Chapter 4
Methods and Models of Optimization

Most engineering problems, including planning, control and design, have more than one solution. The theory of optimization provides a mathematical basis for establishing the acceptability conditions that outline the class of acceptable solutions, for the definition of the criterion that provides the measure of goodness of every individual solution, and the optimization procedure (algorithm) that results in finding the optimal solution, i.e. the solution maximizing the value of the goodness criterion. These three components, the *class of acceptable solutions*, the *criterion of goodness*, and the *optimization procedure* are to be present in any definition of the optimization problem.

The solution vector of the optimization problem is a set of particular numerical values of some optimization variables, $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_\mathrm{n}]^\mathrm{T}$ that represent the nature of the problem. For example, in a resource distribution problem when some material, monetary, or energy resources have to be distributed between n consumers, vector $\boldsymbol{X}$ represents the amounts of these resources designated to each consumer.

The class of acceptable solutions is typically defined as a set of conditions, equations and/or inequalities that the solution vector must satisfy. Thus in the resource distribution problem these conditions include the requirements that the amounts of the resource designated to individual consumers cannot be negative, i.e.

$$\boldsymbol{x}_\mathrm{i} \geq 0,\ \mathrm{i} = 1, 2, \ldots, \mathrm{n}$$

that the sum of the amounts of this resource designated to consumers shall not exceed the total amount available (i = 1,2,.. is the consumer index), i.e.

$$\sum_{\mathrm{i}=1}^{\mathrm{n}} x_\mathrm{i} \ \leq \boldsymbol{P}^{\mathrm{TOT}}$$

V.A. Skormin, *Introduction to Process Control*, Springer Texts
in Business and Economics, DOI 10.1007/978-3-319-42258-9_4

that the amounts of the resources provided to some of the consumers are not negotiable (k is the consumer index), i.e.

$$\boldsymbol{x}_{\mathrm{k}} = P^{\mathrm{K}},\ \mathrm{k} = \mathrm{k}_1,\ \mathrm{k}_2,\ \ldots$$

or shall have allowable minimal and maximal values, i.e.

$$\boldsymbol{P}_{\mathrm{MIN}}{}^{\mathrm{K}} \leq \boldsymbol{x}_{\mathrm{k}} \leq \boldsymbol{P}_{\mathrm{MAX}}{}^{\mathrm{K}},\ \mathrm{k} = \mathrm{k}_1,\ \mathrm{k}_2,\ \ldots$$

It is said that the conditions outlining the class of acceptable solutions reflect the feasibility requirements and the specifics of the problem, and form a special region in the solution space $\boldsymbol{X}$.

The optimization criterion is always a scalar function defined in the solution space

$$\boldsymbol{Q}(\boldsymbol{X}) = \boldsymbol{Q}(\boldsymbol{x}_1, \boldsymbol{x}_2,\ \ldots, \boldsymbol{x}_{\mathrm{n}})$$

that represents the degree of the consistence of any solution vector $\boldsymbol{X}$ to the general goal of the engineering task. For example, the resource distribution problem may reflect the goal of maximizing the resource utilization, and intuitively its solution would provide maximum allowable amounts of the resource to the consumers having the highest coefficients of its utilization, $\boldsymbol{\alpha}_{\mathrm{i}}$, i = 1,2,. . .,n. It could be seen that in this case the criterion could be defined as

$$Q(X) = \sum_{\mathrm{i}=1}^{\mathrm{n}} \alpha_{\mathrm{i}} x_{\mathrm{i}}$$

In the situation when the goal of the resource distribution problem is to minimize the total cost of transporting the resource to the consumers, and intuitively the most remote consumers are expected to receive the least amounts of the resource within the allowable limits, the criterion could be visualized as

$$Q(X) = \sum_{i=1}^{n} \beta_i x_i$$

where $\boldsymbol{\beta}_{\mathrm{i}}$, i = 1,2,. . ., n are the transportation costs per unit of the resource for particular consumers. It is common to refer to the function $\boldsymbol{Q}(\boldsymbol{X})$ as ***criterion***, or ***objective function***, or a ***loss function***, that highlights various aspects of the nature of the optimization problem.

Finally, the optimization procedure must result in the rule that would facilitate the detection of such a point, $\boldsymbol{X}^{\mathrm{OPT}}$, in the region of acceptable solutions in the space $\boldsymbol{X}$ where criterion $\boldsymbol{Q}(\boldsymbol{X})$ has its minimum (maximum) value

$$\boldsymbol{Q}^{\mathrm{OPT}} = \boldsymbol{Q}\left(\boldsymbol{X}^{\mathrm{OPT}}\right)$$

One should realize that the search for the maximum of criterion $\boldsymbol{Q}_1(\boldsymbol{X})$ is equivalent to the search of the minimum of the criterion $\boldsymbol{Q}_2(\boldsymbol{X}) = -\boldsymbol{Q}_1(\boldsymbol{X})$ and vice versa, therefore we will always refer to the task of *optimization* as the task of *minimization*.

Recall the approach to minimization presented as a part of undergraduate calculus. It suggests that if a minimum of some scalar function

$$\boldsymbol{Q}(\boldsymbol{X}) = \boldsymbol{Q}(\boldsymbol{x}_1, \boldsymbol{x}_2, \ \ldots, \boldsymbol{x}_\mathrm{n}),$$

i.e. $\boldsymbol{X}^* = [\boldsymbol{x}_1^*, \boldsymbol{x}_2^*, \ldots, \boldsymbol{x}_\mathrm{n}^*]$, could be found as the solution of the system of n equations,

$$\begin{aligned}
&\frac{\partial}{\partial x_1} Q(x_1, x_2, \ldots, x_n) = f_1(x_1, x_2, \ldots, x_n) = 0 \\
&\frac{\partial}{\partial x_2} Q(x_1, x_2, \ldots, x_n) = f_2(x_1, x_2, \ldots, x_n) = 0 \\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
&\frac{\partial}{\partial x_n} Q(x_1, x_2, \ldots, x_n) = f_n(x_1, x_2, \ldots, x_n) = 0
\end{aligned}$$

While this suggestion is consistent with the rigors of a community college, from the engineering point of view it is quite unrealistic because of the following reasons,

1. Function $\boldsymbol{Q}(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_\mathrm{n})$ is typically so complex that its derivatives

$$\frac{\partial}{\partial x_i} Q(x_1, x_2, \ldots, x_n), \ i = 1, 2, \ \ldots, \ n$$

   are very difficult and sometimes impossible to define analytically
2. Derivatives $\frac{\partial}{\partial x_i} Q(x_1, x_2, \ldots, x_n)$, $i = 1, 2, \ldots, n$ are nonlinear functions of $\boldsymbol{x}_1$, $\boldsymbol{x}_2$, …, $\boldsymbol{x}_\mathrm{n}$, and the system of equations shown above may have multiple solutions, may have no solution, and in any case, cannot be solved analytically.
3. The entire definition of the function minimization task does not address the existence of constraints.
4. The function to be minimized may not have any analytical definition, but for any combination of numerical values of its arguments its value could be defined numerically, for example by conducting an experiment.

In most real life situations, the optimization task could be performed only numerically and be compared with the navigation through a very complex terrain to the highest (in the maximization case) existing peak while avoiding the obstacles and low peaks. The task is aggravated by the fact that the terrain is multidimensional and the obstacles could be detected only by direct contact. Figure 4.1 below drawn by my cartoonist friend Joseph Kogan depicts the task of optimization based on my comments.

Unsurprisingly, the optimization became an engineering tool only due to the proliferation of modern computers. We will present several common models,

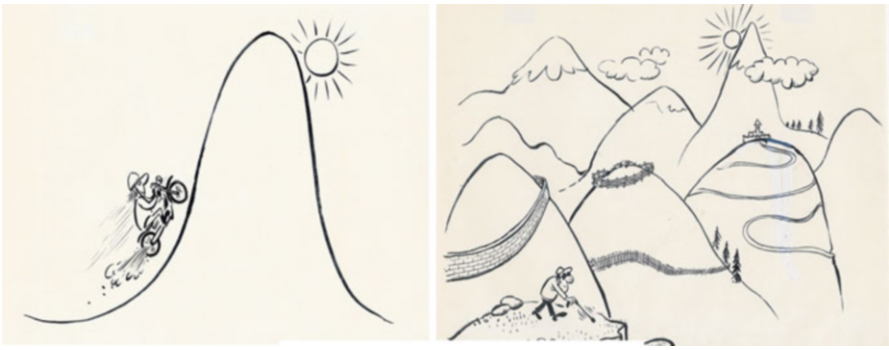


Fig. 4.1 Myth vs. reality of optimization

methods and applications of optimization that should be included in the toolbox of a modern engineer. These techniques will include linear programming, numerical techniques of nonlinear programming (gradient, random and direct search), genetic optimization, and dynamic programming. We do not expect a modern engineer to develop optimization techniques, this is the mathematicians' domain, however a good engineer shall be able to:

- recognize a situation lending itself to an optimization task
- formulate the optimization problem, i.e. define its variables, criterion and constraints
- recognize the resultant problem as one of the typical optimization problems
- find and apply a suitable optimization tool (perhaps available in MATLAB)

4.1 Linear Programming

Linear programming is an optimization technique suitable for the situations when the set of conditions, outlining the region of acceptable solutions, and the goodness criterion are linear functions defined in the solution space.

In a linear programming problem, the region of acceptable solutions is defined by the set of equalities and inequalities as follows:

$$\sum_{i=1}^{n} a_{ij} x_i = b_j \text{ and } \sum_{i=1}^{n} a_{iK} x_i \leq b_K$$

where $\boldsymbol{x}_{\mathrm{i}}$, $\mathrm{i} = 1,2,..,\ n$ are optimization variables that constitute the solution space, $\mathrm{j} = 1,2,\ldots, \boldsymbol{L}$ is the equality index, and $\mathrm{k} = 1,2,\ldots, \boldsymbol{M}$ is the inequality index. Note that the number of equalities must be less than the dimension of the solution space otherwise the region of the acceptable solutions will include only one point (when $n = L$), or could be empty (when $L > n$). One should understand that inequalities

can always be redefined as the standard "greater or equal" type, indeed inequality of "less or equal" type, i.e. $\sum_{i=1}^{n} a_{iK}x_i \leq b_K$ could be easily converted into the "greater or equal" type by changing signs: $-\sum_{i=1}^{n} a_{iK}x_i \geq -b_K$, consequently only the "greater or equal" type inequalities will be considered. Note that the class of acceptable solutions could be empty even when $n > L$: the inequalities and equalities could be mutually contradictive.

The criterion of a linear optimization problem is defined by a linear function,

$$Q(x_1, x_2, \ldots, x_n) = \sum_{i=1}^{n} c_i x_i$$

that has to be minimized,

$$Q(x_1, x_2, \ldots, x_n) = \sum_{i=1}^{n} c_i x_i \rightarrow \min$$

or

$$-Q(x_1, x_2, \ldots, x_n) = -\sum_{i=1}^{n} c_i x_i \rightarrow \min$$

if the original criterion $\boldsymbol{Q}(\boldsymbol{X})$ had to be maximized.

Example 4.1 Consider one of the typical problems of linear programming, the task distribution problem. There are 5 reactors operating at a chemical plant and producing the same product. Due to capacity, design specifics and the technical status, the reactors have different efficiency expressed by the extraction coefficients, $\boldsymbol{\alpha}_j$, j = 1,2,3,4,5. The capacities of these reactors, $\boldsymbol{q}_j$, j = 1,2,3,4,5, are also different

reactor, j	1	2	3	4	5
coefficient $\boldsymbol{\alpha}_j$	0.81	0.76	0.63	0.71	0.68
capacity $\boldsymbol{q}_j$ (units)	150	200	175	120	96

The chemical plant is required to process a certain amount of raw material, say $\boldsymbol{P} = 500$ units that should be rationally distributed between the reactors in the sense that the overall extraction coefficient will be maximized. It could be seen that the solution space of this problem comprises of 5 variables, $\boldsymbol{x}_1$–$\boldsymbol{x}_5$, representing the amount of raw material loaded in respective reactors. The constraints of this problem must address the following requirements:

Amount of raw material loaded in the j-th reactor must be non-negative: $\boldsymbol{x}_j \geq 0$, j = 1,2,. . .,5

The total amount of raw material to be loaded in reactors is defined: $\sum_{j=1}^{5} x_j = P$

The amount of raw material loaded in a particular reactor cannot exceed the capacity of this reactor: $x_j \leq q_j$, j = 1,2,. . .,5

The criterion of this problem could be defined as $\sum_{j=1}^{5} \alpha_j x_j \rightarrow \max$ or $-\sum_{j=1}^{5} \alpha_j x_j \rightarrow \min$

The mathematical formulation of this problem is

$$-0.81x_1 - 0.76x_2 - 0.63x_3 - 0.71x_4 - 0.68x_5 \rightarrow \min$$

subject to conditions

$$x_1 \geq 0, x_2 \geq 0, x_3 \geq 0, x_4 \geq 0, x_5 \geq 0$$
$$-x_1 \geq -150, \; -x_2 \geq -200, \; -x_3 \geq -175, \; -x_4 \geq -120, \; -x_5 \geq -96$$
$$x_1 + x_2 + x_3 + x_4 + x_5 = 500$$

One can realize that this problem has an infinite number of alternative solutions providing that the total amount of raw material ***P*** is less than the total capacity of the reactors, thus creating the opportunity for the optimization. In the case when the total amount of raw material ***P*** is equal to the total capacity of the reactors, only one solution exists and the optimization is impossible. Finally, in the case when the total amount of raw material ***P*** is greater than the total capacity of the reactors, the problem does not have any solution.

It could be also realized that the optimal solution procedure for this problem is quite trivial:

Step 1. The first reactor, having the highest efficiency coefficient, should be loaded to full capacity ($x_1^{OPT} = 150$, 350 units still is to be distributed), then

Step 2. The second most efficient reactor must be loaded to full capacity ($x_2^{OPT} = 200$, 150 units is to be distributed), then

Step 3. The third most efficient reactor must be loaded to full capacity ($x_4^{OPT} = 120$, 30 units is to be distributed), then

Step 4. The fourth most efficient reactor must be loaded with the remaining amount of raw material ($x_5^{OPT} = 30$ units, zero units to be distributed), then $x_3^{OPT} = 0$.

It should be noted that most linear programming problems do not allow for such a simple solution procedure.

Example 4.2 The transportation problem. A product stored at 3 warehouses must be distributed between 5 consumers in such a fashion that the total cost of transporting the product is minimized.

The solution space of this problem is formed by $3 \times 5 = 15$ variables, $\boldsymbol{x}_{\mathrm{jk}}$, $\mathrm{j} = 1,2,3$, $\mathrm{k} = 1,2,3,4,5$, representing the amount of the product delivered from the j-th warehouse to the k-th consumer. Introduce the matrix of transportation costs, $\boldsymbol{c}_{\mathrm{jk}}$, $\mathrm{j} = 1,2,3$, $\mathrm{k} = 1,2,3,4,5$, representing the cost of transportation of one unit of the product from the j-th warehouse to the k-th consumer. Introduce quantities $\boldsymbol{P}_{\mathrm{j}}$, $\mathrm{j} = 1,2,3$, representing the amount of the product at j-th warehouse, and quantities $\boldsymbol{W}_{\mathrm{k}}$, $\mathrm{k} = 1,2,3,4,5$, representing the amount of the product requested by k-th consumer. Then the mathematical formulation of the problem is

$$\sum_{\mathrm{k}=1}^{5} \sum_{\mathrm{j}=1}^{3} c_{jk} x_{jk} \rightarrow \min$$

subject to the following conditions

a) non-negativity, $\boldsymbol{x}_{\mathrm{jk}} \geq 0$, $\mathrm{j} = 1,2,3$, $\mathrm{k} = 1,2,3,4,5$

b) amount of the product available at each warehouse, $\sum_{k=1}^{5} x_{jk} \leq P_j$, $\mathrm{j} = 1,2,3$

c) amount of product delivered to each consumer, $\sum_{\mathrm{j}=1}^{3} x_{jk} = W_k$, $\mathrm{k} = 1,2,3,4,5$

One can realize that the solution of this problem exists if $\sum_{k=1}^{5} W_k \leq \sum_{j=1}^{3} P_j$, however it cannot be obtained without a computationally intensive and rigorously justified algorithm. It also should be noted that typical solutions of linear programming problems comprise non-negative variables and therefore the non-negativity of the solution is assured not by special constraints but by the solution procedure itself.

Example 4.3 The mixing problem. Preparing the right raw material is one of the conditions for obtaining a high quality end product in chemical or metallurgical manufacturing. Assume that the raw material is characterized by percentages of four ingredients: $A_1\%, A_2\%, A_3\%$, and $A_4\%$. The raw material is prepared by mixing six components in the amounts (in tons) $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_6$. Each component contains all four ingredients, but the concentrations are all different, for example $\boldsymbol{a}_{jk}$ (%) is the concentration of the ingredient #j ($j = 1,2,3,4$) in the component #k ($k = 1,2,3,4,5,6$). The cost of each component is given: $\boldsymbol{c}_k$ (\$/ton), ($k = 1,2,3,4,5,6$). Also given are the required total amount of the raw material, $\boldsymbol{P}$ (tons) and the available amounts of the individual components, $\boldsymbol{q}_k$ (tons), ($k = 1,2,3,4,5,6$). It is required to prepare the least expensive mixture.

The problem definition is as follows:
Minimize the cost of the mixture:

$$\sum_{k=1}^{6} c_k x_k \rightarrow \min$$

Subject to constraints on

the total amount of the raw material $\sum_{k=1}^{6} x_k = P$

percentages of four ingredients (j = 1,2,3,4) $\sum_{k=1}^{6} a_{jk}x_k = A_j \cdot P$

available amounts of individual components, ($k = 1,2,3,4,5,6$) $x_k \le q_k$

Again, the optimal problem solution, if it exists, could be obtained via some numerically extensive procedure.

Let us consider such a procedure.

4.1.1 Geometrical Interpretation of Linear Programming

Geometrical interpretation of linear programming is crucial for the understanding of the computational nature of its algorithm. Geometrical interpretation works best for the two-dimensional solution space and the inequality-type constraints.

Consider a straight line in two-dimensional space defined by the equation $\boldsymbol{a}_1 x_1 + \boldsymbol{a}_2 x_2 = \boldsymbol{b}$ like the one below in Fig. 4.2.

