VisSim/OptimizePRO

Version 3

Visual Solutions
INCORPORATED

Visual Solutions, Inc.

VisSim/OptimizePRO User's Guide - Version 3.0

Visual Solutions, Inc.
487 Groton Road
Westford, MA O1886

Windward Technologies, Inc.
12039 Mulholland
Meadows, TX 77477

Contents

Preface

This manual describes how to use VisSim/OptimizePRO for constrained optimization on VisSim simulations and models. VisSim/OptimizePRO was designed and implemented to take advantage of the VisSim environment. You can use all of the features of VisSim to create and organize your cost and constraint functions. VisSim dialog boxes make it easy to set upper and lower bounds on design variables. The Optimization Setup dialog box is used to select an optimization run.

Conventions

This manual assumes that you are already familiar with the VisSim graphical user interface. If you need to review the interface, consult your “VisSim User's Guide.”

The following typographical conventions are used to make this manual:

Visual convention	Where it’s used
SMALL CAPS	To indicate the names of keys on the keyboard.
ALT+F, R SHIFT+INSERT	In procedures, key sequences sometimes follow commands to indicate that the command can be executed from the keyboard. If the keys are separated by plus signs (+), hold down the keys at the same time. If the keys are separated by commas (,), press and release each key in the order shown.
ALL CAPS	To indicate directory names, file names, and acronyms.
Initial Caps	To indicate menu names and command names.

In addition, unless specifically stated otherwise, when you read “click the mouse...” or “click on...,” it means to click the left mouse button.

Technical support service

When you need assistance with a Visual Solutions product, first look in the manual and read the “readme” file. If you cannot find the answer, contact the Technical Support group via a toll call between 9:00 am and 6:00 pm Eastern Standard Time, Monday through Friday, excluding holidays. The phone number is **508-392-0100.**

If you do not have maintenance, you will be charged an additional fee after your second technical support phone call.

When you call in, please have the following information at hand:

- The version of VisSim and the version of the software operating environment that you're using
- The type of hardware that you're using
- All screen messages
- What you were doing when the problem happened
- How you tried to solve the problem

Visual Solutions also has the following fax and e-mail addresses:

Address/Number	What it’s for
508-692-3102	Fax number
bugs@vissol.com	Bug report*
doc@vissol.com	Documentation errors and suggestions
sales@vissol.com	Sales, pricing, and general information
suggest@vissol.com	Priduct suggestions
tech@vissol.com	Technical support

* Visual Solutions provides a standard bug report form in the online Release Notes named README.WRI.

References

1. Lasdon, L.S., Waren, A.D., Jain, A., and Ratner, M., “Design and Testing of a Generalized Reduced Gradient Code for Nonlinear Programming,” *ACM Transactions on Mathematical Software*, Vol. 4, No. 1, March 1978, pp. 34-50.
2. Lasdon, L.S. and Waren, A.D., “Generalized Reduced Gradient Software for Linearly and Nonlinearly Constrained Problems,” in “Design and Implementation of Optimization Software,” H. Greenberg, ed., Sijthoff and Noordhoff, pubs, 1979.

3. Abadie, J. and Carpentier, J. “Generalization of the Wolfe Reduced Gradient Method to the Case of Nonlinear Constraints,” in *Optimization*, R. Fletcher (ed.), Academic Press, London; 1969, pp. 37-47.
4. Murtagh, B.A. and Saunders, M.A. “Large-scale Linearly Constrained Optimization,” *Mathematical Programming*, Vol. 14, No. 1, January 1978, pp. 41-72.
5. Powell, M.J.D., “Restart Procedures for the Conjugate Gradient Method,” *Mathematical Programming*, Vol. 12, No. 2, April 1977, pp. 241-255.
6. Colville, A.R., “A Comparative Study of Nonlinear Programming Codes,” I.B.M. T.R. no. 320-2949 (1968).
7. Himmelblau, D.M., “Applied Nonlinear Programming,” McGraw-Hill Book Co., New York, 1972.
8. Fletcher, R., “A New Approach to Variable Metric Algorithms,” *Computer Journal*, Vol. 13, 1970, pp. 317-322.
9. Smith, S. and Lasdon, L.S., “Solving Large Sparse Nonlinear Programs Using GRG,” ORSA Journal on Computing, Vol. 4, No. 1, Winter 1992, pp. 1-15.
10. Luenbuerger, David G., “Linear and Nonlinear Programming,” Second Edition, Addison-Wesley, Reading, Massachusetts, 1984.
11. Visual Solutions, “VisSim User's Guide,” Visual Solutions, Inc., Westford, Massachusetts, 1995.

Chapter 1

Overview

Simulation design problems tend to have conflicting goals. Usually there is a desired ideal response that cannot be obtained exactly due to either modeling or physical constraints. The system response is determined by a few variables which may in themselves be bounded. Mathematically, such a situation is represented by:

Minimize or maximize $g(X)$,

subject to $glb_i \leq g_i(X) \leq gub_i$ for $i=1,...,m$,
$xlb_j \leq x_j \leq xub_j$ for $j=1,...,n$.

X is a vector on n variables, $x_1,...,x_n$, and the functions $g_1,...,g_m$ all depend on X.

The function to be minimized $g(X)$ is called the objective (or cost) function. Constraints are given by the functions $g_i(X)$. And, the decision variables $x_1,...,x_n$, may have bounds.

In VisSim, the analogous data structures are represented by the following blocks:

Block	Purpose
cost	objective or cost function
globalConstraint	constraint functions
parameterUnknown	decision variables

Upper and lower bounds are set within the globalConstraint Block for the constraint functions, and upper and lower bounds are set within the parameterUnknown block for the decision variables.

VisSim/OptimizePRO uses first partial derivatives of each function g_i with respect to each variable x_j. These are automatically computed by finite difference approximations. After an initial data entry segment, the program operates in two phases. If the initial values of the variables you supply do not satisfy all g_i constraints, a Phase I optimization is started. The Phase I objective function is the sum of the constraint violations. This optimization run terminates either with a message that the problem is infeasible or with a feasible solution. Beware if an infeasibility message is produced, because the program may have become stuck at a local minimum of the Phase I objective, and the problem may actually have feasible solutions.

Phase II begins with a feasible solution, either found by Phase I or with you providing a starting point if it is feasible, and attempts to optimize the objective function. At the conclusion of Phase II, a full optimization cycle has been completed and summary output is provided.

Setting up the problem

Before you can perform constrained optimization, your VisSim diagram must be set up to compute the values of the constraint functions and the objective function for given values of the variables. This is accomplished through the use of a cost block for the objective function and one globalConstraint block for each constraint.

VisSim/OptimizePRO can solve problems without constraints. In this case, there are no globalConstraint blocks.

Creating, organizing, and passing data to VisSim/OptimizePRO

The remaining data that describe the optimization problem must be created, organized, and passed to VisSim/OptimizePRO. The data consist of the number of variables, the lower and upper bounds on the variables, and the lower and upper bounds on the constraints.

