Root finding of nonlinear equation(s)

Task: $f(x)=0 \quad x=?$ $\qquad (y_0 \to x_0)$

$g(x)=y_0 \quad f(x)=g(x)-y_0=0 \quad x=?$

linear case: inverse function is known → direct

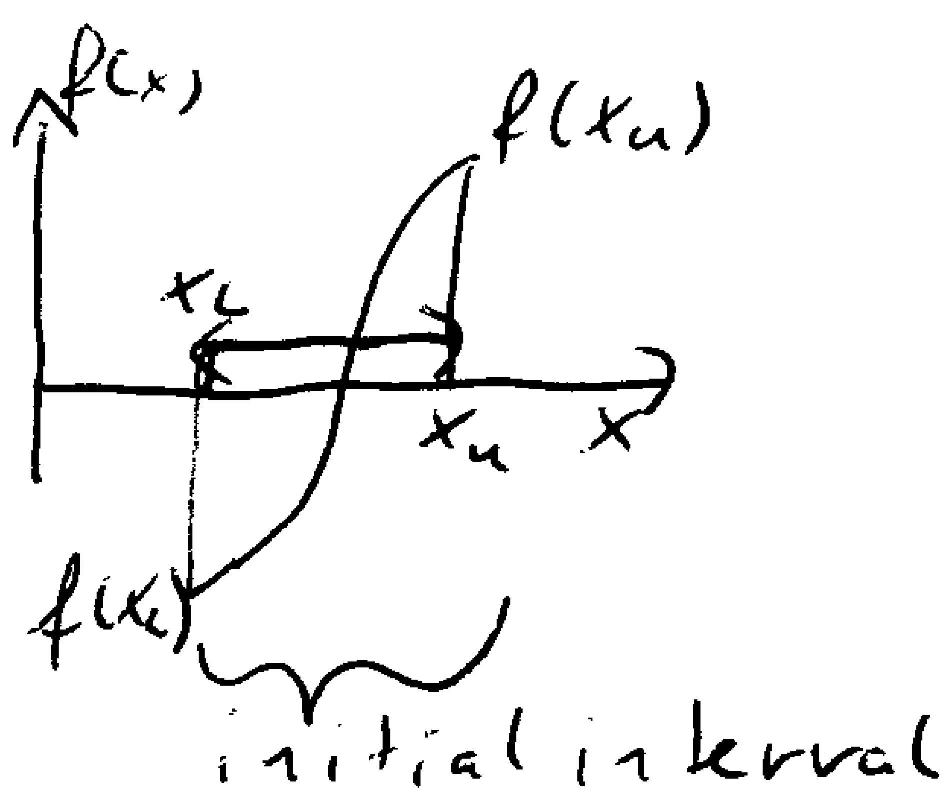
nonlinear case: inverse function is seldom known

iterative methods:

stop: $|x_i - x_{i-1}| < \varepsilon$, $|f(x_i)-f(x_{i-1})| < \varepsilon$, $i > N_{maxit.}$

Bisection – Univariate methods

bracketing of root: $f(x_L)\cdot f(x_u) < 0$



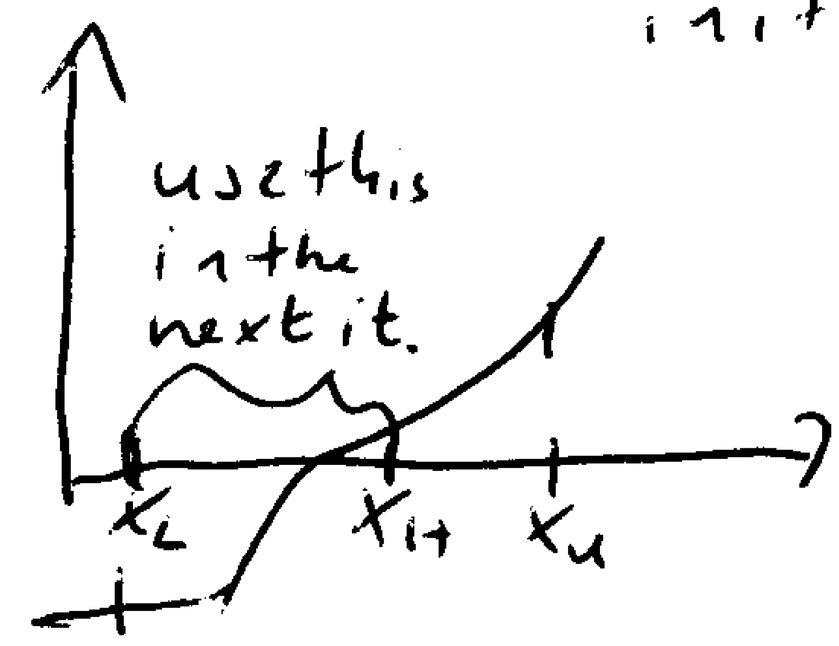
2 starting value: x_u and x_L → continuous $f(x)$ → bracketed root

