}$ (user assessment)

X5a — For discrete/constructed scales:

$RVSR_{ij}$ = relative value of Scale Range i (0–100) for Objective j (j = 1, 2, … J;
  i = 1, 2, … I_j).

Definition: For Objective j, user-assessed relative value of Scale Range i outcome. At least one of the scale ranges should be assigned a relative value of 100 (best outcome).

X5b — For continuous/natural scales:

$RVSR_{ij}$ = relative value of outcomes at scale range endpoints $SRmin_j$ and $SRmax_j$
  (0 or 100) for Objective j (j = 1, 2, … J; i = 1 for $SRmin_{ij}$ and 2 for $SRmax_j$).

Definition: For Objective j, user assesses the best (100) and worst (0) outcomes. $RVSR_{1j}$ should be assigned a relative value of 0 or 100, corresponding to whether the value associated with outcome $SRmin_j$ is worst or best; and $RVSR_{2j}$ should be assigned the opposite relative value (100 or 0), corresponding to whether the value associated with $SRmax_j$ is best or worst.

Row X6

$\overline{RVSR}_{ij}$ = normalized relative value of scale range (0–1.0) for Scale Range i of
  Objective j (j = 1, 2, … J; i = 1, 2, … I_j).

Definition: For each Objective j, $RVSR_{ij}$ values scaled/normalized to span range of 0.0–1.0 for each Scale Range i (i = 1, 2, … I_j).

$\overline{RVSR}_{ij}$ = $RVSR_{ij}/100$ for each Objective j (j = 1, 2, … J; i = 1, 2, … I_j). (Eq. 1)

Row X7

RWO_j = relative weight of objective (0–100) for Objective j (j = 1, 2, … J)
(user assessment).

Definition: User-assessed relative weight for each Objective j as it contributes to overall value of portfolio of projects. One of the relative weights for each objective should be assigned a relative value of 100 (most important objective).

Row X8

$\overline{RWO}_j$ = normalized relative weight of objective for Objective j (j = 1, 2, … J).

Definition: For all objectives, RWO_j scaled to sum to 1.0 over j (j = 1, 2, … J).

$\overline{RWO}_j = RWO_j/(\sum RWO_j$ [summed over j = 1, 2, …J]) for each j = 1, 2, … J).   (Eq. 2)

Row X9

$RLPO_{nij}$ (user assessment)

X9a — For discrete/constructed scales:

$RLPO_{nij}$ = relative likelihood of project outcomes (0–100) for Project n, Scale Range i, of Objective j (j = 1, 2, … J; i = 1, 2, … I_j; n = 1, 2, … N).

Definition: User-assessed relative likelihood of Project n resulting in Scale Level i outcome for Objective j. Assessed values do not need to sum to 100 (or any other number). CIPDSS-PST will normalize assessments in $\overline{RLPO}_{nij}$ to sum to 1.0.

X9b — For continuous/natural scales:

$RLPO_{nij}$ = relative likelihood distribution (minimum, mode, and maximum) of project outcomes (min, mode, max) for Project n, Objective j (j = 1, 2, … J; n = 1, 2, …N; i = 1{min}, 2{mode}, 3{max}), where min, mode, and max are located within the range of ($SRmin_j – SRmax_j$):
Minimum = $RLPO_{n1j} \geq SRmin_j$;
Mode = $RLPO_{n2j}$, and $RLPO_{n1j} \geq RLPO_{n2j} \leq RLPO_{n3j}$;
Maximum = $RLPO_{n3j} \leq SRmax_j$.

Definition: User-assessed triangular distribution of likely outcomes for Project n, Objective j, where i = 1{min}, i = 2{mode}, i = 3{max}.

Row X10

$\overline{RLPO}_{nij}$

X10a — For discrete/constructed scales:

$\overline{RLPO}_{nij}$ = normalized relative likelihood of project outcomes (0–1.0) for Project n, Scale Range i, of Objective j (j = 1, 2, … J; i = 1, 2, … I$_j$; n = 1, 2, … N).

Definition: For each Project n and Objective j, RLPO$_{nij}$ values scaled to sum to 1.0 over i = 1, 2, … Ij).

$\overline{RLPO}_{nij}$ = RLPO$_{nij}$/($\sum$ RLPO$_{nij}$ [summed over i = 1, 2, …I$_j$]) for each j = 1, 2, … J;
n = 1, 2, … N. (Eq. 3a)

X10b — For continuous/natural scales:

$\overline{RLPO}_{nij}$ = normalized relative likelihood distribution of project outcomes for Project n, Objective j (j = 1, 2, … J; n = 1, 2, … N).

(CIPDSS-PST determines curve height at mode, so area under triangular distribution equals 1.0.)

Definition: For each Project n and Objective j, $\overline{RLPO}_{n2j}$ value (triangular distribution height at RLPO$_{n2j}$) calculated so area under triangular distribution between RLPO$_{n1j}$ and RLPO$_{n3j}$ = 1.0.

$\overline{RLPO}_{nij}$ = $\overline{RLPO}_{n1j}$ = 0.

$\overline{RLPO}_{nij}$ = $\overline{RLPO}_{n2j}$ = 2/(RLPO$_{n3j}$ – RLPO$_{n1j}$) for each j = 1, 2, … J;
n = 1, 2, … N. (Eq. 3b)

$\overline{RLPO}_{nij}$ = $\overline{RLPO}_{n3j}$ = 0.

Note: Linear segments fill in triangular distribution between $\overline{RLPO}_{n1j}$, $\overline{RLPO}_{n2j}$, and $\overline{RLPO}_{n3j}$.

Row X11

SLPV$_{nij}$

X11a — For discrete/constructed scales:

SLPV$_{nij}$ = scale-level project value for Project n, Scale Range i, of Objective j (j = 1, 2, … J; i = 1, 2, … I$_j$; n = 1, 2, … N).

Definition: For Project n, the estimated value contribution for each Scale Range i of each Objective j, derived by weighting the normalized relative likelihood of project outcomes (assessed by scale range) by the normalized relative values of each scale range.

$$SLPV_{nij} = (\overline{RLPO}_{nij})(\overline{RVSR}_{ij}) \text{ for } j = 1, 2, \ldots J; i = 1, 2, \ldots I_j; n = 1, 2, \ldots N).$$ (Eq. 4a)

(X11b) — For continuous/natural scales:

It is not meaningful to display distribution results at scale level. Values are calculated internally in CIPDSS-PST for entire range of outcome distributions by multiplying $\overline{RLPO}_{nij}$ distribution by $\overline{RVSR}_{ij}$ distribution. These intermediate results are used to derive the subsequent values of $OLPV_{nj}$ as noted below.

$$SLPV_{nij} = (\overline{RLPO}_{nij})(\overline{RVSR}_{ij}) \text{ for } j = 1, 2, \ldots J; n = 1, 2, \ldots N.$$ (Eq. 4b)

Row X12

$OLPV_{nj}$

X12a — For discrete/constructed scales:

$OLPV_{nj}$ = objective-level project value for Project n, Objective j (j = 1, 2, … J; n = 1, 2, … N).

