Computing Zeros of Polynomials

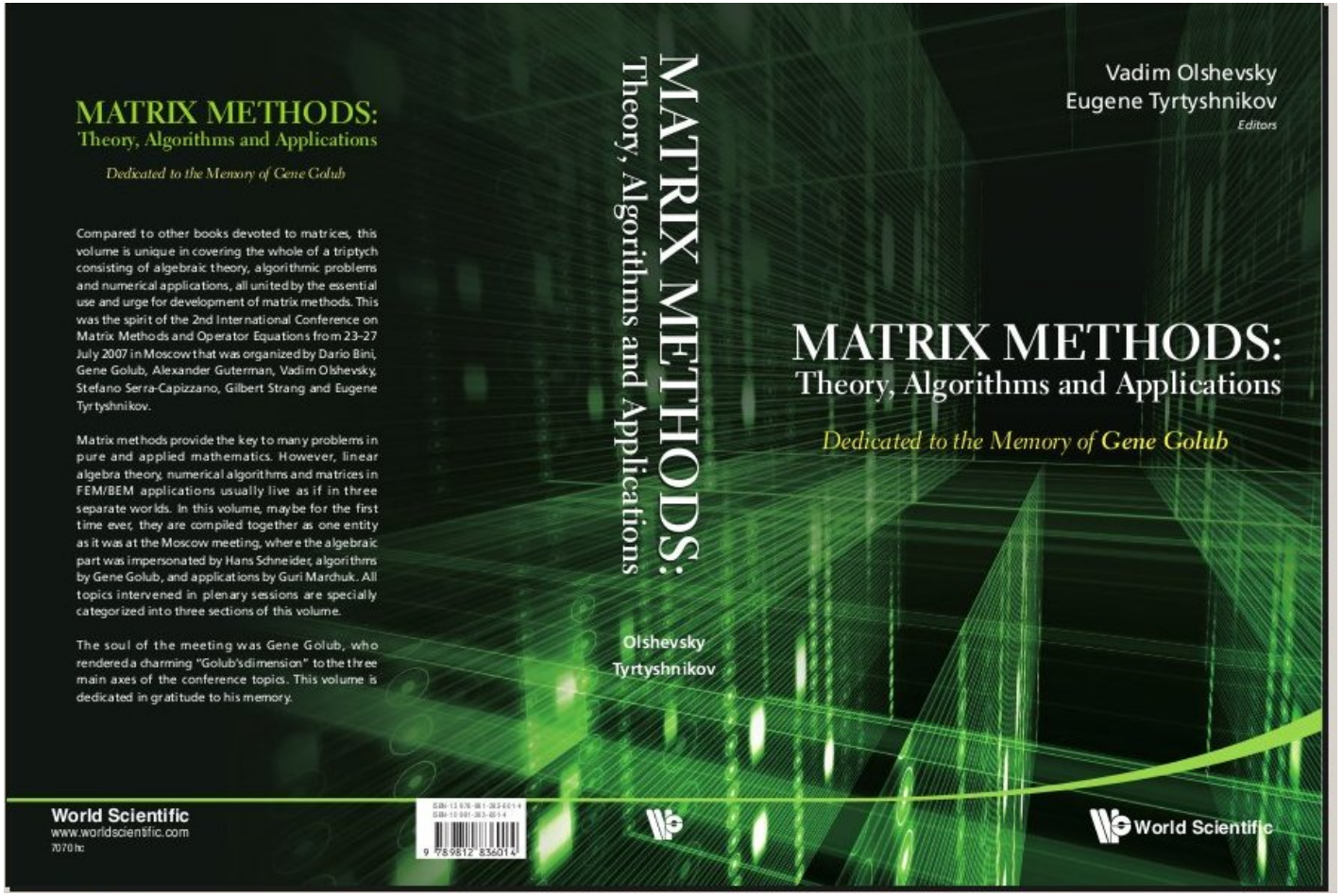
Walter Gander

ETH Zürich, visiting professor HKBU

City University Hong Kong

April 14, 2010

- 60 years Institute of Mathematics and Informatics Bulgarian Academy of Sciences, Sofia, July 6-8, 2007
- II International Conference on Matrix Methods and Operator Equations Moscow, July 22 - 28, 2007



Outline of Talk

- some historical remarks
- computing zeros from coefficients of a polynomial is ill-conditioned
- computing zeros of determinants: eigenvalues
- computing zeros of polynomials given by recurrence relations

