

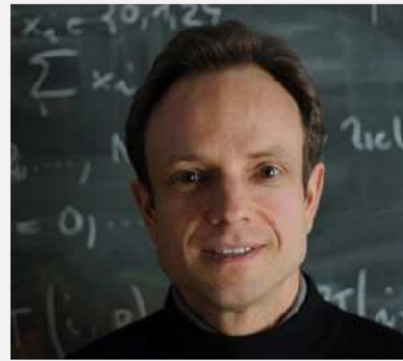
Course Video

Section 8.5.4: Damped Newton Method

Prof. R. Hiptmair, SAM, ETH Zurich

Date: December 5, 2020

(C) Seminar für Angewandte Mathematik, ETH Zürich



Dependencies. Section 8.5.1, Section 8.5.2

Duration: 20 minutes



Note: Possible minor mismatch of video and tablet notes!

[Corrections and updates can be incorporated into tablet notes only]

VIII Iterative Methods for Non-Linear Systems of Equations

8.5 Newton's Method in $\mathbb{R}^n$

NLSE: Given $C^1 F : \mathcal{D} \subset \mathbb{R}^n \longrightarrow \mathbb{R}^n, n \in \mathbb{N}$,
find $\underline{x}^* \in \mathcal{I} : F(\underline{x}^*) = 0$

► **Newton iteration:** (generalizes (8.4.2.1) to $n > 1$)

$$\underline{x}^{(k+1)} := \underline{x}^{(k)} - DF(\underline{x}^{(k)})^{-1}F(\underline{x}^{(k)}) \quad , \quad [\text{if } DF(\underline{x}^{(k)}) \text{ regular}] \quad (8.5.1.6)$$

Terminology: $-DF(\underline{x}^{(k)})^{-1}F(\underline{x}^{(k)})$ is called the **Newton correction**

Summary: Newton's method



converges *asymptotically* very fast: doubling of number of significant digits in each step



often a very small region of convergence, which requires an initial guess rather close to the solution.

} local
cvg.

quadratic local cvg.

8.5.4. Damped Newton Method

Goal: Enlarge domain of conv.

Ex. 8.5.4. | (Local convergence of Newton's method)

$n=1$:

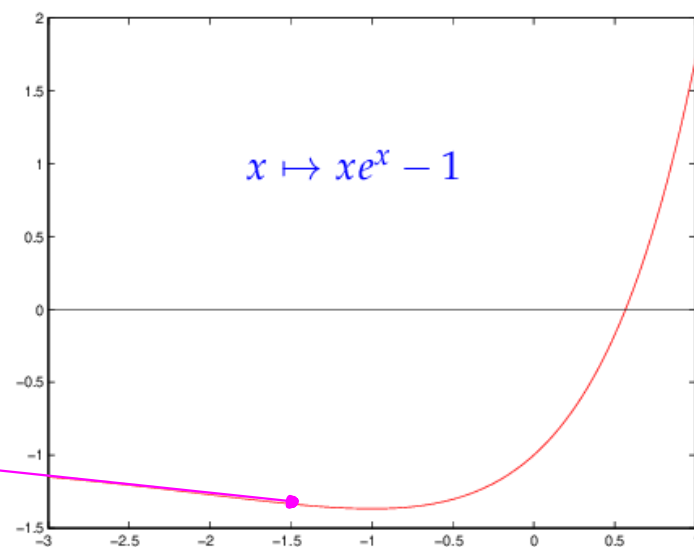
$$F(x) = xe^x - 1 \Rightarrow F'(-1) = 0$$