Definition: For each Project n, the estimated value contribution for each Objective j, derived by summing scale-level project values for all scale ranges in the objective.

$$OLPV_{nj} = \sum (SLPV_{njj} \text{ [summed over } i = 1, 2, \ldots I_j]) \text{ for } j = 1, 2, \ldots J;$$
$$n = 1, 2, \ldots N.$$ (Eq. 5a)

X12b — For continuous/natural scales:

$OLPV_{nj}$ = objective-level project value for Project n, Objective j (j = 1, 2, … J; n = 1, 2, … N).

Definition: For each Project n, the estimated value contribution for each Objective j, derived by integrating scale-level project values over entire scale range of the objective.

$$OLPV_{nj} = \int (SLPV_{njj} \text{ [integrated over } RLPO_{n1j} \text{ to } RLPO_{n3j}])$$
$$\text{for } j = 1, 2, \ldots J; n = 1, 2, \ldots N.$$ (Eq. 5b)

Row X13

OLPU$_{nj}$ = objective-level project utility for Project n, Objective j (j = 1, 2, … J;
    n = 1, 2, …N).

Definition: For Project n, the estimated utility contribution of each Objective j, derived
by multiplying objective-level project values by the normalized relative weight of each
Objective j.

OLPU$_{nj}$ = (OLPV$_{nj}$) ($\overline{RWO}_j$) for Objective j (j = 1, 2, … J; n = 1, 2, … N).   (Eq. 6)

Row X14

TPU$_n$ = total project utility for Project n (n = 1, 2, … N).

Definition: For Project n, the total estimated utility contributions from all J objectives,
derived by summing objective-level project utilities over all objectives.

TPU$_n$ = $\sum$ OLPU$_{nj}$ [summed over j = 1, 2, … J] for n = 1, 2, … N.   (Eq. 7)

NSP = number of selected projects (manually or through optimization) to be included
    in portfolio

Row X15

TU = total utility for all projects, or all "selected" projects, in portfolio.

Definition: The total estimated utility contributions from all N projects in a portfolio or
all NSP-selected projects in a portfolio, derived by summing the total project utilities
over all or selected projects.

TU = $\sum$ TPU$_n$ [summed over n = 1, 2, … N or n = 1, 2, … NSP].   (Eq. 8)

4 CONSTRAINTS AND ANALYSIS

This section describes the capabilities and options for introducing user-defined constraints to address portfolio optimization problems. It also provides an overview of how to graph and interpret the results.

4.1 CONSTRAINTS & ANALYSIS TAB

Once all the objectives and projects are defined, the Constraints & Analysis Tab can be used to proceed with in-depth analysis and optimization. The user can manually select projects for a portfolio and perform limited analysis from the Objectives & Projects Tab, but the Constraints & Analysis Tab opens up many more possibilities for the analysis. Figure 4.1 shows the basic screen layout for the Constraints & Analysis Tab. The portion of this screen outlined in red shows a basic list of the projects, including summaries for total
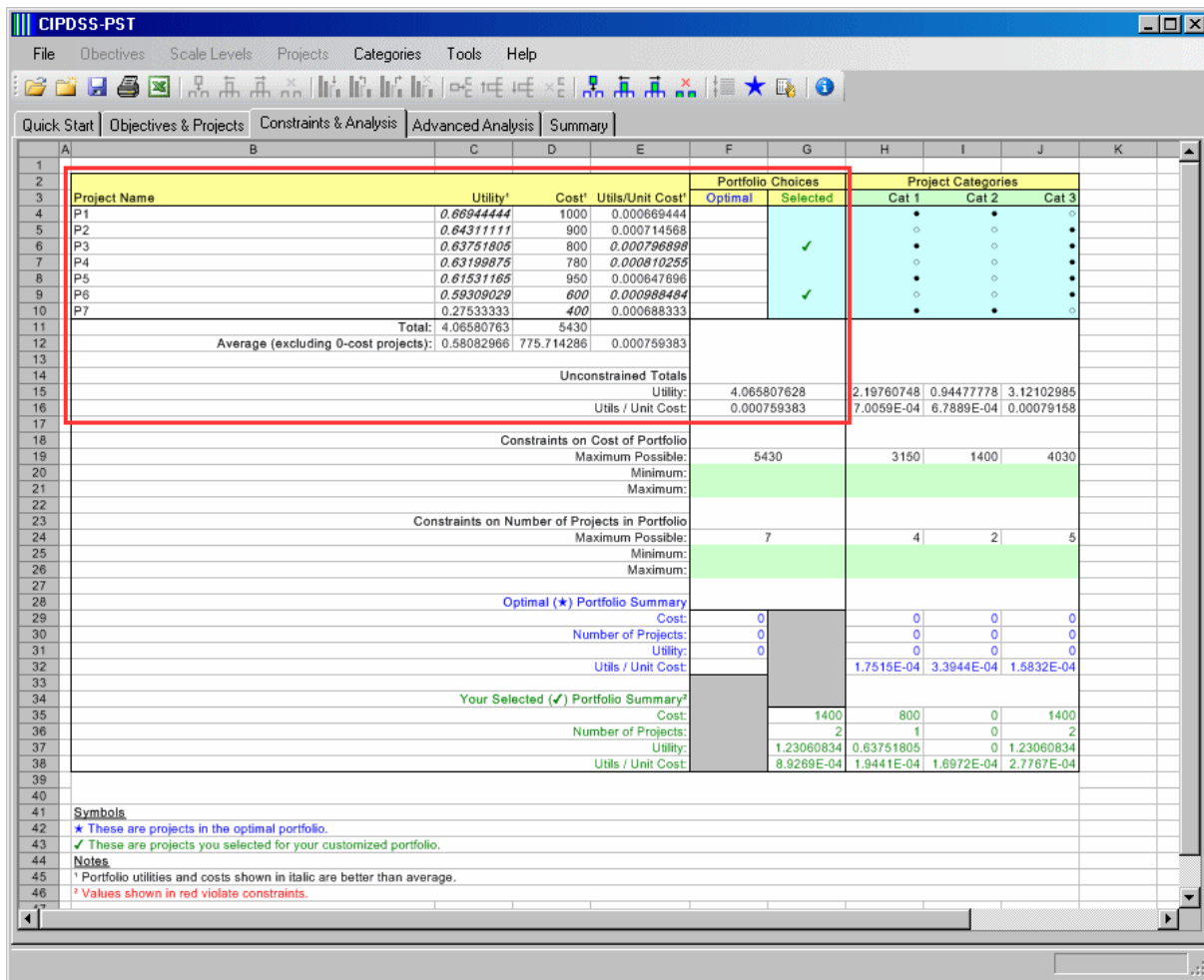
FIGURE 4.1 Capabilities Available from Constraints & Analysis Screen

utility, cost, and utility per unit-cost (utils/unit-cost). The relative value of different projects can be assessed from the utils/unit-cost parameter, without consideration of resource or cost constraints or of inclusion or exclusion requirements. This metric provides a general indicator of which projects should be selected first to maximize the total expected value of a portfolio. However, when user-defined constraints are introduced, this simple approach to rank-ordering projects typically fails to produce the constrained optimal solution. Section 4.1.2 describes how the project categorizations can be used to implement user-defined constraints.

