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

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

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

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

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

$$\frac{\partial \psi}{\partial t} - \vec{\nabla} \cdot \left(K(\psi) \vec{\nabla} \psi \right) = f \tag{1}$$

Nonlinear Systems of Equations

A few examples

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

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

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

$$\frac{\partial \psi}{\partial t} - \vec{\nabla} \cdot \left(K(\psi) \vec{\nabla} \psi \right) = f \tag{1}$$

- Unconstrained optimization

$$\min G(\boldsymbol{x}) \quad \Longrightarrow \quad \text{solve } \boldsymbol{G}'(\boldsymbol{x}) = 0$$

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 ξ, 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.

$$0 = f(x_{k+1}) = f(x_k) + s f'(x_k)$$

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

$$\left\{ \begin{array}{lcl} F_1(x_1, x_2, \cdots, x_n) & = & 0 \\ F_2(x_1, x_2, \cdots, x_n) & = & 0 \\ \ldots & = & 0 \\ F_n(x_1, x_2, \cdots, x_n) & = & 0 \end{array} \right. \tag{1}$$

more sinthetically

$$\boldsymbol{F}(\boldsymbol{x}) = 0$$

where

$$\boldsymbol{F} = \begin{pmatrix} F_1 \\ F_2 \\ \ldots \\ F_n \end{pmatrix} \qquad \boldsymbol{x} = \begin{pmatrix} x_1 \\ x_2 \\ \ldots \\ x_n \end{pmatrix}$$

Let us assume that $\boldsymbol{F}$ be differentiable in an open subset Ω of $\mathbb{R}^n$.

Newton's method for system of nonlinear equations

As in the scalar case, we try to correct an approximation $\boldsymbol{x}_k$ as $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \mathbf{s}$.

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

$$0 = \boldsymbol{F}(\boldsymbol{x}_{k+1}) = \boldsymbol{F}(\boldsymbol{x}_k) + F'(\boldsymbol{x}_k)\mathbf{s}$$

where $F'(\boldsymbol{x}_k)$ is the Jacobian of system (7) evaluated in $\boldsymbol{x}_k$ i. e.

$$\left(F'(\boldsymbol{x})\right)_{ij} = \frac{\partial F_i}{\partial x_j}(\boldsymbol{x})$$

As before the problem is to compute the increment $\mathbf{s}$ which is now a vector of n components.

$$\mathbf{s} = -\left(F'(\boldsymbol{x}_k)\right)^{-1}\boldsymbol{F}(\boldsymbol{x}_k)$$

The kth iteration of the Newton's method is thus written as

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \left(F'(\boldsymbol{x}_k)\right)^{-1}\boldsymbol{F}(\boldsymbol{x}_k)$$

Newton's method for system of nonlinear equations

Some observations:

- The Jacobian matrix $F'(\boldsymbol{x}_k)$ must be invertible.
- Local convergence of the Newton's method can be proved provided that the initial approximation $\boldsymbol{x}_0$ is sufficiently close to the solution.
- Computation of $\boldsymbol{x}_{k+1}$ starting from $\boldsymbol{x}_k$ requires inversion of (possibly large and sparse) Jacobian matrix. This operation is inefficient as known. In practice vector $\mathbf{s}$ is evaluated by solving the following linear system

$$F'(\boldsymbol{x}_k)\mathbf{s} = -\boldsymbol{F}(\boldsymbol{x}_k)$$

- F' is often non symmetric, so GMRES iterative method is suggested for the solution of the Newton system

Algorithm

Let us write a first version of the Algorithm, by taking into account previous comments.

Algorithm Newton 1

Given an initial approximation $\boldsymbol{x}_0$, $k := 0$.

repeat until convergence

- solve: $F'(\boldsymbol{x}_k)\mathbf{s} = -\boldsymbol{F}(\boldsymbol{x}_k)$
- $\boldsymbol{x}_{k+1} := \boldsymbol{x}_k + \mathbf{s}$
- $k := k + 1$

Convergence of Newton's method

Standard Assumptions

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

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

- $F'(\boldsymbol{x}^*)$ is invertible.

