A Globally Convergent Inexact Newton Method with a New Choice for the Forcing Term *

Márcia A. Gomes-Ruggiero Véra Lucia Rocha Lopes
Julia Victoria Toledo-Benavides †

Abstract

In inexact Newton methods for solving nonlinear systems of equations, an approximation to the step s_k of the Newton's system $J(x_k)s = -F(x_k)$ is found. This means that s_k must satisfy a condition like $\|F(x_k) + J(x_k)s_k\| \leq \eta_k \|F(x_k)\|$ for a forcing term $\eta_k \in [0, 1)$. Possible choices for η_k have already been presented. In this work, a new choice for η_k is introduced. The method is globalized by the introduction of a robust backtracking strategy proposed by [2], and its convergence properties are proved. Several numerical experiments with boundary value problems are presented. The numerical performance of the proposed algorithm is analyzed by the performance profile tool proposed by Dolan and Moré in [11]. The results obtained show a competitive inexact Newton method for solving academic and applied problems in several areas.

Keywords. nonlinear systems of equations, inexact Newton method, forcing term, Newton iterative methods, line search.

AMS subject classifications: 65H10, 65F10.

*Supported by FAPESP, CNPq, PRONEX-Optimization.

†Department of Applied Mathematics, IMECC-UNICAMP, University of Campinas, CP 6065, 13083-970 Campinas SP, Brazil. e-mails: marcia@ime.unicamp.br, vlopes@ime.unicamp.br and juliatoledob@yahoo.com.br

Consider the nonlinear system

$$F(x) = 0, \tag{1}$$

where $F \in C^1(D, I\!R^n)$ with D an open convex set, and assume that there is $x_* \in D$ such that $F(x_*) = 0$ with $J(x_*)$ nonsingular, where $J(y)$ is the Jacobian matrix of F at y. From now on, $\|\cdot\|$ is a norm in $I\!R^n$ and also the corresponding induced matrix norm.

The most popular method for solving problem (1) is Newton's method. The k–th step of this method consists on: given x_k, find s_k, the exact solution of

$$J(x_k)s = -F(x_k). \tag{2}$$

Then,

$$x_{k+1} = x_k + s_k. \tag{3}$$

Equation (2) is called *Newton's equation.*

It is well known that Newton's method has local quadratic convergence [9] if the Jacobian matrix is Lipschitz continuous and nonsingular at a solution of the system. On the other hand Newton's method may be computationally expensive: at each iteration the Jacobian matrix at x_k must be computed and the solutions of the linear system (2) is required. To get rid of these drawbacks, several modifications of Newton's method were proposed in the last 40 years, such as the quasi-Newton methods, see [6], [7], [14], [15], [19], [20] and [21], the discrete Newton method, see [9] and the inexact Newton methods, see [8].

These methods are largely used for solving applied problems in several areas such as Geophysics, Engineering, Chemistry, Physics and so on. In this work we focused in a set of boundary value problems. Further applications considered by us are ray-tracing problems in Geophysics [3], [22] and combustion, see [22].

For solving (1) with an inexact Newton method, the step s_k in the k–th iteration (called an *outer iteration*) is found approximately. Solving approximately means that s_k must satisfy a condition like

$$\|F(x_k) + J(x_k)s_k\| \leq \eta_k \|F(x_k)\|, \tag{4}$$

for a *forcing term* $\eta_k \in [0, 1)$ [8]. For solving (4) we use in this work an iterative linear system solver, the *Generalized Minimum Residual* with restarts, GMRES(m), [24]. At the k–th iteration of inexact Newton method, each iteration of GMRES is called an *inner iteration.*

In this work, we make a brief description of commonly used choices for the forcing term η_k, proposed by Eisenstat and Walker in [13]. A new choice is then introduced with the aim of reducing the number of inner iterations performed by the linear solver. The algorithm proposed

incorporates a globalizing nonmonotone backtracking strategy. Monotone backtracking strategies for inexact Newton methods have been widely used in recent researches, [12]. Even though the search direction that satisfies (4) using the 2–norm is a descent direction for $\|F\|_2$ (see [5]) a nonmonotone strategy results in a more tolerant process, mainly at the initial outer iterations. We expect a small number of backtrackings even knowing that some small increases in $\|F\|$ may occur. In Section 1 we describe the new algorithm which incorporates our choice for η_k and a nonmonotone backtracking strategy, and prove convergence results for it. In Section 2 we present and analyze the numerical performance of this algorithm. This is done by plotting the performance profile [11] of the method with the choice proposed, in comparison with the ones proposed in [13] and the choice with a constant value for η_k. Concluding remarks are given in Section 3.

1 The Inexact Newton Method with the New Choice for η_k

The inexact Newton method proposed in this work introduces a new choice for the forcing term η_k. A backtracking strategy is incorporated to increase the algorithm robustness. These features of the method will be described after a brief review of existing inexact Newton methods.

1.1 About Inexact Newton Methods

Dembo, Eisenstat and Steihaug [8] proposed an algorithm for an inexact Newton method for finding x_*, in which an approximate solution s_k for (2) must be found, such that the inexact Newton condition

$$\|F(x_k) + J(x_k)s_k\| \leq \eta_k \|F(x_k)\|, \tag{5}$$

is satisfied for $0 \leq \eta_k \leq \eta < 1$.

