INTERNATIONAL ATOMIC ENERGY AGENCY
UNITED NATIONS EDUCATIONAL, SCIENTIFIC AND CULTURAL ORGANIZATION

INTERNATIONAL CENTRE FOR THEORETICAL PHYSICS
MIRAMARE - P.O.B. 586 - 34100 TRIESTE (ITALY) - TELEPHONES: 224281/2/3/4/5/6 - CABLE: CENTRATOM

SMR/23/54

AUTUMN COURSE ON APPLICATIONS OF ANALYSIS
TO MECHANICS

22 September - 26 November 1976

IDENTIFICATION OF SYSTEMS:
Application to oil field problems

G. CHAVENT
I R I A
Le Chesnay
France

These are preliminary lecture notes intended for participants only.
Copies are available outside the Publications Office (T-floor) or from room 112.

(I) Identification as an optimization problem.

We first want to show, on a very simple example, how an identification problem can be set as an optimization problem.

Let $\Omega \subset \mathbb{R}^n$ be an isotropic body with heat conductibility $a(x)$ at point $x \in \Omega$. Suppose that on the part Γ_1 of the boundary, the body Ω is at a (given) zero temperature, and that on the complementary part Γ_2 of Ω, the body receives at each point s of Γ_2 a know heat flow $g(s)$. The equation giving the steady-state temperature $u(x)$ at every point $x \in \Omega$ is:

$$(P) \begin{cases} -\sum_{i=1}^{n} \frac{\partial}{\partial x_i}\left(a(x)\frac{\partial u}{\partial x_i}\right) = 0 & \text{in } \Omega \\ u(s) = 0 & \text{for } s \in \Gamma_1 \\ a(s)\frac{\partial u}{\partial n}(s) = g(s) & \text{for } s \in \Gamma_2 \end{cases}$$

Starting from (P), it is possible to define two types of optimization problems:

a) A Control Problem: One wants to find a heating law g such that the temperature repartition $u(x)$ be as close as possible to a given repartition $z(x)$, and this without spending too much energy. Define:

Error-Cost function $\quad J(g) = \int_{\Omega} \left(u(x;g) - z(x)\right)^2 dx + \nu \int_{\Gamma_2} g^2(s)\, ds$

Set of admissible controls : $L^2(\Gamma_2)$

so that the control problem may be set as

(C) $\boxed{\begin{array}{l}\text{Min } J(g) \\ g \in L^2(\Gamma_2)\end{array}}$

Remark 1 : As the mapping $g \longrightarrow u$ is linear, the functional $J(g)$ is quadratic, so that it can be proved, under suitable hypotheses, that the optimization problem (C) has a unique solution $\bar{g}$. ■

b) An Identification problem. The resolution of the control problem (C) requires a knowledge of all the parameters appearing in the state equations (P), that is of the heat conductibility function $a(x)$. As this function is not accessible to direct experimental measure, we shall try an indirect determination using some temperature measurements which are available. Suppose (for simplicity only) that the temperature is measured at every point $x \in \Omega$, which results in an observed temperature $z(x)$. Define

Error function $J(a) = \int_\Omega (u(x;a) - z(x))^2 dx$

Set of admissible parameters : We know a priori (strictly positive) lower and upper bounds α and M for the heat conductivity, so that we set

$$\mathcal{O}_{ad} = \{ a(x) / \quad 0 < \alpha \leq a(x) \leq M \quad \text{a.e. on } \Omega \}$$

and we may define the identification problem as:

(I) $\boxed{\begin{array}{l}\text{Min } J(a)\end{array}}$

Remark 2 : Even in the simple case of the linear state equation (P), the mapping $a \longrightarrow u$ is non linear, so that the functional J is generaly non quadratic, and it may happen that the optimization problem (I) do not have a solution ■

The aim of this course is to see how to use the formulation (I) of an identification problem to solve it actually, ie we shall go up to its numerical resolution. The tools we shall develop apply of course to the off-line resolution of control problems.

In the following paragraph, we shall give some notions about optimization algorithms which may be used for the resolution of (I).