You should include one parameterUnknown block in your VisSim diagram for each problem variable. VisSim/OptimizePRO sets the number of problem variables to the number of parameterUnknown blocks.

Bounds on the variables are set in the parameterUnknown block. These should be set and checked against bounds that are listed in the VisSim/OptimizePRO report, as described below. If you don't set bounds on a variable, the upper bound is set to 1.e30 and the lower bound is set to -1.e30. Likewise for constraints. These are set in globalConstraint blocks.

Generating a report file

VisSim/OptimizePRO produces a report file that documents the optimization run and lists the errors that are encountered. This report is written to file named VSMGRG2.TXT in the directory with your VisSim diagram.

To view the report while VisSim/OptimizePRO is running, use the monitor option described in Chapter 3, "Advanced Options." You can alternatively use any text editor to view the report after an optimization run is complete.

The report file is overwritten by each successive optimization run. To save a report for future viewing, save the file under a new name.

When VisSim/OptimizePRO terminates with `inform = 0` or `inform = 1`, the problem has been successfully solved. Other values of `inform` indicate an outcome that may not be successful.

Regardless of the outcome of the optimization run, the report should be reviewed because it contains a summary of starting values and final values for variables, constraints, and the objective function for successful runs. It also contains error messages and other information that can be helpful for runs that are not successful.

Controlling the behavior of VisSim/OptimizePRO

There are many parameters that control the behavior of the VisSim/OptimizePRO algorithm. Each parameter has a default value that is appropriate for most problems. You are not required to take any action in order to use the default parameter values; however, at times, it may be necessary to set one or more of the parameters to a new value in order to make VisSim/OptimizePRO more efficient or to make it possible to solve a difficult problem. For more information on setting parameters, see Chapter 3, “Advanced Options.”

Starting values of the variables

Starting values of the variables are important to the success of VisSim/OptimizePRO. The closer the starting values of the variables are to the final and optimal values, the less time it takes VisSim/OptimizePRO to make the optimization run. In particular, if you choose the initial values to satisfy the constraints, VisSim/OptimizePRO can skip the initial phase of finding variable settings that satisfy the constraint (that is, settings that are feasible).

When VisSim/OptimizePRO begins with infeasible starting values, it must use a Phase I optimization process. This means that VisSim/OptimizePRO uses the sum of the constraint violations as the objective function, temporarily ignoring the real objective function.

It is possible for VisSim/OptimizePRO to terminate at the end of Phase I without finding a feasible starting point. If you believe that the problem can be made feasible, try setting different starting values for the variables and try again. These remedies are described in Chapter 6, “Troubleshooting.”

Chapter 2

Using VisSim/OptimizePRO

This chapter describes the basics of using VisSim /OptimizePRO to solve optimization problems.

Solving a simple optimization problem with no constraints

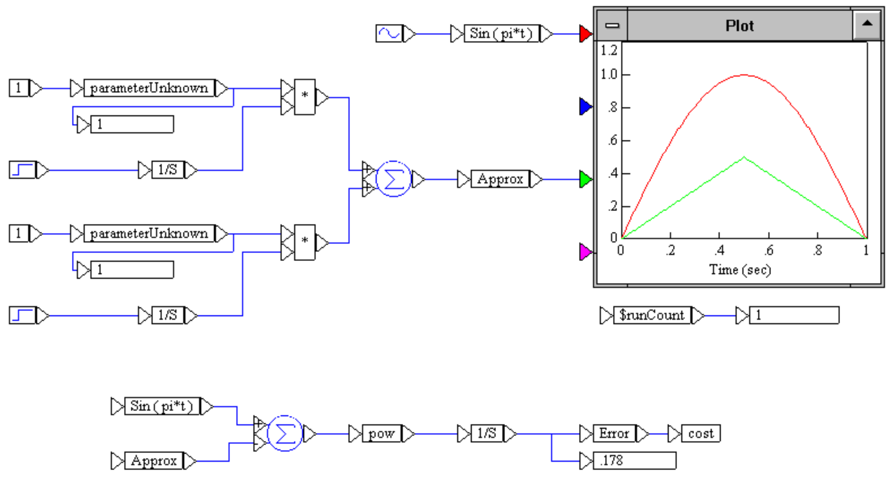
This example uses the diagram named CURV2P.VSM, which represents a simple two-parameter curve fitting application involving the approximation of the function Sin (πt) in the interval from 0 to 1. You will approximate this function by another function composed of two straight line segments. There are no constraints in this problem.

▶ **To open CURV2P.VSM**

1. Do one of the following:
   - From the toolbar, choose .
   - From the File menu, choose the Open command (ALT+F, O).
2. In the Directories box, select \VISSIM.
3. In the File Name box, select CURV2P.VSM.
4. Click on the OK button, or press ENTER.

Understanding the diagram

When you open CURV2P.VSM, the following diagram appears on the screen:



The function Sin (πt) is produced by a sinusoid block with frequency π and amplitude 1. It is wired into a variable block and identified as Sin (pi*t). The approximating function is produced with two step blocks and two integrator blocks. This function is wired into a variable block and identified as Approx. Both curves are plotted.

The cost or objective function is computed by integrating the squared difference of the two curves, (Sin (pi*t) - Approx)^2 , from 0 to 1, as shown in the diagram. The error is wired into the cost block to identify it as the objective function.

Each of the parameterUnknowns is wired to a const block with value 1. This provides starting values for the parameterUnknowns or decision variables. A simulation run plots the two curves and computes the error for the starting values as 0.178.

You want to find the best multipliers for the approximating function in order to produce the smallest error. To do so, the multipliers are wired to parameterUnknown blocks (which alerts VisSim that optimization may be performed on these decision variables) and then into a display block so that the parameters can be monitored during the optimization run. Upper and lower bounds of 10 and -10 respectively have been set for these parameters in the parameterUnknown blocks. To view or change the bounds, choose the Edit menu’s Setup Block command, then click the mouse over the parameterUnknown block.

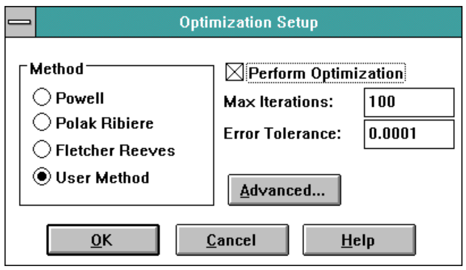
Setting the optimization parameters

To solve the optimization problem, you use the Optimization Setup command. Optimization Setup lets you select VisSim/OptimizePRO to perform the optimization.