As its name indicates, η_k controls the precision of solving (2); at the same time, it determines the number of inner iterations to be performed at each outer iteration. Choosing η_k too small may, at times, increase the number of inner iteration without guaranteeing significant reduction in $\|F\|$, a phenomenon described by Eisenstat and Walker in [13] as *oversolving*. The main purpose of introducing good choices of η_k is to avoid oversolvings; obviously, another purpose is the achievement of fast local convergence also. In [13] the authors introduce two choices for η_k and they prove convergence results for both choices. Their first choice of η_k reflects the agreement between the function and its local linear model:

Choice 1: For $\alpha = (1+\sqrt{5})/2$ and $\eta_0 \in [0,1)$,

$$\eta_k = \frac{\|F(x_k) - F(x_{k-1}) - J(x_{k-1})s_{k-1}\|}{\|F(x_{k-1})\|}, \qquad k = 1,2,3,\ldots, \tag{6}$$

using as safeguard

$$\eta_k = \max\{\eta_k, \eta_{k-1}^{\alpha}\} \quad \text{each time} \quad \eta_{k-1}^{\alpha} > 0.1. \tag{7}$$

The second choice of Eisenstat and Walker in [13] measures the decreasing factor in the value of $\|F\|$:

Choice 2: Given $\gamma \in [0,1]$, $\alpha \in (1,2]$ and $\eta_0 \in [0,1)$, choose

$$\eta_k = \gamma\left(\frac{\|F(x_k)\|}{\|F(x_{k-1})\|}\right)^{\alpha}, \qquad k = 1,2,3,\ldots, \tag{8}$$

using the safeguard

$$\eta_k = \max\{\eta_k, \gamma\eta_{k-1}^{\alpha}\} \quad \text{each time} \quad \gamma\eta_{k-1}^{\alpha} > 0.1. \tag{9}$$

In both cases some other practical safeguards were also used for not allowing η_k to become too small too fast, which causes oversolving.

Under adequate assumptions on F and assuming that $F(x_k) \neq 0$ for all k, the authors in [13] proved superlinear convergence results for the algorithm with Choice 1, and for Choice 2 the convergence is proved to be of $q-$order α if $\gamma < 1$. If $\gamma = 1$ the convergence is of $r-$order α and $q-$order p for every $p \in [1,\alpha)$. In order to get a globally convergent algorithm a line search is usually performed, in which $x_{k+1} = x_k + \alpha_k s_k$, where $\alpha_k > 0$ must be such that a sufficient decrease in $\|F(x)\|$ is achieved.

1.2 The New Choice for the Forcing Term η_k

In this work we propose a strategy in which the forcing term η_{k+1} is made dependent on both the change in $\|F\|$ and the net computational cost invested during the k–th outer iteration, including inner iterations and backtracking. We introduced in the algorithm the global line search used in [2] and [10], focusing on minimizing both the inner and the outer number of iterations.

1.3 The Geometrical Motivation

When trying to choose the next value for the forcing term, η_{k+1}, it seems to be important to consider the following: the change in the norm of F, and the computational cost at the iteration k.

Since each outer iteration involves the solution of a linear system and a line search procedure, we define the computational cost (price_k) as the number of iterations performed by the linear solver (iterin_k) plus the number of function evaluations (feval_k), that is, $\text{price}_k = \text{iterin}_k + \text{feval}_k$. Note that both iterin_k and feval_k represent the total number of inner iterations and the total number of function evaluations performed during the first k outer iterations. In Figure 1 the horizontal axis represents the total number of inner iterations performed from the beginning of the process, and the vertical axis represents the $\log_{10} \|F(x_k)\|$. A situation is shown, where after 20 inner iterations (at the first outer iteration) the value of the norm of the function F has increased. In contrast, the second angle indicates that $\|F\|$ decreased steadily during the nineth outer iteration.

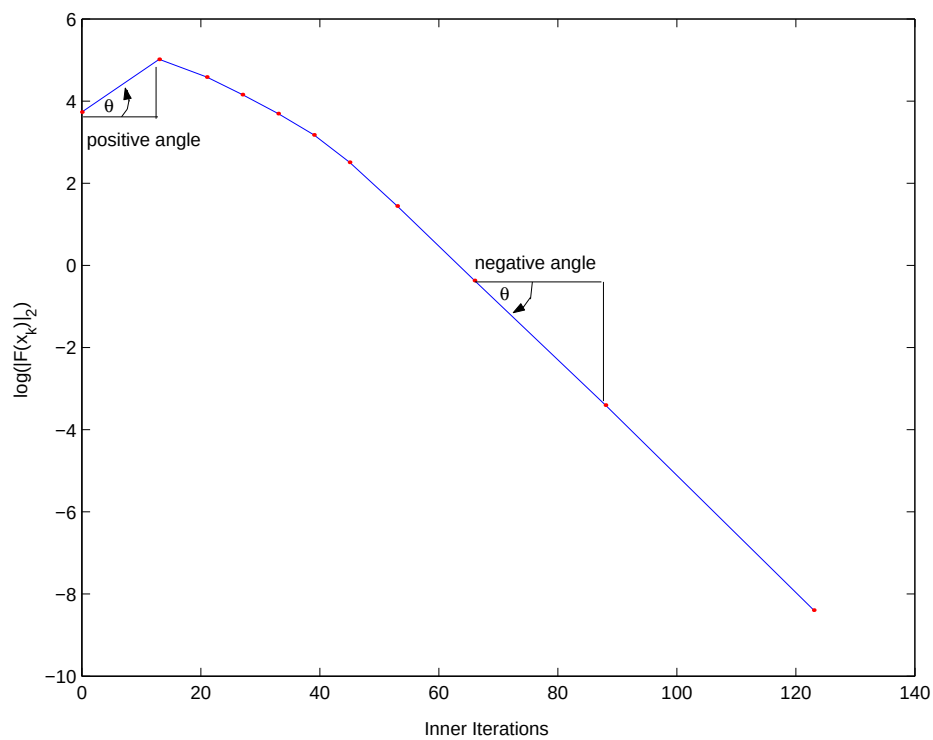

Figure 1: Geometrical motivation for choosing η_k

In general, $\cos(\theta_k)$ can be described as the ratio:

$$\frac{b_k}{\sqrt{a_k^2 + b_k^2}}, \tag{10}$$

where

$$a_k = (\log_{10} \|F_k\| - \log_{10} \|F_{k-1}\|) \text{ and } b_k = \log_{10}(\text{price}_k - \text{price}_{k-1}).$$

Figure 1 shows that the ratio (10) is the cosine of the angle θ_k, ($\theta_k \in (-\pi/2\,,\,\pi/2)$). The value of $\cos(\theta_k)$ is a good measure for the tradeoff between convergence and computational costs. If it is close to -1, we are doing fine and a stricter forcing term may be tried. If it is close to zero, the

iterations are either too costly or are getting nowhere (oversolving) and the forcing term has to be relaxed. If it is positive, $\|F\|$ has actually increased and drastic action is necessary.

