Iterative Methods for Sparse Linear Systems

Luca Bergamaschi

*=[berga@dmsa.unipd.it - http://www.dmsa.unipd.it/~berga.

Department of Mathematical Methods and Models
for Scientific Applications
University of Padova
Summary

- Nonlinear systems of equations. A few examples
- Newton’s method for $f(x) = 0$.
- Newton’s method for systems.
- Local convergence. Exit tests.
- Global convergence. Backtracking. Line search algorithms
- Two computationally useful variants: the Inexact Newton method and the Quasi Newton method.
Nonlinear Systems of Equations

A few examples
Nonlinear Systems of Equations

A few examples

Intersection of curves in $\mathbb{R}^n$.
For example find the intersection between a circle and a hyperbole

\[
\begin{align*}
    x^2 + y^2 &= 4 \\
    xy &= 1
\end{align*}
\]
Nonlinear Systems of Equations

A few examples

- Intersection of curves in $\mathbb{R}^n$. For example find the intersection between a circle and a hyperbole

\[
\begin{align*}
  x^2 + y^2 &= 4 \\
  xy &= 1
\end{align*}
\]

- Flow equation in porous media (Richards’ equation).

\[
\frac{\partial \psi}{\partial t} - \nabla \cdot \left( K(\psi) \nabla \psi \right) = f
\] (1)
Nonlinear Systems of Equations

A few examples

- Intersection of curves in $\mathbb{R}^n$. For example find the intersection between a circle and a hyperbola

  \[
  \begin{cases}
  x^2 + y^2 = 4 \\
  xy = 1
  \end{cases}
  \]

- Flow equation in porous media (Richards’ equation).

  \[
  \frac{\partial \psi}{\partial t} - \vec{\nabla} \cdot (K(\psi) \vec{\nabla} \psi) = f
  \]

- Unconstrained optimization

  \[\min G(x) \implies \text{solve } G'(x) = 0\]
Newton’s method

Given a function \( f \in C^1 \), we aim at finding one solution of the equation

\[
f(x) = 0
\]

Given \( x_k \), an approximation to the solution \( \xi \), we correct it to find \( x_{k+1} = x_k + s \)

We impose the condition \( f(x_{k+1}) = 0 \) and expand \( f(x_{k+1}) \) in Taylor series neglecting the terms of order greater or equal than 2.

\[
0 = f(x_{k+1}) = f(x_k) + sf'(x_k)
\]

from which

\[
s = -\frac{f(x_k)}{f'(x_k)}.
\]

The Newton’s method can therefore be written as

\[
x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}
\]
Newton’s method for system of nonlinear equations

Let us now solve the following nonlinear system

\[
\begin{align*}
F_1(x_1, x_2, \cdots, x_n) &= 0 \\
F_2(x_1, x_2, \cdots, x_n) &= 0 \\
\vdots &= 0 \\
F_n(x_1, x_2, \cdots, x_n) &= 0
\end{align*}
\]

more synthetically

\[ F(x) = 0 \]

where

\[
F = \begin{pmatrix} F_1 \\ F_2 \\ \vdots \\ F_n \end{pmatrix} \quad x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}
\]

Let us assume that \( F \) be differentiable in an open subset \( \Omega \) of \( \mathbb{R}^n \).
Newton’s method for system of nonlinear equations

As in the scalar case, we try to correct an approximation \( x_k \) as \( x_{k+1} = x_k + s \).

Let us impose \( F(x_{k+1}) = 0 \) and as before expand in Taylor series the function \( F(x_{k+1}) \).

\[
0 = F(x_{k+1}) = F(x_k) + F'(x_k)s
\]

where \( F'(x_k) \) is the Jacobian of system (7) evaluated in \( x_k \) i.e.

\[
(F'(x))_{ij} = \frac{\partial F_i}{\partial x_j}(x)
\]

As before the problem is to compute the increment \( s \) which is now a vector of \( n \) components.

\[
s = - \left( F'(x_k) \right)^{-1} F(x_k)
\]

The \( k \)th iteration of the Newton’s method is thus written as

\[
x_{k+1} = x_k - \left( F'(x_k) \right)^{-1} F(x_k)
\]
Newton’s method for system of nonlinear equations

Some observations:

- The Jacobian matrix $F'(x_k)$ must be invertible.
- Local convergence of the Newton’s method can be proved provided that the initial approximation $x_0$ is sufficiently close to the solution.
- Computation of $x_{k+1}$ starting from $x_k$ requires inversion of (possibly large and sparse) Jacobian matrix. This operation is inefficient as known. In practice vector $s$ is evaluated by solving the following linear system

$$F'(x_k)s = -F(x_k)$$

- $F'$ is often non symmetric, so GMRES iterative method is suggested for the solution of the Newton system
Let us write a first version of the Algorithm, by taking into account previous comments.