▶ **To set the optimization parameters**

1. From the Simulate menu, choose the Optimization Setup command (ALT+S, O).

   The following dialog box appears:



2. Make the following selections:

   - Under the Methods box, activate User Method. When you activate this parameter, VisSim uses VisSim/OptimizePRO, by default, to perform the optimization.
   - Activate the Perform Optimization parameter.
   - In the Max Iterations box, enter 100. This parameter sets a limit on the number of simulation runs.
   - In the Error Tolerance box, enter 0.0001. This parameter instructs VisSim/OptimizePRO of the relative accuracy of the simulation runs. In this case, VisSim/OptimizePRO will find three digits of accuracy in the solution.

3. Click on the OK button.

Solving the optimization problem

To solve the optimization problem, you simulate the diagram

▶ **To solve the problem**

- From the Toolbar, click on the toolbar button to start the optimization.

After 28 simulation runs (reference $runCount in the VisSim diagram), the following changes have occurred:

- The cost block has changed from 0.178 to 5.82e-3
- The parameterUnknown, in the upper part of the diagram, has changed to 2.38
- The parameterUnknown block, in the lower part of the diagram, has changed to 2.29

In addition a report is written to VSMGRG2.TXT, which provides more information on the optimization process. This report is described in Chapter 4, "Reading the Report File."

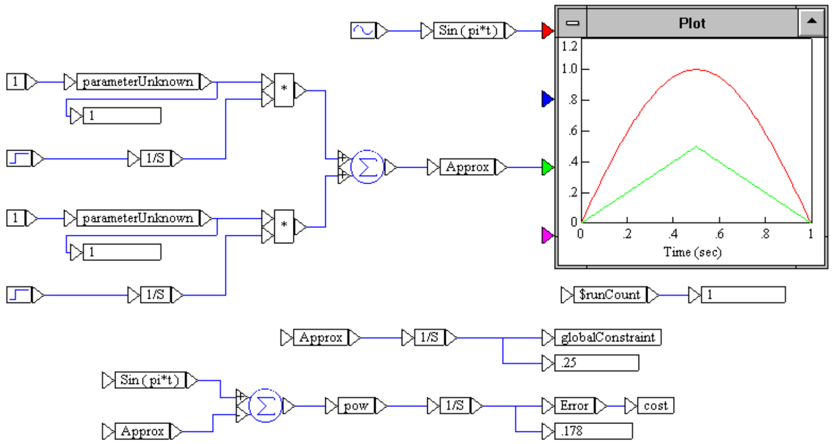
Solving an optimization problem with constraints

To solve a constrained optimization problem, you use globalConstraint blocks. These blocks identify constraints that depend on parameterUnknowns and are more complicated than the bound constraints (which are handled in the parameterUnknown block).

To illustrate the use of the globalConstraint block, the previous problem can be modified by constraining the area under the approximating function so that it cannot exceed 0.4.

▶ **To modify CURV2P.VSM**

1. Add the following blocks to the diagram:
   - From the Optimization category, add a globalConstraint block
   - From the Integration category, add an integrator block
   - From the Signal Consumer category, add a display block
2. Make a copy of the Approx block.
3. Wire the output of the Approx block into the integrator block.
4. Wire the output of the integrator block into the globabConstraint and display blocks.



Setting the constraints

The upper and lower bounds for the globalConstraint block are established in its dialog box. As is usual with VisSim you only have to assign these values once. After that, the upper and lower bounds are remembered in the same manner as other VisSim settings.

▶ **To set the upper and lower bounds**

1. From the Edit menu, choose the Setup Block command (ALT+E, S).
2. Click the mouse over the globalConstraint block.
3. In the Upper Bound box, enter 0.4.
4. In the Lower Bound box, enter 0.0.
5. Click on the OK button.

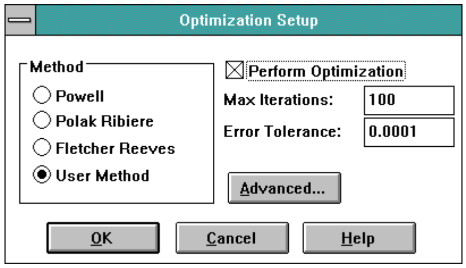
Setting the optimization parameters

The optimization parameters set in the previous example are valid for this example. If you skipped the previous example, you must set them now.

▶ **To set the optimization parameters**

1. From the Simulate menu, choose the Optimization Setup command (ALT+S, O).

   The following dialog box appears:



2. Make the following selections:

   - Under the Methods box, activate User Method. When you activate this parameter, VisSim uses VisSim/OptimizePRO, by default, to perform the optimization.
   - Activate the Perform Optimization parameter.
   - In the Max Iterations box, enter 100. This parameter sets a limit on the number of simulation runs.
   - In the Error Tolerance box, enter 0.0001. This parameter instructs VisSim/OptimizePRO of the relative accuracy of the simulation runs. In this case, VisSim/OptimizePRO will find three digits of accuracy in the solution.

3. Click on the OK button.

Performing a constrained optimization

To solve the constrained optimization problem, click on the  toolbar button.

This constrained optimization run yields the parameterUnknown values of 1.73 and 1.85 and the cost value of 6.21e-2. The constraint is at its upper bound of 0.4 as should be expected.

Note: The exact answer to the “analytic” problem posed here may differ from the computed answer. This discrepancy shows up because the integration methods are not exact. You can verify this by decreasing the integration step size in the dialog box for the Simulate menu’s Simulation Setup command and rerunning the simulation. The VisSim solution to this problem differs (due to numerical truncation errors) from the analytic solution. Taking smaller step sizes makes this relationship clear.

You now have the information necessary to apply this advanced optimization technology to your own problems. For more examples, see Appendix A, “Examples,” where a more complicated curve fitting example involving five parameters and a PID tuning example are presented.

Preparing diagrams for use with VisSim/OptimizePRO

In preparing your own VisSim diagrams for use with VisSim/OptimizePRO, it is important to keep in mind the following facts:

- Every time a parameterUnknown block or a globalConstraint block is pasted onto a VisSim diagram, the upper bound is set to 1.e30 and the lower bound is set to -1.e30. This means that unless you reset these bounds, the decision variable (or constraint, respectively) is unconstrained.
- The following conventions are used to set the upper and lower bounds:
  - 1.e30 represents infinity ($+\infty$)
  - -1.e30 represents minus infinity ($-\infty$)
- The table below shows how to set values of the upper and lower bounds:

Bound Type	Math	Lower bound	Upper bound
Free (no bounds)	$(-\infty, +\infty)$	-1.e30	1.e30
Lower bound only	$[a, +\infty)$	a	1.e30
Upper bound only	$(-\infty, a]$	-1.e30	a
Lower and upper bound	$[a, b]$	a	b
Equality	$[a, a]$	a	a

Chapter 3

Advanced Options

VisSim/OptimizePRO is based on a generalized reduced gradient algorithm. There are several parameters that control the behavior of this algorithm. Each parameter has a default value that is appropriate for most problems. You are not required to take any action in order to use the default parameter settings; however, at times, it may be necessary to set one or more of the parameters to a new value in order to make VisSim/OptimizePRO more efficient or make it possible to solve a difficult problem. This chapter describes how to change VisSim/OptimizePRO algorithmic parameters.

