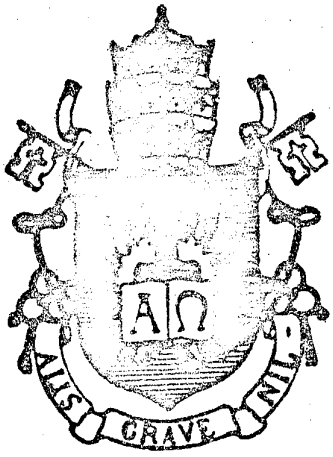

PUC

Série: Monografias em Ciência da Computação,
No. 4/91

A MULTILEVEL CONJUGATE GRADIENT METHOD

Maurício Kischinhevsky

Departamento de Informática

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO
RUA MARQUÊS DE SÃO VICENTE, 225 - CEP-22453
RIO DE JANEIRO - BRASIL

PUC RIO - DEPARTAMENTO DE INFORMÁTICA

Série: Monografias em Ciência da Computação, No. 4/91
Editor: Carlos J. P. Lucena Abril, 1991

A MULTILEVEL CONJUGATE GRADIENT METHOD *

Maurício Kischinhevsky

* Trabalho patrocinado pela Secretaria de Ciência e Tecnologia, da Presidência da República.

In charge of publications:

Rosane Teles Lins Castilho
Assessoria de Biblioteca, Documentação e Informação
PUC Rio - Departamento de Informática
Rua Marquês de São Vicente, 225 - Gávea
22453 - Rio de Janeiro, RJ
Brasil

Tel.:(021)529-9386 Telex:31078 Fax:(021)511-5645
E-mail:rosane@inf.puc-rio.br

Abstract:

This text discusses attempts to put together convenient properties of Conjugate Gradient and Multigrid Methods for the numerical solution of liner systems arising from the discretization of certain partial differential equations. Some trials which one finds in the literature are reviwed, and a new proposal is made that intends to form a competitive alternative to the conventional solvers.

Keywords:

Multilevel conjugate gradient, multigrid method, conjugate gradient method, numerical solution of partial differential equations.

Resumo:

Neste texto discutem-se tentativas de utilização de características de métodos de gradiente conjugados e de redes múltiplas, dando origem a novos métodos para a solução de sistemas de equações lineares provenientes da discretização de certas equações diferenciais parciais. Tentativas presentes na literatura são descritas e uma alternativa é proposta com o intuito de gerar um método competitivo com os resolutores convencionais.

Palavras-chave:

Gradientes conjugados multinível, método de redes múltiplas, gradiente conjugado, resolução numérica de equações diferenciais parciais.

1 Introduction

The interest in efficiently solving large scale sparse systems has led to many strategies, from iterative methods such as Gauss-Jacobi, Gauss-Seidel, SOR, Steepest Descent, Multigrid and Conjugate Gradient to direct ones like Gaussian Elimination, Crout-Banachiewicz LU decomposition or Conjugate Gradient (see,e.g.,[17][1][16][3][11]).

The main problem is to solve

$$Hx = b\,, \quad x, b \in \mathbf{R}^N \;, \quad H \in \mathbf{R}^{N \times N} \tag{1}$$

in a convenient way for the specific application.

Conjugate Gradient (CG) methods are usually considered iterative because successive estimates of the solution vector are generated, with non-decreasing number of significant digits(see convergence aspects below) at each step. However, CG methods can be considered direct in the sense that it spans the whole vector space $\mathbf{R}^N$ in at most N steps, where N is the number of distinct eigenvalues of the system's matrix. The way it is known to be an efficient tool is as an iterative method for large sparse sets of linear equations, rather than as a direct method for full systems.

Multigrid (MG) methods form iterative procedures which are becoming very popular and exhibit asymptotic convergence rates very convenient for large systems, although with a significant cost per iteration.

Some of the conventional methods can be combined in order to provide fast solving of the linear system. The main tools in choosing which one to use in each situation are operations count and convergence properties of the iteration matrices ([10][15][13]). As a consequence one uses smoothers as SOR together with standard Multigrid methods([15]), combined CG-MG methods([4][19][8]), and preconditioning of various types(e.g. LU, SOR, Diagonal) in Preconditioned CG methods.

In the following sections one starts by overviewing some Conjugate Gradient and Multigrid concepts that are useful for the specification of Multilevel Conjugate Gradient methods, and proceeds describing the first attempts to put together such ideas. In the continuation a proposal of a Multi-Level Conjugate Gradient method is described and some properties are discussed. Some preliminary perspectives of such method come next, together with the concluding remarks.

2 A Brief Review on CG and MG concepts

2.1 Some aspects on CG

The Conjugate Gradient Method[17] was proposed as a means to solve a minimization problem for quadratic functionals of the form [1]

$$f(x) = \frac{1}{2}x^T Hx - b^T x + c, \tag{2}$$

where H is positive or negative definite.

In the usual notation the gradient and the Hessian of $f(x)$ are easily found to be $g(x) = Hx - b$ and $H(x) = H$, respectively. Thus, such quadratic functionals have constant Hessian. Moreover, it is important to notice that they provide the simplest examples of functionals that possess a strong local minimizer (that is, a $\hat{x}$ such that $f(\hat{x}) < f(x)$, $\forall x \in \mathrm{R}^N$) or maximizer.

A point $\hat{x}$ is a stationary point of f if the gradient vanishes at $\hat{x}$, that is, if $H\hat{x} - b = 0$. If H is non-singular, $\hat{x}$ is uniquely determined by $\hat{x} = H^{-1}b$ and $f(x)$ may be rewritten as

$$f(x) = \frac{1}{2}(x - \hat{x})^T H(x - \hat{x}) + \hat{c}, \tag{3}$$

where $\hat{c} = -\frac{1}{2}b^T\hat{x} + c$.

If $\{ \lambda_i, v_i \}_{i=1}^N$ are the eigensolutions of H, and H is symmetric, the eigenvalues are real and can be ordered as $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_N$ and the eigenvectors supposed to satisfy $v_i^T v_j = \delta_{i,j}$, $i, j = 1, 2, \ldots, N$.

Define $\Lambda = diag(\lambda_1, \lambda_2, \ldots, \lambda_N)$ and $V = (v_1, v_2, , \ldots, v_N)$. V is an orthogonal matrix, i.e., $V^{-1} = V^T$ and $HV = \Lambda V$.

