ETH Lecture 401-0663-00L Numerical Methods for CSE

Numerical Methods for Computational Science and Engineering

Prof. R. Hiptmair, SAM, ETH Zurich

(with contributions from Prof. P. Arbenz and Dr. V. Gradinaru)

Autumn Term 2015

URL: http://www.sam.math.ethz.ch/~hiptmair/tmp/NumCSE/NumCSE15.pdf

II. Iterative Methods for Non-linear Systems of Equations

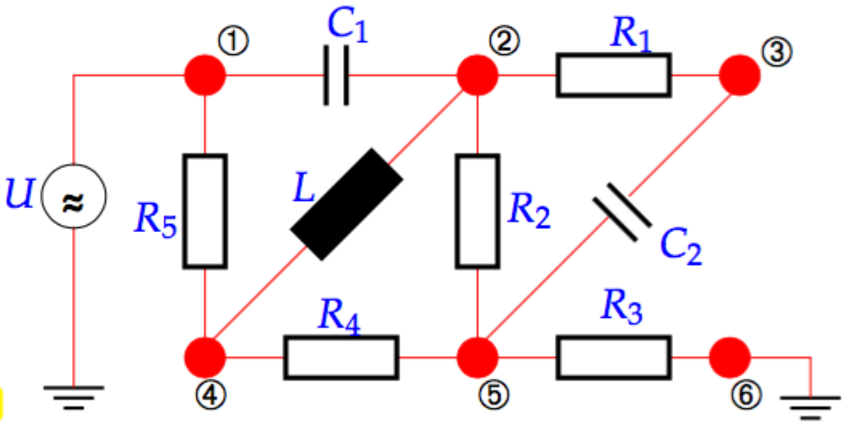
Case study: Nodal analysis of electrical circuits

● $\hat{=}$ nodes

Circuit elements (connect nodes)
↳ all linear*

▭ $\hat{=}$ resistor (resistance R)
▬ $\hat{=}$ coil (inductance L)
⊣⊢ $\hat{=}$ capacitor (capacity C)

— $\hat{=}$ wire



I_{kj} $\hat{=}$ current node k → node j : $I_{kj} = -I_{jk}$

Kirchhoffs law : $\sum_{j\in S(k)} I_{kj} = 0$ (1)

U_k $\hat{=}$ nodal potential

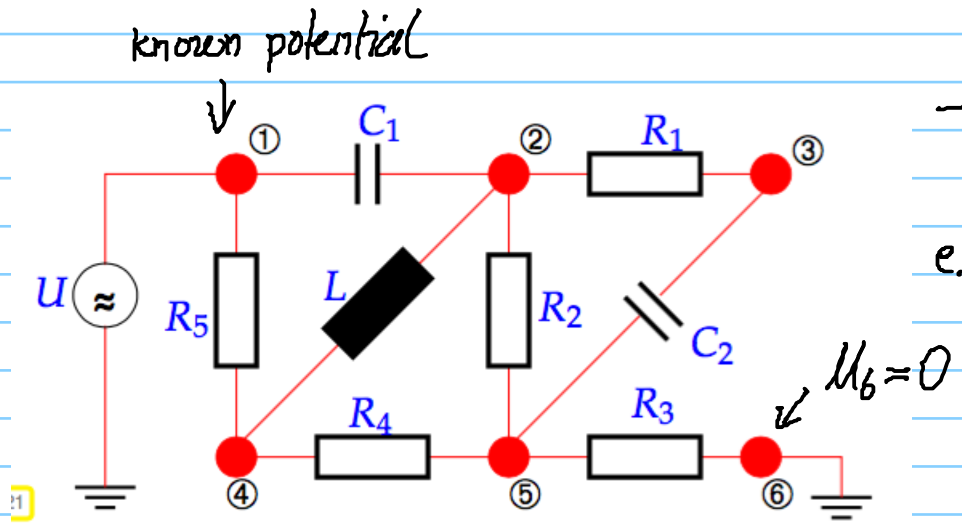
$$* \quad I_{kj} = \alpha\,(U_k - U_j) \qquad (2)$$

↳ R^{-1} or $i\omega C$ or $-i/\omega L$

$\uparrow$ $\in \mathbb{C}$ · complex amplitudes [ω $\hat{=}$ angular frequ.]

$$i(t) = \mathrm{Re}(I e^{i\omega t}) = \mathrm{Re}(I)\cos(\omega t) - \mathrm{Im}(I)\sin(\omega t)$$

Nodal analysis :
- (1) for each node (with unknown pontential)
- Replace all currents using (2)



→ 4 equations from (1)

e.g. $I_{12}+I_{32}+I_{52}+I_{42}=0$

$$\begin{aligned}
②:&\quad \imath\omega C_1(U_2-U_1)+R_1^{-1}(U_2-U_3)-\imath\omega^{-1}L^{-1}(U_2-U_4)+R_2^{-1}(U_2-U_5) &=& \;0\,,\\
③:&\quad R_1^{-1}(U_3-U_2)+\imath\omega C_2(U_3-U_5) &=& \;0\,,\\
④:&\quad R_5^{-1}(U_4-U_1)-\imath\omega^{-1}L^{-1}(U_4-U_2)+R_4^{-1}(U_4-U_5) &=& \;0\,,\\
⑤:&\quad R_2^{-1}(U_5-U_2)+\imath\omega C_2(U_5-U_3)+R_4^{-1}(U_5-U_4)+R_3^{-1}(U_5-U_6) &=& \;0\,,
\end{aligned}$$

$$U_1=U\quad,\quad U_6=0\,.$$

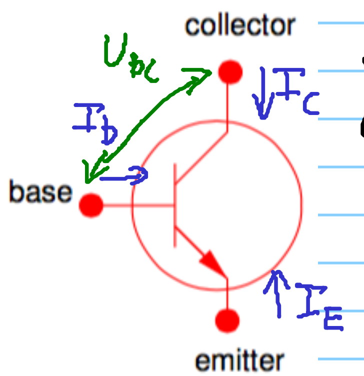
$$\begin{pmatrix}
\imath\omega C_1+\frac{1}{R_1}-\frac{i}{\omega L}+\frac{1}{R_2} & -\frac{1}{R_1} & \frac{i}{\omega L} & -\frac{1}{R_2}\\
-\frac{1}{R_1} & \frac{1}{R_1}+\imath\omega C_2 & 0 & -\imath\omega C_2\\
\frac{i}{\omega L} & 0 & \frac{1}{R_5}-\frac{i}{\omega L}+\frac{1}{R_4} & -\frac{1}{R_4}\\
-\frac{1}{R_2} & -\imath\omega C_2 & -\frac{1}{R_4} & \frac{1}{R_2}+\imath\omega C_2+\frac{1}{R_4}+R_3^{-1}
\end{pmatrix}
\begin{pmatrix}U_2\\U_3\\U_4\\U_5\end{pmatrix}
=
\begin{pmatrix}\imath\omega C_1 U\\0\\\frac{1}{R_5}U\\0\end{pmatrix}$$

$\triangleq$ LSE (in $\mathbb{C}$)

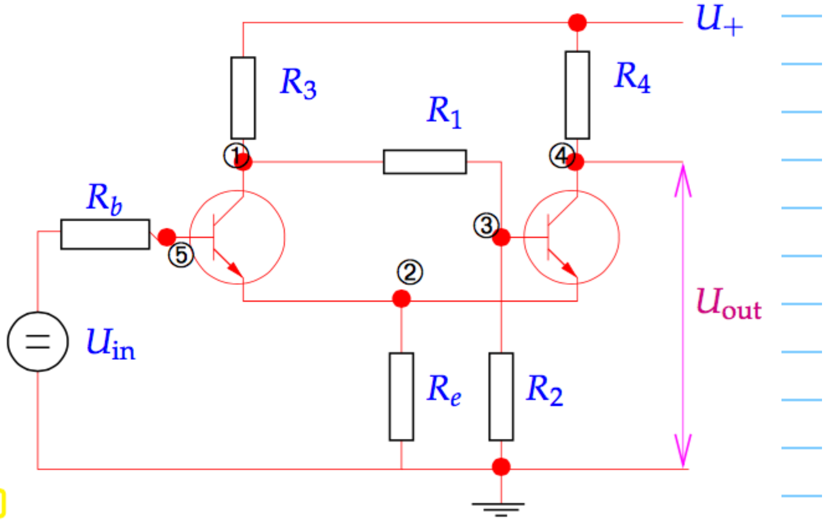
For large circuits : sparse LSE

Schmitt trigger ▷

NPN transistors :



3-port-circuit element



Ebers-Moll model : strongly nonlinear

$$I_C = I_S\left(e^{\frac{U_{BE}}{U_T}}-e^{\frac{U_{BC}}{U_T}}\right)-\frac{I_S}{\beta_R}\left(e^{\frac{U_{BC}}{U_T}}-1\right)=I_C(U_{BE},U_{BC})\,,$$

$$I_B = \frac{I_S}{\beta_F}\left(e^{\frac{U_{BE}}{U_T}}-1\right)+\frac{I_S}{\beta_R}\left(e^{\frac{U_{BC}}{U_T}}-1\right)=I_B(U_{BE},U_{BC})\,,$$

$$I_E = I_S\left(e^{\frac{U_{BE}}{U_T}}-e^{\frac{U_{BC}}{U_T}}\right)+\frac{I_S}{\beta_F}\left(e^{\frac{U_{BE}}{U_T}}-1\right)=I_E(U_{BE},U_{BC})\,.$$

$$\begin{aligned}
①:&\quad R_3(U_1-U_+)+R_1(U_1-U_3)+I_B(U_5-U_1,U_5-U_2) &=0\,,\\
②:&\quad R_eU_2+I_E(U_5-U_1,U_5-U_2)+I_E(U_3-U_4,U_3-U_2) &=0\,,\\
③:&\quad R_1(U_3-U_1)+I_B(U_3-U_4,U_3-U_2) &=0\,,\\
④:&\quad R_4(U_4-U_+)+I_C(U_3-U_4,U_3-U_2) &=0\,,\\
⑤:&\quad R_b(U_5-U_{\text{in}})+I_B(U_5-U_1,U_5-U_2) &=0\,.
\end{aligned}$$

5 equations ↔ 5 unknowns U_1,U_2,U_3,U_4,U_5

In short : $F(\underline{x})=0\;,\;\underline{x}=(U_1,\dots,U_5)^T$

$F : \mathbb{R}^5 \longrightarrow \mathbb{R}^5$

NLSE : Given $F : D \subset \mathbb{R}^n \longrightarrow \mathbb{R}^n$

seek $\underline{x} \in \mathbb{R}^n$: $F(\underline{x}) = 0$

- no general theory
- F $\triangleq$ function $y = F(x)$ "black box"

2.1. Iterative Methods

An iterative method for (approximately) solving the non-linear equation $F(\mathbf{x}) = 0$ is an algorithm generating an arbitrarily long sequence $\left(\mathbf{x}^{(k)}\right)_k$ of approximate solutions.