Therefore, our strategy is to tie the choice of η_k to the variations of $\theta_k \in [-\pi/2,\ 0]$ in the following way: the value of η_k is decreased when θ_k is close to $-\pi/2$; otherwise, its value is increased. Keeping in mind the convergence of the algorithm, we now introduce our choice: given $\rho \in (1,\ 2]$,

$$\eta_k \ = \ [1/(k+1)]^{\rho} \cos^2(\theta_k) \frac{\|F(x_k)\|}{\|F(x_{k-1})\|}. \tag{11}$$

We observe that θ_k is a product of three terms. The first term, $[1/(k+1)]^{\rho}$ guarantees that $\lim_{k\to 0} \eta_k \to 0$, implying a superlinear convergence rate (see [8]). Besides that, this factor constitutes also a weight for the term $\cos^2(\theta_k)$. Observe that, for the same angle θ, $[1/(k+1)]^{\rho} \cos^2(\theta)$ decreases as k increases, ensuring superlinear convergence. The second term, $\cos^2(\theta_k)$ has the geometrical motivation already described. We use the second power to avoid the computation of a square root and also to accelerate the convergence of the process, since the function $\cos(\theta)$ increases very slowly from 0 to 1, when $\theta \in [-\pi/2, 0]$. This power was also found numerically satisfactory. Finally, the factor $\|F(x_k)\|/\|F(x_{k-1})\|$ reflects the decreasing rate in $\|F\|$ from iteration $(k-1)$ to iteration k. Safeguards are triggered in two situations: (i) when $\theta_k > 0$ and (ii) to maintain η_k bounded from 1 as required by the convergence theorems.

Although the expression (11) for η_k is similar to that of Choice 2, from Eisenstat and Walker ([13]), we observe that there exists a crucial difference between them: in our choice γ changes dynamically during the process, while in Choice 2 of [13], γ remains constant.

1.4 Model Algorithm and Convergence

1.4.1 The Algorithm

We are now ready to introduce the algorithm. Assume that $F : I\!R^n \to I\!R^n$ is continuously differentiable in $I\!R^n$ and assume also that there exists $\eta \in (0, 1)$. Let $\sigma \in (0\,, 1-\eta)$ and let $0 < \varrho_{min} < \varrho_{max} < 1$. Assume further that $\{\mu_k\}$ is a positive sequence such that $\sum_{k=0}^{\infty} \mu_k = \mu < \infty$. In this section, $\|\,.\,\|$ means the Euclidean norm.

Let $x_0 \in I\!R^n$ be an arbitrary initial point, and set $k = 0$. Given $x_k \in I\!R^n$, the steps for obtaining the new iterate x_{k+1} of the inexact Newton algorithm with a nonmonotone backtracking strategy and precision $\varepsilon > 0$, are the following:

Algorithm 1 (Inexact Newton method with nonmonotone global strategy):

While $\|F(x_k)\| > \varepsilon$,

`Step 1:` Choose $\eta_k \leq \eta$.

`Step 2:` (Compute the search direction) Find s_k such that $\|F(x_k) + J(x_k)s_k\| \leq \eta_k \|F(x_k)\|$.

`Step 3:` (Backtracking) Set $\xi = 1$, compute $x_{aux} = x_k + \xi s_k$ and $F(x_{aux})$.

`Step 4:` While

$$\|F(x_{aux})\| > [1 - \xi\sigma]\|F(x_k)\| + \mu_k, \tag{12}$$

`step 4.1:` compute $\xi_{new} \in [\varrho_{min}\xi,\ \varrho_{max}\xi]$,

`step 4.2:` set $\xi = \xi_{new}$ and $x_{aux} = x_k + \xi s_k$.

`Step 5:` Set $\xi_k = \xi$; $x_{k+1} = x_k + \xi_k s_k$; $k = k + 1$.

The line search used in step 4, see [2], besides being a global strategy, is intended to avoid outer oversolvings. It is a nonmonotone strategy similar to the one introduced by Li and Fukushima in [18], but less prone to scaling problems. Observe that this iteration is well defined and that s_k is allowed to be equal to zero. Note that the process with the strategy proposed by Birgin, Krejić and Martínez in [2] is asymptotically monotone.

1.4.2 Convergence

In what follows we state and prove convergence results for Algorithm 1.

Lemma 2.5.1. *Let $\{x_k\}$ be a sequence generated by Algorithm 1. If, for some sequence of indices $K_0 \subset \{0, 1, 2, \ldots\}$, $\lim_{k \in K_0} F(x_k) = 0$, then $\lim_{k \to \infty} F(x_k) = 0$. In particular, if x_* is a limit point of x_k such that $F(x_*) = 0$, then every limit point of the sequence x_k is a solution of (1).*

Proof. As in [2] Lemma 1. Note that our $-\sigma$, corresponds to the term $\sigma(\theta - 1)$ in [2]. ∎

The next lemma will be used in the proof of Theorem 2.5.2.

Lemma 2.5.2. *Let $\{x_k\}$ be a sequence generated by Algorithm 1 and assume that all the limit points of the sequence $\{x_k\}$ are solutions of (1). Assume also that x_* is a limit point of $\{x_k\}$ such that $J(x_*)$ is nonsingular and*

$$\lim_{k \to \infty} \|x_{k+1} - x_k\| = 0.$$

Then, the whole sequence converges to x_*.