$$x_H = \frac{x_u + x_L}{2}$$

$x_u = x_H$, if $f(x_L)\cdot f(x_H) < 0$, $x_L = x_H$ else

goto next x_H

$$N_{iter} = \log_2 \frac{[x_L, x_u]}{\varepsilon}$$



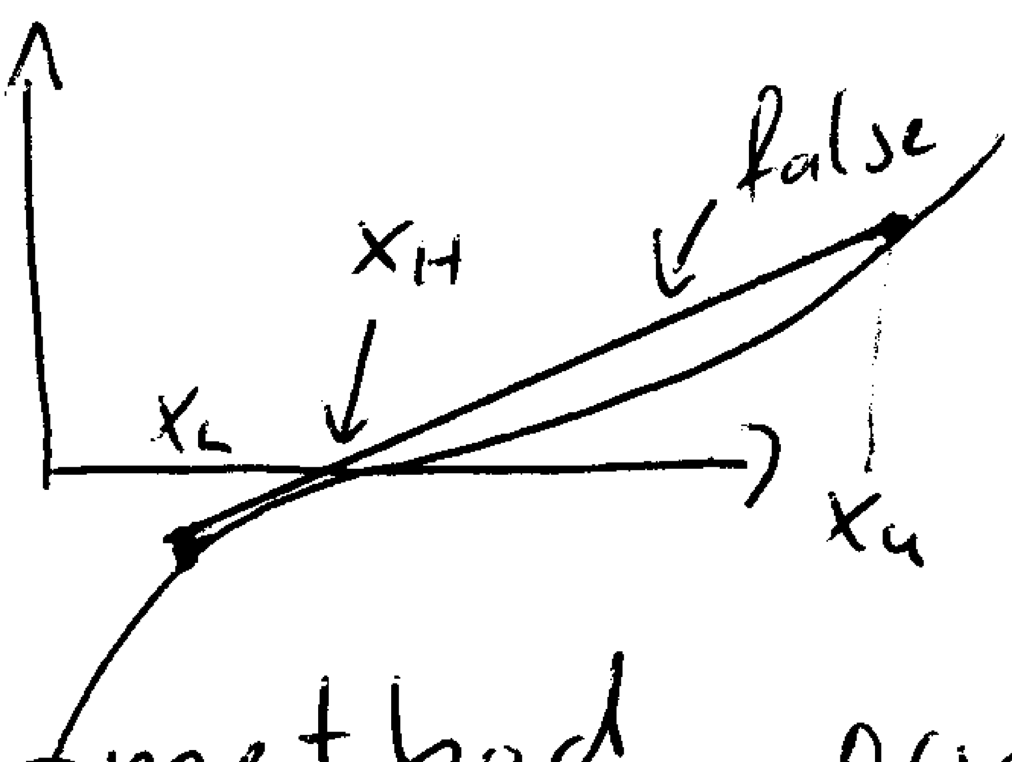
<u>safe</u> method because bracketed root