Notazioni. Let us define:

- the error at iteration k: $\boldsymbol{e}_k = \boldsymbol{x}_k - \boldsymbol{x}^*$

Theorem 1

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

$$\boldsymbol{F}(\boldsymbol{x}) - \boldsymbol{F}(\boldsymbol{x}^*) = \int_0^1 F'(\boldsymbol{x}^* + t(\boldsymbol{x} - \boldsymbol{x}^*))(\boldsymbol{x} - \boldsymbol{x}^*)dt$$

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 δ such that for all $\boldsymbol{x}$ satisfying $\|\boldsymbol{x} - \boldsymbol{x}^*\| < \delta$:

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

Proof

$$\begin{aligned}\|I - F'(\boldsymbol{x}^*)^{-1}F'(\boldsymbol{x})\| &= \|F'(\boldsymbol{x}^*)^{-1}\left(F'(\boldsymbol{x}^*) - F'(\boldsymbol{x})\right) \leq \gamma\|F'(\boldsymbol{x}^*)^{-1}\|\|\boldsymbol{x}_k - \boldsymbol{x}^*\| \\ &\leq \gamma\delta\|F'(\boldsymbol{x}^*)^{-1}\|\end{aligned}$$

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

Convergence of Newton's method

Theorem 2

There exists $\delta > 0$ such that if $\|\boldsymbol{e}_0\| < \delta$ then

$$\|\boldsymbol{e}_{k+1}\| \leq K\|\boldsymbol{e}_k\|^2 \qquad \text{with} \qquad K = \gamma\|(F'(\boldsymbol{x}^*))^{-1}\|$$

Proof

By Theorem 1

$$\boldsymbol{e}_{k+1} = \boldsymbol{e}_k - F'(\boldsymbol{x}_k)^{-1}\boldsymbol{F}(\boldsymbol{x}_k) = F'(\boldsymbol{x}_k)^{-1}\int_0^1 \big(F'(\boldsymbol{x}_k) - F'(\boldsymbol{x}^* + t\boldsymbol{e}_k)\big)\,\boldsymbol{e}_k dt$$

Now by the standard assumptions

$$\begin{aligned}
\|\boldsymbol{e}_{k+1}\| &\leq \|F'(\boldsymbol{x}_k)^{-1}\|\int_0^1 \gamma(1-t)\|\boldsymbol{e}_k\|^2 dt \\
&\leq \frac{1}{2}\gamma\|F'(\boldsymbol{x}_k)^{-1}\|\|\boldsymbol{e}_k\|^2 \\
&\leq \text{(by Lemma 2)} \quad \gamma\|F'(\boldsymbol{x}^*)^{-1}\|\|\boldsymbol{e}_k\|^2 = K\|\boldsymbol{e}_k\|^2
\end{aligned}$$

Convergence of Newton's method

- Convergence is only local $(\|\boldsymbol{e}_0\| < \delta)$.
- As in the scalar case convergence is quadratic

Exit test

When to stop the algorithm?
Theoretically we should stop when $\|\boldsymbol{e}_{k+1}\| < \varepsilon$ (absolute error) or when $\|\boldsymbol{e}_{k+1}\| < \varepsilon\|\boldsymbol{e}_0\|$ (relative error); where ε is a fixed tolerance. As usual the error vector is not known as the exact solution $\boldsymbol{x}^*$ is not known.