Proof. As in [2] Lemma 2. ∎

We conclude this section with the convergence theorems.

Theorem 2.5.1. *Assume that the sequence $\{x_k\}$ is generated by Algorithm 1 and that there exists $M > 0$ such that, for an infinite sequence of indices $K_1 \subset \{0, 1, 2, \ldots\}$, $\|s_k\| < M$. Then, any limit point of the subsequence $\{x_k\}_{k\in K_1}$ is a solution to system (1). Moreover, if a limit point of $\{x_k\}_{k\in K_1}$ exists, then $F(x_k) \to 0$ and every limit point of $\{x_k\}$ is a solution to (1).*

Proof. Let $K_2 \subset K_1$ be a sequence of indices such that $\lim_{k\in K_2} x_k = x_*$. The proof will be done, considering two cases. First, let us assume that $\{\xi_k\}_{k\in K_2}$ does not tend to 0. In this case, there exists a sequence K_3 of indices, $K_3 \subset K_2$ and $\bar{\xi} > 0$ such that $\xi_k \geq \bar{\xi} > 0$, $\forall\, k \in K_3$.

By (12), $\|F(x_{k+1})\| \leq \|F(x_k)\| - \bar{\xi}\sigma\|F(x_k)\| + \mu_k, \;\; \forall k \in K_3$.

But, for all k, including $k \notin K_3$, $\|F(x_{k+1})\| \leq \|F(x_k)\| + \mu_k$. Then, adding all these inequalities in k, we have:

$$\sigma\bar{\xi} \sum_{k\in K_3} \|F(x_k)\| \leq \|F(x_0)\| + \sum_{k=0}^{\infty} \mu_k = \|F(x_0)\| + \mu.$$

Therefore, $\lim_{k\in K_3} \|F(x_k)\| = 0$ and then, $F(x_*) = 0$. By Lemma 2.5.1, we have that $\lim_{k\to\infty} F(x_k) = 0$.

Let us consider now the second case, in which we assume that $\lim_{k\in K_2} \xi_k = 0$. Taking into account the way in which ξ_{new} is chosen, for $k \in K_2$, k large enough, there exists $\xi_k^{'} > \xi_k$, $\xi_k^{'} \in [\xi_k/\varrho_{max}\,,\, \xi_k/\varrho_{min}]$ such that $\lim_{k\in K_2} \xi_k^{'} = 0$ and

$$\|F(x_k + \xi_k^{'} s_k)\| > \|F(x_k)\| - \xi_k^{'}\sigma\|F(x_k)\| + \mu_k.$$

Then,

$$\|F(x_k + \xi_k^{'} s_k)\| > (1 - \xi_k^{'}\sigma)\|F(x_k)\|.$$

Hence,

$$\|F(x_k + \xi_k^{'} s_k) - [F(x_k) + J(x_k)\xi_k^{'} s_k]\| + \|F(x_k) + J(x_k)\xi_k^{'} s_k\| > (1 - \xi_k^{'}\sigma)\|F(x_k)\|.$$

Thus, by the triangular inequality,

$$\|F(x_k + \xi_k^{'} s_k) - F(x_k) - J(x_k)\xi_k^{'} s_k\| + \|\xi_k^{'}[F(x_k) + J(x_k)s_k]\| + (1 - \xi_k^{'})\|F(x_k)\| > (1 - \xi_k\sigma)\|F(x_k)\|.$$

Therefore,

$$\|F(x_k + \xi_k^{'} s_k) - F(x_k) - J(x_k)\xi_k^{'} s_k\| + \xi_k^{'}\eta\|F(x_k)\| + (1 - \xi_k^{'})\|F(x_k)\| > (1 - \xi_k^{'}\sigma)\|F(x_k)\|.$$

After some algebraic manipulation, we have

$$\xi_k^{'}\|F(x_k)\|(1 - \sigma - \eta) < \|F(x_k + \xi_k^{'} s_k) - F(x_k) - J(x_k)\xi_k^{'} s_k\|.$$

Then,

$$(1 - \sigma - \eta)\|F(x_k)\| < \frac{\|F(x_k + \xi_k^{'} s_k) - F(x_k) - J(x_k)\xi_k^{'} s_k\|}{\xi_k^{'}}. \tag{13}$$

Now, $\|s_k\|$ is bounded, $\{\xi_k^{'}\}$ tends to 0, $\sigma+\eta<1$ and $F^{'}$ is continuous. These facts together imply that the right-hand side of (13) tends to 0 when $k \in K_2$. Therefore, $\lim_{k\in K_2} \|F(x_k)\| = 0$ and then $F(x_*) = 0$. The rest of the proof follows from Lemma 2.5.1. ∎

The following lemma will be needed in the proof of Theorem 2.5.2.

Lemma 2.5.3. *If the sequence $\{x_k\}$ is generated by Algorithm 1, then $F(x_k)$ is bounded.*

Proof. By the line search procedure, for any $k = 0, 1, 2, \ldots$,

$$\|F(x_k + \xi_k s_k)\| \leq \|F(x_k)\|(1 - \sigma\xi_k) + \mu_k. \tag{14}$$

Then, for a general k,

$$\|F(x_{k+1})\| \leq \|F(x_0)\| + \sum_{i=0}^{k} \mu_i.$$

Since $\sum_{i=0}^{k} \mu_i = \mu < +\infty$, $\|F(x_k)\| \leq \|F(x_0)\| + \mu$, $\forall\, k$. ∎

Theorem 2.5.2. *Let $\{x_k\}$ be generated by Algorithm 1. Assume that:*
1. F is coercive in $I\!R^n$, that is, $\lim_{\|x_k\|\to\infty} \|F(x_k)\| = +\infty$;
2. $J(x)$ is nonsingular and $\|J(x)^{-1}\| \leq M$, $\forall\, x \in I\!R^n$.
Then there exists $x_* \in I\!R^n$ such that $\lim_{k\to\infty} x_k = x_*$ and $F(x_*) = 0$.