It is known that any point on this line, for example $[x_1^1, x_2^1]$ satisfies this equation, i.e. $\boldsymbol{a}_1 x_1^1 + \boldsymbol{a}_2 x_2^1 = \boldsymbol{b}$. It also known that any point above this line, such as $[x_1^2, x_2^2]$, results in $\boldsymbol{a}_1 x_1^2 + \boldsymbol{a}_1 x_2^2 > \boldsymbol{b}$, and any point below this line, $[x_1^3, x_2^3]$, results in $\boldsymbol{a}_1 x_1^3 + \boldsymbol{a}_2 x_2^3 < \boldsymbol{b}$. Consequently, any condition $\boldsymbol{a}_1 x_1 + \boldsymbol{a}_2 x_2 \le \boldsymbol{b}$ (or $-\boldsymbol{a}_1 x_1 - \boldsymbol{a}_2 x_2 \ge -\boldsymbol{b}$) outlining the class of acceptable solutions indicates that the acceptable solutions must be located on or below the appropriate straight line. At the same time, any condition $\boldsymbol{a}_1 x_1 + \boldsymbol{a}_2 x_2 \ge \boldsymbol{b}$ (or $-\boldsymbol{a}_1 x_1 - \boldsymbol{a}_2 x_2 \le -\boldsymbol{b}$) indicates that acceptable solutions must be located on or above the appropriate straight line. One can visualize a domain of acceptable solutions defined by inequality-type conditions as the part of the plane that simultaneously complies with all inequality-type conditions (highlighted below in Fig. 4.3):

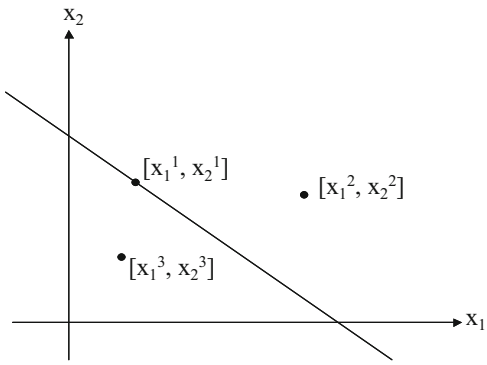


Fig. 4.2 How linear constraints work

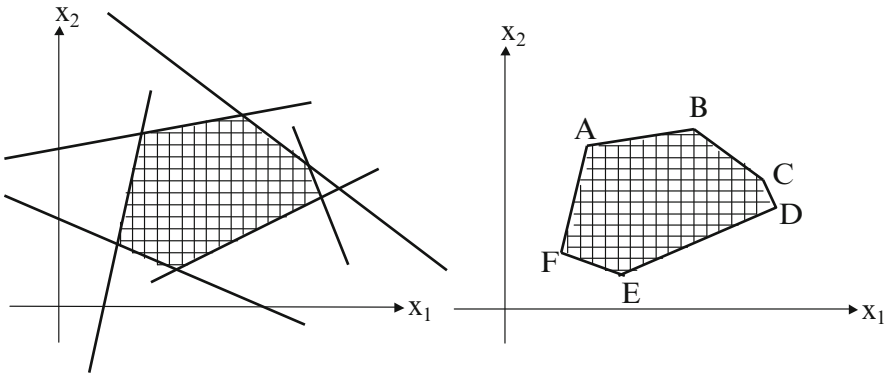


Fig. 4.3 Combination of linear constraints and domain of acceptable solutions

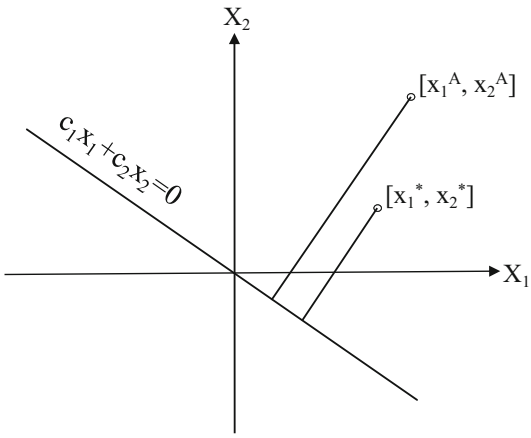


Fig. 4.4 Solution point and criterion value

Now consider a straight line $c_1x_1 + c_2x_2 = 0$ and two points, $[x_1^A, x_2^A]$ and $[x_1^*, x_2^*]$, located in the two-dimensional space. Note that the distance between the straight line and point $[x_1^A, x_2^A]$ is greater than the distance between this line and point $[x_1^*, x_2^*]$. This results in the following result that could be easily verified by a numerical example: $c_1x_1^A + c_2x_2^A > c_1x_1^* + c_2x_2^*$.

Consider the combination of the domain of acceptable solutions bounded by contour ABCDEF and the straight line $c_1x_1 + c_2x_2 = 0$ representing the criterion of a minimization problem seen below in Fig. 4.4. Note that the domain of acceptable solutions bounded by contour ABCDEF, generally speaking, forms a *convex polyhedron* in the n-dimensional space, and its individual vertices (corner points), i.e. A, B, C, . . ., are known as *basic acceptable solutions* of the linear programming problem.

It could be concluded that the solution of the problem $[x_1^{OPT}, x_2^{OPT}]$ minimizing the criterion $Q(x_1,x_2) = c_1x_1 + c_2x_2$ is located in the point that belongs to the domain of acceptable solutions and has the shortest distance from the straight line

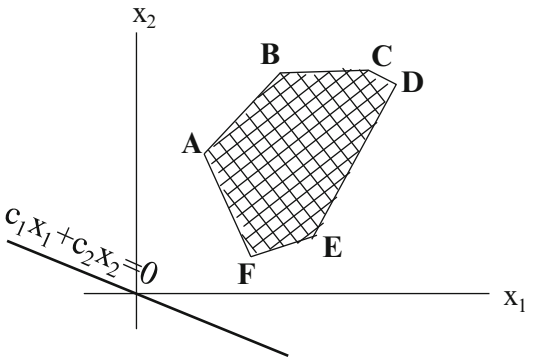


Fig. 4.5 Graphical interpretation of a linear programming problem

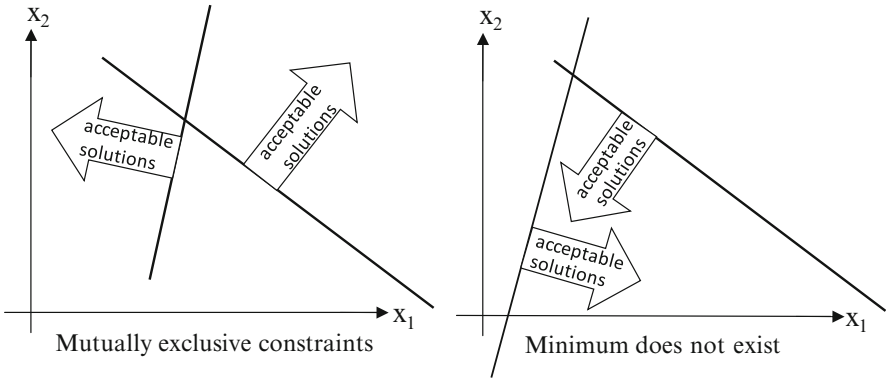


Fig. 4.6 Situations when the solution does not exist

$c_1x_1 + c_2x_2 = 0$. It could be seen that in the above Fig. 4.5 this point is F. Should the solution maximizing the criterion $Q(x_1,x_2) = c_1x_1 + c_2x_2$ be sought, it will be found in the point D that belongs to the domain of acceptable solutions and has the largest distance from the straight line $c_1x_1 + c_2x_2 = 0$.

Now consider the specifics of the linear programming problem preventing us from obtaining its optimal solution. The first condition is caused by the situation where at least two constraints are mutually exclusive, in this case even acceptable solutions do not exist. In the second case, the domain of acceptable solutions is not empty but unbounded, thus the solution minimizing the criterion does not exist. Both cases are shown in Fig. 4.6. Finally, Fig. 4.7 represents the situation where no unique optimal solution minimizing the criterion exists: the straight line representing the criterion is parallel to the side AB of the domain of acceptable solutions.

So far our discussion addressed only the inequality-type constraints. Imagine that a linear programming problem contains k equality-type constraints, m inequality-type constraints and has n solution variables where $n > k$. Assume that the problem is formulated as follows:

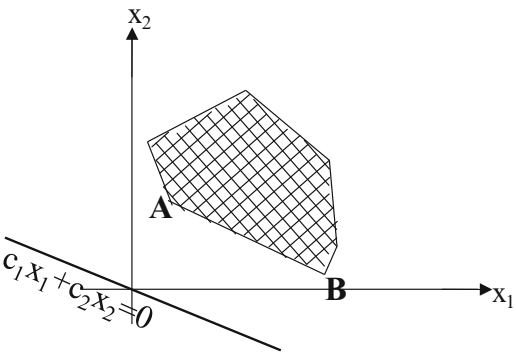


Fig. 4.7 No unique minimum exists

$$\text{minimize} \sum_{i=1}^{n} c_i x_i$$

$$\text{subject to constraints} \sum_{i=1}^{n} p_{ij} x_i \ = q_j \ , \ j = 1, 2, 3, \ \ldots, k$$

$$\text{and} \sum_{i=1}^{n} a_{ij} x_i \ \leq b_j \ , \ j = 1, 2, 3, \ \ldots, m$$

Note that condition $\boldsymbol{n > k}$ creates the situation when $\boldsymbol{k}$ variables could be assigned arbitrary values and removed from the list of solution variables. Since our goal is the minimization of the criterion $\sum_{i=1}^{n} c_i x_i$ we shall assign zero values preferably to those variables that have largest values of the corresponding coefficients $\boldsymbol{c_i}$. This is done by sequential application of a special computational operation known in linear algebra as *pivoting*. Indeed, after $\boldsymbol{k}$ pivoting steps the problem will be reduced to the following definition:

$$\text{minimize} \sum_{i=1}^{n-k} \overline{c}_i x_i$$

$$\text{subject to constraints} \sum_{i=1}^{n-k} \overline{a}_{ij} x_i \ \leq \overline{b}_j \ , \ j = 1, 2, 3, \ \ldots, m$$

where $\overline{a}_{ij}$, $\overline{b}_j$, $\overline{c}_j$, $\quad i = 1, 2, \ \ldots, n-k, \ j = 1, 2, 3, \ \ldots m$ are problem parameters modified by pivoting steps.

In summary, a linear programming procedure intended for solution of a minimization problem with $\boldsymbol{n}$ variables, $\boldsymbol{k}$ equality-type and $\boldsymbol{m}$ inequality-type constraints ($\boldsymbol{n > k}$), could be formulated as follows:

Step 1. Reduction of the dimension of the solution space by the elimination of k strategically chosen variables and setting their values in the optimal solution to zero

Step 2. Finding basic acceptable solutions of the problem by solving possible combinations of $n-k$ out of m equations $\sum_{i=1}^{n-k} \overline{a}_{ij} x_i = \overline{b}_j$, $j = 1, 2, 3, \ldots, m$

Step 3. Finding the optimal solution of the problem as the basic acceptable solution that $\leq$

Note that there are many highly efficient software tools that could be recommended for the solution of a linear programming problem. (For example see http://www.onlinecalculatorfree.org/linear-programming-solver.html).

Example 4.4 Solving a simple linear programming problem given below:

$$\text{Minimize } Q(X) = 3x_1 + 10x_2 + 5x_3 + 2x_4$$

subject to conditions

$$\begin{aligned} &x_1 + x_2 + x_3 + x_4 \leq 125 \\ &x_2 - 8x_3 + x_4 \leq 12 \\ &-x_1 + 2x_2 - 3x_3 + x_4 \leq 24 \\ &x_1 + x_2 = 36 \\ &2x_1 - 5x_2 + 8x_3 + 4x_4 = 16 \end{aligned}$$

The optimal solution (as per tool http://www.onlinecalculatorfree.org/linear-programming-solver.html):

$$Q^{\text{OPT}} = 164; x_1 = 28, x_2 = 8, x_3 = 0, x_4 = 0$$

Example 4.5 A resource distribution problem. A product available from three suppliers is to be provided to four consumers. The amounts of the product requested by individual consumers are respectively: 150, 230, 80 and 290 (units). The amounts of the product available at each supplier are: 300, 270 and 275 units. The transportation costs of the product from each supplier to each consumer in $ per unit are listed in the table below:

	Consumer #1	Consumer #2	Consumer #3	Consumer #4
Supplier #1	25	16	33	48
Supplier #2	45	15	36	11
Supplier #3	21	31	40	52

It is required to minimize the overall transportation cost while satisfying the consumers' demands and not to exceed suppliers' capabilities. The following problem definition is self-explanatory and at the same time is fully consistent with the data format of the tool offered at

http://www.onlinecalculatorfree.org/linear-programming-solver.html

$$\text{Maximize p} = -25x_{11} - 16x_{12} - 33x_{13} - 48x_{14} - 45x_{21} - 15x_{22} - 36x_{23} - 11x_{24} - 21x_{31} - 31x_{32} - 40x_{33} - 52x_{34}$$

subject to

$$\begin{aligned} x_{11} + x_{12} + x_{13} + x_{14} &<= 300 \\ x_{21} + x_{22} + x_{23} + x_{24} &<= 270 \\ x_{31} + x_{32} + x_{33} + x_{34} &<= 275 \\ x_{11} + x_{21} + x_{31} + x_{41} &= 150 \\ x_{12} + x_{22} + x_{32} + x_{42} &= 230 \\ x_{13} + x_{23} + x_{33} + x_{43} &= 80 \\ x_{14} + x_{24} + x_{34} + x_{44} &= 290 \end{aligned}$$

The Optimal Solution: $p = -13{,}550$; $x_{11} = 0$, $x_{12} = 230$, $x_{13} = 70$, $x_{14} = 0$, $x_{21} = 0$, $x_{22} = 0$, $x_{23} = 0$, $x_{24} = 270$, $x_{31} = 150$, $x_{32} = 0$, $x_{33} = 10$, $x_{34} = 20$ and could be summarized as

	Consumer #1	Consumer #2	Consumer #3	Consumer #4	Supplier total
Supplier #1	0	230	70	0	300
Supplier #2	0	0	0	270	270
Supplier #3	150	0	10	20	170
Consumer total	150	230	80	290	**Total transportation cost: $13,550**

4.2 Nonlinear Programming: Gradient

Gradient of a function of several variables, $Q(x_1, x_2, \ldots, x_n)$, is defined as a vector comprising partial derivatives of this function with respect to individual variables, i.e.

$$\nabla Q(X) = \nabla Q(x_1, x_2, \ldots, x_n) = \left[\frac{\partial Q}{\partial x_1} \quad \frac{\partial Q}{\partial x_2} \quad \ldots \quad \frac{\partial Q}{\partial x_n} \right]^{\mathrm{T}}$$

The above expression refers to an analytical definition of the gradient, however, it could be numerically defined at a particular location of the problem space, $\boldsymbol{X}^* = [\boldsymbol{x}1^*, \boldsymbol{x}2^*, \ldots, \boldsymbol{x}n^*]T$. Let us refer to numerically defined gradient as $\nabla \boldsymbol{Q}(\boldsymbol{X}^*)$ where * is the index of the particular point where this gradient is defined. It is known that a numerically defined gradient is a good navigational tool: it is a vector always pointing in the direction of the increase of function $\boldsymbol{Q}$ in the space $\boldsymbol{X}$.

Let us utilize this property of gradient for the minimization of function $\boldsymbol{Q}(\boldsymbol{X})$. First, select some initial point $\boldsymbol{X}^1 = [\boldsymbol{x}_1{}^1, \boldsymbol{x}_2{}^1, \ldots, \boldsymbol{x}_n{}^1]^T$ and numerically evaluate derivatives of function $\boldsymbol{Q}(\boldsymbol{X})$ in the vicinity of this point:

$$\frac{\partial Q(X^1)}{\partial x_1} \approx \frac{Q(x_1{}^1 + \Delta,\ x_2{}^1, \ldots, x_n{}^1) - Q(x_1{}^1,\ x_2{}^1, \ldots, x_n{}^1)}{\Delta}$$

$$\frac{\partial Q(X^1)}{\partial x_2} \approx \frac{Q(x_1{}^1,\ x_2{}^1 + \Delta\ , \ldots, x_n{}^1) - Q(x_1{}^1,\ x_2{}^1, \ldots, x_n{}^1)}{\Delta}$$

$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$

$$\frac{\partial Q(X^1)}{\partial x_i} \approx \frac{Q(x_1{}^1,\ x_2{}^1, \ldots,\ x_i{}^1 + \Delta,\ \ldots, x_n{}^1) - Q(x_1{}^1,\ x_2{}^1, \ldots, x_n{}^1)}{\Delta}$$

$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$

$$\frac{\partial Q(X^1)}{\partial x_n} \approx \frac{Q(x_1{}^1,\ x_2{}^1, \ldots,\ x_n{}^1 + \Delta) - Q(x_1{}^1,\ x_2{}^1, \ldots, x_n{}^1)}{\Delta}$$

where Δ is a small positive increment chosen on the basis of experience and intuition (V.S.: $\Delta = 0.0001$ is a good choice). Note that this approximation of derivatives, known as a forward difference, is not unique but is good enough for most applications. Now, when the direction towards the increase of function $\boldsymbol{Q}(\boldsymbol{X})$ is known, and the direction towards the minimum is the opposite one, we can make a step from the initial point $\boldsymbol{X}^1$ to the new point $\boldsymbol{X}^2$ that is expected to be closer to the point of minimum: $\boldsymbol{X}^2 = \boldsymbol{X}^1 - \mathrm{a} \cdot \nabla \boldsymbol{Q}(\boldsymbol{X}^1)$. Individual components of point $\boldsymbol{X}^2$ will be defined as follows:

$$x_1{}^2 = x_1{}^1 - a \cdot \frac{\partial Q(X^1)}{\partial x_1}$$

$$x_2{}^2 = x_2{}^1 - a \cdot \frac{\partial Q(X^1)}{\partial x_2}$$

$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$

$$x_n{}^2 = x_n{}^1 - a \cdot \frac{\partial Q(X^1)}{\partial x_n}$$

Now the procedure will repeat itself, but derivatives will be calculated in the vicinity of the new point $\boldsymbol{X}^2$ and a transition to the point $\boldsymbol{X}^3$ will be performed. This iterative process will lead to the vicinity of the minimum point of function $\boldsymbol{Q}(\boldsymbol{X})$ providing that some conditions be met. Parameter $\boldsymbol{a}$ in the expressions above is a positive adjustable constant responsible for the convergence rate of the minimization procedure. Its initial value is arbitrarily defined and could be changed (typically decreased) in the process according to the following rule. Assume that the transition from point $\boldsymbol{X}^k$ to $\boldsymbol{X}^{k+1}$ is taking place: $\boldsymbol{X}^{k+1} = \boldsymbol{X}^k - \mathrm{a} \cdot \nabla \boldsymbol{Q}(\boldsymbol{X}^k)$. The transition is successful if $\boldsymbol{Q}(\boldsymbol{X}^{k+1}) < \boldsymbol{Q}(\boldsymbol{X}^k)$, however in the situation when $\boldsymbol{Q}(\boldsymbol{X}^{k+1}) \geq \boldsymbol{Q}(\boldsymbol{X}^k)$

the value of parameter $\boldsymbol{a}$ must be reduced, for example by half, and the transition must be repeated with the value $\boldsymbol{a}^{\mathrm{NEW}} = 0.5\boldsymbol{a}$, i.e. $\boldsymbol{X}^{\mathrm{k+1}} = \boldsymbol{X}^{\mathrm{k}} - \boldsymbol{a}^{\mathrm{NEW}} \cdot \nabla \boldsymbol{Q}(\boldsymbol{X}^{\mathrm{k}})$. If necessary, a value shall be repeatedly reduced until a successful transition will take place. The reduced $\boldsymbol{a}$ value shall be kept unchanged for the consequent step.

Termination conditions for the described procedure could be defined in a number of ways. First, and the simplest, is the definition of the maximum number of iterations (successful reduction steps of the function to be minimized). It is also common to stop the procedure if several (5, 10, 20) iterations did not result in a noticeable change in the optimization variables, i.e. $\left| \boldsymbol{X}^{\mathrm{k-6}} - \boldsymbol{X}^{\mathrm{k-5}} \right| \leq \xi$ and $\left| \boldsymbol{X}^{\mathrm{k-5}} - \boldsymbol{X}^{\mathrm{k-4}} \right| \leq \xi$ and ... and $\left| \boldsymbol{X}^{\mathrm{k+1}} - \boldsymbol{X}^{\mathrm{k}} \right| \leq \xi$ where $\xi > 0$ is a small arbitrary number. A block diagram of the procedure is seen in Fig. 4.8.