(II) Two Unconstrained Optimization Algorithms

The numerical optimization algorithm is always applied to the discretized version (see § IV) of the continuous problem (I). So, from a practical point of view, its not a restriction to suppose that the parameter a is finite dimensional (instead of infinite dimensional as in problem (I)). This hypothesis will hold throughout all the § (II) and (III)

Let:

(2.1) J be an application from $\mathbb{R}^N$ into $\mathbb{R}$

We shall denote by $\nabla J(a)$ the gradient of J (ie the column vector of partial derivatives of J with respect to $a_1 a_2 \cdots a_N$) and by $< , >$ the scalar product in $\mathbb{R}^N$

We are concerned with the following unconstrained optimization problem:

(2.2) find $\tilde{a} \in \mathbb{R}^N$ such that $J(\tilde{a}) \leq J(a) \quad \forall a \in \mathbb{R}^N$

Thm 1: Under the hypothesis:

(2.3) J is continuous

(2.4) $\begin{cases} J(a) \text{ tends to } +\infty \text{ when } a \text{ tends to infinity} \\ \text{ie } \forall K>0, \exists L>0 \text{ s.t. } \|a\| \geq L \Rightarrow J(a) \geq M \end{cases}$

there exists at least a solution $\tilde{a}$ to the problem (2.2)

If moreover J is strictly convex, $\tilde{a}$ is unique.

Proof: choose any $\tilde{a} \in \mathbb{R}^N$ and define:

$$\mathcal{A} = \{ a \in \mathbb{R}^N / J(a) \leq J(\tilde{a}) \}$$

$\mathcal{A}$ is clearly a closed bounded, hence compact subset of $\mathbb{R}^N$ (see (2.3) and (2.4)), so J attains its minimum on $\mathcal{A}$, which is obviously its minimum on all $\mathbb{R}^N$. ∎

We want now to <u>design a method for actually computing $\hat{a}$</u>. Suppose that

(2.5) J is a differentiable mapping

Then noticing that, for every $a \in \mathbb{R}^N$:

$$\frac{d}{d\rho}\left[J(a - \rho \nabla J(a))\right]\Big|_{\rho=0} = -\|\nabla J(a)\|^2 \leq 0$$

we see on the figure 1 that, for sufficiently small ρ:

$$J(a - \rho \nabla J(a)) \leq J(a)$$

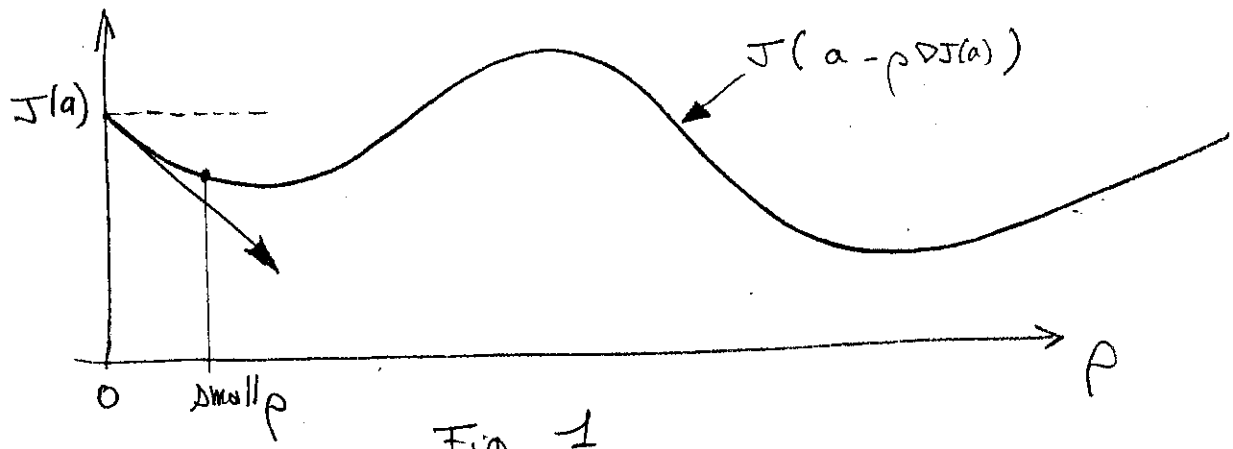


Fig 1

This leads to the

<u>Simple Gradient Algorithm</u>

- choose a <u>gradient step</u> $\rho > 0$
- choose an <u>initial guess</u> $a^0 \in \mathbb{R}^N$
- define a sequence $a^0, a^1, \ldots a^k, \ldots$ etc... by