With the aid of the variable $z = V^T(x - \hat{x})$ it is obtained

$$\hat{f}(z) \equiv f(Vz + \hat{x}) = \frac{1}{2}z^T V^T HVz + \hat{c} = \frac{1}{2}z^T \Lambda z + \hat{c} \quad , \tag{4}$$

or

$$\hat{f}(z) = \frac{1}{2} \sum_{i=1}^{N} \lambda_i z_i^2, \quad z \in R^N \tag{5}$$

Since $\hat{f}(z) = f(x)$ under the transformation $z = V^T(x - \hat{x})$, one can focus on $\hat{f}$. If H is positive definite then all of its eigenvalues are positive and the range of $\hat{f}$ is $[\hat{c}, \infty)$. Clearly $z = 0$ is the strong

[1] For clarity one may ommit the symbol of transposition where no confusion is possible

global minimizer. In the case H is positive definite, for $k > \hat{c}$ the level surface L_k is the ellipsoid

$$\sum_{i=1}^{N} \lambda_i z_i^2 = \hat{k}, \tag{6}$$

where $\hat{k} = 2(k - \hat{c}) > 0$ as sketched in figure 1.

The distortion measure will be given by $K(H) \equiv \frac{\lambda_N}{\lambda_1}$, the spectral condition number of H(λ_1 and λ_N are the smallest and largest eigenvalues of H, respectively). Numerical methods for the minimization of a functional will behave badly if the level surfaces are highly distorted from spherical form, and as is shown in what follows, the spectral condition number plays a critical role.

Of course this analysis will also be valid for arbitrary functionals, provided that a convenient Taylor expansion is performed near a stationary point.

Now the Conjugate Gradient Method is described, showing that it produces the minimizer of f in at most N iterations. In the next section some aspects of convergence are described, and bounds on the number of iterations necessary to provide an atenuation of a constant factor in the error are derived.

In order to obtain a solution with the required accuracy, iterations of the type (d^k is called the search direction)

$$x^{k+1} = x^k + \tau_k d^k \quad k = 1, 2 \ldots \tag{7}$$

which is equivalent to (multiply equation (7) by H and subtract b)

$$g^{k+1} = g^k + \tau_k H d^k \,. \tag{8}$$

Thus,

$$d^k g^{k+1} = d^k g^k + \tau_k d^k H d^k = 0 \,, \tag{9}$$

if

$$\tau_k = -\frac{d^k g^k}{d^k H d^k} \,. \tag{10}$$

The choice made above is important since it causes the minimization of $f(x^k + \tau d^k), -\infty < \tau < \infty$. Moreover, it makes the gradient at x^{k+1} orthogonal to the search direction d^k (as sketched in figure 2)

What is done at this moment is to generate successive search directions via iterations of the form

$$d^{k+1} = -g^{k+1} + \beta_k d^k \quad , \tag{11}$$

where $d^0 = -g^0$ and $\beta_0, \beta_1, \ldots$ will be determined in order to minimize at each iteration the error $\|x - \hat{x}\|_H$ over a certain subset of R^N. Here $\|y\|_{H^{-1}} = (yH^{-1}y)^{\frac{1}{2}} = \sqrt{y^t H^{-1} y}$. The expression $\|y\|_H$ is called the **energy norm** of y and is based on the energy inner product yHy. From the relation $x - \hat{x} = H^{-1}g$ it is easy to obtain $\|x - \hat{x}\|_H = \|g\|_{H^{-1}}$.

That is, working with the gradients rather than directly with x, and noticing that from equations (8) and (11) successive iterations generate a subspace of R^N

$$S_k = SPAN\{Hg^0, H^2g^0, \ldots, H^kg^0\} \quad , \tag{12}$$

which has dimension equal to the number of linearly independent vectors in the set $\{Hg^0, \ldots, H^kg^0\}$.

The parameter β_k for the iterative process may be written as

$$\beta_k = \frac{g^{k+1}Hd^k}{d^kHd^k} \tag{13}$$

and will correspond to the minimization of the gradient in the H^{-1}-norm,

$$\|g^k\|_{H^{-1}} = \min_{h \in S_k} \|g\|_{H^{-1}} = \min_{h \in S_k} \|g^0 + h\|_{H^{-1}} \, . \tag{14}$$

Let $T_k = \{g \in R^N; g = g^0 + h, h \in S_k\}$. It is important to note that $g^k \in T_k$, since

$$g^k = g^0 + \sum_{l=1}^{k} \alpha_l^{(k)} H^l g^0 \quad . \tag{15}$$

With the above choice of β_k one obtains

$$g^k g^l = 0, \qquad l \neq k \tag{16}$$

and

$$d^k H d^l = 0, \qquad l \neq k \, , \tag{17}$$

that is, the gradients are orthogonal to each other and the search directions are orthogonal to each other in the energy inner product.

It is easy to see that equation (14) is equivalent to

$$\|g^0 + h^k\|_{H^{-1}} = \min_{h \in S_k} \|g^0 + h\|_{H^{-1}} \, , \tag{18}$$

where $h^k = g^k - g^0$. Now viewing any $h \in S_k$ as an approximation to the vector $-g^0$, then the corresponding error in the approximation is

$$h - (-g^0) = h + g^0 \,, \tag{19}$$

Thus, the problem becomes that of finding in the subspace S_k the vector h^k that most closely approximates $-g^0$, the error being measured in the H^{-1} norm. According to the projection theorem of Linear Algebra, h^k exists, is unique, and has the property of making the error $g^0 + h^k$ orthogonal to every h in S_k, that is

$$(g^0 + h^k)H^{-1}h = 0 \qquad \forall\, h \in S_k \,. \tag{20}$$

See figure 3 for geometrical interpretation.

Since the minimizing vector is $g^k \equiv g^0 + h$, for any $g \in T_{k-1}$ the vector $h = Hg$ belongs to S_k . Thus, g^k has the property that $g^k g = 0$, for all $g \in T_{k-1}$. So, the subspace of R^N that is spanned by the gradient is increased by one dimension at each iterative step, which is a desirable characteristic, and also guarantees the finite termination of the iterative process in the absence of rounding errors.

As some relations are available that establish that

$$\beta_k = \frac{g^{k+1}g^{k+1}}{g^k g^k} \,, \tag{21}$$

it is easy to obtain an algorithm that requires low computational effort and saves storage to a few vectors in case of compact representation[3] of the matrix H. The algorithm becomes

Procedure Conjugate Gradient(CG)
For $k = 0, 1, \ldots$ **and** x^0 **chosen, set** $g^0 = Hx^0 - b$, $d^0 = -g^0$ **and**
$\tau_k = \frac{g^k g^k}{d^k H d^k}$
$x^{k+1} = x^k + \tau_k d^k$
$g^{k+1} = g^k + \tau_k H d^k$
$\beta_k = \frac{g^{k+1}g^{k+1}}{g^k g^k}$
$d^{k+1} = -g^{k+1} + \beta_k d^k$
End_For
End of CG

Making use of Chebyshev polynomials and extending their validity to $x \in (-\infty, \infty)$ it can be shown ([3]-App.A) that, for any $\epsilon > 0$, if $p(\epsilon)$ is defined to be the smallest integer k such that

$$\|x^k - \hat{x}\|_H \leq \epsilon \|x^0 - \hat{x}\|_H \quad , \qquad \forall x^0 \in R^N \quad , \tag{22}$$

then

$$p(\epsilon) \leq \frac{1}{2}\sqrt{K(H)}ln(\frac{2}{\epsilon}) + 1 \tag{23}$$

where $K(H)$ is the condition number $\frac{\lambda_N}{\lambda_1}$ for matrix H.

