

## CS/MATH 3414 Lab # 1

Filon's method (Shampine, Allen, Pruess, pp. 128–130) for approximating Fourier integrals is

$$\int_a^b f(x) \cos(\omega x) dx = h[\alpha(f(b) \sin(\omega b) - f(a) \sin(\omega a)) + \beta C_e + \gamma C_o] + \mathcal{O}(h^3),$$

where  $\theta = \omega h \ll 1$  and

$$\begin{aligned}\alpha &= (\theta^2 + \theta \sin \theta \cos \theta - 2 \sin^2 \theta) / \theta^3, \\ \beta &= 2(\theta(1 + \cos^2 \theta) - 2 \sin \theta \cos \theta) / \theta^3, \\ \gamma &= 4(\sin \theta - \theta \cos \theta) / \theta^3.\end{aligned}$$

For  $\theta$  small, cancellation error followed by division by a very small  $\theta^3$  is disastrous in these formulas for  $\alpha$ ,  $\beta$ ,  $\gamma$ . A possible remedy is suggested by expanding everything in Maclaurin series, and combining the series analytically. For example,

$$\alpha = \frac{2}{45}\theta^3 - \frac{2}{315}\theta^5 + \frac{2}{4725}\theta^7 - \frac{8}{467,775}\theta^9 + \frac{4}{8,513,505}\theta^{11} + \mathcal{O}(\theta^{13}).$$

Such a series is obtained in Mathematica by, e.g.,

```
alphaser[t_] = Normal[Series[(t^2 + t Sin[t] Cos[t] - 2 (Sin[t])^2) / t^3,
    {t, 0, 11}]] .
```

Find similar series formulas for  $\beta$  and  $\gamma$ .

To compare the trigonometric formulas with the series formulas, take as “truth” the normal Mathematica machine precision (64 bit) evaluation of the trig formulas. For  $\theta = .05, .06, \dots, .15$ , evaluate  $\alpha$ ,  $\beta$ , and  $\gamma$  to seven significant digits using both the trigonometric and series formulas, and compute the absolute errors for both formulas. Make a plot using `LogListPlot[data]` for each of  $\alpha$ ,  $\beta$ ,  $\gamma$  showing the absolute errors for the two formulas.

`LogListPlot` is in the Mathematica package `Graphics`Graphics``. Functions for simulating computer arithmetic are in the package `NumericalMath`ComputerArithmetic``. To compute  $\alpha$  using 7-digit floating point arithmetic one could use:

```
<<NumericalMath`ComputerArithmetic`
<<Algebra`Horner`
SetArithmetic[7, MixedMode -> True]
alphatrig[s_] := Module[{t, cost, sint}, t = ComputerNumber[s];
    cost = ComputerNumber[N[Cos[Normal[t]]]];
    sint = ComputerNumber[N[Sin[Normal[t]]]];
    (t^2 + t sint cost - 2 sint^2)/t^3 ]
alphaser[s_] := Function[t, Horner[Normal[Series[(t^2 + t Sin[t] Cos[t]
    - 2 (Sin[t])^2)/t^3, {t, 0, 11}]]][ComputerNumber[s]]
alpha[s_] = (s^2 + s Sin[s] Cos[s] - 2 (Sin[s])^2)/s^3
```

## CS/MATH 3414 Lab # 2

### A Summation Problem *or* Why a Little Mathematics is Better than a Lot of Dumb Computing

For  $0 \leq x \leq 1$ , consider the function defined by

$$\phi(x) = \sum_{k=1}^{\infty} \frac{1}{k(k+x)}.$$

Try to evaluate  $\phi(0)$  (say to 8 significant digits) directly by computing partial sums of the series. This can be done with Mathematica by

```
p[x_, limit_] := NSum[ 1/(k(k + x)), {k, limit, 1, -1}, NSumTerms -> limit ]
```

and then, for example,

```
p[0, 20]
```

gives the partial sum with 20 terms approximating  $\phi(0)$ .