(2.6) $$a^{k+1} = a^k - \rho \nabla J(a^k)$$

Does a^k converge toward $\hat{a}$? An answer is given

Thm 2 Under the hypothesis (2.4) (2.5) and:

(2.7) $\begin{cases} \nabla J \text{ is a Lipschitz application} \\ \text{ie } \exists M>0 \text{ s.t. } \forall a,b \in \mathbb{R}^N \quad \| \nabla J(a) - \nabla J(b) \| \leq M \| a-b \| \end{cases}$

Then:

i) the exists at least a solution $\hat{a}$ to the problem (2.2), such that:

(2.8) $$\nabla J(\hat{a}) = 0$$

ii) $\exists \bar{\rho}>0$ s.t $\forall \rho \in]0, \bar{\rho}[$, $\forall a^0 \in \mathbb{R}^N$, the sequence $\{a^k\}$ defined by (2.6) contains a subsequence a^N s.t:

(2.9) $$a^N \longrightarrow a^* \quad \text{with} \quad \nabla J(a^*) = 0$$

iii) If moreover J is <u>strictly convex</u>, <u>all the sequence a^k converges toward the unique solution $\bar{a}$</u> of problem (2.2).

Proof: i) is a simple corollary of Thm 1

ii) Let us first show that one may choose ρ so that the sequence $\{J(a^k)\}$ is decreasing. Define R_0 by:

(2.10) $$J(a^{k+1}) - J(a^k) = \langle \nabla J(a^k), a^{k+1} - a^k \rangle + R_0$$

Applying the finite increment theorem to the mapping $a \to J(a) - \nabla J(a^k).a$ one gets:

$$|R_0| \leq \| a^{k+1} - a^k \| \sup_{a \in [a^{k+1}, a^k]} \| \nabla J(a) - \nabla J(a^k) \|$$

Using (2.7) one gets:

(2.11) $$|R_0| \leq M \| a^{k+1} - a^k \|^2$$

Using (2.6) (2.10) (2.11) we have:

(2.12) $$J(a^{k+1}) - J(a^k) \leq -\rho(1 - \rho M) \| \nabla J(a^k) \|^2$$

Define:

(2.13) $$\bar{\rho} = \frac{1}{M}$$

Then for every $\rho \in]0, \bar{\rho}[$ the sequence $\{J(a^k)\}$ is

decreasing (cf (2.12)) and bounded below (from part i) of Thm), and hence converging, so that $J(a^k) - J(a^{k+1}) \to 0$. As

$$(2.14) \qquad 0 \leq \|\nabla J(a^k)\|^2 \leq \frac{1}{\rho(1-\rho M)} \left(J(a^k) - J(a^{k+1}) \right)$$

we see that

$$(2.15) \qquad \nabla J(a^k) \longrightarrow 0 \qquad \text{as } k \to +\infty.$$

As the sequence $\{J(a^k)\}$ is converging, it is bounded above; From (2.4) we deduce then that that the sequence $\{a^k\}$ is bounded, and that there exists a subsequence a^ν converging to some a^*. As ∇J is Lipschitz, it's continuous, hence:

$$(2.16) \qquad \nabla J(a^\nu) \longrightarrow \nabla J(a^*)$$

Comparing (2.15) and (2.16) we see that $\nabla J(a^*) = 0$, which ends the proof of ii)

iii) If J is strictly convex, $\hat{a}$ is the only solution of the equation $\nabla J(a) = 0$, so that from every subsequence $\{a^\nu\}$ of $\{a^k\}$, we can extract a subsequence $\{a^\nu\}$ converging toward $\hat{a}$; this prooves that all the sequence a^k converges to $\hat{a}$. ∎

Remark 3: For $N = 2$ and for a function J whose level lines are ellipses, the simple gradient algorithm propagates as shown in fig 2:

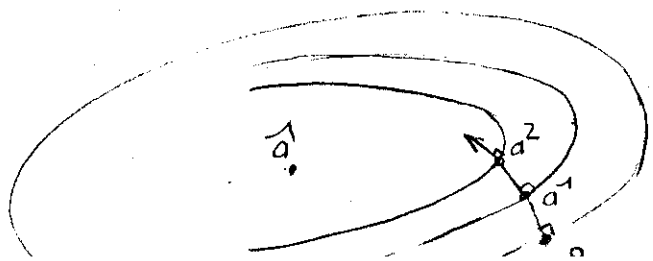


Fig 2

The practical choice of ρ with the simple gradient algorithm is somewhat difficult, as you don't know $\bar{\rho}$:

- <u>if ρ is choosen very small</u>, your algorithm shall be stable ($\rho \in]0, \bar{\rho}[$ very probably!), but the speed of convergence is <u>slow</u> (the amount of change of the parameters at each gradient iteration is small)

- <u>if ρ is choosen large</u>, the speed of convergence shall increase, but the <u>risk of divergence</u> of the algorithm ($\rho \gg \bar{\rho}$) increases simultaneously.