Proof. By Lemma 2.5.3 we have that $F(x_k)$ is bounded. Then, by the coercivity of F, x_k is bounded. Thus $\{x_k\}$ admits a limit point, x_*. Also $\{s_k\}$ is bounded, since

$$\begin{aligned}\|s_k\| = \|J(x_k)^{-1}J(x_k)s_k\| &\leq \|J(x_k)^{-1}\|\|J(x_k)s_k + F(x_k)\| + \|J(x_k)^{-1}\|\|F(x_k)\| \\ &\leq (1+\eta)\|J(x_k)^{-1}\|\|F(x_k)\|. \end{aligned} \tag{15}$$

Then all the hypotheses of Theorem 2.5.1 are verified and so, $\lim_{k\to\infty} \|F(x_k)\| = 0$.

By Assumption 3 and (15), $\lim_{k\to 0} \|s_k\| = 0$. Thus, since $x_{k+1} = x_k + \xi_k s_k$,

$$\lim_{k\to\infty} \|x_{k+1} - x_k\| \leq \lim_{k\to\infty} \|s_k\| = 0.$$

Then, by Lemma 2.5.2, $\lim_{k\to\infty} x_k = x_*$.

By the continuity of F, $\lim_{k\to\infty} F(x_k) = F(x_*) = 0$. ∎

Finally we have the superlinear convergence result stated and proved in the next theorem.

Theorem 2.5.3. *Under the assumptions of Theorem 2.5.2, and assuming also that* $\lim_{k\to\infty} \eta_k = 0$*, then the convergence of* $\{x_k\}$ *to* x_* *is superlinear.*

Proof. By Theorem 2.5.2, $\lim_{k\to\infty} \{x_k\} = x_*$. Due to (15) $\{s_k\}$ is bounded, and by the uniform continuity of $J(x)$ on bounded sets of $I\!R^n$, we have that

$$\|F(x_k + s_k) - [F(x_k) + J(x_k)s_k]\| \leq o(\|s_k\|),\ \forall\, k = 0, 1, 2, \ldots.$$

Hence $\|F(x_k + s_k)\| - \|F(x_k) + J(x_k)s_k\| \leq o(\|s_k\|)$, that is,

$$\|F(x_k + s_k)\| \leq \|F(x_k) + J(x_k)s_k\| + o(\|s_k\|) \leq \eta_k\|F(x_k)\| + o(\|F(x_k)\|,$$

which implies

$$\frac{\|F(x_k + s_k)\|}{\|F(x_k)\|} \leq \eta_k + \frac{o(\|F(x_k)\|)}{\|F(x_k)\|}.$$

Since $\|F(x_k)\| \to 0$, and by assumption $\lim_{k\to\infty} \{\eta_k\} = 0$,

$$\lim_{k\to\infty} \frac{\|F(x_k + s_k)\|}{\|F(x_k)\|} = 0. \tag{16}$$

Therefore, for k large enough,

$$\|F(x_k + s_k)\| \leq (1 - \sigma)\|F(x_k)\|.$$

Then, (14) holds with $\xi_k = 1$. Thus, for k large enough $x_{k+1} = x_k + s_k$. Hence, by (16),

$$\lim_{k\to\infty} \frac{\|F(x_{k+1})\|}{\|F(x_k)\|} = 0. \tag{17}$$

Since $J(x_*)$ is nonsingular, there exist constants $c > 0$ and $C > 0$ such that

$$c\|x - x_*\| \leq \|F(x)\| \leq C\|x - x_*\|,$$

for all x close enough to x_*. Then, by (17)

$$\lim_{k\to\infty} \frac{\|x_{k+1} - x_*\|}{\|x_k - x_*\|} = 0,$$

proving the general superlinear convergence. ∎

It is not difficult to see, by (11), that for the GLT choice, the assumption $\lim_{k\to\infty} \{\eta_k\} = 0$ of Theorem 2.5.3 is satisfied. So, the superlinear convergence is attained with the GLT choice in Algorithm 1.

2 Numerical Experiments

2.1 Introduction

In order to test the new algorithm proposed in this work, we implemented the inexact Newton algorithm, `Algorithm 1`, with four choices for η_k: the choice (11) proposed in this work, `GLT`; the constant choice, `C`, with $\eta_k = 0.01$; and the two choices proposed by Eisenstat and Walker in [13], `EW1`, (6) and `EW2`, (8).

All the tests were performed on a Pentium III - 1.0GHz computer, using the software Matlab 6.1. The linear systems were solved by the Matlab 6.1 `GMRES(m)` package with `m = 30`, allowing a maximum of 100 cycles (3000 iterations). In what follows, we comment on some implementation aspects of the methods:

• Line search procedure: For the parameter σ used in the criterion (12), we took $\sigma = 10^{-4}$, and in Step `4.1` of Algorithm 1 we computed the new step size as $\xi_{new} = 0.5\xi_k$.

• The sequence μ_k: we define $ftip(0) = \|F(x_0)\|$;

$ftip(k) = \min\{\|F(x_k)\|, ftip(k-1)\}$, if k is a multiple of 3 and

$ftip(k) = ftip(k-1)$, otherwise. Then, we set: $\mu_k = ftip(k)/(k+1)^{1.1}$.