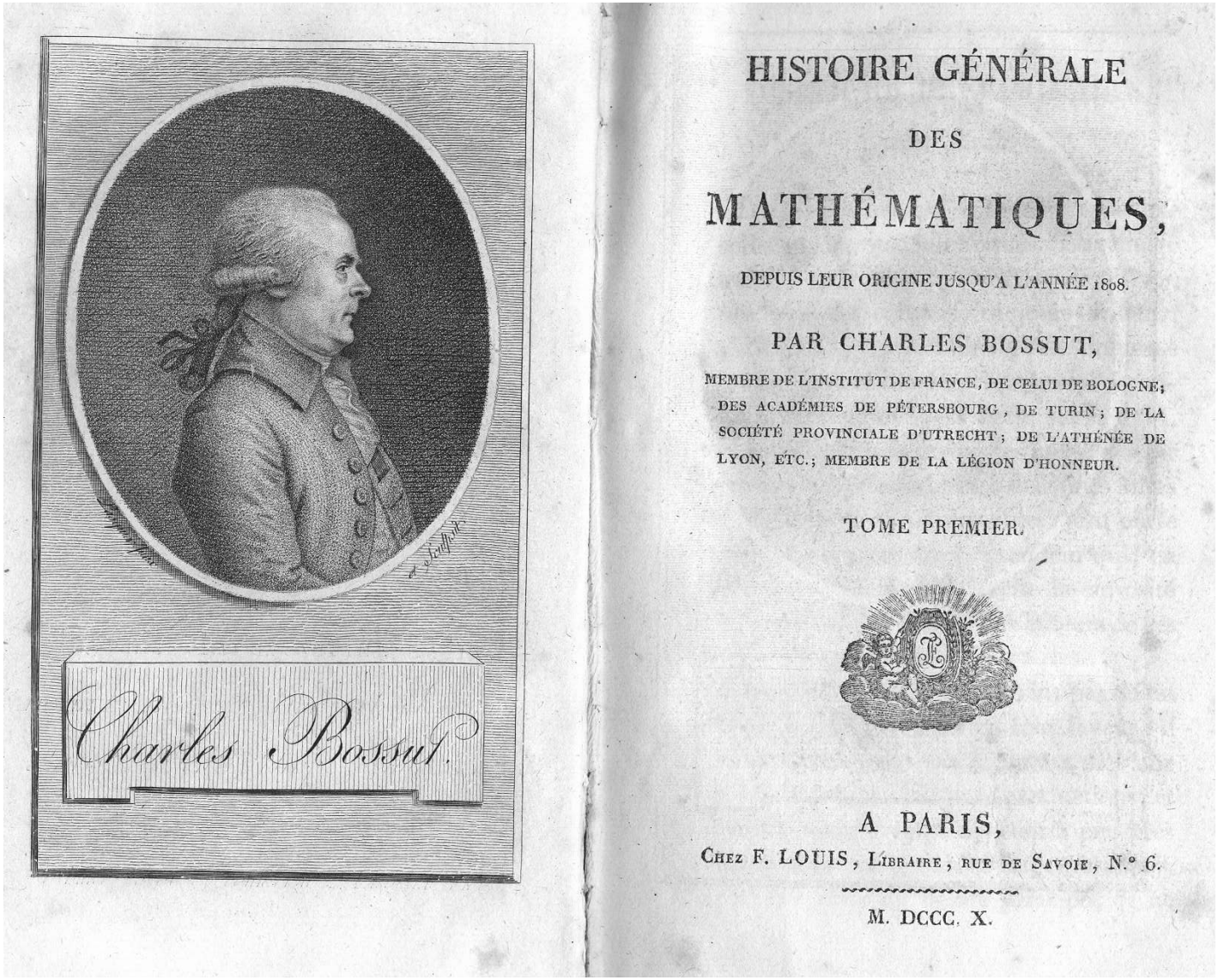
Zeros of Polynomials

- Polynomial of degree n: $P_n(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n$
  given by coefficients: $a_i \in \mathbb{C}$
- Fundamental Theorem of Algebra:
  If $n \geq 1$ and $P_n(x) \neq$ const. then $\exists$ zero $z_1 \in \mathbb{C}$ with $P_n(z_1) = 0$
  Proof by C. F. Gauss (dissertation 1799)
- Deflation $P_{n-1}(x) = P_n(x)/(x - z_1)$
- Factorization $P_n(x) = \sum_{k=0}^{n} a_k x^k = a_n \prod_{i=1}^{n} (x - z_i)$,
- How compute zeros?

Charles Bossut
(1730–1814)

General History
of Mathematics,
from the origins
to the year 1808

Paris, 1810
(2 Volumes)



According to Charles Bossut ...

- 1202 Leonardo of Pisa (Fibonacci): formulas for cubic equations
- 15th century influenced by textbook of Lucas di Borgo (Luca Pacioli) solutions for degree 1 and 2
- 1545 Gerolamo Cardano publishes formulas for cubic equations upsetting Nicolas Tartaglia who had rediscovered them and told him
- Louis (Lodovico) Ferrari (student of Cardano) develops the cubic resolvent for computing zeros of polynomials of degree 4
- But ... Bossut writes:

"Il était naturel de penser que les méthodes pour le troisième et le quatrième degrés devaient s'étendre plus loins, ou faire naître du moins de nouvelles vues sur les formes des racines dans les degrés supérieurs au quatrième. Mais si l'on excepte les équations qui, par des transformations de calcul, se reduisent en dernière analyse aux quatre premiers degrés, l'art de résoudre les équations en général et en toute rigeur n'a fait aucun progrès depuis les traveaux des Italiens que nous venons de citer."

It was plausible to think that the methods for third and fourth degrees would generalize or at least would provide new insights for roots of higher degrees. However, with the exception of equations which could be reduced to equations of the first four degrees, the art of solving equations rigorously in full generality has made no progress at all since the work of the Italians we have cited.

The Barrier

- 1824 Niels Henrik Abel establishes impossibility of solving the general quintic equation by means of radicals
- 1811 Evariste Galois is born → Galois-Theory: necessary and sufficient conditions for a polynomial to be solvable by radicals

Need Numerical Methods

- according to Bossut:
  - 1025 the Arab An-Nasavî uses Horner's scheme
  - 1303 also the Chinese Yü-Kien
  - Horner's scheme also known to Leonardo of Pisa (Fibonacci)
- Newton and Halley originally developed their methods for computing zeros of polynomials

Reduce to Zeros of Polynomials, important for hand calculations

- Goniometric equation: $\tan x = \sin x - \cos x$
- Rationalizing transformation:
  $t := \tan \frac{x}{2} \quad \sin x = \frac{2t}{1+t^2} \quad \cos x = \frac{1-t^2}{1+t^2} \quad \tan x = \frac{2t}{1-t^2}$
- $\Rightarrow \frac{2t}{1-t^2} = \frac{2t}{1+t^2} - \frac{1-t^2}{1+t^2} \iff t^4 + 4t^3 - 2t^2 + 1 = 0$
- Solutions
  complex $t_{1,2} = 0.47636446821 \pm 0.46007673544i$
  real $t_3 = -4.4391091068$ and $t_4 = -0.51361982956$

$$\begin{aligned} x_1 = 2\arctan t_3 + 2\pi k &= -2.6984428 + 2\pi k \\ x_2 = 2\arctan t_4 + 2\pi k &= -0.9489680 + 2\pi k \\ k &= 0, \pm 1 \pm 2 \ldots \end{aligned}$$