$\mathbf{x}^{(k)} \triangleq k$-th iterate

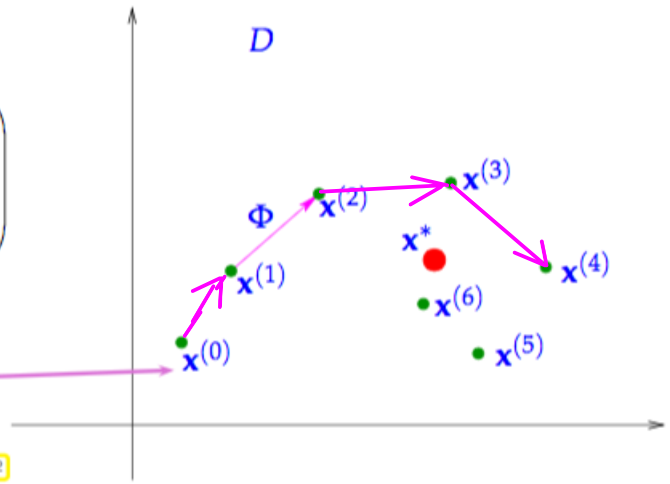


Fig. 62

Iteration error : $e^{(k)} = \underline{x}^{(k)} - \underline{x}^*$, $\epsilon_k := \| x^{(k)} - x^* \|$

More concrete: stationary m-point iteration

$$\underline{x}^{(k+1)} = \Phi_F(x^{(k)}, \ldots, x^{(k-m+1)})$$

↳ iteration function

Initial guess : $\underline{x}^{(0)}, \ldots, \underline{x}^{(m-1)} \in \mathbb{R}^n$

Issues : → well defined ?
→ does it converge ?
→ If yes, does it converge to a solution ?
→ How fast (speed of convergence) ?

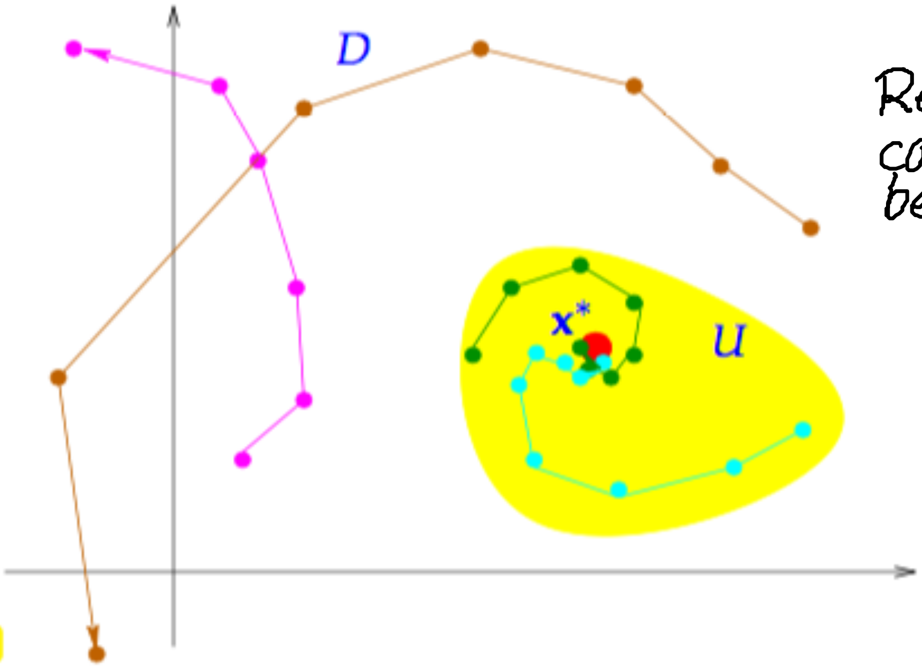
[Initial guess matters much !]

Definition 2.1.8. Local and global convergence → [12, Def. 17.1]

As stationary m-point iterative method converges locally to $\mathbf{x}^* \in \mathbb{R}^n$, if there is a neighborhood $U \subset D$ of $\mathbf{x}^*$, such that

$$\mathbf{x}^{(0)}, \ldots, \mathbf{x}^{(m-1)} \in U \quad \Rightarrow \quad \mathbf{x}^{(k)} \text{ well defined } \quad \wedge \quad \lim_{k \to \infty} \mathbf{x}^{(k)} = \mathbf{x}^*$$

where $(\mathbf{x}^{(k)})_{k \in \mathbb{N}_0}$ is the (infinite) sequence of iterates.
If $U = D$, the iterative method is globally convergent.



Region $\mathcal{U}$ of local convergence may be very small.

2.1.1. Speed of convergence

"Slow methods":

Definition 2.1.9. Linear convergence

A sequence $\mathbf{x}^{(k)}$, $k = 0, 1, 2, \ldots$, in $\mathbb{R}^n$ converges linearly to $\mathbf{x}^* \in \mathbb{R}^n$,

$$\exists L < 1: \quad \left\|\mathbf{x}^{(k+1)} - \mathbf{x}^*\right\| \leq L\left\|\mathbf{x}^{(k)} - \mathbf{x}^*\right\| \quad \forall k \in \mathbb{N}_0 . \qquad (*)$$

smallest possible L in $(*)$: rate of (linear) cvg.

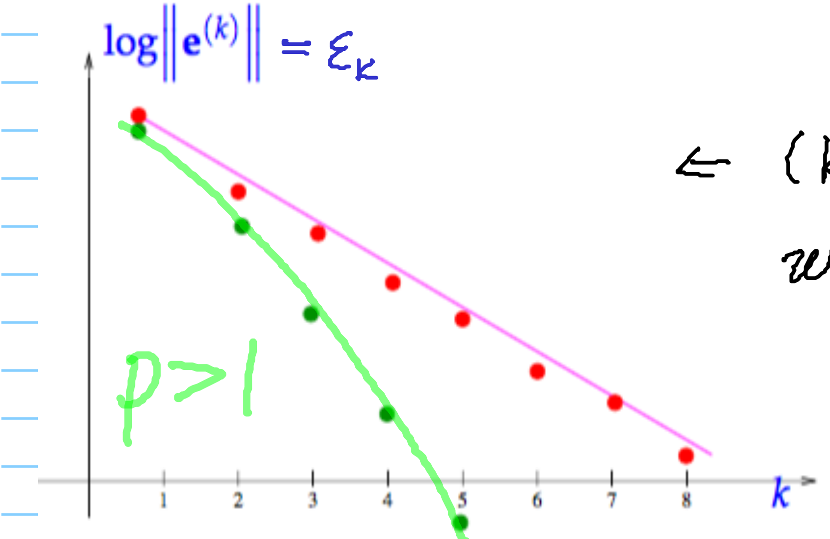
How to tell linear cvg. in numerical test (x^* known)

$\varepsilon_k := \|x^{(k)} - x^*\|$ known,

L.C. $\Rightarrow$ assume $\varepsilon_{k+1} \approx L\,\varepsilon_k \approx L^2 \varepsilon_{k-1} \cdots$

$$\Rightarrow \quad \varepsilon_k \approx L^k \varepsilon_0$$

$$\log \varepsilon_k \approx k \underbrace{\log L}_{\leq 0} + \log \varepsilon_0$$



$\leftarrow$ $(k, \log \varepsilon_k)$ on a straight line with slop $\log L < 0$!

From tabulated values: check, if $\varepsilon_k / \varepsilon_{k-1} \approx L$

"Faster convergence"

Definition 2.1.17. Order of convergence → [12, Sect. 17.2], [4, Def. 5.14], [16, Def. 6.1]

A **convergent** sequence $\mathbf{x}^{(k)}$, $k = 0, 1, 2, \ldots$, in $\mathbb{R}^n$ converges with order p to $\mathbf{x}^* \in \mathbb{R}^n$, if

$$\exists C > 0: \quad \left\|\mathbf{x}^{(k+1)} - \mathbf{x}^*\right\| \leq C\left\|\mathbf{x}^{(k)} - \mathbf{x}^*\right\|^p \quad \forall k \in \mathbb{N}_0 ,$$

and, in addition, $C < 1$ in the case $p = 1$ (linear convergence → Def. 2.1.9).