• Important values and safeguards for the forcing term: for all the choices for η_k we set the initial value: $\eta_0 = 0.1$. We also took $\eta_k = 0.1$ when $\theta_k > 0$. For the choices `EW1`, `EW2` and `GLT`, $\eta_k = \min\{\eta_k, 0.1\}$ if $k \leq 3$, and $\eta_k = \min\{\eta_k, 0.01\}$ if $k > 3$. We also adopted the safeguard introduced in [23]: if $\eta_k \leq 2\varepsilon$ then we set $\eta_k = 0.8\varepsilon\|F(x_k)\|$, where ε is the precision required for the nonlinear system. For the parameter ρ used in the `GLT` choice we took $\rho = 1.1$. In `EW2`, we followed one of the suggestions of the authors in [13]: $\gamma = 1$ and $\alpha = (1+\sqrt{5})/2$.

• Stopping criterion: the process is finished successfully if $\|F(x_k)\| \leq 10^{-6}$ and $k < 100$.

2.2 The Boundary Value Problems

The performance of these four forcing strategies was tested numerically on a set of boundary value problems with Dirichlet boundary conditions of the same general form: finding a function $u : \Omega = [0,1] \times [0,1] \to I\!R$ such that, for $\lambda \in I\!R$,

$$-\Delta u + h(\lambda, u) = f(s,t), \text{ in } \Omega, \quad u(s,t) = 0 \text{ on } \partial\Omega \tag{18}$$

The real valued function $h(\lambda, u)$, the different values of the parameter λ, and the function f define the different problems. In all the tests we used a grid with 63 inner points in each axis. All the derivatives were approximated using central differences. After discretization, the nonlinear

system of equations had 3969 variables. We now make a brief description of the particular problems solved:

- Bratu problem, [1]: the function h is given by $h(\lambda, u) = -\lambda \exp(u)$ and the function $f(s,t)$ is computed such as $u_*(s,t) = 10st(1-s)(1-t)e^{s^{4.5}}$ [17] is the exact solution of the problem. The values of $\lambda \leq 0$ have a physical meaning in this formulation and are considered as easy problems, [16]. Positive values for λ results in a source of academic difficult nonlinear systems, [5], [16]. We generated 22 instances of this problem by choosing: $\lambda = -1000,\ -500,\ -250,\ -100,\ -50,\ -10,\ 1,\ 3,\ 5,\ 7$ and 10. The initial approximations were: $x_0 = (0,\ 0, \ldots, 0)^T$ and a vector whose components were randomly generated in the interval: $[-5,\ 5]$;

- Convection–diffusion problem, [17]: in this problem, $h(x,\ \lambda) = \lambda u(u_s + u_t)$, where u_s and u_t denote the partial derivatives of the function u with respect to s and t and again the function $f(s,t)$ is defined such that $u^*(s,\ t) = 10st(1-s)(1-t)e^{s^{4.5}}$. This problem is considered difficult in [16], in particular for values of λ greater than 50. We solved a set of 9 problems, generated by choosing $\lambda = 5,\ 10,\ 25,\ 50,\ 75,\ 100,\ 110,\ 125$ and 150 with initial approximation $x_0 = (0,\ 0, \ldots, 0)^T$;

- Briggs problem: a problem proposed by Briggs, Henson and McCormick, [4]. In this case $h(\lambda,\ u) = \lambda u \exp(u)$ and $f(s,t) = ((9\pi^2 + \gamma e^{(s^2-s^3)sin(3\pi t)})(s^2 - s^3) + 6s - 2)sin(3\pi t)$. For the Briggs problem we took $\lambda = 10$, 100 and 1000. As initial approximations we chose constant vectors with value $-2,\ -1,\ 0,\ 1,\ 2$ and 10, and a random vector with entries uniformly distributed in $[-2, 2]$. We solved 21 instances for the Briggs problem.

All in all, 52 problems were considered.

It could be interesting to present the numerical results analyzing the behavior of Algorithm 1 with the four choices for η_k, problem by problem. For lack of space we shall not present the detailed numerical data obtained. These are available in [22]. Here we present just one table to illustrate the reduction in oversolving offered by the GLT choice. Table 1 presents some results obtained for the Bratu problem with $\lambda = 1$, the convection-diffusion problem with $\lambda = 150$, and the Briggs problem with $\lambda = 100$. These are considered hard problems. In this table, column `iterex` represents the total number of external iterations performed and column `iterin`, the number of inner iterations, that is, the number of iterations performed by the linear solver.

The best overall performance was displayed by the GLT choice, with the single exception of the number of outer iterations for the Bratu problem. These are good examples to show the `GLT` choice avoiding oversolving, by analyzing the number of inner iterations. Obviously, the `GLT` choice

was not the best one for all the 52 problems. Nevertheless, Table 1 is typical for most of the hard problems examined.

η_k	Bratu problem		Conv-diff problem		Briggs problem	
	iterex	iterin	iterex	iterin	iterex	iterin
Cte	6	818	47	37973	19	271
EW1	7	556	48	27786	18	245
EW2	4	567	47	28314	18	244
GLT	5	502	46	23482	17	237

Table 1: Numerical performance for the four choices.

A concise and informative comparison of the overall behavior of the 4 choices against all the 52 problems considered is given in Figure 2, using performance profiles. Proposed by Dolan and Moré in 2002, see [11], the performance profile is an extremely useful methodology for standardizing the comparison of algorithms. It considers solvers s, tested problems p, and tested measures m. Several measures can be taken, such as number of iterations and `cpu` time. Let $m_{s,p}$ denote the performance measurement required to solve problem p by solver s. For each problem p and solver s the *performance ratio* $r_{s,p}$ is computed as $r_{s,p} = m_{s,p}/\min_{s\in S}\{m_{s,p}\}$ if the problem p is solved by the solver s; otherwise, $r_{s,p} = r_M$, where r_M is a large enough fixed parameter. Then, for each $s \in S$, the cumulative distribution function $\rho_s : I\!R \to [0,1]$, for performance ratio $r_{s,t}$, is built: $\rho_s(t) = \frac{1}{n_p}\text{size}\{p \in P \mid r_{s,p} \leq t\}$. This function represents the performance of the solver s, it is nondecreasing and piecewise constant. At the analysis of solver s two values give us important information: the *efficiency* of a solver s which is indicated by the percentage of problems solved more quickly ($\rho_s(1)$) and its *robustness* which is represented by the value of $\bar{t} \in [0, r_M]$ for which $\rho_s(\bar{t}) = 1$, with default value $\bar{t} = r_M$. Let $\bar{s}$ be the solver s which maximizes the function $\rho_s(1)$. This solver, $\bar{s}$, solves the largest number of problems at the lowest possible value of m. Similarly, the best solver in terms of robustness will be the solver $\hat{s}$ for which $\overline{t_{\hat{s}}} = \min\{\bar{t}_s, \forall s \in S\}$.

