G. GRAVES

D. PINGRY

A. WHINSTON

Water quality control : nonlinear programming algorithm

Revue française d'automatique, informatique, recherche opérationnelle. Recherche opérationnelle, tome 6, nº V2 (1972), p. 49-78

<http://www.numdam.org/item?id=RO_1972__6_2_49_0>

R.A.I.R.O.
(6e année, octobre 1972, V-2, p. 49-78)

WATER QUALITY CONTROL : NONLINEAR PROGRAMMING ALGORITHM

by G. GRAVES (1), D. PINGRY (2), A. WHINSTON (3)

Résumé. — *This paper presents some of the more practical aspects of nonlinear programming along with an application of this technique to a river basin water quality control problem.*

Section II constructs a nonlinear river basin model which is of the form :

Minimize : Total cost of abatement structures

Subject to : Water quality goals satisfied.

The water quality measure used is dissolved oxygen. The treatment alternatives allowed are regional and on-site treatment plants, by pass piping and flow augmentation. Results of an application of this model to the West Fork White River in Indiana are presented.

Section III is a discussion of the nonlinear algorithm which is used to solve the nonlinear problem formulated in Section II. The algorithm described is stepwise in nature. Starting with some point y^i in the domain of the function a direction Δy^i is determined by solving a parametric linear programming problem. A scalar k is then calculated to obtain a new point $y^{i+1}, = y^i + k\Delta y^i$. Discussion is focused on the determination of k and of $\bar{k}$, the parameter of the linear programming problem. The problems of storage space, variable priority classes and calculation of partial derivitives are also considered.

I. INTRODUCTION

The purpose of this paper is to present some of the more practical aspects of nonlinear programming in connection with an application of this technique to water pollution control.

N. B. — The development of the river simulation model in this paper owes much to the DOCAL computer program of the Environmental Protection Agency. This program was originally developed by J. L. Worley of E. P. A. and was applied on the West Fork White River by the Evansville Office of E. P. A. We are especially grateful to Max Noecker and Stanley Smith of the Evansville office who kindly made their data and experience available for our use. Discussions with out colleague J. Hamelink of the Forestry department were helpful. The authors are responsible for all possible errors. The research has been sponsored by the Office of Water Resources Research under contract 14-31-0001-3080.

(1) School of Business University of California Los Angeles, California.
(2) Department of Economics Virginia Polytechnic Institute and State University Blacksburg, Virginia.
(3) Krannert Graduate School of Industrial Administration Purdue University Lafayette, Indiana.

In Section II a description of the pollution problem is given and a formulation as a nonlinear programming problem is presented. Section III presents the theory of the nonlinear programming algorithm with details on its computer implementation.

II. WATER QUALITY BASIN MODEL

This section presents a short discussion of water quality relationships followed by the formulation of a river basin quality model. This model is then solved using the algorithm described in Section III. Problems encountered in this application are examined.

The most prevalent measure of water quality in the literature of water pollution control, is the level of dissolved oxygen concentration. We note that the dissolved oxygen level is often described relative to saturation level of oxygen in a river and is called the dissolved oxygen deficit or DOD. The level of dissolved oxygen concentration, or DO, is dependent on all the sources and sinks of oxygen in the river basin. Typical oxygen sources, as listed in O'Connor [14], are atmospheric reaeration and photosynthetic production. The typical sinks are respiration of bacteria and algae, benthal deposits, and chemical oxidation. When effluent, such as common sewage, is dumped into a river it is decomposed by bacteria. These bacteria require oxygen. The total oxygen required to reduce the organic material of the effluent to stable compounds is called ultimate biochemical oxygen demand or BOD. A model used to describe the relationship between DOD, atmospheric reaeration and bacterial respiration was formulated by Streeter-Phelps in 1925, [16]. Assume

$$\frac{\mathrm{d}b}{\mathrm{d}t} = -K_1 b \tag{1}$$

and

$$\frac{\mathrm{d}d}{\mathrm{d}t} = K_1 b - K_2 d \tag{2}$$

where,

b = biochemical oxygen demand (mg/l)
d = dissolved oxygen deficit (mg/l)
t = time (days)
K_1 = deoxygenation rate (days^{-1})
K_2 = reaeration rate (days^{-1}).

Equations (1) and (2) integrate to yield

(3) $b = b^0 C_1$

and

(4) $d = K b^0 [C_1 - C_2] + d^0 C_2$

where,

b^0 = biochemical oxygen demand when $t = 0$

d^0 = dissolved oxygen deficit when $t = 0$

and

$C_1 = \exp(-K_1 t)$

$C_2 = \exp(-K_2 t)$

$K = K_1/(K_2 - K_1)$

In the context of a river basin, if the velocity of river flow is assumed to be constant over a range, then time can be interpreted as

$$t = \left(\frac{1}{v}\right)a \tag{5}$$

where, v is the velocity of flow and a is the distance from the initial point of concern.

For the purpose of this paper is has also been assumed that the reaeration rate and velocity of flow rate are functions of the volumetric flow. It is also assumed that the reaeration and deoxygenation rates are a function of the temperature. The following equations were used :

$$K_1 = (K_1^{20})\theta_1{}^{T-20} \tag{6}$$

$$K_2 = \alpha F^{\beta}\theta_2{}^{T-20} \tag{7}$$

$$v = \gamma F^{\sigma} \tag{8}$$

These equations are consistant with the approach of others in this area. (See [12] and [18].)

Using the water quality relationships briefly described above, a programming model of the following form can be constructed :

Minimize : Total cost of abatement structures.

Subject to : Water quality goals satisfied.

The constraints of this model are constructed by dividing the river into sections and constraining the water quality, interpreted as the dissolved oxygen deficit level, to be met at the end of each section. A new section begins where one of the following occurs :

1. Effluent flow enters the river.
2. Incremental flow enters the river. (Ground water, tributary flow, etc.)
3. The flow in the main channel is augmented or diverted.

4. The parameters describing the particular river change.

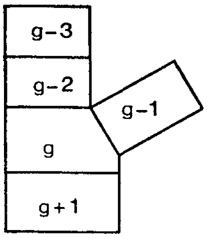


Figure 1
Typical River Sections

The river sections are numbered sequentially starting with the headwaters section and including possible tributaries of a maximum length of one section. See Figure 1.

Explicitly the model is as follows (1) :

Minimize :

(9) $$TC = C^L + C^R + C^R$$

where,

(10) $$C^L = \sum_g \sum_i C^L_{gi} + \sum_m \sum_i C^L_{mi} + \sum_g \sum_m C^L_{gm}$$

(11) $$C^L_{gi} = 1.865\, a_{gi}(f_{gi})^{.598}$$

(12) $$C^L_{mi} = 1.865\, a_{mi}(p_{mi})^{.598}$$

(13) $$C^L_{gm} = 1.865\, a_{gm}(t_{gm})^{.598}$$

and

(14) $$C^p = \sum_m 49.22\, (\sum_i p_{mi})^{3/4}[8.0(r_m - .5)^3 + 1]$$

(15) (2) $$C^R = \sum_g C_g F_{g3}$$

Subject to :

(16) $$d_g = K_g b_g [C_{1g} - C_{2g}] + d^0_g C_{2g} \leqslant d^*_g \qquad g = 1, N$$

where,

(17) $$K_g = K_{1g}/(\alpha_g F_g{}^{\beta g} - K_{1g})$$

(18) $$C_{1g} = \exp(- K_{1g} a_g / \gamma_g F_g^{\sigma g})$$

(19) $$C_{2g} = \exp(- \alpha g F_g^{\beta g - \sigma g} a_g / \gamma_g)$$

and

If section g is the headwaters section or a tributary

(20) $$F_g = \sum_{j=1}^{3} F_{gj}$$

(21) $$b^0_g = \sum_{j=1}^{3} b_{gj} F_{gj} / F_g$$

(22) $$d^0_g = \sum_{j=1}^{3} b_{gj} F_{gj} / F_g$$

(1) See Appendix A for definitions of notations.
(2) The cost functions were obtained from [4] and [12].

if section g is neither a tributary nor directly preceded by a tributary

$$F_g = \sum_{j=1}^{3} F_{gi} + F_{g-1} \tag{23}$$

$$b_g^0 = \left[\sum_{j=1}^{3} b_{gj}F_{gj} + b_{g-1}\right] / F_g \tag{24}$$

$$d_g^0 = \left[\sum_{j=1}^{3} d_{gj}F_{gj} + d_{g-1}F_{g-1}\right] / F_g \tag{25}$$

if section g is directly preceded by a tributary

$$F_g = \sum_{j=1}^{3} F_{gj} + F_{g-1} + F_{g-2} \tag{26}$$

$$b_g^0 = \left[\sum_{j=1}^{3} b_{gj}F_{gj} + b_{g-1}F_{g-1} + b_{g-2}F_{g-2}\right] / F_g \tag{27}$$

$$d_g^0 = \left[\sum_{j=1}^{3} d_{gj}F_{gj} + d_{g-1}F_{g-1} + d_{g-2}F_{g-2}\right] / F_g . \tag{28}$$

Furthermore,

$$F_{g2} = \sum_i f_{gi} + \sum_m t_{gm} \tag{29}$$

$$F_{g2}b_{g2} = \sum_i \bar{b}_i f_{gi} + \sum_m (1 - r_m)\left[\sum_i \bar{b}_i p_{mi} / \sum_i p_{mi}\right] t_{gm} \tag{30}$$

$$F_{g2}d_{g2} = \left[\sum_i \bar{d}_i p_{mi} / \sum_i p_{mi}\right] t_{gm} \tag{31}$$

We note that these quality constraints are sequentially dependent, the quality in each section being a function of the quality in the last section. However, the possibility of tributary, augmentation, incremental and effluent flows entering at downstream points complicates the relationship between the constraints.