Identifying cvg. of order p from measured ε_k

$$\varepsilon_{k+1} \approx C \varepsilon_k^p$$

$$\log \varepsilon_{k+1} \approx \log C + p \log \varepsilon_k$$

$\xrightarrow{\quad}$ subtract $\xrightarrow{\quad}$

$$\log \varepsilon_{k+1} - \log \varepsilon_k \approx p\,(\log \varepsilon_k - \log \varepsilon_{k-1})$$

$$p = \frac{\log \varepsilon_{k+1} - \log \varepsilon_k}{\log \varepsilon_k - \log \varepsilon_{k-1}}$$

Famous example: $\sqrt{}$-iteration $(n = 1, m = 1)$

$$x^{(k+1)} = \tfrac{1}{2}\left(x^{(k)} + \frac{a}{x^{(k)}}\right) , \quad a > 0 \; : x^{(k)} \to \sqrt{a}$$

$$\underbrace{x^{(k+1)} - \sqrt{a}}_{\varepsilon^{(k+1)}} = \underbrace{\frac{1}{2x^{(k)}}}_{\leq \frac{1}{2\sqrt{a}}} \Big(\underbrace{x^{(k)} - \sqrt{a}}_{\varepsilon^{(k)}}\Big)^2 , \quad x^{(k)} > \sqrt{a} \text{ for } k \geq 1$$

⑤ → quadratic convergence $p = 2$

k	$x^{(k)}$	$e^{(k)} := x^{(k)} - \sqrt{2}$	$\log\frac{\lvert e^{(k)}\rvert}{\lvert e^{(k-1)}\rvert} : \log\frac{\lvert e^{(k-1)}\rvert}{\lvert e^{(k-2)}\rvert}$
0	2.00000000000000000	0.58578643762690485	
1	1.50000000000000000	0.08578643762690485	
2	1.41666666666666652	0.00245310429357137	1.850
3	1.41421568627450966	0.00000212390141452	1.984
4	1.41421356237468987	0.00000000000159472	2.000
5	1.41421356237309492	0.00000000000000022	$\boxed{0.630}$

[red ≙ correct digits] roundoff

$\Updownarrow$

relative error

$x^{(k)} = x^*(1+\delta_k)$: $\delta_k \le 10^{-\ell} \Leftrightarrow x^{(k)}$ has ℓ correct digits

doubling of no. of correct digits in every step!

Quadratic cvg. $|x^{(k+1)} - x^*| \approx C\,|x^{(k)} - x^*|^2$

$\rightarrow |\delta_{k+1} x^*| \approx C\,|\delta_k x^*|^2$, $x^* \neq 0$

$\Rightarrow |\delta_{k+1}| \approx C|x^*|\,|\delta_k|^2$

$C, |x^*| \approx 1$, $\delta_k = 10^{-\ell} \Rightarrow \delta_{k+1} \approx 10^{-2\ell}$

Compare linear cvg.: fixed number (also a fraction) of extra correct digits in each step

2.1.2. Termination

Ideal: ↳ STOP, if $x^{(k)}$ is "just good enough"

$\|x^{(k)} - x^*\| \le \tau_{abs}$ or $\|x^{(k)} - x^*\| \le \tau_{rel} \cdot \|x^*\|$

Absolute tolerance relative tolerance

Not practical, because x^* is not known

Practical: [for solving $F(x) = 0$]

① Residual based termination:

$$\text{STOP, if } \underbrace{\|F(x^{(k)})\| \le \tau}_{\text{tells little about } \varepsilon_k}$$

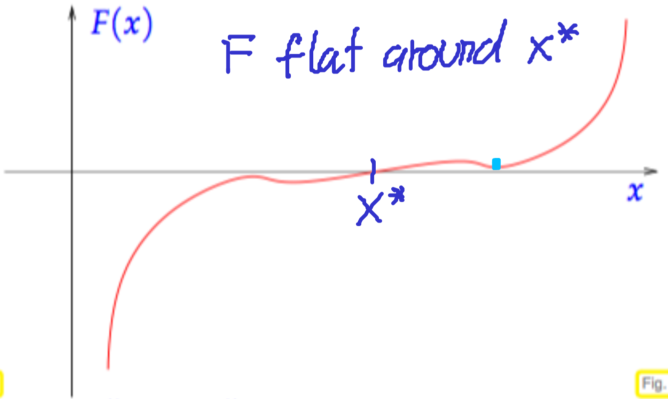
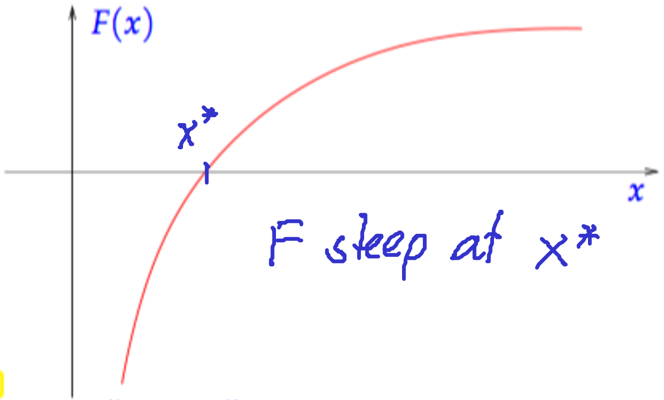


Fig. 67

$\|F(\mathbf{x}^{(k)})\|$ small $\not\Rightarrow |x - x^*|$ small



$\|F(\mathbf{x}^{(k)})\|$ small $\Rightarrow |x - x^*|$ small

⑥ [8.10.2015]

Rep: Solve $F(\underline{x}) = 0$, $F : D \subset \mathbb{R}^n \to \mathbb{R}^n$

Iterative method: $(x^{(k-m+1)}, \dots, x^{(k)}) \longrightarrow x^{(k+1)}$

Cvg of order $p > 1$: $\|e^{(k+1)}\| \le C \|e^{(k)}\|^p$ (*)

If (*), cvg. guaranteed, if $C\|e^{(0)}\|^{p-1} < 1$ → not practical

Hint: $\|e^{(k+1)}\| \le C\|e^{(k)}\|^{p-1} \|e^{(k)}\|$

② Correction based termination:

$$\text{STOP, if } \|\underline{x}^{(k+1)} - \underline{x}^{(k)}\| \le \begin{cases} \tau_{abs} \\ \tau_{rel} \cdot \|x^{(k+1)}\| \end{cases}$$

→ Generically, no guarantees

Exception: Linearly cvg. iteration with known rate $L < 1$

$$\|x^{(k+1)} - x^*\| \le L \|x^{(k)} - x^*\|$$

$$\|x^{(k)} - x^*\| \le \|x^{(k)} - x^{(k+1)}\| + \|x^{(k+1)} - x^*\|$$

$$\le \|x^{(k)} - x^{(k+1)}\| + L\|x^{(k)} - x^*\|$$

$$\Rightarrow \|x^{(k)} - x^*\| \le \frac{1}{1-L}\|x^{(k)} - x^{(k+1)}\|$$

$$\Rightarrow \|x^{(k+1)} - x^*\| \le \frac{L}{1-L}\|x^{(k)} - x^{(k+1)}\|$$

↳ Reliable upper bound: can replace $\|x^{(k)} - x^*\|$ in termination criteria

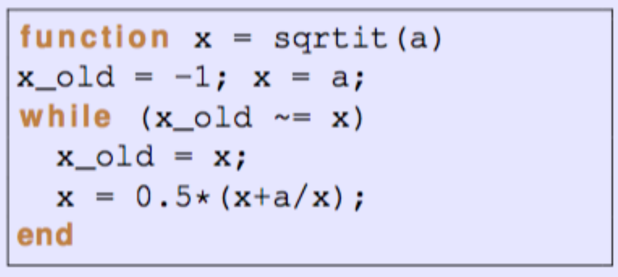
If we overestimate L ⇒ still reliable termination

Remark: → $\|x^{(k)} - x^*\| \le L^{k-1}\|x^{(1)} - x^*\|$

$$\le \frac{L^k}{1-L}\|x^{(1)} - x^{(0)}\|$$

→ Can be used for a priori termination

#steps fixed before start of iteration.



```
function x = sqrtit(a)
x_old = -1; x = a;
while (x_old ~= x)
  x_old = x;
  x = 0.5*(x+a/x);
end
```

← "M-based termination"

⇒ guarantees relative update ≤ EPS

2.2. Fixed point iterations = 1-point methods

$\underline{x}^{(k+1)} = \Phi_F(x^{(k)})$ with iteration function $\Phi_F : U \subset \mathbb{R}^n \to \mathbb{R}^n$

$$\left.\begin{array}{l} x^* = \lim_{k\to\infty} x^{(k)} \\ \Phi \text{ continuous} \end{array}\right\} \Rightarrow \underset{\uparrow \text{ fixed point of } \Phi_F}{\underline{x}^*} = \Phi_F(x^*)$$

FPI is consistent: $\Phi_F(x) = x \Leftrightarrow F(x) = 0$

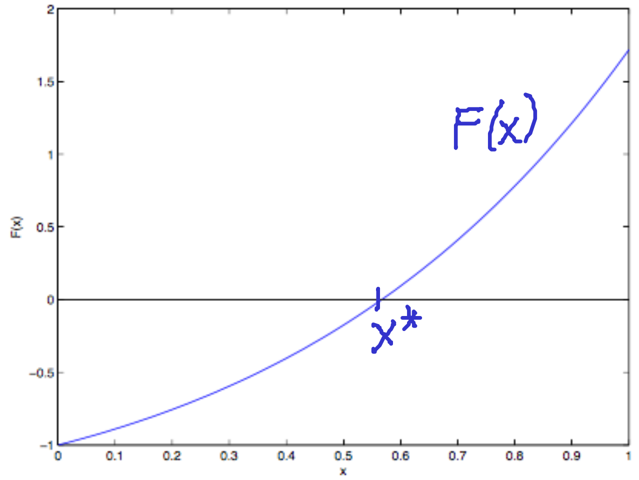
Many Φ_F possible !