The blue-background cells near the top of Figure 4.1, under the heading "Portfolio Choices/Selected," show where the user can click to select (✓) or unselect ( ) projects to be included in a portfolio. As projects are included or excluded (selected or unselected), the totals under each column at the bottom of the screen (displayed in green lettering) change to reflect new costs and utilities.

4.1.1 Portfolio Optimization with Total Portfolio Constraints

Figure 4.2 highlights parameter fields that can be used effectively to optimize portfolio selections by using total-portfolio constraints (non-project-specific constraints). The highlighted columns show:

- Cost constraints (maximum and/or minimum),

- Number of projects constraints (maximum and/or minimum),

- Results for optimized portfolio (cost, number of projects, utility, and utils/unit-cost), and

- Results for manually selected portfolio (cost, number of projects, utility, and utils/unit-cost).

The green-background cells, shown in the featured areas of Figure 4.2, indicate where the user has opportunities to define constraints (on total costs and/or numbers of projects to be selected). For each of these data entry fields, the potential maximums are shown for reference; they are based on the projects that have been defined. If the user enters limits that are outside the range of [0 – potential maximum], CIPDSS-PST will reset the entry to the maximum or minimum possible value. For example, if the user enters $6000K (i.e., $6 million) for the maximum cost constraint, CIPDSS-PST will reset the entry to $5430K, which corresponds to the total cost of all possible projects that have been defined.

After setting the cost and number of projects constraints, the user can activate the optimization function by pressing the optimization button " ★ ". The example in Figure 4.2 shows a case where the maximum cost has been set at $1500K (out of a possible maximum of $5430K).

CIPDSS-PST

File   Objectives   Scale Levels   Projects   Categories   Tools   Help

Quick Start | Objectives & Projects | Constraints & Analysis | Advanced Analysis | Summary |

				Portfolio Choices		Project Categories		
Project Name	Utility¹	Cost¹	Utils/Unit Cost	Optimal	Selected	Cat 1	Cat 2	Cat 3
P1	0.66944444	1000	0.000669444			•	•	○
P2	0.64311111	900	0.000714568	★		○	○	•
P3	0.63751805	800	0.000796896		✓	•	○	•
P4	0.63199875	780	0.000810255			○	○	•
P5	0.61531165	950	0.000647696			•	•	•
P6	0.59309029	600	0.000988484	★	✓	○	○	•
P7	0.27533333	400	0.000688333			•	•	○
Total:	4.06580763	5430						
Average (excluding 0-cost projects):	0.58082966	775.714286	0.000759383					

Unconstrained Totals
Utility: 4.065807628   2.19760748 0.94477778 3.12102985
Utils / Unit Cost: 0.000759383   7.0059E-04 6.7889E-04 0.00079158

Constraints on Cost of Portfolio
Maximum Possible: 5430   3150 1400 4030
Minimum:
Maximum: 1500

Constraints on Number of Projects in Portfolio
Maximum Possible: 7   4 2 5
Minimum:
Maximum:

Optimal (★) Portfolio Summary
Cost: 1500   0 0 1500
Number of Projects: 2   0 0 2
Utility: 1.2362014   0 0 1.2362014
Utils / Unit Cost: 8.5153E-04   1.7515E-04 3.3944E-04 3.5638E-04

Your Selected (✓) Portfolio Summary²
Cost: 1400   800 0 1400
Number of Projects: 2   1 0 2
Utility: 1.23060834   0.63751805 0 1.23060834
Utils / Unit Cost: 8.9269E-04   1.9441E-04 1.6972E-04 2.3797E-04

Symbols
★ These are projects in the optimal portfolio.
✓ These are projects you selected for your customized portfolio.
Notes
¹ Portfolio utilities and costs shown in italic are better than average.
² Values shown in red violate constraints.

Optimization: 2 projects included in optimal portfolio.

FIGURE 4.2 Optimized Results with Cost Constraints

The final optimized results are displayed in Figure 4.2. The portfolio consists of two projects, P2 + P6, with a total utility value of 1.236 and total cost of $1500K. These results are shown side-by-side with the user-selected combination of projects P3 + P6, which yielded a lower utility value of 1.231 and totaled $1400K in cost.

In this simple example, the optimal value may not be difficult to find manually, but if the upper bound on costs is higher, the task becomes more difficult. With a maximum cost limit of $3000K, a user might select a combination of P1 + P5 + P6 + P7, with a total utility of 2.153 and cost of $2950K. Or the user might locate a better solution, with higher utility and lower cost, by selecting P2 + P3 + P4 + P7, resulting in a utility of 2.188 and cost of $2880K. Still, the solution can be further improved as shown with the optimized solution of P1 + P3 + P4 + P7, with a total utility of 2.214 and cost of $2980K (Figure 4.3). Larger problems with more complex constraints can make it virtually impossible to find the optimal solution by manual inspection, so the optimization capability becomes more essential as the problem grows in size and complexity.
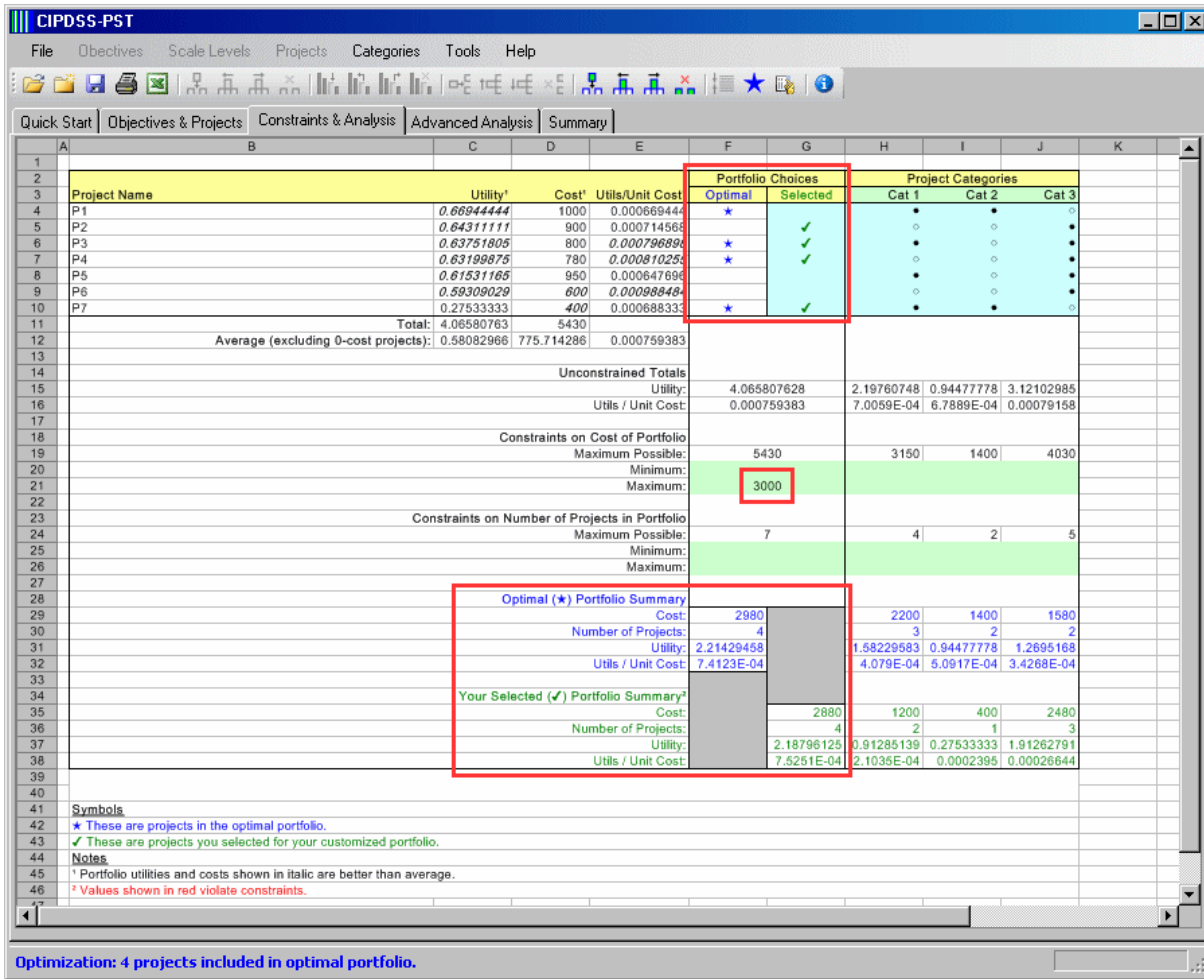
FIGURE 4.3 Optimized Results with Modified Cost Constraints