That is why gradient algorithm have been designed which use a different gradient step ρ^k at each gradient iteration. A popular among those algorithm is the

<u>Steepest Descent Algorithm</u>

- choose an initial guess $a^0 \in \mathbb{R}^N$
- define a sequence $a^0, a^1, \dots a^k, \dots$ by:

$$a^{k+1} = a^k - \rho^k \nabla J(a^k) \tag{2.17}$$

where ρ^k is defined by

$$J(a^k - \rho^k \nabla J(a^k)) \leq J(a^k - \rho \nabla J(a^k)) \quad \forall \rho > 0 \tag{2.18}$$

<u>Remark 4</u> : Graphical interpretation (compare with Remark 3)

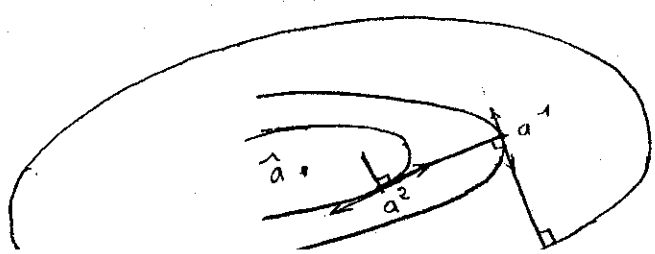


Fig 3

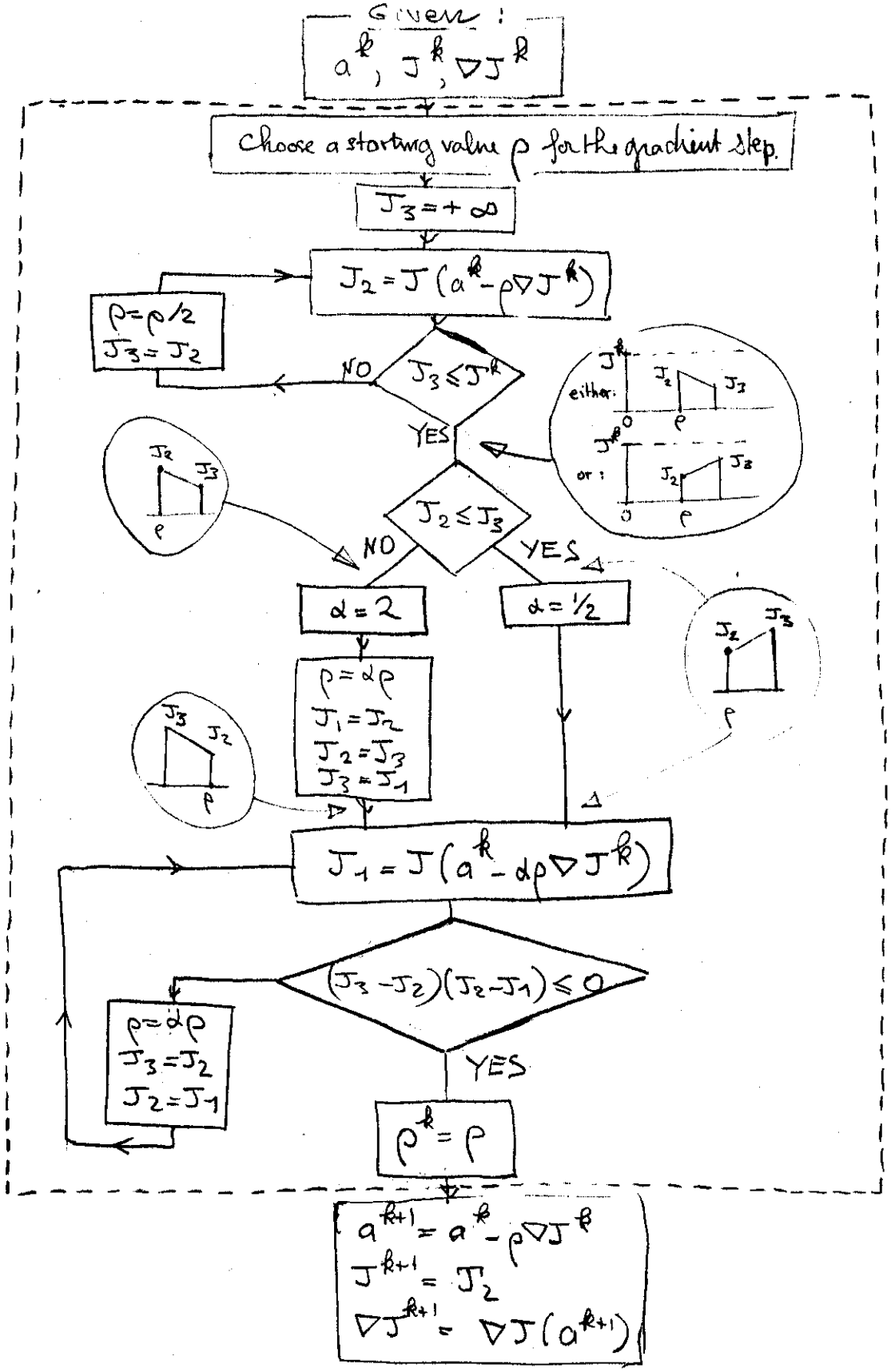
As for the simple gradient algorithm, it's possible to give a proof of convergence of the steepest descent algorithm under similar hypothesis.