False position method

bracketing, 2 starting as x_u, x_L and $f(x_L)\cdot f(x_u) < 0$

$$x_N = \frac{x_L\cdot f(x_u) - x_u\cdot f(x_L)}{f(x_u)-f(x_L)} \quad \text{then} \quad \begin{cases} x_u = x_N & \text{if } f(x_L)\cdot f(x_N) < 0 \\ x_L = x_N & \text{else} \end{cases}$$

next x_N



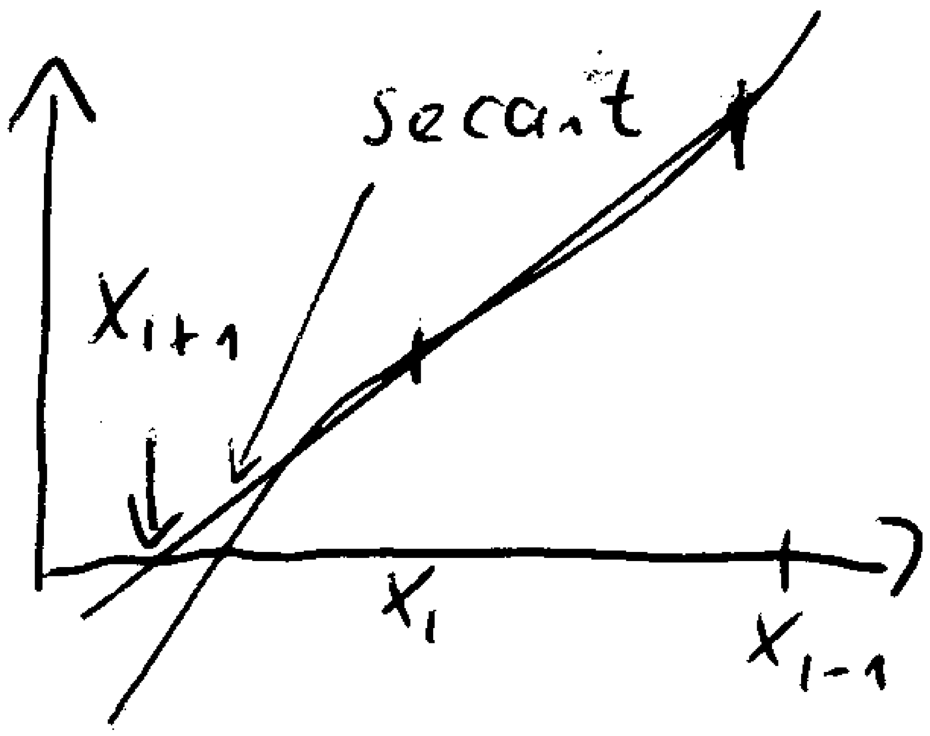
safe method – bracketed root

Secant method

$f(x_i)\cdot f(x_{i+1}) < 0$ is <u>not</u> necessary!

$$x_{i+1} = \frac{x_{i-1}\cdot f(x_i) - x_i \cdot f(x_{i-1})}{f(x_i)-f(x_{i-1})}, \quad \text{next } i$$