Ex :

$$F(x) = xe^x - 1\,, \quad x \in [0,1]\,.$$

Different fixed point forms:

$$\begin{aligned}\Phi_1(x) &= e^{-x}\,,\\ \Phi_2(x) &= \frac{1+x}{1+e^x}\,,\\ \Phi_3(x) &= x+1-xe^x\,.\end{aligned}$$



$x^{(0)} = 0.5$

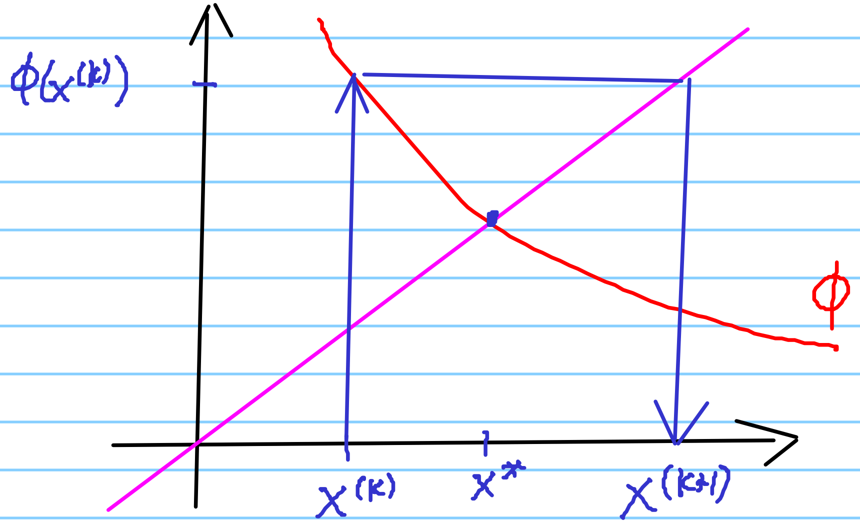
	Φ_1	Φ_2	Φ_3
k	$\lvert x_1^{(k+1)} - x^*\rvert$	$\lvert x_2^{(k+1)} - x^*\rvert$	$\lvert x_3^{(k+1)} - x^*\rvert$
0	0.067143290409784	0.067143290409784	0.067143290409784
1	0.039387369302849	0.000832287212566	0.108496074240152
2	0.021904078517179	0.000000125374922	0.219330611898582
3	0.012559804468284	0.000000000000003	0.288178118764323
4	0.007078662470882	0.000000000000000	0.723649245792953
5	0.004028858567431	0.000000000000000	0.410183132337935
6	0.002280343429460	0.000000000000000	1.186907542305364
7	0.001294757160282	0.000000000000000	0.146569797006362
8	0.000733837662863	0.000000000000000	0.310516641279937
9	0.000416343852458	0.000000000000000	0.357777386500765
10	0.000236077474313	0.000000000000000	0.974565695952037

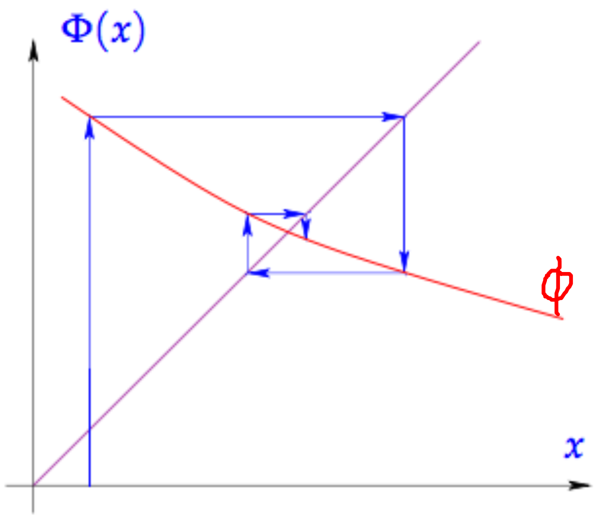
↓

Linear cvg. quadratic cvg. no cvg.

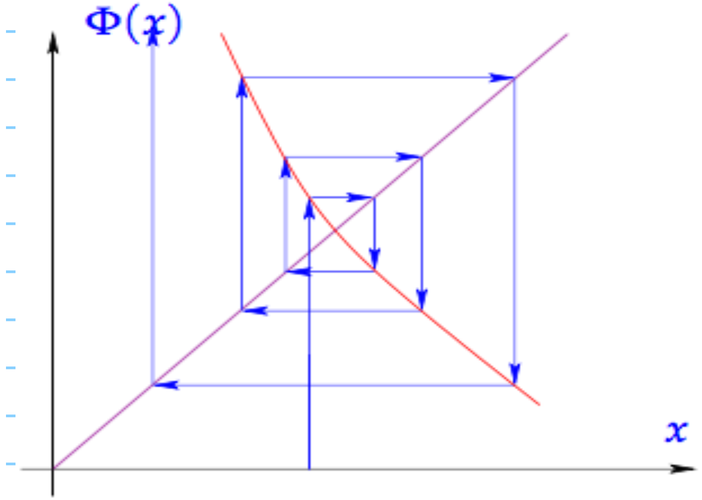
How to predict this ?

Visualization of FPI in 1D :

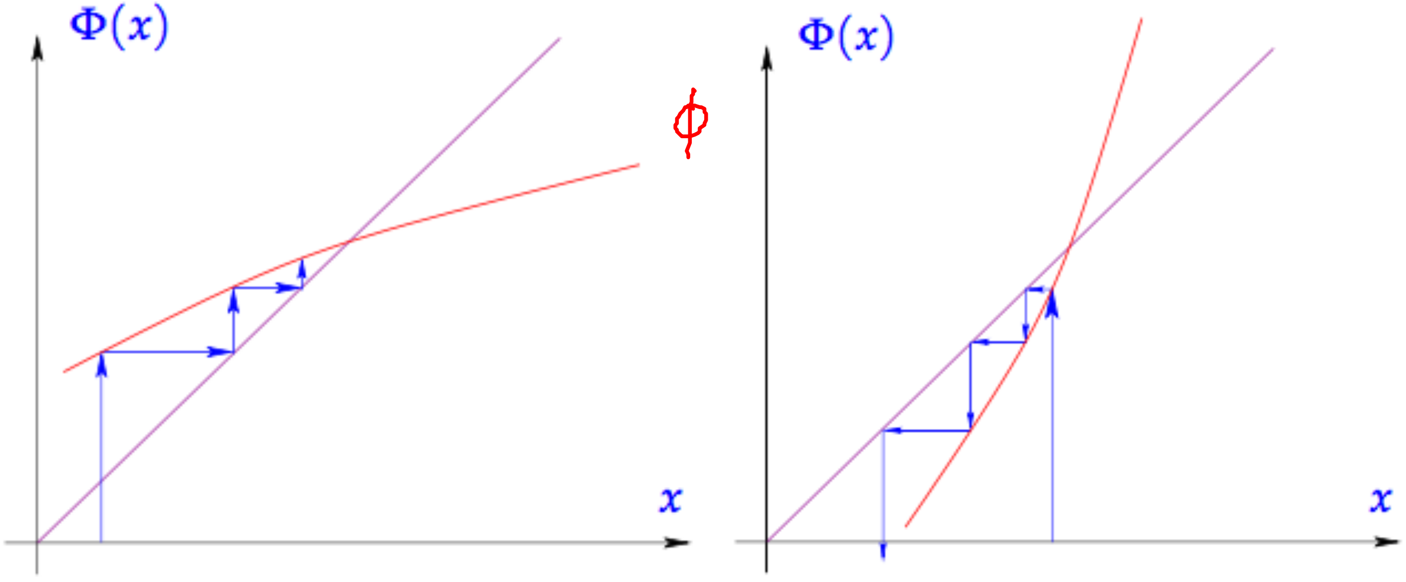




$-1 < \Phi'(x^*) \le 0$ ➤ convergence



$\Phi'(x^*) < -1$ ➤ divergence



$0 \le \Phi'(x^*) < 1$ ➤ convergence $1 < \Phi'(x^*)$ ➤ divergence

⇒ Slope of $x \to \phi(x)$ at fixed point crucial !

Analysis (ϕ smooth) : Taylor expansion around x^*

$$x^{(k+1)} - x^* = \phi(x^{(k)}) - \phi(x^*)$$

neglect

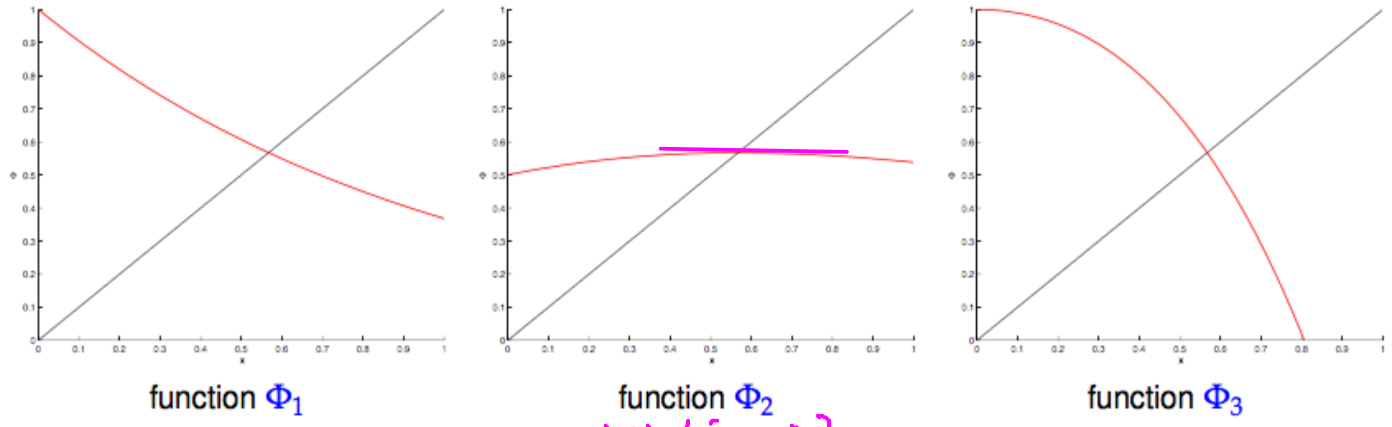
$$= \phi'(x^*)(x^{(k)}-x^*) + \tfrac{1}{2}\phi''(x^*)(x^{(k)}-x^*)^2 + \cancel{\tfrac{1}{6}\phi'''(x^*)(x^{(k)}-x^*)^3 + \dots}$$

If $|x^{(k)}-x^*| \ll 1$

If $|\phi'(x^*)| < 1$ ⇒ local linear cvg, rate $\approx \phi'(x^*)$

If $\phi'(x^*) = 0$ ⇒ local quadratic cvg.