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

$$\frac{\|\boldsymbol{F}(\boldsymbol{x}_k)\|}{\|\boldsymbol{F}(\boldsymbol{x}_0)\|} < \varepsilon$$

- Test on the difference. Stop when

$$\|\mathbf{s}\| = \|\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\| < \varepsilon$$

Exit test

Motivations

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

$$\frac{1}{4\kappa}\frac{\|\boldsymbol{e}_k\|}{\|\boldsymbol{e}_0\|} \leq \frac{\|\boldsymbol{F}(\boldsymbol{x}_k)\|}{\|\boldsymbol{F}(\boldsymbol{x}_0)\|} \leq 4\kappa\frac{\|\boldsymbol{e}_k\|}{\|\boldsymbol{e}_0\|}$$

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

- Test on the difference

$$\begin{aligned}\boldsymbol{x}_{k+1} - \boldsymbol{x}_k &= \boldsymbol{x}_{k+1} - \boldsymbol{x}^* + \boldsymbol{x}^* - \boldsymbol{x}_k = \boldsymbol{e}_{k+1} - \boldsymbol{e}_k \\ \|\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\| &= \|\boldsymbol{e}_k\| + O(\|\boldsymbol{e}_k\|^2)\end{aligned}$$

  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).

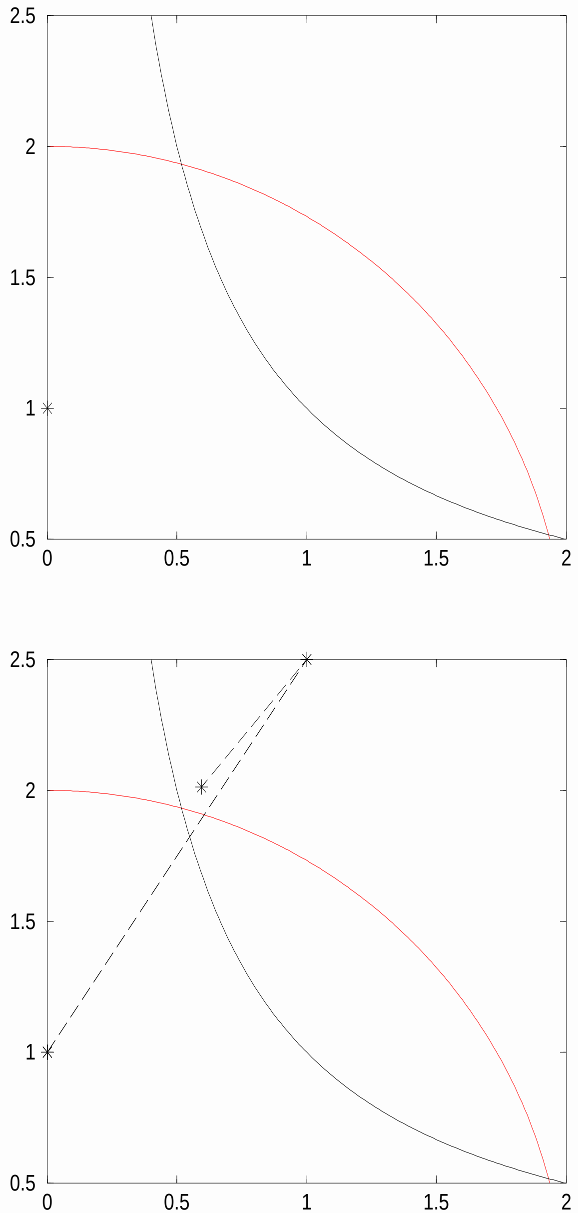
Example

$$\begin{cases} x^2 + y^2 - 4 = 0 \\ xy - 1 = 0 \end{cases} \qquad \boldsymbol{x}^{(0)} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