2 starting, not safe

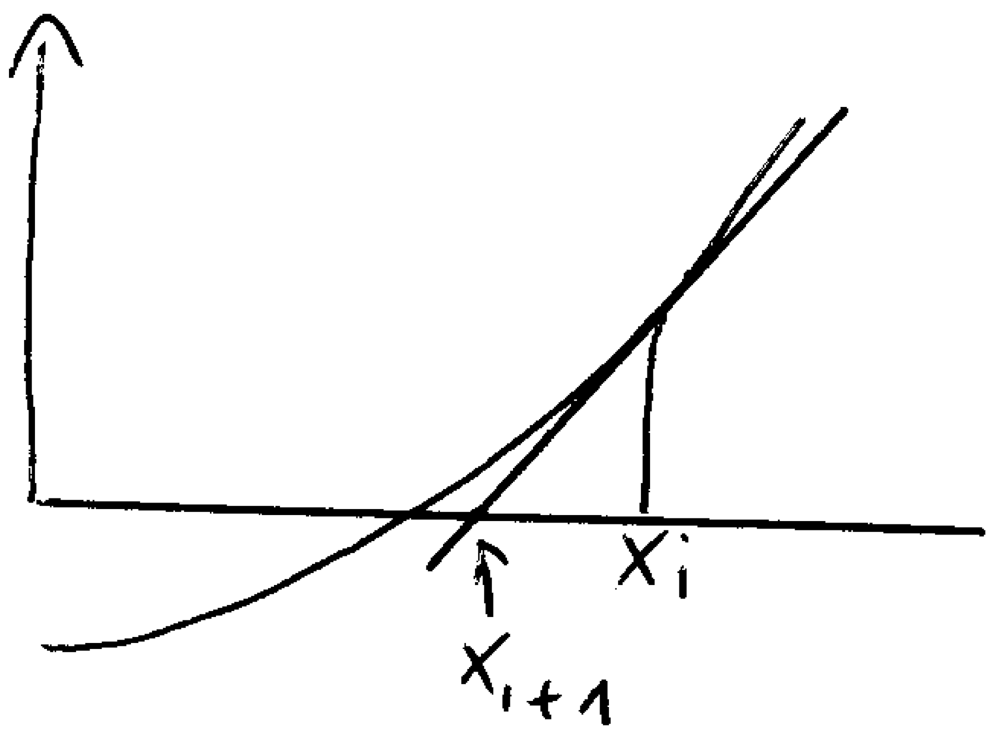


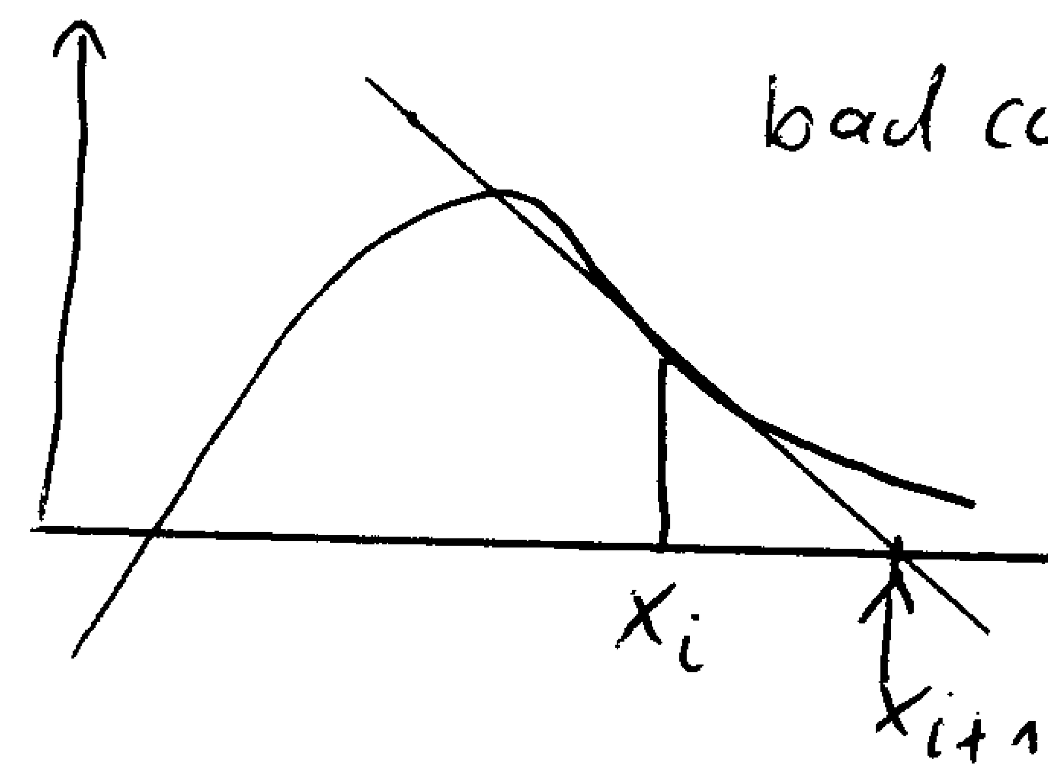
Newton-method

small secant → derivate

1 starting point, known $f(x)$ and $f'(x)$, not safe

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$





