FAST INTERSECTION METHODS FOR THE SOLUTION OF SOME NONLINEAR SYSTEMS OF EQUATIONS

BERNARD BEAUZAMY

Received 13 July 2003 and in revised form 23 February 2004

We give a fast method to solve numerically some systems of nonlinear equations. This method applies basically to all systems which can be put in the form $U \circ V(X) = Y$, where U and V are two possibly nonlinear operators. It uses a modification of Newton's algorithm, in the sense that one projects alternatively onto two subsets. But, here, these subsets are not subspaces any more, but manifolds in a Euclidean space.

1. Introduction

As we will see below (Section 3), a water distribution problem can be put in the form

$$\Lambda f(AX + B) = D, \tag{1.1}$$

where $\Lambda : n \times N, A : N \times n, X : n \times 1, B : N \times 1$, and $D : n \times 1$ are matrices with real entries. The unknowns X represent the pressures (to be determined) at all points of the water distribution system. The matrices Λ, A, B, D are known.

The function f is defined by the formula

$$f(x) = x|x| \tag{1.2}$$

and this is a differentiable increasing function which is invertible. The inverse is the function

$$g(x) = \sqrt{|x|}\ \text{sign}\ (x). \tag{1.3}$$

In problem (1.1), the meaning of $f(X)$, if

$$X = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}, \tag{1.4}$$

Journal of Applied Mathematics 2004:2 (2004) 127–136
2000 Mathematics Subject Classification: 65D99, 14Q10
URL: http://dx.doi.org/10.1155/S1110757X04307084

is of course

$$f(X) = \begin{pmatrix} f(x_1) \\ \vdots \\ f(x_n) \end{pmatrix}. \tag{1.5}$$

So, we see that problem (1.1) is well posed: there are as many unknowns as equations, but the problem is nonlinear, due to the presence of the function f. If f was a linear function, the problem would easily be solved by the usual techniques of linear algebra.

The classical approaches to problem (1.1), as it stands here, are by means of optimization techniques. Indeed if one wants to solve $G = 0$, one looks for the minima of F, where $F' = G$. See [5, 6, 10] for precise descriptions of these algorithms.

Techniques for nonlinear optimization work in a satisfactory manner for problems of limited size, but here the unknowns, for large systems, may be of order 10 000, 100 000, or more. So fast techniques are needed if one wants the computation to be performed in a reasonable time. The technique we present here is much faster than optimization tools, for it uses only intersections of subspaces; see [8].

2. The intersection procedure

We introduce two subsets of $\mathbb{R}^N$

$$\begin{aligned} F &= \{Y \in \mathbb{R}^N;\ \Lambda Y = D\}, \\ G &= \{Y \in \mathbb{R}^N;\ \exists X \in \mathbb{R}^n,\ Y = f(AX + B)\}. \end{aligned} \tag{2.1}$$

We look for the intersection of F and G. Indeed, if $Y \in F$, $\Lambda Y = D$ and if $Y \in G$, there exist X such that $Y = f(AX + B)$. Therefore we get (1.1). The set F is an affine subspace of $\mathbb{R}^N$ (translate of a linear subspace). The set G is not an affine subspace, due to the function f which is not linear, it is precisely a manifold, contained in $\mathbb{R}^N$. But, as we will see later, we will still work on an affine subspace. We still speak of a "subspace," though it is formally a manifold.

Because N is usually much larger than n, neither F nor G is reduced to single points. They are usually large-dimensional subspaces. Therefore, it would be unwise to try to get a complete formal description of both of them and then take the intersection. Such an approach would be very slow, costly, and quite useless, we want to find the intersection, not to describe both.

In order to find the intersection of F and G, we will use a Newton-type algorithm. We start at any given point and we project alternatively upon F and G. We now describe this procedure.

2.1. Projecting upon the affine subspace F. The orthogonal projection onto a linear or affine subspace is quite simple, conceptually speaking. But an efficient computer implementation requires some precaution.