**Algorithm Newton 1**

Given an initial approximation \( x_0 \), \( k := 0 \).

repeat until convergence

- solve: \( F'(x_k)s = -F(x_k) \)
- \( x_{k+1} := x_k + s \)
- \( k := k + 1 \)
Convergence of Newton’s method

Standard Assumptions

- Equation $F(x) = 0$ has one solution which we call $x^*$.
- Function $F'$ is Lipschitz continuous: there exists a real number $\gamma$ such that

$$
\|F'(y) - F'(x)\| \leq \gamma \|y - x\|
$$

- $F'(x^*)$ is invertible.

Notazioni. Let us define:

- the error at iteration $k$: $e_k = x_k - x^*$

Theorem 1

Let the standard assumption hold, then for every $x \in \Omega$

$$
F(x) - F(x^*) = \int_{0}^{1} F'(x^* + t(x - x^*))(x - x^*) dt
$$

Proof

It is the fundamental theorem of calculus
Convergence of Newton’s method

Lemma 1 (Banach Lemma)
If $A, B$ are matrices such that $\|I - BA\| < 1$ then

$$\|A^{-1}\| \leq \frac{\|B\|}{1 - \|I - BA\|}$$

Lemma 2
Let the standard assumption hold. Then there is $\delta$ such that for all $x$ satisfying $\|x - x^\ast\| < \delta$:

$$\|F'(x)^{-1}\| \leq 2\|F'(x^\ast)^{-1}\|$$

Proof

$$\|I - F'(x^\ast)^{-1}F'(x)\| = \|F'(x^\ast)^{-1}(F'(x^\ast) - F'(x))\| \leq \gamma\|F'(x^\ast)^{-1}\|\|x_k - x^\ast\|$$

$$\leq \gamma\delta\|F'(x^\ast)^{-1}\|$$

Choose $\delta < \frac{1}{2\gamma\|F'(x^\ast)^{-1}\|}$ so that $\|I - F'(x^\ast)^{-1}F'(x)\| < 1/2$ and apply the Banach Lemma with $A = F'(x)$ and $B = F'(x^\ast)^{-1}$. 

Iterative Methods – p.10/29
Convergence of Newton’s method

**Theorem 2**
There exists $\delta > 0$ such that if $\|e_0\| < \delta$ then

$$\|e_{k+1}\| \leq K\|e_k\|^2 \quad \text{with} \quad K = \gamma \| (F'(x^*))^{-1} \|
$$

**Proof**
By Theorem 1

$$e_{k+1} = e_k - F'(x_k)^{-1} F(x_k) = F'(x_k)^{-1} \int_0^1 \left( F'(x_k) - F'(x^* + t e_k) \right) e_k dt$$

Now by the standard assumptions

$$\|e_{k+1}\| \leq \|F'(x_k)^{-1}\| \int_0^1 \gamma (1 - t) \|e_k\|^2 dt$$

$$\leq \frac{1}{2} \gamma \|F'(x_k)^{-1}\| \|e_k\|^2$$

$$\leq \text{(by Lemma 2)} \gamma \|F'(x^*)^{-1}\| \|e_k\|^2 = K\|e_k\|^2$$
Convergence of Newton’s method

- Convergence is only local \((\|e_0\| < \delta)\).
- As in the scalar case convergence is quadratic

Exit test

When to stop the algorithm?
Theoretically we should stop when \(\|e_{k+1}\| < \varepsilon\) (absolute error) or when \(\|e_{k+1}\| < \varepsilon\|e_0\|\) (relative error); where \(\varepsilon\) is a fixed tolerance. As usual the error vector is not known as the exact solution \(x^*\) is not known.

- Exit test on the (relative) residual. Stop when

\[
\frac{\|F(x_k)\|}{\|F(x_0)\|} < \varepsilon
\]

- Test on the difference. Stop when

\[
\|s\| = \|x_{k+1} - x_k\| < \varepsilon
\]
Exit test

Motivations

- Test on the residual. It can be shown that for $\delta$ sufficiently small holds:

$$\frac{1}{4\kappa} \frac{\|e_k\|}{\|e_0\|} \leq \frac{\|F(x_k)\|}{\|F(x_0)\|} \leq 4\kappa \frac{\|e_k\|}{\|e_0\|}$$

where $\kappa = \|F'(x^*)\| \|(F'(x^*))^{-1}\|$ is the condition number of $F'(x^*)$. If $F'(x^*)$ is well conditioned ($\kappa \approx 1$), the test on the residual is similar to the test on the relative error.

