Numerical Programming 1 (CSE)

Technische Universität München, WS 2011/2012

Lectures: Caroline Lasser
Tutorials: Ilja Klebanov

(updated January 31, 2012)

Contents

1 What can't be ignored

1.1 Floating point numbers (25.10.)

We use p digits $d_1, \dots, d_p$ and an exponent e to write a floating point number with base b as

$$x = \pm b^e \left(\frac{d_1}{b} + \frac{d_2}{b^2} + \cdots + \frac{d_p}{b^p} \right).$$

For $b = 10$, this means $x = \pm 10^e \cdot 0.d_1 d_2 \dots d_p$.

Definition.

$$\mathbb{F} = \mathbb{F}(b, p, [e_{\min}, e_{\max}]) = \left\{ \pm m \cdot b^{e-p} \mid b^{p-1} \le m < b^p, e_{\min} \le e \le e_{\max} \right\}$$

```
function floating_point
b=2; p=3; emi=-1; ema=3; F = 0;
for m = b^(p-1):b^p-1, F = [F, m*b.^([emi:ema]-p)]; end
plot(F,zeros(length(F)),'r*')
```

Range. The range is $x_{\min} = b^{e_{\min}-1} \le |x| \le b^{e_{\max}}(1 - b^{-p}) = x_{\max}$.

Gaps. The spacing grows by a factor b at each power of b: Let $\varepsilon = b^{1-p}$ be the machine precision, that is, the distance from 1 to the next larger $x \in \mathbb{F}$. The distance from $x \in \mathbb{F}$ to the nearest $y \in \mathbb{F}$, $y \ne x$ is at least $b^{-1}\varepsilon|x|$ and at most $\varepsilon|x|$.

```
function IEEE_double
realmin, realmax, eps
```

Let $\mathbb{F}_\infty = \mathbb{F}(b, p,]-\infty, \infty[)$ and $\mathrm{fl} : \mathbb{R} \to \mathbb{F}_\infty$, $x \mapsto$ nearest $y \in \mathbb{F}_\infty$. In case of a tie, $\mathrm{fl}(x)$ is the y with even d_p (round to even). Overflow, if $|\mathrm{fl}(x)| > x_{\max}$, underflow, if $0 < |\mathrm{fl}(x)| < x_{\min}$.

Theorem. For all $|x| \in [x_{\min}, x_{\max}]$ there exists $|\delta| \le \frac{\varepsilon}{2}$ with $\mathrm{fl}(x) = x(1 + \delta)$.

```
function wobbling
x=linspace(1,16,1e3);
plot(x,eps(single(x))./x);
```

Wobbling. The relative distance to the next larger $x \in \mathbb{F}$ varies by b. Therefore, $b = 2$ is favoured.

Standard model. Let $\star$ be one of the arithmetic operations $+, -, \cdot, /$ and $\widetilde{\star}$ its floating point analogue. For all $x, y \in \mathbb{F}$ with $|x \star y| \in [x_{\min}, x_{\max}]$, there exists $|\delta| \le \frac{\varepsilon}{2}$ with $x \widetilde{\star} y = (x \star y)(1 + \delta)$.

Literature. [TB, Lecture 13], [H, Chapter 2], [O]

1.2 Conditioning (27.10.)

We want to evaluate a function f in x. How sensitive is $f(x)$ to slight perturbations of x?

Definition. Let $f : \mathbb{R}^n \to \mathbb{R}^m$ and $x \in \mathbb{R}^n$.

$$\kappa_f(x) = \lim_{\delta \to 0} \sup_{\|\Delta x\| < \delta} \frac{\|f(x + \Delta x) - f(x)\|}{\|f(x)\|} \cdot \frac{\|x\|}{\|\Delta x\|}.$$

For continuously differentiable f, we have $f(x+\Delta x) - f(x) = f'(x)\Delta x + o(\Delta x)$ as $\Delta x \to 0$ by Taylor's theorem. In this case, $\kappa_f(x) = \|f'(x)\| \cdot \|x\|/\|f(x)\|$.

Matrix norms. We will use three norms on $\mathbb{R}^n$:

$$\|x\|_1 = \sum_{j=1}^{n} |x_j|, \qquad \|x\|_2 = \sqrt{\sum_{j=1}^{n} |x_j|^2}, \qquad \|x\|_\infty = \max\{|x_j| : j = 1, \ldots, n\}.$$

The corresponding matrix norms $\|A\| = \max\{\|Ax\| \cdot \|x\|^{-1} : \|x\| \le 1\}$ are

$$\begin{aligned} \|A\|_1 &= \max\{\|A(:,j)\|_1 : i = 1, \ldots, n\} && \text{(maximum column sum)} \\ \|A\|_2 &= \sigma_{\max}(A) = \sqrt{\lambda_{\max}(A^*A)} && \text{(largest singular value)} \\ \|A\|_\infty &= \max\{\|A(i,:)\|_1 : i = 1, \ldots, m\} && \text{(maximum row sum)} \end{aligned}$$

Scalar multiplication. $f : \mathbb{R} \to \mathbb{R}$, $x \mapsto \frac{x}{2}$, $\kappa_f(x) = \frac{1}{2} \cdot \frac{|x|}{|x/2|} = 1$.

Square root. $f : [0, \infty) \to [0, \infty)$, $x \mapsto \sqrt{x}$, $\kappa_f(x) = \frac{|x^{-1/2}/2| \cdot |x|}{|x^{1/2}|} = \frac{1}{2}$.

Cancellation. $f : \mathbb{R}^2 \to \mathbb{R}$, $x \mapsto x_1 - x_2$, $f'(x) = (1, -1)$, $\|f'(x)\|_\infty = 2$, $\kappa_f(x) = 2\|x\|_\infty / |x_1 - x_2|$.

Literature. [TB, Lecture 3 & 12]

1.3 Stability (3.11.)

Let $f : \mathbb{R}^n \to \mathbb{R}^m$ and $\widetilde{f} : \mathbb{R}^n \to \mathbb{R}^m$ a corresponding algorithm. Analysing a computation's accuracy, we estimate

$$\begin{aligned}\frac{\|f(x) - \widetilde{f}(x+\Delta x)\|}{\|f(x)\|} &\leq \frac{\|f(x) - f(x+\Delta x)\|}{\|f(x)\|} + \frac{\|f(x+\Delta x) - \widetilde{f}(x+\Delta x)\|}{\|f(x)\|} \\ &\leq \kappa_f(x)\|\Delta x\|/\|x\| + ?\end{aligned}$$

Definition.

$$\sigma_{\widetilde{f}}(x) = \lim_{\delta \to 0} \sup_{\varepsilon < \delta} \frac{\|f(x) - \widetilde{f}(x)\|}{\|f(x)\| \cdot (\kappa_f(x) + 1) \cdot \varepsilon}$$

Discussion. With machine precision ε, the input is $x + \Delta x$ with $\|\Delta x\|/\|x\| = O(\varepsilon)$. Then,

$$\frac{\|f(x) - \widetilde{f}(x+\Delta x)\|}{\|f(x)\|} \leq \kappa_f(x) \cdot O(\varepsilon) + \sigma_{\widetilde{f}}(x) \cdot (\kappa_f(x) + 1) \cdot O(\varepsilon).$$

For well-conditioned problems forward stable algorithms achieve $O(\varepsilon)$ accuracy.

Definition.

$$\rho_{\widetilde{f}}(x) = \lim_{\delta \to 0} \sup \left\{ \frac{\|\Delta x\|}{\|x\| \cdot \varepsilon} : \widetilde{f}(x) = f(x + \Delta x),\, \varepsilon < \delta \right\}$$

Discussion. For well-conditioned problems, a forward stable algorithm computes nearly the right answer to nearly the right question, whereas a backward stable algorithm computes the exact answer to nearly the right problem. More precisely, $\sigma_{\widetilde{f}}(x) \leq \rho_{\widetilde{f}}(x)$.