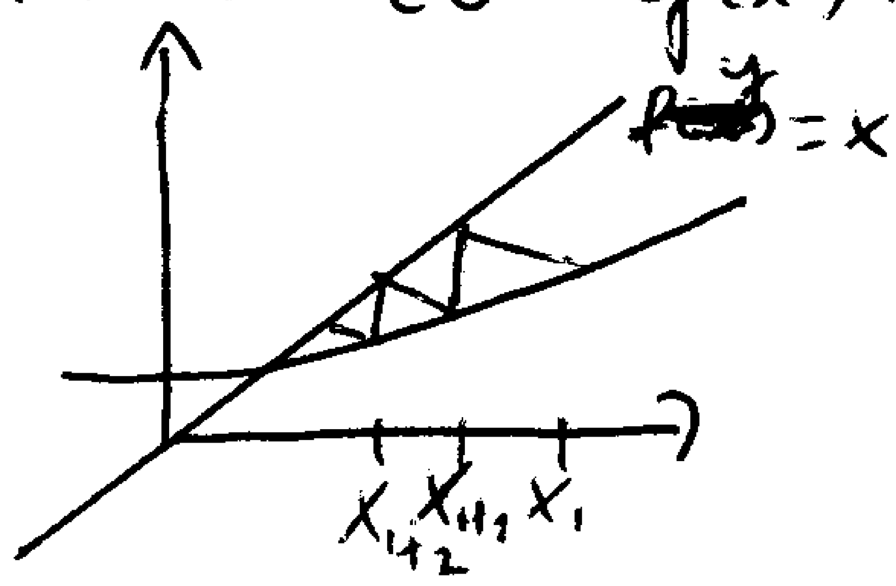
~~Wegstein-method~~

Direct iteration

rearrange $f(x)=0$ to $g(x)=x$

1 starting x

not safe



$$x_{i+1} = g(x_i)$$

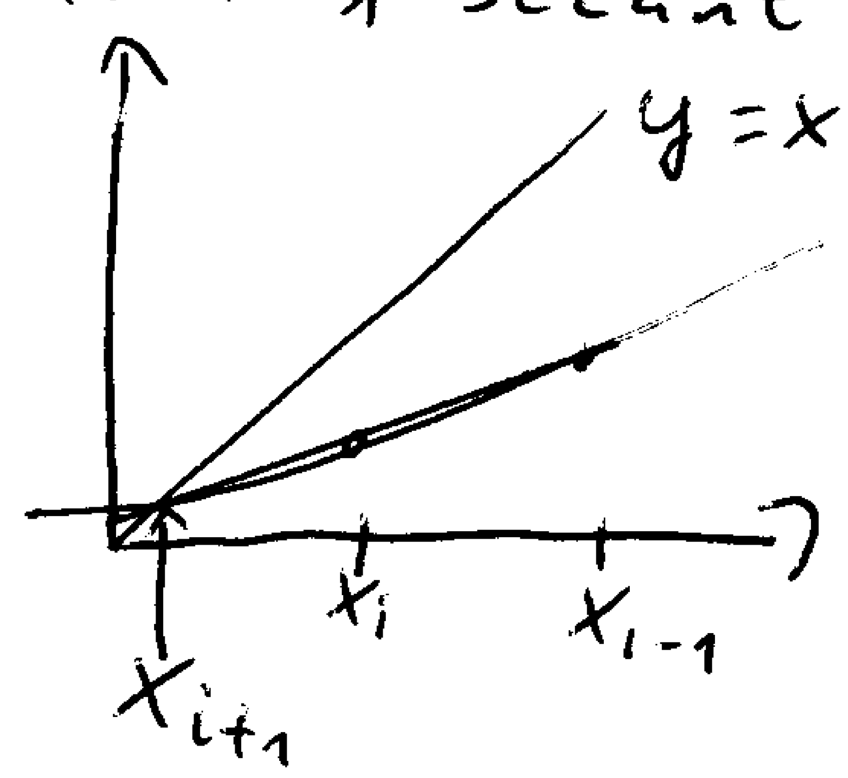
intersection of $g(x)$ and the $y=x$ line