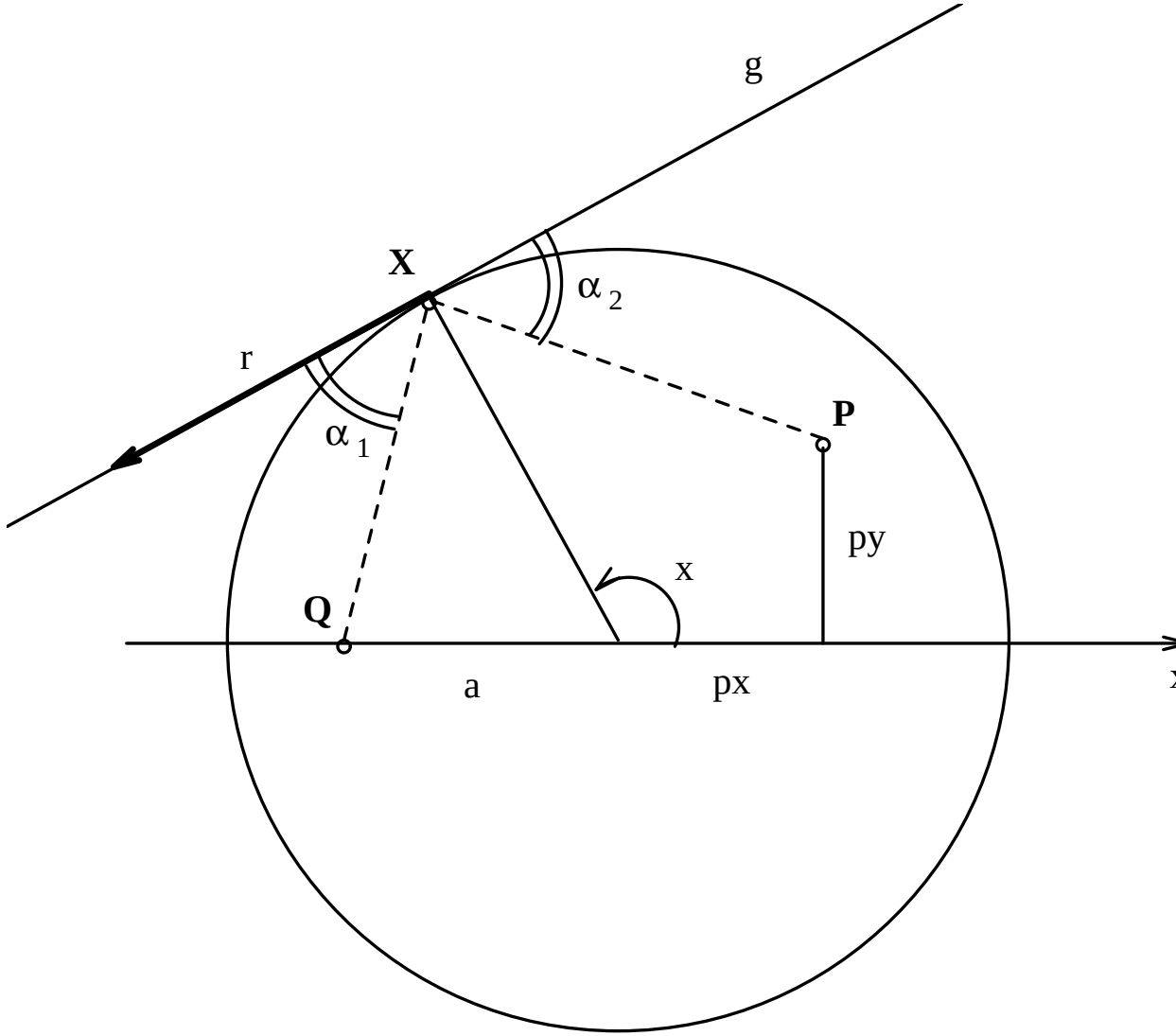
Circular Billiard

- rotational symmetry
  w.l.o.g. $\Rightarrow$ Q on x-axis
- radius w.l.o.g. $r = 1$

$$\mathbf{X} = \begin{pmatrix} \cos x \\ \sin x \end{pmatrix}$$

$$\Rightarrow \mathbf{r} = \begin{pmatrix} -\sin x \\ \cos x \end{pmatrix}$$

- law of impact
  $\alpha_1 = \alpha_2 \iff$
  $f(x) = (\mathbf{e}_{XQ} + \mathbf{e}_{XP})^T \mathbf{r} = 0$
  equation for angle x



Expanding f yields

bill.m

$$f(x) = \left(\frac{px - \cos x}{\sqrt{(px - \cos x)^2 + (py - \sin x)^2}} + \frac{a - \cos x}{\sqrt{(a - \cos x)^2 + \sin x^2}} \right) \sin x$$

$$- \left(\frac{py - \sin x}{\sqrt{(px - \cos x)^2 + (py - \sin x)^2}} - \frac{\sin x}{\sqrt{(a - \cos x)^2 + \sin x^2}} \right) \cos x = 0$$

- red: parameters
- again a goniometric equation

$\exists$ "analytical" solution?

- Rationalizing $t := \tan \frac{x}{2}$

$$f(x) = g(t) = \frac{2\,t}{1+t^2}\left(\frac{px-\frac{1-t^2}{1+t^2}}{\sqrt{\left(px-\frac{1-t^2}{1+t^2}\right)^2+\left(py-\frac{2\,t}{1+t^2}\right)^2}}+\frac{a-\frac{1-t^2}{1+t^2}}{\sqrt{\left(a-\frac{1-t^2}{1+t^2}\right)^2+\frac{4\,t^2}{(1+t^2)^2}}}\right)$$
$$-\frac{1-t^2}{1+t^2}\left(\frac{py-\frac{2\,t}{1+t^2}}{\sqrt{\left(px-\frac{1-t^2}{1+t^2}\right)^2+\left(py-\frac{2\,t}{1+t^2}\right)^2}}-\frac{2\,\frac{t}{1+t^2}}{\sqrt{\left(a-\frac{1-t^2}{1+t^2}\right)^2+\frac{4\,t^2}{(1+t^2)^2}}}\right)=0$$

More complex equation!

- However, Maple command `solve(g = 0, t);` tells t is root of polynomial of degree 4:

$$(a+1)py\,t^4+(4a\,px+2a+2px)\,t^3-6a\,py\,t^2+(-4a\,px+2a+2px)\,t+(a-1)py=0$$

Algebraic Eigenvalue Problem

- Eigenvalues of matrix A are zeros of the characteristic polynomial: $P_n(x) = \det(\lambda I - A)$
- linear algebra textbook algorithm to compute eigenvalues
  - compute the coefficients of $P_n(x)$ by expanding the determinant $\det(\lambda I - A)$
  - and compute zeros using e.g. Newton's method
- discovery of Jim Wilkinson: unstable algorithm

Rounding Errors and Condition

wilkmaple.mv

Jim Wilkinson, NPL

- tests a computer with the polynomial with zeros $1, 2, \ldots, 20$

$$P(x) = \prod_{j=1}^{20}(x-j) = x^{20} - 210x^{19} + \ldots$$

- Program computes complex zeros!
- Program correct! But rounding errors have great effect
- Computing zeros from coefficients is ill-conditioned

Condition of Zeros

change coefficient

$P(2) = -210$ a little

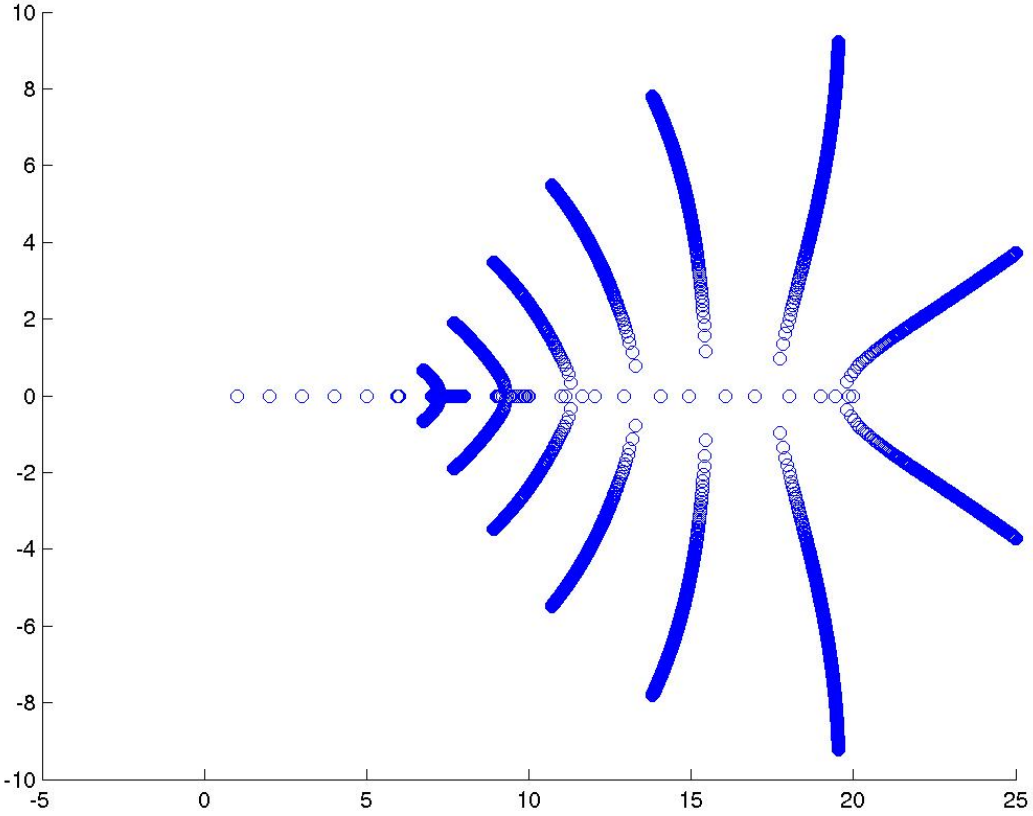
and observe zeros

```
axis([-5 25 -10 10])
hold
P=poly(1:20)
for lamb = 0: 1e-11:1e-8
  P(2) = P(2)*(1-lamb);
  Z =  roots(P);
  plot(real(Z),imag(Z),'o')
end
```

after terminating we have

$P(2) = -209.9989$

wilk.m



Evaluating the Characteristic Polynomial

- textbook algorithm computes coefficients

$$\det(\lambda I - A) \rightarrow \{c_k\} \rightarrow P(\lambda) = \sum_{k=0}^{n} c_k \lambda^k$$

- Matlab reverses computing
  - coefficients: `P = poly(A)` solves EV-problem by QR-Algorithm and expands linear factors!

$$P_n(\lambda) = (\lambda - \lambda_1)(\lambda - \lambda_2) \cdots (\lambda - \lambda_n)$$

  - roots of polynomial: `lambda = roots(P)` forms companion matrix and uses again QR-Algorithm to solve again EV-problem!