Subtraction. $f : \mathbb{R}^2 \to \mathbb{R}$, $x \mapsto x_1 - x_2$, $\widetilde{f}(x) = \mathrm{fl}(x_1) \widetilde{-} \mathrm{fl}(x_2)$ backward stable.

Inner product. $f : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$, $(x, y) \mapsto x^T \cdot y$, $\widetilde{f}(x, y) = \widetilde{\sum}_j \mathrm{fl}(x_j) \widetilde{\cdot} \mathrm{fl}(y_j)$ backward stable.

Outer product. $f : \mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}^{m \times n}$, $(x, y) \mapsto (x \otimes y)_{ij} = x_i y_j$, $\widetilde{f}(x, y)_{ij} = \mathrm{fl}(x_i) \widetilde{\cdot} \mathrm{fl}(y_j)$, forward but not backward stable.

Adding to one. $f : \mathbb{R} \to \mathbb{R}$, $x \mapsto x + 1$, $\widetilde{f}(x) = \mathrm{fl}(x) \widetilde{+} 1$ is forward but not backward stable for small x.

Literature. [TB, Lecture 14 & 15]

1.4 Conditioning & Stability (8.11.)

```
function wilkinson_polynomial
a = poly(1:20); x = roots(a); plot(real(x),imag(x),'*'); hold on
for n=1:100, r = randn(size(a)); b = a.*(1 + 1e-10*r);
x = roots(b); plot(real(x),imag(x),'*'), end
```

Polynomial root finding is typically ill-conditioned.

Multiple roots. Consider $p(x) = x^2 - 2x + 1 = (x-1)^2$ and $q(x) = x^2 - 2x + 0.9999 = (x - 0.99)(x - 1.01)$. The roots of $x^2 - 2x + a_0$ are $2 \pm \sqrt{4 - 4a_0}$. The condition number of $f : [0, 1) \to \mathbb{R}$, $a_0 \mapsto \sqrt{1 - a_0}$ is

$$\kappa_f(a_0) = \frac{|f'(a_0)| \cdot |a_0|}{|f(a_0)|} = \frac{a_0}{2(1 - a_0)} \overset{a_0 \to 1}{\longrightarrow} \infty.$$

Simple roots. Let a_i be the ith coefficient of p and x_j the jth root. Since $p(x) = p'(x_j)(x - x_j) + O(|x - x_j|^2)$,

$$\kappa_{x_j}(a_i) = \lim_{\delta \to 0} \sup_{|\Delta a_i| < \delta} \frac{|x_j(a_i + \Delta a_i) - x_j(a_i)| \cdot |a_i|}{|x_j(a_i)| \cdot |\Delta a_i|} = \frac{|a_i x_j(a_i)^{i-1}|}{|p'(x_j)|}.$$

```
function wilkinson_condition
a = poly(1:20); kappa = a(6)*15^14/(factorial(14)*factorial(5))
```

Eigenvalues. Computing the eigenvalues of $A \in \mathbb{R}^{n \times n}$ by root finding for the characteristic polynomial $p(x) = \det(x\mathrm{Id} - A)$ is not forward stable.

```
function eigenvalues
b = [1,1]; x = roots([1,-b(1)-b(2),b(1)*b(2)])
b(1) = 1+1e-14; y = roots([1,-b(1)-b(2),b(1)*b(2)])
```

Product estimates. Let $f = h \circ g$, that is $f(x) = h(g(x)) = h(y)$, and $\widetilde{f} = \widetilde{h} \circ \widetilde{g}$. Then,

$$\begin{aligned} \sigma_{\widetilde{f}}(x)\kappa_f(x) &\le \kappa_h(y) \cdot \left(\sigma_{\widetilde{h}}(\widetilde{y}) + \sigma_{\widetilde{g}}(x)\kappa_g(x)\right), \\ \rho_{\widetilde{f}}(x) &\le \rho_{\widetilde{g}}(x) + \kappa_{g^{-1}}(y)\rho_{\widetilde{h}}(\widetilde{y}). \end{aligned}$$

Rules of Thumb. Algorithms built on well-conditioned problems solved by forward stable algorithms are forward stable. For backward stability, the intermediate steps should have a well-conditioned inverse mapping.

Literature. [TB, Lecture 12 & 15], [B]

2 Nonlinear equations

2.1 Newton's method (10.11.)

Let $f : \mathbb{R} \to \mathbb{R}$ be a bounded function with continuous derivatives. We compute zeros of f, that is, x_* with $f(x_*) = 0$.

Conditioning. We use the norm

$$\|\Delta f\|_\infty = \sup\{|\Delta f(x)| : x \in \mathbb{R}\}.$$

We assume that $f + \Delta f$ has a zero $\widetilde{x}_*$ with $\widetilde{x}_* \to x_*$ as $\|\Delta f\|_\infty \to 0$. Since $f(\widetilde{x}_*) = -\Delta f(\widetilde{x}_*)$ and $f(\widetilde{x}_*) = f'(x_*)(\widetilde{x}_* - x_*) + O(|\widetilde{x}_* - x_*|^2)$, we obtain

$$\kappa_{x_*}(f) = \lim_{\delta \to 0} \sup_{\|\Delta f\|_\infty < \delta} \frac{|\widetilde{x}_* - x_*| \cdot \|f\|_\infty}{\|\Delta f\|_\infty \cdot |x_*|} = \frac{\|f\|_\infty}{|f'(x_*)| \cdot |x_*|}.$$

Therefore, zeros with small first derivative are badly conditioned.

Newton's method. Starting from x_0, we construct a sequence $x_1, x_2, \dots$ with $\lim_{k\to\infty} x_k = x_*$. Newton's method defines

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}.$$

This can be motivated geometrically as well as analytically via $0 = f(x_*) = f(x_k) + f'(x_k)(x_* - x_k) + O(|x_* - x_k|^2)$.

```
function newton_reciprocal
a = 0.5; x = 0.1; x = 2*x - a*x^2; x-1/a,
a = 0.5; x = 10; x = 2*x - a*x^2; x-1/a
a = 1e-10; x = a; x = 2*x - a*x^2; x-1/a
function newton_square_root
x = 2; x = (x + 2/x)/2, x-sqrt(2)
```

Theorem. If $f'(x_*) \neq 0$ and x_0 is sufficiently close to x_*, then $\lim_{k\to\infty} x_k = x_*$ and

$$\lim_{k\to\infty} \frac{x_{k+1} - x_*}{(x_k - x_*)^2} = \frac{f''(x_*)}{2f'(x_*)}$$

Remark. The theorem says that Newton's method is quadratically convergent.

Literature. [S, Lectures 2 & 5]

2.2 Newton's method for systems (17.11.)

Fixed point methods. Let $\varphi(x) = x - f(x)/f'(x)$. Newton's method is $x_{k+1} = \varphi(x_k)$ and computes a fixed point of φ, that is, x_* with

$$\varphi(x_*) = x_*.$$

We observe that $\varphi'(x_*) = 0$, $\varphi''(x_*) = f''(x_*)/f'(x_*)$.

Theorem. Let $|\varphi'(x_*)| < 1$. If $\varphi^{(p)}(x_*) \neq 0$, $\varphi'(x_*) = \ldots = \varphi^{(p-1)}(x_*) = 0$ and x_0 is sufficiently close to a fixed point x_* of φ, then $x_{k+1} = \varphi(x_k)$ converges to x_* and

$$\lim_{k\to\infty} \frac{x_{k+1} - x_*}{(x_k - x_*)^p} = \frac{1}{p!}\varphi^{(p)}(x_*).$$

Systems. For zeros of $f : \mathbb{R}^n \to \mathbb{R}^n$, Newton's method is

$$x_{k+1} = x_k - f'(x_k)^{-1} f(x_k),$$

where $f'(x_k) \in \mathbb{R}^{n\times n}$ is the Jacobian of f at x_k. This requires two steps: solve the linear system $f'(x_k)y_k = f(x_k)$, update $x_{k+1} = x_k - y_k$.