Accessing the Advanced options

The Advanced options are contained in the dialog box for the Optimization Setup command. When you click on Advanced, a dialog box showing all the VisSim/OptimizePRO parameters is displayed. You can change any value displayed. The new values remain in effect for the duration of the session. The File menu's Save and Save As commands save the advanced parameter settings with the VisSim diagram.

The following table describes the VisSim/OptimizePRO advanced parameters.

Parameter	Description	Default Value
`doscale`	Scaling. `0` No scaling. `1` The problem is scaled so that the maximum value of any row or column of the initial gradient array is less than or equal to 1.0.	`doscale = 0`

Parameter	Description	Default Value
`epinit`	To run the problem with `epnewt` initially set fairly large and then tighten at the end of the optimization, assign `epinit` the initial tolerance and `epnewt` the final one.	`epinit = Error Tolerance` (Optimization Setup)
`epnewt`	A constraint is assumed to be binding if it is within `epnewt` of one of its bounds. `epnewt` and `epinit` should be set together so that `epinit ≤ epnewt`.	`epnewt = Error Tolerance` (Optimization Setup)
`epskt`	The convergence criteria and requires that the K-T factor is ≤ `epskt`.	`epskt = 0.01`
`epspiv`	If, in constructing the basis inverse, the absolute value of a prospective pivot element is less than `epspiv`, the pivot is rejected and another pivot element is be sought.	`epspiv = Error Tolerance` (Optimization Setup)
`epstop`	This specifies the VisSim/OptimizePRO convergence criteria. If the fractional change in the objective function is less than `epstop` for `nstop` consecutive iterations, and if the K-T factor is ≤ `epskt`, the program accepts the current point as optimal. VisSim/OptimizePRO accepts the current point as optimal if the Kuhn-Tucker optimality conditions are satisfied to within `epstop`, that is, if the K-T factor is ≤ `epstop`.	`epstop = Error Tolerance*10` (Optimization Setup)

Parameter	Description	Default Value
`ipr`	Print level for VisSim/OptimizePRO report. `0` Print initial and final variable and function values `1` Print initial and final variable and function values plus one summary line for each one dimensional search. Values of `ipr > 1` and `≤ 6` are permitted, but require knowledge of the internal workings of VisSim/OptimizePRO and are not recommended for general use.	`ipr = 1`
`iquad`	Method for initial estimates of basic variables for each one dimensional search. `0` Tangent vectors and linear extrapolation `1` Quadratic extrapolation	`iquad = 0`
`itlim`	If the Newton procedure takes `itlim` iterations without converging, the iterations are stopped and corrective action taken.	`itlim = 10`
`limeval`	Limit on the number of simulation runs. `limeval=0` permits an unlimited number of simulation runs.	`limeval = Max Iterations` (Optimization Setup)
`limser`	If the number of completed one dimensional searches exceeds `limser`, VisSim/OptimizePRO terminates and returns `inform = 3`.	`limser = 10000`
`maximize`	The objective function is maximized if `maximize = 1`. The default is to minimize the objective function.	`maximize = 0`

Parameter	Description	Default Value
`monitor`	The report produced by VisSim/OptimizePRO is written to VSMGRG2.TXT located in the directory with the current VisSim diagram. Setting `monitor = 1` instructs VisSim/OptimizePRO to display the report while the optimization run is being performed. The monitor option provides a convenient way to keep track of long optimization runs. The monitor displays the VisSim/OptimizePRO report in a window with menu items that can be used to save the report in a file for future reference.	`monitor = 0`
`nstop`	If the fractional change in the objective function is less than `epstop` for `nstop` consecutive iterations, VisSim/OptimizePRO accepts the current point as optimal.	`nstop = 3`
`ph1eps`	If `ph1eps` is nonzero, the phase 1 objective is augmented by a multiple of the true objective. The multiple is selected so that, at the initial point, the ratio of the true objective and the sum of the infeasibilities is `ph1eps`. Setting `ph1eps = 0.0` produces the most efficient way to reach a feasible point (a point where all constraints are satisfied). Setting `ph1eps > 0.0` causes VisSim/OptimizePRO to reach feasibility without ignoring the objective function.	`ph1eps = 0.0`
`pstep`	This is the step size used for estimating partial derivatives of functions with respect to the variables.	`pstep = Error Tolerance` (Optimization Setup)

Chapter 4

Reading the Report File

This chapter describes the report file produced by VisSim/OptimizePRO for each optimization run. It also describes the termination messages VisSim/OptimizePRO produces that describe the outcome of the optimization process.

Report file

The report file summarizes the problem starting values, optimization process, and final results of an optimization run. By default, the report is written to VSMGRG2.TXT in the directory containing the current VisSim diagram. If you selected the monitor option, the report is displayed in a window during the optimization run.

The report file is divided into five sections: Problem Description, Starting Values, Solution Process, Final Results, and Summary. Most of the information in the report is self-explanatory, but some of the terms require an understanding of the Generalized Reduced Gradient (GRG) algorithm. For more information on the GRG algorithm, refer to the book titled, “Linear and Nonlinear Programming.”

Problem Description section

The Problem Description section displays the current versions of VisSim/OptimizePRO and GRG2, the specific GRG algorithm on which VisSim/OptimizePRO is based, along with the date and time of the run.

The number of variables and the number of functions (the sum of the number of constraints and objective function) are displayed, along with an indication of whether the problem is one of minimization or maximization.

When you change algorithmic parameters, they are also listed here.

The information in this section should confirm that the problem has been set up correctly.

```
GRG2 95.10 Report: Date Mon Oct 02 10:00:00 1995

Problem Title:   VisSim Version 2.5, Optimization by VisSim /
OptimizePRO


Problem Parameters:


Number of variables is  2

Number of functions is  2

Objective function will be MINimized
```

Starting Values section

The Starting Values section lists the starting values for the problem variables and the computed values of the constraint functions and the objective function. The functions (constraints) table shows the values computed using the starting values of the variables. The notation used in the `Status` column, for variables and constraints, and in the `Type` column for constraints, is described in the following table.

Status	Description	Type	Description
UL	Constraint or variable is at its upper bound	EQ	Equality constraint
LL	Constraint or variable is at its lower bound	LE	Upper bound constraint
EQ	Equality constraint	GE	Lower bound constraint.
****	Constraint is violated, value is less than lower bound or greater than upper bound	RNGE	Lower and upper bound constraint
FREE	Variable has no lower bound and no upper bound	OBJ	Objective function
FX	Fixed variable, lower bound equals upper bound	NA	Constraint function ignored because it has no lower bound and no upper bound

```
Starting Values


Functions:
        Function                      Initial   Lower     Upper
 No.    Name        Status      Type  Value     Bound     Bound
 ___    _________   ______      ____  ________  ________  ______
    1       G                    OBJ  0.178051
    2       G                   RNGE  0.249988         0      0.4


Variables:
        Variable                      Initial   Lower     Upper
 No.    Name        Status            Value     Bound     Bound
 ___    _________   ______            ________  ________  ______
    1       X                                1       -10       10
    2       X                                1       -10       10
```

Solution Process section

The solution process is iterative. It begins at the starting values and ends at the final values. At each iteration, VisSim/OptimizePRO attempts to improve the objective function. If the problem is not feasible, Phase I may result in an objective function that is worse than the starting value, but this is necessary in order to achieve feasibility. The iteration log shows the iteration number in column 1. Information in columns 2 to 9 is described in the following table.