Suggestion: figure out what  $n$  should be such that

$$\sum_{k=n+1}^{\infty} \frac{1}{k^2} < 10^{-8}.$$

Then compute  $p[0, n]$ . Do you think this partial sum will in fact be accurate to  $10^{-8}$ ?

## A Summation Problem (Part 2)

Consider now a series derived from  $\phi(x)$  that converges much faster than the series for  $\phi(x)$ .

Let

$$\psi(x) = \frac{\phi(x) - \phi(1)}{1 - x} = \sum_{k=1}^{\infty} \frac{1}{k(k+1)(k+x)}.$$

Note that  $\psi(2)$  can be evaluated exactly by figuring out the partial fraction expansion

$$\frac{1}{k(k+1)(k+2)} = \frac{A}{k} + \frac{B}{k+1} + \frac{C}{k+2}.$$

Then the series

$$\frac{\psi(x) - \psi(2)}{2 - x} = \sum_{k=1}^{\infty} \frac{1}{k(k+1)(k+2)(k+x)},$$

which converges rapidly, can be evaluated numerically. Finally, since  $\phi(1)$  and  $\psi(2)$  are known,  $\phi(x)$  can be deduced.

To determine how many terms to use in the sum, note that the error in the partial sum

$$\sum_{k=1}^N \frac{1}{k(k+1)(k+2)(k+x)},$$

which is what you compute, is bounded by

$$\sum_{k=N+1}^{\infty} \frac{1}{k(k+1)(k+2)(k+x)} \leq \sum_{k=N+1}^{\infty} \frac{1}{k^4} \leq \int_N^{\infty} \frac{dx}{x^4} = \frac{1}{3N^3} \leq .5 \cdot 10^{-7}.$$

Solving the last inequality for  $N$  tells you how many terms you need to meet the accuracy requirement.

For your information,  $\phi(0) = 1.64493\ 40668\ 48226$ .

---

## CS/MATH 3414 – Linear Algebra

**Definition.** A vector space over the field  $F$  (whose elements are called *scalars*) consists of a set  $V$  (whose elements are called *vectors*) together with a binary operation  $V \times V \rightarrow V$  denoted by  $(x, y) \rightarrow x + y$ , and a binary operation  $F \times V \rightarrow V$  denoted by  $(\lambda, x) \rightarrow \lambda x$ , with the properties:

- (1)  $(x + y) + z = x + (y + z) \quad \forall x, y, z \in V$ ,
- (2)  $x + y = y + x \quad \forall x, y \in V$ ,
- (3)  $\exists$  a unique element  $0 \in V$  such that  $x + 0 = x \quad \forall x \in V$ ,
- (4)  $\forall x \in V \quad \exists$  a unique element  $-x \in V$  such that  $x + (-x) = 0$ ,
- (5)  $(\lambda + \mu)x = \lambda x + \mu x \quad \forall \lambda, \mu \in F$  and  $\forall x \in V$ ,
- (6)  $(\lambda\mu)x = \lambda(\mu x) \quad \forall \lambda, \mu \in F$  and  $\forall x \in V$ ,
- (7)  $\lambda(x + y) = \lambda x + \lambda y \quad \forall \lambda \in F$  and  $\forall x, y \in V$ ,
- (8)  $0x = 0 \quad \forall x \in V$ ,
- (9)  $1x = x \quad \forall x \in V$ .

**Definition.** A *subspace*  $S$  of a vector space  $V$  is a subset  $S \subset V$  such that  $S$  is also a vector space.

**Definition.** The vectors  $x_1, \dots, x_k \in V$  are *independent* if the linear combination  $\alpha_1 x_1 + \dots + \alpha_k x_k = 0 \Rightarrow \alpha_1 = \dots = \alpha_k = 0$ .  $x_1, \dots, x_k$  are said to *generate* (or *span*)  $V$  if for each  $x \in V \exists \alpha_1, \dots, \alpha_k \in F$  such that  $x = \alpha_1 x_1 + \dots + \alpha_k x_k$ . The vectors  $x_1, \dots, x_k$  are a *basis* for  $V$  if they are independent and generate  $V$ .  $V$  is called *finite dimensional* if  $V$  is generated by a finite set of vectors.