Avoiding Coefficients

given A, λ_j, want compute $P(\lambda_j)$ and $P'(\lambda_j)$

- by Gaussian elimination $C(\lambda_j) = A - \lambda_j I = LU$

$$\Rightarrow P(\lambda_j) = \det(C(\lambda_j)) = \det(A - \lambda_j I) = \det(L)\det(U) = \prod_{k=1}^{n} u_{kk}$$

- by Algorithmic Differentiation $P'(\lambda_j)$
- high computational effort – but stable!
- Newton's iteration $\lambda_{j+1} = \lambda_j - \dfrac{P(\lambda_j)}{P'(\lambda_j)}$

Computing Determinants by Gaussian Elimination

```
function f = determinant(C)
n = length(C);
f = 1;
for i = 1:n
   [cmax,kmax]= max(abs(C(i:n,i)));
   if cmax == 0  %  Matrix singular
      f = 0; return
   end
   kmax = kmax+i-1;
   if kmax ~= i
      h = C(i,:); C(i,:) = C(kmax,:); C(kmax,:) = h;
      f = -f;
   end
   f = f*C(i,i);
  % elimination step
   C(i+1:n,i) = C(i+1:n,i)/C(i,i);
   C(i+1:n,i+1:n) = C(i+1:n,i+1:n) - C(i+1:n,i)*C(i,i+1:n);
end
```

Derivative of Determinant by Algorithmic Differentiation

```
function [f,fs] = deta(C,Cs)
% DETA computes f = det(C) and derivative fs with
%      Cs = derivative of C.
n = max(size(C));
fs =0; f=1;
for i = 1:n
   [cmax,kmax]= max(abs(C(i:n,i)));
   if cmax == 0  %  Matrix singular
      f = 0; fs = 1; return
   end
   kmax = kmax+i-1;
   if kmax ~= i
      h = C(i,:); C(i,:) = C(kmax,:); C(kmax,:) = h;
      h = Cs(kmax,:); Cs(kmax,:) = Cs(i,:); Cs(i,:) = h;
      fs = -fs;  f = -f;
   end
   fs =fs*C(i,i)+f*Cs(i,i) ; f = f*C(i,i);
```

Derivative by Algorithmic Differentiation (cont.)

```
% elimination step
Cs(i+1:n,i) = (Cs(i+1:n,i)*C(i,i)-Cs(i,i)*C(i+1:n,i))/C(i,i)^2;
C(i+1:n,i) = C(i+1:n,i)/C(i,i);
Cs(i+1:n,i+1:n) = Cs(i+1:n,i+1:n) - Cs(i+1:n,i)*C(i,i+1:n)- C(i+1:n,i)*Cs(i,i+1:n);
C(i+1:n,i+1:n) = C(i+1:n,i+1:n) - C(i+1:n,i)*C(i,i+1:n);

end % for i
```

Suppression of zeros instead Deflation

We don't want to recompute already computed zeros therefore

- suppress already computed zeros: $x_1, \ldots, x_k$

$$P_{n-k}(x) := \frac{P_n(x)}{(x - x_1) \cdots (x - x_k)}$$

$$P'_{n-k}(x) = \frac{P'_n(x)}{(x - x_1) \cdots (x - x_k)} - \frac{P_n(x)}{(x - x_1) \cdots (x - x_k)} \sum_{i=1}^{k} \frac{1}{x - x_i}$$

- Newton-Maehly iteration

$$x_{new} = x - \frac{P_{n-k}(x)}{P'_{n-k}(x)} = x - \frac{P_n(x)}{P'_n(x)} \frac{1}{1 - \frac{P_n(x)}{P'_n(x)} \sum_{i=1}^{k} \frac{1}{x - x_i}}$$

Fighting Overflow (determinants soon overflow)

- we do not need $f = \det(A - \lambda I)$ and $fs = \frac{d}{d\,\lambda} \det(A - \lambda I)$
- we only want the fraction $fss = \frac{f}{fs}$
- Solution: derivative of $\log(f)$

$$lfs := \frac{d}{d\,\lambda} \log(f) = \frac{fs}{f} \quad \text{inverse Newton correction}$$

instead updating $f = f \times a_{ii}$ form logarithm $lf = lf + \log(a_{ii})$ and derivative

$$lfs = lfs + \frac{as_{ii}}{a_{ii}}$$

- Explicit expression: Formula of Jacobi $C(\lambda) = A - \lambda I$

$$lfs = \frac{fs}{f} = \text{trace}\left(C^{-1}(\lambda)\, C'(\lambda)\right)$$

Improved Algorithm $ffs = \frac{f}{f'}$

```
function ffs = deta(C,Cs)
% DETA computes Newton correction ffs = f/fs
n = length(C); lfs = 0;
for i = 1:n
   [cmax,kmax]= max(abs(C(i:n,i)));
   if cmax == 0  %  Matrix singular
      ffs = 0; return
   end
   kmax = kmax+i-1;
   if kmax ~= i
      h = C(i,:); C(i,:) = C(kmax,:); C(kmax,:) = h;
      h = Cs(kmax,:); Cs(kmax,:) = Cs(i,:); Cs(i,:) = h;
   end
   lfs = lfs + Cs(i,i)/C(i,i);
  % elimination step
   Cs(i+1:n,i) = (Cs(i+1:n,i)*C(i,i)-Cs(i,i)*C(i+1:n,i))/C(i,i)^2;
   C(i+1:n,i) = C(i+1:n,i)/C(i,i);
   Cs(i+1:n,i+1:n) = Cs(i+1:n,i+1:n) - Cs(i+1:n,i)*C(i,i+1:n)- ...
                     C(i+1:n,i)*Cs(i,i+1:n);
   C(i+1:n,i+1:n) = C(i+1:n,i+1:n) - C(i+1:n,i)*C(i,i+1:n);
end
ffs = 1/lfs;
```

Test Results random matrix with EV $1, 2, \ldots, n$

test2a.m, test2b.m

- symmetric matrix

  norm of difference: exact – computed eigenvalues

n	$roots(poly(A))$	$eig(A)$	$\det(A - \lambda I)$
50	$1.3598e{+}02$	$3.9436e{-}13$	$4.7243e{-}14$
100	$9.5089e{+}02$	$1.1426e{-}12$	$1.4355e{-}13$
150	$2.8470e{+}03$	$2.1442e{-}12$	$3.4472e{-}13$
200	$- - -$	$3.8820e{-}12$	$6.5194e{-}13$

  for $n = 200$: polynomial coefficients overflow

- non symmetric matrix

n	$roots(poly(A))$	$eig(A)$	$\det(A - \lambda I)$
50	$1.3638e{+}02$	$3.7404e{-}12$	$2.7285e{-}12$
100	$9.7802e{+}02$	$3.1602e{-}11$	$3.5954e{-}11$
150	$2.7763e{+}03$	$6.8892e{-}11$	$3.0060e{-}11$
200	$- - -$	$1.5600e{-}10$	$6.1495e{-}11$