The variables of the programming model above are :

f_{gi}	Flow from polluter i to section g.
p_{mi}	Flow from polluter i to treatment plant m.
t_{gm}	Flow from treatment plant into section g.
r_m	Percent removal of BOD at treatment plant m.
F_{g3}	Flow augmentation flow in section g.

These variables allow for the possibility of three possible treatment techniques :

1. By-pass piping.
2. Regional and on-site treatment plants.
3. Flow augmentation.

The particular pattern of piping flows determines the regional and on-site plants operating. In addition to the quality constraints given above, flow conservation constraints are needed around the polluters and treatment plants. These constraints are as follows :

$$(32) \qquad \sum_g f_{gi} + \sum_m p_{mi} - \bar{f}_i = 0 \qquad (i = 1, p)$$

$$(33) \qquad \sum_m p_{mi} - \sum_m t_{gm} = 0 \qquad (i = 1, p, g = 1, N)$$

The major problem of adapting this model for solution by the nonlinear algorithm presented in Section III is the calculation of the partial derivative of the constraints and objective function. These calculations are necessary to set up a local L. P. problem to determine a direction of search. If all the constraints were of a different functional form the computer coding necessary for partial derivative evaluation would be imense for any large-scale problem. However, as in most large-scale problems, the constraints of the water-quality model can be classified into a relatively few functionally homogeneous groupings.

The function grouping for the water-quality model are represented in the tableau below.

	f_{gi}	p_{mi}	t_{gm}	r_m	F_{g3}
Quality Constraints	Λ_{11}	Λ_{12}	Λ_{13}	Λ_{14}	Λ_{15}
Flow Conservation (polluter).......	Λ_{21}	Λ_{22}	Λ_{23}	Λ_{24}	Λ_{25}
Flow Conservation (treatment plants)	Λ_{31}	Λ_{32}	Λ_{33}	Λ_{34}	Λ_{35}
Objective Function	Λ_{41}	Λ_{42}	Λ_{43}	Λ_{44}	Λ_{45}

Each Λ_{ij} represents a matrix of partial derivatives between the constraints and the appropriate variables. For example, if the quality constraints were labeled $Q_1, \ldots, Q_N$ then the sub-matrix Λ_{11} would be

$$\Lambda_{11} = \begin{bmatrix} \frac{\partial Q_1}{\partial f_{11}} & \cdots & \frac{\partial Q_1}{\partial f_{Np}} \\ \cdot & & \\ \cdot & & \\ \cdot & & \\ \frac{\partial Q_N}{\partial f_{11}} & \cdots & \frac{\partial Q_N}{\partial f_{Np}} \end{bmatrix}$$

Note that the Tableau Λ where

$$\Lambda = \begin{bmatrix} \Lambda_{11} \ldots \Lambda_{15} \\ \cdot \\ \cdot \\ \Lambda_{41} \ldots \Lambda_{45} \end{bmatrix}$$

is comparable to Tableau (1) in Section III. Each block in Λ can be treated as a unit for the computer coding of the problem. Some of the blocks require little calculation. For example, the blocks such as Λ_{21}, Λ_{22}, etc. are simply linear functions with coeficients of 1 requiring only that a 1 be placed in the appropriate place. On the other hand other blocks such as Λ_{14} and Λ_{15} require considerable sorting and calculation. As an example of the complexity of these calculations, Appendix B gives the necessary equations for the calculation of Λ_{14} and Λ_{15}. The elements of Λ are generated a column at a time and only the collumns associated with the basis variables of the L. P. problem are stored. The other columns are generated and updated as needed as explained in Section III.

We notice that even with the simplification of the problem from the treatment of the homogenous function blocks, that there are still 20 different blocks to deal with. This means 20 different partial derivative forms must be pre-calculated and programmed. In the water-quality case the « not quite sequential nature » of the constraints further complicates the problem.

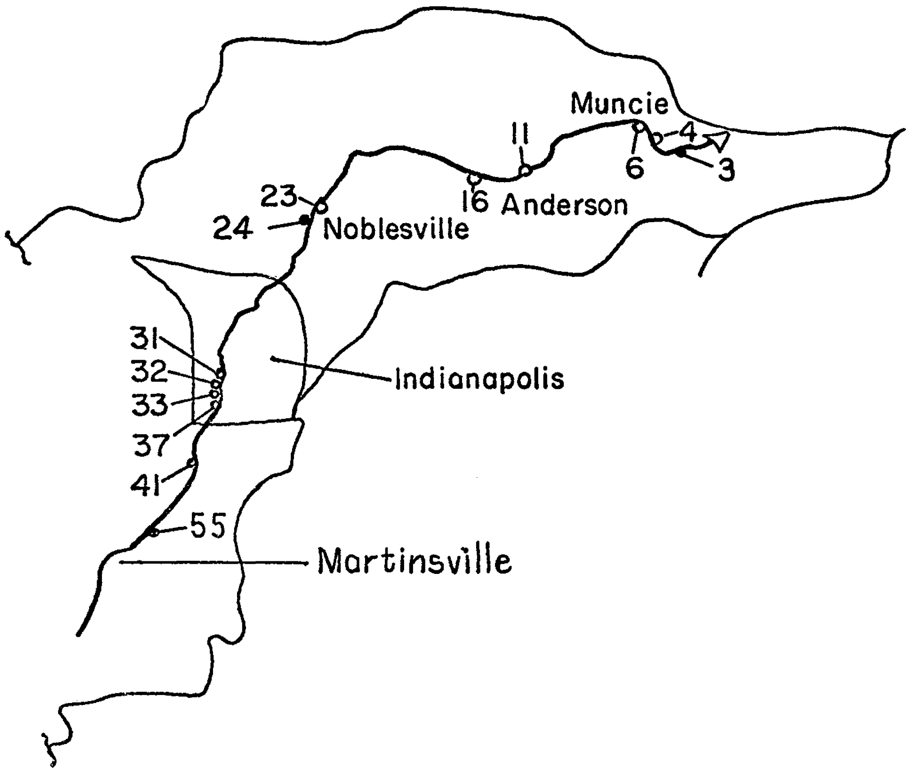
Because of the vast number of variables in the water quality problem the following solution technique was adopted to search the feasible set. A number of piping patterns which seemed reasonable from our knowledge of the problem were read in as the initial solution and the percent removal variables of the appropriate treatment plants and flow augmentation were given a high priority level. This technique saved considerable computer time and still allowed the feasible set to be adequately searched.

The programming model as proposed has been applied to the West Fork White River in Indiana. The West Fork White River has its source near the

Indiana-Ohio border and flows southwesterly for 371 miles through the state of Indiana. At this point it joins the East Fork White River and flows to the Ohio.

Map I

West Fork White River, Location of Polluters



The major city on the West Fork is Indianapolis, a city of over 600,000 which is 234 miles from the mouth. Two other cities, Anderson and Muncie, are upstream from Indianapolis. The concentration of population and industry around these three cities cause the major portion of the pollution problem in the West Fork White.