Termination. For x_k sufficiently close to x_*,

$$\frac{1}{4\kappa} \cdot \frac{\|x_k - x_*\|}{\|x_k - x_0\|} \leq \frac{\|f(x_k)\|}{\|f(x_0)\|} \leq 4\kappa \cdot \frac{\|x_k - x_*\|}{\|x_k - x_0\|},$$

where $\kappa = \|f'(x_*)\| \cdot \|f'(x_*)^{-1}\|$ is the condition number of the Jacobian $f'(x_*)$. One might terminate, if

$$\|f(x_k)\| \leq \tau_{\text{rel}}\|f(x_0)\| + \tau_{\text{abs}}.$$

Alternatively, one might terminate, if $\|y_k\| \leq \tau_{\text{rel}}\|x_k - x_0\|$, since $\|x_k - x_*\| = \|y_k\| + O(\|x_k - x_*\|^2)$.

```
function newton_system
x = linspace(-2,2); [X1,X2] = meshgrid(x);
Y = sin(pi/2*X1) + X2.^3; contour(X1,X2,Y), colorbar
hold on, t = linspace(0,2*pi); plot(cos(t),sin(t),'-r')
x = [1;1]; plot(x(1),x(2),'*r')
x = x - [2*x(1), 2*x(2); pi/2*cos(pi/2*x(1)), 3*x(2)^2]\...
[x(1)^2 + x(2)^2-1; sin(pi/2*x(1)) + x(2)^3];
```

Literature. [S, Lecture 3], [K, Chapter 5]

3 Interpolation

3.1 Polynomial interpolation (22.11.)

Let $x_0, \ldots, x_n$ be distinct points in $[-1, 1]$ and $y_0, \ldots, y_n \in \mathbb{R}$. Polynomial interpolation provides *the* polynomial p of degree $\leq n$ such that $p(x_j) = y_j$ for all $j = 0, \ldots, n$.

Lagrange polynomials. Let $\ell_j(x)$ be the polynomial of degree $\leq n$ such that $\ell_j(x_k) = \delta_{kj}$. Then,

$$p(x) = \sum_{j=0}^{n} y_j \ell_j(x).$$

```
x = linspace(-1,1,7); y = [0 0 0 0 0 1 0]; p = polyfit(x,y,6);
xx = linspace(-1,1); yy = polyval(p,xx); plot(xx,yy,x,y,'*')
```

Conditioning. Let the nodes $x_0, \ldots, x_n$ be given and $P_n(y)$ be the interpolating polynomial for $y = (y_0, \ldots, y_n)$. The absolute condition number is

$$\kappa_P(y) = \lim_{\delta \to 0} \sup_{\|\Delta y\|_\infty < \delta} \frac{\|P_n(y + \Delta y) - P_n(y)\|_\infty}{\|\Delta y\|_\infty} = \sup_{x \in [-1,1]} \sum_{j=0}^{n} |\ell_j(x)| =: \Lambda_n,$$

the Lebesgue constant of the nodes. Always, $\Lambda_n \geq \frac{2}{\pi} \log(n+1) + 0.5$.

```
n = [10:10:1000]; c = 2/pi*log(n+1)+1; e = 2.^(n-2)./n.^2;
subplot(2,1,1), plot(n,c,'b'), subplot(2,1,2), semilogy(n,e,'r')
```

First barycentric interpolation formula. $\ell_j(x) = \frac{\prod_{k \neq j}(x - x_k)}{\prod_{k \neq j}(x_j - x_k)} = \ell(x) \frac{w_j}{x - x_j}$ with $\ell(x) = \prod_{k=0}^{n}(x - x_k)$ the node polynomial. This gives

$$p(x) = \ell(x) \sum_{j=0}^{n} \frac{y_j w_j}{x - x_j}.$$

The formula is backward stable and requires $O(n)$ flops for precomputed weights.

Interpolation error. If $y_j = f(x_j)$, then $f(x) - p(x) = \ell(x) f^{(n+1)}(\xi)/(n+1)!$ for some $\xi \in [-1, 1]$.

```
n = 8; x = cos(pi/n*[0:n]); % x = linspace(-1,1,n+1);
y = 1./(1+25*x.^2);
for j=1:n+1, w(j) = prod(1./(x(j) - x([1:j-1, j+1:end])) ); end
xx = linspace(-1,1); yy = 1./(1+25*xx.^2); l = 1; s = 0;
for j=1:n+1, l = l.*(xx-x(j)); s = s + y(j)*w(j)./(xx-x(j)); end
p = l.*s; plot(xx,yy,xx,p,x,y,'*')
```

Literature. [T, Chapter 5 & 15]

3.2 Trigonometric interpolation (24.11)

Let $x_0, \ldots, x_{n-1}$ be distinct points in $[0, 2\pi]$ and $y_0, \ldots, y_{n-1} \in \mathbb{C}$. Trigonometric interpolation provides *the* trigonometric polynomial

$$p(x) = \sum_{j=0}^{n-1} c_j e^{ijx}$$

such that $p(x_j) = y_j$ for all $j = 0, \ldots, n-1$. From now on, $x_j = 2\pi j/n$.

Conditioning. As for algebraic interpolation, the absolute condition number is the Lebesgue constant Λ_n, and one can prove $\Lambda_n \leq \frac{2}{\pi}\log(n) + \frac{5}{3}$.

Fast Fourier Transform. We observe $e^{ijx_k} = e^{2\pi ijk/n} = \omega_n^{jk}$, write $y_k = \sum_{j=0}^{n-1} c_j \omega_n^{jk}$ and compute $c_l = \frac{1}{n}\sum_{k=0}^{n-1} \omega_n^{-lk} y_k$. Let $n = 2^m$. Since $\omega_{2n}^2 = \omega_n$,

$$c_l = \frac{1}{n}\sum_{k=0}^{n/2-1} \omega_{n/2}^{-lk}\, y_{2k} + \frac{1}{n}\omega_n^{-l}\sum_{k=0}^{n/2-1} \omega_{n/2}^{-lk}\, y_{2k+1} = \ldots$$

This is the basic observation for the FFT, the computation of $c = (c_0, \ldots, c_{n-1})$ in $O(n\log(n))$ flops, which can be done with a backward stable algorithm.

```
n = 32; y = zeros(1,n); y(1) = 1; p = fft(y);
subplot(1,2,1), plot([0:n-1],real(p),'b-*'),
subplot(1,2,2), plot([0:n-1],imag(p),'r-*')
```

Fourier Series. Let $f : [-\pi, \pi] \to \mathbb{C}$ be 2π-periodic, continuously differentiable. Then, $f(x) = \sum_{j=-\infty}^{\infty} \widehat{f}_j e^{ijx}$, where

$$\widehat{f}_j = \frac{1}{2\pi}\int_{-\pi}^{\pi} f(x)e^{-ijx}dx$$

is the jth Fourier coefficient. We write the trigonometric interpolant for $x_j = 2\pi j/n$, $y_j = f(x_j)$ as $p(x) = \sum_{j=-n/2}^{n/2-1} c_j e^{ijx}$ and observe

$$c_j = \frac{1}{n}\sum_{k=-n/2}^{n/2-1} y_k \omega_n^{-jk} = \frac{1}{2\pi}\cdot\frac{2\pi}{n}\sum_{k=-n/2}^{n/2-1} f(x_k)e^{-ijx_k} \approx \widehat{f}_j.$$

Aliasing. The interpolation property yields $c_j = \sum_{k=-\infty}^{\infty} \widehat{f}_{j+kn}$.

```
n = 8; x = 2*pi/n*[0:n-1]; y = sin(x) + 3*sin(5*x);
xx = linspace(0,2*pi,100); yy = sin(xx) + 3*sin(5*xx);
z = interpft(y,100); plot(xx,yy,'b-',xx,z,'r-',x,y,'ro')
n = 16; x = 2*pi/n*[0:n-1]; y = sin(x) + 3*sin(5*x);
z = interpft(y,100); hold on, plot(xx,z,'g-')
```