From a numerical point of view, the difficulty is the determination of ρ^k at each gradient-iteration. Happily, a very precise determination of the ρ^k which minimizes J along the direction opposite to the gradient do not increase the performances of the steepest descent algorithm. So a very crude determination of ρ^k by dichotomy is enough. Moreover, the determination of the global minimum of J on the direction opposite to the gradient (as required in (2.18)) is not numerically possible; and we have to be content with a ρ^k which gives approximately a local minimum.

The block-diagram of fig 4 gives a way to compute such a ρ^k, knowing a^k, J^k and ∇J^k. It requires the choice of a starting value for ρ to initiate the dichotomy; a simple choice for this is to take $\rho = \rho^{k-1}$, so that ρ has to be chosen only for the first iteration.

The upper loop is designed to eventually reduce sufficiently the starting value of ρ so that the corresponding criterion J be at least smaller or equal to the value J^{in} of the criterion at the last iteration. As it is clear from fig 1, one should go out of this loop after a finite number of cycles. If not, this means that the computed vector ∇J^k is not the gradient of J, which may be due to

to approximations and/or rounding errors when a^k is very near to $\hat{a}$; we shall say in that case that the 1-D approximate minimization is "unsuccessful" (see fig)



(III) A constrained optimization algorithm

As we have seen in §(I), the identification problem as a generally constrained optimization problem, ie the parameter a has to stay in a given (generally convex) closed set $\mathcal{A}_{ad}$ of $\mathbb{R}^N$

We shall give a slight modification of the gradient algorithm of §(II) which makes it possible, in a simple but common case, to take in account those constraints.

Suppose that $a^k \in \mathcal{A}_{ad}$ (see fig 5). Then it may happen that the point $\tilde{a}^{k+1} = a^k - \rho \nabla J^k$ do not belong to $\mathcal{A}_{ad}$.

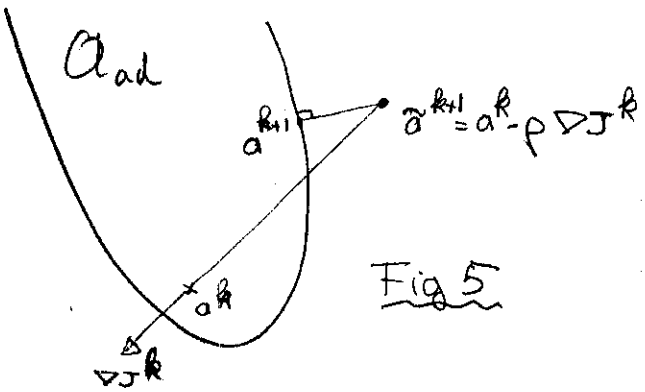


Fig 5

So, denoting by P the projection operator on $\mathcal{A}_{ad}$, we shall replace $\tilde{a}^{k+1}$ by its projection on $\mathcal{A}_{ad}$:

$$(3.1) \qquad a^{k+1} = P\left(a^k - \rho \nabla J^k\right)$$

With this simple modification, both algorithm of §(I) can be viewed as constrained optimization algorithm.