For the purpose of this paper we have chosen a length of the West Fork White which runs from the headwaters above Muncie to Spencer below

k	a_g	K^{20}_{1g}	α_g	β_g	γ_g	σ_g
1	5.600000	.115000	6.450000	-.249000	.044500	.550000
2**	.100000	.100000	6.450000	-.249000	.044500	.550000
3	6.200000	.103000	6.450000	-.249000	.044500	.550000
4	1.900000	.100000	6.450000	-.249000	.044500	.550000
5	3.600000	.100000	6.450000	-.249000	.044500	.550000
6	3.700000	.600000	6.450000	-.249000	.044500	.550000
7*	.100000	.115000	6.450000	-.249000	.044500	.550000
8	3.200000	.630000	6.450000	-.249000	.044500	.550000
9	2.800000	.630000	6.450000	-.249000	.044500	.550000
10	3.000000	.630000	6.450000	-.249000	.044500	.550000
11	4.300000	.623000	6.450000	-.249000	.044500	.550000
12	1.900000	.308000	.039800	.538000	.012500	.728000
13	1.000000	.308000	2.760000	-.117000	.065000	.471000
14**	.100000	.102000	6.450000	-.249000	.044500	.550000
15	2.400000	.304000	2.760000	-.117000	.065000	.471000
16	3.400000	.805000	2.760000	-.117000	.065000	.471000
17	4.000000	.805000	2.760000	-.117000	.065000	.471000
18	5.000000	.788000	2.760000	-.117000	.065000	.471000
19**	.100000	.104000	6.450000	-.249000	.044500	.550000
20	3.000000	.600000	2.760000	-.117000	.065000	.471000
21	2.500000	.600000	2.760000	-.117000	.065000	.471000
22	.500000	.100000	2.760000	-.117000	.065000	.471000
23	3.300000	.100000	.037200	.403000	.004500	.775000
24	5.100000	.100000	3.270000	-.140000	.064000	.448000
25	1.200000	.275000	3.290000	-.140000	.140000	.685000
26	.800000	.300000	3.290000	-.140000	.014000	.685000
27**	.100000	.103000	6.450000	-.249000	.044500	.550000
28	2.900000	.300000	3.290000	-.140000	.014000	.685000
29	5.800000	.100000	3.290000	-.140000	.014000	.685000
30	5.000000	.100000	3.290000	-.140000	.014000	.685000
31	4.800000	.100000	.141000	.183000	.005600	.715000
32	5.400000	.095000	.141000	.183000	.005600	.715000

**Tributaries

TABLE 1. — RIVER SECTION DATA

33	5.000000	.091000	.003340	.645000	.002300	.780000
34**	.100000	.096000	6.450000	-.249000	.044500	.550000
35	1.800000	.093000	.003340	.645000	.002300	.780000
36	.500000	.093000	.003340	.645000	.002300	.780000
37	.300000	.093000	.003340	.645000	.002300	.780000
38	1.800000	.092000	.003310	.619000	.000700	.935000
39	.700000	.092000	.003310	.619000	.000700	.935000
40	.700000	.201000	.003310	.619000	.000700	.935000
41	.200000	.201000	.003310	.620000	.000700	.935000
42**	.100000	.940000	6.450000	-.249000	.044500	.550000
43	.800000	.201000	.003310	.619000	.000700	.935000
44	2.400000	.175000	3.550000	-.197000	.005000	.765000
45	2.300000	.175000	3.550000	-.197000	.005000	.765000
46	1.600000	.213000	3.550000	-.197000	.005000	.765000
47	2.300000	.213000	3.550000	-.197000	.005000	.765000
48	3.300000	.225000	3.550000	-.197000	.005000	.765000
49	4.000000	.225000	3.550000	-.197000	.005000	.765000
50	4.000000	.308000	3.550000	-.197000	.005000	.765000
51	4.400000	.308000	3.550000	-.197000	.005000	.765000
52	2.600000	.308000	3.550000	-.197000	.005000	.765000
53**	.100000	.100000	3.550000	-.197000	.005000	.765000
54	1.200000	.310000	3.550000	-.197000	.005000	.765000
55	2.800000	.298000	.037300	.403000	.044500	.775000
56	4.400000	.284000	5.500000	-.233000	.066000	.438000
57	4.700000	.204000	5.500000	-.233000	.066000	.438000
58	4.900000	.238000	5.500000	-.233000	.066000	.438000
59	5.000000	.244000	5.500000	-.233000	.066000	.438000
60	5.000000	.244000	5.500000	-.233000	.066000	.438000
61	5.000000	.182000	5.500000	-.233000	.066000	.438000
62	5.000000	.182000	5.500000	-.233000	.066000	.438000

** Tributaries

TABLE 1. — *(continued)*

Indianapolis. The section described is 172.8 miles long and is divided into 62 sections based on information about polluters, incremental flow, and river parameters. The sections range in length from .1 miles to 6.2 miles. The section parameters necessary for the implementation of the model are given in Table 1. The incremental flows, both in and out are given in Table 2. There are thirteen major polluters considered on the main stream. These are listed in Table 3.

For the purpose of this application we assume that all polluters are already treating on site at a level of 80 percent removal (secondary treatment). The basin solution in this case will contain only advanced (tertiary) waste treatment necessary to achieve required standards. The quality standards assumed will be 5 mg/l for every section.

The least-cost solution for teritary treatment for the West Fork White River is to build a regional plant for pollutes 40 and 46 at section 46 treating at the .95 level. The rest of the polluters, except for 6 and 16, remain treating on site at 80 percent removal. The polluters 6 and 16 must treat on site at levels

k	Flow (cfs)	B.O.D. (mg/ℓ)	D.O. (mg/ℓ)
1	52.000000	2.920000	2.300000
2	6.100000	2.630000	6.700000
4	-21.200000	-0.000000	-0.000000
5	1.000000	214.000000	1.000000
7	30.300000	9.800000	3.700000
9	4.000000	23.200000	5.600000
10	6.400000	13.200000	2.400000
14	3.000000	5.500000	5.200000
15	72.000000	7.700000	5.400000
19	44.000000	7.180000	5.800000
20	-35.000000	-0.000000	-0.000000
22	23.000000	12.100000	5.600000
27	16.100000	6.900000	6.400000
28	13.000000	5.000000	6.500000
30	14.000000	5.000000	5.900000
32	-214.000000	-0.000000	-0.000000
33	28.000000	5.000000	7.900000
34	21.000000	10.620000	6.200000
35	5.800000	12.400000	6.600000
39	5.000000	9.300000	7.500000
42	1.000000	1.000000	6.000000
44	8.000000	13.900000	6.000000
46	8.000000	5.000000	3.900000
48	9.000000	5.000000	6.000000
50	10.000000	23.200000	1.000000
52	21.000000	16.700000	1.600000
53	33.700000	8.000000	8.300000
55	6.000000	13.700000	2.100000
57	14.000000	5.000000	6.000000
58	35.000000	5.000000	8.000000
59	38.000000	5.000000	6.000000
61	71.000000	5.000000	7.000000
62	25.000000	5.000000	7.000000

TABLE 2. — INCREMENTAL FLOW DATA

of respectively 93 and 86. They are not close enough together to use a regional plant since the piping costs would exceed any gains from economies of scale. The cost of the basin solution is $ 2,013,296. Of this amount, $ 1,587,054 is for the regional plant at 46, $ 208,577 is for the pipeline from polluter 40 to the regional plant and the rest is divided between polluters 6 and 16 in amounts of $ 158,068 and $ 59,597 respectively.

k	Flow (cfs)	B.O.D. (mg/ℓ)	D.O. (mg/ℓ)
4	.27	40.000000	4.000000
6	20.83	322.000000	3.000000
11	.30	298.000000	0.000000
16	24.40	200.000000	2.000000
26	.78	270.000000	2.000000
35	13.20	20.500000	6.500000
36	.93	10.000000	8.300000
37	13.00	20.000000	2.300000
40	195.00	450.000000	2.600000
41	10.00	20.000000	2.000000
42	61.0	30.440000	4.000000
46	185.00	450.000000	3.900000
56	.93	300.000000	0.000000

TABLE 3. — POLLUTER DATA

III. NONLINEAR ALGORITHM

The algorithm employed for solving the nonlinear programming problem of this paper is a general purpose algorithm which solves problems of the form :

$$\text{(34)} \qquad \begin{array}{ll} \text{Subject to} & g^i(y) \leqslant 0 \qquad i = 1, m-1 \\ \text{Minimize} & g^m(y) \end{array}$$

where y is a vector in E^n and $g^i(y)$, $i = 1, m$, are continuous functions with continuous partial derivatives defined on some open set. The vector y is assumed to be bounded from above and below.

$$\text{(35)} \qquad BL \leqslant y \leqslant UB.$$

The vectors BL and UB are also members of E^n.

The method to be discussed here was originally described by Graves in [7]. The method can also be used as a second order procedure as presented in [6]. This paper will be limited to the discussion of the algorithm as it was used to solve the large scale water pollution problem described in Section II.