**Theorem.** Every finite dimensional vector space  $V$  has a basis.

**Theorem.** Every basis of a finite dimensional vector space  $V$  has the same number of elements.

**Definition.** The *dimension*  $\dim V$  of a finite dimensional vector space  $V$  is the number of elements in a basis of  $V$ .

**Theorem.** Let  $x_1, \dots, x_n \in V$  and  $\dim V = n$ . Then the following are equivalent:

- (1)  $x_1, \dots, x_n$  are a basis for  $V$ ,
- (2)  $x_1, \dots, x_n$  are independent,
- (3)  $x_1, \dots, x_n$  generate  $V$ .

---

Examples of vector spaces:

1.  $F = Q = \{\text{rational numbers}\}$ ,  $V = F^n = \{n\text{-tuples of elements of } F\}$ .
  2.  $F = E = \{\text{real numbers}\}$ ,  $V = F^n$ .
  3.  $F = C = \{\text{complex numbers}\}$ ,  $V = F^n$ .
  4.  $F$  a field,  $V = F^{m \times n} = \{m \times n \text{ matrices with elements in } F\}$ .
  5.  $F$  a field,  $V = \mathcal{P}_n(F) = \{\text{polynomials of degree } \leq n \text{ with coefficients in } F\}$ .
  6.  $F = E$ ,  $V = C[a, b] = \{\text{continuous real-valued functions defined on the closed interval } [a, b]\}$ .
-

**Definition.** Let  $V$  and  $W$  be vector spaces over the field  $F$ . A function  $T : V \rightarrow W$  is a *linear transformation (homomorphism)* if  $T(\alpha x + \beta y) = \alpha T(x) + \beta T(y)$  for all  $x, y \in V$  and  $\alpha, \beta \in F$ . The *kernel* (null space) of  $T$  is

$$\ker T = \{x \mid T(x) = 0\}.$$

The *image* (range) of  $T$  is

$$\operatorname{im} T = \{T(x) \mid x \in V\}.$$

The identity function  $id_V$  on  $V$  maps every vector to itself.  $T : V \rightarrow W$  is *invertible* if  $\exists$  a homomorphism  $S : W \rightarrow V$  such that  $S \circ T = id_V$  and  $T \circ S = id_W$ .

**Theorem.** Let  $V$  be a finite dimensional vector space and  $T : V \rightarrow W$  a homomorphism. Then

$$\dim V = \dim (\ker T) + \dim (\operatorname{im} T).$$

**Theorem.** Let  $a_1, \dots, a_n$  be a basis for the vector space  $V$ , and  $T : V \rightarrow W$  be a homomorphism. Then the following hold:

- (1)  $\ker T = \{0\} \Leftrightarrow T(a_1), \dots, T(a_n)$  are independent;
- (2)  $\operatorname{im} T = W \Leftrightarrow T(a_1), \dots, T(a_n)$  generate  $W$ ;
- (3)  $T(a_1), \dots, T(a_n)$  are a basis for  $W \Leftrightarrow T$  is invertible.

### Matrix Theory

A real  $m \times n$  matrix  $A \in E^{m \times n}$  can be regarded as a linear transformation from  $E^n$  to  $E^m$ , sending a vector  $x \in E^n$  to the vector  $Ax \in E^m$ . Thus all of the above theorems apply to matrices.

**Definition.** For a scalar  $A \in E^{1 \times 1}$ ,  $\det A = A$ . For  $A \in E^{n \times n}$ , the *determinant* of  $A$  is

$$\det A = \sum_{i=1}^n (-1)^{i+1} A_{i1} \det A[i, 1],$$

where  $A[i, j]$  is the submatrix of  $A$  obtained by deleting the  $i$ th row and  $j$ th column. The number  $\operatorname{cof} A_{ij} = (-1)^{i+j} \det A[i, j]$  is called the *cofactor* of the  $i, j$  matrix element  $A_{ij}$ . The *adjoint* matrix of  $A$ , denoted  $\operatorname{adj} A$ , has as its  $i, j$  element the number  $\operatorname{cof} A_{ji}$ .