2.2 Preconditioning

The process of preconditioning is a means of solving the minimization problem for the functional $\tilde{f}(y)$ related to $f(x)$, that is,

$$\tilde{f}(y) = \frac{1}{2}y\tilde{H}y - \tilde{b}y + \tilde{c}\,, \qquad y \in R^N, \tag{24}$$

where $\tilde{H} = E^{-1}HE^{-T}$, $\tilde{b} = E^{-1}b$ and $\tilde{c} = c$. The main requirement is that one has $K(\tilde{H}) < K(H)$ (preferably much less than) in order to increase the rate of convergence in comparison to the related Conjugate Gradient Method. In general, a good preconditioning matrix has the following properties:

(1) $K(\tilde{H})$ is significantly less than $K(H)$

(2) the factors of E can be determined quickly and do not require excessive storage in relation to H

(3) the system $Eh^k = g^k$ can be solved much more efficiently than $Hx = b$

It follows that one may have some different forms of preconditioners. The differences among them refer to the strategy of balancing computational effort for items (2) and (3) above and the reduction of condition number (cf. item(1)). The convergence analysis of the Preconditioned Conjugate Gradient Method (PCG) leads to the same results of that for CG, provided that one reads $\tilde{H}$ instead of H .

In [3], chapter7 it is shown that when a boundary value problem(BVP) with smooth coefficients is solved through the finite element method(FEM), the use of linear (or bilinear) basis functions makes $K(\tilde{H}) \approx h^{-1}$ for SSOR preconditioning (here h is the greatest element edge length in the mesh). In the other hand, for the CG method one obtains $K(H) \approx h^{-2}$. Thus for a model problem such as Poisson's equation in n-dimensional unitary cube and Dirichlet conditions, using $N \approx h^{-n}$ and $w(H) =$ half bandwidth of $H \approx N^{(1-\frac{1}{n})}$ table 1 can be generated.

Method	$n = 2$	$n = 3$
CG	$\mathcal{O}(N^{1.5})$	$\mathcal{O}(N^{1.33})$
PCG	$\mathcal{O}(N^{1.25})$	$\mathcal{O}(N^{1.17})$

Table 1 - Computational Cost of iterative methods in units of h^{-n},
that is, volume of the domain in which the BVP is being solved

Preconditioning can be a powerfull tool in the improvement of convergence rate for CG, provided that it lowers the global cost, even though it causes an increase in the cost for each iterative step.

2.3 Some aspects of MG

To motivate the use of multigrid concepts, consider a classical iterative method such as damped Jacobi iteration (alternatively Gauss-Seidel or SOR could be used) to solve the linear system of equations (1).

In the definition of an iterative process it is required that the true solution $\hat{x} = H^{-1}b$ appears as a fixed point of a general expression

$$x^{k+1} = Mx^k + Nb\,, \tag{25}$$

where M is called the iteration matrix. As the error $e^k = x^k - \hat{x}$ from one step to the next is given by

$$e^{k+1} = Me^k\,, \tag{26}$$

it immediately follows that the iteration matrix is the amplification matrix for the error, that is,

$$e^k = M^k e^0\,. \tag{27}$$

Consequently the iterative process is convergent for all initial guesses x^0 if and only if the spectral radius

$$\rho(M) \equiv \lim_{k \to \infty} \|M^k\|^{\frac{1}{k}} < 1. \tag{28}$$

Moreover the convergence will be exponential with asymptotic rate at least $\rho(M)$, i.e.,

$$\|x^k - \hat{x}\| \leq Kk^p\rho^k(M) \tag{29}$$

for some $K, p < \infty$ (K depends on x^0) [21].

As an example consider the damped Jacobi iteration to solve the Poisson equation in a discretized domain, that is,

$$-\Delta\phi_h = f_h\,, \qquad x \in \Omega_h = [0,1] \times [0,1] \tag{30}$$

one may generate the corresponding linear system, build the iteration matrix M_{DJ} and solve exactly for its eigenvalues[13], obtaining

$$\lambda_p = (1-\omega) + \frac{\omega}{2}(cosp_1 + cosp_2)\,, \tag{31}$$

where ω is the damping factor, $0 < \omega \leq 1$, and

$$p_1, p_2 = \frac{\pi}{L+1}, \frac{2\pi}{L+1}, \cdots, \frac{L\pi}{L+1}\,, \tag{32}$$

where $L = \frac{1}{h}$. The spectral radius $\rho(M_{DJ})$ is the absolute value of the eigenvalue of $\rho(M_{DJ})$ with largest magnitude. That is,

$$\begin{aligned} \rho(M_{DJ}) &= \lambda_{\frac{\pi}{L+1},\frac{\pi}{L+1}} \\ &= 1-\omega\{1-cos(\tfrac{\pi}{L+1})\} \\ &= 1-\mathcal{O}(L^{-2}) \end{aligned} \tag{33}$$

Thus, it is to be noted that the convergence will be slow, since $\mathcal{O}(L^2)$ iterations will be necessary for the damped Jacobi iteration to converge adequately. Such behavior is called **critical slowing down**, and is responsible for the increase in computational labor when L(number of grid points) is large.

Here it is convenient to highlight what has happened. The slow modes ($\lambda \approx 1$)are the long wavelenght ones ($p_1 p_2 \ll 1$). It is easy to see what occurs as a consequence of the algorithm's locality: in a single step one grid point can only inform its nearest neighbors about its instantaneous value. For the true solution to be reached (representing a "thermalization" of the dynamical system of interacting particles) this "information" executes a random walk[23] around the lattice in such a way that it must move to and from the boundaries until equilibrium is attained (this takes the time of order L^2 since the average distance in t time steps of a two dimensional random walk is $\sqrt{t}$)[13].

As the discussion above states the question of critical slowing down(for more details see [13]), it is time to mention one of the possible remedies which comes with case for those familiar with Renormalization Group concepts ([18],[24]). That is: do not deal with all lenght scales at once, but define instead a sequence of problems in which each lenght scale, beginning with the smallest and working towards the largest, can be handled separately. Such ideas were (even previously to the work of Wilson[24]) used by Fedorenko[12] to develop an algorithm that is called the Multigrid Method, for the solution of linear system problems.