The algorithm to be described is stepwise in nature. Starting with some point y^j in the domain of the function, a direction Δy^j and a scalar k are determined. A new point y^{j+1} is calculated.

$$y^{j+1} = y^j + k\,\Delta y^j \tag{36}$$

The vector Δy^j is also a vector in E^n.

The object of making the step to y^{j+1} is to either reduce the value of the objective function, if y^j is a feasible solution to the nonlinear programming problem, or obtain a « more feasible » solution to the nonlinear problem, if y^j is an infeasible solution. The phrase « more feasible » is interpreted in terms of the algorithm to mean « reduce the value of SUPG », where SUPG is defined to be :

$$\text{SUPG} = g^w(y^j)$$

where w is the index of the most infeasible constraint. If y^j is a feasible solution to (34) then $\text{SUPG} = 0$.

The completion of the determination of Δy^j and k, and the calculation of y^{j+1} will complete what will be known in the paper as the $j + 1^{th}$ nonlinear iteration. Each nonlinear iteration will consist of several local linear programming problems to determine Δy^j. The solution of each one of these linear programming problems will complete what will be known as a linear iteration.

For the purpose of our exposition the nonlinear iteration will be divided into two major parts. The first is the determination of Δy^j as a solution to a parametric linear programming problem. The second is the determination of k.

Since we have assumed that the functions $g^i(y^j)$, $i = 1, m$, are continuous, and have continuous partial derivatives on some open set D, the following approximation theorem can be used :

$$\text{Let } \nabla g^i(y^j)^T = \left[\frac{\partial g^i(y^j)}{\partial y_1} \;\ldots\ldots\ldots, \frac{\partial g^i(y^j)}{\partial y_n}\right]$$

be the gradient of $g^i(y^j)$ at the point y^j, where y^j is a member of a closed bounded subset E of the open set D.

Then,

$$g^i(y^j + \Delta y^j) = g^i(y^j) + \nabla g^i(y^j)^T \Delta y^j + R^i(\Delta y^j)$$

where

$$\operatorname*{limit}_{\Delta y^j \to 0} \frac{R^i(\Delta y^j)}{|\Delta y^j|} = 0$$

uniformly for $y^j \in E$.

The direction of improvement for nonlinear iteration $j + 1$ is obtained from the function above by estimating the term $R^i(\Delta y^j)$, and solving the associated local linear programming problem.

$$\text{(37)} \qquad \begin{aligned} &\text{Subject to : } \nabla g^i(y^j)^T \Delta y^j \leqslant -g^i(y^j) - \bar{k} r^{ij} \\ &\text{Minimize : } \nabla g^m(y^j)^T \Delta y^j \end{aligned}$$

The r^{ij} term is the estimated error for the i^{th} equation during the $j + 1^{th}$ nonlinear iteration. The value of r^{ij} is determined during the nonlinear iteration using the following equation :

$$\text{(38)} \qquad r^{ij} = g^i(y^{j-1} + \Delta y^{j-1}) - g^i(y^{j-1}) - \nabla g^i(y^{j-1})^T y^{j-1}$$

where k is implicitly assumed to be one. The absolute value of r^{ij} is used for the linear programming problem.

The parameter $\bar{k}$ will be adjusted in the course of the nonlinear iteration. The value of $\bar{k}$ is greater than, or equal to zero, and is estimated from the length of the previous step. The role of $\bar{k}$ in the linear and nonlinear problem will be discussed in detail later in this section, as will the estimating procedures used to obtain $\bar{k}$.

The linear programming problem (37) can be treated as a parametric programming problem with $\bar{k}$ as the modifying parameter and can be written in tableau form as illustrated in Tableau (1).

The vectors Δy^j and v are $nx1$. The members of vector Δy^j are the primal variables, and the members of vector v are the slack variables of the dual problem. The vector of the dual variables is

$$\mathrm{x}^T = [\mathrm{x}_1, \ldots\ldots, \mathrm{x}_{m-1}]$$

and the vector of the primal slack variables is

$$\mathrm{z}^T = [\mathrm{z}_1, \ldots\ldots, \mathrm{z}_{m-1}]$$

Φ_p is the value of the primal objective function and Φ_d is the value of the dual objective function. The labels $(VBV)_p$ and $(VBV)_d$ are respectively the values of the current basic primal variables and the basic dual variables. The labels $(BV)_p$ and $(BV)_d$ are the basic variables associated with the given values.

TABLEAU 1

	Δy^j	$(VBV)_p$	$(BV)_p$
$-x_1$ $\vdots$ $-x_{m-1}$	$\nabla g^1(y^j)^T$ $\vdots$ $\nabla g^{m-1}(y^j)^T$	$-g^1(y^j)-\bar{k}r^{1j}$ $\vdots$ $-g^{m-1}(y^j)-\bar{k}r^{m-1,j}$	z_1 $\vdots$ z_{m-1}
$(VBV)_d$	$\nabla g^m(y^j)^T$	$-g^m(y^j)-\bar{k}r^{mj}$	$-\Phi_p$
$(BV)_d$	$-v$	$-\Phi_d$	

In order to generate a typical tableau of the linear programming problem, Tableau 1 can be rearranged such that the first n_3 columns are associated with the entering primal variables, and the first n_3 rows are associated with the leaving slack varibles. This is done in Tableau 2.

TABLEAU 2

	(Δy^j)	$(\Delta y^j)_2$	$(VBV)_p$	$(BV)_p$
$(-x)_1$ $(-x)_2$	B_1 B_2	C_1 C_2	d_1 d_2	$(z)_1$ $(z)_2$
$(VBV)_d$	B_3	C_3	d_3	$-\Phi_p$
$(BV)_d$	$(-v)_1$	$(-v)_2$	$-\Phi_d$	

B_1 is a $n_3 x n_3$ matrix, where $n_3 \leqslant \min(m-1, n)$.

The n_3 variables are brought into the basis by block pivoting on the matrix B_1 as shown in Tableau 3.

TABLEAU 3

	$(x)_1$	$(\Delta y^j)_2$	$(VBV)_p$	$(BV)_p$
$(-v)_1$ $(-x)_2$	B_1^{-1} $-B_2B_1^{-1}$	C_1 C_2	$B_1^{-1}d_1$ $d_2 - B_2B_1^{-1}d_1$	$(\Delta y^j)_1$ $(z)_2$
$(VBV)_d$	$-B_3B_1^{-1}$	C_3	$d_3 - B_3B_1^{-1}d_1$	$-\Phi_p$
$(BV)_d$	$(-x)_1$	$(-v)_2$	$-\Phi_d$	

The current values of the primal variables as displayed in Tableau 3 are

$$y^j = \begin{bmatrix}(\Delta y^j)_1\\(\Delta y^j)_2\end{bmatrix} = \begin{bmatrix}B_1^{-1}d_1\\0\end{bmatrix}$$

The current values of the dual variables are :

$$x = \begin{bmatrix}(x)_1\\(x)_2\end{bmatrix} \begin{matrix}=\\=\end{matrix} \begin{bmatrix}B_3B_1^{-1}\\0\end{bmatrix}$$

Let

$$C = \begin{bmatrix}C_1\\C_2\\C_3\end{bmatrix}, \qquad B = \begin{bmatrix}B_1\\B_2\\B_3\end{bmatrix}$$

Note that the matrix C was not updated by the pivoting operation. This was done to reflect the internal operation of the algorithm in its computer application. The matrix is not stored as such, but is generated as needed during the course of algorithm, and updated by the use of the updated B matrix. Since at the present point all of the members of the vector Δy^j associated with C are zero, it is not necessary to know the values of the updated C.

A typical column in C is

$$c_i = \begin{bmatrix}\dfrac{\partial g^1(y^j)}{\partial y_i}\\ \cdot\\ \cdot\\ \cdot\\ \cdot\\ \dfrac{\partial g^{m-1}(y^j)}{\partial y_i}\end{bmatrix}$$

If it is determined to introduce Δy_i into the basis, then it is necessary to update C^i. In order to illustrate this procedure let

$$C_i = \begin{bmatrix} C_1^i \\ C_2^i \\ C_3^i \end{bmatrix}$$

where C_1^i, C_2^i and C_3^i correspond to C_1, C_2 and C_3 in Tableau 2. The updated version of C^i is

$$C^i = \begin{bmatrix} B_1^{-1}C_1^i \\ C_2^i - B_2B_1^{-1}C_1^i \\ C_3^i - B_3B_1^{-1}C_1^i \end{bmatrix}$$

Since at any point in the algorithm the values of

$$B_1^{-1}, -B_2B_1^{-1} \text{ and } -B_3B_1^{-1}$$

are stored, the column C^i can easily be updated. This feature of the program reduces the necessary storage requirements, which is especially important in large scale problems.