In practice, those constrained optimization algorithm are effective only if the computation of the projection on $\mathcal{A}_{ad}$ is numerically possible. As this computation consists in the minimization on $\mathcal{A}_{ad}$ of the distance between $\tilde{a}^{k+1}$ and a, it is generally of the same order of difficulty than the initial optimization problem.

However, in the very simple case of side constraints, ie

$$(3.2) \qquad \mathcal{A}_{ad} = \left\{ a \in \mathbb{R}^N \;/\; \underline{a}_i \leq a_i \leq \overline{a}_i \quad \forall i = 1, 2, \dots N \right\}$$

the projection operator is straightforward:

$$(3.3) \qquad (Pa)_i = \begin{cases} \underline{a}_i & \text{if} \quad a_i \leq \underline{a}_i \\ a_i & \text{if} \quad \underline{a}_i \leq a_i \leq \overline{a}_i \end{cases}$$

This type of constraints corresponds to the data of lower and upper bounds on the unknown coefficients, which is a very common case; the steepest descent algorithm with projection works for all of those case; its block-diagram is given in the fig 6. The block "1-D approximate minimization of..." can be replaced by the dashed block of fig 4, provided that in this latter $a^k - \rho \nabla J^k$ be replaced everywhere by $P(a^k - \rho \nabla J^k)$

Remark 5: Change of unknown parameters. Suppose that the parameter a is an affine function of p parameters $b_1 \dots b_p$ (generally $p \leq N$, but $p \geq N$ works too):

$$a = \Theta b + \tilde{a} \qquad (3.4)$$

$\Theta = N \times p$ matrix, $\tilde{a} \in \mathbb{R}^N$

and that the optimization algorithm is written for the minimization of J with respect to a (as in fig 6)

If now we want to minimize J with respect to b, the only constraints on b being the side constraints on a which translate to side-constraints on b, we have to compute:

$$b^{k+1} = b^k - \rho^k \frac{dJ}{db}(b^k) \qquad (3.5)$$

which, by multiplication by the matrix Θ and adding $\tilde{a}$ gives:

$$a^{k+1} = a^k - \rho^k \, \Theta \frac{dJ}{db}(b^k) \qquad (3.6)$$

As:

$$\frac{dJ}{db}(b^k) = {}^t\Theta \, \nabla J(a^k) \qquad (3.7)$$

we see that the only modifications to do in the program of fig 6 are:

i) relax the side constraints on a which are not side constraints on b by taking, for the corresponding indices i, $\underline{a}_i = -\infty$ and $\overline{a}_i = +\infty$

ii) Insert at misection point 1 the following block:

$$\frac{dJ}{db}(b^k) = \Theta \nabla J(a^k)$$
$$\nabla J(a^k) = \Theta \frac{dJ}{db}(b^k)$$

(iii) Insert at insertion point 2 the block

$$b^{k+1} = b^k - \rho^k \frac{dJ}{db}(b^k)$$

This remark has two practical consequences:

- When writing an optimization program, take as unknown all the parameters with respect to which an optimization may be performed. The selection of the parameters with respect to which the optimization is effectively performed is then easily done with an adequate matrix Θ: if for instance only the p first parameters $a_1 \cdots a_p$ are unknown ($p \leq N$) and the $N-p$ parameters $a_{p+1} \cdots a_N$ are known, we take

$$\Theta = \begin{bmatrix} 1 & & 0 \\ & \ddots & \\ 0 & & 1 \\ 0 & & 0 \\ \vdots & & \vdots \\ 0 & \cdots & 0 \end{bmatrix} \begin{matrix} \} p \\ \} N-p \end{matrix} \qquad \tilde{a} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ a_{p+1} \\ \vdots \\ a_N \end{bmatrix}$$

(Θ: p columns)

so that the modifications mentioned in the points (i) (ii) of remark 5 resume in the insertion at point 1 of:

$$\frac{\partial J(a^k)}{\partial a_i} = 0 \qquad i = p+1, \cdots N$$

- When the unknown parameter is a function, take as numerical unknown vector $a = (a_1 \cdots a_N)$ when writing the identification program some discrete approximation of the function. It will make easier any change in the choice of a closed-form formula for representing that