It is interesting to notice that in the above expression for $\rho(M_{DJ})$ the shortest wavelenghts, that is $p_1, p_2 \approx 1$ will provide

$$\begin{aligned} \rho(M_{DJ}) &= 1-\omega(1-cos\tfrac{L\pi}{L+1}) \\ &= 1-\omega(1-(1-\tfrac{1}{2}\cdot(\tfrac{L\pi}{L+1})^2+\tfrac{1}{4!}\cdot(\tfrac{L\pi}{L+1})^4-\ldots)) \\ &= \mathcal{O}(1) \end{aligned} \tag{34}$$

That is, the highest frequencies (shortest wavelenghts) components of the error (cf. $e^k = x^k - \hat{x}$) will decrease in a number of steps that basically independs of L. Pratically what happens is that such high frequency components of the error vanish in a few iterations, leaving only longest wavelenght modes with significant errors. But long wavelenghts are adequately modeled in a coarser grid, hence the strategy of transferring the approximate solution to a coarser grid, where they are dealt with requiring much less computational effort (since the number of sites-or grid points will be much less

than in the finest grid). After solving such a coarser grid problem the solution is transferred back to the finest grid, where the first step to proceed with the iterative search for the solution is to smooth the solution brought from the next coarser grid, diminishing the high frequency errors in very few smoothing steps.

2.3.1 MG algorithm

The algorithm can be set in more precise terms. The goal is to solve equation(1), where H is a non-singular linear operator from an N-dimensional real vector space U to another N-dimensional real vector space V. In order to specify the algorithm, consider now:

(1) A sequence of coarse-grid spaces $U_M \equiv U$, U_{M-1}, $U_{M-2}, \ldots$, U_0 and $V_M \equiv V$, V_{M-1}, $\ldots$, V_0 . Here, $dim\ U_l = dim\ V_l \equiv N_l$, for $0 \leq l \leq M$, and $N = N_M > N_{M-1} > \ldots > N_0$.

(2) Restriction (or "averaging") operators $I_l^{l-1} : V_l \rightarrow V_{l-1} for 1 \leq l \leq M$.

(3) Prolongation (or "interpolation") operators $I_{l-1}^{l} : U_{l-1} \rightarrow U_l$ for $1 \leq l \leq M$

(4) Coarse grid operators $H_l : U_l \rightarrow V_l$ for $0 \leq l \leq M-1$. Of course $H_M \equiv H$. Each of the operators H_l is assumed to be non singular.

(5) Basic (or "smoothing") iterations $\delta_l : U_l \times V_l \rightarrow U_l$ for $0 \leq l \leq M-1$. The role of δ_l is to take an approximate solution x_l^1 to the equation $H_l x_l = b_l$ and compute a new (hopefully better) approximate solution $x_l^2 = \delta(x_l^1, b_l)$. In general δ_l has two ingredients, namely δ_l^{PRE} and δ_l^{POS} , which may be the same but need not be .

(6) Cycle control parameters (integers) $\gamma_l > 0$ for $1 \leq l \leq M$, which control the number of times that the coarse grids are visited.

The MG method is then defined recursively as

Procedure MG(l, x, b)
comment : takes an approximate solution x to the equation $H_l x = b$, and overwrites it with a better approximation

$x \leftarrow \delta_l^{PRE}(x, b)$
if $l > 0$ **then**
$d \leftarrow -I_l^{l-1}(H_l x - b_l)$
$\psi \leftarrow 0$
for $j = 1$ **until** γ_l **do** MG $(l-1, \psi, d)$
$x \leftarrow x + I_{l-1}^{l}\psi$
endif
$x \leftarrow \delta_l^{POS}(x, b)$
End

This constitutes a single step of the multigrid algorithm. In practice these steps would be repeated several times, as in other iterative process, until the error has been reduced to an acceptably small value.

The advantage of multigrid over traditional iterative processes is that, with a suitable choice of the ingredients $I_l^{l-1}, I_{l-1}^{l}, H_l$ and so on, only a few (maybe five or ten) iterations are needed to reduce the error to a small value, *independent of the lattice size* L. This contrasts favorably with the behavior of the damped Jacobi iteration, that requires $\mathcal{O}(L^2)$ iterations.

The MG algorithm above can be regarded as consisting of three main steps (Hackbush[15]):

(1) **Pre-smoothing.** A few iterations of the basic smoother (e.g. damped Jacobi) to the given approximate solution. This produces a better approximate solution in which the high frequency (short wavelenght) components of the error have been reduced significantly. The low-frequency (long wavelenght) components are, however, still large.

(2) **Coarse-grid correction.** The residual $H_l x - b_l$ is computed and transferred to the next coarser grid (level $l-1$) using the restriction operator I_l^{l-1}. Then the result, d, is used as the right hand side of the auxiliary equation $H_{l-1}x = d$, which is solved approximately by γ_l iterations of the Multigrid algorithm at level $l-1$ (recursive definition) with an initial guess $\psi = 0$. The approximation ψ to the solution is transferred back to grid l using the prolongation operator I_{l-1}^{l}, and is used to correct the approximate solution x . The goal of this coarse grid correction is to reduce significantly the low-frequency components of the error in x (hopefully without creating large new high-frequency error components).

(3) **Post smoothing.** A few more iterations of the basic smoother are performed (this is to protect against any high-frequency errors which may inadvertently have been created by the coarse-grid correction step) .

2.3.2 Practical Considerations

Most commonly one uses a uniform factor of 2 coarsening between each grid Ω_l and the next coarser grid Ω_{l-1} . The coarse grid points could be either a subset of the fine grid points or a subset of the dual lattice. In the first group is **standard coarsening** which chooses the coarse-grid points jumping one grid point and picking the next, in all directions the problem is being solved. In the second group a frequently used coarsening is the **red-black** (checkerboard) which divides the grid in a checkerboard fashion, selecting one of the colors as only fine grid points, while in the other color the coarse and fine grid points coincide.

If the coarse grid points are a subset of the fine grid points, then the most simple restriction operator is the **trivial restriction**, defined as

$$(I_l^{l-1}x_l)_v = (x_{l-1})_v \qquad \forall v \in \Omega_{l-1} \subset \Omega_l \tag{35}$$

where here (.) is the value of the variable at grid point v.

However this restriction becomes in general too crude, thus some kind of local averaging (or "coarse graining") is needed, in order to accentuate the low frequency components. For a coarse grid like those whose grid-points do not coincide with fine grid points, **block averaging** is the natural choice(that is, an average is taken using the fine-grid points closer to the coarse-grid point under consideration). For a coarsening such as the standard one, the "nine-point restriction" (or "full weighting") is commonly used, performed with the nine-point-cell of which the coarse grid point is the center (see figure 4).