4.1.2 Portfolio Optimization with Project-specific Constraints

In addition to constraints that apply to the total portfolio, CIPDSS-PST provides the user with options for implementing project-specific constraints. Figure 4.4 illustrates the screen area where the user is allowed to define project categorizations and then assign each project to any of the predefined categories. These categorizations provide a great deal of control for the user to introduce new limitations and conditional constraints for the portfolio optimization process.

New categories are added by pressing the add button " ⬚ " and entering a name in the green-background cell near the top of the screen. The order of categories can be rearranged after the categories are defined by using the move left button " ⬚ " or move right button " ⬚ ". Categories can be deleted by using the delete category button " ⬚ ".

Once the categories are defined, the user can introduce many associations between the individual projects and the different categorizations. For the example shown in Figure 4.4, three project categories have been defined: Cat 1, Cat 2, and Cat 3. The icon " • " indicates that a

CIPDSS-PST

File   Objectives   Scale Levels   Projects   Categories   Tools   Help

Quick Start | Objectives & Projects | Constraints & Analysis | Advanced Analysis | Summary

Project Name	Utility¹	Cost¹	Utils/Unit Cost¹	Optimal	Selected	Cat 1	Cat 2	Cat 3
				Portfolio Choices		Project Categories		
P1	0.66944444	1000	0.000669444	★		•	•	○
P2	0.64311111	900	0.000714568		✓	○	○	•
P3	0.63751805	800	0.000796898	★	✓	•	○	•
P4	0.63199875	780	0.000810255	★	✓	○	○	•
P5	0.61531165	950	0.000647696			•	○	•
P6	0.59309029	600	0.000988484			○	○	•
P7	0.27533333	400	0.000688333	★	✓	•	•	○
Total:	4.06580763	5430						
Average (excluding 0-cost projects):	0.58082966	775.714286	0.000759383					

Unconstrained Totals

		Utility:	4.065807628			2.19760748	0.94477778	3.12102985
		Utils / Unit Cost:	0.000759383			7.0059E-04	6.7889E-04	0.00079158

Constraints on Cost of Portfolio

		Maximum Possible:	5430			3150	1400	4030
		Minimum:						
		Maximum:	3000					

Constraints on Number of Projects in Portfolio

		Maximum Possible:	7			4	2	5
		Minimum:						
		Maximum:						

Optimal (★) Portfolio Summary

		Cost:	2980			2200	1400	1580
		Number of Projects:	4			3	2	2
		Utility:	2.21429458			1.58229583	0.94477778	1.2695168
		Utils / Unit Cost:	7.4123E-04			4.079E-04	5.0917E-04	3.4268E-04

Your Selected (✓) Portfolio Summary²

		Cost:		2880		1200	400	2480
		Number of Projects:		4		2	1	3
		Utility:		2.18796125		0.91285139	0.27533333	1.91262791
		Utils / Unit Cost:		7.5251E-04		2.1035E-04	0.0002395	0.00026644

Symbols
★ These are projects in the optimal portfolio.
✓ These are projects you selected for your customized portfolio.
Notes
¹ Portfolio utilities and costs shown in italic are better than average.
² Values shown in red violate constraints.

Optimization: 4 projects included in optimal portfolio.

FIGURE 4.4 Portfolio Optimization with Project-specific Categorization Constraints

project is associated with a given category, while the icon "O" indicates that a given project is not associated with that category. So in this example, Projects P1 and P7 are associated with categories Cat 1 and 2; Projects P2, P4, and P6 are associated with Cat 3; and Projects P3 and P5 are associated with Cat 1 and 3.