Literature. [M, Chapter 8]

3.3 Spline interpolation (29.11.)

Linear splines. Let $x_0, \dots, x_n \in [a,b]$ with $a = x_0 < \dots < x_n = b$ and $y_0, \dots, y_n \in \mathbb{R}$. The continous function $s : [a,b] \to \mathbb{R}$, which is linear on each interval $[x_j, x_{j+1}]$ and satisfies $s(x_j) = y_j$ for all $j = 0, \dots, n$, is *the* interpolating linear spline:

$$s(x) = y_j + \frac{y_{j+1} - y_j}{x_{j+1} - x_j}(x - x_j), \qquad x \in [x_j, x_{j+1}].$$

Approximation error. Let $f : [a,b] \to \mathbb{R}$ be twice continuously differentiable and $y_j = f(x_j)$ for all j. Then, $f(x) - s(x) = \frac{1}{2} f''(\xi)(x - x_j)(x - x_{j+1})$ for $x \in [x_j, x_{j+1}]$. If $h_j = x_{j+1} - x_j$ and $h = \max\{h_0, \dots, h_{n-1}\}$, then

$$\|f - s\|_\infty \le \frac{h^2}{8} \|f''\|_\infty.$$

```
x = linspace(-2,8,6); y = x.^2 + 10./(sin(x)+1.2);
xx = linspace(-2,8); yy = xx.^2 + 10./(sin(xx)+1.2);
z = interp1(x,y,xx); plot(xx,yy,'b-',xx,z,'r-',x,y,'ro')
```

Cubic splines. Twice continuously differentiable functions $s : [a,b] \to \mathbb{R}$, which are cubic polynomials on each interval $[x_j, x_{j+1}]$ and satisfy $s(x_j) = y_j$ for all $j = 0, \dots, n$, are interpolating cubic splines:

$$s(x) = a_j + b_j(x - x_j) + \frac{s_j}{6h_j}(x_{j+1} - x)^3 + \frac{s_{j+1}}{6h_j}(x - x_j)^3$$

with $a_j = y_j - \frac{1}{6} s_j h_j^2$, $b_j = d_j - \frac{1}{6}(s_{j+1} - s_j)h_j$, $d_j = (y_{j+1} - y_j)/h_j$. The second derivatives $s_j = s''(x_j)$ satisfy

$$\frac{h_j}{6} s_j + \frac{h_j + h_{j+1}}{3} s_{j+1} + \frac{h_{j+1}}{6} s_{j+2} = d_{j+1} - d_j, \qquad j = 1, \dots, n-2.$$

End conditions. $s'(a) = f'(a)$, $s'(b) = f'(b)$ yields *the* complete spline. The not-a-knot condition is

$$s|_{[x_0,x_1]} = s|_{[x_1,x_2]}, \quad s|_{[x_{n-2},x_{n-1}]} = s|_{[x_{n-1},x_n]}.$$

The resulting tridiagonal linear systems are well-conditioned and can always be solved in a backward stable way in $O(n)$ flops.

Approximation error. If f is four times continuosly differentiable, then both cubic splines satisfy

$$\|f^{(r)} - s^{(r)}\|_\infty \le C_r h^{4-r} \|f^{(4)}\|_\infty, \qquad r = 0, 1, 2.$$

```
z = interp1(x,y,xx,'spline'); hold on, plot(xx,z,'g-')
```

Literature. [S2, Chapter 10 & 11]

4 Quadrature

4.1 Basics (1.12.)

Let $f : [a, b] \to \mathbb{R}$ be continuous. Let $x_0, \ldots, x_n \in [a, b]$ and $y_0, \ldots, w_n \in \mathbb{R}$. We approximate

$$I(f) = \int_a^b f(x)dx \approx \sum_{j=0}^{n} w_j f(x_j) = Q(f).$$

Conditioning. If $\|w\|_1 = |w_0| + \ldots + |w_n|$, then

$$\kappa_I(f) = \frac{(b-a)\|f\|_\infty}{|I(f)|}, \qquad \kappa_Q(f) = \frac{\|w\|_1 \|f\|_\infty}{|Q(f)|}.$$

Hence, integration and quadrature of oscillatory functions are ill-conditioned.

Stability. If $Q(1) = I(1)$, then

$$\kappa_Q(f) \geq \kappa_I(f) + (b-a)\|f\|_\infty \cdot r(I(f), Q(f)) \qquad (*)$$

with $|r(I(f), Q(f))| \leq |I(f) - Q(f)|$. $(*)$ is an equality iff $w_0, \ldots, w_n \geq 0$. Therefore, positive weights are preferred.

Simple interpolatory rules. Let p_1, p_2, p_3 be the interpolating polynomials for the nodes $\{\frac{a+b}{2}\}$, $\{a, b\}$, and $\{a, \frac{a+b}{2}, b\}$, respectively. Then,

$$\begin{aligned} M_{[a,b]}(f) &= I(p_1) = (b-a)f(\tfrac{a+b}{2}), \\ T_{[a,b]}(f) &= I(p_2) = \tfrac{b-a}{2}(f(a) + f(b)), \\ S_{[a,b]}(f) &= I(p_3) = \tfrac{b-a}{6}(f(a) + 4f(\tfrac{a+b}{2}) + f(b)). \end{aligned}$$

M and T are exact for linear integrands, S for cubic ones.

Composite rules. Let n be even and $x_j = a + jh$, $j = 0, \ldots, n$ with $h = (b-a)/n$ and $f_j = f(x_j)$. The composite rules are

$$\begin{aligned} M_n(f) &= \sum_{j=0}^{n/2-1} M_{[x_{2j}, x_{2(j+1)}]}(f) = \sum_{j=0}^{n/2-1} 2hf_{2j+1}, \\ T_n(f) &= \sum_{j=1}^{n} T_{[x_{j-1}, x_j]}(f) = \tfrac{h}{2}(f_0 + f_n) + \sum_{j=1}^{n-1} hf_j, \\ S_n(f) &= \sum_{j=0}^{n/2-1} S_{[x_{2j}, x_{2(j+1)}]}(f) = \tfrac{h}{3}(f_0 + \sum_{j=0}^{n/2-1} 4f_{2j+1} + \sum_{j=1}^{n/2-1} 2f_{2j} + f_n) \end{aligned}$$

M_n and T_n are both $O(h^2)$, S_n is $O(h^4)$ accurate.

```
n = 10; x = linspace(-1,1,n); y = exp(-x.^2/2)/sqrt(2*pi);
T = trapz(x,y), I = quad('exp(-x.^2/2)/sqrt(2*pi)',-1,1)
```

Literature. [S, Chapter 21]

4.2 Gauß quadrature (6.12.)

Let $w : [a, b] \to [0, \infty[$. We compute

$$I_w(f) = \int_a^b f(x)w(x)dx \approx Q(f) = \sum_{j=0}^{n} w_j f(x_j)$$

for $f : [a, b] \to \mathbb{R}$ continuous. Let $\mathbb{P}_n$ the space of polynomials of degree $\leq n$.

Weights. $\forall p \in \mathbb{P}_n : Q(p) = I_w(p)$ if and only if $\forall j = 0, \ldots, n : w_j = I_w(\ell_j)$. Moreover, $\forall p \in \mathbb{P}_{2n} : \ Q(p) = I_w(p)$ implies $\forall j = 0, \ldots, n : \ w_j \geq 0$.

Idea of Gauß quadrature. Let Q be exact on $\mathbb{P}_n$. We determine nodes such that Q is exact on $\mathbb{P}_{2n+1}$. Let $p \in \mathbb{P}_{2n+1}$ and $p_{n+1} \in \mathbb{P}_{n+1}$. We write $p = qp_{n+1} + r$ with $q, p \in \mathbb{P}_n$. Then, $I_w(p) = I_w(qp_{n+1}) + Q(r)$ and we obtain $I_w(p) = Q(p)$, if

$$\forall q \in \mathbb{P}_n : I_w(qp_{n+1}) = 0 = Q(qp_{n+1}).$$

This works, if $x_0, \ldots, x_n$ are the roots of p_{n+1} and p_{n+1} is orthogonal on $\mathbb{P}_n$.