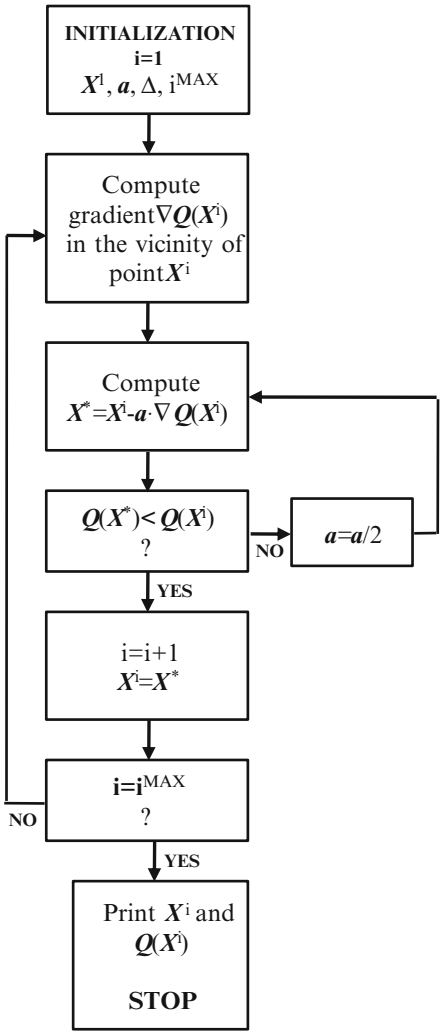


Fig. 4.8 Block diagram of gradient minimization procedure

The gradient minimization procedure is quite common due to its simplicity. It does not require analytical expressions for derivatives. Values of function Q may be defined by analytical expressions or experimentally. The drawbacks of this approach are also evident. The function must be continuous, otherwise working with derivatives presents an impossible task. This reality creates difficulties with constrained minimization. The approach implies that the function to be minimized has only one minimum point: it works only as a local minimization technique.

4.3 Nonlinear Programming: Search

Search-based optimization presents a valuable alternative to gradient optimization: it does not utilize derivatives of the function to be optimized thus expanding the range of its applications to discontinuous functions. But, how common are the discontinuous functions? It is common to introduce constraints in the optimization procedure through so-called penalty functions, and penalty functions are the typical sources of discontinuities. Therefore, search becomes very useful in many practical problems.

4.3.1 Penalty Functions

Consider the following optimization problem where criterion and constraints are represented by generally speaking, nonlinear functions $\boldsymbol{Q}(.)$ and $\boldsymbol{f}_{\mathrm{i}}(.)$, $\mathrm{i} = 1,2\ldots$:

$$\text{Minimize}\, \boldsymbol{Q}(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots \boldsymbol{x}_{\mathrm{n}})$$

subject to conditions

$$\begin{gathered}
\boldsymbol{f}_1(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots \boldsymbol{x}_{\mathrm{n}}) \geq \boldsymbol{a}_1 \\
\boldsymbol{f}_2(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots \boldsymbol{x}_{\mathrm{n}}) \geq \boldsymbol{a}_2 \\
\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
\boldsymbol{f}_{\mathrm{K}}(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots \boldsymbol{x}_{\mathrm{n}}) \geq \boldsymbol{a}_{\mathrm{K}}
\end{gathered}$$

Introduce penalty functions defined as

$$P_{\mathrm{i}}(x_1, x_2, \ldots, x_{\mathrm{n}}) = \begin{cases} C_{\mathrm{i}} \cdot [f_{\mathrm{i}}(x_1, x_2, \ldots, x_{\mathrm{n}}) - a_{\mathrm{i}}]^2, & \text{if } f_{\mathrm{i}}(x_1, x_2, \ldots, x_{\mathrm{n}}) \geq a_{\mathrm{i}} \\ 0, & \text{if } f_{\mathrm{i}}(x_1, x_2, \ldots, x_{\mathrm{n}}) < a_{\mathrm{i}} \end{cases}$$

$$\text{or } P_{\mathrm{i}}(x_1, x_2, \ldots, x_{\mathrm{n}}) = \begin{cases} C_{\mathrm{i}} \cdot |f_{\mathrm{i}}(x_1, x_2, \ldots, x_{\mathrm{n}}) - a_{\mathrm{i}}|, & \text{if } f_{\mathrm{i}}(x_1, x_2, \ldots, x_{\mathrm{n}}) \geq a_{\mathrm{i}} \\ 0, & \text{if } f_{\mathrm{i}}(x_1, x_2, \ldots, x_{\mathrm{n}}) < a_{\mathrm{i}} \end{cases}$$

$$\text{or } P_{\mathrm{i}}(x_1, x_2, \ldots, x_{\mathrm{n}}) = \begin{cases} C_{\mathrm{i}}, & \text{if } f_{\mathrm{i}}(x_1, x_2, \ldots, x_{\mathrm{n}}) \geq a_{\mathrm{i}} \\ 0, & \text{if } f_{\mathrm{i}}(x_1, x_2, \ldots, x_{\mathrm{n}}) < a_{\mathrm{i}} \end{cases}$$

where $\boldsymbol{C}_{\mathrm{i}} \gg 1$ are arbitrary weights reflecting the importance of particular constraints, i = 1,2,...K. Then the original constrained optimization problem can be represented by the following unconstrained optimization problem

$$\text{Minimize } L(x_1, x_2, \ldots, x_{\mathrm{n}}) = Q(x_1, x_2, \ldots, x_{\mathrm{n}}) + \sum_{\mathrm{i}=1}^{\mathrm{K}} P_{\mathrm{i}}(x_1, x_2, \ldots, x_{\mathrm{n}})$$

Function $\boldsymbol{L}(.)$ is commonly referred to as the "loss function". It could be seen that due to the definition of penalty functions $\boldsymbol{P}_{\mathrm{i}}(.)$ it is a discontinuous function. It also could be seen that due to large values of weights $\boldsymbol{C}_{\mathrm{i}}$ virtually any minimization algorithm would first "drive" penalty values to zero, and then, when constraints are satisfied, minimize the original function $\boldsymbol{Q}(.)$.

Consider the following example illustrating the introduction of penalty functions.

Example 4.6 unconstrained optimization problem

Minimize $Q(x_1, x_2, x_3) = 5(x_1 + 6)^2 + 2(x_1 \cdot x_2 - 6x_3)^2 - 10x_2(x_3 - 2)^3$

subject to conditions:

$$\begin{aligned} & x_1 + x_2 + 6x_3 = 10 \\ & 0 \leq x_1 \leq 25 \\ & -10 \leq x_2 + x_3 \leq 10 \\ & x_1 - 4x_3 \leq 100 \end{aligned}$$

Define penalty functions representing the imposed constraints:

$$P_1 = 10^{15} \cdot [x_1 + x_2 + 6x_3 - 10]^2$$

$$P_2 = \begin{cases} 10^{10} \cdot {x_1}^2, & \text{if } x_1 < 0 \\ 0, & \text{if } x_1 \geq 0 \end{cases}$$

$$P_3 = \begin{cases} 10^{10} \cdot (x_1 - 25)^2, & \text{if } x_1 > 25 \\ 0, & \text{if } x_1 \leq 0 \end{cases}$$

$$P_4 = \begin{cases} 10^{10} \cdot (x_2 + x_3 - 10)^2, & \text{if } (x_2 + x_3 - 10)^2 > 0 \\ 0, & \text{otherwise} \end{cases}$$

$$P_5 = \begin{cases} 10^{10} \cdot (x_1 - 4x_3 - 100)^2, & \text{if } x_2 - 4x_3 > 100 \\ 0, & \text{otherwise} \end{cases}$$

The resultant loss function

$$L(x_1, x_2, x_3) = 5(x_1 + 6)^2 + 2(x_1 \cdot x_2 - 6x_3)^2 - 10x_2(x_3 - 2)^3 + \sum_{\mathrm{i}=1}^{5} P_{\mathrm{i}}(x_1, x_2, x_3)$$

could be easily defined by a computer code. Understandably, it should be minimized by a procedure that does not utilize derivatives

$$\frac{\partial L(x_1, x_2, x_3)}{\partial x_1}, \frac{\partial L(x_1, x_2, x_3)}{\partial x_2}, \frac{\partial L(x_1, x_2, x_3)}{\partial x_3}$$

It should also be noted that due to nonlinear criterion and constraints, this problem most likely does not have one minimum, and finding the global minimum presents an additional challenge. As it is commonly done when some parameters are arbitrarily chosen (in this case, weight coefficients) the user shall inspect the obtained solution and if necessary, change the weight values. It is a good practice to demonstrate that the solution does not depend on the choice of the weights.

4.3.2 Random Search

This approach could be perceived as the most straight forward "trial-and-error" technique utilizing the full power of a modern computer and perhaps a supercomputer. It facilitates finding the global solution of linear and nonlinear, constrained and unconstrained, continuous and discontinuous optimization problems. Its only drawback is the gigantic amount of computations that is prohibitive in many practical situations. The strategy of random search is illustrated by Fig. 4.9.

4.3.3 Simplex Method of Nelder and Mead

Direct search is a much more efficient alternative to random search. One can define direct search as a thoughtful and insightful trial-and-error approach. It still has to start from some initial conditions but its steps are based on a reasonable expectation of success. It works well with continuous and discontinuous, linear and nonlinear, constrained and unconstrained functions. Its only drawback compared to random search is the inherent inability to assure that the global minimum be found. This fault is not that crucial: direct search is typically used in realistic situations where properly chosen, the initial point guarantees that the global minimum can be found. Since direct search does not call for a gigantic number of steps, it could be used in situations when values of the objective functions are defined by computer simulations and even by physical experiments.

Although there is a good number of direct search procedures utilizing different rationale for making the "next step," one of the most practical is the Simplex Method by Nelder-Mead (1965). The algorithm works with $n + 1$ vertices of a simplex (convex polytope) defined in the n-dimensional search space. It calculates (obtains) numerical values of the function to be minimized at every vertex, compares these values, and implements some rules for replacing the worst vertex (i.e.

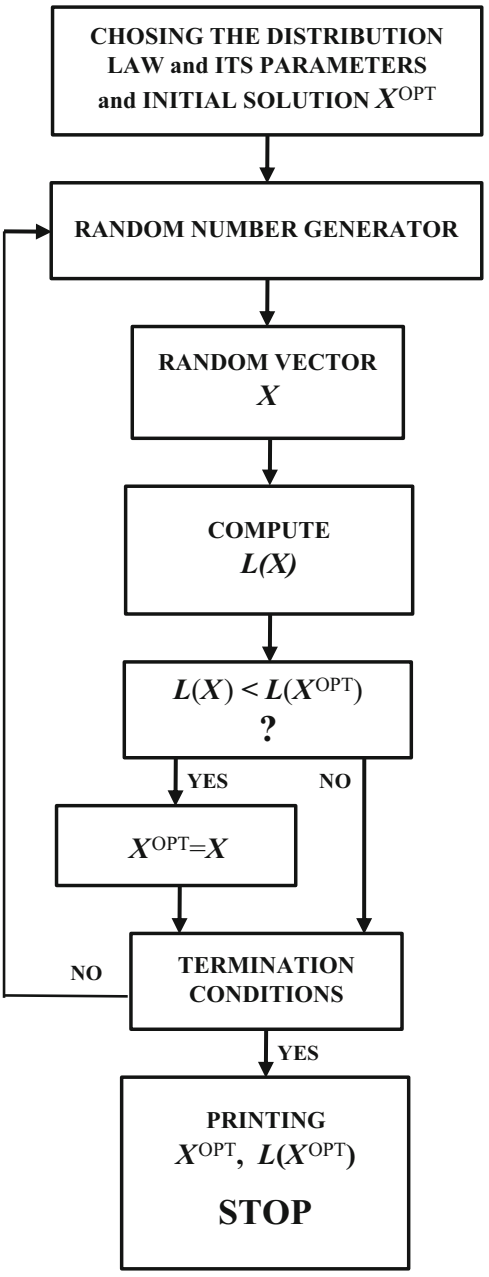


Fig. 4.9 Random search

the one with the largest value of the objective function). This process could be best illustrated in two dimensional space when simplex, with its three vertices, is just a triangle.

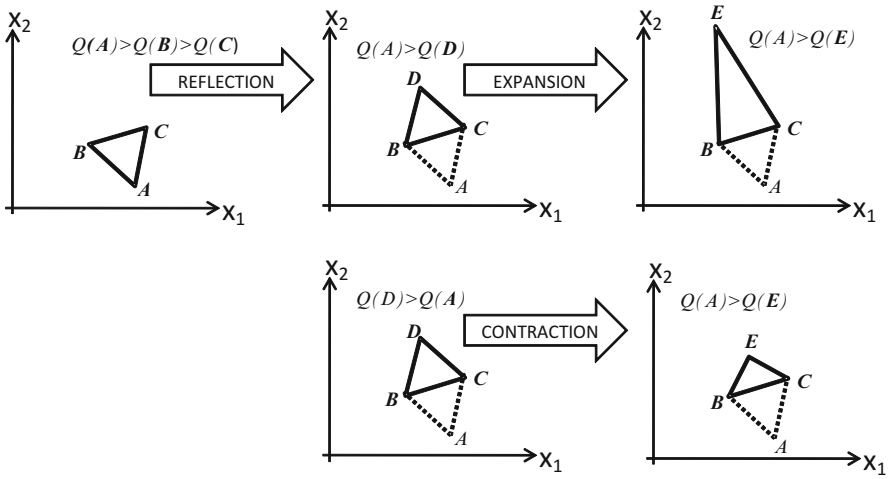


Fig. 4.10 How the simplex procedure works

First assume that the initial simplex with vertices A, B, C is established. It is often done by specifying some initial point, say point A, and the step size that determines the size of the resultant initial simplex, i.e. triangle ABC. Next is the evaluation of the objective function $Q(x_1,x_2)$ at each vertex (x_1, x_2 are coordinates of points A, B, C) thus resulting in numerical values $Q(A)$, $Q(B)$ and $Q(C)$. Assume that the comparison reveals that $Q(A) > Q(B) > Q(C)$, and since our task is minimization, the "worst" point is A. Then as seen in Fig. 4.10 above, the algorithm performs a special operation, reflection, thus establishing a new point D. What happens next, depends on the value $Q(D)$. If $Q(A) > Q(D)$, the algorithm performs expansion as shown above, creating a new point E. The expansion could be repeated providing that still $Q(A) > Q(E)$. In the situation when $Q(D) > Q(A)$, the contraction is performed. It should be performed repeatedly until condition $Q(A) > Q(E)$ is achieved. Upon the establishment of the "new" point E, the "old" point A is discarded. Now the new simplex with vertices B, C, and E is ready for performing the same computational cycle.

The termination conditions can be defined in terms of the total number of steps (optimization cycles), or in terms of the distance between vertices of the simplex.

It is good to realize that besides "purely computational" applications, the Simplex procedure can be implemented in the "(wo)man in the loop" regime for the real-time optimization of technical systems that could be represented by a simulator. Figure 4.11 below illustrates an application of the Simplex optimization to the tuning of a PID (proportional-integral-derivative) controller. The Vissim-based simulator (see http://www.vissim.com/) features a controlled process with a PID controller with manually adjustable parameters K_P, K_I, and K_D known as proportional, integral and derivative gains.

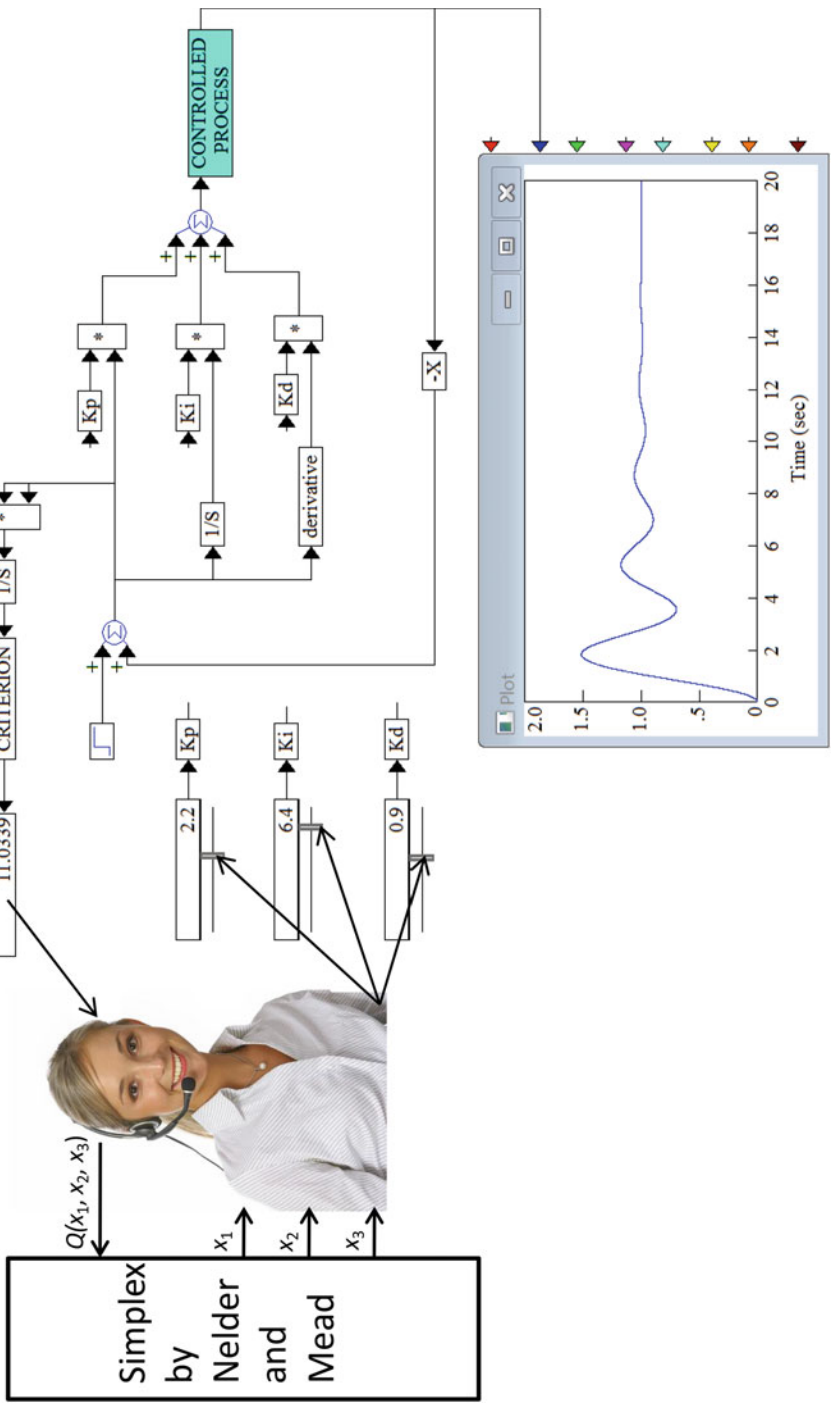


Fig. 4.11 Simplex application to tuning PID Controller

The optimization criterion is the commonly used ITSE (integral-time-squared-error) defined as

$$Q(\mathrm{K_P}, \mathrm{K_I}, \mathrm{K_D}) = \int_0^T t \cdot e^2 \cdot dt$$

where ***e*** is the system error (the discrepancy between the actual and desired system output values), ***t*** is continuous time, and ***T*** is the simulation period. It is known from Controls that minimization of ITSE-type criteria leads to the most desirable transient process in the system.