In Example :


function Φ_1 function Φ_2 (flat in x^*) function Φ_3

Remains true for $n > 1$:

Lemma 2.2.10. Sufficient condition for local linear convergence of fixed point iteration → [12, Thm. 17.2], [4, Cor. 5.12]

If $\Phi : U \subset \mathbb{R}^n \mapsto \mathbb{R}^n$, $\Phi(\mathbf{x}^*) = \mathbf{x}^*$, Φ differentiable in $\mathbf{x}^*$, and $\|D\Phi(\mathbf{x}^*)\| < 1$, then the fixed point iteration (2.2.2) converges locally and at least linearly.

matrix norm, Def. 1.5.68 !

Jacobian $\in \mathbb{R}^{n,n}$

If $D\phi(x^*) = 0$ ⇒ local quadratic cvg.

Lemma 2.2.12. Sufficient condition for linear convergence of fixed point iteration

Let U be convex and $\Phi : U \subset \mathbb{R}^n \mapsto \mathbb{R}^n$ be continuously differentiable with

$$L := \sup_{\mathbf{x}\in U} \|D\Phi(\mathbf{x})\| < 1 .$$

If $\Phi(\mathbf{x}^*) = \mathbf{x}^*$ for some interior point $\mathbf{x}^* \in U$, then the fixed point iteration $\mathbf{x}^{(k+1)} = \Phi(\mathbf{x}^{(k)})$ converges to $\mathbf{x}^*$ at least linearly with rate L. [global cvg. in $\mathcal{U}$]

2.3. $n=1$: Zero finding

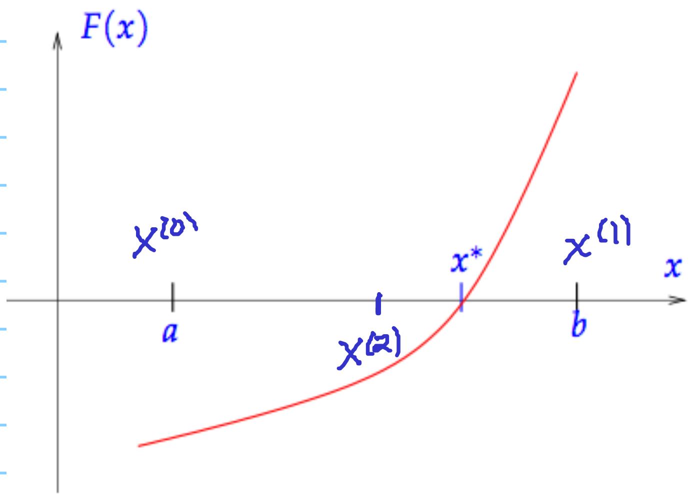
Seek $x^* \in \mathbb{R}$: $F(x^*)=0$, $\underset{\text{continuous}}{F} : I \subset \mathbb{R} \to \mathbb{R}$

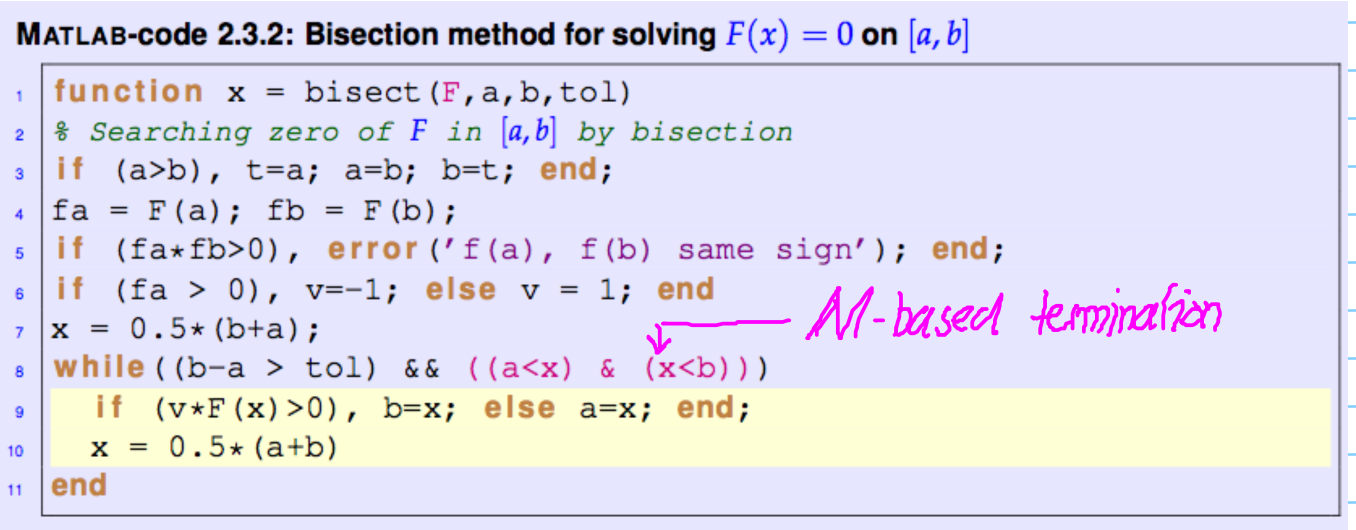
2.3.1. Bisection

Interval $[a,b]$:

$\left.\begin{matrix} F(a)<0 \\ F(b)>0 \end{matrix}\right\} \Rightarrow \exists x^* \in]a,b[:$ $F(x^*)=0$

[intermediate value theorem]





MATLAB-code 2.3.2: Bisection method for solving $F(x)=0$ on $[a,b]$

```
function x = bisect(F,a,b,tol)
% Searching zero of F in [a,b] by bisection
if (a>b), t=a; a=b; b=t; end;
fa = F(a); fb = F(b);
if (fa*fb>0), error('f(a), f(b) same sign'); end;
if (fa > 0), v=-1; else v = 1; end
x = 0.5*(b+a);
while((b-a > tol) && ((a<x) & (x<b)))
  if (v*F(x)>0), b=x; else a=x; end;
  x = 0.5*(a+b)
end
```

M-based termination

$\Rightarrow \quad |x^* - x^{(k)}| \leq 2^{-k}|b-a|$ "kind of lin. cvg."

$\to$ robust method : $|x^{(k)} - x^*| \leq tol$ is guaranteed
simple method , only F-evaluation required

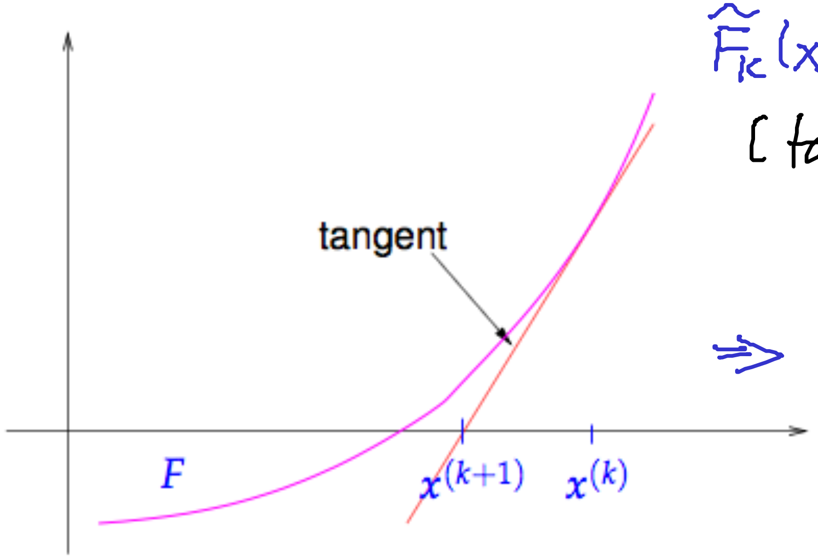
2.3.2. Model Function methods

$\hookrightarrow$ In step k : • replace F with $\underset{\uparrow \text{ should be simple}}{\widetilde{F}_k}$

• $x^{(k+1)}$: $\widetilde{F}_k(x^{(k+1)})=0$

2.3.2.1 Newton's method

Assume: F differentiable



$$\widetilde{F}_k(x) = F(x^{(k)}) + F'(x^{(k)})(x-x^{(k)})$$

[tangent in $(x^{(k)}, F(x^{(k)}))$]

$$\widetilde{F}_k(x^{(k+1)}) \overset{!}{=} 0$$

$$\Rightarrow \quad x^{(k+1)} = x^{(k)} - \frac{F(x^{(k)})}{F'(x^{(k)})}$$

$[F'(x^{(k)}) \neq 0 !]$

$\hat{=}$ fixed point iteration with $\phi(x) = x - \frac{F(x)}{F'(x)}$

$$\phi'(x) = 1 - \frac{(F'(x))^2 - F(x)F''(x)}{(F'(x))^2} = \frac{F(x)F''(x)}{(F'(x))^2}$$

$\{F(x^*)=0 \Leftrightarrow \phi(x^*)=x^*\} \Longrightarrow \phi'(x^*)=0 \Rightarrow$ local quad. cvg.