Orthogonal polynomials. $(p_n)_{n\geq 0}$ is a familiy of orthogonal polynomials, if $p_n \in \mathbb{P}_n$, $p_n \neq 0$ for all n and

$$\forall n \neq m : I_w(p_n p_m) = 0.$$

Each family satisfies a 3-term recurrence

$$a_n p_{n+1}(x) = (b_n + c_n x)p_n(x) - d_n p_{n-1}(x)$$

with $(b_n)_{n\geq 0}$, $(c_n)_{n\geq 0}$, $(d_n)_{n\geq 0}$ sequences of real numbers.

	$[a, b]$	w	a_n	b_n	c_n	d_n
Legendre P_n	$[-1, 1]$	1	$n + 1$	0	$2n + 1$	n
Hermite H_n	$]-\infty, +\infty[$	e^{-x^2}	1	0	2	$2n$
Laguerre L_n	$[0, \infty[$	e^{-x}	$n + 1$	$2n + 1$	-1	n

Eigenvalue problems. The 3-tem recurrence allows to characterize the nodes as the eigenvalues of a symmetric tridiagonal eigenvalue problem, while the weights can be obtained from the eigenvectors.

```
function gauss_legendre
n = 6; beta = .5./sqrt(1-(2*(1:n)).^(-2));
T = diag(beta,1) + diag(beta,-1); [V,D] = eig(T);
x = diag(D); w = 2*V(1,:).^2; I = w*cos(x);
error = I - 2*sin(1)
```

Literature. [S, Chapter 23]

4.3 Monte Carlo quadrature (13.12.)

Let $\Omega \subset \mathbb{R}^d$ and $w : \Omega \to [0, \infty[$ such that $\int_\Omega w(x)dx = 1$. Then,

$$I_w(f) = \int_\Omega f(x)w(x)dx = \mathbb{E}_w(f).$$

Here, $I_w(f)$, $I_w(f^2)$ exist and are finite.

Strong law of large numbers. Let $(X_n)_{n>0}$ be independent identically distributed random variables distributed according to w. Then,

$$\mathbb{P}\Big(\lim_{n\to\infty} \frac{1}{n}\sum_{j=1}^{n} f(X_j) - \mathbb{E}_w(f)\Big) = 1.$$

```
function plain_mc
n = 10; x = rand(n,1); plot(x,zeros(n,1),'ro'), I = sum(x)/n
```

Expectation & variance. Let $Q_n(f) = \frac{1}{n}\sum_{j=1}^{n} f(X_j)$. Then, $\mathbb{E}(Q_n(f)) = \mathbb{E}_w(f)$ and $\mathbb{V}(Q_n(f)) = \mathbb{V}_w(f)/n$. Therefore,

$$\sqrt{\mathbb{E}([Q_n(f) - \mathbb{E}_w(f)]^2)} = \frac{\sigma_w(f)}{\sqrt{n}}.$$

Statistics. Let $q_1, \ldots, q_m$ be independent runs of $Q_n(f)$. Then,

$$\overline{q} = \frac{1}{m}\sum_{j=1}^{m} q_j, \qquad \overline{v} = \frac{1}{m-1}\sum_{j=1}^{m}(q_j - \overline{q})^2$$

are unbiased estimators of $\mathbb{E}_w(f)$ and $\mathbb{V}_w(f)$, respectively. That is, $\mathbb{E}(\overline{q}) = \mathbb{E}(Q_n(f))$ and $\mathbb{E}(\overline{v}) = \mathbb{V}(Q_n(f))$.

```
function integrate_mc
n = 1e+4; for j=1:10, q(j) = sum(rand(n,1))/n; end,
error = abs(mean(q)-0.5), error_est = sqrt(var(q)),
error_L2 = 1/2/sqrt(n)
```

```
function normal_distr
n = 1e+4; for j=1:10, x = 1+randn(n,1)*2; q(j) = sum(x)/n; end,
error = abs(mean(q)-1),
n = 100; m = 2; q = zeros(2,m);
for j=1:m, x = repmat([1,2],n,1)+randn(n,2)*chol([1,0.5;0.5,2]);
plot(x(:,1),x(:,2),'ro'); q(:,j) = sum(x,1)/n; end,
error = norm(mean(q,2)'-[1,2])
```

Literature. [DR, Chapter 5.9]

5 Linear systems

5.1 Basics (15.12.)

Let $A \in \mathbb{R}^{n\times n}$ be invertible, $b \in \mathbb{R}^n$. We compute $x \in \mathbb{R}^n$ with $Ax = b$.

Conditioning. Let $f : \mathbb{R}^n \to \mathbb{R}^n$, $b \mapsto A^{-1}b$. Then,

$$\kappa_f(b) = \lim_{\delta\to 0} \sup_{\|\Delta b\|<\delta} \frac{\|A^{-1}(b+\Delta b) - A^{-1}b\| \cdot \|b\|}{\|A^{-1}b\| \cdot \|\Delta b\|} = \frac{\|A^{-1}\| \cdot \|b\|}{\|A^{-1}b\|}$$

and $\kappa_f(Ab) = \|A^{-1}\| \cdot \|Ab\|/\|b\| \leq \|A^{-1}\| \cdot \|A\| =: \kappa(A)$ for all $b \in \mathbb{R}^n$.

```
A = [0.2161, 0.1441; 1.2969 0.8648]; b = [0.144; 0.8642];
x = A\b; b = b + 1e-8*rand(2,1); dev = A\b - x, kappa = cond(A)
```

Preconditioning. An invertible $B \in \mathbb{R}^{n\times n}$ with $\kappa(BA) \ll \kappa(A)$ is called a preconditioner for A.

Literature. [S, Lecture 16]

5.2 Gaussian elimination (20.12.)

Triangular systems. Let $L, U \in \mathbb{R}^{n\times n}$ be a lower and upper triangular matrix, $l_{ij} = 0$ for $i < j$ and $u_{ij} = 0$ for $i > j$, respectively. Let L, U be invertible, that is $l_{ii} \neq 0$ and $u_{ii} \neq 0$ for all i. We write $Lx = b$ as

$$\begin{pmatrix} \lambda & 0 \\ l & L_* \end{pmatrix} \begin{pmatrix} x_1 \\ x_* \end{pmatrix} = \begin{pmatrix} \lambda y \\ x_1 l + L_* x_* \end{pmatrix} = \begin{pmatrix} \beta \\ b_* \end{pmatrix}$$

with $L_* \in \mathbb{R}^{(n-1)\times(n-1)}$ upper triangular, $l, x_*, b_* \in \mathbb{R}^{n-1}$, $\lambda \neq 0$, $x_1, \beta \in \mathbb{R}$.

```
function forward_sub
n = 100; L = tril(rand(n))+eye(n); spy(L);
b = rand(n,1); xx = L\b;
for i=1:n, x(i)=b(i)/L(i,i); b(i+1:n)=b(i+1:n)-x(i)*L(i+1:n,i);
end, error = norm(x-xx)
```

Substitution. Forward (back) substitution is a backward stable algorithm for solving lower (upper) triangular systems using $\sum_{i=1}^{n}(2(n-i)+1) = n^2$ flops.