As a hypothetical interpretation, Cat 1 projects might correspond to R&D-related efforts, Cat 2 projects could correspond to new ventures, and Cat 3 could represent maintenance-related projects. So in this example, if the user wanted to ensure that at least one project was selected from each of the general categories, a set of minimum constraints (each set at the value of 1) could be introduced in the green-background data entry cells labeled "Constraints on Number of Projects in Portfolio," across from the "Minimum" heading and under each category (Figure 4.5). And if the user also wanted to ensure that no more than two projects were selected from any of the categories, similar entries set at the value of 2 could be entered across from the "Maximum" heading for "Constraints on Number of Projects in Portfolio" data entry fields, under each project category. These entries are also shown in Figure 4.5.
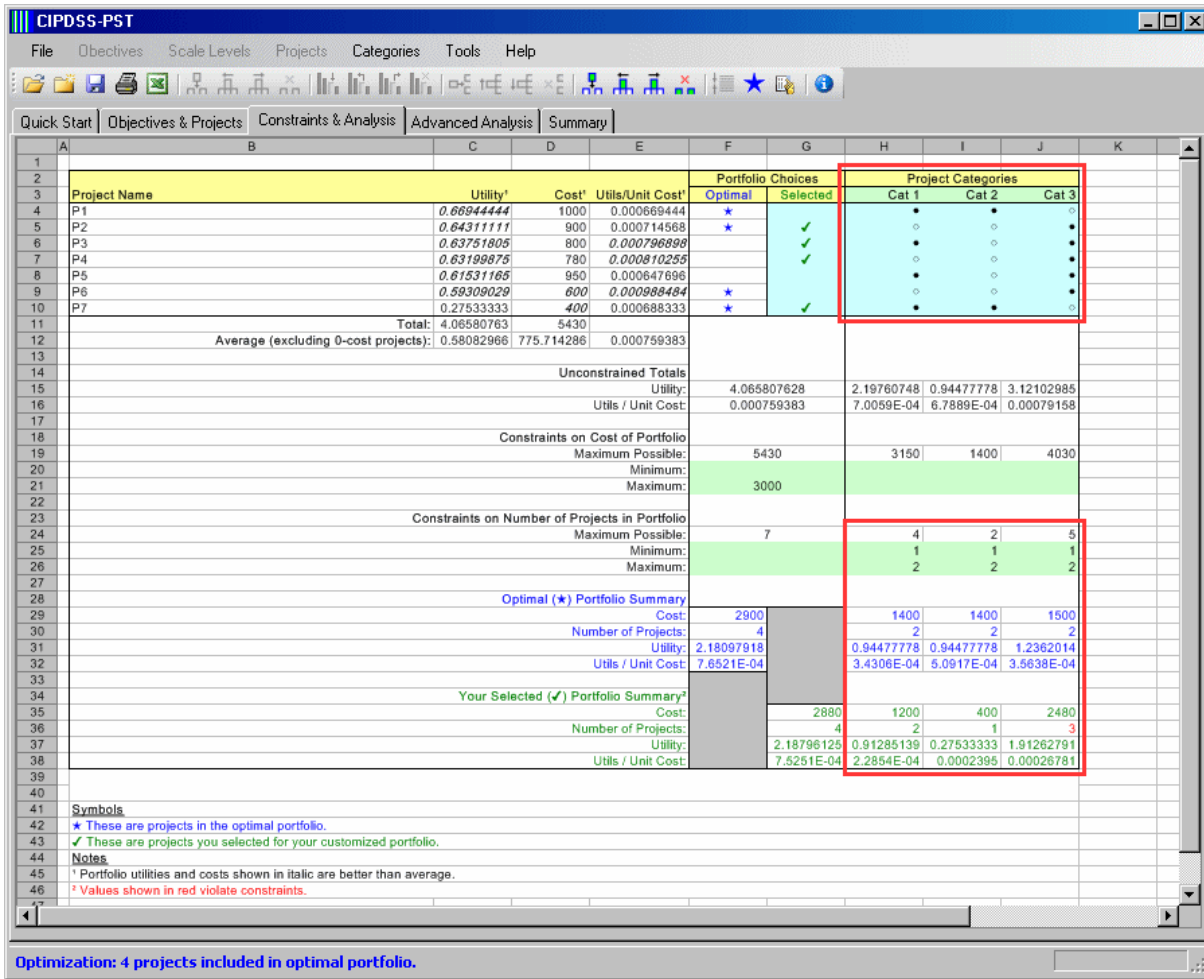
FIGURE 4.5 Results with Project Categorization Constraints Included

After these new constraints have been defined, the problem can be reoptimized, and the results as shown in Figure 4.5 will demonstrate that the selection of projects has changed: Now projects P1, P2, P6, and P7 represent the optimal collection of projects to satisfy all of the constraints. The total utility of this solution has dropped from 2.214 (in the case with only one constraint — a total cost constraint) to 2.181 (in a new case with a total cost constraint and also the project minimum and maximum constraints).

An additional observation that can be made from Figure 4.5 is that one of the summary fields for the user-defined portfolio has changed to the color red under the Cat 3 column. This indicates that the user selections violated at least one constraint for numbers of projects in that category. In this case, the user-selected portfolio included three projects from Cat 3, and this choice violated the requirement to include no more than two projects from any of the three categories. The red highlights in CIPDSS-PST are there to help the user recognize when manually selected portfolios do not meet the prescribed limitations.

In some cases — especially when many constraints have been defined — the problem may not have a valid solution (i.e., it may be overconstrained). In these cases, CIPDSS-PST will print an "Optimization: Optimal Portfolio Could Not Be Found" message at the bottom of the screen to warn the user that an optimal solution was not found and that the constraints need to be relaxed.

The preceding example uses constraints on the *numbers* of projects to illustrate this use of project-specific limitations. CIPDSS-PST also allows similar limitations to be placed on the *costs* of projects in each category. Implementation is parallel for costs and numbers of projects.

4.1.3 Portfolio Optimization with More Complex Constraints

In addition to simple upper and lower bounds on groups of projects, the category constraints can also be used to construct sophisticated conditional constraints on various projects. For example, suppose that a user would like the final portfolio to include one of two projects but not both of them. This can be accomplished by creating a new category, identifying both of the projects as belonging to this category, and then constraining the number of projects with a minimum of "1" and a maximum of "1." The result is that exactly one project is selected in the optimal portfolio, and CIPDSS will select the best of those two choices on the basis of the project utilities and any other constraints that are defined for the problem.

Table 4.1 summarizes some of the basic constraints (maximums and minimums) and more sophisticated constraints (conditional constraints) that can be implemented in CIPDSS-PST. This list is not exhaustive. The user can construct more complex constraints to meet the needs of project-specific considerations.

The preceding examples are intended to provide insights on creative ways to implement conditional constraints by using the simple category constraint framework provided in CIPDSS-PST.

One note of caution to the user: When introducing "pseudo" projects (as outlined in Table 4.1), be careful when viewing and interpreting the cost and utility totals as reported in column and screen summaries. There is a potential for double counting in the grand totals, because costs for a given project may be embedded in more than one of the projects/pseudo-projects. Cost totals reported for the final solutions are valid as long as care is used in defining constraints that force the "dependent" project alternatives to be implemented in a way that ensures independence. In other words, it is okay to define separately a Project A (with only costs for A) and a Project A+B (with costs for both A and B), as long as there is a constraint that ensures that only one of these choices can be selected for the final portfolio.

TABLE 4.1 Examples of Optional Constraint Types and Guidelines for Implementation

Constraint Description	Constraint Implementation
Minimum: Include at least "**n**" projects from Category **X** in the solution.	Create a Category **X** for this constraint, designate the intended projects to be included in the category, and constrain the minimum number of projects to be "**n**."
Maximum: Include no more than "**n**" projects from Category **Y** in the solution.	Create a Category **Y** for this constraint, designate the intended projects to be included in the category, and constrain the maximum number of projects to be "**n**."
One of Multiple: Include one and only one project from Category **Z** in the solution.	Create a Category **Z** for this constraint, designate the intended projects to be included in the category, and constrain the minimum and maximum number of projects to be "**1**."
Prerequisite: Project **A** is prerequisite for Project **B**. (If Project **B** is selected, then Project **A** must also be selected. But Project **A** can be selected without Project **B** being included.)	Create a Category **XX**. Define Project **A** as usual and define a pseudo "project" named **A+B** (with characteristics and costs associated with the implementation of both projects). Then assign both **A** and **A+B** to Category **XX**, and constrain Category **XX** to include no more than one project in the solution (maximum = 1).
Mutual Dependency: Projects **A** and **B** must be completed in tandem or not at all. (Include Project **B** if Project **A** is selected **and** include Project **A** if Project **B** is selected.)	Combine project definitions for **A** and **B** into one pseudo "Project" named **A+B** (with characteristics and costs associated with the implementation of both projects). No other constraints are needed.
Multiple "Chained" Prerequisites: Projects **A and B** are prerequisites for Project **C**, and Project **A** is a prerequisite for Project **B**. (Include Project **C** only if **A and B** are selected, and include **B** only if **A** is selected.)	Create a Category **YY** for this constraint. Then define Project **A** as usual, and define a pseudo "Project" named **A+B** (with characteristics and costs associated with both projects), and a pseudo "Project" **A+B+C** (with characteristics and costs associated with all three projects). Designate all three of these "Projects" (**A**, **A+B**, and **A+B+C**) to belong to Category **YY**. Then constrain Category **YY** to include no more than one project in the solution (maximum = 1).
Multiple "Independent" Prerequisites: Projects **A and B** are both prerequisites for Project **C**, but Projects **A** and **B** are otherwise independent of each other if **C** is not selected. (Include Project **C** only if **A and B** are both selected, but Project **A** is not necessary for **B**, and Project **B** not necessary for **A**.)	Create a Category **ZZ** for this constraint. Define Projects **A** and **B** as usual, and also define pseudo "Projects" **A+B** and **A+B+C**. Designate all four of these "Projects" (**A**, **B**, **A+B**, and **A+B+C**) to belong to Category **ZZ**. Then constrain Category **ZZ** to include no more than one project in the solution (maximum = 1).