Generalization: λ-Matrix Example from Tisseur and Meerbergen: Quadratic Eigenvalue Problem, SIAM. Rev. 2001

- $$Q(\lambda) = \begin{pmatrix} \lambda+1 & 6\lambda^2-6\lambda & 0 \\ 2\lambda & 6\lambda^2-7\lambda+1 & 0 \\ 0 & 0 & \lambda^2+1 \end{pmatrix} = \lambda^2 M + \lambda C + K$$

  $$M = \begin{pmatrix} 0 & 6 & 0 \\ 0 & 6 & 0 \\ 0 & 0 & 1 \end{pmatrix} \text{ singular}, \quad C = \begin{pmatrix} -1 & -6 & 0 \\ 2 & -7 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad K = I$$

- $P(\lambda) = \det(Q(\lambda)) = \det(M)\lambda^6 + \cdots$ has only degree 5
  $= -6\lambda^5 + 11\lambda^4 - 12\lambda^3 + 12\lambda^2 - 6\lambda + 1$

- $\Rightarrow$ quadr. EV-Problem has 5 finite + one infinite eigenvalue

Reverse Polynomial

- Nonlinear EV-Problem: $Q(\lambda)\mathbf{x} = \lambda^2 M\mathbf{x} + \lambda C\mathbf{x} + K\mathbf{x} = 0$

  $P(\lambda) = \det(Q(\lambda)) = \det(M)\lambda^6 + \cdots$

- change of variable: $\mu = 1/\lambda$

$$\tilde{Q}(\mu)\mathbf{x} := \mu^2 Q\left(\frac{1}{\mu}\right)\mathbf{x} = M\mathbf{x} + \mu C\mathbf{x} + \mu^2 K\mathbf{x} = 0$$

  Now $\tilde{P}(\mu) = \det(\tilde{Q}(\mu)) = \det(K)\mu^6 + \cdots$
  of degree 6 if $\det(K) \neq 0$

- zero EV of $\tilde{Q}(\mu)$ are infinite EV of $Q(\lambda)$

- reverse polynomial

$$\tilde{P}(\mu) = \mu^6 P\left(\frac{1}{\mu}\right)$$

Solving Quadratic Eigenvalue Problems

1. Reduce to general EV-Problem by "linearization" (if $\det(M) \neq 0$):

$$P(\lambda) = \det(\lambda^2 M + \lambda C + K) = 0 \iff \det\left(\begin{bmatrix} M & 0 \\ 0 & K \end{bmatrix} - \lambda \begin{bmatrix} 0 & M \\ -M & -C \end{bmatrix}\right) = 0$$

2. Brute force: Newton's iteration for $P(\lambda) = 0$

   - compute correction $\frac{P(\lambda)}{P'(\lambda)}$ using function `PPs = deta(Q,Qs)` with

$$Q(\lambda) = \begin{pmatrix} \lambda + 1 & 6\lambda^2 - 6\lambda & 0 \\ 2\lambda & 6\lambda^2 - 7\lambda + 1 & 0 \\ 0 & 0 & \lambda^2 + 1 \end{pmatrix} \Rightarrow Qs(\lambda) = \begin{pmatrix} 1 & 12\lambda - 6 & 0 \\ 2 & 12\lambda - 7 & 0 \\ 0 & 0 & 2\lambda \end{pmatrix}$$

   - Want all solutions: have to know the degree of $P(\lambda)$

Example 1

test4a.m

```
clear, format compact, format short e
n=5,
for k=1:n
  lam = rand(1)*i; ffs = 1; q=0;
  while abs(ffs)>1e-14
    Q = [lam+1 6*lam^2-6*lam    0
        2*lam  6*lam^2-7*lam+1  0
         0          0        lam^2+1];
    Qs = [1  12*lam-6   0
          2  12*lam-7   0
          0     0    2*lam];
    ffs = deta(Q,Qs)
    s = 0;
    if k>1
       s = sum(1./(lam-lamb(1:k-1)));
    end
    lam = lam-ffs/(1-ffs*s); q=q+1;
  end
  [k, q], lam, lamb(k) = lam;
end
lamb = lamb(:)
```

k	λ_k	q
1	$0 + 1.0000e{+}00i$	6
2	$3.3333e{-}01 - 4.4366e{-}32i$	9
3	$5.0000e{-}01 - 4.5139e{-}36i$	8
4	$1.0000e{+}00$	8
5	$-1.6816e{-}44 - 1.0000e{+}00i$	2

Example 2: Reverse Polynomial

test4b

k	μ_k	q	$\lambda = 1/\mu$
1	$-3.1554e{-}29 - 1.0000e{+}00i$	5	i
2	$9.1835e{-}41 - 9.1835e{-}41i$	7	∞
3	$1.0000e{+}00$	8	1
4	$2.0000e{+}00 - 2.4074e{-}35i$	9	$1/2$
5	$0 + 1.0000e{+}00i$	8	$-i$
6	$3.0000e{+}00 - 5.9165e{-}31i$	2	$1/3$

Mass-spring system example (Tisseur-Meerbergen)

test5a.m

$$\det(Q(\lambda)) = 0$$

where

$$Q(\lambda) = \lambda^2 M + \lambda C + K$$

$$M = I$$

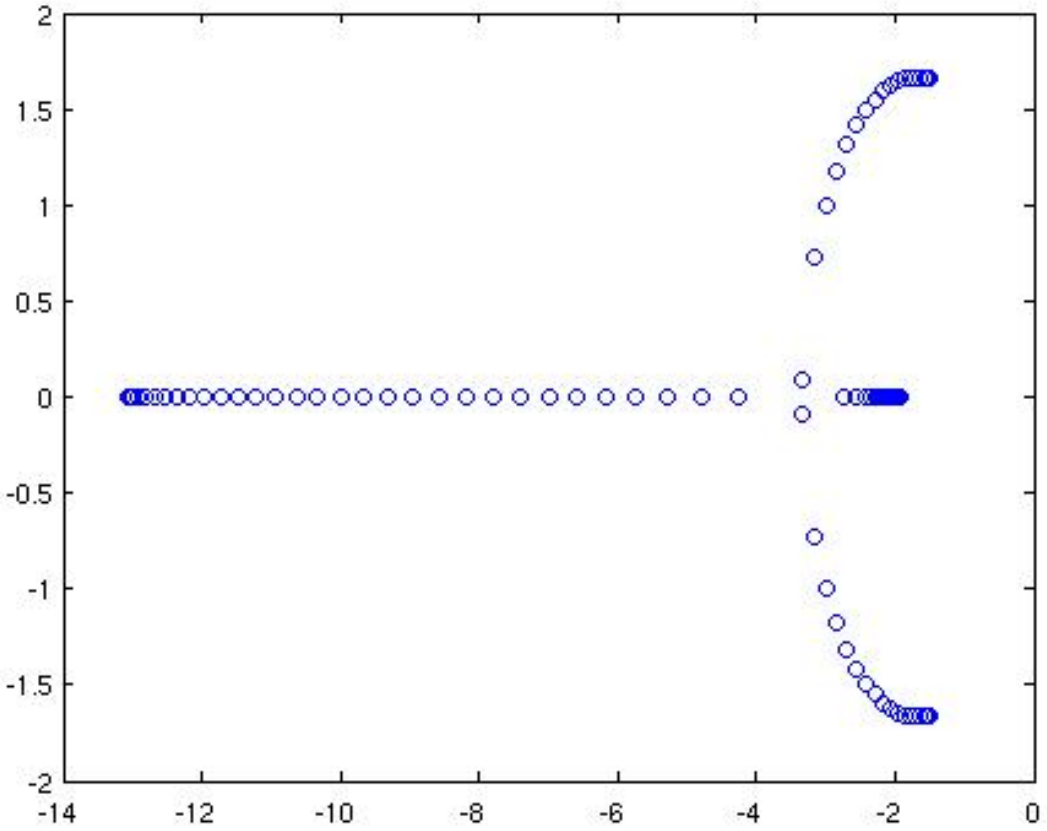
$$C = \tau\,\mathrm{tridiag}(-1, 3, -1)$$