Column	Heading	Description
1	`Itn No.`	Iteration number.
2	`Objective Function`	Value of the objective function. If some constraints are not feasible (see column 5), the objective function value is the sum of the constraint violations.
3	`Binding Constrs`	Number of constraints that are at either their lower or upper bound. The binding constraints provide a set of nonlinear equations that can be used to solve some of the variables in terms of other variables. See also `Super Basics`.
4	`Super Basics`	Independent variables of the reduced problem. Variables are classified by VisSim/OptimizePRO as basic, nonbasic, and superbasic, as described later under “Final Results section.”
5	`Infeas Constr`	Number of infeasible constraints; that is, the constraints that violate their bounds for the current values of the variables.

Column	Heading	Description
6	`Norm of Red. Grad`	Norm of the reduced gradient; also referred to as the Kuhn-Tucker factor or the K-T factor. This number is the maximum absolute value of gradient of the reduced objective function with respect to the set of superbasic variables scaled by the value of the variable and the inverse of the value of the objective function. When the value of the K-T factor is less than `epstop` (the stopping criteria), VisSim/OptimizePRO terminates the iteration and returns the current variable values.
7	`Hessian Cond. No.`	Condition number of the Hessian. The Hessian is the matrix of second derivatives of the objective function with respect to the variables. A Large condition number indicates an ill-conditioned problem that may be difficult to solve accurately.
8	`Step Size`	Step size for the current iteration.
9	`Degen Step`	A “T” in this column indicates a degenerate step (iteration); otherwise this column is blank. When VisSim/OptimizePRO reclassifies variables from basic to superbasic and from superbasic to basic, the objective function does not change and the iteration is flagged as degenerate to avoid the appearance of convergence.

```
Itn   Objective   Binding   Super   Infeas   Norm of    Hessian   Step   Degen
No.   Function    Constrs   Basics  Constr   Red.Grad   Cond.No   Size   Step
                                                        .
___   _________   _______   ______  ______   ________   _______   ____   _____
                  _
  0    0.178051      0        2       0          0.39         1      1
  1    0.100372      1        1       0         0.081         1    0.2
  2   0.0621484      1        1       0       0.00011         1   0.51
```

Final Results section

The Final Results section lists the final values for the problem variables and the computed values of the constraint functions and the objective function. The functions table shows the values of the constraint functions computed using the final values of the variables.

The GRG algorithm uses the binding constraints to solve for some variables, classified as “basic,” in terms of the remaining variables, thus reducing the number of independent variables. The variables that are not basic are classified as “nonbasic,” if they are at one of their bounds and “superbasic,” otherwise. The classification is given in the `Status` column of the Final Results table. The reduced objective function is the objective function treated as a function of the nonbasic and superbasic variables. The gradient of the reduced objective function is called the reduced gradient.

The `Reduced Gradient` column gives the value of the reduced gradient for each nonbasic and each superbasic variable. Reduced gradient values for basic variables are zero, by definition. Since nonbasic variables are at one of their bounds, the reduced gradient, with respect to a nonbasic variable, should have the "right" sign. For example, when the problem is a minimization and the nonbasic variable is at its lower bound, the reduced gradient should be greater than or equal to zero. This is an indication that a small increase in the value of the variable results in an increase in the value of the objective (that is, a move away from the optimum).

The reduced gradient of a nonbasic variable that is at its upper bound should be less than or equal to zero. Reduced gradient values are affected by the scale of the variable and the scale of the objective function. This is why VisSim/OptimizePRO scales these reduced gradient values by the value of the variable and by the reciprocal of the value of the objective function and then computes the K-T factor as the maximum over all the superbasics. The K-T factor is printed in the solution process section of the report in the `Norm of Red.Grad` column. A small K-T factor is a good indication that a local optimum has been located. If this value is less than or equal to `epstop`, the stopping criteria has been satisfied and VisSim/OptimizePRO terminates with `inform = 0`. VisSim/OptimizePRO also terminates, with `inform = 1`, when the objective function converges to a relative error of `epstop` for `nstop` consecutive iterations.

Final Results

Functions:

No.	Name	Initial Value	Final Value	Status	Distance from Nearest Bound		Lagrange Multiplier
1	G	0.17805	0.62148	Objective			
2	G	0.24999	0.4	UpperBnd	6.43e-009	:U	-0.51091

Variables:

No.	Name	Initial Value	Final Value	Status	Distance from Nearest Bound		Reduced Gradient
1	X	1	1.7278	Basic	8.272	:U	
2	X	1	1.8531	SupBasic	8.147	:U	0.000114

Summary section

The Summary section of the VisSim/OptimizePRO report provides a brief summary of the run. The minimized or maximized objective function value is given along with the VisSim/OptimizePRO termination message. The messages are defined later in this chapter, under "Termination messages `(inform)`."

The number of simulation runs made for evaluating constraint functions and the objective function is also given as "number of function evaluations." Finally, the summary section displays the time used by the run.

The termination message provides valuable information on the final results returned by VisSim/OptimizePRO. The message that follows `inform = 0` gives the K-T factor that has been described above. This is the best indication that VisSim/OptimizePRO has been successful in locating a local optimum of the problem to the desired accuracy. When the termination message is `inform = 1`, VisSim/OptimizePRO also gives the K-T factor, which may be small enough to accept the final results. If you do not think the K-T factor is small enough, you can restart VisSim/OptimizePRO from the final results, but with a smaller value of `epstop`. This process usually results in a smaller K-T factor. If it does not, it may indicate that a better result is not possible (due to inaccurate function values).

```
                                Summary

MINimized objective function value is 0.0621484
Termination: INFORM = 0
       Kuhn-Tucker conditions are satisfied to
       within  0.00011 for the current variable values.
       Relative change in the objective function value
       is   0.038 for the last iteration.
Number of function evaluations 19
Time used is 53.826 seconds.
```

Termination messages (`inform`)

VisSim/OptimizePRO returns the outcome of the optimization process through the value of `inform` in the Summary section of the report. All possible values of `inform` are described below. The table below describes all the values of inform. Refer to this table and to Chapter 6, "Troubleshooting," when the results are not what you expected.