4.2 RESULTS AND SENSITIVITY ANALYSIS

The optimal portfolio selections are displayed in tabular form on the "Constraints and Analysis" screen (Figure 4.6). The listings highlighted in blue *letters* (not the blue *background* fields) correspond to the optimized selections. Figure 4.6 shows that these summary listings include project selections (blue stars), total cost, total number of projects selected, total utility, and average utility per unit-cost (utls/unit-cost). Across the page from these total portfolio summary values are the category-specific summaries.

FIGURE 4.6 Optimal Portfolio Results

The summary results shown in *green* letters correspond to the user-specified portfolio selections. Entries in *red* letters in the user-specified area indicate values that violate one or more of the constraints.

Examples in Section 3 and Section 4 illustrate that the results can (and might be expected to) change in response to user-defined constraints. However, if new constraints are not binding (i.e., if they are already met by an existing optimal solution), the optimal solution may not change from a previous solution.

The user can take advantage of the project category constraint features to assist with sensitivity analysis. As shown by the example illustrated in Figure 4.6, if the user wanted to explore the second-best solution to this problem, one approach would be to create a new category (e.g., Cat 4), assign all of the optimal projects to this category, and then use the maximum constraint for that category to ensure that the previous solution could not be a feasible alternative for the next solution. So in this case, Projects P1, P2, P6, and P7 would all be assigned to Cat 4, and the maximum constraint for Cat 4 would be set at "3" to ensure that one or more of the

previous optimal projects would not be not included in a new optimal solution. Continuing to add constraints of this type can iteratively reveal the next-best solution or nearly next-best solution for the portfolio selections.

This approach is not guaranteed to produce the true next-best solution, however, because there might be other near-optimal solutions of interest that include *all* of the previous project selections, plus one or more additional projects. Costs for such a solution would be higher than the original solution, but the overall utility would also presumably be higher. Such solutions could be feasible and of interest but would not be revealed from using the added category constraint as described above. So another step would be needed to show possible competing candidates for the next-best solution. That next step would be to rerun the modified problem, but with the "Cat 4 maximum = 3" constraint replaced by "Cat 4 minimum = 4" and "minimum = 5" set for all (the entire set of) projects. This step would ensure that the previous set of four projects were included in a new solution, along with at least one additional project. The solution from this experiment would then be compared with the solution from the results of "Cat 4 maximum = 3" to determine which of the two outcomes represented the next-best outcome.

Another, more straightforward method of examining alternative optimal solutions is to modify the maximum or minimum cost constraints to ensure that the previous optimum is not feasible for a subsequent solution. For the example in Figure 4.6, which shows an optimal solution with a total cost of $2900K, the user could change the maximum cost to $2899K, and this would force an alternative solution to be found. Similarly, the user might set the minimum cost constraint at $2901K, which would also force another solution to be located. The directions (higher or lower cost constraints) for following these sensitivity tests would depend on the user's interests and the nature of the problem being examined. However, the user should be aware that in using this approach, multiple solutions that have identical costs might be overlooked.

4.3 EXPORTING RESULTS

Results from CIPDSS-PST can be exported for further analysis or reporting. One method is to prepare a summary report by using the Word button " 🖨 ". This feature prepares a Microsoft Word document that includes definitions and assessments of projects, objectives, constraints, and solutions. The other method is to use the Excel button " 🗷 ", which saves an ".xls" file for later use as a spreadsheet in Microsoft Excel. The user should be aware that saved spreadsheets do not contain the equations or optimization functionality embedded in CIPDSS-PST; thus, saved spreadsheets would be intended primarily for review and formatting or for post-optimization calculations. The user could add customized calculations into one of these post-optimization spreadsheets to derive additional problem analysis metrics or compare alternative solutions.

4.4 ADVANCED ANALYSIS CAPABILITIES

In addition to the usual constrained optimization features already described, CIPDSS-PST offers an advanced formulation feature. Under the "Advanced Analysis" tab, the user can customize any type of constraint that the optimization engine can accept. To use

this feature, the user is encouraged to become familiar with lp_solve capabilities and syntax (Notebaert et al. 2008)). Very complex constraints can be constructed to accommodate special needs, and the user can also modify the objective function specifications. The downside of using this feature is that the constraint construction is not menu-driven and requires greater attention from the user to avoid errors or improper constraint representations. It is also important for the user to clear (delete) any customized modifications when he/she is finished using the advanced features, so that the problem formulation matches the data input menus (as contained in the "Objectives & Projects" and "Constraints & Analysis" tabs).

4.5 USER PREFERENCES

The user preferences button " 📇 " is included for customizing how the CIPDSS-PST display appears and how the screens respond to scrolling. This button brings up the window shown in Figure 4.7.

As shown, the preferences window provides options for these default behaviors:

- Displaying or not displaying all rows (i.e., expanded or hidden);

- Saving the changes that a user has made to column widths and row heights;

- Showing row and column headings (numbers and letters), as in conventional spreadsheet format;

- Showing grid lines for rows and columns;
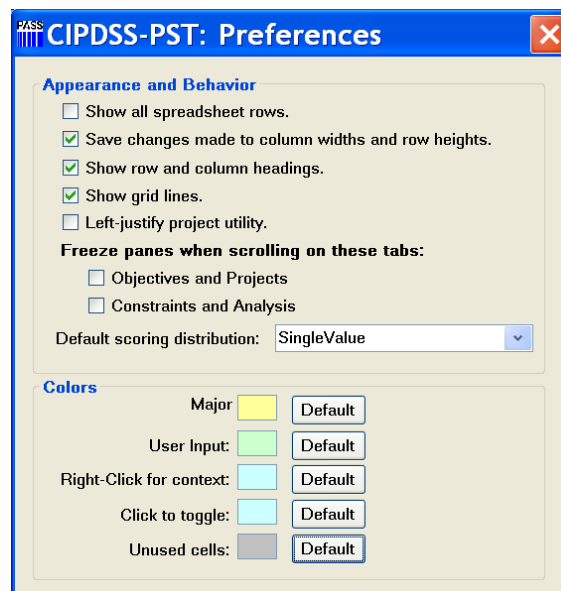
FIGURE 4.7 User Preferences Menu

- Applying left justification to the total project utility (so screen is always in view without the user having to pan across wide pages);

- Freezing header panes (to keep row and column labels in view while user is panning and scrolling);

- Selecting the default type of project scoring distribution to be used when new projects are being added; and

- Choosing colors.