4.3.4 Exercise 4.1

Problem 1 Solving a mixing problem. The table below contains characteristics of several materials that are to be mixed to obtain a raw material for a metallurgical process. Obtain the mixture recipe that would have the following required chemical composition and total volume at minimum cost. The mixture characteristics are as follows:

$\mathrm{Fe} \geq 20\,\%$, $\mathrm{Zn} \geq 10\,\%$, $\mathrm{SiO_2} \leq 42\,\%$, $\mathrm{Cu} \geq 5\,\%$, total weight 500 tons

	Fe %	Zn %	SiO_2 %	Cu %	Cost, $/ton	Availability
Material 1	15	38	41	6	120	250 tons
Material 2	40	12	40	1	150	590 tons
Material 3	35	5	27	28	211	1000 tons
Material 4	16	11	21	18	140	520 tons
Material 5	33	1	60	5	75	2500 tons
Material 6	7	23	45	25	214	800 tons

Problem 2 Solving an LSM parameter estimation problem using a gradient procedure. Generate input and the output variables as follows (k = 1, 2,. . ., 500):

$$\begin{aligned}
\mathrm{x_1(k)} &= 5 + 3 \cdot \mathrm{Sin}\,(17 \cdot \mathrm{k}) + \mathrm{Sin}\,(177 \cdot \mathrm{k}) + .3 \cdot \mathrm{Sin}\,(1771 \cdot \mathrm{k}) \\
\mathrm{x_2(k)} &= 1 - 2 \cdot \mathrm{Sin}\,(91 \cdot \mathrm{k}) + \mathrm{Sin}\,(191 \cdot \mathrm{k}) + .2 \cdot \mathrm{Sin}\,(999 \cdot \mathrm{k}) \\
\mathrm{x_3(k)} &= 3 + \mathrm{Sin}\,(27 \cdot \mathrm{k}) + .5 \cdot \mathrm{Sin}\,(477 \cdot \mathrm{k}) + .1 \cdot \mathrm{Sin}\,(6771 \cdot \mathrm{k}) \\
\mathrm{x_4(k)} &= -.1 \cdot \mathrm{x_1(k)} + .3 \cdot \mathrm{x_2(k)} + 2.5 \cdot \mathrm{Sin}\,(9871 \cdot \mathrm{k}) + .7 \cdot \mathrm{Cos}(6711 \cdot \mathrm{k}) \\
\mathrm{y(k)} &= 2 \cdot \mathrm{x_1(k)} + 3 \cdot \mathrm{x_2(k)} - 2 \cdot \mathrm{x_3(k)} + 5 \cdot \mathrm{x_4(k)} + .3 \cdot \mathrm{Sin}\,(1577 \cdot \mathrm{k}) \\
&\quad + .2 \cdot \mathrm{Cos}(7671 \cdot \mathrm{k})
\end{aligned}$$

Obtain "unknown" coefficients of the regression equation

$$y^{MOD}(k) = a_1x_1(k) + a_2x_2(k) + a_3x_3(k) + a_4x_4(k)$$

using the least squares method implemented via the gradient procedure listed below (that could be rewritten in MATLAB). Assume zero initial values of the coefficients. Compute the coefficient of determination of the obtained regression equation.

Problem 3 Utilize data of Problem #2 to obtain coefficients of the regression equation $v^{MOD}(k) = a_1x_1(k) + a_2x_2(k) + a_3x_3(k) + a_4x_4(k)$ applying the gradient procedure. It is required, however, that all regression coefficients be positive. Show the obtained coefficients. Compute the coefficient of determination for the resultant regression equation. Explain the change in the coefficient of determination comparing with Problem #2

```
 PROGRAM GRADIENT
  DIMENSION X(10),X1(10),DER(10)
  WRITE(*,*)' ENTER NUMBER OF VARIABLES '
  READ(*,*) N
  WRITE(*,*)' ENTER THE gain OF THE PROCEDURE '
  READ(*,*)A
  WRITE(*,*)' ENTER INITIAL NUMBER OF STEPS '
  READ(*,*) NSTEP
  H=.001
  DO 1 I=1,N
  WRITE(*,*)' ENTER INITIAL VALUE FOR X(',I,')'
1  READ(*,*)X(I)
10  CONTINUE
  K=1
  CALL SYS(N,X,Q)
  QI=Q
100 CONTINUE
  DO 4 I=1,N
  X(I)=X(I)+H
  CALL SYS(N,X,Q1)
  DER(I)=(Q1-Q)/H
  X(I)=X(I)-H
 4  CONTINUE
 50 CONTINUE
  DO 5 I=1,N
 5  X1(I)=X(I)-DER(I)*A
  CALL SYS(N,X1,Q1)
  IF(Q1.GE.Q) A=A/2
  IF(Q1.GE.Q) GOTO 50
  DO 30 I=1,N
 30 X(I)=X1(I)
  Q=Q1
  IF(ABS(Q).LE.1e-5)GOTO 2
  K=K+1
  IF(K.GT.NSTEP) GOTO 2
```

```
    GOTO 100
 2  CONTINUE
    WRITE(*,*)' ITERATIONS RUN: ',NSTEP
    WRITE(*,*)' INITIAL CRITERION AVLUE: ',QI
    WRITE(*,*)' CRITERION VALUE REACHED: ',Q
    DO 7 I=1,N
 7  WRITE(*,*)' OPTIMAL VALUE: X(',I,')=',X(I)
    WRITE(*,*)' ENTER ADDITIONAL NUMBER OF STEPS '
    IF(ABS(Q).LE.1e-5)CALL EXIT
    READ(*,*) NSTEP
    IF(NSTEP.EQ.0)CALL EXIT
    GOTO 10
    END
C
    SUBROUTINE SYS(N,X,Q)
    DIMENSION X(10)
    Q=0.
    DO 1 I=1,N
    Q=Q+(X(I)-5.*I)**2
 1  CONTINUE
    Q=Q**2
    RETURN
         END
```

4.4 Genetic Optimization

Genetic optimization algorithms possess the advantages of random and direct search optimization procedures. Combined with the availability of high performance computers they alleviate major obstacles in the way of solving multivariable, nonlinear constrained optimization problems. It is believed that these algorithms emulate some concepts of the natural selection process responsible for the apparent perfection of the natural world. One can argue about the concepts, but the terminology of genetic optimization is surely adopted from biological sciences.

Assume that we are in the process of finding the optimum, say the maximum, of a complex, multivariate, discontinuous, nonlinear cost function $\boldsymbol{Q}(\boldsymbol{X})$. The constraints of the problem have already been addressed by the penalty functions introduced in the cost function and contributing to its complexity.

Introduce the concepts of an individual, generation, and successful generation. An *individual* is an entity that is characterized by its location in the solution space, $\boldsymbol{X}^{\mathrm{I}}$ and the corresponding value of the function $\boldsymbol{Q}$, i.e. $\boldsymbol{Q}(\boldsymbol{X}^{\mathrm{I}})$. A *generation* is a very large number of individuals created during the same cycle of the optimization procedure. A *successful generation* is a relatively small group of $\boldsymbol{K}$ individuals that have some common superior trait, for example, they all have the highest associated values $\boldsymbol{Q}(.)$ within their generation. The genetic algorithm consists of repeated cycles of creation of successful generations.

Creation of the Initial Generation First, the feasibility range $[x_k^{MIN}, x_k^{MAX}]$ for each solution variable x_k k = 1,2,3,. . .,n, is to be established. Each interval $[x_k^{MIN}, x_k^{MAX}]$ is divided into the same number of subintervals, say L, thus resulting in a grid within the solution space with numerous nodes. The next task is the evaluation of function Q at each node of the grid, i.e. the creation of individuals "residing" at every node. During this process the successful generation is selected consisting of K individuals that have the highest values of the function Q. It is done by forming a group of individuals ordered according to their Q values, i.e.

$$Q(X^K) \le Q(X^{K-1}) \le \ldots Q(X^2) \le Q(X^1) \quad (*)$$

Any newly generated individual X^I is discarded if $Q(X^I) \le Q(X^K)$. However if $Q(X^I) > Q(X^L)$, it is included in the group replacing the individual X^L with the lowest Q value. Therefore the successful generation still includes K individuals that are being renumbered and reordered to assure (*). This process is repeated each time a new individual is generated i.e. until the entire initial generation is created and analyzed.

Creation of the Next Successful Generation involves only members of the existing successful generation. Two techniques are utilized for this purpose, parenting and mutation. Parenting (crossover) involves two individuals, X^A and X^B and results in an "offspring"

$$X^C = \left[x_1^C, x_2^C, \ldots x_k^C, \ldots x_n^C\right]^T$$

defined as follows:

$$\begin{aligned}
x_1^C &= \lambda_1 x_1^A + (1-\lambda_1) x_1^B \\
x_2^C &= \lambda_2 x_2^A + (1-\lambda_2) x_2^B \\
&\ldots\ldots\ldots\ldots \\
x_k^C &= \lambda_k x_k^A + (1-\lambda_k) x_k^B \\
&\ldots\ldots\ldots\ldots \\
x_n^C &= \lambda_n x_n^A + (1-\lambda_n) x_n^B
\end{aligned}$$

where $0 < \lambda_k < 1$ are random numbers generated by a random number generator. Then, based on the computation of $Q(X^C)$ the newly created individual X^C is accepted into the successful generation or discarded. The parenting process is repeated several number times for every combination of two members of the original successful generation.

The mutation process implies that every member of the original successful generation, X^I originates a "mutant" $X^M = [x_1^M, x_2^M, \ldots x_k^M, \ldots x_n^M]^T$ defined as follows:

$$
\begin{aligned}
&\boldsymbol{x}_1{}^{\mathrm{M}} = \alpha_1 \boldsymbol{x}_1{}^{\mathrm{I}} \\
&\boldsymbol{x}_2{}^{\mathrm{M}} = \alpha_2 \boldsymbol{x}_2{}^{\mathrm{I}} \\
&\ldots\ldots\ldots\ldots \\
&\boldsymbol{x}_{\mathrm{k}}{}^{\mathrm{M}} = \alpha_{\mathrm{k}} \boldsymbol{x}_{\mathrm{k}}{}^{\mathrm{A}} \\
&\ldots\ldots\ldots\ldots \\
&\boldsymbol{x}_{\mathrm{n}}{}^{\mathrm{M}} = \alpha_{\mathrm{n}} \boldsymbol{x}_{\mathrm{n}}{}^{\mathrm{A}}
\end{aligned}
$$

where α_k are normally distributed random numbers generated by a random number generator. Based on the computation of $\boldsymbol{Q}(\boldsymbol{X}^{\mathrm{M}})$ the newly created individual $\boldsymbol{X}^{\mathrm{M}}$ is accepted into the successful generation or discarded. The mutation process is repeated several number times for every member of the original successful generation.

Understandably, parenting and mutation upon completion results in a new successful generation that is to be subjected to a new cycle of the procedure unless the termination conditions be satisfied. The most common termination condition refers to the variability within a successful generation, and could be expressed as:

$$\sum_{\mathrm{i}=1}^{K-1} \left| X^{\mathrm{i}} - X^{\mathrm{i}+1} \right| \leq \delta$$

where $\delta > 0$ is some judiciously chosen small positive number.

It is good to remember that genetic optimization is capable of finding a global minimum of virtually any function $\boldsymbol{Q}(\boldsymbol{X})$. Moreover, it works even when this function does not exist as an analytical expression: in this situation for any particular $\boldsymbol{X}^{\mathrm{I}}$ the value of $\boldsymbol{Q}(\boldsymbol{X}^{\mathrm{I}})$ could be determined by running a computer simulation or by an experiment. Figure 4.12 provides a block diagram of the genetic optimization procedure.

The following MATLAB code implementing a genetic optimization procedure was written by my former student Dr. Jozef Sofka

```
%genetic algorithm for minimization of a nonlinear function
%(c) Jozef Sofka 2004
%number of crossovers in one generation
cross=50;
%number of mutations in one generation
mut=30;
%extent of mutation
mutarg1=.5;
%size of population
population=20;
%number of alleles
al=5;
%trying to minimize function
%abs(a^2/b+c*sin(d)+b^c+1/(e+a)^2)
 clear pop pnew;
  %definition of "best guess" population
  pop(1:population,1)=12+1*randn(population,1);
  pop(1:population,2)=1.5+.1*randn(population,1);
  pop(1:population,3)=13+1*randn(population,1);
```

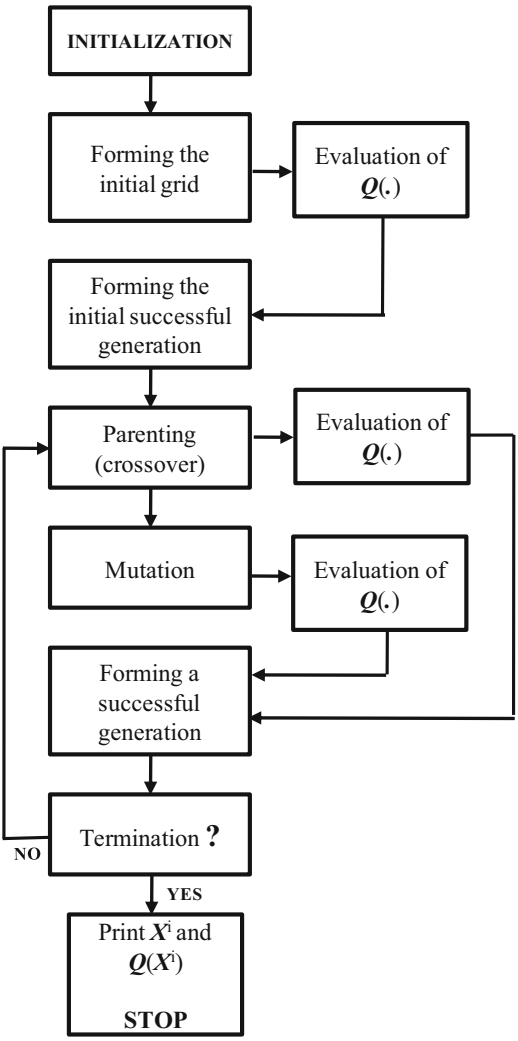


Fig. 4.12 Block diagram of a genetic optimization procedure

```
  pop(1:population,4) = 1.5 + .2*randn(population,1);
  pop(1:population,5) = (.5*randn(population,1));
% evaluation of fitness population
for f = 1:population
    e(f) = abs(pop(f,1)^2/pop(f,2) + pop(f,3)*sin(pop(f,4)) + pop(f,2)
^pop(f,3) + 1/(pop(f,5) + pop(1))^2);
  end
[q,k] = sort(e);
%number of generations
for r = 1:500
  parameters(r,1:al) = pop(k(1),1:al);
  fitness(r) = e(k(1));
%crossover
for f = 1:cross
```

```
  p1=round((rand+rand)/2*(population-1))+1;
  p2=round((rand+rand)/2*(population-1))+1;
  p3=(2*rand-.5);
      pnew(f,:)=pop(k(p1),1:al)+p3*(pop(k(p2),1:al)-pop(k(p1),1:
al));
  %evaluation of fitness
   fit(f)=abs(pnew(f,1)^2/pnew(f,2)+pnew(f,3)*sin(pnew(f,4))+pnew
(f,2)^pnew(f,3)+1/(pnew(f,5)+pnew(1))^2);
  end
%selection
for f=1:cross
  if (fit(f)<e(k(population-3)))
    pop(k(population),:)=pnew(f,:);
    e(k(population))=fit(f);
    [q,k]=sort(e);
    end
    end
%mutation
for f=1:mut
  p=round(rand*(population-1))+1;
  o=round((al-1)*rand)+1;
  pnew(f,:)=pop(p,:);
  pnew(f,o)=pnew(f,o)+mutarg1*randn(1,1);
  %evaluation of fitness
   fit(f)=abs(pnew(f,1)^2/pnew(f,2)+pnew(f,3)*sin(pnew(f,4))+pnew
(f,2)^pnew(f,3)+1/(pnew(f,5)+pnew(1))^2);
  end
%selection
for f=1:mut
  if (fit(f)<e(k(population-1)))
    pop(k(population),:)=pnew(f,:);
    e(k(population))=fit(f);
    [q,k]=sort(e);    end
  end
end
fprintf('Parameters a=%f; b=%f; c=%f; d=%f; e=%f\n', ...,
pop(k(1),1), pop(k(1),2), pop(k(1),3), pop(k(1),4), pop(k(1),5))
fprintf('minimize   function   abs(a^2/b+c*sin(d)+b^c+1/(e+a)^2)
\n')
figure
plot(parameters)
figure
  semilogy(fitness)
```

4.4.1 Exercise 4.2

Problem 1 Use Simplex Optimization procedure (to be provided) to tune parameters of a PID controller as shown in Fig. 3.3. The simulation setup could be implemented in Simulink or Vissim. The following transfer function is recommended for the controlled plant:

$$G(s) = \frac{s+6}{s^3 + 6s^2 + 10s + 10}$$

To show the effectiveness of the tuning procedure provide a sequence (five or so) of numerical values of the parameters of the controller, values of the criterion, and the system step responses.

Problem 2 Given input–output data representing a highly nonlinear, static process:

x_1	x_2	x_3	y
1	1	1	17.59141
1	1	2	21.59141
1	2	2	44.94528
2	2	2	81.89056
2	2	3	89.89056
2	3	3	216.8554
3	3	3	317.2831
−3	3	3	−285.2831
−3	−3	3	15.25319
−3	−3	−3	−0.496806
−3	3	−3	−301.0331
−1	3	−3	−100.1777
−1	3	5	−36.42768
−5	2	4	−152.7264
5	2	1	188.7264

Given the configuration of the mathematical model of this process:

$$y^{MOD} = a_1 x_1 e^{a_2 x_2} + a_3{}^{(a_4 x_3 + a_5)}$$

Utilize the Genetic Optimization (GO) program provided above and the input/output data to estimate unknown parameters of the mathematical model given above. Experiment with values of the control parameters of the GO procedure. Compute the coefficient of determination for the obtained regression model and comment on the model accuracy. Document your work.

4.5 Dynamic Programming

Many physical, managerial, and controlled processes could be considered as a sequence of relatively independent but interrelated stages. This division, natural or imaginative, could be performed in the spatial, functional, or temporal domains. The following diagram in Fig. 4.13 represents a typical multi-stage process containing four stages. Every stage or sub-process is relatively independent in the sense that it is characterized by its own (local) input $\boldsymbol{x}_i$, local output $\boldsymbol{y}_i$, local control

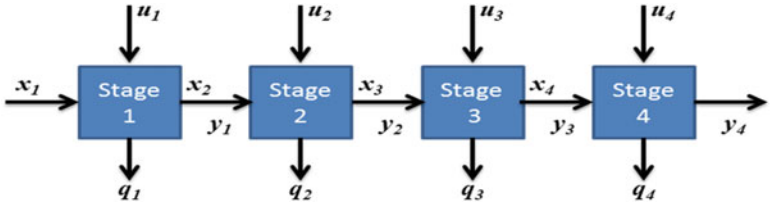


Fig. 4.13 Multi-stage process with four stages

effort $\boldsymbol{u}_{\mathrm{i}}$, and the local goodness criterion $\boldsymbol{q}_i$. Both the output and the criterion of each stage (sub-process) are defined by its local input and the control effort, i.e. $\boldsymbol{y}_i = \boldsymbol{y}_i(\boldsymbol{x}_i, \boldsymbol{u}_i)$ and $\boldsymbol{q}_i = \boldsymbol{y}_i(\boldsymbol{x}_i, \boldsymbol{u}_i)$.