k	$x_1^{(k)}$	$x_2^{(k)}$	$\lVert e^{(k)} \rVert$	$\lVert \boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)} \rVert$	$\dfrac{\lVert e^{(k+1)} \rVert}{\lVert e^{(k)} \rVert^2}$
0	0.000000000	1.000000000	$0.107\times10^{+01}$		
1	1.000000000	2.500000000	$0.745\times10^{+00}$	$0.180\times10^{+01}$	0.655899
2	0.595238095	2.011904761	$0.111\times10^{+00}$	$0.634\times10^{+00}$	0.200716
3	0.520020336	1.934236023	0.337×10^{-02}	0.108×10^{-00}	0.271153
4	0.517640404	1.931853966	0.327×10^{-05}	0.337×10^{-02}	0.288114
5	0.517638090	1.931851652	0.309×10^{-11}	0.327×10^{-05}	0.288656

$$\lVert \boldsymbol{F}(\boldsymbol{x}^{(0)}) \rVert = 3.16 \qquad \lVert \boldsymbol{F}(\boldsymbol{x}^{(1)}) \rVert = 3.58$$

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: $\|\boldsymbol{F}(\boldsymbol{x}_{k+1})\| < \|\boldsymbol{F}(\boldsymbol{x}_k)\|$
- If the above condition is not satisfied, then the Newton step is reduced $\Longrightarrow$ "backtracking" or "linesearch".

Algorithm: Newton 2.

Given an initial approximation $\boldsymbol{x}_0$, $k := 0$.

repeat until convergence

- solve: $F'(\boldsymbol{x}_k)\mathbf{s} = -\boldsymbol{F}(\boldsymbol{x}_k)$

- ● $\boldsymbol{x}_t := \boldsymbol{x}_k + \mathbf{s}$
- if $\|F(\boldsymbol{x}_t)\| < \|\boldsymbol{F}(\boldsymbol{x}_k)\|$ then $\boldsymbol{x}_{k+1} := \boldsymbol{x}_t$
- else $\mathbf{s} := \mathbf{s}/2$, go to (●)
- $k := k + 1$

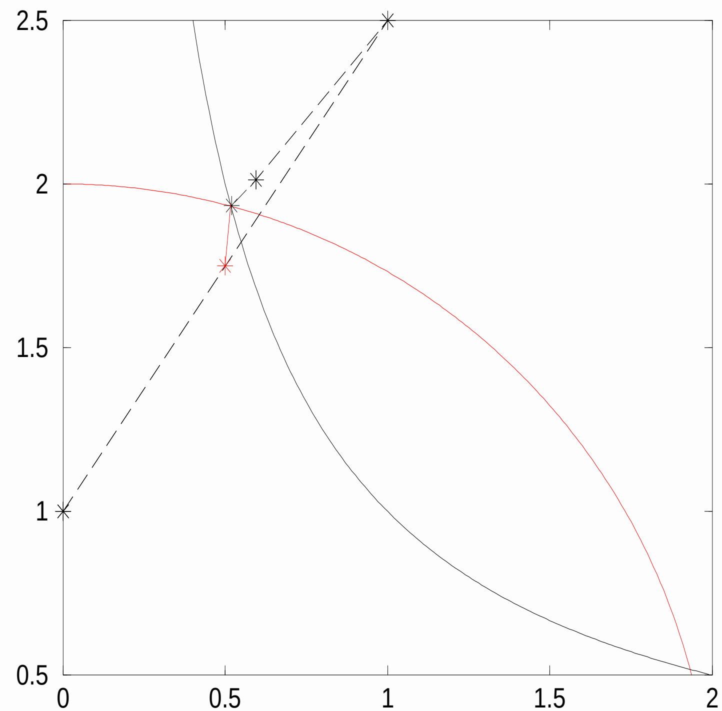
Example

Newton con backtracking

$$\begin{cases} x^2 + y^2 - 4 = 0 \\ xy - 1 = 0 \end{cases} \qquad \boldsymbol{x}^{(0)} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

k	$x_1^{(k)}$	$x_2^{(k)}$	$\Vert e^{(k)}\Vert$	$\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)}$	$\frac{\Vert e^{(k+1)}\Vert}{\Vert e^{(k)}\Vert^2}$
0	0.000000000	1.000000000	$0.106597\times10^{+01}$		
1	0.500000000	1.750000000	$0.182705\times10^{+00}$	0.901388E+01	0.160790

$$\Vert\boldsymbol{F}(\boldsymbol{x}^{(0)}))\Vert = 3.16 \qquad \Vert\boldsymbol{F}(\boldsymbol{x}^{(1)})\Vert = 0.699$$