From the duality theorem of linear programming there are three possible termination conditions to the local linear programming problem (42).

(A) There exists a finite value V and feasible vectors

$$y^{j*} \text{ and } x^* \text{ such that}$$

$$B_3\Delta y^{j*} = d_1x^* = v$$

The vectors y^{j*} and x^* are the optimal solution to (37).

(B) The constraints for the primal problem are infeasible and the dual problem is unbounded or the constraints of the dual problem are inconsistent.

(C) The primal problem is unbounded and the dual problem is infeasible.

The initial solution in the domain of the functions $g^i(y^j)$, $i = 1, m$, y^j, is not required to be a feasible solution to the nonlinear problem stated in (34). As was discussed above, if y^j is not a feasible solution to (34), then SUPG > 0. If y^j is a feasible solution to (34), then SUPG $= 0$. The goal of each nonlinear iteration is either to reduce the value of the objective function $g^m(y^j)$, or move closer to feasibility, which is interpreted to mean reduce the value of SUPG.

The case of nonlinear infeasibility will usually imply that the local linear problem (37) will be infeasible for any Δy^j in the ε region around y^j. In this case $\nabla g(y^j)^T \Delta y^j$ is chosen as the objective function and a linear programming problem such as (39) will be constructed.

$$(39)\quad \begin{aligned} &\text{Subject to : } \nabla g^p(y^j)^T \Delta y^j \leqslant -g^p(y^j) - \bar{k} r^{pj} \qquad p \in H \\ &\text{Minimize : } \nabla g^w(y^j)^T \Delta y^j \end{aligned}$$

where

$$H = \{ i \mid \nabla g^i(y^j)^T \Delta y \leqslant -g^i(y^j) - \bar{k} r^{ij} \text{ for some } \Delta y^j \}$$

Since this problem is consistent and bounded it can only terminate in condition (A). However, it is still possible that the entire linear problem (37) is infeasible, or in other words terminates in condition (B). In this case the honlinear problem will also be infeasible at the new point, y^{j+1}, ignoring errors. Mathematically this would mean that

$$\nabla g^i(y^j)^T \Delta y^j > -g^i(y^j) - \bar{k} r^{ij} \text{ for some } i,\ i = 1, m - 1.$$

However, if a gain has been made in SUPG, then the algorithm proceeds through the nonlinear iteration with the determination of k. Of course, if the gain is large enough feasibility may be reached and the local linear problem would terminate in condition (A).

If the local linear programming problem (37) terminates with condition (B) bolding and no gain has been made in SUPG, then it is assumed that the nonlinear problem is inconsistent and the algorithm terminates unless $\bar{k}$ can be adjusted as will be discussed later.

The other possible termination condition (A) implies that a feasible solution to the entire linear problem (37) has been obtained, and ignoring errors, y^{j+1} is a feasible solution to the nonlinear problem. The algorithm at this point will check for a gain in $g^m(y)$. If at the new $y^{j+1} = y^j + \Delta y^i$, there is no gain in $g^m(y)$, then we assume that the local minimum has been reached. If there is a gain in $g^m(y)$, then another nonlinear step is taken. A graphical interpretation of these decision rules is shown in Figure 2, and Figure 3 which will be explained in detail.

At this point it is necessary to explain in some detail the role that the parameter $\bar{k}$ plays in the final determination of Δy^j. In order to see the role $\bar{k}$ plays more clearly, it is necessary to write mathematical expressions for the statements, « gain in $g^m(y)$, » and « gain in SUPG ». If the local linear problem terminates in condition (A), then

$$(40)\qquad \nabla g^m(y^j)^T y^j = B_3 B_1^{-1} d_1$$

or

$$(41)\qquad \nabla g^m(y^j)^T \Delta y^j = \sum_{i=1}^{m-1} (-x_i) g^i(y^j) + \bar{k} \sum_{i=1}^{m-1} (-x_i) r^{ij}$$

In order for a gain to be made in the nonlinear objective function, the following inequality must hold :

$$\Delta g^m(y^j)^T \Delta y^j < -\varepsilon \tag{42}$$

Using equations (41) and (42), the condition for a gain in $g^m(y^j)$ is

$$\sum_{i=1}^{m-1} (-x_i) g^i(y^j) + \bar{k} \sum_{i=1}^{m-1} (-x_i) r^{ij} < -\varepsilon \tag{43}$$

At this point, the dual variables are less than, or equal to zero. The r^{ij} are assumed greater than or equal to zero, and since the nonlinear problem is feasible $g^i(y^j) < 0$. This information implies that :

$$\sum_{i=1}^{m-1} (-x_i) g^i(y^j) < 0 \tag{44}$$

and

$$\bar{k} \sum_{i=1}^{m-1} (-x_i) r^{ij} > 0 \tag{45}$$

From (43) and (45), it is clear that as $\bar{k}$ approaches zero, the gain in $g^m(y^j)$ would be greater. Therefore, if the linear problem terminaties in condition (A), and there is no gain in $g^m(y)$, then $\bar{k}$ can be adjusted downwards, which effectively is relaxing the linear constraints. As $\bar{k}$ goes to zero, condition (43) becomes

$$\sum_{i=1}^{m-1} (-x_i) g^i(y^j) < -\varepsilon \tag{46}$$

The same sort of condition can be derived in the case when the linear programming problem terminates in condition (B). Condition (47) is for a gain in SUPG.

$$\sum_{p} (-x_p) g^p(y^j) < -\varepsilon \qquad p \in H \tag{47}$$

Using these results, the criteria that the algorithm uses for nonlinear optimality and nonlinear infeasibility can be written as follows :

Optimality :

If $\bar{k}$ is adjusted as low as possible, the linear program terminates in condition (A) and

$$\sum_{i=1}^{m-1} (-x_i) g^i(y^j) \geqslant -\varepsilon$$

then y^j is assumed to be the optimal solution to the nonlinear problem.

Infeasibility :

If $\bar{k}$ is adjusted as low as possible, the linear program terminates in condition (B), and

$$\sum_p (-x_p) g^p(y^j) \geqslant -\varepsilon \qquad p \in H$$

then the contraints of the nonlinear problem are assumed to be inconsistent.

Using the criteria described above for optimality and infeasibility, the steps actually taken in the computer program will be described in sequence using Figure 2, and Figure 3.

Before the steps of the actual program are discussed one additional feature of this algorithm must be mentioned. It is possible to divide the n variables in the nonlinear problem into IPRN priority classes. For example, assume that IPRN $= 2$. This implies that every variable is either in priority class one or two. All of the variables in priority class one would be used to try and obtain a gain in SUPG or $g^m(y)$. The second priority class variables would not be considered unless no gain could be made using the priority one variables with $\bar{k}$ adjusted to zero. The number of priority classes is unlimited.

The procedure followed by the algorithm in a dynamic sense is as follows :

1. The variables, Δy^j, associated with the matrix C, which are currently not in the basis of the linear programming problem, are scanned for possible entry. All of the variables will be out of the basis at the outset of each nonlinear iteration. The scanning is accomplished by updating the element in each column of C associated with the current linear objective function. If priority classes are used, then only those variables in C which have a priority level less than or equal to the current level, IPRC, are checked. See Box 1 in Figure 2.
2. The variable associated with the updated element of highest absolute value is selected to enter the linear tableau. This criteria is used because this variable locally affects the objective function more than the other variables. See Box 2 in Figure 2.
3. The element with the largest absolute value is tested to see if it is significantly different from zero. If it is, the algorithm proceeds to step 4 and the solution of the linear programming problem. If not, it is assumed that the addition of the variable associated with the largest element would not affect the objective function significantly, since the appropriate coefficient is so small. In this case, $\bar{k}$ is adjusted downwards, or if $\bar{k} = 0$, the number of priority classes considered is expanded. If the current priority class is the last one available, then the algorithm will terminate. The termination will mean one of two things; the nonlinear problem is infeasible since no gain can be made in SUPG $= g^w(y) > 0$, or the local minimum to the nonlinear problem has been attained and no gain can be made in $g^m(y)$. See Figure 3.