**Lemma.** Let  $A \in E^{n \times n}$ . For any  $j$ ,  $1 \leq j \leq n$ ,

$$\det A = \sum_{i=1}^n (-1)^{i+j} A_{ij} \det A[i, j] = \sum_{i=1}^n (-1)^{j+i} A_{ji} \det A[j, i].$$

**Lemma.** Let  $A, B \in E^{n \times n}$  and  $I$  denote the  $n \times n$  identity matrix. Then

- (1)  $\det A = \det A^t$ ,
- (2)  $\det(AB) = \det A \det B$ ,
- (3)  $A(\operatorname{adj} A) = (\operatorname{adj} A)A = (\det A)I$ .

**Theorem.** Let  $A \in E^{n \times n}$ . Then the following are equivalent:

- (1)  $\ker A = \{0\}$  (columns of  $A$  are independent),
- (2)  $\text{im } A = E^n$  (columns of  $A$  generate  $E^n$ ),
- (3) the columns  $A_1, \dots, A_n$  of  $A$  are a basis for  $E^n$ ,
- (4) the rows  $A_1, \dots, A_n$  of  $A$  are a basis for  $E^n$ ,
- (5)  $\det A \neq 0$ ,
- (6)  $A$  is invertible.

**Definition.** A matrix  $A \in E^{n \times n}$  is upper triangular, lower triangular, diagonal, if  $A_{ij} = 0$  for  $i > j$ ,  $A_{ij} = 0$  for  $i < j$ ,  $A_{ij} = 0$  for  $i \neq j$ , respectively.

**Lemma.** If  $A \in E^{n \times n}$  is upper triangular, lower triangular, or diagonal, then

$$\det A = \prod_{i=1}^n A_{ii}.$$

**Lemma.** Let  $A \in E^{n \times n}$  be invertible. Then if  $A$  is upper triangular, lower triangular, or diagonal, the inverse  $A^{-1}$  is also upper triangular, lower triangular, or diagonal, respectively. If  $B, C \in E^{n \times n}$  are upper triangular, lower triangular, or diagonal, then the product  $BC$  is also upper triangular, lower triangular, or diagonal, respectively.

## CS/MATH 3414 – Norms and Inner Products

Let  $V$  be a real vector space. An *inner product* on  $V$  is a function from  $V \times V \rightarrow$  reals. The inner product of two vectors  $u, v$  is denoted by  $(u, v)$  or  $\langle u, v \rangle$ , and has the properties:

1.  $(u + v, w) = (u, w) + (v, w)$  for all  $u, v, w \in V$ ;
2.  $(\alpha u, v) = \alpha(u, v)$  for all real  $\alpha$  and  $u, v \in V$ ;
3.  $(u, v) = (v, u)$  for all  $u, v \in V$ ;
4.  $(u, u) > 0$  for all  $u \neq 0, u \in V$ .

The following are examples of inner products:

1.  $V = E^n \equiv \{n\text{-tuples of real numbers}\}, (x, y) = \sum_{i=1}^n x_i y_i$ ,
2.  $V = \mathcal{P}_2 = \{\text{polynomials of degree } \leq 2\}, (p, q) = \sum_{i=1}^3 p(i)q(i)$ ,
3.  $V = \mathcal{P} = \{\text{all polynomials}\}, (p, q) = \int_{-1}^1 p(x)q(x)dx$ .

Two vectors  $u, v$  are *orthogonal* (think “perpendicular”) if  $(u, v) = 0$ . A sequence of vectors  $\varphi_1, \varphi_2, \dots, \varphi_k$  is *orthonormal* if

$$(\varphi_i, \varphi_j) = \delta_{ij} = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases} \quad \forall i, j.$$

Example: the vectors  $\varphi_0 = 1, \varphi_1 = x, \varphi_2 = 3x^2 - 1, \varphi_3 = 5x^3 - 3x$  are mutually orthogonal with respect to the inner product  $(f, g) = \int_{-1}^1 f(x)g(x)dx$  on the vector space  $\mathcal{P}$ .