Let, as before,

$$F = \{Y \in \mathbb{R}^N;\ \Lambda Y = D\}, \tag{2.2}$$

where $\Lambda : n \times N$, $Y : N \times 1$, and $D : n \times 1$ are matrices with real entries.

We write Λ in the form

$$\Lambda = RQ, \tag{2.3}$$

where Q is a square matrix, $N \times N$ and orthogonal, and R is of the form (R_1, O), where R_1 is a lower-triangular, invertible, square $n \times n$ matrix and O is an $n \times (N-n)$ matrix entirely made of zeros. This decomposition follows from the classical QR decomposition, taking transpose.

So we get

$$RQY = D \tag{2.4}$$

and, left-multiplying by R_1^{-1},

$$R'QY = D' \tag{2.5}$$

with $R' = (I, O)$, I is the $n \times n$ identity and $D' = R_1^{-1}D$.

The orthogonal projection of a point Y in $\mathbb{R}^N$ upon F is the point Y' defined by $Y' \in F$ and $Y - Y'$ is orthogonal to a basis of F.

There are n equations of the first type and $N-n$ of the second type.

Let

$$D' = \begin{pmatrix} d_1' \\ \vdots \\ d_n' \end{pmatrix}. \tag{2.6}$$

We construct a basis for F, set $f_j = Q^{-1}(d_1'e_1 + \cdots + d_n'e_n + e_j)$, $j = n+1, \ldots, N$.

These vectors are clearly independent. Moreover,

$$R'Qf_j = R'(d_1'e_1 + \cdots + d_n'e_n + e_j) = d_1'e_1 + \cdots + d_n'e_n = D', \tag{2.7}$$

which shows that they are a basis for F.

The conditions $Y - Y'$ orthogonal to f_j can be written as

$$\langle Y - Y', Q^{-1}(d_1'e_1 + \cdots + d_n'e_n + e_j)\rangle = 0, \quad j = n+1, \ldots, N, \tag{2.8}$$

which means that

$$\langle Q(Y - Y'), d_1'e_1 + \cdots + d_n'e_n + e_j\rangle = 0. \tag{2.9}$$

Set $Z = QY$, $Z' = QY'$. We get

$$\sum_{i=1}^{n} d_i'(z_i - z_i') + z_j - z_j' = 0, \quad j = n+1, \ldots, N. \tag{2.10}$$

The condition $Y' \in F$ can be written as

$$R'QY' = D', \qquad R'Z' = D', \tag{2.11}$$

and thus

$$z'_1 = d'_1, \ldots, z'_n = d'_n. \tag{2.12}$$

Putting back into (2.10), we get

$$\begin{aligned} z'_j &= d'_j, \quad j = 1, \ldots, n, \\ z'_j &= z_j + \sum_{i=1}^{n} d'_i (z_i - d'_i), \quad j = n+1, \ldots, N. \end{aligned} \tag{2.13}$$

This gives explicitly the projection. Then we find Y' taking ${}^tQZ'$, the matrix Q is orthogonal and so its inverse is equal to its transpose.

Remark 2.1. If F is a hyperplane, with the equation $a_1x_1 + \cdots + a_Nx_N = b$, the projection Y' of Y can be computed simply. Indeed, $Y' \in F$ and $Y' - Y$ is colinear with the vector $(a_1, \ldots, a_N)$. So we have

$$y_j - y'_j = \lambda a_j, \qquad \sum_{j=1}^{N} a_j y'_j = b. \tag{2.14}$$

Thus

$$\lambda = \frac{b - \sum_{j=1}^{N} a_j y_j}{\sum a_i^2}. \tag{2.15}$$

We may use this approach when F is no longer a hyperplane, but any affine subspace.

Indeed, any affine subspace is an intersection of hyperplanes (if the subspace has codimension p, it is the intersection of p hyperplanes). So we can project onto the hyperplanes, one after the other. But, in this way, the point we obtain is not exactly inside the intersection; it gets closer to it when we iterate the algorithm.