Inexact Newton Methods

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

$$A(\psi)\psi = \boldsymbol{b}(\psi), \qquad \boldsymbol{F}(\boldsymbol{x}) = A(\boldsymbol{x})\boldsymbol{x} - \boldsymbol{b}(\boldsymbol{x}), \qquad F' = A + \frac{\partial(A)}{\partial \boldsymbol{x}}\boldsymbol{x}$$

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

$$\|F'(\boldsymbol{x}_k)\mathbf{s} + \boldsymbol{F}(\boldsymbol{x}_k)\| \leq \eta_k \|\boldsymbol{F}(\boldsymbol{x}_k)\|$$

Inexact Newton Methods

Convergence results:

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

Practical choices for η_k. Fix a maximum tolerance $\eta_{\max}$.

$$\eta_k = \begin{cases} \min(\eta_{\max}, \|\boldsymbol{F}(\boldsymbol{x}_k)\|) \\ \min(\eta_{\max}, \gamma \frac{\|\boldsymbol{F}(\boldsymbol{x}_k)\|^2}{\|\boldsymbol{F}(\boldsymbol{x}_{k-1})\|^2}) \end{cases}$$

- 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

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \boldsymbol{F}'(\boldsymbol{x}_k)^{-1}\boldsymbol{F}(\boldsymbol{x}_k)$$

we solve

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - {B_k}^{-1}\boldsymbol{F}(\boldsymbol{x}_k)$$

Quasi-Newton Methods

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

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

$$B_k + \frac{\boldsymbol{F}(\boldsymbol{x}_{k+1})\boldsymbol{s}}{\boldsymbol{s}^T \boldsymbol{s}}$$

where $\boldsymbol{y} = \boldsymbol{F}(\boldsymbol{x}_{k+1}) - \boldsymbol{F}(\boldsymbol{x}_k)$.
the Broyden upate formula satisfies:

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

$$B_{k+1} = \underset{B \,:\, B\boldsymbol{s} = \boldsymbol{y}}{\text{argmin}} \|B - B_k\|$$

Convergence Results

Definition . A sequence $\boldsymbol{x}_n$ converges superlinearly to $\boldsymbol{x}^*$ if there are $\alpha > 1$ and $K > 0$ such that

$$\|\boldsymbol{x}_{k+1} - \boldsymbol{x}^*\| \leq K \|\boldsymbol{x}_k - \boldsymbol{x}^*\|^\alpha$$

Let us now define the error in jacobian approximations:

$$E_k = B_k - F'(\boldsymbol{x}^*)$$

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.

$$\|E_{k+1}\| \leq \|E_k\| + \frac{\gamma}{2}(\|\boldsymbol{e}_k\| + \|\boldsymbol{e}_{k+1}\|)$$

Convergence Results and implementation

Theorem.
Let the standard assumption holds. Then there are δ and δ_B such that if $\|\boldsymbol{e}_0\| < \delta$ and $\|E_0\| < \delta_B$ the Broyden sequence exists and $\boldsymbol{x}_n \to \boldsymbol{x}^*$ superlinearly.