At the same time, individual stages (sub-processes) are interrelated. Indeed the output of every stage, except the last ($\boldsymbol{n}$-th) stage, serves as the input of the consequent stage, i.e. for $\boldsymbol{i} = 1, 2, 3, \ldots, \boldsymbol{n} - 1$ $\boldsymbol{y}_i = \boldsymbol{x}_{i+1}$. This reality results in the following relationships that links the entire sequence:

$$\begin{aligned}\boldsymbol{y}_i &= \boldsymbol{y}_i(\boldsymbol{x}_i, \boldsymbol{u}_i) = \boldsymbol{y}_i[\boldsymbol{y}_{i-1}(\boldsymbol{x}_{i-1}, \boldsymbol{u}_{i-1}), u_i] = \boldsymbol{y}_i(\boldsymbol{x}_{i-1}, \boldsymbol{u}_{i-1}, \boldsymbol{u}_i) \\ &= \boldsymbol{y}_i[\boldsymbol{y}_{i-2}(\boldsymbol{x}_{i-2}, \boldsymbol{u}_{i-2}), \boldsymbol{u}_{i-1}, \boldsymbol{u}_i] = \boldsymbol{y}_i(\boldsymbol{x}_{i-2}, \boldsymbol{u}_{i-2}, \boldsymbol{u}_{i-1}, \boldsymbol{u}_i) = \ldots \\ &= \boldsymbol{y}_i(\boldsymbol{x}_1, \boldsymbol{u}_1, \boldsymbol{u}_2, \boldsymbol{u}_3, \ldots, \boldsymbol{u}_i)\end{aligned}$$

and similarly $\boldsymbol{q}_i = \boldsymbol{q}_i(\boldsymbol{x}_1, \boldsymbol{u}_1, \boldsymbol{u}_2, \boldsymbol{u}_3, \ldots, \boldsymbol{u}_i)$, where $\boldsymbol{i} = 1,2,3,\ldots, \boldsymbol{n}$ is the sequential number of the stage.

These relationships indicate that the output and criterion value of any stage of the process, except the first stage, are defined by the input of the first stage, control effort applied at this stage and control efforts applied at all previous stages. In addition to the above relationships, the stages of the process are linked by the "overall goodness criterion" defined as the sum of all "local" criteria, $Q = \sum_{k=1}^{n} q(x_k, u_i)$ where $\boldsymbol{n}$ is the total number of the stages. It could be seen that the overall criterion depends on the input of the first stage and all control efforts, i.e.

$$\boldsymbol{Q} = \boldsymbol{Q}(\boldsymbol{x}_1, \boldsymbol{u}_1, \boldsymbol{u}_2, \boldsymbol{u}_3, \ldots, \boldsymbol{u}_n)$$

Therefore the optimization problem of a multistage process implies the minimization (maximization) of the overall criterion $\boldsymbol{Q}(.)$ with respect to control efforts applied at individual stages, $\boldsymbol{u}_k$, $k = 1, 2, \ldots \boldsymbol{n}$, for any given input of the first stage, $\boldsymbol{x}_1$, and may be subject to some constraints imposed on the outputs of the individual stages, $\boldsymbol{y}_k$, $k = 1, 2, \ldots \boldsymbol{n}$. One can realize that the process optimization problem cannot be solved by the independent optimization of the individual stages with respect to their "local" criteria, $\boldsymbol{q}_k$, $k = 1, 2, \ldots \boldsymbol{n}$. The optimal control strategy must be "wise": "local" optimization of any sub-process may result in such an output that will completely jeopardize the operation of the consequent stages thus causing poor operation of the entire multistage process. Therefore, optimization of

any stage of a multi-stage process must take into account the consequences of this optimization for all consequent stages. Selection of any "local" control effort cannot be performed without assessing its impact on the overall criterion.

Dynamic programming is an optimization technique intended for the optimization of multi-stage processes. It is based on the fundamental principle of optimality of dynamic programming formulated by Richard Bellman. ***A problem is said to satisfy the Principle of Optimality if the sub-solutions of an optimal solution of the problem are themselves optimal solutions for their sub-problems.*** Fortunately, optimization problems of multi-stage processes do satisfy the Principle of Optimality that offers a powerful solution approach in the most realistic situations. The key to the application of the Principle of Optimality is in the following statement that is stemming from this principle: ***any last portion of an optimal sequence of steps is optimal***.

Let us illustrate this principle using the chart below in Fig. 4.14 that presents a process comprising of 12 sequential stages divided into two sections, AB and BC. It is assumed that each j-th stage of this process is characterized by its "local" criterion, $\boldsymbol{q}_j$. Assume that the overall criterion of the process is defined as the sum of local criteria: $Q = \sum_{j=1}^{12} q_j$

Let us define the sectional criteria for each of the two sections: $Q_{AB} = \sum_{j=1}^{5} q_j$ and $Q_{BC} = \sum_{j=6}^{12} q_j$. Assume that for every stage of the process some control effort is chosen, such that the entire combination of these control efforts, $\boldsymbol{u}_j^{OPT}$, $j = 1, 2, \ldots, 12$, optimizes (minimizes) the overall process criterion Q. Then according to the principle of dynamic programming control efforts $\boldsymbol{u}_j^{OPT}$, $j = 6, 7, \ldots, 12$ optimize the last section of the sequence, namely BC, thus bringing criterion $Q_{BC} = \sum_{j=6}^{12} q_j$ to its optimal (minimal) value. At the same time, control efforts $\boldsymbol{u}_j^{OPT}$, $j = 1, 2, \ldots, 5$ are not expected to optimize section AB of the process, thus criterion $Q_{AB} = \sum_{j=6}^{12} q_j$ could be minimized by a completely different combination of control efforts, say $\boldsymbol{u}_j^{ALT}$, $j = 1, 2, \ldots, 5$.

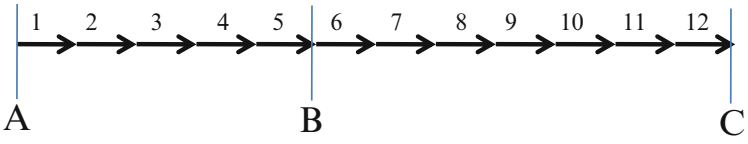


Fig. 4.14 Twelve stage process

The fundamental principle provides the framework for a highly efficient and versatile optimization procedure of dynamic programming that works on a step-by-step basis and defines optimal control efforts for individual stages of the multi-stage process. It is important that control decisions made at each step of the procedure do not optimize individual stages of the process, i.e. do not solve the "local" optimization problems. Instead, they optimize the ***last portion*** of the entire process that starts at the stage in question and end at the last stage of the process.

When doing so, every step of the optimization procedure takes into account not only the particular stage of the process but also all consequent stages. The procedure is iterative, therefore it shall start from the last section of the multistage process where there are no consequent stages to be considered. At the same time, the optimal solution of the control problem $\boldsymbol{u}_j^{OPT}$, cannot be explicitly defined without knowing the input $\boldsymbol{x}_j$ applied to the appropriate section of the process. Therefore, the dynamic programming procedure is performed in two steps: conditional optimization and unconditional optimization. Conditional optimization starts from the end of the process addressing the last stage of the process first, then the last two stages of the process, then the last three stages, and finally the entire process. Why is it called conditional?—because at the first step, the procedure defines the optimal conditional control effort (OCCE) for the last stage of the process that is dependent on the input of the last stage of the process:

$$\boldsymbol{u}_N^{OPT} = F(\boldsymbol{x}_N)$$

that minimizes the sectional criterion

$$\boldsymbol{Q}_N(\boldsymbol{x}_N, \boldsymbol{u}_N) = \boldsymbol{q}_N(\boldsymbol{x}_N, \boldsymbol{u}_N)$$

(Note that the sectional criterion is marked by the index of the first stage of the section). Now the output of the last stage of the process (and the output of the entire process) is

$$\boldsymbol{y}_N = \boldsymbol{y}_N\left(\boldsymbol{x}_N,\ \boldsymbol{u}_N^{OPT}\right)$$

This solution must be consistent with the required (allowed) value of the output of the process, $\boldsymbol{Y}^*$, i.e.

$$\boldsymbol{y}_N = \boldsymbol{y}_N\left(\boldsymbol{x}_N,\ \boldsymbol{u}_N^{OPT}\right) = \boldsymbol{Y}^*$$

At the next step the OCCE for the second stage from the end of the process is defined as a function of the input applied to this stage:

$$\boldsymbol{u}_{N-1}^{OPT} = F(\boldsymbol{x}_{N-1})$$

that minimizes the sectional criterion for the last two stages of the process:

$$\boldsymbol{Q}_{N-1}\left(\boldsymbol{x}_{N-1}, \boldsymbol{u}_{N-1}, \boldsymbol{u}_N^{OPT}\right) = \boldsymbol{q}_{N-1}(\boldsymbol{x}_{N-1}, \boldsymbol{u}_{N-1}) + \boldsymbol{q}_N\left(\boldsymbol{x}_N, \boldsymbol{u}_N^{OPT}\right)$$

Note that in this expression $\boldsymbol{x}_N$ does not work as an independent factor, it is defined as the output of the previous stage of the process:

$$\boldsymbol{x}_N = \boldsymbol{y}_{N-1}\left(\boldsymbol{x}_{N-1}, \boldsymbol{u}_{N-1}^{OPT}\right)$$

and therefore criterion Q_{N-1} actually depends only on two variable factors, $\boldsymbol{x}_{N-1}$ and $\boldsymbol{u}_{N-1}$:

$$\boldsymbol{Q}_{N-1}(\boldsymbol{x}_{N-1}, \boldsymbol{u}_{N-1}) = \boldsymbol{q}_{N-1}(\boldsymbol{x}_{N-1}, \boldsymbol{u}_{N-1}) + \boldsymbol{q}_N\left[\boldsymbol{y}_{N-1}(\boldsymbol{x}_{N-1}, \boldsymbol{u}_{N-1}), \boldsymbol{u}_N^{OPT}\right]$$

The solution must ensure that the resultant output

$$\boldsymbol{y}_{N-1} = \boldsymbol{y}_{N-1}\left(\boldsymbol{x}_{N-1},\ \boldsymbol{u}_{N-1}^{OPT}, \boldsymbol{u}_N^{OPT}\right)$$

is within its allowed limits, i.e.

$$\boldsymbol{y}_{N-1}{}^{\text{MIN}} \leq \boldsymbol{y}_{N-1} \leq \boldsymbol{y}_{N-1}{}^{\text{MAX}}$$

Now let us define the OCCE for the third stage from the end of the process as a function of the input applied to this stage:

$$\boldsymbol{u}_{N-2}^{OPT} = F(\boldsymbol{x}_{N-2})$$

that minimizes the sectional criterion that "covers" the last three stages:

$$\begin{aligned}\boldsymbol{Q}_{N-2}\left(\boldsymbol{x}_{N-2}, \boldsymbol{u}_{N-2}, \boldsymbol{u}_{N-1}^{OPT}, \boldsymbol{u}_N^{OPT}\right) = \boldsymbol{q}_{N-2}(\boldsymbol{x}_{N-2}, \boldsymbol{u}_{N-2}) + \boldsymbol{q}_{N-1}\left(\boldsymbol{x}_{N-1}, \boldsymbol{u}_{N-1}^{OPT}\right) \\ + \boldsymbol{q}_N\left(\boldsymbol{x}_N, \boldsymbol{u}_N^{OPT}\right)\end{aligned}$$

Again, in this expression $\boldsymbol{x}_{N-1}$ and $\boldsymbol{x}_N$ are not independent factors, they are defined as outputs of the previous stages of the process:

$$\boldsymbol{x}_{N-1} = \boldsymbol{y}_{N-2}(\boldsymbol{x}_{N-2}, \boldsymbol{u}_{N-2}) \text{ and } \boldsymbol{x}_N = \boldsymbol{y}_{N-1}\left(\boldsymbol{x}_{N-1}, \boldsymbol{u}_{N-1}^{OPT}\right)$$

and therefore criterion $\boldsymbol{Q}_{N-2}$ actually depends only on two variable factors, $\boldsymbol{x}_{N-2}$ and $\boldsymbol{u}_{N-2}$

$$\begin{aligned}\boldsymbol{Q}_{N-2} = \boldsymbol{q}_{N-2}(\boldsymbol{x}_{N-2}, \boldsymbol{u}_{N-2}) + \boldsymbol{q}_{N-1}\left[\boldsymbol{y}_{N-2}\left(\boldsymbol{x}_{N-2}, \boldsymbol{u}_{N-2}^{OPT}\right), \boldsymbol{u}_{N-1}^{OPT}\right] \\ + \boldsymbol{q}_N\left[\boldsymbol{y}_{N-1}\left(\boldsymbol{x}_{N-1}, \boldsymbol{u}_{N-1}^{OPT}\right), \boldsymbol{u}_N^{OPT}\right] = \boldsymbol{Q}_{N-2}\left(\boldsymbol{x}_{N-1}, \boldsymbol{u}_{N-2},\ \boldsymbol{u}_{N-1}^{OPT}, \boldsymbol{u}_N^{OPT}\right)\end{aligned}$$

The optimal value of this criterion is:

$$\boldsymbol{Q}_{N-2}\left(\boldsymbol{x}_{N-1}, \boldsymbol{u}_{N-2,}^{OPT},\ \boldsymbol{u}_{N-1,}^{OPT}\boldsymbol{u}_{N}^{OPT}\right)$$

Again, the appropriate output,

$$\boldsymbol{y}_{N-2} = \boldsymbol{y}_{N-2}\left(\boldsymbol{x}_{N-2},\ \boldsymbol{u}_{N-2}^{OPT}\right)$$

must be consistent with the allowed value for the output of the appropriate stage of the process:

$$\boldsymbol{y}_{N-2}{}^{\text{MIN}} \leq \boldsymbol{y}_{N-2} \leq \boldsymbol{y}_{N-2}{}^{\text{MAX}}$$

It could be seen that eventually the procedure defines the control effort for the first stage of the process as a function of the input applied to this stage:

$$\boldsymbol{u}_{1}^{OPT} = F(\boldsymbol{x}_1)$$

that minimizes the sectional criterion

$$\boldsymbol{Q}_1\left(\boldsymbol{x}_1, \boldsymbol{u}_1, \boldsymbol{u}_2^{OPT}, \boldsymbol{u}_3^{OPT}, \ldots,\ \boldsymbol{u}_{N-1}^{OPT}, \boldsymbol{u}_N^{OPT}\right)$$

However, the input of the first stage (the input of the overall process), $\boldsymbol{x}_1$, is explicitly known, therefore the control effort $\boldsymbol{u}_1^{OPT}$ could be explicitly defined. This results in the explicitly defined output of the first stage, $\boldsymbol{y}_1$. Since $\boldsymbol{x}_2 = \boldsymbol{y}_1$, the optimal conditional control effort

$$\boldsymbol{u}_{2}^{OPT} = F(\boldsymbol{x}_2)$$

could be explicitly defined thus resulting in an explicit definition of the output of the second stage and the input of the third stage, and so on... It could be seen that the procedure moves now from the first stage of the process to the last stage, converting conditional control efforts into explicitly defined unconditional optimal control efforts.

Let us consider the application of the outlined approach to the following numerical example representing the so-called optimal routing problem.

Example 4.7 Apply dynamic programming to establish the optimal ("minimum cost") route within the following graph in Fig. 4.15

It could be seen that the transportation problem featured by the above graph consists of five stages and four steps. Step 1 consists of four alternative transitions: 1/1 → 2/1, 1/1 → 2/2, 1/1 → 2/3 and 1/1 → 2/4 with the associated costs of 5, 3, 1, and 2 (units). Step 2 consists of 12 alternative transitions: 2/1 → 3/1, 2/1 → 3/2, 2/

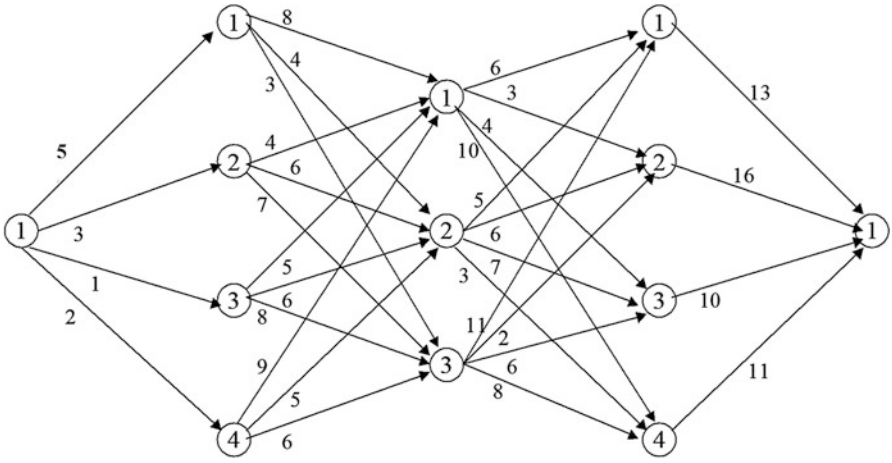


Fig. 4.15 Process graph

$1 \rightarrow 3/3$ with the associated costs of 8, 4, and 3 (units); $2/2 \rightarrow 3/1$, $2/2 \rightarrow 3/2$, $2/2 \rightarrow 3/3$ with the associated costs of 4, 6, and 7 (units); $2/3 \rightarrow 3/1$, $2/3 \rightarrow 3/2$, $2/3 \rightarrow 3/3$ with the associated costs of 5, 6, and 8 (units); and $2/4 \rightarrow 3/1$, $2/4 \rightarrow 3/2$, $2/4 \rightarrow 3/3$ with the associated costs of 9, 5, and 6 (units). Step 3 also consists of 12 alternative transitions: $3/1 \rightarrow 4/1$, $3/1 \rightarrow 4/2$, $3/1 \rightarrow 4/3$, $3/1 \rightarrow 4/4$ with the associated costs of 6, 3, 4, and 10 (units); $3/2 \rightarrow 4/1$, $3/2 \rightarrow 4/2$, $3/2 \rightarrow 4/3$, $3/2 \rightarrow 4/4$ with the associated costs of 5, 6, 7, and 3 (units); $3/3 \rightarrow 4/1$, $3/3 \rightarrow 4/2$, $3/3 \rightarrow 4/3$, $3/3 \rightarrow 4/4$ with the associated costs of 11, 2, 6, and 8 (units). Finally, the last step, 4, consists of four alternative transitions: $4/1 \rightarrow 5/1$, $4/2 \rightarrow 5/1$, $4/3 \rightarrow 5/1$ and $4/4 \rightarrow 5/1$ with the associated costs of 13, 16, 10, and 11 (units). It is required to establish such a sequence of transitions (optimal path) that would lead from the initial to the final stage (nodes of the above graph) and had the minimal sum of the transition costs.

Could we have established the optimal path by considering all possible alternative paths within this graph?—perhaps, but the required computational effort is expected to be very high. Should the number of stages and alternative transitions at every step be greater, this approach will become prohibitively formidable.