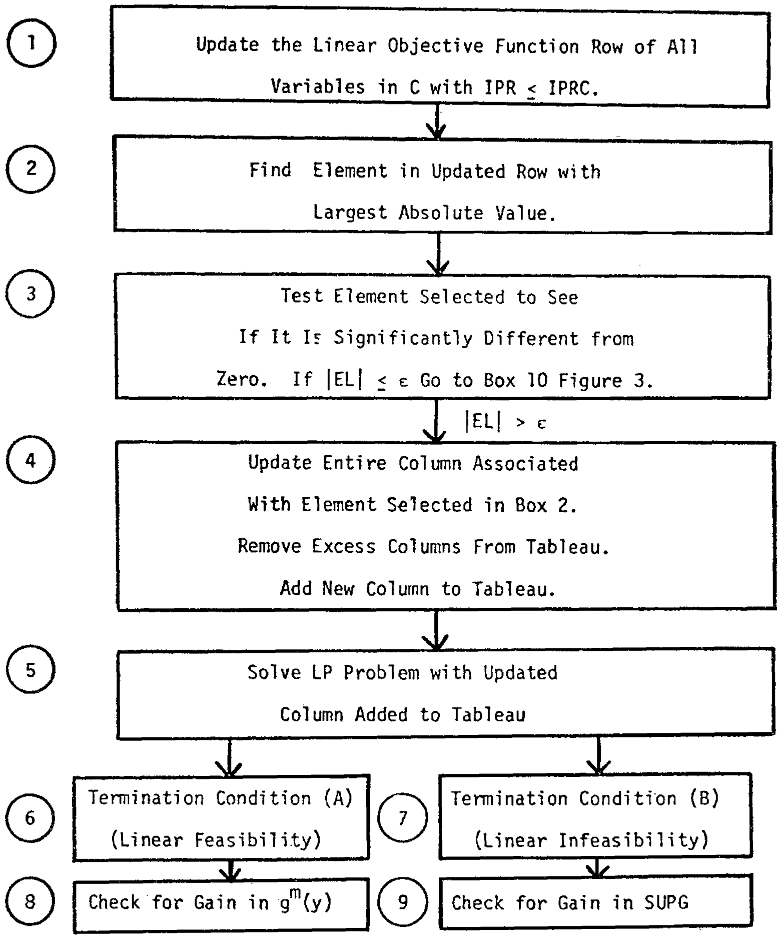

1
Update the Linear Objective Function Row of All Variables in C with IPR ≤ IPRC.
2
Find Element in Updated Row with Largest Absolute Value.
3
Test Element Selected to See If It Is Significantly Different from Zero. If |EL| ≤ ε Go to Box 10 Figure 3.
|EL| > ε
4
Update Entire Column Associated With Element Selected in Box 2. Remove Excess Columns From Tableau. Add New Column to Tableau.
5
Solve LP Problem with Updated Column Added to Tableau
6
Termination Condition (A) (Linear Feasibility)
7
Termination Condition (B) (Linear Infeasibility)
8
Check for Gain in g^m(y)
9
Check for Gain in SUPG

Figure 2
Selection of Variable and Solution to LP Problem

4. After selecting the variable column to be added to the linear programming problem, the entire column in *C* associated with the selected variable is updated. At this point, the linear programming tableau is augmented by this new column and if a gain was made in the linear objective function on the previous linear iteration, the variables rejected from the basis are removed from the linear tableau. See Box 4 Figure 2.
5. The linear programming algorithm now takes over and solves the local problem set up in the previous steps. The linear programming problem will terminate in one of the three terminal conditions discussed above. If terminal condition (*A*) is reached, linear feasibility, then the objective

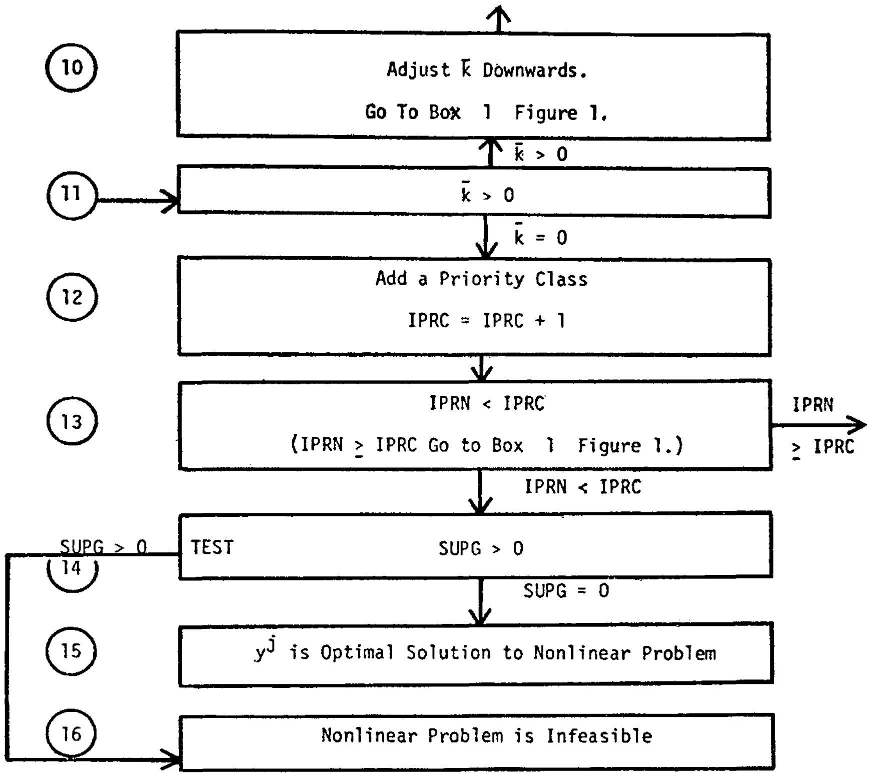


Figure 3
Adjustment of k and Priority Classes

function $g^m(y)$ is tested to see if a gain greater than some tolerance level was made. If terminal condition (B) is attained, then the algorithm tests for a sufficient gain in SUPG. If either of these gains are successfully made, the algorithm leaves the linear programming subproblem. If the gains are not made, then the algorithm returns to selecting variables to enter the linear tableau. If terminal condition (C) is reached, the algorithm again returns to selecting variables in order to bound the primal problem. See Boxes 5-9 in Figure 2.

The first step after the successful conclusion of the local linear problem is to select the « best » value of $\bar{k}$. This calculation is called the post-optimal adjustment and proceeds in two different manners, depending on the value of SUPG.

If the value of SUPG is zero, or we have nonlinear feasibility, then the value of the objective function is written as a function of $\bar{k}$, and this function is solved for the value of $\bar{k}$, which will minimize $g^m(y)$.

In the case where SUPG < 0, or we have nonlinear infeasibility, $\bar{k}$ is approximated by choosing a trial value such that G is zero, where G is defined to be

$$G = (\Delta y^{j-1})^T(\Delta y^{j-1}) - (\Delta y^j)^T(\Delta y^j) \tag{48}$$

The value of $(\Delta y^{j-1})^T(\Delta y^{j-1})$ is stored from nonlinear iteration $j - 1$. The value of $(\Delta y^j)^T(\Delta y^j)$ can be written as a function of $\bar{k}$. We know that

$$\Delta y^j = \begin{bmatrix} -B_1^{-1}d_1 \\ 0 \end{bmatrix} \tag{49}$$

or

$$\Delta y^j = \begin{bmatrix} (BT) - \bar{k}(PT) \\ 0 \end{bmatrix} \tag{50}$$

where

$$BT = -B_1^{-1}g^p(y^j) \tag{51}$$

and

$$PT = -\bar{k}B_1^{-1}r^{pj} \tag{52}$$

Using (50), (51) and (52) G can be written as a quadratic function in $\bar{k}$ which can be solved for $\bar{k}$.

In either case, SUPG = 0, or SUPG > 0, the value of $\bar{k}$ must be tested to see that it does not violate bounds which are implied by the bounds on y. Since the value of y^{j+1} must satisfy equation (36), and we know that Δy^j is a function of $\bar{k}$ from equation (50), equation (35) can be written

$$BL \leqslant y^j - B_1^{-1}g^p(y^j) - \bar{k}B_1^{-1}r^{pj} \leqslant UB \tag{53}$$

Solving for $\bar{k}$ and assuming $PT > 0$:

$$\frac{BL - y^j}{PT} - BT \leqslant \bar{k} \leqslant \frac{UB - y^j}{PT} - BT \tag{54}$$

or if $PT > 0$

$$\frac{BL - y^j}{PT} - BT \geqslant \bar{k} \geqslant \frac{UB - y^j}{PT} - BT \tag{55}$$

If the value of $\bar{k}$ determined in the post optimal adjustment violates the greatest lower bound on $\bar{k}$, or results, in the case of SUPG = 0, in no gain in $g^m(y)$, the value of $\bar{k}$ is adjusted downwards. The control of the problem is then passed back to the linear programming part of the algorithm.