This theorem states that we can make $\|E_k\|$ as small as we want by properly choosing the initial vector $\boldsymbol{x}_0$ 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 $J(\boldsymbol{x}_k)^{-1}$?
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 ${B_k}^{-1}\boldsymbol{F}(\boldsymbol{x}_k)$ 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

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

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

$$B_{k+1} = B_k + \boldsymbol{u}_k \boldsymbol{v}_k,$$

where we can define among the others

$$\boldsymbol{u}_k = \frac{\boldsymbol{F}(\boldsymbol{x}_{k+1})}{\|\boldsymbol{s}_k\|}, \qquad \boldsymbol{v}_k = \frac{\boldsymbol{s}_k}{\|\boldsymbol{s}_k\|}, \qquad \text{so that}$$

Sparse implementation of Broyden method

$$
\begin{aligned}
B_{k+1}^{-1} &= (B_k + \boldsymbol{u}_k \boldsymbol{v}_k^T)^{-1} = \left(I - \frac{(B_k{}^{-1} \boldsymbol{u}_k) \boldsymbol{v}_k^T}{1 + \boldsymbol{v}_k^T B_k{}^{-1} \boldsymbol{u}_k} \right) B_k{}^{-1} \\
&= \left(I - \boldsymbol{w}_k \boldsymbol{v}_k^T \right) B_k{}^{-1}
\end{aligned}
$$

Where we have defined $\boldsymbol{w}_k = \dfrac{B_k{}^{-1} \boldsymbol{u}_k}{1 + \boldsymbol{v}_k^T B_k{}^{-1} \boldsymbol{u}_k}$.

Now by induction

$$
B_k{}^{-1} = \left(I - \boldsymbol{w}_{k-1} \boldsymbol{v}_{k-1}^T \right) \left(I - \boldsymbol{w}_{k-2} \boldsymbol{v}_{k-2}^T \right) \cdots \left(I - \boldsymbol{w}_0 \boldsymbol{v}_0^T \right) B_0^{-1}
$$

Sparse implementation of Broyden method

Important results: $\boldsymbol{s}_k = -{B_k}^{-1}\boldsymbol{F}_k$ is accomplished by

1. Solving the system $B_0\boldsymbol{z}_0 = -\boldsymbol{F}_k$
2. Computing $\alpha_0 = \boldsymbol{w}_0^T\boldsymbol{z}_0$, then $\boldsymbol{z}_1 = \boldsymbol{z}_0 - \alpha_0\boldsymbol{w}_0$
   Computing $\alpha_1 = \boldsymbol{w}_1^T\boldsymbol{z}_1$, then $\boldsymbol{z}_2 = \boldsymbol{z}_1 - \alpha_1\boldsymbol{w}_1$
   $\cdots$
   Computing $\alpha_{k-1} = \boldsymbol{w}_{k-1}^T\boldsymbol{z}_{k-1}$, then $\boldsymbol{z}_k = \boldsymbol{z}_{k-1} - \alpha_{k-1}\boldsymbol{w}_{k-1}$

Problem. We do not know how to compute $\boldsymbol{w}_j, j = 1, \cdots, k-1$.

Let us define $\boldsymbol{p} = \left(I - \boldsymbol{w}_{k-2}\boldsymbol{v}_{k-2}^T\right) \cdots \left(I - \boldsymbol{w}_0\boldsymbol{v}_0^T\right)\boldsymbol{F}(\boldsymbol{x}_k)$
It follows that

$$
\begin{aligned}
\boldsymbol{s}_k &= -{B_k}^{-1}\boldsymbol{F}_k = -\left(I - \boldsymbol{w}_{k-1}\boldsymbol{v}_{k-1}^T\right)\boldsymbol{p} = \boldsymbol{w}_{k-1}(\boldsymbol{v}_{k-1}^T\boldsymbol{p}) - \boldsymbol{p} \\
B_{k-1}^{-1}\boldsymbol{u}_{k-1} &= B_{k-1}^{-1}\frac{\boldsymbol{F}_k}{\|\boldsymbol{s}_{k-1}\|} = \frac{\boldsymbol{p}}{\|\boldsymbol{s}_{k-1}\|} \\
\boldsymbol{w}_{k-1} &= \frac{B_{k-1}^{-1}\boldsymbol{u}_{k-1}}{1 + \boldsymbol{v}_{k-1}^T B_{k-1}^{-1}\boldsymbol{u}_{k-1}} = \frac{\boldsymbol{p}}{\|\boldsymbol{s}_{k-1}\| + \boldsymbol{v}_{k-1}^T\boldsymbol{p}}
\end{aligned}
$$