- Test on the difference

$$x_{k+1} - x_k = x_{k+1} - x^* + x^* - x_k = e_{k+1} - e_k$$

$$\|x_{k+1} - x_k\| = \|e_k\| + O(\|e_k\|^2)$$

The difference at step $k+1$ has the same order of magnitude as the error at previous step $k$. (Exit on the difference is a pessimistic test).
Example

\[
\begin{align*}
\begin{cases}
  x^2 + y^2 - 4 &= 0 \\
  xy - 1 &= 0
\end{cases}
\end{align*}
\]

\[\mathbf{x}^{(0)} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}\]

<table>
<thead>
<tr>
<th>$k$</th>
<th>$x_1^{(k)}$</th>
<th>$x_2^{(k)}$</th>
<th>[|e^{(k)}|]</th>
<th>[|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}|]</th>
<th>[|e^{(k+1)}|]</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.000000000</td>
<td>1.000000000</td>
<td>0.107 × 10^{+01}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1.000000000</td>
<td>2.500000000</td>
<td>0.745 × 10^{+00}</td>
<td>0.180 × 10^{+01}</td>
<td>0.655899</td>
</tr>
<tr>
<td>2</td>
<td>0.595238095</td>
<td>2.011904761</td>
<td>0.111 × 10^{+00}</td>
<td>0.634 × 10^{+00}</td>
<td>0.200716</td>
</tr>
<tr>
<td>3</td>
<td>0.520020336</td>
<td>1.934236023</td>
<td>0.337 × 10^{−02}</td>
<td>0.108 × 10^{−00}</td>
<td>0.271153</td>
</tr>
<tr>
<td>4</td>
<td>0.517640404</td>
<td>1.931853966</td>
<td>0.327 × 10^{−05}</td>
<td>0.337 × 10^{−02}</td>
<td>0.288114</td>
</tr>
<tr>
<td>5</td>
<td>0.517638090</td>
<td>1.931851652</td>
<td>0.309 × 10^{−11}</td>
<td>0.327 × 10^{−05}</td>
<td>0.288656</td>
</tr>
</tbody>
</table>

\[\|\mathbf{F}(\mathbf{x}^{(0)})\| = 3.16\]  \[\|\mathbf{F}(\mathbf{x}^{(1)})\| = 3.58\]
Example
Global Convergence

- Convergence of Newton’s method not guaranteed. Frequently Newton’s step moves away from the solution.
- To avoid divergence we accept Newton’s step if the following condition holds: \( \| F(x_{k+1}) \| < \| F(x_k) \| \)
- If the above condition is not satisfied, then the Newton step is reduced \( \Rightarrow \) “backtracking” or “linesearch”.

Algorithm: Newton 2.

Given an initial approximation \( x_0 \), \( k := 0 \).

repeat until convergence

- solve: \( F'(x_k)s = -F(x_k) \)
- \( x_t := x_k + s \)
- if \( \| F(x_t) \| < \| F(x_k) \| \) then \( x_{k+1} := x_t \)
- else \( s := s/2 \), go to (●)
- \( k := k + 1 \)
Example

Newton con backtracking

\[
\begin{align*}
\begin{cases}
x^2 + y^2 - 4 &= 0 \\
xy - 1 &= 0
\end{cases} \\
x^{(0)} &= \begin{pmatrix} 0 \\ 1 \end{pmatrix}
\end{align*}
\]

<table>
<thead>
<tr>
<th>(k)</th>
<th>(x_1^{(k)})</th>
<th>(x_2^{(k)})</th>
<th>(|e^{(k)}|)</th>
<th>(x^{(k+1)} - x^{(k)})</th>
<th>(|e^{(k+1)}| / |e^{(k)}|^2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.000000000</td>
<td>1.000000000</td>
<td>0.106597 \times 10^{+01}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.500000000</td>
<td>1.750000000</td>
<td>0.182705 \times 10^{+00}</td>
<td>0.901388E+01</td>
<td>0.160790</td>
</tr>
</tbody>
</table>

\(\|F(x^{(0)})\| = 3.16\) \quad \(\|F(x^{(1)})\| = 0.699\)
Inexact Newton Methods

- Idea: try to avoid oversolving the linear systems at every Newton iteration.
- Example. Discretized Richards’ equation (steady state)

\[ A(\psi)\psi = b(\psi), \quad F(x) = A(x)x - b(x), \quad F' = A + \frac{\partial(A)}{\partial x}x \]