When implementing the freeze pane options, the user will notice a double image of the row and column headers until the first panning or scrolling has been performed. Thereafter, the screen will show a single image of the headers, and they will stay in view while the rest of the table entries are panned or scrolled for viewing.

5 OPTIMIZATION ENGINE

CIPDSS-PST uses an optimization engine to derive the optimal mix of projects that satisfies the constraints and maximizes the total portfolio utility value. The optimization engine is invoked by pressing the optimization button "★", and the results are automatically shown within the screen depicted in Figure 2.2. The following section describes the motivation for using optimization, the criteria used to select the optimization engine, and the optimizer capabilities.

5.1 RATIONALE FOR USING OPTIMIZATION

In the context of portfolio management, the potential analytical methods and strategies vary over a broad range that includes manual techniques, heuristics, and optimization. To help users make decisions and selections, a wide range of metrics (measures of effectiveness) can be calculated that provide indicators for using resources efficiently or maximizing objectives (e.g., expected return per dollar invested, person-weeks per net-project-impact).

Manual methods rely on the analyst to choose among selected options (e.g., investments or projects) while recognizing limitations in resources (e.g., monetary, human, equipment, time, material resources). The challenge in using manual methods is that as the number of possible activities, resource options, and/or resource constraints grows in size, the number of feasible combinations also grows, and it grows very rapidly. Therefore, manual selections are likely to produce significantly suboptimal results. Although these results may be feasible, it is likely that better solutions that would make much better use of resources would be available.

Heuristic methods can often improve upon manual selections, because metrics are systematically used as a guide for the selection process. By employing fast computational methods, heuristics can examine very large numbers of combinations of project selections and then select the choices that provide good measures of effectiveness (efficient use of resources to satisfy overall objectives). While heuristics routinely improve upon manual selection methods, they do not ensure the optimal use of resources.

Finding the true optimal solution is the focus of more formal optimization methods. As the number of candidate projects grows larger than a few , and as the number of constraints or restrictions grows beyond a small number, the ability to find the best solution by using manual or heuristic methods becomes exponentially more difficult. Through their evolution over time, many "solvers" have reached a point where they are effective and stable in solving large-scale problems containing hundreds or thousands (and even tens of thousands) of variables and constraints. Such large-scale problems far exceed the capability of manual methods and generally challenge heuristic methods when it comes to locating the best solutions.

To summarize, the benefits of using a formal optimization engine are to:

• Identify the true optimal (instead of merely reasonable or good) solutions,

- Maximize the use of resources (human, monetary, material, equipment, etc.), and

- Methodically examine alternate near-optimal solutions in sensitivity tests.

The optimization engine candidates and selection process for CIPDSS-PST are described below.

5.2 OPTIMIZER SELECTION CRITERIA

The following issues were considered during the optimizer selection process:

1. Cost and licensing requirements (expense and portability),

2. Problem-solving capabilities:

    – Zero-one LP and
    – Mixed-integer LP,

3. Problem size and speed

4. Ease of use and implementation:

    – Hands-off operations (robust feasible solution capability) and
    – Application interfaces (APIs),

5. Reliability (prior testing and verification),

6. Software documentation, and

7. Maintenance.

These seven categories formed the general basis for selecting the optimization engine. The selection criteria were applied informally and were intended to *guide* the process of selecting a new solver without requiring rigorous comparative measurements. Each consideration is briefly discussed in the following sections.

5.2.1 Cost and Licensing Requirements

Cost-of-purchase and licensing issues were a significant consideration in software selection; not so much from the standpoint of the one-time purchase expense but more with respect to reoccurring costs and licensing restrictions for each copy of CIPDSS-PST to be distributed. One-time purchase fees were considered acceptable as long as the expense was within reasonable limits. However, a primary goal was to develop an end-product that could be shared openly with multiple users across organizational boundaries. This consideration meant

that once the solver was obtained or licensed, it needed to be free of licensing restrictions for additional copies.

A fixed dollar amount (for one-time purchase) was not quantified during the review and screening process because solvers that required a purchase fee generally also required licensing fees for each add-on user or installation. Because the extent of distribution for CIPDSS-PST could not be anticipated at the time of development, the licensing criterion was treated as a need for unlimited distribution rights.

These considerations significantly limited the field of choices for selection. However, a number of public-domain solvers did meet and other selection criteria. Commercial software was not automatically eliminated from consideration, since it was thought that some licensing arrangements might permit a large number of copies to be distributed for a predetermined cost.

5.2.2 Problem-solving Capabilities

For portfolio optimization problems, it is important that a solver be able to explicitly treat projects as discrete, indivisible entities. Simple linear programming (LP) methods are not adequate because, although they can solve conventional problems, they cannot treat decision variables (x_i) as being discrete (i.e., only able to take on integer values 0, 1, 2, 3, …). For portfolio optimization, the user generally needs to treat projects as all-or-nothing candidates; each project is either chosen for inclusion or not included in the final selection (equates mathematically to a "0–1" choice). Other integer multiple representations (0, 1, 2, 3, …) can occur in conjunction with nondivisible resources, such as the number of discrete machines available to complete different tasks. These portfolio optimization problems contrast with conventional LP problems, which potentially allow *portions* of projects (fractional parts such as 0.35 of Project A and 0.77 of Project B) to be selected.

So although simple problems can sometimes be solved with the generalized LP formulation of the type:

$$\text{Maximize } f(x_i) \text{ where } f(x_i) = ax_1 + bx_2 + cx_3 + \ldots kx_n,$$

the CIPDSS-PST application requires a solver capable of the following additional requirement:

$$\text{and where some } x_i \text{ are integers } (0, 1, 2, 3, \ldots) \text{ or binary } (0, 1).$$

In general, CIPDSS-PST uses the binary restriction, but a solver that also offers integer or real value solutions was selected. Although the additional real and integer capabilities were expected to have limited use, the developers added them because they did not want to unnecessarily limit the modeling tool in case applications that required the additional flexibility arose. Solvers that can accommodate the integer and binary stipulations are typically referred to as mixed-integer LP solvers (i.e., MINT-LPs or MILPs).

5.2.3 Problem Size and Speed

The size and speed criteria were considered critical for practical reasons. It was anticipated that problems that might include dozens of objectives, hundreds of projects, and hundreds of constraints could occur. These could lead to formulations requiring thousands of variables. And, as noted above, the solver needed to handle integer and zero-one variables.

Some solvers, such as Excel's standard optimizer (spreadsheet implementation), satisfy the cost and licensing requirement but cannot handle sizeable problems and constraints or become very slow when they are looking for the solution to such problems.

5.2.4 Ease of Use and Implementation

Because CIPDSS-PST was developed for a mixed audience of potential users, the optimization engine needed to have a "hands-off" robustness and stability. Some optimization engines have sensitive internal parameter settings (e.g., search directions or step size) that require attention and monitoring in order to function efficiently and ensure optimal solutions. Although those types of solvers may be customized to solve specific types or sizes of problems faster than other more general algorithms, their requirement of constant attention to optimization parameters was considered unacceptable for this application.