$$x^{(0)} < -1 \Rightarrow x^{(k)} \rightarrow -\infty,$$

$$x^{(0)} > -1 \Rightarrow x^{(k)} \rightarrow x^*,$$

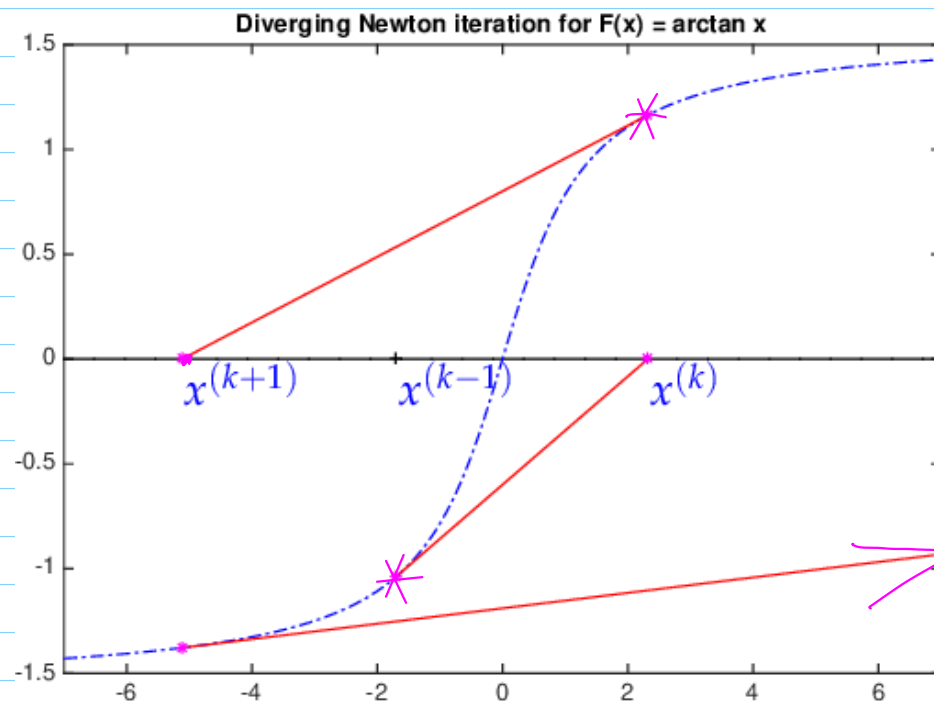
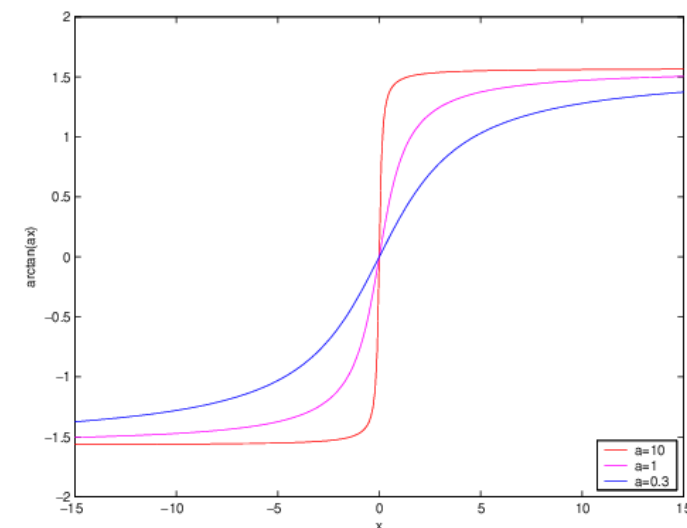
→ Newton correct in the wrong direction

▷ no remedy



$$F(x) = \arctan(ax), \quad a > 0, x \in \mathbb{R}$$

with zero $x^* = 0$.



"Overshooting"
Newton correction

③



we observe "overshooting" of Newton correction

Idea: **damping** of Newton correction:

$$\text{With } \lambda^{(k)} > 0: \mathbf{x}^{(k+1)} := \mathbf{x}^{(k)} - \lambda^{(k)} \mathbf{D}F(\mathbf{x}^{(k)})^{-1} F(\mathbf{x}^{(k)}) . \quad (8.5.4.2)$$

Terminology: $\lambda^{(k)}$ = damping factor, $\lambda \in]0, 1]$.

chosen by heuristics

$\Rightarrow \mathbf{x}^{(k)}, \lambda^{(k)}$ not affected when replacing $F \rightarrow AF$
 $A \in \mathbb{R}^{n \times n}$ regular

Affine invariant damping strategy

Choice of damping factor: affine invariant **natural monotonicity test** (NMT) :

$$\text{choose "maximal" } 0 < \lambda^{(k)} \leq 1: \quad \|\Delta \bar{\mathbf{x}}(\lambda^{(k)})\|_2 \leq (1 - \frac{\lambda^{(k)}}{2}) \|\Delta \mathbf{x}^{(k)}\|_2 \quad (8.5.4.4)$$

where $\Delta \mathbf{x}^{(k)} := \mathbf{D}F(\mathbf{x}^{(k)})^{-1} F(\mathbf{x}^{(k)}) \rightarrow$ current Newton correction ,
 $\Delta \bar{\mathbf{x}}(\lambda^{(k)}) := \mathbf{D}F(\mathbf{x}^{(k)})^{-1} F(\mathbf{x}^{(k)}) - \lambda^{(k)} \Delta \mathbf{x}^{(k)} \rightarrow$ tentative simplified Newton correction .