A *norm* on  $V$  is a function from  $V \rightarrow$  reals denoted by  $\|u\|$  with the properties

1.  $\|u\| \geq 0$  for all  $u \in V$  with equality if and only if  $u = 0$ ;
2.  $\|\alpha u\| = |\alpha| \|u\|$  for all real  $\alpha$  and  $u \in V$ ;
3.  $\|u + v\| \leq \|u\| + \|v\|$  for all  $u, v \in V$ .

Examples of norms:

1.  $V = E^n, \|x\|_p = (\sum_{i=1}^n |x_i|^p)^{1/p}, p \geq 1$ .
2.  $V = E^n, \|x\|_\infty = \max_{1 \leq i \leq n} |x_i|$ .
3.  $V = \mathcal{P} = \{\text{polynomials}\}, \|p\|_\infty = \max_{a \leq x \leq b} |p(x)|$ .
4. (IMPORTANT)  $\|u\| = \sqrt{(u, u)}$ , where  $(u, v)$  is any inner product on any vector space  $V$ .
5.  $\|f\|_1 = \int_a^b |f(x)|dx$  is a norm on  $V = C[a, b] = \{\text{continuous functions defined on the closed interval } [a, b]\}$ .

**IMPORTANT FACT** (Cauchy-Schwarz inequality):  $|(u, v)| \leq \|u\| \|v\|$ .

*Matrix norms:*

Choose a norm  $\|\cdot\|$  on  $E^n$ , and let  $A$  be an  $n \times n$  matrix. The *norm of  $A$*  with respect to the vector norm  $\|\cdot\|$  is

$$\|A\| = \sup_{\|x\| \neq 0} \frac{\|Ax\|}{\|x\|} = \max_{\|x\|=1} \|Ax\|.$$

For vector norm

the corresponding matrix norm is

$$\|x\|_\infty$$

$$\|A\|_\infty = \max_i \sum_{j=1}^n |A_{ij}|$$

$$\|x\|_1$$

$$\|A\|_1 = \max_j \sum_{i=1}^n |A_{ij}|$$

$$\|x\|_2$$

$$\|A\|_2 = \sqrt{\rho(A^t A)} = \text{square root of largest eigenvalue of } A^t A$$

### CS/MATH 3414 – Lab # 3

#### (5) 1. Norms.

The goal of this exercise is to understand the geometry of different norms in two dimensions. Find all the 2-dimensional vectors  $(x_1, x_2)$  in  $E^2$  whose  $p$ -norms are one, for  $p = 1, 2, 3, 5, 10, 20$ , and draw all six cases together in one picture.

`ImplicitPlot[ eqn, {x, a, b}, {y, c, d} ]` uses a contour plotting method, drawing a graph of the set of points that satisfy the equation  $eqn$ . The variable  $x$  is associated with the horizontal axis and ranges from  $a$  to  $b$ . The variable  $y$  is associated with the vertical axis and ranges from  $c$  to  $d$ . For example,

```
ImplicitPlot[x - y == 0, {x, 0, 1}, {y, 0, 1}]
```

draws a line segment between  $(0, 0)$  and  $(1, 1)$  which satisfies the equation  $x - y = 0$ .