LU decomposition. Consider an invertible $A \in \mathbb{R}^{n\times n}$ admitting a decomposition $A = LU$ with L lower unit triangular and U upper triangular such that

$$\begin{pmatrix} \alpha & a^T \\ b & A_* \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ l & L_* \end{pmatrix} \begin{pmatrix} \mu & u^T \\ 0 & U_* \end{pmatrix} = \begin{pmatrix} \mu & u^T \\ \mu l & l u^T + L_* U_* \end{pmatrix}$$

with $A_*, L_*, U_* \in \mathbb{R}^{(n-1)\times(n-1)}$, $a, b, l, u \in \mathbb{R}^{n-1}$, $\alpha, \mu \neq 0$.

```
function LU_fac
n=100; A=rand(n); LL=tril(A,-1)+eye(n); UU=triu(A)+eye(n);
A=LL*UU; cond(A), L=eye(n); U=zeros(n,n);
for i=1:n, U(i,i:n)=A(i,i:n); L(i+1:n,i)=A(i+1:n,i)/A(i,i);
A(i+1:n,i+1:n)=A(i+1:n,i+1:n)-L(i+1:n,i)*U(i,i+1:n); end,
norm(L*U-LL*UU)
```

Operation count. $\sum_{i=1}^{n}(2(n-i)^2 + (n-i)) \approx \int_0^n 2x^2 dx = \frac{2}{3}n^3$ flops. Solving $Ax = b$ via $Ly = b$, $Ux = y$ is dominated by the cost for the LU decomposition.

Pivoting. The maximal entry in magnitude of each column vector $a_{i:n,i}$ does not vanish. We permute rows to divide by this maximal entry. This also yields $|l_{i,j}| \leq 1$ for all i, j, which is important for stability.

```
function pivoting
[y,ii]=max(abs(A(i:n,i))); ii=ii+i-1;
A([ii,i],i:n)=A([i,ii],i:n); L([ii,i],1:i-1)=L([i,ii],1:i-1);
```

Literature. [TB, Lecture 20 & 21]

5.3 Iterative methods (22.12.)

Jacobi & Gauß-Seidel iteration. We write the ith row of the linear system $Ax = b$ as $x_i = (b_i - \sum_{j \neq i} a_{ij} x_j)/a_{ii}$. We construct a sequence $(x^k)_{k>0}$ in $\mathbb{R}^n$ by the Jacobi iteration

$$x_i^{k+1} = (b_i - \sum_{j \neq i} a_{ij} x_j^k)/a_{ii}$$

and hope for $\lim_{k \to \infty} x^k = A^{-1}b$. The Gauß-Seidel iteration is defined by

$$x_i^{k+1} = (b_i - \sum_{j<i} a_{ij} x_j^{k+1} - \sum_{j>i} a_{ij} x_j^k)/a_{ii}.$$

Matrix form. Let $A = D - L - U = \operatorname{diag}(A) + \operatorname{tril}(A, -1) + \operatorname{triu}(A, 1)$. We write the Jacobi iteration as

$$x^{k+1} = D^{-1}b + D^{-1}(L + U)x^k.$$

The Gauß-Seidel iteration takes the form

$$x^{k+1} = D^{-1}(b + Lx^{k+1} + Ux^k) = (D - L)^{-1}b + (D - L)^{-1}Ux^k.$$

```
n = 100; A = rand(n); for i=1:n, A(i,i)= sum(abs(A(i,:))); end,
b = rand(n,1); D = diag(diag(A)); L = -tril(A,-1); U =-triu(A,1);
m = 10; x = rand(n,1); y = x;
for k=1:m, k; x=D\b + D\(L+U)*x; y=(D-L)\b + (D-L)\U*y;
norm(x-A\b), norm(y-A\b), pause, end
```

Splitting. Let $A = M - N$ with M invertible. We write $Ax = (M - N)x = b$ as $x = M^{-1}b + M^{-1}Nx$ and derive a stationary iteration

$$x^{k+1} = M^{-1}b + M^{-1}Nx^k.$$

The Jacobi and Gauß-Seidel iteration are stationary with $A = D - (L + U)$ and $A = (D - L) - U$, respectively.

Convergence. We write $x^{k+1} - x = M^{-1}N(x^k - x) = (M^{-1}N)^{k+1}(x^0 - x)$. Sufficient conditions for convergence are

$$\lim_{k \to \infty} (M^{-1}N)^k = 0, \qquad \|M^{-1}N\| < 1.$$

```
norm(D\(L+U)), norm((D-L)\U)
```

Literature. [S2, Lecture 25]

6 Least-squares-problems

6.1 Normal equations (10.1.)

Let $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$ with $m \geq n$. We seek a solution to the overdetermined linear system $Ax = b$ in the following sense: $x \in \mathbb{R}^n$ shall minimize

$$\|Ax - b\|_2 = \sqrt{\sum_{j=1}^{m} ((Ax)_j - b_j)^2}.$$

We solve a least squares problem.

Polynomial data fit. Fitting a polynomial of degree $n-1$ to m data points $(x_1, y_1), \ldots, (x_m, y_m)$ with $m \geq n$ can be formulated as a linear system $Ac = y$ with $A \in \mathbb{R}^{m\times n}$ a Vandermonde matrix,

$$A = \begin{pmatrix} x_1^{n-1} & x_1^{n-2} & \ldots & 1 \\ x_2^{n-1} & x_2^{n-2} & \ldots & 1 \\ \vdots & & & \vdots \\ x_m^{n-1} & x_m^{n-2} & \ldots & 1 \end{pmatrix}.$$

```
function polyfit
n = 11; x = linspace(-1,1,n); y = rand(n,1);
A = vander(x); cond(A), c = A\y;
xx = linspace(-1,1); plot(xx,polyval(c,xx),x,y,'o')
A = A(:,3:n); d = A\y; hold on, plot(xx,polyval(d,xx),'g')
```

Normal equations. $x \in \mathbb{R}^n$ minimizes the residual norm $\|r\|_2 = \|b - Ax\|_2$ if and only if $r \perp \text{range}(A)$ if and only if $A^T r = 0$ if and only if x satisfies the normal equations

$$A^T A x = A^T b.$$

The solution x is unique if and only if $\text{rank}(A) = n$.

```
function laeuchli
d = 1e-7; A = [1,1; d,0; 0,d]; cond(A), b = [2;d;d]; xx = [1;1];
x = A'*A\(A'*b); y = A\b; norm(x-xx), norm(y-xx), cond(A'*A)
```

Cholesky decomposition. $A^T A \in \mathbb{R}^{n\times n}$ is symmetric. If $\text{rank}(A) = n$, then $A^T A$ is positive definite, that is, $x^T(A^T A)x > 0$ for all $x \neq 0$. Therefore,

$$A^T A = LL^T$$

for a uniquely determined lower triangular matrix L with positive diagonal entries. L can be computed backward stably in approximately $\frac{1}{3}n^3$ flops.

Literature. [TB, Lecture 11]

6.2 QR decomposition (12.1.)

Orthogonal matrices. A matrix $Q \in \mathbb{R}^{n\times n}$ is orthogonal, if

$$Q^T = Q^{-1}.$$

From $Q^TQ = \mathrm{Id}$ we deduce $q_i^Tq_j = \delta_{ij}$ for the column vectors $q_1, \ldots, q_n$. Moreover, $(Qx)^TQy = x^Ty$ and $\|Qx\|_2 = \|x\|_2$ for all $x, y \in \mathbb{R}^n$. Consequently $\|Q\|_2 = 1$ and $\kappa(Q) = 1$.

QR decomposition. Any $A \in \mathbb{R}^{m\times n}$ with $m \geq n$ admits a QR decomposition

$$A = QR$$

with $Q \in \mathbb{R}^{m\times m}$ an orthogonal matrix and $R \in \mathbb{R}^{m\times n}$ upper triangular. If $\mathrm{rank}(A) = n$, then R can be chosen with positive diagonal entries.