Sparse implementation of Broyden method

Now combining $\boldsymbol{s}_k = \boldsymbol{w}_{k-1}(\boldsymbol{v}_{k-1}^T \boldsymbol{p}) - \boldsymbol{p}$ with $\boldsymbol{w}_{k-1} = \dfrac{\boldsymbol{p}}{\|\boldsymbol{s}_{k-1}\| + \boldsymbol{v}_{k-1}^T \boldsymbol{p}}$ we obtain

$$\|\boldsymbol{s}_{k-1}\|\boldsymbol{w}_{k-1} = \boldsymbol{p} - \boldsymbol{w}_{k-1}\boldsymbol{v}_{k-1}^T \boldsymbol{p}$$

hence

$$\boldsymbol{w}_{k-1} = \frac{\boldsymbol{s}_k}{\|\boldsymbol{s}_{k-1}\|}$$

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

$${B_k}^{-1} = \prod_{j=0}^{k-1} \left(I + \frac{\boldsymbol{s}_{j+1}\boldsymbol{s}_j^T}{\|\boldsymbol{s}_j\|_2^2} \right)$$

NOTE: We know $\boldsymbol{s}_k$ as a function of ${B_k}^{-1}$ and ${B_k}^{-1}$ as a function of $\boldsymbol{s}_k$.

Sparse implementation of Broyden method

let us write $\boldsymbol{s}_k$ as

$$\boldsymbol{s}_k = -{B_k}^{-1}\boldsymbol{F}_k = -\left(I + \frac{\boldsymbol{s}_k\boldsymbol{s}_{k-1}^T}{\|\boldsymbol{s}_{k-1}\|_2^2}\right)\prod_{j=1}^{k-2}\left(I + \frac{\boldsymbol{s}_{j+1}\boldsymbol{s}_j^T}{\|\boldsymbol{s}_j\|_2^2}\right)\boldsymbol{F}_k$$

$$= -\left(I + \frac{\boldsymbol{s}_k\boldsymbol{s}_{k-1}^T}{\|\boldsymbol{s}_{k-1}\|_2^2}\right)B_{k-1}^{-1}\boldsymbol{F}_k \qquad \text{(-4)}$$

Finally we solve (4) to obtain

$$\boldsymbol{s}_k = \frac{B_{k-1}^{-1}\boldsymbol{F}_k}{1 + \boldsymbol{s}_{k-1}^T B_{k-1}^{-1}\boldsymbol{F}_k/\|\boldsymbol{s}_{k-1}\|_2^2}$$

Broyden Algorithm (sketch)

- INPUT: $\boldsymbol{x}_0, B_0$. Set $k := 0, \boldsymbol{x} := \boldsymbol{x}_0$.
- First step: Solve $B_0 \boldsymbol{s}_0 = -\boldsymbol{F}(\boldsymbol{x}_0)$
- REPEAT until convergence
  - $x := x + \boldsymbol{s}_k$
  - Solve $B_0 \boldsymbol{z} = -\boldsymbol{F}(\boldsymbol{x})$
  - $k := k + 1$.
  - FOR $j := 1$ TO $k - 1$
    - $\boldsymbol{z} := \boldsymbol{z} + \dfrac{\boldsymbol{s}_{j+1} \boldsymbol{s}_j^T}{\|\boldsymbol{s}_j\|_2^2}$
  - $\boldsymbol{s}_k := \dfrac{\boldsymbol{z}}{1 + \boldsymbol{s}_{k-1}^T \boldsymbol{z} / \|\boldsymbol{s}_{k-1}\|_2^2}$
- END REPEAT

And this is also THE END of the course.