According to the dynamic programming procedure, let us define conditionally optimal transitions for the last step of the process, step #4. This task is quite simple: if the starting node of the stage #4 is 4/1 then the optimal (and the only) transition to the last stage is $4/1 \rightarrow 5/1$ with the cost of 13 units. Should we start from node 4/2, the optimal (and the only) transition is $4/2 \rightarrow 5/1$ with the cost of 16 units, and so on. The results of the conditional optimization of the step #4 are tabulated below

Conditional optimization of step 4

Starting node of the stage 4	Final node of the stage 5	Transition costs	Optimal transition	Total cost for this portion of the path
4/1	5/1	13	4/1 → 5/1	13
4/2	5/1	16	4/2 → 5/1	16
4/3	5/1	10	4/3 → 5/1	10
4/4	5/1	11	4/4 → 5/1	11

Let us compile the table representing conditional optimization of the last two steps of the transportation process, namely steps 3 and 4. Assuming that the starting node of the stage 3 is 3/1 then the first available transition within step 3 is 3/1 → 4/1 with the cost of 6 units. At the next step, this transition will be followed by 4/1 → 5/1 and the total cost of both transitions, 3/1 → 4/1 → 5/1, is 19 units. Then, consider the second available transition within step 3, 3/1 → 4/2. It comes with the cost of 3 units and must be followed by the transition 4/2 → 5/1 with the total cost of transition 3/1 → 4/2 → 5/1 of 19 units. Upon consideration of transitions 3/1 → 4/3 → 5/1 and 3/1 → 4/4 → 5/1 it could be seen that for 3/1 as the entry point to step 3 the best transition is 3/1 → 4/3 → 5/1 with the lowest total cost of 14 units.

Conditional optimization of steps 3 and 4

Starting point of steps 3	Alternative transitions to states	Transition costs	Possible transition	Total cost for two stages	Optimal transition
3/1	4/1	6	3/1 → 4/1 → 5/1	6 + 13 = 19	
	4/2	3	3/1 → 4/2 → 5/1	3 + 16 = 19	
	4/3	4	3/1 → 4/3 → 5/1	4 + 10 = 14	3/1 → 4/3 → 5/1
	4/4	10	3/1 → 4/4 → 5/1	10 + 11 = 21	
3/2	4/1	5	3/2 → 4/1 → 5/1	5 + 13 = 18	
	4/2	6	3/2 → 4/2 → 5/1	6 + 16 = 22	
	4/3	7	3/2 → 4/3 → 5/1	7 + 10 = 17	
	4/4	3	3/2 → 4/4 → 5/1	3 + 11 = 14	3/2 → 4/4 → 5/1
3/3	4/1	11	3/3 → 4/1 → 5/1	11 + 13 = 24	
	4/2	2	3/3 → 4/2 → 5/1	2 + 16 = 18	
	4/3	6	3/3 → 4/3 → 5/1	6 + 10 = 16	3/3 → 4/3 → 5/1
	4/4	8	3/3 → 4/4 → 5/1	8 + 11 = 19	

Now let us compile the table representing conditional optimization of the last three steps of the transportation process, namely step 2 followed by steps 3 and 4. Assume that the starting point of stage 2 is 2/1 and the first available transition is 2/1 → 3/1 with the cost of 8 units. The optimal transition from 3/1 to the last stage has been already established: 3/1 → 4/3 → 5/1 and its cost is 14 units, therefore the cost of transition 2/1 → 3/1 → 4/3 → 5/1 is 8 + 14 = 22 units. Assume that the starting point is 2/1 and the chosen transition is 2/1 → 3/2 with the cost of 4 units. The

already established optimal transition from 3/2 to the last stage is $3/2 \rightarrow 4/4 \rightarrow 5/1$ with the cost of 14 units, therefore the cost of transition $2/1 \rightarrow 3/2 \rightarrow 4/4 \rightarrow 5/1$ is $4 + 14 = 18$ units. Now assume that the starting point is still 2/1 and the chosen transition is $2/1 \rightarrow 3/3$ with the cost of 3 units. The already established optimal transition from 3/3 to the last stage is $3/3 \rightarrow 4/3 \rightarrow 5/1$ with the cost of 16 units and the total cost is $3 + 16 = 19$ units. This indicates that the optimal path from point 2/1 to 5/1 is $2/1 \rightarrow 3/2 \rightarrow 4/4 \rightarrow 5/1$ with the cost of 18 units. In the similar fashion optimal paths from points 2/2, 2/3 and 2/4 to point 5/1 are to be established. They are: $2/2 \rightarrow 3/2 \rightarrow 4/4 \rightarrow 5/1$ with the cost of 18 units, $2/3 \rightarrow 3/2 \rightarrow 4/4 \rightarrow 5/1$ with the cost of 19 units, and $2/4 \rightarrow 3/2 \rightarrow 4/4 \rightarrow 5/1$ with the cost of 18 units.

Conditional optimization of steps 2, 3 and 4

Starting point of step 2	Alternative transitions to states	Transition costs	Possible transition	Total cost for two stages	Optimal transition
2/1	3/1	8	$2/1 \rightarrow 3/1$	$8 + 14 = 22$	
	3/2	4	$2/1 \rightarrow 3/2$	$4 + 14 = 18$	$2/1 \rightarrow 3/2 \rightarrow 4/4 \rightarrow 5/1$
	3/3	3	$2/1 \rightarrow 3/3$	$3 + 16 = 19$	
2/2	3/1	4	$2/2 \rightarrow 3/1$	$4 + 14 = 18$	$2/2 \rightarrow 3/1 \rightarrow 4/3 \rightarrow 5/1$
	3/2	6	$2/2 \rightarrow 3/2$	$6 + 14 = 20$	
	3/3	7	$2/2 \rightarrow 3/3$	$7 + 16 = 23$	
2/3	3/1	5	$2/3 \rightarrow 3/1$	$5 + 14 = 19$	$2/3 \rightarrow 3/1 \rightarrow 4/3 \rightarrow 5/1$
	3/2	6	$2/3 \rightarrow 3/2$	$6 + 14 = 20$	
	3/3	8	$2/3 \rightarrow 3/3$	$8 + 16 = 24$	
2/4	3/1	9	$2/4 \rightarrow 3/1$	$9 + 14 = 23$	
	3/2	5	$2/4 \rightarrow 3/2$	$5 + 14 = 19$	$2/4 \rightarrow 3/2 \rightarrow 4/4 \rightarrow 5/1$
	3/3	6	$2/4 \rightarrow 3/3$	$6 + 16 = 22$	

Finally, let us compile the table representing optimization of all four steps of the transportation process. Note that the optimization results are not conditional anymore: the transition process is originated at the very particular point, 1/1. Assume that the first available transition is $1/1 \rightarrow 2/1$ with the cost of 5 units. The optimal transition from 2/1 to the last stage has been already established: $2/1 \rightarrow 3/2 \rightarrow 4/4 \rightarrow 5/1$ and its cost is 18 units, therefore the cost of transition $1/1 \rightarrow 2/1 \rightarrow 3/1 \rightarrow 4/3 \rightarrow 5/1$ is $5 + 18 = 23$ units. Assume that the chosen transition is $1/1 \rightarrow 2/2$ with the cost of 3 units. The already established optimal transition from 2/2 to the last stage is $2/2 \rightarrow 3/1 \rightarrow 4/3 \rightarrow 5/1$ with the cost of 18 units, therefore the cost of transition $1/1 \rightarrow 2/2 \rightarrow 3/1 \rightarrow 4/3 \rightarrow 5/1$ is $3 + 18 = 21$ units. Now assume that the chosen transition is $1/1 \rightarrow 2/3$ with the cost of 1 units. The already established optimal transition from 2/3 to the last stage is $2/3 \rightarrow 3/1 \rightarrow 4/3 \rightarrow 5/1$ with the cost of 19 units and the total cost of transition $1/1 \rightarrow 2/3 \rightarrow 3/1 \rightarrow 4/3 \rightarrow 5/1$ is $1 + 19 = 20$ units. Should the chosen transition be $1/1 \rightarrow 2/4$ with the cost of 2 units, and since the already established optimal transition from 2/4 to the last stage is $2/4 \rightarrow 3/$

$2 \rightarrow 4/4 \rightarrow 5/1$ with the cost of 19 units, the total cost of transition $1/1 \rightarrow 2/4 \rightarrow 3/2 \rightarrow 4/4 \rightarrow 5/1$ is $5 + 19 = 21$ units. This clearly indicates that the optimal path from point 1/1 to 5/1 is $1/1 \rightarrow 2/3 \rightarrow 3/1 \rightarrow 4/3 \rightarrow 5/1$. See this analysis summarized in the table below.

Optimization of steps 1, 2, 3 and 4

Starting point of step 1	Alternative transitions to states	Transition costs	Possible transition	Total cost for two stages	Optimal transition
1/1	2/1	5	$1/1 \rightarrow 2/1$	$5 + 18 = 23$	
	2/2	3	$1/1 \rightarrow 2/2$	$3 + 18 = 21$	
	2/3	1	$1/1 \rightarrow 2/3$	$1 + 19 = 20$	$1/1 \rightarrow 2/3 \rightarrow 3/1 \rightarrow 4/3 \rightarrow 5/1$
	2/4	2	$1/1 \rightarrow 2/4$	$2 + 19 = 21$	

Consider another quite practical example that ideally lends itself to the application of dynamic programming. It is the optimization of a sequence of manufacturing processes that could be found in chemistry and metallurgy. Each process has its own mathematical description representing quality/quantity of its end product and manufacturing costs as functions of the characteristics of the raw material $\boldsymbol{x}_\mathrm{i}$ and control efforts $\boldsymbol{u}_\mathrm{i}$. Consider the mathematical model of i-th manufacturing process within a sequence consisting of N processes:

characteristic of the end material $\boldsymbol{y}_\mathrm{i} = \boldsymbol{y}_\mathrm{i}(\boldsymbol{x}_\mathrm{i},\boldsymbol{u}_\mathrm{i})$, $\mathrm{i} = 1,2,\ldots,N$
manufacturing cost $\boldsymbol{q}_\mathrm{i} = \boldsymbol{q}_\mathrm{i}(\boldsymbol{x}_\mathrm{i},\boldsymbol{u}_\mathrm{i})$, $\mathrm{i} = 1,2,\ldots,N$
quality/quantity requirements $\boldsymbol{y}_\mathrm{i}^{\mathrm{MIN}} \leq \boldsymbol{y}_\mathrm{i} \leq \boldsymbol{y}_\mathrm{i}^{\mathrm{MAX}}$, $\mathrm{i} = 1,2,\ldots,N$
connection to neighboring processes $\boldsymbol{y}_\mathrm{i} = \boldsymbol{x}_{\mathrm{i}+1,}$ $\mathrm{i} = 1,2,\ldots,N$

For simplicity, let us assume that the above functions are scalar and are represented on the basis of their mathematical model by numerical values of $\boldsymbol{y}_\mathrm{i}$ and $\boldsymbol{u}_\mathrm{i}$ for discretized $\boldsymbol{x}_\mathrm{i} = k \cdot \Delta\boldsymbol{x}_\mathrm{i}$ and $\boldsymbol{u}_\mathrm{i} = m \cdot \Delta\boldsymbol{u}_\mathrm{i}$, i.e. $\boldsymbol{y}_\mathrm{i}(k,m) = \boldsymbol{y}_\mathrm{i}(k \cdot \Delta\boldsymbol{x}_\mathrm{i},m \cdot \Delta\boldsymbol{u}_\mathrm{i})$ and $\boldsymbol{q}_\mathrm{i}(k,m) = \boldsymbol{q}_\mathrm{i}(k \cdot \Delta\boldsymbol{x}_\mathrm{i},m \cdot \Delta\boldsymbol{u}_\mathrm{i})$ where k, $m = 1,2,3,\ldots$. Does this representation of the manufacturing process result in the loss of accuracy? No, providing that the discretization steps $\Delta\boldsymbol{x}_\mathrm{i}$, $\Delta\boldsymbol{u}_\mathrm{i}$ are judiciously chosen.

Example 4.8 Apply dynamic programming to optimize the operation of a sequence of three manufacturing processes represented by the tabulated description below. Note that the inputs of the individual processes are defined in % assuming that the 100 % value of the respective input corresponds to the maximum value of the output of the previous process. To simplify the problem further, the control efforts are defined not by real numbers, but as "control options." The overall cost of manufacturing is defined as the sum of costs of individual processes. Finally, it could be seen that the specified acceptability limits of the process outputs are different from their feasibility limits that could be seen in the tables.

	PROCESS #1					
	Output $y_1(x,u)$, $30 \geq y_1 \geq 10$			Cost $q_1(x,u)$		
	U=			U=		
X%	1	2	3	1	2	3
$33 \geq x \geq 0 \rightarrow$	2.000	50.000	18.000	21.000	16.000	73.000
$66 \geq x \geq 34 \rightarrow$	9.000	13.000	19.000	27.000	19.000	130.000
$100 \geq x \geq 67 \rightarrow$	11.000	19.000	31.000	29.000	18.000	21.000

	PROCESS #2					
	Output $y_2(x,u)$, $70 \geq y_2 \geq 20$			Cost $q_2(x,u)$		
	U=			U=		
X%	1	2	3	1	2	3
$33 \geq x \geq 0 \rightarrow$	8.000	11.000	21.000	70.000	76.000	100.000
$66 \geq x \geq 34 \rightarrow$	24.000	13.000	90.000	61.000	64.000	92.000
$100 \geq x \geq 67 \rightarrow$	1.000	15.000	35.000	55.000	77.000	88.000

	PROCESS #3					
	Output $y_3(x,u)$, $30 \geq y_3 \geq 15$			Cost $q_3(x,u)$		
	U=			U=		
X%	1	2	3	1	2	3
$33 \geq x \geq 0 \rightarrow$	12.000	50.000	18.000	21.000	16.000	73.000
$66 \geq x \geq 34 \rightarrow$	9.000	13.000	19.000	27.000	29.000	130.000
$100 \geq x \geq 67 \rightarrow$	16.000	19.000	31.000	29.000	28.000	21.000

First, let us address the issue of acceptability limits of the process outputs. Computationally, it could be done by replacing associate cost values by penalties (10^{15}) in the situations when output values are not acceptable—this will automatically exclude some cases from consideration, see the modified tables below

	PROCESS #1					
	Output $y_1(x,u)$, $30 \geq y_1 \geq 10$			Cost $q_1(x,u)$		
	U=			U=		
X%	1	2	3	1	2	3
$33 \geq x \geq 0 \rightarrow$	2.000	50.000	18.000	.10E+16	.10E+16	73.000
$66 \geq x \geq 34 \rightarrow$	9.000	13.000	19.000	.10E+16	19.000	130.000
$100 \geq x \geq 67 \rightarrow$	11.000	19.000	31.000	29.000	29.000	.10E+16

	PROCESS #2					
	Output $y_2(x,u)$, $70 \geq y_2 \geq 20$			Cost $q_2(x,u)$		
	U=			U=		
X%	1	2	3	1	2	3
$33 \geq x \geq 0 \rightarrow$	8.000	11.000	21.000	.10E+16	.10E+16	100.000
$66 \geq x \geq 34 \rightarrow$	24.000	13.000	90.000	61.000	.10E+16	.10E+16
$100 \geq x \geq 67 \rightarrow$	1.000	15.000	35.000	.10E+16	.10E+16	88.000

	PROCESS #3					
	Output $y_3(x,u)$, $30 \geq y_3 \geq 15$			Cost $q_3(x,u)$		
	U=			U=		
X%	1	2	3	1	2	3
$33 \geq x \geq 0 \rightarrow$	12.000	50.000	18.000	.10E+16	.10E+16	73.000
$66 \geq x \geq 34 \rightarrow$	9.000	13.000	19.000	.10E+16	.10E+16	130.000
$100 \geq x \geq 67 \rightarrow$	16.000	19.000	31.000	29.000	28.000	.10E+16

The following analysis of the problem solution is based on the printout of a specially written computer code. According to the Principle of Optimality, the solution starts from the conditional optimization of the last, third, process. It will provide an optimal recipe for the process operation for every possible grade of the process input. The printout below considers application of various control options when the input of the process is between 0 and 33 % of its maximum attainable value (grade 1). It could be seen that the acceptable value of the process output is obtained only when control option #3 is applied. This defines option #3 as the conditional optimal control option, and the associated cost of 73 units as the conditionally minimal cost.

```
          PROCESS # 3
INP # 1 CONTR# 1 Q= .10000E+16 Y= 12.00
INP # 1 CONTR# 2 Q= .10000E+16 Y= 50.00
INP # 1 CONTR# 3 Q= .73000E+02 Y= 18.00
     OPT: INP # 1, CONTR# 3, QSUM= .73000E+02, Y= 18.00
```

The following printout presents similar results for the situations when the input grade is 2 and 3.

```
INP # 2 CONTR# 1 Q= .10000E+16 Y= 9.00
INP # 2 CONTR# 2 Q= .10000E+16 Y= 13.00
INP # 2 CONTR# 3 Q= .13000E+03 Y= 19.00
     OPT: INP # 2, CONTR# 3 QSUM= .13000E+03, Y= 19.00
INP # 3 CONTR# 1 Q= .29000E+02 Y= 16.00
INP # 3 CONTR# 2 Q= .28000E+02 Y= 19.00
INP # 3 CONTR# 3 Q= .10000E+16 Y= 31.00
  OPT: INP # 3, CONTR# 2 QSUM= .28000E+02, Y= 19.00
```

Now consider conditional optimization of process #2. Note that QSUM represents the sum of costs associated with the chosen input and control option of process #2 and the consequent conditionally optimal input + control option of process #3. Consider the application of various control options when the input of process #2 is of grade 1 (i.e. between 0 and 33 % of its maximum attainable value). The resultant QSUM value includes the specific cost at the process #2 and the consequent already known optimal cost at process #3. Since the first two control options are penalized for resulting in unacceptable values of the output, the optimal result is offered by the control option #3 and the accumulated cost value is QSUM = 73 + 100 = 173 units. Some additional information seen in the printout addresses the following issue. Note that the action at step #2 has resulted in $y_2 = 21$ units, then how does one determine the consequent action at process #3? It could be seen that the highest y_2 value in the output of process #2 is 90 units. Therefore the output value $y_2 = 21$ falls within 0–33 % of the y_2 range, i.e. $y_2 = 21$ constitutes grade #1 of the input product for process #3. Based on the conditional optimization of process #3, for the input grade #1 control option #1 with the associate cost of 73 units is optimal (see Y = 21.00 => 1 + (.73000E + 02) QSUM = .17300E + 03)

```
                    PROCESS # 2
INP  #  1  CONTR#  1  Q=.10000E+16  Y=  8.00  =>1+(.73000E+02)
QSUM=.10000E+16
INP  #  1  CONTR#  2  Q=.10000E+16  Y=  11.00  =>1+(.73000E+02)
QSUM=.10000E+16
INP  #  1  CONTR#  3  Q=.10000E+03  Y=  21.00  =>1+(.73000E+02)
QSUM=.17300E+03
        OPT: INP # 1, CONTR# 3, QSUM=.17300E+03, Y= 21.00 ==>1
```

Similar analysis is conducted to perform conditional optimization of process #2 for two other grades of the input.

```
INP  #  2  CONTR#  1  Q=.61000E+02  Y=  24.00  =>1+(.73000E+02)
QSUM=.13400E+03
INP  #  2  CONTR#  2  Q=.10000E+16  Y=  13.00  =>1+(.73000E+02)
QSUM=.10000E+16
INP  #  2  CONTR#  3  Q=.10000E+16  Y=  90.00  =>3+(.28000E+02)
QSUM=.10000E+16
  OPT: INP # 2, CONTR# 1, QSUM=.13400E+03, Y= 24.00 ==>1
INP  #  3  CONTR#  1  Q=.10000E+16  Y=  1.00  =>1+(.73000E+02)
QSUM=.10000E+16
INP  #  3  CONTR#  2  Q=.10000E+16  Y=  15.00  =>1+(.73000E+02)
QSUM=.10000E+16
INP  #  3  CONTR#  3  Q=.88000E+02  Y=  35.00  =>2+(.13000E+03)
QSUM=.21800E+03
  OPT: INP # 3, CONTR# 3, QSUM=.21800E+03, Y= 35.00 ==>2
```

Consider conditional optimization of process #1, that results in the optimization of the entire combination of three sequential processes. Consider the application of various control options when the input of process #1 is of grade 2 (i.e. between 34 and 66 % of its maximum attainable value). The resultant QSUM value includes the

specific cost at the process #1 and the consequent already known optimal costs at process #2 and #3. The first control option results in the unacceptable value of the output and is penalized. The application of control option #2 results in $y_1 = 13$ or #1 grade of the input for process #2, and the cost of 19 units. The already established optimal decisions for this input grade for process #2 come with the cost of 173 units. Consequently QSUM $= 19 + 173 = 192$ units. The application of control option #3 results in $y_1 = 19$ (or #2 grade of the input for process #2), and the cost of 130 units. The already established optimal decisions for this input grade for process #2 comes with the cost of 134 units. Therefore QSUM $= 130 + 134 = 264$ units. It is clear that the control option #2 is optimal grade #2 of the input material.

```
         PROCESS # 1
INP # 2 CONTR#  1  Q=.10000E+16  Y=    9.00 =>1+(.17300E+03)
QSUM=.10000E+16
INP # 2 CONTR#  2  Q=.19000E+02  Y=   13.00 =>1+(.17300E+03)
QSUM=.19200E+03
INP # 2 CONTR#  3  Q=.13000E+03  Y=   19.00 =>2+(.13400E+03)
QSUM=.26400E+03
        OPT: INP # 2, CONTR# 2, QSUM=.19200E+03, Y=  13.00 ==>1
```

Consider conditional optimization of process #1 when the input of process #1 is of grade #1 and grade #3 is featured below.