Remark 2.2. It will be convenient to choose a basis for which $Q = I$, if we want to work on F. But one cannot do this for both F and G at the same time.

2.2. Projection onto the manifold G. First, we define an affine subspace

$$H = \{Y \in \mathbb{R}^N;\ \exists X \in \mathbb{R}^n \text{ s.t. } Y = AX + B\}. \tag{2.16}$$

Then $G = f(H)$.

It will be convenient to put H in the previous form, so we can argue as we did for F:

$$H = \{Y \in \mathbb{R}^N;\ UY = V\}, \tag{2.17}$$

where U is a given $n \times N$ matrix and V is a given $n \times 1$ matrix (i.e., a column vector).

This is done in the following way: $Y = AX + B$ is equivalent to $Y - B = AX$, that is,

$$Y - B \in \mathrm{Im}\, A = \text{orthogonal}\left(\mathrm{Ker}^t A\right) \tag{2.18}$$

(see, e.g., Beauzamy [2]), and so this happens if and only if $P_o(Y - B) = 0$, where P_o is the orthogonal projection onto $\mathrm{Ker}^t A$. This gives the expression (2.17).

In order to project onto G, we set $Y = f(Z)$, $Z \in \mathbb{R}^N$, and we look for Y' in the form $Y' = f(Z')$, $Z' \in \mathbb{R}^N$.

Then $f(Z') \in G$ if and only if $Z' \in H$, since f is invertible.

The condition $\|Y - Y'\|$ minimum is equivalent to $\|f(Z) - f(Z')\|$ minimum.

We take as a distance

$$\mathrm{dist}\,(Z_1, Z_2) = \|f(Z_1) - f(Z_2)\|. \tag{2.19}$$

This is not the usual Euclidean distance, but the set intersections will be the same.

In order to project the point Y onto the manifold G, we project the point $Z = g(Y)$ onto H for the usual metrics (recall that g is the inverse function to f). So we find the projection Z' and we take $Y' = f(Z')$.

So the algorithm is finally described as the iteration of the procedure

$$\Pi = P_F \circ f \circ P_H \circ g, \tag{2.20}$$

where Π is a nonlinear operator. Quite clearly, any fixed point of Π is a solution to our problem, since it is in the intersection of F and G. Indeed, let x be a fixed point of Π. Then clearly $x \in F$ and $f \circ P_H \circ g(x) = x$, which implies $P_H \circ g(x) = g(x)$, which shows that $g(x) \in H$. But then x belongs to $g^{-1}(H) = f(H) = G$, which proves our claim.

We observe that the nonlinear operator Π is not a contraction in general, that is, it does not satisfy the property

$$\|\Pi x - \Pi y\| \leq \|x - y\| \tag{2.21}$$

for all x and y. Indeed, this property would imply, for all points x and y in F,

$$\|f \circ P_H \circ g(x) - f \circ P_H \circ g(y)\| \leq \|x - y\|, \tag{2.22}$$

and if $\xi = g(x)$, $\eta = g(y)$, (2.22) becomes, using the distance (2.19),

$$\mathrm{dist}\,(P_H(\xi), P_H(\eta)) \leq \mathrm{dist}(\xi, \eta), \tag{2.23}$$

and this property is not true in general. The orthogonal projection onto a subspace is not a contraction for a distance which is not the Euclidean distance (see Beauzamy [1] and Beauzamy and Maurey [3]). Since the orthogonal projection is not a contraction, a fortiori it is not a strict contraction, that is, it does not satisfy

$$\mathrm{dist}\,(P_H(\xi), P_H(\eta)) \leq c\,\mathrm{dist}(\xi, \eta) \tag{2.24}$$

for some $c < 1$, so the usual theorems about fixed points for strict contractions do not apply here (see, e.g., Goebel and Kirk [7]).

So we leave it as an open question to prove the convergence of the algorithm as it is defined here precisely. In practice, other choices can be taken for the projection onto G (for instance, to move in some direction if necessary) and one can easily construct algorithms for which the sequence of iterates x_n converges.