Least squares problem. We write

$$\|b - Ax\|_2^2 = \|(Q^Tb)(1:n) - R(1:n,:)x\|_2^2 + \|(Q^Tb)(n+1:m)\|_2^2$$

and obtain the least squares solution by solving the upper triangular system $R(1:n,:)x = (Q^Tb)(1:n)$.

```
function least_squares
d = 1e-7; A = [1,1; d,0; 0,d]; b = [2;d;d]; xx = [1;1];
[Q,R] = qr(A); b = (Q'*b); x = R(1:2,:)\b(1:2); norm(x-xx)
```

Gram-Schmidt orthogonalization. Let $\mathrm{rank}(A) = n$ and set $a_j = A(:,j)$. We seek orthonormal vectors $q_1, \ldots, q_n \in \mathbb{R}^m$ such that

$$\forall j = 1, \ldots, n : \ \mathrm{span}\{a_1, \ldots, a_j\} = \mathrm{span}\{q_1, \ldots, q_j\}.$$

This means $a_1 = r_{11}q_1$, $a_2 = r_{12}q_1 + r_{22}q_2$, $\ldots$, $a_n = r_{1n}q_1 + \ldots + r_{nn}q_n$ or

$$A = \hat{Q}\hat{R}$$

with $\hat{Q} \in \mathbb{R}^{m\times n}$ a matrix with orthonormal columns and $\hat{R} \in \mathbb{R}^{n\times n}$ upper triangular. This is a reduced QR decomposition.

Existence. From $q_2 = (a_2 - r_{12}q_2)/r_{22}$ we deduce $r_{12} = q_1^Ta_2$, and from $q_3 = (a_2 - r_{13}q_1 - r_{23}q_2)/r_{33}$ we obtain $r_{13} = q_1^Ta_3$, $r_{23} = q_2^Ta_3$. Hence,

$$\forall i < j : r_{ij} = q_i^Ta_j,$$

and the diagonal elements of R are used for normalization.

Literature. [TB, Lecture 2 & 7]

6.3 Householder triangularization (17.1.)

```
function gram_schmidt
m=25; n=15; p=linspace(0,1,m); A=vander(p); A=A(:,11:end);
% classical
for j=1:n, v=A(:,j);
for i=1:(j-1), R(i,j)=Q(:,i)'*A(:,j); v=v-R(i,j)*Q(:,i); end
R(j,j)=norm(v); Q(:,j)=v/R(j,j); end,
norm(Q*R-A)/norm(A), norm(Q'*Q-eye(n)), cond(A)
% modified
V = A; for i=1:n, R(i,i)=norm(V(:,i)); Q(:,i)=V(:,i)/R(i,i);
for j=i+1:n, R(i,j)=Q(:,i)'*V(:,j); V(:,j)=V(:,j)-R(i,j)*Q(:,i);
end, end, norm(Q*R-A)/norm(A), norm(Q'*Q-eye(n))
```

Gram-Schmidt orthogonalization (classical). The classical method orthogonalizes q_j with respect to $a_1, \ldots, a_{j-1}$. It might produce a not-orthogonal Q-factor. Operation count: approximately $\sum_{i=1}^{n} \sum_{j=i+1}^{n} 4m \approx 2mn^2$ flops.

Gram-Schmidt orthogonalization (modified). The modified method orthogonalizes $q_{j+1}, \ldots, q_n$ with respect to q_j. It is backward stable with the same operation count as the classical method.

Householder reflectors. Instead of orthogonalizing one triangularizes by orthogonal elimination. One uses n orthogonal matrices of the form

$$Q_k = \begin{pmatrix} \mathrm{Id}_{k-1} & 0 \\ 0 & F \end{pmatrix} \in \mathbb{R}^{m \times m} \qquad (k = 1, \ldots, n)$$

with F an orthogonal matrix called Householder reflector: For any vector $v \in \mathbb{R}^{m-(k-1)}$, the matrix

$$F = \mathrm{Id} - 2vv^T / \|v\|^2$$

satisfies $F = F^T$ and $FF^T = \mathrm{Id}$.

Literature. [TB, Lecture 10 & 16]

7 Eigenvalue problems

7.1 Basics (19.1.)

Householde reflectors, continued. Let x be the last $m-(k-1)$ rows of the kth column of the matrix $Q_{k-1}Q_2Q_1A$ and $v = \mathrm{sign}(x_1)\|x\|e_1 + x$. Then, $2v^Tx = \|v\|^2$ and the Householder reflector $F = \mathrm{Id} - 2vv^T/\|v\|^2$ satisfies $Fx = -\mathrm{sign}(x_1)\|x\|e_1$, as desired. F reflects across the hyperplane orthogonal to v. Altogether,

$$Q_nQ_{n-1}\dots Q_1A = Q^TA = R.$$

Householder triangularization is backward stable and requires approximately $2n^2(m-n/3)$ flops.

```
R = triu(randn(50)); [Q,X] = qr(randn(50)); A = Q*R;
[QQ,RR] = qr(A); norm(Q-QQ), norm(R-RR)/norm(R),
norm(A-QQ*RR)/norm(A), norm(QQ'*QQ-eye(50))
```

Eigenvalues. Let $A \in \mathbb{C}^{n\times n}$. $\lambda \in \mathbb{C}$ is called an eigenvalue of A, if there exists a vector $x \in \mathbb{C}^n$, $x \neq 0$ such that $Ax = \lambda x$.

Characteristic polynomial. λ is an eigenvalue of A if and only if it is a zero of the characteristic polynomial

$$p(\lambda) = \det(\lambda\mathrm{Id} - A) = \lambda^n + \ldots + (-1)^n\det(A).$$

There are n not necessarily distinct zeros $\lambda_1,\ldots,\lambda_n \in \mathbb{C}$ such that $p(\lambda) = (\lambda-\lambda_1)\cdot\ldots\cdot(\lambda-\lambda_n)$. The number of times λ_j appears in the factorization of p is the algebraic multiplicity of λ_j.

The adjoint matrix. The adjoint matrix $A^* \in \mathbb{C}^{n\times n}$ is defined as

$$(A^*)_{i,j} = \overline{A_{j,i}}.$$

λ is an eigenvalue of A if and only if $\overline{\lambda}$ is an eigenvalue of A^*. Let $y \in \mathbb{C}^n$, $y \neq 0$ satisfy $A^*y = \overline{\lambda}y$. Then, $y^*A = \lambda y^*$, and y^* is called a left eigenvector of A corresponding to λ.

Double eigenvalues. The matrix

$$A_\varepsilon = \begin{pmatrix} 1 & 1 \\ \varepsilon & 1 \end{pmatrix}$$

has the characteristic polynomial $p(\lambda) = (1-\lambda)^2 - \varepsilon$ and the eigenvalues $1 \pm \sqrt{\varepsilon}$. Hence, an ε-perturbation of A_0 perturbs the eigenvalue by $\sqrt{\varepsilon}$.

Literature. [S2, Lecture 12]

7.2 Power iterations (24.1.)

Dominant eigenvector. Let $A \in \mathbb{C}^{n\times n}$ have a basis of normalized eigenvectors, $Ax_j = \lambda_j x_j$, $j = 1, \ldots, n$ with $|\lambda_1| > |\lambda_2| \geq \ldots \geq |\lambda_n|$. Let $v = \alpha_1 x_1 + \ldots + \alpha_n x_n$ with $\alpha_1 \neq 0$. Then,

$$A^k v = \alpha_1 \lambda_1^k x_1 + |\lambda_1|^k \left(\alpha_2 \frac{\lambda_2^k}{|\lambda_1|^k} + \ldots + \alpha_n \frac{\lambda_n^k}{|\lambda_1|^k} x_n \right),$$

and the distance of $A^k v/\|A^k v\|$ to $\operatorname{span}(x_1)$ is $O(|\lambda_2/\lambda_1|^k)$.

```
A=-[1 2;3 4]; [V,D]=eig(A);
plot([0,V(1,2),-V(1,2)],[0,V(2,2),-V(2,2)],'r-'),
axis([-1 1 -1 1]), hold on, v=[1;0]; plot(v(1),v(2),'o'),
for k=1:3, v=A*v/norm(A*v); plot(v(1),v(2),'o'), pause, end
```