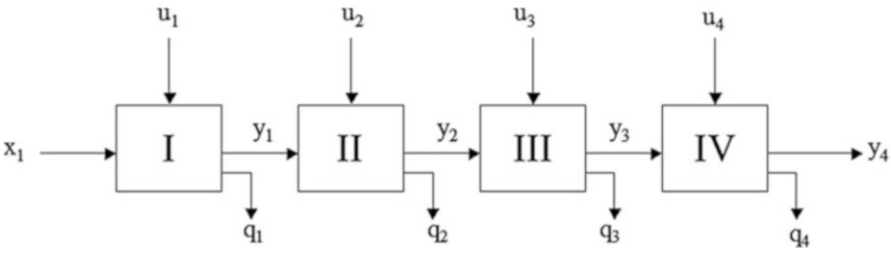
```
INP # 1 CONTR#  1  Q=.10000E+16  Y=    2.00 =>1+(.17300E+03)
QSUM=.10000E+16
INP # 1 CONTR#  2  Q=.10000E+16  Y=   50.00 =>3+(.21800E+03)
QSUM=.10000E+16
 INP # 1 CONTR#  3  Q=.73000E+02  Y=   18.00 =>2+(.13400E+03)
QSUM=.20700E+03
   OPT: INP # 1, CONTR# 3, QSUM=.20700E+03, Y=  18.00 ==>2
INP # 3 CONTR#  1  Q=.29000E+02  Y=   11.00 =>1+(.17300E+03)
QSUM=.20200E+03
INP # 3 CONTR#  2  Q=.18000E+02  Y=   19.00 =>2+(.13400E+03)
QSUM=.15200E+03
INP # 3 CONTR#  3  Q=.10000E+16  Y=   31.00 =>2+(.13400E+03)
QSUM=.10000E+16
   OPT: INP # 3, CONTR# 2, QSUM=.15200E+03, Y=  19.00 ==>2
```

Finally, the following printout summarizes the results of the optimization of the entire sequence of three processes for every grade of the raw material.

```
   OPTIMAL PROCESS OPERATION
RAW MATERIAL GRADE:     1      2      3
   PROCESS # 1
CONTROL OPTION:         3      2      2
  OUTPUT =           18.00  13.00  19.00
   PROCESS # 2
CONTROL OPTION:         3      1      3
  OUTPUT =           21.00  24.00  35.00
   PROCESS # 3
CONTROL OPTION:         3      3      2
  OUTPUT =           18.00  19.00  19.00
  ================================
 TOTAL COST:         207.00  192.00  152.00
```

4.5.1 Exercise 4.3

Problem 1 Apply dynamic programming to optimize the following sequence of manufacturing processes.



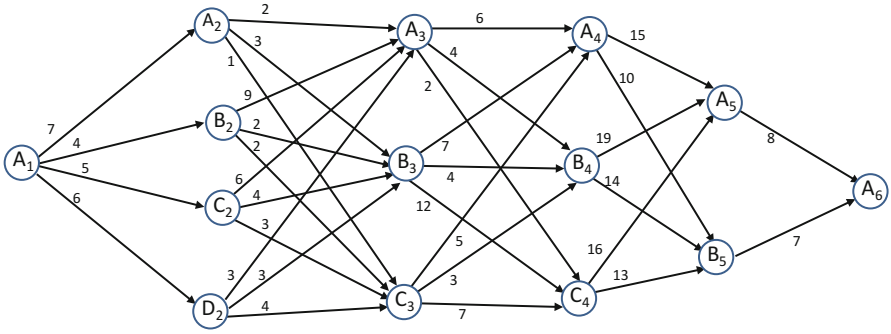
The characteristics of each process are given below:

	$y_1(x,u)$			$q_1(x,u)$			$y_2(x,u)$			$q_2(x,u)$		
	$u=1$	$u=2$	$u=3$	$u=1$	$u=2$	$u=3$	$u=1$	$u=2$	$u=3$	$u=1$	$u=2$	$u=3$
$10<x\leq 40$	25	45	55	25	28	25	65	44	74	13	21	33
$40<x\leq 70$	37	48	63	27	33	27	66	50	81	15	22	37
$70<x\leq 100$	45	58	79	22	24	25	78	62	96	18	28	40

	$y_3(x,u)$			$q_3(x,u)$			$y_4(x,u)$			$q_4(x,u)$		
	$u=1$	$u=2$	$u=3$	$u=1$	$u=2$	$u=3$	$u=1$	$u=2$	$u=3$	$u=1$	$u=2$	$u=3$
$10<x\leq 40$	13	45	92	16	18	9	56	85	97	2	4	3
$40<x\leq 70$	48	18	68	13	17	8	42	61	81	3	6	4
$70<x\leq 100$	81	66	21	10	14	6	21	39	70	4	5	3

It is known that $x_1 = 37$ (units) and the end product must be such that $70 \leq y_4 \leq 85$. Obtain the optimal choice of control options for each process that would minimize the sum of "local" criteria, $Q = q_1 + q_2 + q_3 + q_4$, and define the corresponding values of the characteristics of the intermediate products.

Problem 2 Use dynamic programming to solve the optimal routing problem based on the graph below featuring the available transitions and the associated costs.



Solutions

Exercise 4.1: Problem 1

The first constraint reflects the requirement on the total weight of the mixture:

$$x_1 + x_2 + x_3 + x_4 + x_5 + x_6 = 500$$

The following expressions represent the required concentrations of each chemical ingredient in the final mixture:

$$Fe = .15 \cdot x_1 + .40 \cdot x_2 + .35 \cdot x_3 + .16 \cdot x_4 + .33 \cdot x_5 + .07 \cdot x_6 \geq .20 \cdot 500$$

$$Zn = .38 \cdot x_1 + .12 \cdot x_2 + .05 \cdot x_3 + .11 \cdot x_4 + .01 \cdot x_5 + .23 \cdot x_6 \geq .10 \cdot 500$$

$$SiO_2 = .41 \cdot x_1 + .40 \cdot x_2 + .27 \cdot x_3 + .21 \cdot x_4 + .60 \cdot x_5 + .45 \cdot x_6 \leq .42 \cdot 500$$

$$Cu = .06 \cdot x_1 + .01 \cdot x_2 + .28 \cdot x_3 + .18 \cdot x_4 + .05 \cdot x_5 + .25 \cdot x_6 \geq .05 \cdot 500$$

The feasibility constraints reflect the availability of the materials. The amount of each material used is equal to the percentage of that material multiplied by the total weight of the end mixture. This must be no greater than the available weight of the material.

$$x_1 \leq 250$$

$$x_2 \leq 590$$

$$x_3 \leq 1000$$

$$x_4 \leq 520$$

$$x_5 \leq 2500$$

$$x_6 \leq 800$$

It should be noted that all variables of this problem are non-negative, but this requirement is very common for linear programming problems and is addressed by the solution algorithm.

The "minimum cost" requirement is addressed as follows:

$$120 \cdot x_1 + 150 \cdot x_2 + 211 \cdot x_3 + 140 \cdot x_4 + 75 \cdot x_5 + 214 \cdot x_6 \rightarrow Min$$

The problem solution is obtained by the use of the linear programming software available in MATLAB. The optimal weight of each material in tons is:

x_1	x_2	x_3	x_4	x_5	x_6	Total	Cost,$
68.2363	0.0000	0.0000	197.5259	234.2378	0.0000	500.0000	53409.80

and the chemical composition of the mixture is

$Fe\%$	$Zn\%$	$SiO_2\%$	$Cu\%$
23.83	10.00	42.00	10.27

Exercise 4.1: Problem 2

For this problem, we were required to use the gradient-based LSM procedure to find the optimal solution of the a coefficients in the following equation.

$$y^{mod}(k) = a_1 \cdot x_1(k) + a_2 \cdot x_2(k) + a_3 \cdot x_3(k) + a_4 \cdot x_4(k)$$

The method for the gradient-based LSM is a simple iterative procedure which "moves" the point representing unknown coefficients in four-dimensional space in the direction toward the minimum value of the criterion. In this case, the criterion, Q, is calculated as the sum of squared values of the discrepancy $e(k) = y(k) - y^{mod}(k)$:

$$A_{new} = \begin{bmatrix} A(1) - \gamma \cdot \dfrac{\Delta Q[A(1),A(2),A(3),A(4)]}{\Delta(1)} \\ A(2) - \gamma \cdot \dfrac{\Delta Q[A(1),A(2),A(3),A(4)]}{\Delta(2)} \\ A(3) - \gamma \cdot \dfrac{\Delta Q[A(1),A(2),A(3),A(4)]}{\Delta(3)} \\ A(4) - \gamma \cdot \dfrac{\Delta Q[A(1),A(2),A(3),A(4)]}{\Delta(4)} \end{bmatrix}$$

where $\dfrac{\Delta Q[A(1),A(2),A(3),A(4)]}{\Delta(i)}$ are estimated partial derivatives of the LSM criterion Q with respect to particular coefficients ($i = 1,2,3,4$) chosen to be 0.0001, and $\gamma > 0$, is a scaling factor. Initially, γ is chosen to be 0.02, however, in the case of an unsuccessful step leading to an increase of criterion Q criterion

instead of a decrease, the magnitude of gamma is cut in half. This ensures that the procedure will converge.

The results of this procedure were reached after 250 iterations, starting with zero initial conditions. This procedure could be less accurate but also faster if the termination conditions were made less strict. For this termination condition, the change between the newly generated Q and the previous Q needs to be less than .0000001 in absolute. The optimal result was:

$$A = \begin{bmatrix} 2.0001 \\ 2.9918 \\ -2.0001 \\ 5.0235 \end{bmatrix}$$

Since the coefficient of determination for this model is 0.9996, this is an excellent model of our linear system.

Exercise 4.1: Problem 3

This problem differs from the previous one because of the additional requirement: all model parameters are to be positive. This condition is achieved by the use of penalty functions added to the original LSM criterion Q. In this problem the criterion to be minimized is:

$$Q_1 = Q + \sum_{i=1}^{4} P_i$$

where $Pi = \begin{cases} 0 \text{ if } A(i) \geq 0 \\ 10^{10} \cdot A(i)^2 \text{ if } A(i) < 0 \end{cases}$, i = 1,2,3,4

The optimal result, with a coefficient of determination of 0.9348, reached after 300 iterations was:

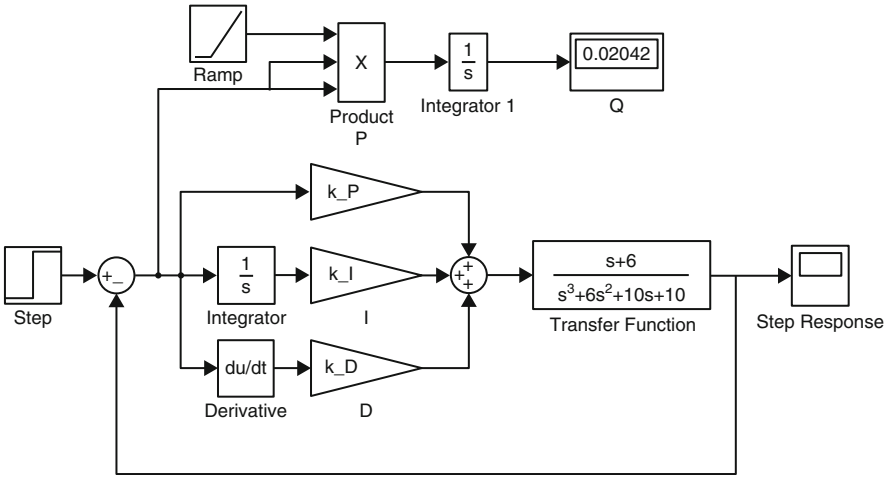
$$A = \begin{bmatrix} 1.0790 \\ 2.4456 \\ .0009 \\ 4.4795 \end{bmatrix}$$

Comparing the coefficient of determination to the one from Problem 2, $0.9348 < 0.9996$ Therefore the coefficients found in Problem 2 are a better representation of the actual system. Since the actual system includes a

negative coefficient for a_3, not allowing negative coefficients in Problem 3 impacted the ability of the optimization to get close to the actual coefficient values.

Exercise 4.2: Problem 1

For this problem, we were given the following transfer function of the controlled plant, and were required to use the Simplex code provided to find the optimal coefficients of a PID controller. The PID controller was configured to use the system error as its input for the controller.



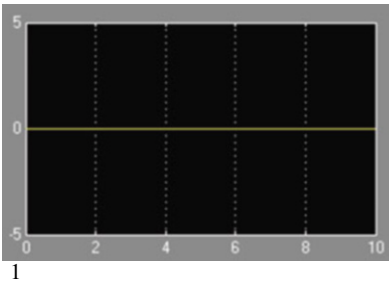
As could be seen, the optimization criterion, known as "integral-time-error-squared" was chosen. The Simplex procedure began with zero initial coefficient values, and progressively changed these values, minimizing the criterion. All 30 iteration of the Simplex procedure are shown below.

Iteration	k_P	k_I	k_D	Q
1	0.000	0.000	0.000	50.00000
2	0.094	0.024	0.024	38.18000
3	0.024	0.094	0.024	25.61000
4	0.024	0.024	0.094	41.24000
5	0.094	0.094	0.094	24.15000
6	0.141	0.141	0.141	17.33000
7	0.149	0.149	0.031	16.29000
8	0.212	0.212	0.000	10.76000

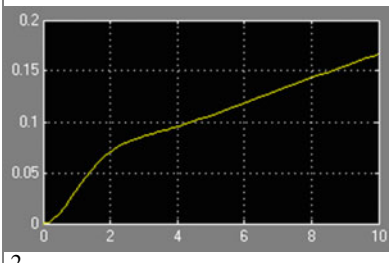
(continued)

Iteration	k_P	k_I	k_D	Q
9	0.157	0.275	0.086	7.86200
10	0.189	0.401	0.118	4.21000
11	0.338	0.409	0.149	3.87000
12	0.495	0.566	0.212	2.06500
13	0.456	0.644	0.079	1.62100
14	0.613	0.896	0.047	0.86220
15	0.652	1.029	0.251	0.72740
16	0.872	1.438	0.377	0.48830
17	1.131	1.532	0.306	0.39070
18	1.603	2.098	0.401	0.29570
19	1.563	2.388	0.338	0.34080
20	2.079	3.054	0.697	0.29080
21	2.813	4.133	1.021	0.26310
22	3.114	4.308	0.796	0.25040
23	4.235	5.743	1.006	0.21720
24	4.203	5.594	1.281	0.19470
25	5.523	7.197	1.752	0.15960
26	6.778	9.284	2.119	0.14560
27	9.365	12.877	2.978	0.11670
28	9.936	13.079	2.802	0.10360
29	13.498	17.552	3.693	0.08105
30	14.689	19.341	4.609	0.08550

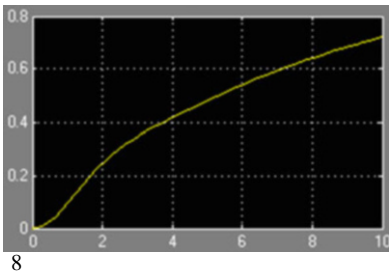
The following plots illustrate gradual improvement of the closed-loop step response of the system, iterations 1, 2, 8, 12, 14, 16 are shown below. Technically, the procedure could be terminated after the 16-th iteration when the design requirements were met.



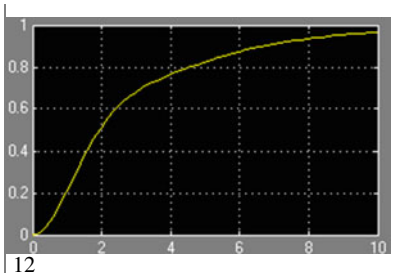
1



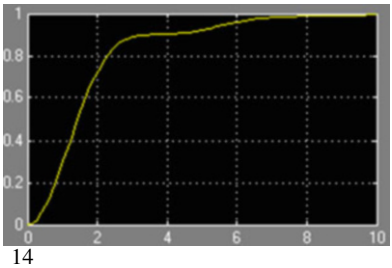
2



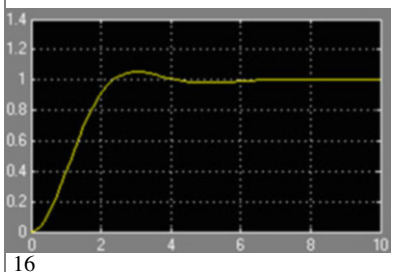
8



12



14



16

Exercise 4.2: Problem 2

The task is to estimate parameters a_1, a_2, a_3, a_4, a_5 of the following model

$$y^{MOD} = a_1 x_1 e^{a_2 x_2} + a_3{}^{(a_4 x_3 + a_5)}$$

based on the input–output data presented in the table below. It could be seen that while the Least Squares Method (LSM) is to be applied, due to the nonlinearity of the model, traditional LSM equation is unusable and the LSM criterion,

$$Q(a_1,\ a_2,\ a_3,\ a_4,\ a_5) = \sum_i \left[y(i) - y^{MOD}(i)\right]^2$$

could be minimized only by a genetic optimization (GO) procedure.

Assume that a "generation" size is 20. To start the procedure, begin with 20 randomly generated sets of 5 coefficients compiled into in a 20×5 matrix, with each row representing a particular set of coefficients (an individual). In each generation these 20 "individuals" will become the source of off-spring and mutants. Each combination of two individuals within a generation results in 5 off-spring. Each individual in the generation originates 5 mutants.

It could be seen that this process results in an immense number of new individuals, however, each newly created individual is subjected to the "fitness test" and only the 20 most fit individuals are included in the next generation. Each "next generation" is subjected to the same treatment until some termination conditions are met.