$$K = \kappa\,\mathrm{tridiag}(-1, 3, -1)$$

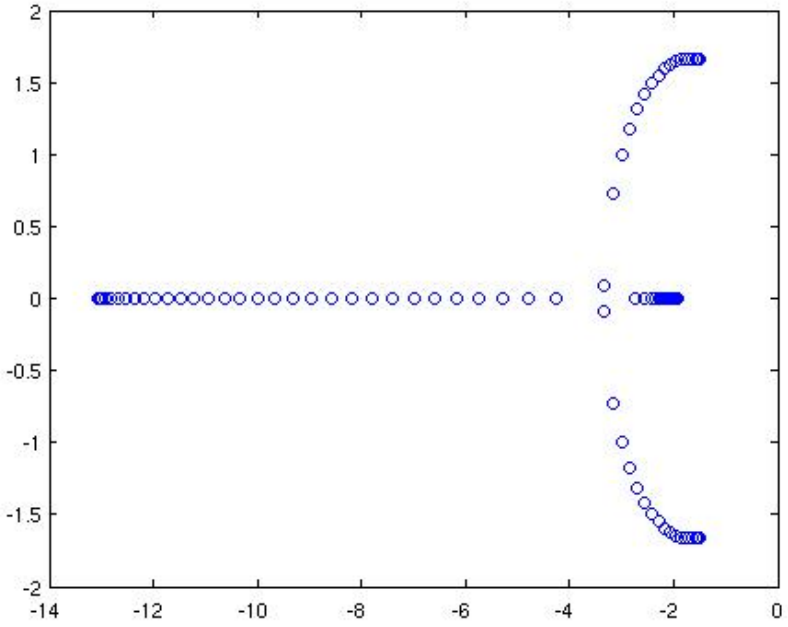
$\kappa = 5,\ \tau = 3$

nonoverdamped

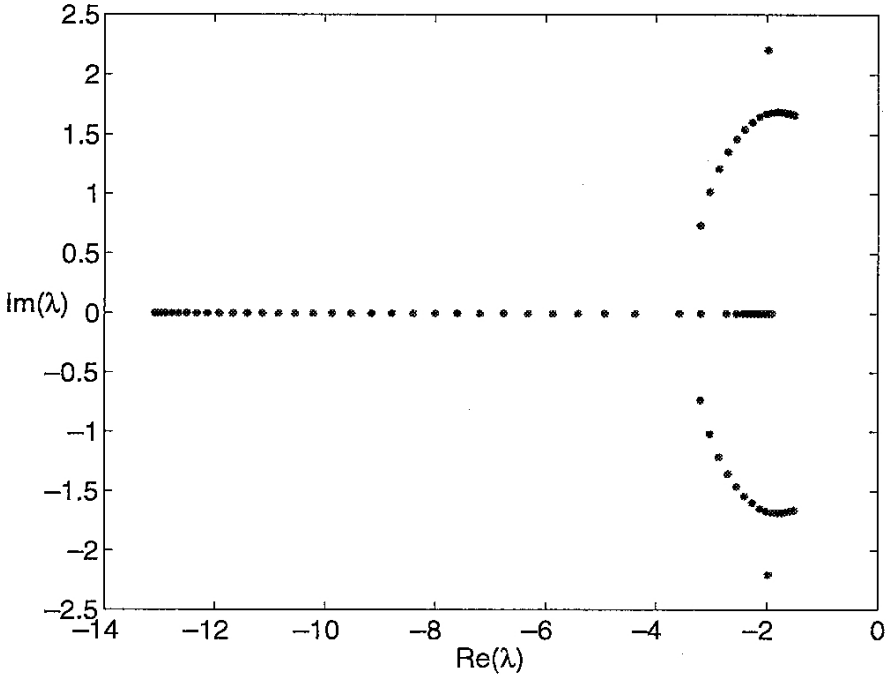
$n = 50$, 100 Eigenvalues



Our Result



Tisseur-Meerbergen (wrong picture!)



Example cont.

test5b.m

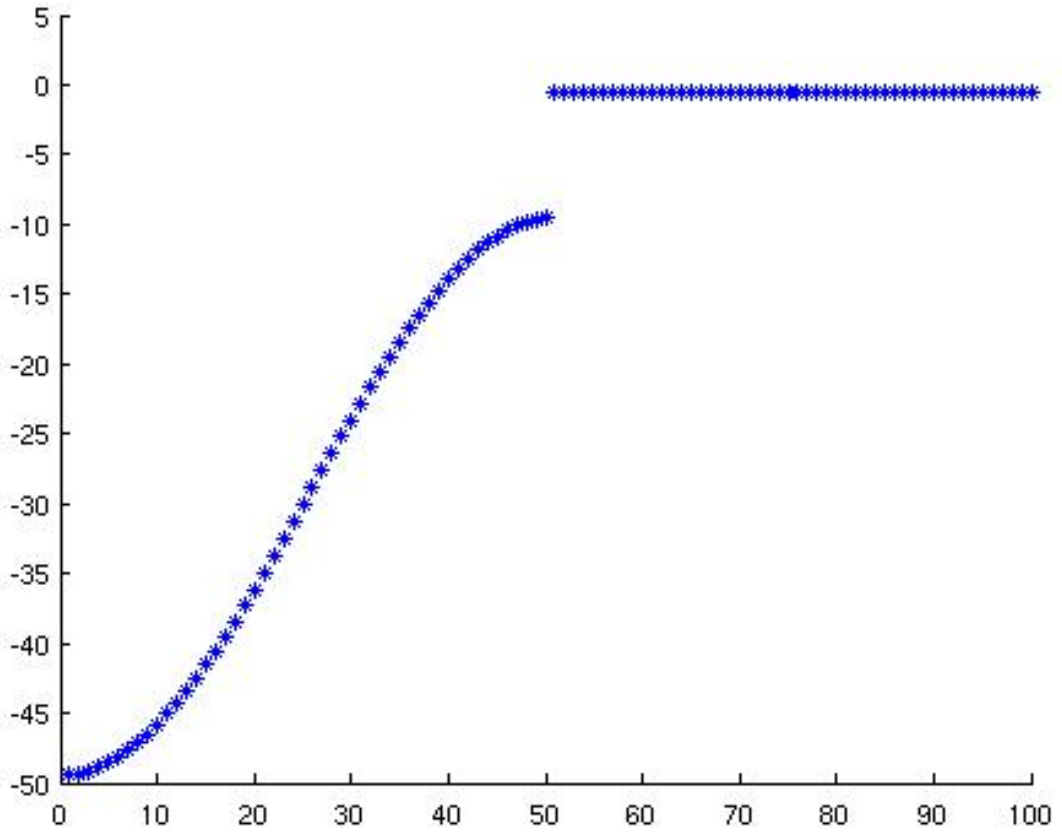
Eigenvalues of mass-spring system

$$Q(\lambda) = \lambda^2 M + \lambda C + K$$
$$M = I$$
$$C = \tau\,\mathrm{tridiag}(-1, 3, -1)$$
$$K = \kappa\,\mathrm{tridiag}(-1, 3, -1)$$