In this work we considered the following four performance measures: the total number of inner iterations, the total number of outer iterations, the total number of number of function evaluations, and the total `cpu` time. Figure 2 shows the performance profile when Algorithm 1 was applied for solving the set of 52 problems with the four choices for the forcing term η_k. At the four diagrams in Figure 2, the constant choice ($\eta_k = 0.01$) is represented by a line (...); the choices `EW1` and `EW2` by lines (− − −) and (. − . − .) respectively. Finally the `GLT` choice proposed in this work is represented by a continuous line.

Considering the number of inner iterations we can observe that choices `EW1`, `EW2` and `GLT` had a similar behavior: `EW1` solved almost 40% of the problems more quickly. Nevertheless the `GLT` choice was the most robust, attaining $\bar{t} \simeq 1.3$. Observe that the `GLT` choice solved almost 90% of the problems with the minimum value of number of outer iterations. Even though `EW1` was the best solver in terms of inner iterations, it had a poor performance in terms of outer iterations, solving only about 50% of the problems with minimum value. Note also that the number of function evaluations follows the same pattern as the one of outer iterations. In terms of `cpu` time `GLT` and `EW2` had a similar behavior. Actually the `GLT` choice was the most robust and efficient for almost all the measures used.

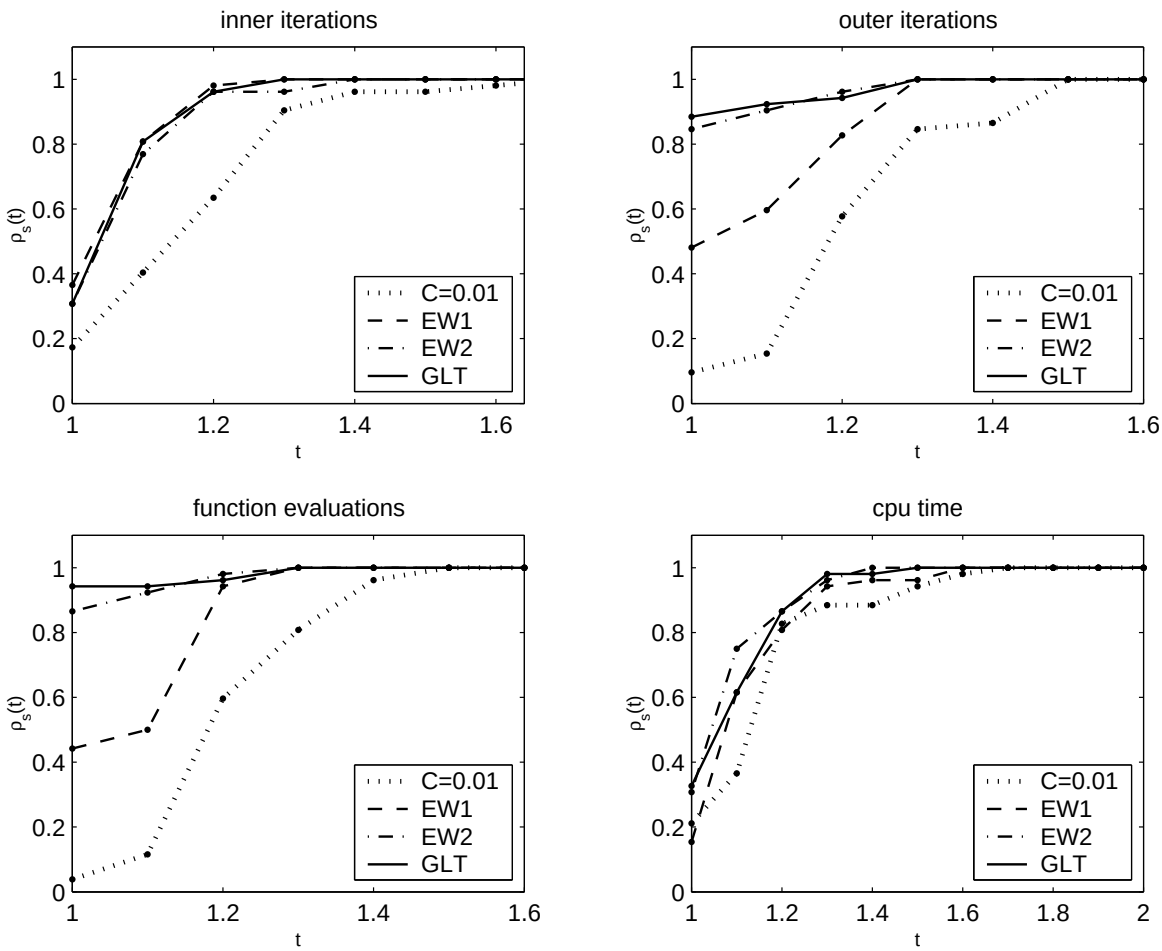

Figure 2: Performance profile using the inner and outer iterations, number of function evaluations and `cpu` time as measures.