The coarsening becomes different, and computationally much more expensive, in case the linear operator deals with discontinuous coefficients in the domain Ω of discretization[2].

If the coarse-grid points are a subset of the fine grid points, **trivial prolongation** may be performed.That is:

$$(I_{l-1}^{l}x_{l-1})_v = (x_l)_v \tag{36}$$

in case $v \in \Omega_{l-1}$ (and zero otherwise). In order to improve such prolongation operator, other schemes may be tried: for example, when a nine-point cell is used the counterpart of "full-weighting" restriction may be used, with the same weighting scheme, yelding one of the simplest schemes, named **piecewise linear interpolation**; while the "block averaging" concept will yield the **piecewise constant injection** that also weights equally the nearest neighbors.

Given a prolongation operator one can always define a restriction operator to be its adjoint (transpose), and vice-versa. For trivial prolongation the adjoint is trivial restriction, and for piecewise-constant injection it is block averaging, and so forth.

The most frequent choice of H_l is

$$H_{l-1} = I_l^{l-1} H_l I_{l-1}^l. \tag{37}$$

One note that for certain problems, the adequate consideration of boundary conditions add some additional grid lines (or planes, if in 3−dimensions) and gives rise to altered right-hand-side of the resulting linear system to be solved (see,e.g.,[5]), in case of Neumann or mixed Neumann-Dirichlet boundary conditions.

All classical iterative methods for the solution of linear systems can be used at this moment. The most usual are Jacobi (or damped Jacobi), Gauss-Seidel (depending on the ordering of the grid-points its performance may be increased, that is, lexicographhic or red-black will exhibit different performances [2]), SOR - *successive over-relaxation*-iteration or Conjugate Gradient. But some of the iterative methods will turn out to be too expensive, since they handle many frequencies together (e.g.,CG method), while others such as SOR will work well because the long-wavelenghts are not specially under consideration at the smoothing step (in general the optimal ω for SOR will only be known after testing). Long wavelenghts are dealt with by moving from one grid to a coarser. But high frequency errors may still be present and can be handled by simple Gauss-Seidel(GS).

It is observed that the total number of relaxation steps employed in δ_l^{PRE} and δ_l^{POS} (both being performed through GS) matters more than each of the numbers individually.

In the coarser grid the number of grid point is small enough, so some direct method is used (e.g., LU decomposition).

The parameters γ_l that drive the Multigrid algorithm are usually chosen to be $\gamma_l = \gamma = 1$ (called V-cycle) or $\gamma_l = \gamma = 2$ (called W-cycle), for all grids $l = 1, 2, .., M$ in such a way that, at level l one iteration of the MG algorithm requires one visit to grid l , γ visits to grid $l-1$, γ^2 visits to grid $l-2$, and so forth. Thus, γ determines the degree of emphasis placed on the coarse-grid updates.

2.3.3 Computational Cost

In order to estimate the computational work (cost) required for one iteration of the multigrid algorithm, one sets W_l to represent the computational cost (as usual, it is measured in number of arithmetic operations) for handling the smoothing, restriction, prolongation, and residual computation. For a factor-of-2 coarsening in d dimensions, it comes

$$W_l \approx 2^{-d(M-l)} W_M \ , \tag{38}$$

[2] In fact red-black ordering enables the use of vector and parallel processing since it does not couple grid-points of the same color for the most usual nearest neighbor operators

giving a total work for one MG iteration of

$$\begin{aligned} work(MG) &= \sum_{l=M}^{0} \gamma^{M-l} W_l \\ &\approx W_M \sum_{l=M}^{0} (\gamma 2^{-d})^{M-l} \\ &\leq W_M (1-\gamma 2^{-d})^{-1} \qquad if \quad \gamma < 2^d \end{aligned} \tag{39}$$

Thus, provided that $\gamma < 2^d$, the work required for one entire multigrid iteration is no more than $(1-\gamma 2^{-d})^{-1}$ times the work required for Pre and Post-smoothing (say, using Gauss-Seidel) iterations on the finest grid alone, *irrespective of the total number of levels.*

For certain classes of operators H and suitable choices of the coarse grids, restrictions, prolongations, coarse grid operators, smoothing iterations, and cycle control parameters, it can be proved [15][20][13], that the multigrid iteration matrices M_l satisfy a **uniform** bound

$$\|M_l\| \leq C < 1\,, \tag{40}$$

valid irrespective of the number of levels. Thus, a fixed number of multigrid iterations (maybe some five or ten) are sufficient to reduce the error to a small value, *independent of the lattice size* L. That is, **critical slowing down has been eliminated.**

The efficiency of the multigrid method arises then from the combination of two key features, namely

- The convergence rate for M_l. This means that only $\mathcal{O}(1)$ iterations are needed, independent of the lattice size.
- The work estimate above means that each iteration requires only a computational cost of order L^d (the finest-grid lattice volume).

It follows that the complete solution of the linear system $Hx = b$ requires asymptotically a computational cost of the order of finest grid's number of sites (grid points).

2.4 Attempts to merge MG and CG

Looking forward to making use of both CG and MG concepts, two main groups of attempts were performed. CG methods were used as smoothers for the MG method[6] and MG cycles were implemented as preconditioners for CG iterations [4][19].

Sometimes the use of CG as smoother may not decrease the global MG cost. This is the case in problems with smooth solutions on a sequence of uniform grids. However, if the coefficients of the partial differential equation and the solution are rough and the grids are irregular or not uniformly refined, CG will be convenient for its adaptive nature. That is, as CG does not handle preferably on the high frequency errors (as do conventional smoothers), it will provide an improvement over GS or SOR, for example, leading to reduced global cost (in [6] Poisson's equation is solved to illustrate these aspects).

In [8] a two dimensional linear elasticity problem is presented, whose solution through a standard multigrid algorithm with GS relaxation led to poor reduction factors (≈ 0.6 per unknown per step). With the use of a MG cycle as preconditioner for CG, the PCG thus generated showed reduction factors of 10^{-h} in 7 steps. As the MG employed is only a preconditioner, the problem to be solved does not have to be exactly the same. Thus it is possible to work on simpler problems which could then be solved simultaneously to take advantage of parallelism (see,e.g.[9]).

3 MGCG algorithm

3.1 Basic concepts

The goal here is to merge the philosophy underlying MG with the convenience of finite termination and low cost per iteration offered by CG method. On such grounds one proposes an alternative algorithm, hopefully with an attractive asymptotic rate. A rather different emphasis is adopted, in comparison with those of the previous section. The MGCG method to be described is an unique direction strategy, instead of "back-and-forth" recursive defect correction strategy[10][14][15]. It is analysed under a geometric (vector spaces spanning) point of view and further applied to a simple elliptic boundary value model problem with continuous coefficients.