The first part of the nonlinear iteration is now completed. Note that each adjustment in $\bar{k}$ will change the optimal value of Δy^j. Therefore, the determination of Δy^j is not complete until the post-optimal adjustment is finished.

It is at this point the new estimates of the error terms r^{ij} are calculated. This is done by evaluating the functions $g^i(y^j + \Delta y^j)$, $i = 1, m - 1$, and using equation (38). The new values or r^{ij} are used for the rest of the $j + 1$ — th nonlinear iteration and the absolute values are used for the local linear problems in nonlinear iteration $j + 2$.

The next step in the algorithm is to calculate the range of values for k which will maintain a feasible solution or give the best gain in SUPG. After a range of values is determined, the optimal value of k is chosen from the range determined.

The range is calculated by using the following equation :

$$(56) \qquad g^i(y) = g^i(y^j) + g^i\nabla(y^j)\Delta y^j k + k^2 r^{ij} \leqslant D\,(\mathrm{SUPG})$$

If SUPG $= 0$ then equation (56) just says that the new values of $g^i(y)$, $i = 1, m - 1$, must be feasible. If SUPG > 0, then D is initially set equal to zero. The quadratic functions in k are then solved.

$$g^i(y^j) - D\,\mathrm{SUPG} + \nabla g^i(y^j)\Delta y^j k + k^2 r^{ij} = 0, \qquad i = 1, m - 1$$

This will give the value or values of k where the constraints $g^i(y)$ will go infeasible. If the lower bound on k is greater than the upper bound, then the value of D is increased and the quadratic problem is again solved. If as D approaches one, the upper bound continues to be lower, than the lower bound, then we say that the interval determination failed and no k can be found which will improve SUPG. In this case the algorithm terminates.

If SUPG $= 0$, the algorithm determines the interval which will maintain the feasibility of y. After the interval is determined, the best value of k is found by evaluating the function $g^m(y^j + k\Delta y^j)$ for different k's in the range given. The gain in the objective function is checked to see if it exceeds some preset tolerance level. If not, than it is assumed that we are within the lenght of that tolerance level of a local optimal solution and the algorithm terminates.

If a gain in SUPG is made, or a reduction in $g^m(y)$, the algorithm takes another nonlinear iteration. This procedure is repeated until either a local minimum or infeasibility is encountered.

The programming code of the algorithm described above is made up of sixteen subroutines as illustrated in Figure 4. The roles of some of the major subroutines are explained as follows :

A. NONLIN : A single pass through NONLIN completes a nonlinear iteration. The role of NONLIN is to call in order the linear programming subroutines and the subroutines which adjust the values of k and $\bar{k}$.

B. SETUP : A specific problem will require the reading in of data, the calculation of certain parameters and other necessary housekeeping. SETUP fulfills this role and will be unique for any given problem.

C. FCNGEN : It is necessary at certain points in the algorithm to evaluate the functions $g^i(y)$ for some specific y. FCNGEN provides this information and is also unique for any given problem.

D. LINP : The subroutines which solve the local linear programming problems are controlled by the subroutine LINP.

E. RCK : The selection of the variables to enter the basis is done by RCK. This is done by updating the appropriate element of the column in C associated with the variable being checked.

F. COLGEN : It is necessary to evaluate the column of C associated with a variable entering the basis for the current value of y. This is done in COLGEN.

G. POAD : The post-optimal adjustment is done in POAD.

H. INCON : The parameter $\bar{k}$ is adjusted downwards when required in INCON.

I. INVDET : The bounds on k are calculated in INVDET.

J. GENMIN : After the bounds on k have been calculated, if SUPG $= 0$, then GENMIN is called to find the optimal value of k.

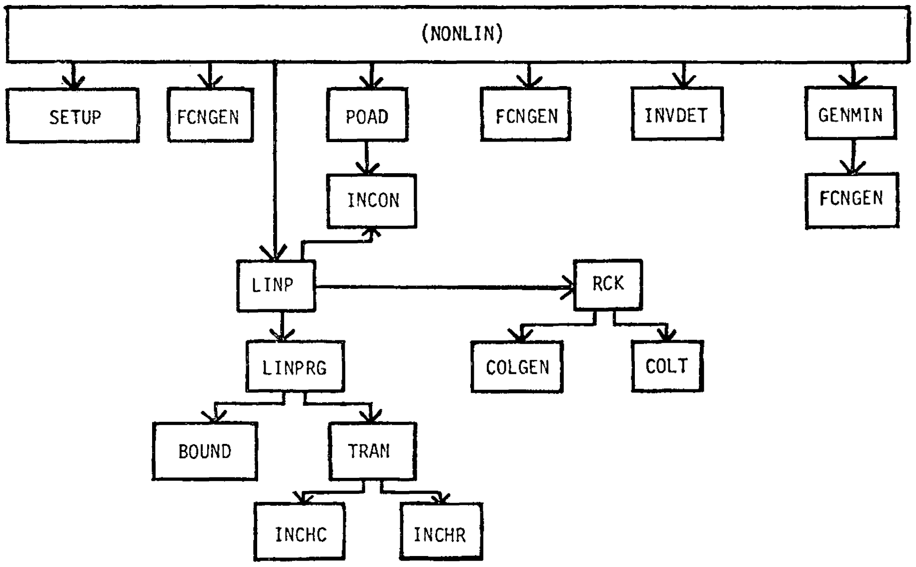


Figure 4

Nonlinear Programming Algorithm

Appendix A

Water-Quality Model Notation
Partial Derivative Blocks

a_g	Length of section *g*
a_{gi}	Length of pipe from polluter *i* to section *g*
a_{mi}	Length of pipe from polluter *i* to treatment plant *m*
a_{gm}	Length of pipe from treatment plant *m* to section *g*
$\bar{b}$	BOD of effluent from polluter *i*
b_g^0	BOD of river at headwaters of section *g*
b_{g1}	BOD of incremental flow entering section *g*
b_{g2}	BOD of effluent flow entering section *g*
b_{g3}	BOD of augmentation flow entering section *g*
C_l	Cost per flow unit of augmentation flow from reservoir l
C_{gi}^L	Cost of pipe from polluter *i* to section *g*
C_{mi}^L	Cost of pipe from polluter *i* to treatment plant *m*
C_{gm}^L	Cost of pipe from treatment plant *m* to section *g*
C^P	Cost of treatment plants
C^R	Cost of Reservoirs
$\bar{d}_i$	DOD of effluent from polluter *i*
d_g	DOD at end of section *g*
d_g^*	DOD goal in section *g*
d_g^0	DOD at headwaters of section *g*
d_{g1}	DOD of incremental flow entering section *g*
d_{g2}	DOD of effluent flow entering section *g*
d_{g3}	DOD of augmentation flow entering section *g*
f_{gi}	Flow from polluter *i* to section *i*
F_{g1}	Incremental flow entering section *g*
F_{g2}	Effluent flow entering section *g*
F_{g3}	Augmentation flow entering section *g*
K_{1g}	Rate of deoxygenat in section *g*
K_{1g}^{20}	Rate of deoxygenation at 20° C in section *g*
K_{2g}	Rate of reaeration in section *g*
p_{mi}	Flow from polluter *i* to treatment plant *m*
r_m	Percent removal of BOD at treatment plant *m*
t_{gm}	Flow from treatment plant *m* to section *g*
T	Temperature
v_g	Velocity of flow in river section *g*
α_g	Estimated parameter
β_g	Estimated parameter
γ_g	Estimated parameter

σ_g	Estimated parameter
θ_1	Estimated parameter
θ_2	Estimated parameter

APPENDIX B

Typical Element of Block Λ_{14}

$$\frac{\partial d_g}{\partial r_m} = K_g \frac{\partial b_g^0}{\partial r_m} [C_{1g} - C_{2g}] + \frac{\partial d_g}{\partial r_m} C_{2g}$$

The particular form of the partial derivative depends on the relation between g and m. The crucial questions are whether or not the effluent from treatment plant m is being dumped upstream, downstream or in section g and whether or not g is a tributary.

Let $z = \text{Min}\{g \mid t_{gm} > 0\}$ and $Z = \{g \mid t_{gm} > 0 \text{ and } g \neq z\}$.