Below are several successive generation created by GO and accompanying values of the coefficient of determination of the resultant model.

Initial generation

a_1	a_2	a_3	a_4	a_5
2.8985	1.1513	1.3472	0.7978	−1.7259
1.6345	0.3528	−3.1390	−0.4051	−1.6175
4.0328	−0.2817	−0.8104	−0.2262	2.0505
0.5651	−0.3717	−1.4612	2.1874	0.2249
0.0178	−2.6064	0.7837	2.0933	0.9262
−0.4143	1.3217	0.6731	−1.0293	0.6581
0.3766	2.1826	2.5247	−1.7790	−0.3017
1.9690	−1.3128	−0.5600	2.6983	−2.0503
1.5223	−1.7655	−1.0972	−6.0455	1.9243
−1.9364	2.2881	−2.5293	1.9482	−1.2985
−0.1199	3.7093	0.4981	0.0415	−0.1648
−2.6940	2.6746	−2.0002	1.0056	−0.6448
−1.6309	3.0015	−1.4742	2.2930	2.2985
−1.6146	3.2141	3.1189	1.3045	−4.4747
3.7037	−0.5679	0.2711	−1.9776	0.4899
1.3725	−0.5816	0.3215	−2.4576	1.0954
0.9418	2.1572	−0.3685	3.4800	−3.1552
−2.9340	−4.9244	−0.0846	0.5385	−1.4570
3.5418	−0.5850	−0.3802	−4.0273	0.3840
0.5160	−4.4928	−0.1777	3.0757	−1.5804

Generation 1

a_1	a_2	a_3	a_4	a_5	Determination coefficient
2.8985	1.1513	1.3472	0.7978	−1.7259	0.8825
2.2881	1.2683	1.3268	0.3294	−0.2899	0.8756
3.1209	1.2110	1.8546	−1.8284	0.2756	0.8755
2.5022	1.2605	0.6157	−0.9998	1.3549	0.8656
1.8744	1.2679	−0.9757	0.6641	2.0549	0.8601
1.1644	1.4379	−1.7683	−1.0123	−0.8917	0.8532
1.6140	1.2957	−0.5244	−0.5290	1.4760	0.8408
0.7947	1.6388	0.9696	−2.4417	0.6111	0.8397
2.0885	1.1947	0.8106	−0.2046	−0.5338	0.8384
0.8033	1.5508	2.2729	−0.0392	−0.5913	0.8352

(continued)

a_1	a_2	a_3	a_4	a_5	Determination coefficient
2.5520	1.1144	0.4245	−0.3246	0.5987	0.8319
1.1761	1.5435	1.4602	−2.2510	−0.0922	0.8282
3.4343	1.0007	−0.5871	0.2609	1.1236	0.8243
1.6577	1.2627	−0.9837	1.3665	−0.4513	0.8215
1.3407	1.4972	−2.5164	0.8116	−2.9722	0.8175
2.9522	1.0467	0.1449	0.5600	1.8093	0.8157
0.6689	1.5793	−0.7417	2.8365	0.1993	0.8074
2.7650	1.0530	1.1006	0.2516	−1.6950	0.7990
1.8029	1.2170	−1.9619	1.3318	−1.5114	0.7959
2.8907	1.2643	0.7140	0.1415	−2.4496	0.7925

Generation 3

a_1	a_2	a_3	a_4	a_5	Determination coefficient
3.2330	1.1311	0.1180	0.1429	−1.5847	0.9203
3.4089	1.1194	0.1505	0.2626	−1.3844	0.9178
3.0811	1.1600	0.1540	0.3674	−1.2171	0.9152
3.2224	1.1584	0.1650	0.2263	−1.4428	0.9142
3.2893	1.1546	0.1817	0.8279	−0.1388	0.9137
2.5298	1.2368	0.0907	0.5090	−0.1884	0.9114
3.0611	1.1398	0.0935	0.0080	−1.5563	0.9112
2.4017	1.2431	0.2474	0.7796	−0.6685	0.9102
3.3596	1.1058	0.1555	0.1667	−1.3748	0.9095
2.8410	1.1452	0.1536	0.0956	−1.7017	0.9089
2.4652	1.2277	0.0785	0.3539	−0.4969	0.9087
2.2753	1.2568	0.2190	0.5816	−0.8525	0.9073
2.3878	1.2545	0.1447	0.5450	−0.3459	0.9068
2.6861	1.1952	0.2399	0.2100	−1.6736	0.9057
3.1428	1.1634	0.2227	0.1615	−1.6702	0.9056
2.6985	1.1850	0.2020	0.1200	−1.6083	0.9054
2.4388	1.2495	0.0313	0.3191	−0.1237	0.9054
3.2899	1.1293	0.2615	0.2999	−1.4908	0.9051
2.3510	1.2336	0.2180	0.3131	−1.4478	0.9048
2.8936	1.1421	0.1960	0.1882	−1.6563	0.9044

Generation 5

a_1	a_2	a_3	a_4	a_5	Determination coefficient
3.2187	1.1474	0.1081	0.1070	−1.5610	0.9243
3.2307	1.1444	0.1065	0.0930	−1.5475	0.9242
3.1887	1.1535	0.1066	0.0969	−1.5458	0.9241
3.1930	1.1516	0.1153	0.0942	−1.6730	0.9241
3.1678	1.1503	0.1164	0.1062	−1.5840	0.9239
3.3344	1.1401	0.1163	0.1289	−1.5773	0.9239
3.2312	1.1458	0.1056	0.0839	−1.5436	0.9239

(continued)

a_1	a_2	a_3	a_4	a_5	Determination coefficient
3.1755	1.1476	0.1139	0.0927	−1.5828	0.9238
3.1045	1.1533	0.1032	0.0814	−1.5745	0.9238
3.1020	1.1555	0.1101	0.0826	−1.5823	0.9238
3.1053	1.1523	0.1109	0.0850	−1.5895	0.9237
3.2088	1.1392	0.1033	0.0831	−1.5563	0.9237
3.3173	1.1454	0.1160	0.1322	−1.5786	0.9236
3.1123	1.1533	0.1164	0.1001	−1.5813	0.9236
3.2996	1.1295	0.1009	0.0858	−1.5604	0.9236
3.3068	1.1395	0.1328	0.1453	−1.6407	0.9236
3.3615	1.1333	0.1098	0.1346	−1.5013	0.9235
3.2392	1.1462	0.1213	0.1365	−1.5729	0.9235
3.2177	1.1370	0.1105	0.0952	−1.5843	0.9235
3.0982	1.1542	0.1172	0.1026	−1.5828	0.9235

Generation 7

a_1	a_2	a_3	a_4	a_5	Determination coefficient
3.2964	1.1409	0.1069	0.1003	−1.5657	0.9244
3.2921	1.1406	0.1063	0.0987	−1.5740	0.9244
3.2906	1.1400	0.1057	0.1014	−1.5538	0.9244
3.2617	1.1426	0.1080	0.1004	−1.5746	0.9244
3.2937	1.1406	0.1141	0.1063	−1.6061	0.9244
3.2497	1.1445	0.1063	0.1002	−1.5612	0.9244
3.2837	1.1424	0.1048	0.0966	−1.5579	0.9244
3.2246	1.1478	0.1122	0.1016	−1.6072	0.9244
3.2764	1.1418	0.1104	0.1047	−1.5810	0.9244
3.2292	1.1473	0.1077	0.1007	−1.5728	0.9244
3.2272	1.1467	0.1149	0.1005	−1.6283	0.9244
3.2825	1.1422	0.1085	0.0975	−1.5793	0.9244
3.2831	1.1400	0.1081	0.1029	−1.5699	0.9244
3.2317	1.1464	0.1090	0.1021	−1.5838	0.9244
3.2291	1.1462	0.1074	0.0996	−1.5691	0.9244
3.2202	1.1473	0.1083	0.1002	−1.5767	0.9244
3.2817	1.1425	0.1073	0.1012	−1.5615	0.9244
3.2629	1.1422	0.1116	0.1037	−1.5985	0.9244
3.2778	1.1426	0.1187	0.1053	−1.6354	0.9244
3.2414	1.1462	0.1072	0.1030	−1.5708	0.9244

Generation 9

a_1	a_2	a_3	a_4	a_5	Determination coefficient
3.2931	1.1403	0.1067	0.0996	−1.5671	0.9244
3.2946	1.1402	0.1062	0.0997	−1.5652	0.9244
3.2915	1.1403	0.1077	0.1000	−1.5749	0.9244
3.2941	1.1402	0.1060	0.0996	−1.5644	0.9244
3.2921	1.1404	0.1069	0.0999	−1.5686	0.9244
3.2934	1.1404	0.1059	0.0986	−1.5632	0.9244
3.2905	1.1404	0.1056	0.0991	−1.5608	0.9244
3.2936	1.1404	0.1113	0.1011	−1.5986	0.9244
3.2937	1.1404	0.1064	0.0998	−1.5654	0.9244
3.2936	1.1402	0.1061	0.1000	−1.5628	0.9244
3.2917	1.1402	0.1062	0.0996	−1.5629	0.9244
3.2895	1.1405	0.1055	0.0987	−1.5605	0.9244
3.2943	1.1403	0.1063	0.1001	−1.5638	0.9244
3.2900	1.1404	0.1057	0.0986	−1.5612	0.9244
3.2913	1.1403	0.1060	0.0996	−1.5622	0.9244
3.2918	1.1404	0.1064	0.0999	−1.5648	0.9244
3.2902	1.1402	0.1056	0.0985	−1.5612	0.9244
3.2901	1.1406	0.1072	0.0997	−1.5719	0.9244
3.2918	1.1406	0.1057	0.0988	−1.5631	0.9244
3.2933	1.1405	0.1065	0.0999	−1.5674	0.9244

After 9 iterations of the GO procedure stopped, and the resultant value of the coefficient of determination was 0.9244 for coefficients [3.2931, 1.1403, 0.1067, 0.0996, −1.5671], and the model expression is: $y^{\mathrm{MOD}} = 3.2931x_1e^{1.1403x_2} + 0.1067^{0.0996x_3 - 1.5671}$

Exercise 4.3: Problem 1

Conditional optimization of Process IV:

If $x_4 = [10, 40]$
Choose $u_4 = 2$
Cost = 4

If $x_4 = (40, 70]$
Choose $u_4 = 3$
Cost = 4

If $x_4 = (70, 100]$
Choose $u_4 = 3$
Cost = 3

Conditional optimization of Process III and Process IV:

If $x_3 = [10, 40]$
If $u_3 = 1$, cost $= 16 + \text{cost}(x_4 = 13) = 16 + 4 = 20$
If $u_3 = 2$, cost $= 18 + \text{cost}(x_4 = 45) = 18 + 4 = 22$
If $u_3 = 3$, cost $= 9 + \text{cost}(x_4 = 92) = 9 + 3 = \mathbf{12}$ (optimal)
Choose $u_3 = 3$
Cost = 12

If $x_3 = (40, 70]$
If $u_3 = 1$, cost $= 13 + \text{cost}(x_4 = 48) = 13 + 4 = 17$
If $u_3 = 2$, cost $= 17 + \text{cost}(x_4 = 18) = 17 + 4 = 21$
If $u_3 = 3$, cost $= 8 + \text{cost}(x_4 = 68) = 8 + 4 = \mathbf{12}$ (optimal)
Choose $u_3 = 3$
Cost = 12

If $x_3 = (70, 100]$
If $u_3 = 1$, cost $= 10 + \text{cost}(x_4 = 81) = 10 + 3 = 13$
If $u_3 = 2$, cost $= 14 + \text{cost}(x_4 = 66) = 14 + 4 = 18$
If $u_3 = 3$, cost $= 6 + \text{cost}(x_4 = 21) = 6 + 4 = \mathbf{10}$ (optimal)
Choose $u_3 = 3$
Cost = 10

Conditional optimization of Process II, Process III and Process IV:

If $x_2 = [10, 40]$
If $u_2 = 1$, cost $= 13 + \text{cost}(x_3 = 65) = 13 + 12 = \mathbf{25}$ (optimal)
If $u_2 = 2$, cost $= 21 + \text{cost}(x_3 = 44) = 21 + 12 = 33$
If $u_2 = 3$, cost $= 33 + \text{cost}(x_3 = 74) = 33 + 10 = 43$
Choose $u_2 = 1$
Cost = 25

If $x_2 = (40, 70]$
If $u_2 = 1$, cost $= 15 + \text{cost}(x_3 = 66) = 15 + 12 = \mathbf{27}$ (optimal)
If $u_2 = 2$, cost $= 22 + \text{cost}(x_3 = 50) = 22 + 12 = 33$
If $u_2 = 3$, cost $= 37 + \text{cost}(x_3 = 81) = 37 + 10 = 47$
Choose $u_2 = 1$
Cost = 27

If $x_2 = (70, 100]$
If $u_2 = 1$, cost $= 18 + \text{cost}(x_3 = 78) = 18 + 10 = \mathbf{28}$ (optimal)
If $u_2 = 2$, cost $= 28 + \text{cost}(x_3 = 62) = 28 + 12 = 40$
If $u_2 = 3$, cost $= 40 + \text{cost}(x_3 = 96) = 40 + 10 = 50$
Choose $u_2 = 1$
Cost $= 28$

Conditional optimization of Process I, Process II, Process III, and Process IV:

If $x_1 = [10, 40]$
If $u_1 = 1$, cost $= 25 + \text{cost}(x_2 = 25) = 25 + 25 = \mathbf{50}$ (optimal)
If $u_1 = 2$, cost $= 28 + \text{cost}(x_2 = 45) = 28 + 27 = 55$
If $u_1 = 3$, cost $= 25 + \text{cost}(x_2 = 55) = 25 + 27 = 52$
Choose $u_1 = 1$
PATH: $u_1 = 1 \rightarrow u_2 = 1 \rightarrow u_3 = 3 \rightarrow u_4 = 3$
Cost $= 50$

If $x_1 = (40, 70]$
If $u_1 = 1$, cost $= 27 + \text{cost}(x_2 = 37) = 27 + 25 = \mathbf{52}$ (optimal)
If $u_1 = 2$, cost $= 33 + \text{cost}(x_2 = 48) = 33 + 27 = 60$
If $u_1 = 3$, cost $= 27 + \text{cost}(x_2 = 63) = 27 + 28 = 55$
Choose $u_1 = 1$
PATH: $u_1 = 1 \rightarrow u_2 = 1 \rightarrow u_3 = 3 \rightarrow u_4 = 3$
Cost $= 52$

If $x_1 = (70, 100]$
If $u_1 = 1$, cost $= 22 + \text{cost}(x_2 = 45) = 22 + 27 = \mathbf{49}$ (optimal)
If $u_1 = 2$, cost $= 24 + \text{cost}(x_2 = 58) = 24 + 27 = 51$
If $u_1 = 3$, cost $= 25 + \text{cost}(x_2 = 79) = 25 + 28 = 53$
Choose $u_1 = 1$
PATH: $u_1 = 1 \rightarrow u_2 = 1 \rightarrow u_3 = 3 \rightarrow u_4 = 3$
Cost $= 49$

Optimal Plan

Since $x = 37$, the optimal path is $u_1 = 1 \rightarrow u_2 = 1 \rightarrow u_3 = 3 \rightarrow u_4 = 3$ and the cost $= 50$.

Exercise 4.3: Problem 2

Conditional optimization of particular stages of the process starting from the last stage:

Stage 5 $\rightarrow 6$
If at A5 $\rightarrow$ A6 $-$ Cost $= 8$ (optimal)
If at B5 $\rightarrow$ A6 $-$ Cost $= 7$ (optimal)

Stage 4 $\rightarrow 5$
If at A4 $\rightarrow$ ~~A5 – Cost~~ $= ~~15~~ + ~~8~~ = ~~23~~$
$\rightarrow$ B5 $-$ Cost $= 10 + 7 = 17$ (optimal)
If at B4 $\rightarrow$ ~~A5 – Cost~~ $= ~~19~~ + ~~8~~ = ~~27~~$
$\rightarrow$ B5 $-$ Cost $= 14 + 7 = 21$ (optimal)
If at C4 $\rightarrow$ ~~A5 – Cost~~ $= ~~16~~ + ~~8~~ = ~~24~~$
$\rightarrow$ B5 $-$ Cost $= 13 + 7 = 20$ (optimal)

Stage 3 $\rightarrow 4$
If at A3 $\rightarrow$ ~~A4 – Cost~~ $= ~~6~~ + ~~17~~ = ~~23~~$
$\rightarrow$ ~~B4 – Cost~~ $= 4 + ~~21~~ = ~~25~~$
$\rightarrow$ C4 $-$ Cost $= 2 + 20 = 22$ (optimal)
If at B3 $\rightarrow$ A4 $-$ Cost $= 7 + 17 = 24$ (optimal)
$\rightarrow$ ~~B4 – Cost~~ $= ~~4~~ + ~~21~~ = ~~25~~$
$\rightarrow$ ~~C4 – Cost~~ $= ~~12~~ + ~~20~~ = ~~32~~$
If at C3 $\rightarrow$ A4 $-$ Cost $= 5 + 17 = 22$ (optimal)
$\rightarrow$ B4 $-$ Cost $= 3 + 21 = 24$
$\rightarrow$ C4 $-$ Cost $= 7 + 20 = 27$

Stage 2 $\rightarrow 3$
If at A2 $\rightarrow$ ~~A3 – Cost~~ $= ~~2~~ + ~~22~~ = ~~24~~$
$\rightarrow$ ~~B3 – Cost~~ $= ~~3~~ + ~~24~~ = ~~27~~$
$\rightarrow$ C3 $-$ Cost $= 1 + 22 = 23$ (optimal)
If at B2 $\rightarrow$ ~~A3 – Cost~~ $= ~~9~~ + ~~22~~ = ~~31~~$
$\rightarrow$ ~~B3 – Cost~~ $= ~~2~~ + ~~24~~ = ~~26~~$
$\rightarrow$ C3 $-$ Cost $= 2 + 22 = 24$ (optimal)
If at C2 $\rightarrow$ ~~A3 – Cost~~ $= ~~6~~ + ~~22~~ = ~~28~~$
$\rightarrow$ ~~B3 – Cost~~ $= 4 + ~~24~~ = ~~28~~$
$\rightarrow$ C3 $-$ Cost $= 3 + 22 = 25$ (optimal)
If at D2 $\rightarrow$ A3 $-$ Cost $= 3 + 22 = 25$ (optimal)
$\rightarrow$ ~~B3 – Cost~~ $= ~~3~~ + ~~24~~ = ~~27~~$
$\rightarrow$ ~~C3 – Cost~~ $= ~~4~~ + ~~22~~ = ~~26~~$

Stage 1 $\rightarrow 2$
If at A1 $\rightarrow$ ~~A2 – Cost~~ $= ~~7~~ + ~~23~~ = ~~30~~$
$\rightarrow$ B2 – Cost $= 4 + 24 = 28$ (optimal)
$\rightarrow$ ~~C2 – Cost~~ $= ~~5~~ + ~~25~~ = ~~30~~$
$\rightarrow$ ~~D2 – Cost~~ $= ~~6~~ + ~~25~~ = ~~31~~$

Optimal Path

A1 $\rightarrow$ B2 $\rightarrow$ C3 $\rightarrow$ A4 $\rightarrow$ B5 $\rightarrow$ A6, Cost: 28

Bibliography

http://www.onlinecalculatorfree.org/linear-programming-solver.html
MATLAB: https://www.mathworks.com/products/matlab-home/
Simulink: http://www.mathworks.com/products/simulink/
VISSIM: http://www.vissim.com/