One initially consider the basic CG ideas, namely the successive generation of subspaces with increasing dimensionality. In addition one build a bridge from one subspace to the one with immediately higher dimensionality. This has the important underlying assumption that the initial estimate is crucial to the efficiency of the CG algorithm, and the links between pairs of grids are constructed through usual MG prolongation and restriction operators.

3.2 Description of MGCG Algorithm

The algorithm can be described in very few steps, namely the one-way loop from the coarsest to the finest grid, with the smoothing and grid to grid transportation inside its scope.

That is,

```
Procedure MGCG ( H ,b , I , m , x )
    Initialize x^l = x^0
    Loop l := 1 to m
        Solve through a few steps of Conjugate Gradient to improve x^l
        Transport to next finest grid, initializing x^l, x^l ← I^l_{l-1} x^{l-1}
        Relax ν times to improve x^{l-1} as solution of H^{l-1} x^{l-1} = b^{l-1}
    Endloop
    Solve through Conjugate Gradient to improve x^m
End of MGCG
```

The tasks of the algorithm are summarized below:

- In the list of parameters the data structures for b, H and I where constructed before the call for MGCG. It contains, respectively, the vectors $b^l, l = 0, \cdots, m$, the matrices $H^l, l = 0, \cdots, m$, which represent the right hand side vectors and the matrices of the various levels' linear systems, and the transport operators $I^j_{j-1}, j = 1, \cdots, m$ from one grid to the next finest.
- The initial problem to be solved should produce a good initial estimate for x^0. This will decrease (see (28)) significantly the cost of improving it.
- The relaxation step is a way of obtaining a more locally (in the context of short wavelenghts of previous sections) accurate estimate result in a given level, having started with a cruder vector estimate transported from the next coarser grid. The optimal value of ν is, then, to be found for each problem in order to lower the overall computational cost.
- The Conjugate Gradient method is implemented, at each level, as a usual minimization procedure, but starting from a good initial "guess" which comes from a next "coarser problem" that mimics the properties of the present grid's problem.
- Transportation of variables from one grid to the next finer may follow all prescriptions that appear in the MG literature, and we initially adopt bilinear interpolation. Any costless procedure will be convenient for the performance of the global algorithm.

3.3 Theoretical Aspects

In this section a result is proved that states the finite termination of the MGCG algorithm, once the algorithm proposed has been presented as a generic procedure.

Using the same notation as Polak[22] to start the description of the result concerning MGCG one set:

- $I_{N_i}^{N_{i+1}}$ or I_i^{i+1} - transport operator from vectors in $\mathbf{R}^{N_i}$ to $\mathbf{R}^{N_{i+1}}$.
- $\tilde{I}_i^k = I_{k-1}^k \cdot I_{k-2}^{k-1} \cdots I_i^{i+1}$ if $i < k$ and $\tilde{I}_k^i = I_{i-1}^i \cdots I_k^{k+1}$ in case $i > k$.

Theorem. Consider a sequence $H^{N_0}, \cdots, H^{N_k}$ of matrices in $\mathbf{R}^{N_i \times N_i}$ $(i = 0, \cdots, k)$ and transport operators such that

$$I_i^{i+1} \ : \ I_i^{i+1} \ w^i \rightarrow w^{i+1} \tag{41}$$

Suppose,

$$I_{N_i}^{N_{i+1}} = (\ I_{N_{i+1}}^{N_i}\ )^{-1} \ = \ (\ I_{N_{i+1}}^{N_i}\ )^t \tag{42}$$

(as suggested in [7]), and all H^j are positive definite matrices. For $i = 0, 1, 2, \cdots$ let

$$r_{k+1} = I_k^{k+1} \cdot \{r_k - \alpha_k H^k \rho_k\} \tag{43}$$

and

$$\rho_{k+1} = r_{k+1} + \beta_k I_k^{k+1} \rho_k \ , \tag{44}$$

where

$$\alpha_k = \frac{r_k r_k}{\rho_k H^k \rho_k} \tag{45}$$

and

$$\beta_k = \frac{r_{k+1} r_{k+1}}{r_k r_k} = \frac{r_{k+1} \cdot \{H^{k+1} I_k^{k+1} \rho_k\}}{\{I_k^{k+1} \rho_k\}\{H^{k+1} I_k^{k+1} \rho_k\}}. \tag{46}$$

Then

$$r_j \tilde{I}_i^j r_i = \delta_{ij} \cdot (r_i r_i) \ , \tag{47}$$

$$\{\tilde{I}_i^k \rho_i\}\{H^k \rho_k\} = \delta_{ij} \cdot (\rho_i H^i \rho_i) \tag{48}$$

and $r_i = \rho_i = 0 \ , \forall \ i > m$ with $m \leq d - 1$ ($N_0, \cdots, N_d$).

Proof

First part: One proves by induction that after **d** iterations, null vectors are generated.
if $r_{m+1} = 0$, then $\rho_{m+1} = 0$ (via the way of obtaining ρ_{k+1})
if $\rho_{m+1} = 0$, one must prove that $r_{m+1} \cdot (I_m^{m+1} \rho_m) = 0$. But as $r_0 = \rho_0$, it follows that

$$\begin{aligned} I_0^1 \rho_0 r_1 &= \rho_0 [I_0^1 \{r_0 - \alpha_0 H^0 \rho_0\}] \\ &= (I_0^1 \rho_0) \cdot (I_0^1 r_0) - \tfrac{r_0 r_0}{\rho_0 H^0 \rho_0} \cdot (I_0^1 \rho_0 \ I_0^1 H^0 \rho_0) \\ &= \rho_0 r_0 - \tfrac{r_0 r_0}{\rho_0 H^0 \rho_0} \cdot (I_0^1 \rho_0 \ I_0^1 H^0 \rho_0) \\ &= 0 \end{aligned} \tag{49}$$

Here it was used the fact $(I_k^{k+1}w_k)\cdot(I_k^{k+1}v_k) = I_{k+1}^k I_k^{k+1} w_k\ v_k = w_k v_k$.