Wegstein-method

$$x_{i+1} = (1-c)x_i + c \cdot g(x_i)$$

combination of secant + direct iteration

$$c = \frac{1}{1 - \frac{g(x_i) - g(x_{i-1})}{x_i - x_{i-1}}}$$



convergence:

error: $e = x_i - x_{exact}$

$$|e_{i+1}| \approx c \cdot |e_i|^p \quad \text{(for large } i\text{-s)}$$

method	p
Newton	2
bisection	1
direct	1
secant	1,6

$p=2 \approx\!\!>$ two times more significant places of x

For special shapes special methods: Brent, Müller, Ridders

Roots of polynomials

previous methods: we may find the same root again

solution: find a root and then eliminate the root using the root factor form of polynomials (reduce the degree of the polynomial)

$$P_n(x) = a_0 + a_1 x^1 + a_2 x^2 + \dots = \prod_{i=1}^{n} (x - \overset{\text{roots}}{x_i})$$

if x_1 is known only approximatively:

$$P_n(x) = (x - x_1)\, Q_{n-1}(x) + R \leftarrow \text{residual}$$

Newton method + reducing the deegre of the polynomial

n times (root finding, reduction)

Newton:

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$

$P_n(x)$ and $P_n'(x)$ is neccssary

synthetic division:

$$P_n(x) = Q_{n-1}(x)(x - x_1) + R \xrightarrow{x \to x_1} P_n(x) = R$$

$$P_n'(x) = (x - x_1) Q_{n-1}'(x) + 1 \cdot Q_{n-1}(x) \xrightarrow{x \to x_1} P_n'(x_1) = Q_{n-1}(x_1)$$

$$P_n(x) = \sum_{i=0}^{n} a_i x^i = (x - x_1) \sum_{i=0}^{n-1} b_i x^i + R = (x - x_1)\, Q_{n-1}(x) + R$$

using equal coefficients method:

$$\left.\begin{aligned} b_{n-1} &= a_n \\ b_{n-2} &= a_{n-1} + x_1 b_{n-1} \\ &\vdots \\ b_0 &= a_1 + x_1 b_1 \\ R &= a_0 + x_1 b_0 \end{aligned}\right\}$$

we can easily get all b_i after finding x_i

Calculate polynomials in Horner-form (cheapest way)

$$P_n(x) = (((c_n x + c_{n-1}) \cdot x + c_{n-2}) \cdot x + c_{n-3}) \dots + c_0$$

Eigenvalue method

create a „companion" matrix as $(n \times n)$

$$A = \begin{pmatrix} -\frac{a_0}{a_n} & -\frac{a_1}{a_n} & -\frac{a_2}{a_n} & \cdots & -\frac{a_{n-1}}{a_n} \\ 1 & 0 & 0 & & 0 \\ 0 & 1 & 0 & & 0 \\ & & \vdots & & \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

~~rott~~ roots = eigenvalues ← see later

safe method, good for complex ones, no need to take care on multiple ones...