- \( F' \) has the same size and sparsity pattern as \( A \).
- A single iteration with the Newton’s method:
  - solve: \( F'(x_k)s = -F(x_k) \)
  - \( x_{k+1} := x_k + s \)
  - \( k := k + 1 \)
- Solve the linear system with an iterative method of choice with variable tolerance. Formally:

\[ \|F'(x_k)s + F(x_k)\| \leq \eta_k \|F(x_k)\| \]
Inexact Newton Methods

Convergence results:

- If \( \eta_k \to 0 \) then convergence of the Inexact Newton Methods is \textit{superlinear}.
- If in addition \( \eta_k = O(\|F(x_k)\|) \) then again we obtain \textit{quadratic} convergence.

Practical choices for \( \eta_k \). Fix a maximum tolerance \( \eta_{\text{max}} \).

\[
\eta_k = \begin{cases} 
\min(\eta_{\text{max}}, \|F(x_k)\|) \\
\min(\eta_{\text{max}}, \gamma \frac{\|F(x_k)\|^2}{\|F(x_{k-1})\|^2})
\end{cases}
\]

- Convergence of Newton iterations still very rapid
- Linear system solution very cheap especially at the first Newton steps.
Quasi-Newton Methods

Motivation: Jacobian matrix

- Not always explicitly available (sometimes function $F$ is known as a set of data)
- or
- Differentiation of $F$ may be too costly to be afforded at every Newton iteration

A possible answer to this problem is given by the quasi-Newton methods which compute a sequence of approximate Jacobians possibly starting from the ‘true’ initial Jacobian.

Instead of solving

$$x_{k+1} = x_k - F'(x_k)^{-1}F(x_k)$$

we solve

$$x_{k+1} = x_k - B_k^{-1}F(x_k)$$
Quasi-Newton Methods

Sequence of $B_k$ can be constructed in many ways. The simplest approach is due to Broyden:

$$B_{k+1} = B_k + \frac{(y - B_k s)s}{s^T s}$$

where $y = F(x_{k+1}) - F(x_k)$.

the Broyden update formula satisfies:

1. the secant condition, namely $B_{k+1} s = y$.
2. $B_{k+1}$ is the closest matrix to $B_k$ in the Frobenius norm among all the matrices satisfying the secant condition.

$$B_{k+1} = \arg \min_{B : Bs = y} \|B - B_k\|$$
Convergence Results

**Definition.** A sequence \( x_n \) converges superlinearly to \( x^* \) if there are \( \alpha > 1 \) and \( K > 0 \) such that

\[
\| x_{k+1} - x^* \| \leq K \| x_k - x^* \|^\alpha
\]

Let us now define the error in jacobian approximations:

\[
E_k = B_k - F'(x^*)
\]

The first Theorem states that the difference between the exact and the approximate Jacobian does not grow with the Newton iteration. This property is also called *bounded deterioration*.

**Theorem.**

\[
\| E_{k+1} \| \leq \| E_k \| + \frac{\gamma}{2}(\| e_k \| + \| e_{k+1} \|)
\]
**Convergence Results and implementation**

**Theorem.** Let the standard assumption holds. Then there are $\delta$ and $\delta_B$ such that if $\|e_0\| < \delta$ and $\|E_0\| < \delta_B$ the Broyden sequence exists and $x_n \to x^*$ superlinearly.

This theorem states that we can make $\|E_k\|$ as small as we want by properly choosing the initial vector $x_0$ and the initial Jacobian approximation $B_0$.

If it is the case, the convergence of the iteration remains very fast (superlinear convergence).

**Problem.** How to implement solution of Newton system with $B_k^{-1}$ instead of $J(x_k)^{-1}$? Note that even if $B_0$ is sparse $B_1$ is not.

Careful implementation should avoid inversion of dense matrices.
Sparse implementation of Broyden method

Need to compute $B_k^{-1} F(x_k)$ without

1. Computing $B_k^{-1}$ since we do not want to invert matrices.
2. Computing $B_k$ since it is dense.

First result we will use: the Sherman Morrison formula:

**Theorem 1**

$$(B + uv^T)^{-1} = \left( I - \frac{(B^{-1}u)v^T}{1 + v^T B^{-1}u} \right) B^{-1}$$

In our context we can write $B_{k+1}^{-1}$ in terms of $B_k^{-1}$ as

$$B_{k+1} = B_k + u_k v_k,$$

where we can define among the others

$$u_k = \frac{F(x_{k+1})}{\|s_k\|}, \quad v_k = \frac{s_k}{\|s_k\|},$$

so that
Sparse implementation of Broyden method

\[ B_{k+1}^{-1} = (B_k + u_k v_k^T)^{-1} = \left( I - \frac{(B_k^{-1} u_k)v_k^T}{1 + v_k^T B_k^{-1} u_k} \right) B_k^{-1} \]

\[ = \left( I - w_k v_k^T \right) B_k^{-1} \]

Where we have defined \( w_k = \frac{B_k^{-1} u_k}{1 + v_k^T B_k^{-1} u_k} \).