In addition, some optimizers inherently interface more easily with other modeling components and software modules. Including various APIs in the CIPDSS-PST design was considered a significant plus to ensure that its development cost and time would be reasonable.

5.2.5 Prior Testing and Verification

It was considered highly important that users be confident in the functionality of the optimization engine for CIPDSS-PST. The candidate solvers considered included those that had been thoroughly tested, debugged, and verified against standard and nonstandard test problems.

5.2.6 Software Documentation

Up-to-date documentation of the optimizer software was considered an important criterion that would help in its implementation. Some solvers that might otherwise have satisfied the functional requirements do not have the detailed or useful documentation that would ensure that their integration could be completed successfully.

5.2.7 Maintenance

As a final consideration, optimizers were examined with regard to the amount of ongoing attention that is paid to maintenance issues and user feedback. Nearly all computer software can

be expected to encounter unanticipated bugs or odd behavior when it is subjected to rigorous and extensive testing — testing beyond the amount that might be feasible during the development process. Finding optimization engines that are routinely maintained for any newly discovered bugs or exceptions represents a very positive component of the selection process.

5.3 INITIAL OPTIMIZER CANDIDATES

In the initial screening process, a large number of optimizers were considered and reviewed for possible use in CIPDSS-PST. The list included:

- LINDO/LINGO (Schrage 1991),

- GAMS (Brooke et al. 1992),

- Excel embedded spreadsheet solver (Microsoft 2003),

- lp_solve (Linderoth and Ralphs 2005; Notebaert et al. 2008),

- COIN solvers/components (Linderoth and Ralphs 2005),

- CBC (Linderoth and Ralphs 2005), and

- Others (e.g., ABACUS, BCP, bonsaiG, GLPK, MINTO, and SYMPHONY) (Linderoth and Ralphs 2005).

The original candidates were assembled from in-house software packages, literature reviews, and web searches.

5.4 SELECTED SOLVER

After careful consideration, one optimization engine emerged as the best candidate for CIPDSS-PST: lp_solve. This MILP solver has been well tested, can efficiently solve the types and sizes of problems anticipated for portfolio optimization, and can be easily licensed as an integrated product within other software packages at no cost for each installation. lp_solve is flexible in terms of its integration with various programming environments, such as C++, Excel, and Java, among many others. Furthermore, lp_solve has been thoroughly documented and is currently maintained with regard to user-reported issues, fixes, and improvements.

Some options that had already been licensed and installed at Argonne and proven to be functionally viable (e.g., LINDO/LINGO and GAMS) ultimately had to be eliminated from consideration because of other issues, such as cost and licensing. To ensure portability to other organizations without the need for special licensing, these candidates were treated as being less favorable.

The Excel solver option, because it is a standard plug-in for anyone who already has a licensed copy of Excel, satisfied the licensing requirement. Excel could also be a feasible candidate because of its widespread use. However, it is limited in terms of the size of the problems it can tackle, and it has a number of functional parameters that require attention and adjustment to ensure optimality. Moreover, other issues, such as its poor speed, eliminated Excel from the final selection.

Collections of solvers, such as COIN and others in the original candidate list, are well-respected in the optimization community but have serious implementation issues for a hands-off application like CIPDSS-PST. Some of these solvers are modular, and orchestrating the various modules, setting proper values for internal parameters, and making adjustments to address the various types of problems require a significant amount of attention.

5.5 IMPLEMENTATION

lp_solve was selected for implementation, and it was successfully integrated into CIPDSS-PST with very few difficulties. Once installed, the optimizer allowed CIPDSS-PST designers to expand some of its planned features and capabilities. Included in its expanded capabilities are:

- *Access to problem reformulations* (manual edits to constraints and objective function). This feature gives an experienced user a limitless range of problem-customization options.

- *Access to diagnostics.* If an optimal solution to a problem cannot be located or a set of problem conditions that is infeasible to solve is encountered, this feature provides possible leads for rectifying the problem.

- *Ability to construct conditional constraints* (e.g., prerequisites or other inclusion and exclusion criteria).

Interactions with the lp_solve developers have been positive exchanges. When CIPDSS-PST was implemented, a subtle (nonfatal) program bug was discovered. It was addressed by the lp_solve maintenance team/community.

6 CONCLUSIONS

CIPDSS-PST provides a powerful and versatile tool for selecting, optimizing, and analyzing portfolios. The software facilitates the entire process of portfolio assessment, including constructing the fundamental evaluation objectives, evaluating individual projects, and analyzing the results. The software also facilitates comparing user-preferred selections with optimal project selections.

A wide range of constraints to account for various resource limitations and upper and lower limits on the number of projects to be included in the portfolio can be applied in CIPDSS-PST. Conditional constraints can also be introduced, so that the inclusion of one or more projects can force other projects to either be excluded or become mandatory. The optimized solution will then reflect these additional constraints in the sets of projects selected for the optimal portfolio.

On the basis of criteria that included portability, cost, licensing considerations, and problem size capability, lp_solve was chosen as the optimization engine. It performed very well in all test cases and continues to be maintained, updated, and verified by a large user community. It has the speed to solve large-scale problems and the analytical features to handle complex constraint specifications. It can be freely distributed as long as it is appropriately acknowledged in the application software.

The CIPDSS-PST software can be readily applied to other nonportfolio resource allocation problems. Its framework is very flexible, allowing the user to define problem objectives and a rating scale for each alternative (which is not necessarily a project) to be considered. With such versatility, the CIPDSS-PST software is a powerful tool applicable to a broad range of resource-constrained optimization problems.

During the tool design process, significant attention was devoted to simplifying data input and problem definition procedures. The simplification effort resulted in a single spreadsheet-like form for entering all of the critical problem definition information (objective definitions and project evaluations) and project characterizations. A second screen shows all the requirements for defining constraints and viewing outcomes. The end result is a compact interface that facilitates problem construction and analysis.

7 REFERENCES

Brooke, A., et al., 1992, *GAMS: A User's Guide*, The Scientific Press, South San Francisco, Calif.

Hammond, J.S., et al., 1998, *Smart Choices: A Practical Guide to Making Better Decisions,* Harvard Business School Press, Sept.

Jusko, M., et al., 2006, unpublished information, Argonne National Laboratory, Argonne, Ill., Apr.

Keeney, R.L., 1992, *Value-focused Thinking: A Path to Creative Decisionmaking,* Harvard University Press, Cambridge, Mass.

Keeney, R.L., and H. Raiffa, 1993, *Decisions with Multiple Objectives — Preferences and Value Tradeoffs,* Cambridge University Press, Cambridge and New York.

Linderoth, J.T., and T.K. Ralphs, 2005, *Noncommercial Software for Mixed-Integer Linear Programming*, Jan. revision.

Microsoft Corporation, 2003, *Microsoft Office Excel,* part of *Microsoft Office Professional Edition 2003*.

Notebaert, P., et al., 2008, *Introduction to lp_solve 5.5.0.12*, http://lpsolve.sourceforge.net/5.5/.

Schrage, L., 1991, *LINDO: An Optimization Modeling System*, The Scientific Press, South San Francisco, Calif.

Decision and Information Sciences Division
Argonne National Laboratory
9700 South Cass Avenue, Bldg. 900
Argonne, IL 60439-4867

www.anl.gov