$\kappa = 5$, $\tau = 10$

overdamped

$n = 50$, 100 Eigenvalues



Cubic Eigenvalue Problem (Arbenz/Gander)

test6.m

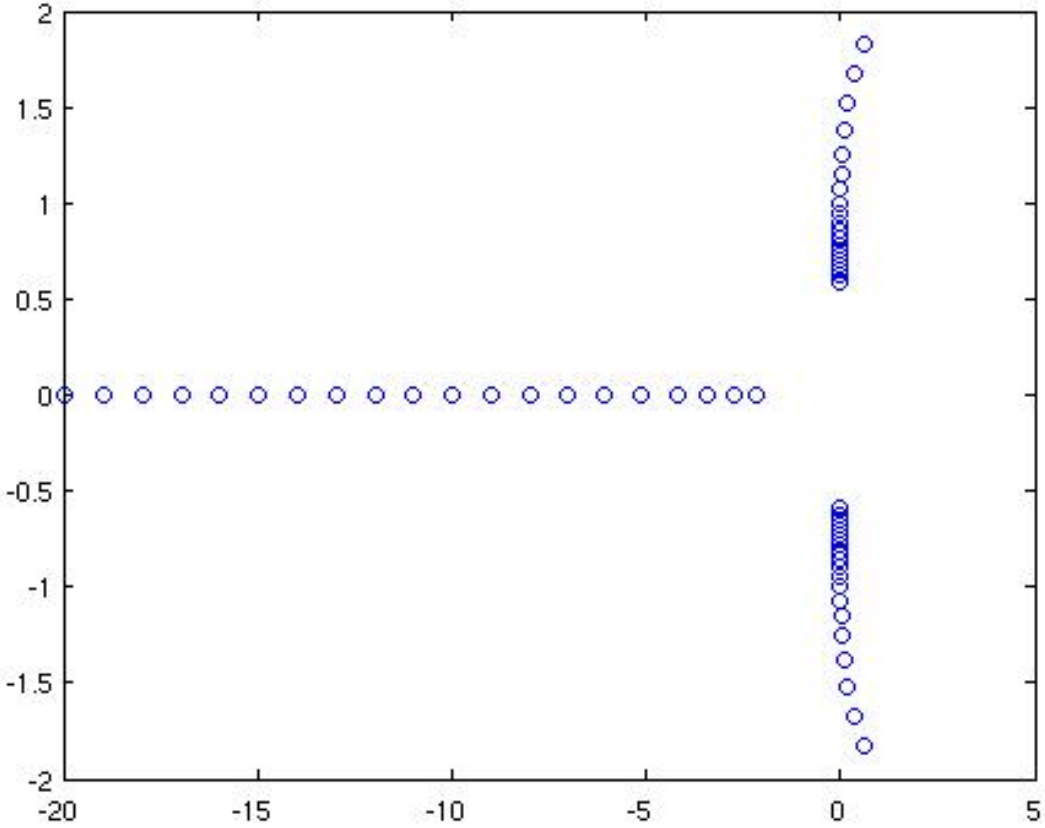
$$Q(\lambda) = \lambda^3 A_3 + \lambda^2 A_2 + \lambda A_1 + A_0$$

$$A_0 = \begin{pmatrix} 8 & 1 & & \\ 1 & 8 & \ddots & \\ & \ddots & \ddots & 1 \\ & & 1 & 8 \end{pmatrix}$$

$$A_2 = \mathrm{diag}(1, 2, \ldots, n)$$

$$A_1 = A_3 = I$$

$n = 20 \Rightarrow 60$ Eigenvalues



Orthogonal Polynomials, Tridiagonal Matrices

- three-term recurrence relation for orthogonal polynomials:

  $p_{-1}(x) \equiv 0,\ p_0(x) \equiv 1,\ \beta_0 = 0$

  $$p_{k+1}(x) = (x - \alpha_{k+1})p_k(x) - \beta_k p_{k-1}(x), \quad k = 0, 1, 2, \ldots$$

- coefficients $\alpha_k,\ \beta_k$
- p_k is of degree k
- differentiating yields recurrence for derivative $p'_k(x)$:

  $$\begin{aligned} p'_0(x) &= 0, \quad p'_1(x) = 1 \\ p'_{k+1}(x) &= p_k(x) + (x - \alpha_{k+1})p'_k(x) - \beta_k p'_{k-1}(x) \quad k = 1, 2, \ldots \end{aligned}$$

- uses recurrences to evaluate Newton correction p_n/p'_n

evalpol avoids under–/overflow by scaling.

Idea of Heinz Rutishauser for evaluating continued fractions. (Algol 60 book)

```
function r = evalpol(x,alpha,beta);
% EVALPOL2  evaluates the three-term recurrence with
%           its derivative with rescaling and returns
%           the ratio
n = length(alpha);
p1s = 0; p1 = 1;
p2s = 1; p2 = x-alpha(1);
for k = 1:n-1
   p0s =p1s; p0=p1;
   p1s = p2s; p1 =p2;
   p2s = p1 + (x-alpha(k+1))*p1s - beta(k)*p0s;
   p2 = (x-alpha(k+1))*p1 - beta(k)*p0;
   maxx = abs(p2)+abs(p2s);
   if maxx > 1e20;
      d = 1e-20;
   elseif maxx < 1e-20
      d = 1e20;
   else
      d = 1;
   end
   p1 = p1*d; p2 =p2*d;
   p1s = p1s*d; p2s = p2s*d;
end
r = p2/p2s;
```

Gauss Quadrature: Algorithm by Golub-Welsch

- nodes for quadrature formula = zeros of orthogonal polynomials
  = eigenvalues of a symmetric tridiagonal matrix

$$T = \begin{pmatrix} a_1 & b_1 & & \\ b_1 & a_2 & \ddots & \\ & \ddots & \ddots & b_{n-1} \\ & & b_{n-1} & a_n \end{pmatrix}$$

- eigenvalues are real and simple if T unreduced ($b_i \neq 0$)
- characteristic polynomial $f_n(\lambda) = \det(\lambda I - T)$ is computed by three-term recurrence

$$\begin{aligned} f_0(\lambda) &= 1, \quad f_1(\lambda) = \lambda - a_1 \\ f_j(\lambda) &= (\lambda - a_j) f_{j-1}(\lambda) - b_{j-1}^2 f_{j-2}(\lambda), \quad j = 2, 3 \ldots, n \end{aligned}$$