⑩

Example: $F(x) = x^2 - a \Rightarrow F'(x) = 2x$

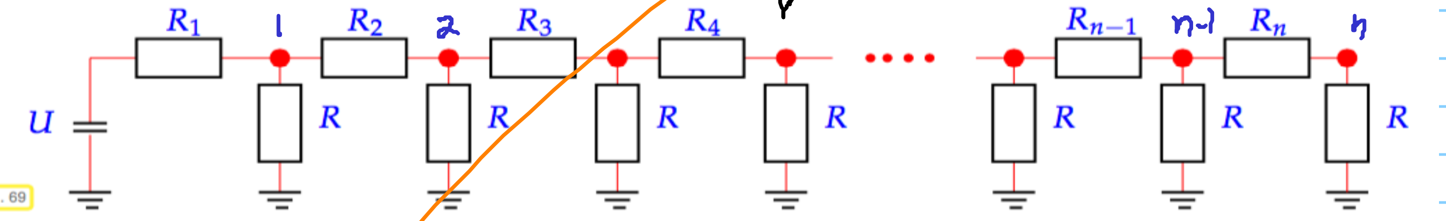
N.I.: $x^{(k+1)} = x^{(k)} - \frac{(x^{(k)})^2 - a}{2x^{(k)}} = \frac{1}{2}\left(x^{(k)} + \frac{a}{x^{(k)}}\right)$

Drawback:
- local cvg.
- $F'(x)$ not available

Case study: Finding F'

want to achieve target potential here by varying R

Linear circuit:



Nodal analysis ⇒ LSE: $(A + xI)\underline{u}(x) = \underline{b}$ (∗)

$[x = 1/R]$

$\uparrow$ symmetric tridiagonal matrix

$\uparrow$ vector of nodal pot.

$$\Rightarrow \quad F(x) = e_k^T \underline{u}(x) - 1 = 0$$

$$\Rightarrow \quad F'(x) = e_k^T \underline{u}'(x)$$

$$A = \begin{bmatrix} \frac{1}{R_1} + \frac{1}{R_2} & -\frac{1}{R_2} & & \\ -\frac{1}{R_2} & \frac{1}{R_2} + \frac{1}{R_3} & -\frac{1}{R_3} & \\ & \ddots & \ddots & -\frac{1}{R_n} \\ & & \cdots & \frac{1}{R_n} \end{bmatrix} \in \mathbb{R}^{n,n}$$

▷ $(A + xI)\underline{u}(x) = \underline{b}$ can be solved with effort $O(n)$

$\underline{u}'(x)$ by implicit differentiation

$\frac{d}{dx}$ on (∗) & product rule:

$$I \cdot \underline{u}(x) + (A + xI)\underline{u}'(x) = 0$$

$$\underline{u}'(x) = -(A + xI)^{-1}\underline{u}(x)$$

Newton iteration:

(i) Solve $(A + x^{(k)}I)\underline{u} = \underline{b}$

(ii) Solve $(A + x^{(k)}I)\underline{u}' = -\underline{u}$

(iii) $x^{(k+1)} = x^{(k)} - \frac{e_k^T \underline{u} - 1}{e_k^T \underline{u}'}$

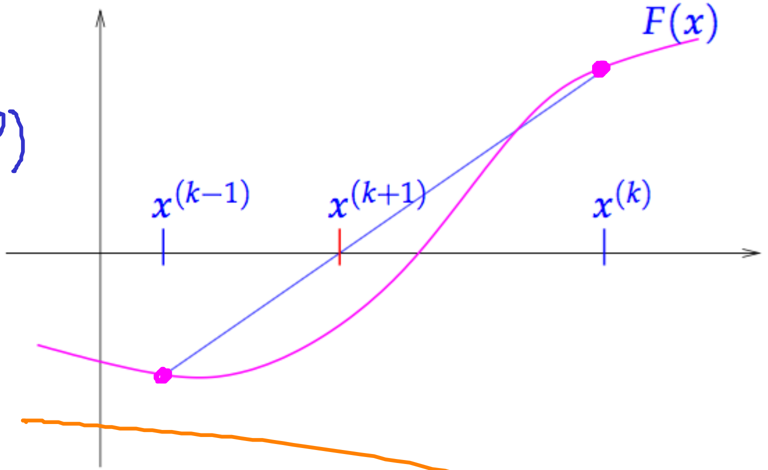
2.3.2.2. Multi-point method $(m > 1)$

Simplest : secant method

$$\tilde{F}_k(x) = F(x^{(k)}) + \Delta s\,(x - x^{(k)})$$

$$\Delta s = \frac{F(x^{(k)}) - F(x^{(k-1)})}{x^{(k)} - x^{(k-1)}}$$

$$x^{(k+1)} = x^{(k)} - \frac{F(x^{(k)})}{\Delta s}$$



```
function x = secant(x0,x1,F,rtol,atol)
fo = F(x0);
for i=1:MAXIT
  fn = F(x1);
  s = fn*(x1-x0)/(fn-fo); % correction
  x0 = x1; x1 = x1-s;
  if ((abs(s) <
     max(atol,rtol*min(abs([x0;x1])))))
    x = x1; return; end
  fo = fn;
end
```

Correction based termination

1 F-evaluation per step

No derivatives

↓

black-box suitable

Exp. : Cvg. of secant method

$F(x) = xe^x - 1\ ,\ x^{(0)} = 0\ ,\ x^{(1)} = 5$

k	$x^{(k)}$	$F(x^{(k)})$	$e^{(k)} := x^{(k)} - x^*$	$\frac{\log\lvert e^{(k+1)}\rvert - \log\lvert e^{(k)}\rvert}{\log\lvert e^{(k)}\rvert - \log\lvert e^{(k-1)}\rvert}$
2	0.00673794699909	-0.99321649977589	-0.56040534341070	
3	0.01342122983571	-0.98639742654892	-0.55372206057408	24.43308649757745
4	0.98017620833821	1.61209684919288	0.41303291792843	2.70802321457994
5	0.38040476787948	-0.44351476841567	-0.18673852253030	1.48753625853887
6	0.50981028847430	-0.15117846201565	-0.05733300193548	1.51452723840131
7	0.57673091089295	0.02670169957932	0.00958762048317	1.70075240166256
8	0.56668541543431	-0.00126473620459	-0.00045787497547	1.59458505614449
9	0.56713970649585	-0.00000990312376	-0.00000358391394	1.62641838319117
10	0.56714329175406	0.00000000371452	0.00000000134427	
11	0.56714329040978	-0.00000000000001	-0.00000000000000	

↓

Fractional (!) order of cvg. ($p \approx 1.6$)

S.M. : $x^{(k+1)} = \Phi(x^{(k)}, x^{(k-1)})$

$$\Phi(x,y) = x - \frac{x-y}{F(x)-F(y)} \cdot F(x)$$

$F(x^*) = 0$: $\Phi(x^*\, x^*) = x^*$

$$\frac{\partial\Phi}{\partial x}(x^*,x^*) = \frac{\partial\Phi}{\partial y}(x^*,x^*) = \frac{\partial^2\Phi}{\partial x^2}(x^*,x^*) = \frac{\partial^2\Phi}{\partial y^2}(x^*,x^*) = 0$$

▷ Asymptotic error recursion by Taylor exp.

$$e^{(k+1)} = \Phi(x^* + e^{(k)}, x^* + e^{(k-1)}) - x^*$$

$$= K\, e^{(k)} e^{(k-1)} + \text{" ~~small~~ term "}$$

$$\left.\begin{aligned}|e^{(k)}| &\approx C|e^{(k-1)}|^p\\ |e^{(k+1)}| &\approx C^{p+1}|e^{(k-1)}|^{p^2}\end{aligned}\right\}\text{ for order } p$$

Plug into error recursion:

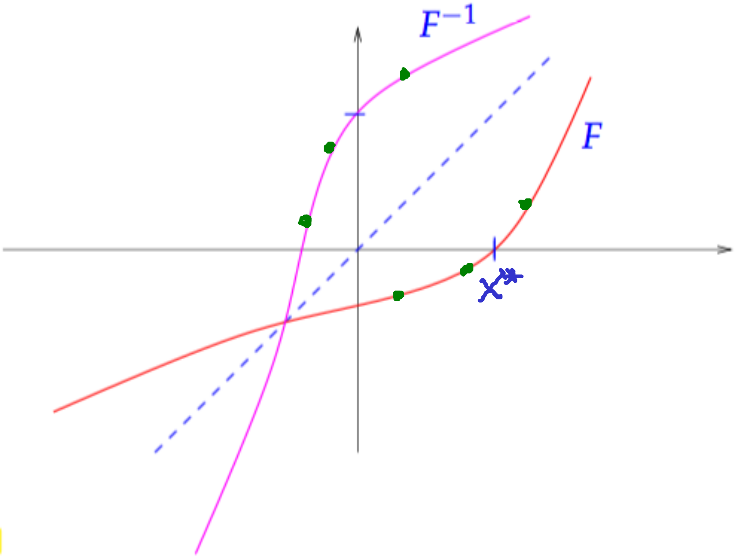
$$|e^{(k-1)}|^{p^2-p-1} \approx K^{-p}C = \text{const.} \quad \forall k$$

$$\rightarrow \quad p^2-p-1=0 \quad \Rightarrow \quad p=\tfrac{1}{2}(1+\sqrt{5}) \approx 1.6$$

More general: Inverse interpolation method

Assume: $F: I \subset \mathbb{R} \rightarrow \mathbb{R}$ strictly monoton

Idea: $F(x^*)=0 \iff F^{-1}(0)=x^*$



Interpolate F^{-1} by a polynomial p of degree $m-1$ in m points

$$p(F(x^{(k-j)})) = x^{(k-j)}$$

$$j=0,\dots,m-1$$

$$x^{(k+1)} := p(0)$$

$m=2 \Rightarrow$ secant method

$m=3 \Rightarrow$ quadratic inverse interpolation

MAPLE code:

```
p := x-> a*x^2+b*x+c;
solve({p(f0)=x0,p(f1)=x1,p(f2)=x2},{a,b,c});
assign(%); p(0);
```