Rayleigh quotient. Let $A \in \mathbb{C}^{n\times n}$. For $x \in \mathbb{C}^n$ we define

$$r(x) = \frac{x^* A x}{x^* x}.$$

If (x, λ) is an eigenpair of A, then $r(x) = \lambda$. Let $y \in \mathbb{C}^n$ with $y^* y = 1$ and $y^* x = 0$. Then, for all $h > 0$

$$r(x + hy) = \frac{\lambda + h x^* A y + h^2}{1 + h^2} = \lambda + O(h).$$

If $A = A^*$, then $r(x + hy) = \lambda + O(h^2)$.

```
A=[1 2;3 4]; ll=eig(A), v=[1;0];
for k=1:10, v=A*v/norm(A*v); l(k)=v'*A*v; end,
A=[1 2;2 3]; lls=eig(A), v=[1;0];
for k=1:10, v=A*v/norm(A*v); ls(k)=v'*A*v; end,
semilogy([1:10],abs(l-ll(2)),[1:10],abs(ls-lls(2)))
```

Inverse iteration. Let $A \in \mathbb{C}^{n\times n}$ be invertible. (x, λ) is an eigenpair of A if and only if $(x, 1/\lambda)$ is an eigenpair of A^{-1}. The inverse iteration approximates (x_n, λ_n) with convergence rate $O(|\lambda_n/\lambda_{n-1}|^k)$.

```
A=[1 2;3 4]; ll=eig(A); v=[1;0];
for k=1:10, v=A\v; v=v/norm(v); l(k)=v'*A*v; end
hold on, semilogy([1:10],abs(l-ll(1)),'r')
```

Literature. [TB, Lecture 27]

7.3 QR algorithm (26.1)

Shifted iterations. Let $A \in \mathbb{C}^{n\times n}$ be invertible and $\mu \in \mathbb{C}$. (x, λ) is an eigenpair of A if and only if $(x, \lambda-\mu)$ is an eigenpair of $A-\mu\mathrm{Id}$. The inverse iteration with shift μ approximates the eigenvalue λ_j closest to μ with convergence rate

$$O(|(\mu - \lambda_j)/(\mu - \lambda_{j'}|^k),$$

where $\lambda_{j'}$ is the second closest eigenvalue to μ.

```
A=ones(3,3)+diag([1,2,3]); ll=eig(A); mu=2.5; v=[1;0;0];
for k=1:10, v=(A-mu*eye(3))\v; v=v/norm(v); l(k)=v'*A*v; end
semilogy([1:10],abs(l-ll(2)))
```

Similarity transformations. Let $X \in \mathbb{C}^{n\times n}$ be invertible. The matrix $X^{-1}AX$ has the same eigenvalues as A. Let $Q \in \mathbb{C}^{n\times n}$ be unitary. We decompose $Q = (Q_\diamond, q_n)$ and obtain

$$Q^*AQ = \begin{pmatrix} Q_\diamond^* A Q_\diamond & Q_\diamond^* A q_n \\ q_n^* A Q_\diamond & q_n^* A q_n \end{pmatrix}.$$

If $q_n^* A = \lambda q_n^*$, then $(Q^*AQ)(n, :) = \lambda e_n^*$. By deflation, the process can be continued to obtain an upper triangular matrix whose diagonal elements are the eigenvalues of A. If A is hermitian, the resulting matrix is diagonal.

QR iteration. Let $\kappa \in \mathbb{C}$ and

$$q_n^* = \frac{e_n^*(A - \kappa\mathrm{Id})^{-1}}{\|e_n^*(A - \kappa\mathrm{Id})^{-1}\|}.$$

If $A - \kappa\mathrm{Id} = QR$, then $q_n^* = r_{nn} e_*(A - \kappa\mathrm{Id})^{-1}$. That is, the nth column of Q is an approximate left eigenvector of A.

Literature. [S2, Lecture 15]

8 Exam preparation

8.1 Summary (31.1.)

QR iteration. If $A - \kappa\mathrm{Id} = QR$, then

$$Q^*AQ = RQ + \kappa\mathrm{Id}.$$

```
A=ones(3,3)+diag([1,2,3]);
for k=1:10, [Q,R]=qr(A); A=R*Q, pause, end
```

Rayleigh quotient shift. Let $A^{(k)}$ be the kth iterate. Then,

$$\kappa = e_n^* A^{(k)} e_n = a_{nn}$$

is viewed as an approximate eigenvalue.

```
A=ones(3,3)+diag([1,2,3]); l=[];
while length(A(:))>1, kappa=A(end,end);
[Q,R]=qr(A-kappa*eye(size(A))); A=R*Q+kappa*eye(size(A)), pause
if norm(A(end,1:end-1))<1e-4, l=[l;A(end,end)];
A=A(1:end-1,1:end-1); end, end, l=[l;A(end,end)]
```

Summary. What is ...?

Newton's method. We solve $f(x_*) = 0$ by $x_{k+1} = x_k - f'(x_k)^{-1} f(x_k)$.

Interpolation. We construct p with $p(x_j) = y_j$ for all $j =, \ldots, n$.

> **Algebraic.** $p(x) = a_n x^n + \ldots + a_0 = y_0 \ell_0(x) + \ldots + y_n \ell_n(x)$.
> **Trigonometric.** $p(x) = c_n e^{inx} + \ldots + c_0$. (FFT)
> **Cubic spline.** $p \in C^2[a,b]$ is cubic on all $[x_{j-1}, x_j]$.

Quadrature. We compute $I(f) = \int_a^b f(x)dx$ by $Q(f) = \sum_{j=0}^n w_j f(x_j)$.

> **Composite methods.** $Q(f) = I(p)$ on each $[x_{j-1}, x_j]$.
> **Gaussian rules.** $I(p) = Q(p)$ for all p of degree $\leq 2n+1$.
> **Monte Carlo methods.** Pseudorandom nodes, $w_j = 1/(n+1)$ for all j.

Linear systems. We solve $Ax = b$ for $A \in \mathbb{C}^{n\times n}$ invertible.

> **Gaussian elimination.** We triangularize $A = LU$ by $\frac{2}{3}n^3$ flops.
> **Iterative methods.** $A = M - N$, $x_{k+1} = M^{-1}(b + Nx_k)$.

Least squares problems. Minimize $\|Ax - b\|_2$ for $\mathrm{rank}(A) = n \leq m$.

> **Normal equations.** We solve $A^*Ax = A^*b$. Cholesky: $\frac{1}{3}n^3$ flops.
> **QR factorization.** $A = QR$. Householder: $2n^2(m - \frac{1}{3}n)$ flops.

Eigenvalue problems. We solve $Ax = \lambda x$ for $A \in \mathbb{C}^{n\times n}$.

> **Power iterations.** $A^k x$ converges to a dominant eigenvector.
> **QR iteration.** $A_k - \kappa_k\mathrm{Id} = QR$, $A_{k+1} = RQ + \kappa_k\mathrm{Id}$

8.2 Questions (2.2.)

8.3 Applications (7.2.)

8.4 Exam (9.2.)

References

[DR] P. Davis, P. Rabinowitz, Methods of numerical integration (2nd ed.), Dover Publications, 2007.

[B] F. Bornemann, A Model for Understanding Numerical Stability. IMA J. Numer. Anal. 27, 219–231, 2007.

[H] N. Higham, Accuracy and Stability of Algorithms (2nd ed.), SIAM, 2002.

[K] C. Kelley, Iterative methods for linear and nonlinear equations, SIAM, 1995.

[M] C. Moler, Numerical Computing with MATLAB (rev. reprint), SIAM, 2008.

[O] M. Overton, Numerical Computing with IEEE Floating Point Arithmetic, SIAM, 2001.

[S] G. Stewart, Afternotes on numerical analysis, SIAM, 1996.

[S2] G. Stewart, Afternotes goes to graduate school, SIAM, 1998.

[T] L. N. Trefethen, Approximation theory and approximation practice, June 2011, `http://www2.maths.ox.ac.uk/chebfun/ATAP/`.

[TB] L. N. Trefethen, D. Bau, Numerical Linear Algebra, SIAM, 1997.