Now by induction

\[ B_k^{-1} = \left( I - w_{k-1} v_{k-1}^T \right) \left( I - w_{k-2} v_{k-2}^T \right) \cdots \left( I - w_0 v_0^T \right) B_0^{-1} \]
Sparse implementation of Broyden method

Important results: $s_k = -B_k^{-1} F_k$ is accomplished by

1. Solving the system $B_0 z_0 = -F_k$
2. Computing $\alpha_0 = w_0^T z_0$, then $z_1 = z_0 - \alpha_0 w_0$
   Computing $\alpha_1 = w_1^T z_1$, then $z_2 = z_1 - \alpha_1 w_1$
   \[ \vdots \]
   Computing $\alpha_{k-1} = w_{k-1}^T z_{k-1}$, then $z_k = z_{k-1} - \alpha_{k-1} w_{k-1}$

Problem. We do not know how to compute $w_j, j = 1, \ldots, k - 1$.

Let us define $p = \left( I - w_{k-2} v_{k-2}^T \right) \cdots \left( I - w_0 v_0^T \right) F(x_k)$

It follows that

\[ s_k = -B_k^{-1} F_k = -\left( I - w_{k-1} v_{k-1}^T \right) p = w_{k-1} (v_{k-1}^T p) - p \]

\[ B_{k-1}^{-1} u_{k-1} = B_{k-1}^{-1} \frac{F_k}{\|s_{k-1}\|} = \frac{p}{\|s_{k-1}\|} \]

\[ w_{k-1} = \frac{B_{k-1}^{-1} u_{k-1}}{1 + v_{k-1}^T B_{k-1}^{-1} u_{k-1}} = \frac{p}{\|s_{k-1}\| + v_{k-1}^T p} \]
Sparse implementation of Broyden method

Now combining $s_k = w_{k-1}(v_{k-1}^T p) - p$ with $w_{k-1} = \frac{p}{\|s_{k-1}\| + v_{k-1}^T p}$ we obtain

$$\|s_{k-1}\| w_{k-1} = p - w_{k-1} v_{k-1}^T p$$

hence

$$w_{k-1} = \frac{s_k}{\|s_{k-1}\|}$$

Hence $B_k^{-1}$ can be written in terms of sequence $\{s_j\}$ only as

$$B_k^{-1} = \prod_{j=0}^{k-1} \left( I + \frac{s_{j+1}s_{j}^T}{\|s_{j}\|^2} \right)$$

NOTE: We know $s_k$ as a function of $B_k^{-1}$ and $B_k^{-1}$ as a function of $s_k$. 

Iterative Methods – p.27/29
Sparse implementation of Broyden method

let us write $s_k$ as

$$s_k = -B_k^{-1}F_k = - \left( I + \frac{s_k s_{k-1}^T}{\|s_{k-1}\|_2^2} \right) \prod_{j=1}^{k-2} \left( I + \frac{s_{j+1} s_j^T}{\|s_j\|_2^2} \right) F_k$$

$$= - \left( I + \frac{s_k s_{k-1}^T}{\|s_{k-1}\|_2^2} \right) B_{k-1}^{-1} F_k \quad (-4)$$

Finally we solve (4) to obtain

$$s_k = \frac{B_{k-1}^{-1} F_k}{1 + s_{k-1}^T B_{k-1}^{-1} F_k / \|s_{k-1}\|_2^2}$$
Broyden Algorithm (sketch)

**INPUT**: \(x_0, B_0\). Set \(k := 0, x := x_0\).

**First step**: Solve \(B_0s_0 = -F(x_0)\)

**REPEAT** until convergence

- \(x := x + s_k\)
- **Solve** \(B_0z = -F(x)\)
- \(k := k + 1\).
- **FOR** \(j := 1\) **TO** \(k - 1\)
  - \(z := z + \frac{s_{j+1}s_j^T}{\|s_j\|^2}\)
  - \(s_k := \frac{z}{1 + s_{k-1}^T z/\|s_{k-1}\|^2}\)

**END REPEAT**

And this is also **THE END** of the course.