Inform	Meaning
0	Kuhn-Tucker conditions satisfied. This is the best possible indicator that an optimal point has been found.
1	Fractional change in objective less than `epstop` for `nstop` consecutive iterations. This is not as good as `inform = 0`, but still indicates the likelihood that an optimal point has been found.
2	All remedies have failed to find a better point. You should check cost and constraint functions and bounds for consistency and, perhaps, try other starting values.
3	Number of completed one-dimensional searches exceeded `limser`. You should check cost and constraint functions and bounds for consistency and, perhaps, try other starting values.
4	Objective function is unbounded. VisSim/OptimizePRO has observed dramatic change in the objective function over several steps. This is a good indication that the objective function is unbounded. If this is not the case, you should check the cost and constraint functions and bounds for consistency.

Inform	Meaning
5	Feasible point not found. VisSim/OptimizePRO was not able to find a feasible point. If the problem is believed to be feasible, you should check the cost and constraint functions and bounds for consistency and, perhaps, try other starting values.
6	Degeneracy has been encountered. The point returned may be close to optimal. You should check the cost and constraint functions and bounds for consistency and, perhaps, try other starting values.
7	Noisy and non-smooth function values. Possible singularity or errors in the cost and constraint function evaluations. The solution returned may be as accurate as possible with the current simulation set-up. It may be necessary to increase the accuracy of the cost and constraint functions by reducing the step size set in the Simulation Setup dialog box.
8	The optimization process was terminated by a user request through the Simulation menu Stop command.
9	Maximum number of simulation runs exceeded. The number of simulation runs is set under Max Iterations in the Optimization Setup dialog box. Set Max Iterations to 0 if you want VisSim/OptimizePRO to continue to run until it converges to a solution or until you explicitly stop it by choosing the Stop command from the Simulate menu.
-1	Fatal Error. Some condition, such as number of variables ≤ 0, was encountered. VisSim/OptimizePRO documented the condition in the report and terminated. In this case, you need to correct the VisSim diagram and rerun VisSim/OptimizePRO.

Messages `inform = 0`, `inform = 1`, and `inform = 2` are by far the most common. Message `inform = 0` implies the highest level of confidence that at least a local optimum has been found; message `inform = 1` implies less confidence; and message `inform = 2` even less.

In message `inform = 0`, the Kuhn-Tucker conditions are first-order necessary conditions that hold if the current point is at least a local optimum and all functions have continuous first partial derivatives. For further explanation, refer to “Design and Testing of a Generalized Reduced Gradient Code for Nonlinear Programming,” “Generalized Reduced Gradient Software for Linearly and Nonlinearly Constrained Problems,” and “Applied Nonlinear Programming,” as described under “References” in the Preface.

In message `inform = 2`, the following sequence of events has occurred:

1. No improved point was located along the last search direction.
2. Change of basis was attempted (if one had not already been done).
3. If the search direction was not the negative reduced gradient, this direction is tried.

4. If any variables with values at a bound have reduced gradient components indicating that releasing them from that bound could improve the objective, one such variable is allowed to leave its bound.

In other words, VisSim/OptimizePRO tried all known remedies, and none of these remedies improved the objective function, so the program terminated.

Regardless of which of termination messages `inform = 0`, `1`, and `2` is returned, the current point may be (nearly) optimal. Message `inform = 0` may fail to appear because the variables or constraints of the problem are poorly scaled; see Chapter 6, "Troubleshooting."

Message `inform = 5` is returned when Phase I terminates and the final point is not feasible. In this case, there may be no point satisfying all problem constraints. If this is not the believed to be the case, the user can select new starting values for the parameterUnknowns and rerun.

If you are unsatisfied with the solution found by VisSim/OptimizePRO, see Chapter 6, "Troubleshooting."

Chapter 5

Tolerances and Algorithmic Options

This chapter describes the changes to tolerances and algorithmic options that may improve performance of VisSim/OptimizePRO. All tolerances and algorithms are set through Advanced Options in the dialog box for the Optimization Setup command, as described in Chapter 3, "Advanced Options."

Tolerances

The default parameter settings are appropriate for most problems. Several of the numerical tolerances are based on the value of Error Tolerance which is set in the Optimization Setup dialog box. This value should not be greater than 1.0e-2. If it is, VisSim/OptimizePRO uses 1.0e-2 in its place to set the other tolerances. The Error Tolerance should reflect the accuracy of the VisSim simulation computations and are based on the step size and other VisSim settings.

`epnewt` tolerance

The most critical tolerance is `epnewt`. Increasing it can sometimes speed convergence (by requiring fewer Newton iterations) while decreasing it occasionally yields a more accurate solution or gets the iterations moving if the algorithm gets "stuck." Values larger than `1.0e-2` should be treated cautiously, as should values smaller than `1.0e-6`.

`epinit` tolerance

Choosing a value for `epinit` different from `epnewt` has helped solve a few problems that were not solved otherwise. Suggested values are `epinit =1.0e-4`, `epnewt =1.0e-6`.

`epstop` tolerance

Choosing a smaller value for `epstop` usually improves the accuracy of the final solution.

`epspiv` tolerance

If the problem is degenerate and this is slowing computations, choosing a larger value for `epspiv` may help by allowing pivots on elements that were previously rejected. If convergence of the iterations is a problem, reducing `epspiv` and/or increasing `epnewt` may help.

`ephlep` tolerance

A nonzero value of `ephlep` causes Phase I to consider the true objective along with the sum of infeasibilities and may yield a better point at the end of Phase I.

Algorithmic Options

This section briefly describes the effects of changing the `kderiv` and `iquad` algorithms.

`kderiv` option

The central differences, `kderiv = 1`, are more accurate than forward differences. Central differences are exact for quadratic functions, while forward differences are exact only for linear functions. However, because central differences require two function evaluations per derivative — while forward differences require only one — selecting `kderiv = 1` may double your computing time.

For more information on forward differences and central differences, see “Inaccurate numerical derivatives,” in Chapter 6, “Troubleshooting.”

`iquad` option

Quadratic extrapolation can often speed computations by providing better initial values for the iterations. This option is selected by `iquad = 1`. It is unnecessary if all constraints are linear.

Chapter 6

Troubleshooting

It is possible for VisSim/OptimizePRO to terminate at a point that is not optimal. If *X* is the vector of n variables when VisSim/OptimizePRO terminates (final values of the parameterUnknowns), then one of the following conditions must hold:

- *X* closely approximates a global optimum
- *X* closely approximates a local optimum, but not a global optimum
- *X* does not closely approximate even a local optimum

These statements apply in both Phase I and Phase II optimization.

This chapter explains when each condition holds true. It also describes what to do if *X* is not a global optimum.

Errors in the problem set-up

To aid in spotting errors in the input data and to see if the cost function and constraint functions are being computed correctly, all starting values are written to the report file at the start of a run. If a problem is being solved for the first time, it may be useful to make a run which prints the input data and then stops.

▶ **To print input data and stop**

1. From the Simulate menu, choose the Optimization Setup command (ALT+S, O).
2. In the Max Iterations box, enter 1.
3. Activate the Perform Optimization parameter.
4. Click on the OK button.
5. From the toolbar, click on the toolbar button.