3 Conclusions

In this work we proposed a new inexact Newton method with a different choice for the forcing term η_k and with a robust line search strategy, which resulted in an algorithm with a global convergence result. We attribute the robustness of our algorithm to both the `GLT` choice of the forcing term and the inclusion of a good backtracking strategy. However, it is the `GLT` choice for η_k which had the decisive influence on the performance, since, for comparing the different choices, the same backtracking strategy was introduced in the different solvers. The numerical experiments showed that this new algorithm is competitive in terms of the numbers of inner and outer iterations performed, allowing us to conclude that our objectives were obtained: to build an inexact Newton algorithm avoiding the high number of inner iterations without performing a very large number of outer iterations. Moreover, it is important to observe that these objectives were achieved without an increase in the number of function evaluations.

The constant choice has the only advantage of being easy to implement and `EW1` and `EW2` had a very similar and efficient performance. However, `GLT` seems to be superior to the other choices because it is faster in terms of number of outer iterations, number of function evaluations and `CPU` time. This choice can also solve the whole set of problems with the minimum value of t for almost all the measures.

Acknowledgements. The authors are indebted to two anonymous referees for their careful reading of this paper and also for several suggestions.

References

[1] Averick, B. M., Carter, R. G., Moré, J. J. and Xue, G.-L. (1992). "The Minpack-2 Test Problem Collection," *Preprint MCS-P153-0692, Mathematics and Computer Science Division, Argonne National Laboratory.*

[2] Birgin, E. G., Krejić, N. and Martínez, J. M. (2003). "Globally Convergent Inexact Quasi-Newton Methods for Solving Nonlinear Systems," *Numerical Algorithms*, 32, 249–260.

[3] N. Bleistein, *Mathematical Methods for Wave Phenomena*, Academic Press, 1984.

[4] Briggs, W. L., Henson, V. E. and McCormick, S. F. (2000). *A Multigrid Tutorial*, 2nd. edition, SIAM.

[5] Brown, P. N. and Saad, Y. (1990). "Hybrid Krylov Methods for Nonlinear Systems of Equations." *SIAM J. Sci. Stat. Comput.* 11, No. 3, 450–481.

[6] Broyden, C. G. (1965). "A Class of Methods for Solving Sparse Nonlinear Systems," *Math. Comput.* 25, 285–294.

[7] Broyden, C. G., Dennis Jr., J. E. and Moré, J. J. (1973). "On the Local and Superlinear Convergence of Quasi–Newton Methods." *J. Inst. Math. Appl.* 12, 223-245.

[8] Dembo, R. S., Eisenstat, S. C. and Steihaug, T. (1982). "Inexact Newton Methods," *SIAM J. Numer. Anal.* 19, No. 2, 401-408.

[9] Dennis, Jr., J. E. and Schnabel, R. B. (1996). *Numerical Methods for Unconstrained Optimization and Nonlinear Equations.* SIAM Classics in Applied Mathematics.

[10] Diniz-Ehrhardt, M. A., Gomes-Ruggiero, M. A., Lopes, V. L. R. and Martínez, J. M. (2003). "Discrete Newton's Method with Local Variations for Solving Large–Scale Nonlinear Systems," *Optimization* 52, Nos.4–5, 417–440.

[11] Dolan, E. D. and Moré, J. J. (2002). "Benchmarking Optimization Software with Performance Profiles," *Math. Program.* Ser. A 91, 201-213.

[12] Eisenstat, S. C. and Walker, H. F. (1994). "Globally Convergent Inexact Newton Methods," *SIAM J. Optimization* 4, No. 2, 393-422.

[13] Eisenstat, S. C. and Walker, H. F. (1996). "Choosing the Forcing Terms in Inexact Newton Method," *SIAM J. Sci. Comput.* 17, No. 1, 16-32.

[14] Gomes–Ruggiero, M. A. and Martínez, J. M. (1992). "The Column–updating Method for Solving Nonlinear Equations in Hilbert Space," *Mathematical Modeling and Numerical Analysis* 26, No. 2, 309–330.

[15] Gomes–Ruggiero, M. A., Martínez, J. M. and Moretti, A. C. (1992). "Comparing Algorithms for Solving Sparse Nonlinear Systems of Equations," *SIAM Journal Scientific and Statistical Computing* 13, No.2, 459-483.

[16] Gomes–Ruggiero, M. A., Kozakevich, D. N. and Martínez. (1996). "A Numerical Study on Large–scale Nonlinear Solver," *Computers and Mathematics with Applications* 32, No. 3, 1–13.

[17] Kelley, C. T. (1995). *Iterative Methods for Linear and Nonlinear Equations.* SIAM.

[18] Li, D-H. and Fukushima, M. (2000). "Derivative–free Line Search and Global Convergence of Broyden–like Method for Nonlinear Equations," *Optimization Methods and Software* 13, 181-201.

[19] Lopes, V. L. R. and Martínez, J. M. (1995). "Convergence Properties of the Inverse Column–updating Method," *Optimization Methods and Software* 6, 127-144.

[20] Martínez, J. M. (1984). "A Quasi–Newton Method with Modification of One Column per Iteration," *Computing* 33, 353-362.

[21] Martínez, J. M. and Zambaldi, M. C. (1992). "An Inverse Column–updating Method for Solving Large–scale Nonlinear Systems of Equations." *Optim. Methods and Software*, 1, 129-140.

[22] Toledo-Benavides, J. V. (2005). "Um método Newton–GMRES globalmente convergente com um nova escolha para o termo forçante e algumas estratégias para melhorar o desempenho de GMRES(m)", PhD Thesis, Imecc–Unicamp, T/Unicamp T575m.

[23] Pernice, M., and Walker, H. (1998). "NITSOL: a Newton Iterative Solver for Nonlinear Systems," *SIAM J. Sci. Comput.* 19, No. 1, 302-318.

[24] Saad, Y. and Schultz, M. H. (1986). "GMRES: a Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems," *SIAM J. Sci. Stat. Comput.* 7, No. 3.