Indeed, let P_G be the projection of closest approximation onto the manifold G, that is,

$$P_G(x) = \left\{ z \in G;\ \|z - x\| = \inf_{y \in G} \|y - x\| \right\}. \tag{2.25}$$

We consider the iteration of the projection onto G and then onto F, that is, $P_F \circ P_G$. Then, starting from any point x, the distances decrease strictly:

$$\begin{aligned} \|P_F \circ P_G(x) - P_G(x)\| &< \|P_G(x) - x\|, \\ \|P_G \circ P_F \circ P_G(x) - P_F \circ P_G(x)\| &< \|P_F \circ P_G(x) - P_G(x)\|, \end{aligned} \tag{2.26}$$

and so on. Since the intersection of F and G exists, the limit of the decreasing sequence will be zero.

The problem with this algorithm is that the practical implementation of the projection P_G onto a manifold is quite hard to implement and the computation will be lengthy. This is why we chose to project onto a subspace.

As it stands here, the algorithm we describe here will be fast. The matrix Λ being given, the decompositions (2.3) and (2.5) are computed only once. Then, for any point Z, the projection Z' onto F, given by formulas (2.13), is obtained as a sum involving only one multiplication. Such an operation is extremely fast and problems with $N \approx 10^5$ can be handled in milliseconds. Moreover, standard routines exist for such tasks.

2.3. Taking constraints into account. In practice, constraints are added to problem (1.1) for practical reasons: the flow is limited, the water is compelled by a valve to run in one direction only, or the pressure is limited, and so on. These constraints appear as follows:

(i) constraints upon the elements of X, such as $x_i \geq 0$ or $x_i \leq C$;
(ii) constraints upon the elements of $Q = AX + B$, such as $q_j \geq 0$ or $q_j \leq C$;
(iii) the elements of the matrix Λ may contain variable coefficients, in the formula $\Lambda Y = D$, some of the $\lambda_{i,j}$ depend on Y (in practice, they may only take two values, depending only on one variable).

The first two types of constraints are easily handled by our method. Instead of an affine subspace, one only has a "piece" of a subspace, that is, the intersection with half-spaces. We see how one handles the third type, which is more difficult.

Some lines of the matrix are still constant. We treat them as before, altogether (each line gives a hyperplane).

Assume we have only one variable coefficient. It may take two values, λ_1 and λ_1'; λ_1 if $y_1 > 0$ and λ_1' if $y_1 < 0$.

So we have two possible hyperplanes,

$$\lambda_1 x_1 + \cdots + \lambda_N x_N = c_1, \tag{2.27}$$

and the same with λ_1'.

We project the point Y upon the first one, we get Y', and Y'' when we project upon the second one.

Several cases may occur:

(i) $y_1' > 0$ and $y_1'' > 0$: only the first one is acceptable;
(ii) $y_1' > 0$ and $y_1'' < 0$: both are acceptable and we choose the closest;
(iii) $y_1' < 0$ and $y_1'' > 0$: none is acceptable and we project upon $x_1 = 0$;
(iv) $y_1' < 0$ and $y_1'' < 0$: only the second is acceptable.

So, in fact, we are working on half-hyperplanes. The principle will be the same if two coefficients are variable.

One might make a continuous (or even differentiable) junction between both pieces and project onto this differentiable manifold, but this would be much longer to obtain. In our case, we may even predict which piece of the hyperplane will be used.

3. Modeling the physical problem

In this section, we explain how (1.1) is obtained from the physical problem; further reading may be found in [4, 9, 11]. We have a water distribution problem, with sources, consumers, and pipes in between, with many interconnections. A *node* is a place where a consumer or a source stands. There are S sources and N consumers (nodes which are not sources).