The report file, VSMGRG2.TXT, can now be viewed and verified.

Inaccurate numerical derivatives

VisSim/OptimizePRO uses finite difference approximations to compute derivatives of the problem functions. These are computed by VisSim/OptimizePRO routines using forward differences or using central differences. Use of forward differences is the default option. To invoke central differences, use the `kderiv` option, as described in Chapter 5, “Tolerances and Algorithmic Options.”

Finite difference derivatives have roughly half the precision of the constraint and cost functions. Hence, if each function is accurate to five or six significant digits, then the derivatives have about two or three significant digits, which is adequate in most instances. However, if each function has four or fewer significant digits, then the derivatives have two or less, which may seriously hinder the optimizer.

Low precision often occurs when numerical routines are used to evaluate one or more of the functions. This can occur in chemical process models where recycle loops require iterative solution of a system of implicit nonlinear equations, or when differential equations are solved numerically. Then, the accuracy of these numerical calculations determines the accuracy of the functions.

Inaccurate derivatives can cause VisSim/OptimizePRO to terminate at a non-optimal point, often with the message `inform = 2`. (See “Termination messages (inform)” in Chapter 4, “Reading the Report File,” for an explanation of this message.) Possible remedies include:

- Increasing the accuracy of the functions computed (by decreasing the simulation step size)
- Using central differences (by specifying the `kderiv` option)
- Trying different step sizes (via the `pstep` option)

Scaling

Proper scaling of both variables and problem functions is very important for successful operation of VisSim/OptimizePRO. VisSim/OptimizePRO uses a relative measure to compare quantities. This means that VisSim/OptimizePRO can handle functions that vary by several orders of magnitude and still perform well. The relative measure is changed to an absolute measure when comparing quantities that are less than one in absolute value. If the cost function or a constraint function operates in a range that is much less than one, these functions should be scaled to bring the values up to one.

Poor scaling can be the cause of inaccurate function values. If terms or factors of the problem functions vary by several orders of magnitude, the result of combining these terms or factors through floating point operations can produce an inaccurate result.

If scaling problems are suspected of causing difficulties, variables should be scaled so that a unit change (changes of 1.0) represents a small but significant change in that variable. In addition, it is advisable to avoid having the constraint or objective functions much more sensitive to some variables than others. A symptom of bad scaling is the presence of very large derivative values. The `doscale` option provides one type of scaling that is based on gradient values.

Bounds that cannot be violated

It is possible for VisSim/OptimizePRO to require simulations with some variable slightly outside of its bounds. If this is likely to cause underflow or overflow or an error termination (such as, attempting to take the square root or log of a negative number), preventive action must be taken in VisSim. If, for example, the problem variables all have positive lower bounds and there is a division by some of the variables, a test should be set up at the start of simulation comparing the variable with a small positive number less than the lower bounds. If any variable is less than this value, set all the affected functions to an arbitrarily large number such as `1.e30` and return. This causes the step size to be cut back.

Local and global optima

Neither VisSim/OptimizePRO nor any other nonlinear optimization package can guarantee finding a global optimum in cases where there are distinct local optima. If you know that your problem is convex (that is, minimizing a convex objective over a convex feasible region), then any local optimum is global, so this problem cannot occur. Otherwise, based on your knowledge of the problem, you must try a variety of starting points to determine the local/global issue. If all starting points yield approximately the same final point, and that point satisfies what you know about the problem, then you can be fairly confident that the point is globally optimal.

Solutions that are not accurate enough

There are several methods for assessing the accuracy of a solution:

- Use knowledge of the problem
- Vary one or more variables and observe the behavior of the objective and constraint functions
- Try different starting points
- Observe initial and final magnitudes of the reduced gradients of the superbasic and nonbasic variables

With respect to the second method, some of the nonbasic variables can be fixed at their current values (set upper bound = lower bound = current value for the affected parameterUnknowns) and others can be varied using VisSim/OptimizePRO. If this does not produce a significantly improved feasible point, confidence in the current solution is increased. Confidence is also increased if different starting points lead to nearly the same final point.

With respect to the fourth method, the best indication of accuracy occurs if the Kuhn-Tucker conditions are satisfied (`inform = 0`). However, if they are not, but the reduced gradient components of superbasic variables have been reduced from their initial values, by say three orders of magnitude or more, while reduced gradients of nonbasic variables at bound have the correct sign, this is symptomatic of reasonably high accuracy.

If perceived accuracy is not sufficient, reducing `epstop` and/or increasing `nstop` often helps. It is also often effective to reduce `epnewt` if `epstop` is reduced, since this increases the accuracy of the reduced gradient computation.

For information on additional parameter changes and option choices that may be helpful, see Chapter 5, “Tolerances and Algorithmic Options.”

Appendix A

Examples

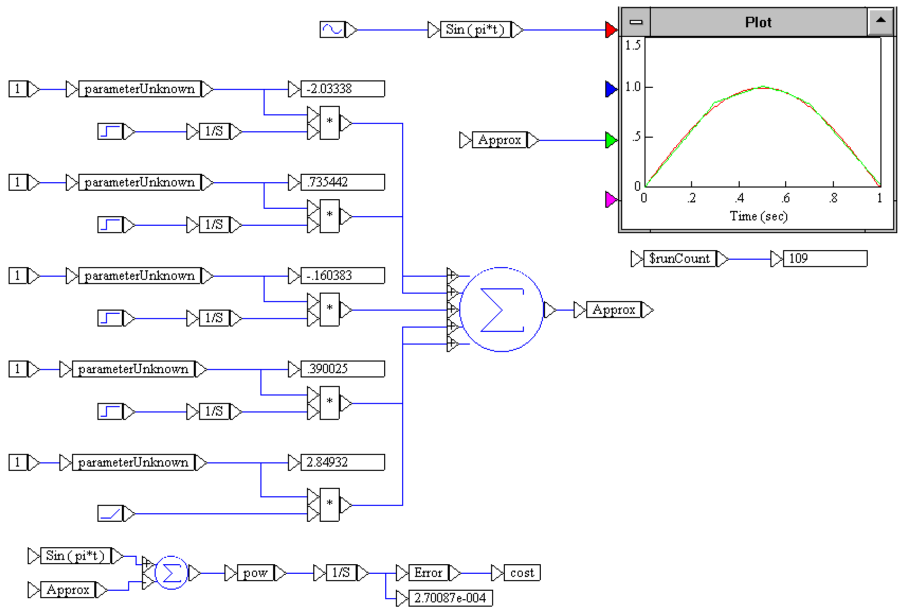
This chapter uses two examples to further illustrate the power and functionality of VisSim/OptimizePRO. These examples build on the material presented in Chapter 2, "Using VisSim OptimizePRO."

Fitting a function with five parameters — CURV5P.VSM

CURV5P.VSM is a continuation of the CURV2P.VSM, which is described in "Solving a simple optimization problem with no constraints," in Chapter 2, "Using VisSim/OptimizePRO."