Specialization for Real Simple Zeros

clever implementation of Newton-Maehly by Bulirsch-Stoer

- Newton has bad global convergence

$$f(x) = c_n x^n + \cdots + c_0, \quad f'(x) = n c_n x^{n-1} + \cdots + c_1$$

Newton step $$x_{new} = x - \frac{f(x)}{f'(x)} \approx x - \frac{c_n x^n}{n c_n x^{n-1}} = \frac{n-1}{n} x$$

- start iterating from right with double steps

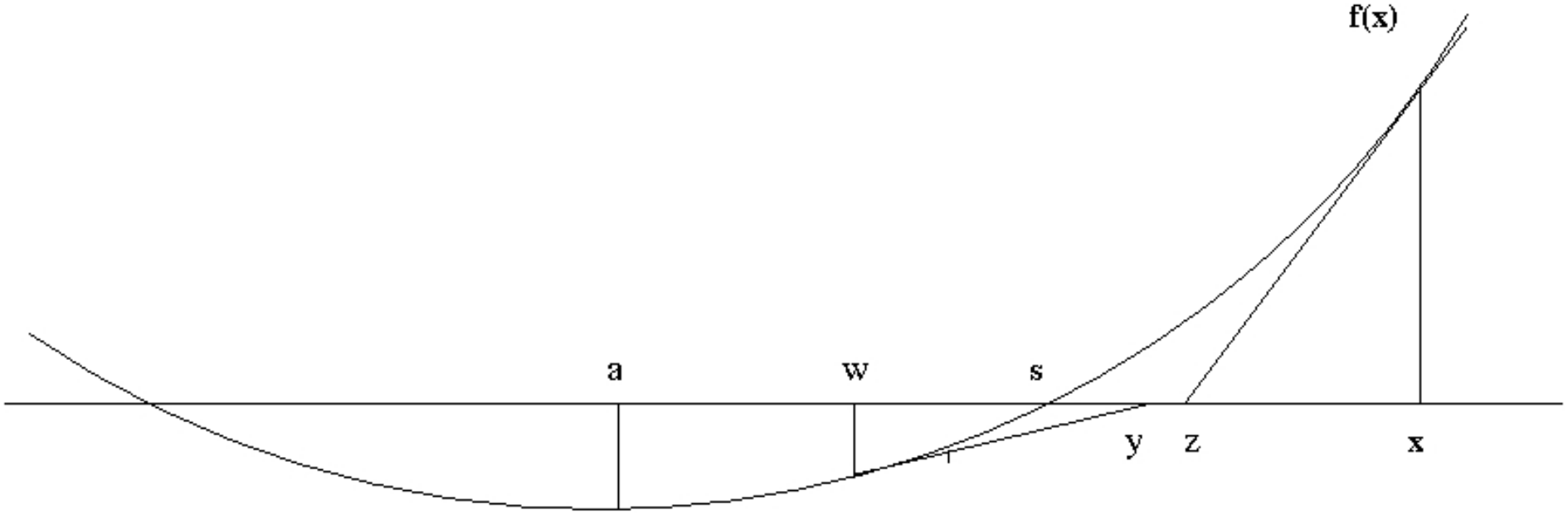
$$x = w; \quad w = x - 2\,\frac{f(x)}{f'(x)};$$

until monotonicity is lost: $w > x$

Bulirsch-Stoer-Maehly (cont.)

Theorem $a < w$ Newton double step w before local extremum a

$s \leq y \leq z$ backward step y between zero s and Newton step z



Algorithm

1. Start with Newton double step

$$x := y; \quad y := x - 2\frac{f(x)}{f'(x)}$$

   until numerically $y \geq x$

2. Continue iteration with normal Newton steps

$$x := y; \quad y := x - \frac{f(x)}{f'(x)}$$

   until $y \geq x$. Zero $s = y$ is computed to machine precision

3. Suppress new zero s and continue iteration with $x = w < s$ found by Newton double step

Elegant machine-independent stopping criterium!

Bulirsch-Stoer Variant of Newton-Maehly

```
function xi = newtonmaehly(evalpol, alpha,beta,K,x0);
xi=[];
for k = 1 : K,
   y = x0;  x = y+10;
   while y<x   % Newton double step
       x = y; r = evalpol(x,alpha,beta);
       s = sum(1./(x-xi(1:k-1)));  % Maehly
       y = x - 2*r/(1-r*s);
   end
   x0 = x;
   % single backward step
   y  = x - r/(1-r*s); x = y+10;
   while y<x   %  Newton single steps
       x = y; r = evalpol(x,alpha,beta);
       y = x - r/(1-r*sum(1./(x-xi(1:k-1))));
   end
   xi(k) = y;  k, y
end
xi = xi(:);
```

Results for 1-dim Laplace Operator

test8.m

- $T = \mathrm{tridiag}(1, -2, 1)$
- exact eigenvalues

$$-4\sin^2\left(\frac{j\pi}{2(n+1)}\right), \quad j = 1, \ldots, n$$

- computed eigenvalues

  Table shows norm of differences to exact values

n	`newtonmaehly`	`eig`
600	$7.1149e{-15}$	$1.9086e{-14}$
1000	$8.9523e{-15}$	$2.8668e{-14}$
5000	$2.0558e{-14}$	$9.3223e{-14}$
10000	$2.8705e{-14}$	$---$

Conclusions

- computing zeros of polynomials from the coefficients is ill-conditioned
- often polynomials are not given or do not need to be represented by the coefficients – computing zeros using other representations (e.g. three term recurrences) may be stable
- today, medium size nonlinear eigenvalue problems can be solved stably by brute force (determinants and algorithmic differentiation) on a laptop
- Linpack Benchmark 1000×1000 full matrix, free style programming

year	computer	cost (CHF)	time
1991	Supercomputer NEC SX-3	30'000'000	0.8 sec
2006	Laptop IBM X41 (using Matlab)	2'500	1 sec
2009	Laptop Lenovo X61 (using Matlab)	2'000	0.2 sec

References

- P. Arbenz and W. Gander, Solving nonlinear Eigenvalue Problems by Algorithmic Differentiation, Computing 36, 205-215, 1986.
- Walter Gander, Zeros of Determinants of λ-Matrices In: Matrix Methods: Theory, Algorithms and Applications, Dedicated to the Memory of Gene Golub. Vadim Olshevsky & Eugene Tyrtyshnikov eds. World Scientific Publishers, 2010
- W. Gander and D. Gruntz, The Billiard Problem, Int. J. Math. Educ. Sci. Technol., 1992, Vol. 23, No. 6, 825-830.
- G. H. Golub and J. H. Welsch, Calculation of Gauss Quadrature Rules, Math. Comp., 23 (1969), pp. 221–230.
- H. J. Maehly, Zur iterativen Auflösung algebraischer Gleichungen, ZAMP (Zeitschrift für angewandte Mathematik und Physik), (1954), pp. 260–263.
- H. Rutishauser, Description of Algol 60, Springer, 1967.
- J. Stoer and R. Bulirsch, Introduction to Numerical Analysis, Springer, 1991.
- F. Tisseur and K. Meerbergen, The Quadratic Eigenvalue Problem, SIAM. Rev., 43, pp. 234–286, 2001.