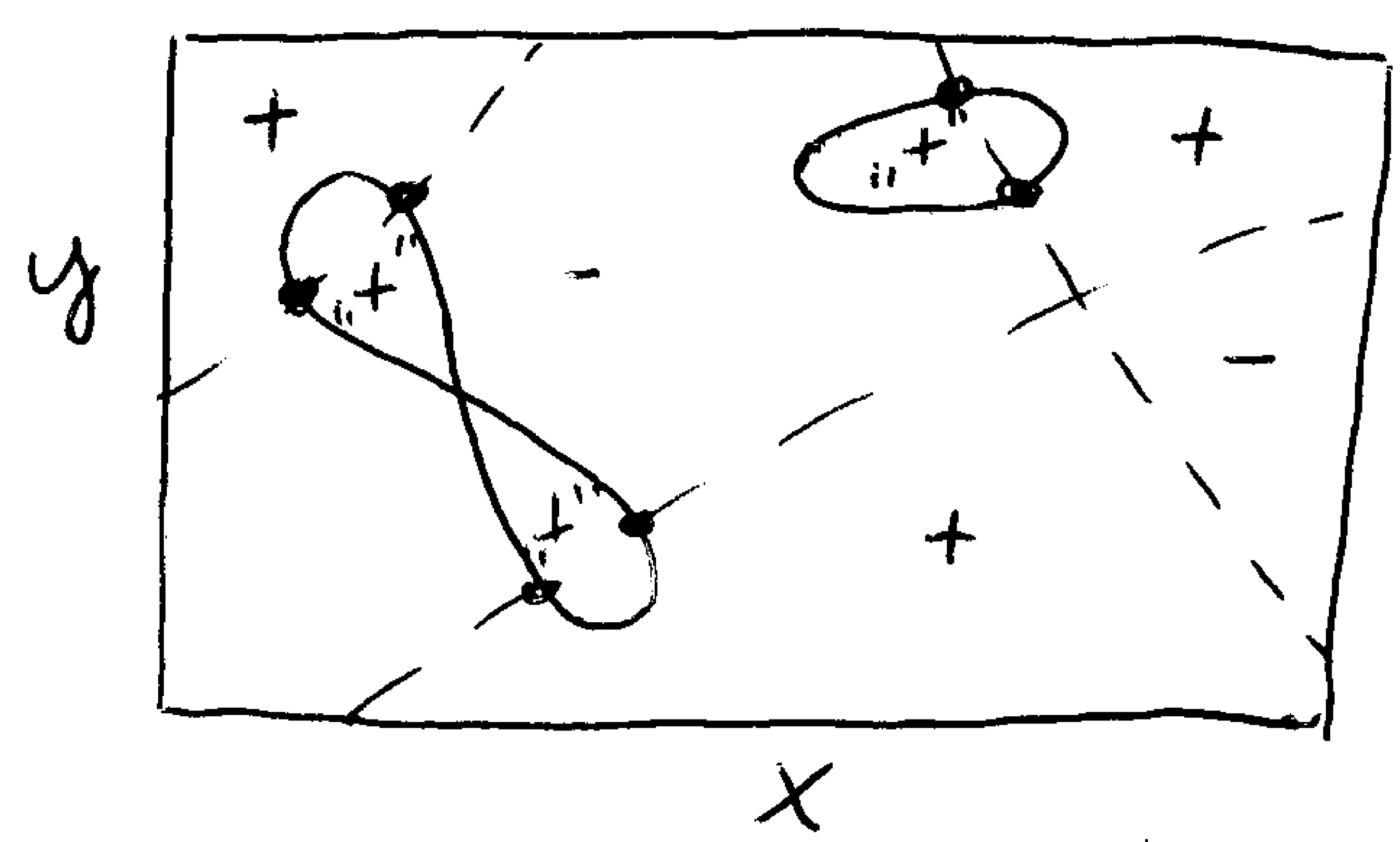
for complex use routines in e.g. Num. Recipes

Roots of nonlinear $\boxed{\text{sets}}$ of equations

$\underline{f}(\underline{x}) = 0 \quad \underline{X} = ?$ n-dimensional ones

$n=2$

$f(x,y),\ g(x,y)$



---- zero lines of $f(x)$

—— zero lines of $g(x)$

no bracketing!

small possibility of trial and error

direct iteration

$$\underline{X}^{(i+1)} = \underline{g}(\underline{x}^{(i)}) \quad \text{or} \quad \underline{X}^{(i+1)} = (1-c)\underline{X}^{(i)} + c \cdot \underline{g}(\underline{x}^{(i)})$$

$0 < c < 1$ damped

$1 < c$ over relaxed

Wegstein weights: $\underline{c} \to$ individual

$$c_i = \frac{1}{1 - \dfrac{g_i(\underline{x}^{(k)}) - g_i(\underline{x}^{(k-1)})}{x_i^{(k)} - x_i^{(k-1)}}}$$

Newton-Raphson method

1 dim: $f(x) = f(x_0) + \frac{df}{dx}(x - x_0) \rightarrow x^{k+1} = x^k - \frac{f(x^k)}{f'(x^k)}$

multidim:

$$\underline{f}(\underline{x}) = \underline{f}(\underline{x}_0) + J(\underline{x}_0)(\underline{x} - \underline{x}_0) \qquad J_{ij}(x) = \frac{\partial f_i}{\partial x_j} \text{ „Jacobian"}$$

two possibilities:

1) $\underline{x}^{k+1} = \underline{x}^k - \left(J(\underline{x}^k)\right)^{-1} \cdot \underline{f}(\underline{x}^k)$ scaled N^3 (with inversion)

2.) using linear equations solvers:

$$\underbrace{\underline{f}(\underline{x})}_{0} = \underline{f}(\underline{x}_0) + \underbrace{J(\underline{x}_0)}_{A} \cdot \underline{x} + J(\underline{x}_0) \cdot \underline{x}_0$$

$-\underline{b}$ $\underline{x}$

solve $A\underline{x} = \underline{b}$

1 possibility:
- J calculated analytically or numerically („finite difference")

backtracking: step is too large, use line search as

$$\underline{x}^{k+1} = \underline{x}^k - \lambda \cdot \left(J(\underline{x}^k)\right)^{-1} \underline{f}(\underline{x}^k)$$

optimize λ to have smallest $\left(\underline{f}(\underline{x}^{k+1})\right)^2$

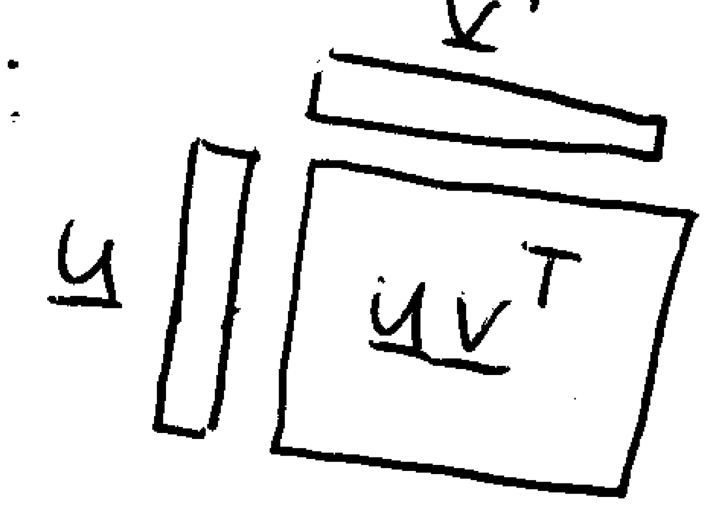
Quasi Newton methods

- use only approximate J-s
- update directly the inverse of J (J^{-1})

Using Sherman-Morrison form:

$$\left(A + \underline{u}\underline{v}^T\right)^{-1} = A^{-1} - \frac{A^{-1}\underline{u}\underline{v}^T A^{-1}}{1 + \underline{v}^T A^{-1} \underline{u}}$$

$\underline{u}\underline{v}^T$ is an outer product:



Broyden-method

$$\underline{x}^{k+1} = \underline{x}^k - \left(B^{k+1}\right)^{-1} \cdot \underline{f}(\underline{x}^k)$$

$\uparrow$ $k+1$, because we iterate first B and then $\underline{x}$

assumptions:
$$\underbrace{\underline{f}(\underline{x})^k - \underline{f}(\underline{x})^{k+1}}_{\Delta \underline{f}^k} = B^{k+1} \cdot \underbrace{(\underline{x}^{(k)} - \underline{x}^{(k+1)})}_{\Delta \underline{x}^k}$$

+ ..... secant method + SH form

iteration:

$$\left(B^{k+1}\right)^{-1} = \left(B^k\right)^{-1} - \left(\left(B^k\right)^{-1} \cdot \Delta \underline{f}^k - \Delta \underline{x}^k\right)\left(\Delta \underline{x}^k\right)^T \cdot \left(B^k\right)^{-1} \Big/ \left(\left(\Delta \underline{x}^k\right)^T \left(B^k\right)^{-1} \cdot \Delta \underline{f}^{(k)}\right)$$

initial $(B^0)^{-1}$ somehow

convergence < quadratic