$\hookrightarrow$ tentative next iterate

If quad. cvg. $\Rightarrow \|\Delta \bar{\mathbf{x}}^{(k)}\| \leq \frac{1}{2} \|\Delta \mathbf{x}^{(k)}\|$

If NMT fails $\Rightarrow \lambda \leftarrow \frac{1}{2}$ & retry

passed $\Rightarrow \bullet (8.5.4.2), k \leftarrow k+1$

$\bullet \lambda \leftarrow 2\lambda$ in next step

A reject-accept approach

C++ code 8.5.4.5: Generic damped Newton method based on natural monotonicity test

```
1 template <typename FuncType, typename JacType, typename VecType>
2 void dampnewton(const FuncType &F, const JacType &DF,
3                 VecType &x, double rtol, double atol)
4 {
5     using index_t = typename VecType::Index;
6     using scalar_t = typename VecType::Scalar;
7     const index_t n = x.size(); // No. of unknowns
8     const scalar_t lmin = 1E-3; // Minimal damping factor
9     scalar_t lambda = 1.0; // Initial and actual damping factor
10    VecType s(n), st(n); // Newton corrections
11    VecType xn(n); // Tentative new iterate
12    scalar_t sn, stn; // Norms of Newton corrections
13
14    do {
15        auto jacfac = DF(x).lu(); // LU-factorize Jacobian
16        s = jacfac.solve(F(x)); // Newton correction
17        sn = s.norm(); // Norm of Newton correction
18        lambda *= 2.0;
19        do {
20            lambda /= 2; // Reduce damping factor
21            if (lambda < lmin) throw "No convergence: lambda -> 0";
22            xn = x - lambda*s; // Tentative next iterate
23            st = jacfac.solve(F(xn)); // Simplified Newton correction
24            stn = st.norm();
25        } while (stn > (1-lambda/2)*sn); // Natural monotonicity test
26        x = xn; // Now: xn accepted as new iterate
27        lambda = std::min(2.0*lambda, 1.0); // Try to mitigate damping
28    }
29    // Termination based on simplified Newton correction
30    while ((stn > rtol*x.norm()) && (stn > atol));
31 }
32
```

(4)

Exp. 8.5.4.6 (Damped Newton method)

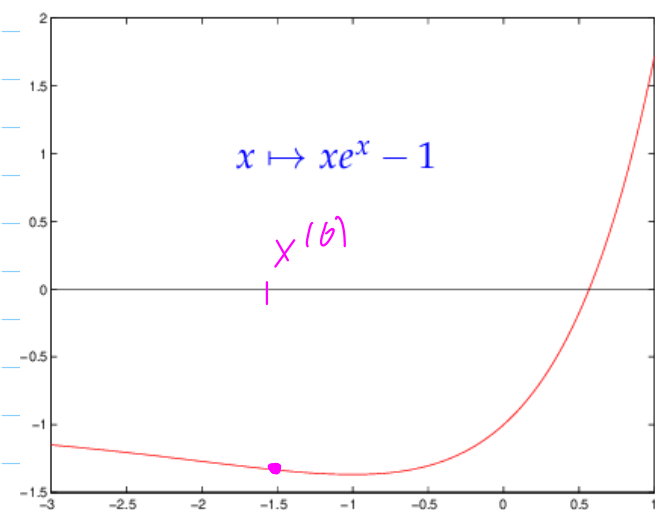
$$F(x) = \arctan(x),$$

- $x^{(0)} = 20$
- $q = \frac{1}{2}$
- LMIN = 0.001

k	$\lambda^{(k)}$	$x^{(k)}$	$F(x^{(k)})$
1	0.03125	0.94199967624205	0.75554074974604
2	0.06250	0.85287592931991	0.70616132170387
3	0.12500	0.70039827977515	0.61099321623952
4	0.25000	0.47271811131169	0.44158487422833
5	0.50000	0.20258686348037	0.19988168667351
6	1.00000	-0.00549825489514	-0.00549819949059
7	1.00000	0.00000011081045	0.00000011081045
8	1.00000	-0.00000000000001	-0.00000000000001

▷ Review questions 8.5.4.8

Exp. 8.5.4.7 (Failure of damped Newton method)



k	$\lambda^{(k)}$	$x^{(k)}$	$F(x^{(k)})$
1	0.25000	-4.4908445351690	-1.0503476286303
2	0.06250	-6.1682249558799	-1.0129221310944
3	0.01562	-7.6300006580712	-1.0037055902301
4	0.00390	-8.8476436930246	-1.0012715832278
5	0.00195	-10.5815494437311	-1.0002685596314

Bailed out because of $\lambda < \text{LMIN}$!

$$x^{(0)} = -1.5$$

5

7