CURV5P.VSM approximates the function Sin (πt) on the interval 0 to 1. Five line segments are used in this diagram to get a better fit than what was gotten in CURV2P.VSM with only two parameters.

This gives a problem with five variables. As before, the objective function is the integral of the squared error between the two curves. Starting with all five parameterUnknowns set to 1, the starting value of the objective function is 0.14. VisSim/OptimizePRO converges after 109 simulation runs with the minimized value of the objective function at 0.00027.



PID tuning problem with three parameters — PIDTUNEZ.VSM

PIDTUNEZ.VSM, an adaptation PID controller tuning diagram (PIDTUNE.VSM) distributed with the VisSim base product, describes a PID tuning problem with three parameters and one constraint.

The PID controller has three decision variables:

- A proportional gain (*pg*)
- An integral gain (*ig*)
- A derivative gain (*dg*)

Once these variables are set, a simulation run produces the resulting output signal (*s*). Assuming that a desired response (*r*) is postulated and that the functions *r* and *s* are defined on the interval [0, .1], one measure of the fidelity of *s* to *r* is to compute the integrated squared error between these two functions. This integral is the cost function. Notice that this function depends on *s*, which depends on both the input signal (which is assumed to be given and fixed) and the PID variables. Thus, choosing the decision variables optimally would be equivalent to choosing the *pg*, *ig*, and the *dg* to minimize the integrated squared error between the response and the resulting output signal.

Taking this example one step further, it is easy to see that certain values for the gains are not reasonable. Thus, you can restrict the decision variables by:

$10 \le pg \le 1000$

$200 \le ig \le 200$

$0 \le dg \le 100$

You set these upper and lower bounds in the Setup dialog box for the parameterUnknown blocks. Notice that the integral gain is fixed at 200.

Finally, to prevent the output signal (s) from overshooting the upper value of $r(t)$ by a certain amount, you can set the globalConstraint as:

$\max \{m(t) : 0 \le t \le .1\} \le .35$

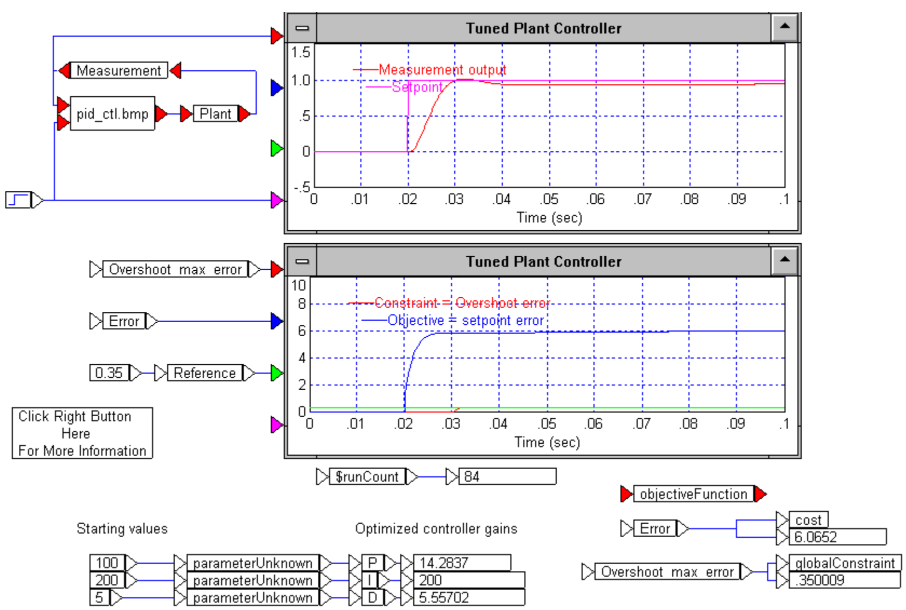
where

$m(t) := 20\,(s(t) - r(t))$

In this problem, this has the effect that $s(t) \le 1+(.35)/20$.

If you start at $pg = 100$, $ig = 200$, and $dg = 5$, the value of the objective function is 6.02. The overshoot error is 1.41, which exceeds the maximum allowed overshoot of 0.35. Thus, the problem is infeasible at the starting point. VisSim/OptimizePRO converged after 84 simulation runs with the objective function at 6.06 and the overshoot error at its upper limit of 0.35.

The PIDTUNEZ.VSM diagram, on the following page, shows the final and optimal values of the gains.



Appendix B

Installing VisSim/OptimizePRO

The Install program that comes on your VisSim/OptimizePRO disk installs the VisSim/OptimizePRO program and other utility files on the Windows 95, Windows 98,and Windows NT4.0+ platforms. You use the Install program to:

- Install VisSim/OptimizePRO on your computer for the first time
- Upgrade your existing copy of VisSim/OptimizePRO to a more recent version of the software

Installation Requirements

VisSim/OptimizePRO runs on personal computers using the Intel 80286 or higher processor, including the IBM Personal System/2 Series, the IBM PC AT, and 100% compatibles.

To install VisSim/OptimizePRO, your computer must have the following components:

- VisSim 3.0+
- MB RAM
- KB free hard disk space
- 3 1/2" floppy drive
- EGA or higher resolution monitor

Installation Procedure

You use the Install program to install VisSim/OptimizePRO on your computer for the first time or to upgrade your existing copy of VisSim/OptimizePRO to a more recent version of the software.

When you upgrade VisSim/OptimizePRO, the installation program replaces the old program and utility files with new ones. If there are existing files that you want to retain, Install gives you the opportunity to specify the files not to be overwritten.

▶ To install or upgrade VisSim/OptimizePRO

This procedure assumes that you are installing from drive A to your hard disk. If you are installing from a different drive, substitute the correct drive designation in the installation procedure.

1. Start Microsoft Windows..
2. Insert the VisSim/OptimizePRO disk into drive A.
3. Do one of the following:
   - From the Start menu, choose the Run command
   - In the Command Line box, type A:\INSTALL.
   - Click on the Open button.
4. In the Command Line box, type A:INSTALL and click on the OK button, or press ENTER.
5. An Install dialog box appears.

   Install asks you where you want to install VisSim/OptimizePRO. You can accept the default path (C:\VISSIM) or type in a different directory. Make sure that the VisSim/OptimizePRO files are installed on the same disk and directory that contain your VISSIM.EXE.

   If you choose to install VisSim/OptimizePRO over an earlier version, Install replaces all VisSim files (.VSM files) with new ones. You can control which existing VisSim files are overwritten as a result of the installation by activating the Ask Before Overwriting Existing Files check box. An X in the Ask Before OverWriting Existing Files check box activates this option.
6. To accept the information in the dialog box, click on the Continue button, or press ENTER.

   Install displays a dialog box that shows the progress of the installation. When the installation is complete, VisSim displays a final dialog box.
7. Click on the OK button, or press ENTER.

Index

R

T

U

V