Suppose that $\{I_j^{j+1}\rho_j\}\cdot r_{j+1} = 0\ ,\forall\ j\in[1,\cdots,m-1]$ (Induction Hypothesis). For $j=m$:

$$\begin{aligned} I_j^{m+1}\rho_m r_{m+1} &= \rho_m\cdot\{r_m - \alpha_m H^m\rho_m\} \\ &= \rho_m r_m - \alpha_m(\rho_m H^m\rho_m) \\ &= \{r_m + \beta_{m-1}I_{m-1}^m\rho_{m-1}\}\ r_m - \alpha_m\ (\rho_m H^m\rho_m) \\ &= r_m r_m - \tfrac{r_m r_m}{\rho_m H^m\rho_m}\cdot\rho_m H^m\rho_m = 0. \end{aligned} \tag{50}$$

Here the induction hypothesis was used at the last line. *This ends the first part of the proof.*

Second Part: One now proves the following Orthogonality Relations:

$$r_i\ \tilde{I}_j^i r_j = \delta_{ij}\cdot(r_i r_i) \tag{51}$$

and

$$\tilde{I}_i^k\rho_i\ H^k\rho_k = \delta_{ij}\cdot(\rho_i H^i\rho_i) \tag{52}$$

By induction, suppose that for some natural $0\le k<m$

$$(\tilde{I}_i^j r_i)\cdot r_j = (\tilde{I}_i^j\rho_i)\cdot(H^j\rho_j) = 0\ ,\forall i\neq j,\ 0\le i,j\le k\ . \tag{53}$$

Let $i\in\{1,\cdots,k-1\}$. So

$$\begin{aligned} r_{k+1}\cdot\{\tilde{I}_i^{k+1}r_i\} &= I_k^{k+1}\{r_k-\alpha_k H^k\rho_k\}\cdot\{\tilde{I}_i^{k+1}r_i\} \\ &= (I_k^{k+1}r_k)\cdot\{\tilde{I}_i^{k+1}r_i\} - \alpha_k(I_k^{k+1}H^k\rho_k)\cdot(\tilde{I}_i^{k+1}r_i) \\ (hyp.) \quad &-\tfrac{r_k r_k}{\rho_k H^k\rho_k}\cdot(I_k^{k+1}H^k\rho_k)\cdot(\tilde{I}_i^{k+1}r_i) \\ &= -\tfrac{r_k r_k}{\rho_k H^k\rho_k}\cdot(I_k^{k+1}H^k\rho_k)\cdot\tilde{I}_i^{k+1}\{\rho_i-\beta_{i-1}I_{i-1}^i\rho_{i-1}\} = 0\ , \end{aligned} \tag{54}$$

since $I_k^{k+1}H^k\rho_k\ \tilde{I}_i^{k+1}\rho_i = H^k\rho_k\ \tilde{I}_i^k\rho_i = 0\ (hyp.)$.

It is also true that

$$r_{k+1}I_k^{k+1}\rho_k = \{r_k-\alpha_k H^k\rho_k\}\cdot\rho_k = r_k\rho_k - r_k r_k = 0.$$

Similarly,

$$\begin{aligned} \rho_{k+1}(I_k^{k+1}H^k\rho_k) &= (r_{k+1}+\beta_k I_k^{k+1}\rho_k)\cdot(I_k^{k+1}H^k\rho_k)\cdot\{(I_k^{k+1}H^k\rho_k)\ \tilde{I}_i^{k+1}r_i\} \\ &= r_{k+1}\cdot(I_k^{k+1}H^k\rho_k) - \tfrac{r_{k+1}H^k I_k^{k+1}\rho_k}{I_k^{k+1}\rho_k(H^{k+1}I_k^{k+1}\rho_k)}\cdot(\rho_k H^k\rho_k) = 0\ , \end{aligned} \tag{55}$$

since $H^k = I_{k+1}^k H^{k+1} I_k^{k+1}$. That is, for $0 \leq i \leq k-1$ one has

$$\begin{aligned} \rho_{k+1} \cdot (\tilde{I}_i^{k+1} H^i \rho_i) &= \{r_{k+1} + \beta_k I_k^{k+1} \rho_k\} \cdot (\tilde{I}_i^{k+1} H^i \rho_i) \\ &= r_{k+1} \cdot (\tilde{I}_i^{k+1} H^i \rho_i) \\ &= r_{k+1} \cdot \tilde{I}_i^{k+1} \{-\tfrac{1}{\alpha_i}[I_{i+1}^i r_{k+1} - r_i]\} = 0 \ , \end{aligned} \tag{56}$$

since $i \leq k-1$ and $\alpha_i \neq 0$, $i = 0, \cdots, n$.

Therefore, one concludes that $r_i \tilde{I}_j^i r_j = 0 = \rho_i I_j^i H^j \rho_j$, for $i \neq j$ and $0 \leq i, j \leq k$. As each pair of such vectors is orthogonal to each other, and they are all non-zero, its total number cannot exceed d, that is $m \leq d$.

End of Second Part.

This completes the whole proof!

For practical reasons the implementation of algorithm MGCG will perform some Conjugate Gradient iterations before moving to the next finer subspace. Thus, several successive values of N_i are equal to it. For this group of, say, p values $N_i, N_{i+1}, \cdots, N_{i+p-1}$ there is a simplification in that $I_k^{l+1} = I_i^{i+1} = I$, $l, k = i, \cdots, i+p-2$ and the matrices H^k also remain the same.

With the above, MGCG becomes a true generalization of conventional CG, allowing simultaneously multilevel calculations.

4 Concluding Remarks

Further efforts will be performed which will check the extension of applicability for this MGCG method in particular through the solution of elliptic model problems with discontinuous coefficients, which come up, e.g. in neutron diffusion[2] or Petroleum Reservoir modelling [5]. Also, it should be tested in some hyperbolic problems.

In most complex applications the strict use of our MGCG algorithm should be tested together with some Domain Decomposition techniques(e.g.[9]). In the regions where smooth variations of coefficients characterize the problem our method can be more effective, since it provides low cost for the global procedure.

It is worth remarking, finally, that a new proposal for a mixed Conjugate Gradient + Multigrid Method was performed, making use of many interesting features of both classes of methods, which has to be submitted to many tests, in order to evaluate its applicability in general problems.