We define the following:

(i) $t_{i,j}$ is the physical pipe from node i to node j;
(ii) $q_{i,j}$ the flow from node i to node j; some orientation is chosen and this flow is positive or negative;
(iii) $\lambda_{i,j}$ a physical coefficient on the pipe $t_{i,j}$;
(iv) c_i the water consumption at node i;
(v) C the total water consumption of the water distribution system $C = \sum_{i=1}^{N} c_i$;
(vi) H_i a physical coefficient associated with the node i;
(vii) T is the number of pipes in the system; we count the pipes which go from a source to a node.

We set $n = M + S - 1$.

We have the following equations.

(i) At each node i,

$$\sum_j q_{j,i} = c_i. \tag{3.1}$$

This equation says that the (algebraic) sum of all quantities of water arriving at a node is equal to the consumption at this node.

(ii) For each pipe $t_{i,j}$,

$$g_{i,j}(q_{i,j}) = H_i - H_j \tag{3.2}$$

with $g_{i,j}(x) = \lambda_{i,j} x|x|$.

(iii) For the sources, the coefficients H_s are known:

$$H_s = \pi_s. \tag{3.3}$$

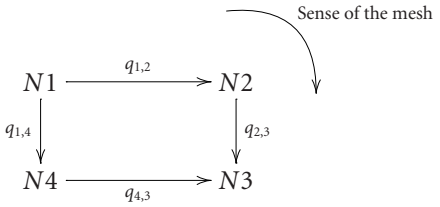
We want to compute all the flows $q_{i,j}$ in all pipes, the flow coming out of each source, and the coefficients H_i at each node. So, altogether, we have $T + S + N$ unknowns.

From (3.2), we deduce that for any closed circuit m (a closed circuit is called a *mesh* in the technical vocabulary),

$$\sum_{(i,j)\in m} \lambda_{i,j} q_{i,j} \left| q_{i,j} \right| = 0. \tag{3.4}$$

The sum is computed after a sense is defined on the mesh.

Example 3.1. The net



(3.5)

leads to the equation

$$\lambda_{1,2} q_{1,2} \left| q_{1,2} \right| + \lambda_{2,3} q_{2,3} \left| q_{2,3} \right| + \lambda_{3,4} q_{3,4} \left| q_{3,4} \right| + \lambda_{4,1} q_{4,1} \left| q_{4,1} \right| = 0 \tag{3.6}$$

or rather

$$\lambda_{1,2} q_{1,2} \left| q_{1,2} \right| + \lambda_{2,3} q_{2,3} \left| q_{2,3} \right| + \lambda_{4,3} q_{4,3} \left| q_{4,3} \right| - \lambda_{1,4} q_{1,4} \left| q_{1,4} \right| = 0. \tag{3.7}$$

Let M be the number of meshes in the system.

In order to write a general equation describing the net, we define a matrix W, $T \times M$, as follows:

(i) if the pipe t does not belong to the mesh m, then $W_{t,m} = 0$;
(ii) if the pipe t belongs to the mesh m, then $W_{t,m} = \pm 1$, depending on whether they have the same orientation or not.

We also introduce a matrix U, $T \times S$ matrix, whose term $U_{t,s}$ is obtained as follows:

(i) each mesh of the water system is "broken," withdrawing one pipe in each mesh;
(ii) all sources are turned off, except the source s;
(iii) the flows in all pipes are computed starting from the extremities of the tree, and moving upwards, one adds the consummations to be satisfied.

Then the coefficient $U_{t,s}$ is the flow in the pipe t, with only the source s active after normalization, that is, after division by the total consumption C.

Let

$$G = \begin{pmatrix} s_1 \\ \vdots \\ s_s \end{pmatrix} \tag{3.8}$$

be the column vector of the contributions of the sources (which are unknown) and let

$$V = \begin{pmatrix} V_1 \\ \vdots \\ V_M \end{pmatrix} \tag{3.9}$$

be the column vector of the additional flows inside the meshes, when one puts back the pipes which were withdrawn (see above). These flows are of course unknown.

Then the basic equation is

$$Q = U \times G + W \times V. \tag{3.10}$$

We can rephrase our problem as follows.