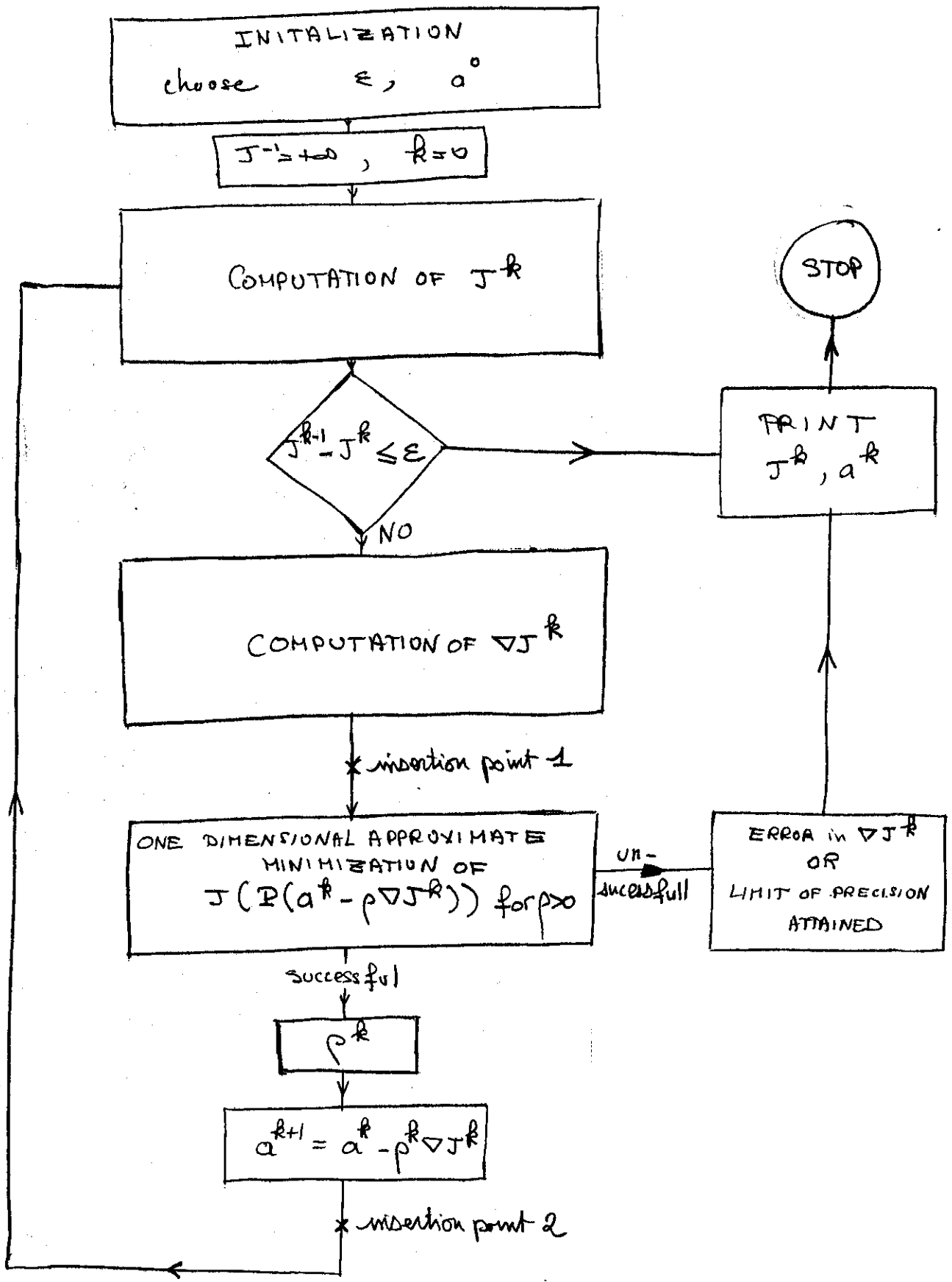


Fig 6 The Steepest Descent Algorithm with projection

In the following paragraphs, we shall study the blocks "Computation of J^k" and "computation of ΔJ^k" when J is a function defined through the resolution of a state equation, as in the problem (I) of § (I).