List of References

1- Albrecht,P. ; "Análise Numérica- Um Curso Moderno" - ed.LTC (1973)

2- Alcouffe,R.E.,Brandt,A.,Dendy,J.E.Jr & Painter,J.W. - "The Multigrid method for the diffusion equation with strongly discontinuous coefficients" - SIAM J.Sci. Statistics and Computation, 2, 430-454 (1981)

3- Axelsson,O & Barker,V.A. - "Finite Element Solution of Boundary Value Problems - Theory and Computation" - Academic Press-1984

4- Axelsson,O & Gustafsson,I.; "A combination of preconditioning and multigrid methods" Technical Report 8120, Univ.Nijmegen (1981)

5- Aziz,K. & Settari,A.; "Petroleum Reservoir Simulation" ; ed. London: Applied Science (1979)

6- Bank,R.E. & Douglas,C. 'Sharp Estimates for Multigrid Rates of Convergence with General Smoothing and Acceleration"; SIAM J. Numer. Anal. vol22.,No.4 (1985)

7- Behie,A. & Forsyth Jr,R -Sixth Annual Symposium on Reservoir Simulation -SPE- New Orleans (1982)

8- Braess,D. ; "On the Combination of the Multigrid Method and Conjugate Gradients";

9- Bramble,J.H.,Pasciak,J.E.,Schatz,A.H.; "The construction of Preconditioners for Elliptic Problems by Substructuring"; I. [vol. 47,No 175, 103-134]; II. [vol 49, No.170, 1-16]-Math.Comp. (1986)

10- Brandt,A.;"Multilevel Adaptive Solutions to boundary-value problems" - Mathematics of Computation **31**,333-390 (1977)

11- Cláudio,D. & Marins,J. ;"Cálculo Numérico Computacional"- ed. Atlas (1989)

12- Fedorenko,R.P.; URSS-Computational and Mathematical Physics 4 227 (1964)

13- Goodman,J & Sokal,A - "Multigrid Monte Carlo Methods: Conceptual Foundations"; Phys. Rev. **40**, 2035-2071 (1989)

14- Hackbush,W.;"Convergence of multigrid iteration applied to difference equations"; Math. Comp. **34** 425-440 (1980)

15- Hackbush,W.; Multigrid Methods and Applications- Springer, Berlin (1985)

16- Hackbush,W. & Trottenberg,U.,eds.; "Multigrid Methods"- Lecture Notes in Mathematics No. 960-Springer, Berlin (1982)

17- Hestenes,M.R. & Stiefel,E. ; "Method of Conjugate Gradients for solving linear systems"; Journal of Research of the National Bureau of Standards"; **49**, 409-436 (1952)

18- Kadanoff,L.P.; Physics **2**, 263-290 (1966)

19- Kettler,R. & Meijerink,J.A. ; "A multigrid method and a combined multigrid-conjugate gradient method for elliptic problems with strongly discontinuous coefficients in general domains"; Tech.Report 604, Shell Oil Company (1981)

20- McCormick,S.F., ed.; "Multigrid Methods" [SIAM Philadelphia] (1987)

21- Ortega,J.M.; "Numerical Analysis: A Second Course"- Academic Press, New York (1972)

22- Polak,E. - "Computational Methods in Optimization. A Unified Approach"; Academic Press (1971)

23- Reif,F. "Fundamentals of Statistical and Thermal Physics" ed. MacGraw Hill (1965)

24- Wilson,K..G.; Physical Review B **4**, 3174 (1971)

List of Figures

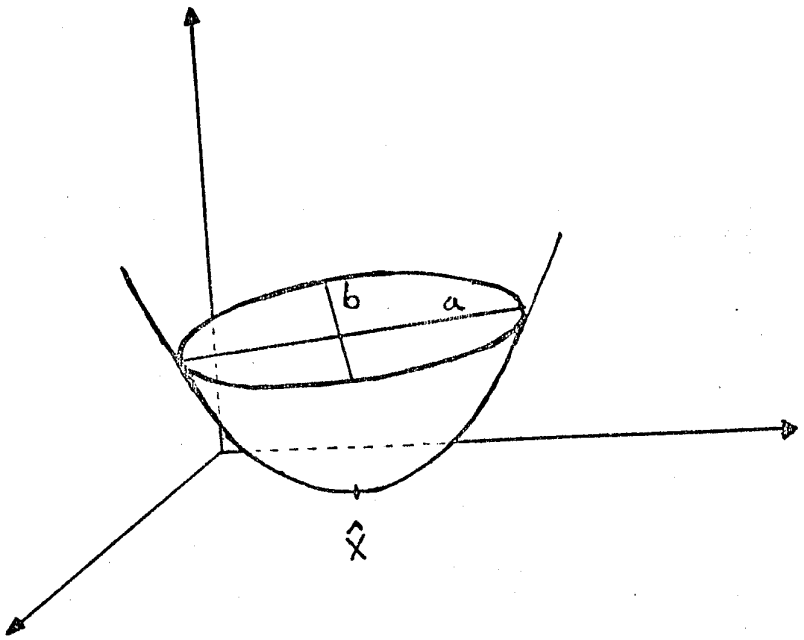

Figure1- Two dimensional distortion measure, $\frac{a}{b}$

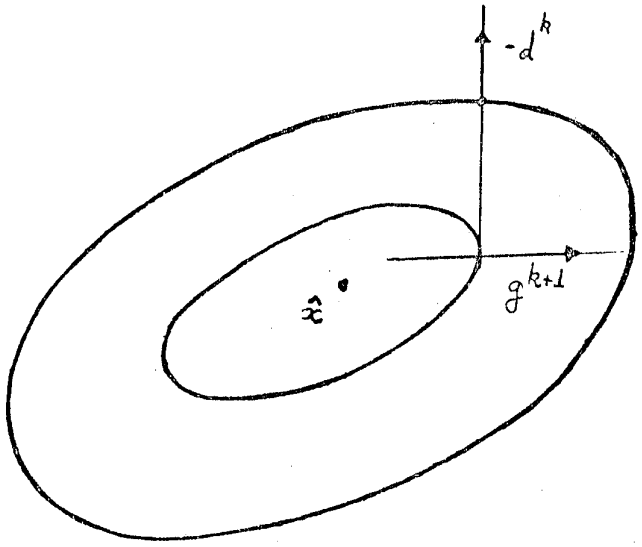

Figure 2: The new search direction(d^{k+1}) is obtained combining the previous one(d^k) with the present gradient(g^{k+1}).

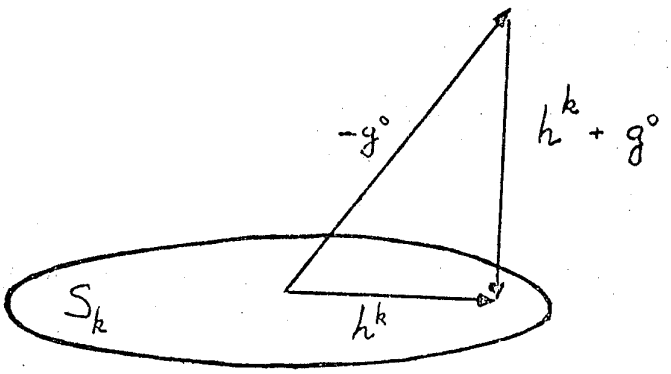

Figure 3- The gradient g^k is orthogonal to all $g \in T_{k-1}$, in order to minimize $\|h\| \in S_{k-1}$.

1	2	1
2	4	2
1	2	1

Figure 4- Stencil for "full weighting"