→ parabola

$$\blacktriangleright \quad x^{(k+1)} = \frac{F_0^2(F_1x_2-F_2x_1)+F_1^2(F_2x_0-F_0x_2)+F_2^2(F_0x_1-F_1x_0)}{F_0^2(F_1-F_2)+F_1^2(F_2-F_0)+F_2^2(F_0-F_1)}\,.$$

$$(\,F_0 := F(x^{(k-2)}), F_1 := F(x^{(k-1)}), F_2 := F(x^{(k)}), x_0 := x^{(k-2)}, x_1 := x^{(k-1)}, x_2 := x^{(k)}\,)$$

k	$x^{(k)}$	$F(x^{(k)})$	$e^{(k)} := x^{(k)} - x^*$	$\frac{\log\lvert e^{(k+1)}\rvert-\log\lvert e^{(k)}\rvert}{\log\lvert e^{(k)}\rvert-\log\lvert e^{(k-1)}\rvert}$
3	0.08520390058175	-0.90721814294134	-0.48193938982803	
4	0.16009252622586	-0.81211229637354	-0.40705076418392	3.33791154378839
5	0.79879381816390	0.77560534067946	0.23165052775411	2.28740488912208
6	0.63094636752843	0.18579323999999	0.06380307711864	1.82494667289715
7	0.56107750991028	-0.01667806436181	-0.00606578049951	1.87323264214217
8	0.56706941033107	-0.00020413476766	-0.00007388007872	1.79832936980454
9	0.56714331707092	0.00000007367067	0.00000002666114	1.84841261527097
10	0.56714329040980	0.00000000000003	0.00000000000001	

↑ $p \approx 1.8$

2.3.3. Asymptotic efficiency

$\downarrow$ high accuracy

$\hookrightarrow = \frac{\text{gain}^{*}\,(\text{extra correct digits})}{\text{work}}$

work $\Leftrightarrow$ # F-eval. + # F'-eval. = W (per step)

$* \quad \|e^{(k)}\| \le \rho \|e^{(0)}\|$, digits gained $-\log_{10}\rho$

$\hookrightarrow \rho \ll 1$

Efficiency $= \frac{|\log \rho|}{W \cdot K(\rho)}$

$K(\rho)$ = minimal no. of steps for *

Assume iteration of order $p > 1$:

$$\|e^{(k)}\| \le C \|e^{(k-1)}\|^{p}$$

$$\|e^{(k)}\| \le \ldots\ldots C^{1+p+p^2+\dots+p^{k-1}} \|e^{(0)}\|^{p^k}$$

$*$ is guaranteed, if $C^{\frac{p^k-1}{p-1}} \|e^{(0)}\|^{p^k-1} \le \rho$

$[L_0 := C^{\frac{1}{p-1}} \|e^{(0)}\| < 1] \quad \vdots \quad K \ge \frac{\log\left(\frac{\log\rho}{\log L_0} + 1\right)}{\log p}$

$$\log p \cdot K \gtrsim \log|\log\rho| - \cancel{\log|\log L_0|}$$

$$\gtrsim \log|\log\rho|$$

Asymp. Efficiency $\approx \frac{\log p}{\log|\log\rho|} \frac{|\log\rho|}{W}$

$$\approx \underline{\underline{\frac{\log p}{W}}} \cdot \underset{\substack{\uparrow \\ \text{independent of } p \\ \text{and } W}}{\frac{|\log\rho|}{\log|\log\rho|}}$$

(p=2) (p≈1.6)

Newton vs. Secant method.

$\downarrow$

$\frac{\log 2}{2W_u} : \frac{\log 1.6}{W_u} \approx 0.7$

▷ S.M. more efficient than Newton

2.4. Newton's method

Now: Find $x^* \in \mathbb{R}^n$: $F(x^*) = 0$, $F: D \subset \mathbb{R}^n \to \mathbb{R}^n$

As before: Model function method based on local linearization.

$$F(x) \longrightarrow \widetilde{F}_k(x) = F(x^{(k)}) + \underset{\substack{\uparrow \\ \text{Jacobian} \in \mathbb{R}^{n,n}}}{DF(x^{(k)})}(x - x^{(k)})$$

Newton iteration: $x^{(k+1)} = x^{(k)} - DF(x^{(k)})^{-1}F(x^{(k)})$

```
template <typename FuncType,typename JacType,typename VecType>
void newton(const FuncType &F,const JacType &DFinv,
            VecType &x,double rtol,double atol)
{
  using index_t = typename VecType::Index;
  using scalar_t = typename VecType::Scalar;
  const index_t n = x.size();
  VecType s(n);
  scalar_t sn;
  do {
    s = DFinv(x,F(x)); // compute Newton correction
    x -= s;            // compute next iterate
    sn = s.norm();
  }
  // correction based termination (relative and absolute)
  while ((sn > rtol*x.norm()) && (sn > atol));
}
```

→ Newton correction

→ correction based t.c.

☞ Objects of type **JacType** must provide a method

z = `VecType operator (const VecType &x,const VectType &f);`

→ Solves $DF(x)z = f$

$n \gg 1$: Computation of Newton correction can be expensive, asymptotic effort $O(n^3)$

Remark: Affine invariance of Newton's method

$\underbrace{\text{Newton it. for } G_A(x)}$ $= A \cdot F(x)$, $A \in \mathbb{R}^{n,n}$ regular

$$\longrightarrow x^{(k+1)} = x^{(k)} - \underbrace{[A \cdot DF(x^{(k)})]^{-1} A F(x^{(k)})}_{DF(x^{(k)})^{-1}F(x^{(k)})}$$

$[\; DG_A(x) = A \cdot DF(x) \;]$

The same N.I. for all A!

▷ Termination criteria etc. for Newton's method should say STOP at the same index for all A!

Case study: Quasi-linear system of equations

→ Homework Sheet #5

$$A(\underline{x})\underline{x} = \underline{b}$$

"LSE with solution dependent system matrix"

$$\mathbf{A}(\mathbf{x})\mathbf{x} = \mathbf{b}\ , \quad \mathbf{A}(\mathbf{x}) = \begin{pmatrix} \gamma(\mathbf{x}) & 1 & & & & \\ 1 & \gamma(\mathbf{x}) & 1 & & & \\ & \ddots & \ddots & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & & & 1 & \gamma(\mathbf{x}) & 1 \\ & & & & 1 & \gamma(\mathbf{x}) \end{pmatrix} \in \mathbb{R}^{n\times n},$$

$$\gamma(x) = 3 + \|\underline{x}\|.$$

Sought: Newton's method for $F(\underline{x}) = A(\underline{x})\underline{x} - \underline{b} \overset{!}{=} \underline{0}$

$$\mathbf{A}(\mathbf{x})\mathbf{x} = \mathbf{T}\mathbf{x} + \mathbf{x}\|\mathbf{x}\|_2\ , \quad \mathbf{T} := \begin{pmatrix} 3 & 1 & & & & \\ 1 & 3 & 1 & & & \\ & \ddots & 3 & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & & & 1 & 3 & 1 \\ & & & & 1 & 3 \end{pmatrix}.$$

Derive:

- $\mathsf{D}\{x \to Tx\} = T \cdot I = T$ (Jacobian of a linear mapping)

- $\mathsf{D}\{\underline{x} \to \underline{x}\|x\|_2\} = \mathsf{D}\{\underline{x} \to [x_i\sqrt{x_1^2 + \ldots + x_n^2}]_{i=1}^n\}$

  → Jacobian by partial differentiation (safe & tedious)

- Product rule: $\mathsf{D}\{x \to F(x)\cdot G(x)\}h$ ← perturbation vector $h \in \mathbb{R}^n$

  $= \mathsf{D}F(x)h \cdot G(x) + F(x)\,\mathsf{D}G(x)h$

⌐ General derivatives → Analysis

$$\underset{\substack{\uparrow \\ \text{perturbation}}}{\phi(x+h)} = \phi(x) + \mathsf{D}\phi(x)h + \underset{\substack{\uparrow \\ \text{"tends} \to 0 \text{ faster than } h\text{"}}}{o(h)}$$

- Chain rule

$$\mathsf{D}\{x \to F(G(x))\}h = \mathsf{D}F(G(x))\,\mathsf{D}G(x)h$$

Apply this to $\underline{x} \to \underline{x}\cdot\|\underline{x}\|_2$

(i) Product rule: $\mathsf{D}\{\underline{x} \to \underline{x}\|x\|\}\underline{h} = \underline{I}\cdot\underline{h}\cdot\|x\| + \underline{x}\ \mathsf{D}\{\underline{x} \to \|x\|\}h$

(ii) $\|x\|_2 = \sqrt{} \circ \underline{x}^T\underline{x}$ Chain rule: $[F \leftrightarrow \sqrt{},\ G \leftrightarrow \underline{x}^T\underline{x}]$

$$\mathsf{D}\{\underline{x} \to \|\underline{x}\|\}\underline{h} = \frac{1}{2\|x\|_2}\cdot(\underbrace{h^Tx + x^Th}_{=}) = \frac{1}{\|x\|_2}x^Th$$

(inner product!)

$$D\{x \to x\|x\|\}\underline{h} = \|x\|_2 \cdot \underline{h} + \underline{x} \cdot \frac{1}{\|x\|_2}\underline{x}^T h = \underbrace{\left(\|x\|_2 \cdot I + \frac{xx^T}{\|x\|_2}\right)}_{\text{Jacobian}}\underline{h}$$

$\Rightarrow$ For $F(x) = A(\underline{x})\underline{x} - \underline{b}$

$$\underset{\uparrow}{DF(x)} = \underbrace{T + \|x\|_2 \cdot I}_{A(x)} + \underset{\uparrow}{\frac{\underline{x}\underline{x}^T}{\|x\|_2}}$$

rank-1-modification of $A(x)$

tensor product: rank-1-matrix

Case study: Matrix inversion a la Newton

$A \in \mathbb{R}^{n,n}$, regular, solves: $F(X) = 0$, $F(X) = A - X^{-1}$

$[F(A^{-1}) = A - (A^{-1})^{-1} = A - A = 0]$ $[\ F : \mathbb{R}^{n,n} \to \mathbb{R}^{n,n}\ ]$