Find the q_s and the V_m (a total of $S+M$ unknowns) such that

$$\begin{gathered} \sum_s q_s = C, \\ \sum_{\substack{\text{path} \\ S_1 \to S_i}} \lambda_{i,j} q_{i,j} \left| q_{i,j} \right| = H(1) - H(i), \\ \sum_{\text{mesh } m} \lambda_{i,j} q_{i,j} \left| q_{i,j} \right| = 0, \end{gathered} \tag{3.11}$$

a total of $1 + (S-1) + M$ equations.

Set $n = M + S - 1$, n is the number of variables after simplifications.

Let $f : \mathbb{R}^p \to \mathbb{R}^p$ be the function defined by

$$f(x_1, \ldots, x_p) = \begin{cases} x_1 \left| x_1 \right|, \\ \quad \vdots \\ x_p \left| x_p \right|. \end{cases} \tag{3.12}$$

The previous equations can be written as

$$\sum_{t=1}^{T} \mu_{k,t} f(q_1) = d_k \tag{3.13}$$

for $K = 1, \ldots, n$.

The index t replaces the (i, j) of the pipes $t_{i,j}$ after sequential numbering $\mu_{k,t} = 0$ or $\pm\lambda_t$.

We set

$$Q = (q_t)_{t=1,\dots,T}, \qquad X = \begin{pmatrix} s_1 \\ \vdots \\ s_{s-1} \\ V_1 \\ \vdots \\ V_M \end{pmatrix}. \tag{3.14}$$

So X is an n-dimensional vector.

Using equation (3.10), we find that Q can be written as

$$Q = AX + B, \tag{3.15}$$

where B is a T-dimensional column vector and A is a $T \times n$ matrix. Our formulation $\Lambda f(AX + B) = D$ in (1.1) comes out instantly, with

$$\Lambda = (\mu_{k,t})_{k=1,\dots,n,\ t=1,\dots,T}, \qquad D = (d_k)_{k=1,\dots,n}. \tag{3.16}$$

References

[1] B. Beauzamy, *Projections contractantes dans les espaces de Banach*, Bull. Sci. Math. (2) **102** (1978), no. 1, 43–47 (French).

[2] ———, *Introduction to Operator Theory and Invariant Subspaces*, North-Holland Mathematical Library, vol. 42, North-Holland Publishing, Amsterdam, 1988.

[3] B. Beauzamy and B. Maurey, *Points minimaux et ensembles optimaux dans les espaces de Banach*, J. Funct. Anal. **24** (1977), no. 2, 107–139 (French).

[4] J. Bonnin, *Aide-Mémoire d'Hydraulique Urbaine*, vol. 42, Eyrolles, Paris, 1982.

[5] J.-C. Culioli, *Introduction à l'Optimisation*, Ellipses, Paris, 1994.

[6] R. Fletcher, *Practical Methods of Optimization. Vol. 1*, John Wiley & Sons, Chichester, 1980.

[7] K. Goebel and W. A. Kirk, *Topics in Metric Fixed Point Theory*, Cambridge Studies in Advanced Mathematics, vol. 28, Cambridge University Press, Cambridge, 1990.

[8] G. H. Golub and G. A. Meurant, *Résolution Numérique des Grands Systèmes Linéaires* [*Numerical Solution of Large Linear Systems*], Collection de la Direction des Études et Recherches d'Électricité de France, vol. 49, Eyrolles, Paris, 1983.

[9] T. V. Hromadka, *Computer Methods in Urban Hydrology: Rational Methods and Unit Hydrograph Methods*, Lighthouse Publications, California, 1983.

[10] L. S. Lasdon, *Optimization Theory for Large Systems*, The Macmillian, New York, 1970.

[11] T. R. Lazaro, *Urban Hydrology: a Multidisciplinary Perspective*, revised ed., Technomic Publishing, Pennsylvania, 1990.

Bernard Beauzamy: Société de Calcul Mathématique, SA, 111 Faubourg Saint Honoré, 75008 Paris, France

E-mail address: bernard.beauzamy@wanadoo.fr