Case (a)

If $g < z$

or if $g > z$ and $g \notin Z$ and g is a tributary

then $\frac{\partial d_g^0}{\partial r_m} = \frac{\partial b_g^0}{\partial r_m} = 0$

Case (b)

If $g = z$

then

$$\frac{\partial b_g^0}{\partial r_m} = -\left[\sum_i \bar{b}_i p_{mi} / \sum_i p_{mi}\right] t_{gm} (1/F_g)$$

$$\frac{\partial d_g^0}{\partial r_m} = 0$$

Case (c)

If $g > z$, $g \notin Z$, and g and $g-1$ are not tributaries

then

$$\frac{\partial b_g^0}{\partial r_m} = \left(\frac{\partial b_{g-1}^0}{\partial r_m}\right) \frac{C_{1,g-1}}{F_g}$$

$$\frac{\partial d_g^0}{\partial r_m} = \left(\frac{\partial d_{g-1}}{\partial r_m}\right) \frac{1}{F_g}$$

Case (d)

If $g > z$, $g \notin Z$, g is not a tributary, and $g - 1$ is a tributary then

$$\frac{\partial b_g^0}{\partial r_m} = \left(\frac{\partial b_{g-1}^0}{\partial r_m}\right)\frac{C_{1,g-1}}{F_g} + \left(\frac{\partial b_{g-2}^0}{\partial r_m}\right)\frac{C_{1,g-2}}{F_g}$$

$$\frac{\partial d_g^0}{\partial r_m} = \left(\frac{\partial d_{g-1}}{\partial r_m}\right)\left(\frac{1}{F_g}\right) + \left(\frac{\partial d_{g-2}}{\partial r_m}\right)\left(\frac{1}{F_g}\right)$$

Case (e)

If $g > z$, $g \in Z$, and g and $g - 1$ are not tributaries then

$$\frac{\partial b_g^0}{\partial r_m} = \frac{\partial b_g^0}{\partial r_m}\bigg|\ \text{Case } (c) \quad + \frac{\partial b_g^0}{\partial r_m}\bigg|\ \text{case } (b)$$

$$\frac{\partial d_g^0}{\partial r_m} = \frac{\partial d_g^0}{\partial r_m}\bigg|\ \text{Case } (c) \quad + \frac{\partial d_g^0}{\partial r_m}\bigg|\ \text{Case } (b)$$

Case (f)

If $g > z$, $g \in Z$, g is not a tributary, and $g - 1$ is a tributary then

$$\frac{\partial b_g^0}{\partial r_m} = \frac{\partial b_g^0}{\partial r_m}\bigg|\ \text{Case } (d) \quad + \frac{\partial b_g^0}{\partial r_m}\bigg|\ \text{Case } (b)$$

$$\frac{\partial d_g^0}{\partial r_m} = \frac{\partial d_g^0}{\partial r_m}\bigg|\ \text{Case } (d) \quad + \frac{\partial d_g^0}{\partial r_m}\bigg|\ \text{Case } (b)$$

Typical Element of Block Λ_{15}

$$\frac{\partial d_g}{\partial F_{l3}} = \left(\frac{\partial K_g}{\partial F_{l3}}\right) b_g^{A0}[C_{1g} - C_{2g}] + K_g \left(\frac{\partial b_g^0}{\partial F_{l3}}\right)[C_{1g} - C_{2g}]$$
$$+ K_g b_g^0 \left[\frac{\partial C_{1g}}{\partial F_{l3}} - \frac{\partial C_{2g}}{\partial F_{l3}}\right] + \left(\frac{\partial d_g^0}{\partial F_{l3}}\right) C_{2g} + d_g^0 \left(\frac{\partial C_{2g}}{\partial F_{l3}}\right)$$

where,

$$\frac{\partial K_g}{\partial F_{l3}} = \alpha_g F_g^{\beta_g - 1}$$

$$\frac{\partial C_{1g}}{\partial F_{l3}} = C_{1g} K_{1g} a_g \sigma_g \gamma_g^{-1} F_g^{-\sigma_g - 1}$$

$$\frac{\partial C_{2g}}{\partial F_{l3}} = - C_{2g} \alpha_g a_g \gamma_g^{-1} (\beta_g - \sigma_g) F_g^{\beta_g - \sigma_g - 1}$$

The values of

$$\frac{\partial b_g^0}{\partial F_{l3}} \text{ and } \frac{\partial d_g^0}{\partial F_{l3}}$$

depend on the relation between g and l.

Case (a)

If $g < l$ or if $g > l$ and g is a tributary, then

$$\left(\frac{\partial b_g^0}{\partial F_{l3}}\right) = \left(\frac{\partial d_g^0}{\partial F_{l3}}\right) = 0$$

Case (b)

If $g = l$, then

$$\frac{\partial b_g^0}{\partial F_{l3}} = (b_{l3} - b_g^0)/F_g$$

$$\frac{\partial d_g^0}{\partial F_{l3}} = (d_{l3}^0 - d_g^0)/F_g$$

Case (c)

If $g > l$, g is not a tributary, and g-1 is not a tributary, then

$$\frac{\partial d_g^0}{\partial F_{l3}} = (b_{g-1}^0 C_{1,g-1} - b_g^0)/F_g$$

$$\frac{\partial d_g^0}{\partial F_{l3}} = (d_{g-1}^0 - d_g^0)/F_g$$

Case (d)

If $g > l$, and g is not a tributary, and $g - 1$ is a tributary, then

$$\frac{\partial b_g^0}{\partial F_{l3}} = (b_{g-2}^0 C_{1,g-2} - b_g^0)/F_g$$

$$\frac{\partial d_k^B}{\partial F_l{}^3} = (d_{g-2}^0 - d_g^0)/F_g$$

BIBLIOGRAPHY

1. CLOUGH D. J. and BAYER M. B., « Optimal Waste Treatment and Pollution Benefits on a Closed River System », *Canadian Operational Research Journal*, vol. 6, N° 3, nov. 1968.
2. DEININGER R. A., « Water Quality Management; Economically Optimal Pollution Control System », Unpublished Ph. D. Dissertation, North-western University, Evanston, Illinois, 1964.
3. DOBBINS William E., « BOD and Oxygen Relationships in Streams », *Journal San. Eng. Div.*, Proc. Amer. Soc. Civil Engr., 90, N° SA3, june 1964, p. 53.

4. Frankel R. J., « Economic Evaluation of Water Quality; an Engineering Economic Model for Water Quality Management », *SERL Report* N° 65-3, University of California, Berkeley, Calif., jan. 1965.
5. Graves G., « A Complete Constructive Algorithm for the General Mixed Linear Programming Problem », *Naval Research Logistics Quarterly*, vol. 12, 1965.
6. Graves G. W. and Whinston A., « The Application of a Nonlinear Algorithm to a Second Order Representation of the Problem », *Centre d'Etudes de Recherche, opérationnelle*, volume 11, N° 2, 1969.
7. Graves G., « Development and Testing of a Nonlinear Programming Algorithm », Aerospace Corporation, june 1964.
8. Graves G. W., Hatfield G. B. and Whinston A., « Water Pollution Control Using By-Pass Piping », *Water Resources Research*, vol. 5, N° 1, feb. 1969.
9. Graves G. W., Hatfield G. B. and Whinston A., « Water Pollution Control with Regional Treatment », Technical Report, Federal Water Pollution Control Administration, Forthcoming.
10. Graves G. W., Pingry D. E. and Whinston A., « Application of a Large-Scale Nonlinear Programming Problem to Pollution Control », American Federation of Information Processing Societies, *Proceedings Fall Joint Computer Conference*, vol. 39, 1971.
11. Jaworski Norbert A., Weber Walter J. Jr. and Deininger Rolf A., « Optimal Reservoir Releases for Water Quality Control », *Journal of the Sanitary Engineering Division*, ASCE, vol. 96, N° SA3, june 1970.
12. Linaweaver F. P. and Clark C. S., « Cost of Water Transmission », *Journal of American Water Works Association*, 1549-1560, december 1964.
13. Loucks D. P., Revelle C. S. and Lynn W. R., « Linear Programming Models for Water Pollution Control », *Management Science*, vol. 14, N° 4, dec. 1967.
14. O'Conner Donald J., « The Temporal and Spocial Distribution of Dissolved Oxygen in Streams », *Water Resources Research*, vol. 3, N° 1, 1967.
15. Schaumburg G. W., *Water Pollution Control in the Delaware Estuary*, Harvard University Water Program, Harvard University, Cambridge, Massachussets, may 1967.
16. Streeter H. W. and Phelps E. B., « A Study of the Pollution and Natural Purification of the Ohio River », *U.S. Public Health Bulletin*, N° 146, feb. 1925.
17. Thomann R. V., « Mathematical Model for Dissolved Oxygen », *Journal San. Eng. Div.*, Proc. Amer. Soc., Civil Engr., 89, N° SA5, oct. 1963.
18. Worley J. L., Burgess F. J. and Towne W. W., « Identification of Low-Flow Augmentation Requirements for Water Quality Control by Computer Techniques », *Journal Water Pollution Control Federation*, vol. 37, N° 5, may 1965.