What is DF?

$$\mathrm{Inv}(X) := X^{-1} \iff \mathrm{Inv}(X) \overset{\substack{\text{product!}\\ \downarrow}}{\cdot} X = I$$

Implicit differentiation: "$\frac{d}{dX}$": $D\mathrm{inv}(X)H \cdot X + \mathrm{Inv}(X) \cdot H = 0$

by product rule

$$\Rightarrow D\mathrm{inv}(X)H = -X^{-1} \cdot H \cdot X^{-1} = -DF(X)H$$

Newton update* S solves: $[\ DF(X^{(k)})S = F(X^{(k)})\ ]$

$\Leftrightarrow$ $(X^{(k)})^{-1}S(X^{(k)})^{-1} = A - (X^{(k)})^{-1}$, $S \in \mathbb{R}^{n,n}$

$\Leftrightarrow$ $S = X^{(k)}AX^{(k)} - X^{(k)}$

$\Rightarrow$ Newton iteration:

$$X^{(k+1)} = X^{(k)} - X^{(k)}AX^{(k)} + X^{(k)}$$
$$= X^{(k)}(2I - AX^{(k)})$$
$$X^{(k)} \longrightarrow A^{-1} \quad \text{for } K \to \infty$$

*Newton iteration: $x^{(k+1)} = x^{(k)} - \underbrace{DF(x^{(k)})^{-1}F(x^{(k)})}$

$\hat{=}$ Newton update, Newton correction

$$\underline{s} := DF(\underline{x}^{(k)})^{-1}F(\underline{x}^{(k)}) \iff DF(x^{(k)})\underline{s} = F(\underline{x}^{(k)})$$

2.4.2. Convergence of Newton's method

Newton's iteration for $F(\underline{x}) = 0$ as fixed point iteration:

$$\underline{x}^{(k+1)} = \underline{x}^{(k)} - DF(\underline{x}^{(k)})^{-1}F(\underline{x}^{(k)})$$

$$\Leftrightarrow \quad \underline{x}^{(k+1)} = \Phi(\underline{x}^{(k)}), \quad \Phi(\underline{x}) = x - DF(x)^{-1} \cdot F(x)$$

$$D\Phi(x)\underline{h} = \cancel{\underline{h}} - D\{\underline{x} \to DF(x)^{-1}\}\underline{h} \cdot F(\underline{x}) - \cancel{DF(\underline{x})^{-1}DF(x)\underline{h}}$$

[product rule !] $= -D\{\cdots\}\underline{h} \cdot F(\underline{x})$

$$F(\underline{x}^*) = 0 \Longrightarrow D\Phi(x^*) = 0$$

Lemma 2.2.18 $\Rightarrow$ $\boxed{\text{Local quadratic cvg. !}}$

2.4.3. Termination of Newton iteration

(local) quad. cvg.

$\|x^{(k+1)}-x^*\| \ll \|x^{(k)}-x^*\|$ only few step will suffice

$\Rightarrow \|x^{(k+1)}-x^{(k)}\| \approx \|x^{(k)}-x^*\|$ [$n \gg 1$: a single step is expensive]

$\Rightarrow$ Correction based termination :

In raw form: One redundant step !

▷ Idea: use simplified Newton correction

$$\Delta\bar{x}^{(k)} := DF(x^{(k-1)})^{-1}F(x^{(k)}) \rightarrow \text{Effort } O(n^2)$$

↑ LU-decomposition available

STOP, if $\|\Delta\bar{x}^{(k)}\| \le \tau_{rel}\|x^{(k)}\|$

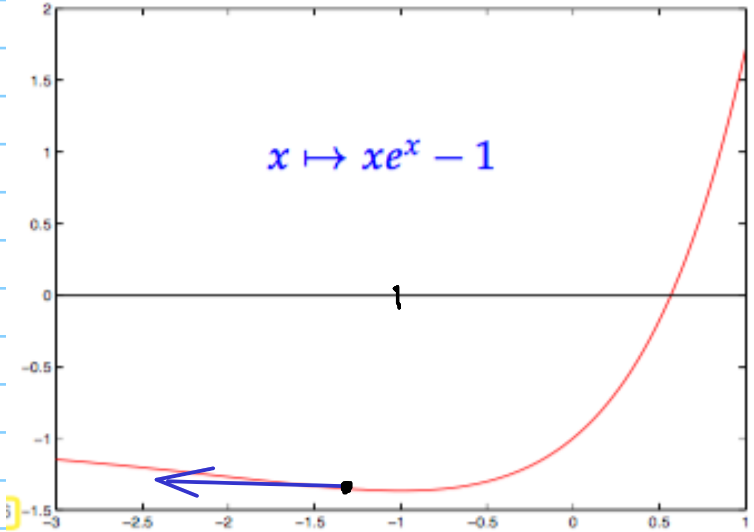
or $\|\Delta\bar{x}^{(k)}\| \le \tau_{abs}$

Note: Affine invariant, because this is true for $\Delta\bar{x}^{(k)}$!

2.4.4. Damped Newton Method

"Usually": convergence really local !

Study in 1D:



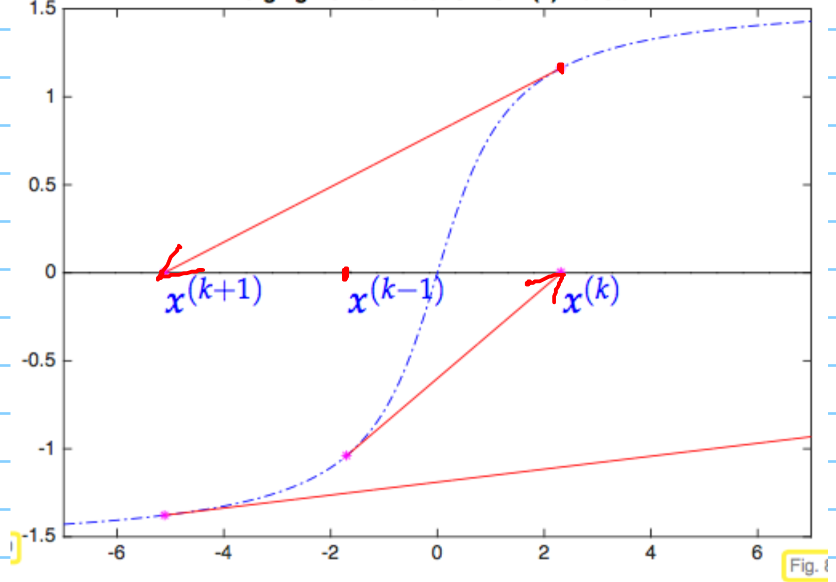
$x^{(0)} < -1$

$x^{(k)} \longrightarrow -\infty$

"wrong direction"

hopeless

Diverging Newton iteration for F(x) = arctan x



" Overshooting Newton corrections "

▶ we observe "overshooting" of Newton correction

Idea: **damping** of Newton correction:

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

Terminology: $\lambda^{(k)}$ = damping factor , $0 < \lambda^{(k)} \leq 1$

A time-tested heuristics : simplified Newton correction

Affine invariant damping strategy

Choice of damping factor: affine invariant natural monotonicity test [7, Ch. 3]:

choose "maximal" $0 < \lambda^{(k)} \leq 1$:
$$\left\|\Delta\bar{\mathbf{x}}(\lambda^{(k)})\right\| \leq \left(1 - \frac{\lambda^{(k)}}{2}\right)\left\|\Delta\mathbf{x}^{(k)}\right\|_2 \qquad (2.4.49)$$

where

$\Delta\mathbf{x}^{(k)} := \mathrm{D}F(\mathbf{x}^{(k)})^{-1}F(\mathbf{x}^{(k)})$ → current Newton correction ,

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

Practice : If (2.4.49) fails $\Rightarrow$ $\lambda \leftarrow \lambda/2$

If (2.4.49) satisfied : – 1 step with damping factor λ

– $2\cdot\lambda$ for next step

---

[*] A convergence monitor : warns user, when method fails.

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

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

We observe that damping is effective and asymptotic quadratic convergence is recovered.

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

2· 2· 2· 2·

C++11 code 2.4.50: Generic damped Newton method based on natural monotonicity test

```cpp
template <typename FuncType,typename JacType,typename VecType>
void dampnewton(const FuncType &F,const JacType &DF,
                VecType &x,double rtol,double atol)
{
  using index_t = typename VecType::Index;
  using scalar_t = typename VecType::Scalar;
  const index_t n = x.size();
  const scalar_t lmin = 1E-3; // Minimal damping factor
  scalar_t lambda = 1.0; // Initial and actual damping factor
  VecType s(n),st(n); // Newton corrections
  VecType xn(n);      // Tentative new iterate
  scalar_t sn,stn;    // Norms of Newton corrections

  do {
    auto jacfac = DF(x).lu(); // LU-factorize Jacobian
    s = jacfac.solve(F(x));   // Newton correction
    sn = s.norm();            // Norm of Newton correction
    lambda *= 2.0;
    do {
      lambda /= 2;
      if (lambda < lmin) throw "No convergence: lambda -> 0";
      xn = x-lambda*s;              // Tentative next iterate
      st = jacfac.solve(F(xn));     // Simplified Newton correction
      stn = st.norm();
      std::cout << "Inner: |stn| = " << stn << std::endl;
    }
    while (stn > (1-lambda/2)*sn); // Natural monotonicity test
    x = xn; // Now:  xn accepted as new iterate
    lambda = std::min(2.0*lambda,1.0); // Try to mitigate damping
  }
  // Termination based on simplified Newton correction
  while ((stn > rtol*x.norm()) && (stn > atol));
}
```

[*]