#### (5) 2. Matrix conditioning.

The goal of this part is to understand the action of a matrix on vectors of unit length; everything is considered in two dimensions for simplicity. The following `ParametricPlot[]` command plots the image of the unit circle under a matrix  $A$ , i.e., the set

$$\{Ax \mid \|x\|_2 = 1\}.$$

```
ParametricPlot[{A[[1]] . {Cos[t], Sin[t]}, A[[2]] . {Cos[t], Sin[t]}}, {t, 0, 2 Pi}]
```

plots the action of  $A$  on the points on the unit circle, which must be parametrized by  $\{Cos[t], Sin[t]\}$  for  $0 \leq t \leq 2\pi$ . Plot this image for several matrices  $A$ , such as

$$\begin{pmatrix} 1 & 2 \\ 2 & 1.0 \end{pmatrix}, \quad \begin{pmatrix} 1 & 2 \\ 2 & 4.001 \end{pmatrix}, \quad \begin{pmatrix} 3 & 17 \\ 2 & 11.0 \end{pmatrix}.$$

A matrix is said to be *ill conditioned* if it greatly distorts the unit circle, i.e., if the ratio of the major axis to the minor axis in the image is large. Which of the above matrices is ill conditioned? The *condition number*

$$\text{cond } A = \|A\| \|A^{-1}\|,$$

a measure of ill conditioning, can be computed in Mathematica by

```
cond[a_] := Module[{t}, t = SingularValues[a]; Max[ t[[2]] ]/Min[ t[[2]] ] ]
```

## CS/MATH 3414 Lab # 4

(20) 1. Consider the  $n \times n$  linear system  $Ax = b$ , where

$$A_{ij} = (n + 1 - j)^{n+1-i}, \quad 1 \leq i, j \leq n \quad \text{and} \quad b_i = \sum_{j=1}^n (-1)^{j+1} A_{ij}, \quad 1 \leq i \leq n.$$

Denote the exact solution  $(1, -1, 1, -1, \dots)$  by  $\bar{x}$ , a computed solution by  $\hat{x}$ , the error by  $e = \hat{x} - \bar{x}$ , and the residual by  $r = b - A\hat{x}$ . For  $n = 5, 6, \dots, 15$  compute

- (a)  $\|e\|_\infty / \|\bar{x}\|_\infty$ ,
- (b)  $\|r\|_\infty / \|b\|_\infty$ ,
- (c)  $\text{cond } A$  (estimate only),
- (d)  $|\det A|$ .

Some engineering textbooks state that the way to check for ill conditioning is to test for a small determinant. What do you think of that? It is also almost universally believed that the way to verify an answer is to “plug it back in” and check if the residual  $\|r\|/\|b\|$  is small. How about that?

---

A very sophisticated software package, LAPACK, is available to you. The routines performing the LU decomposition and system solution (described in the text) are DGETRF, DGETRS, and DGECON (DGTSV for tridiagonal systems) from LAPACK. The complete LAPACK library (source) is in the UNIX directories

```
malamute.cslab.vt.edu:/home/faculty/lw/mathsoft/lapack.dir
```

and

```
malamute.cslab.vt.edu:/home/faculty/lw/mathsoft/blas.dir
```

and the object libraries are in

```
malamute.cslab.vt.edu:/home/faculty/lw/bin/lapack.olb
```

and

```
malamute.cslab.vt.edu:/home/faculty/lw/bin/blas.olb
```

A sample UNIX (Korn shell) script for compiling is in

```
malamute.cslab.vt.edu:/home/faculty/lw/bin/ftn90
```

A wealth of Fortran 90 information can be found starting at the URLs:

```
http://www.nag.co.uk/nagware/Fortran90.html
```

```
http://csep1.phy.ornl.gov/CSEP/PL/PL.html
```

```
http://www.tc.cornell.edu/Services/Edu/Topics/Fortran90/less.asp
```

## MATRIX DECOMPOSITION THEOREMS

*Theorem 1.* Let  $A$  be a real  $n \times n$  matrix and  $A_k$  the submatrix of  $A$  formed from the first  $k$  rows and columns. Let  $\det A_k \neq 0$  for  $k = 1, \dots, n$ . Then  $\exists$  a unique unit (ones on the diagonal) lower triangular matrix  $L$  and a unique upper triangular matrix  $U$  such that  $A = LU$ .

*Theorem 2.* Under the hypothesis of Theorem 1,  $\exists$  a unique lower triangular matrix  $L$ , a unique diagonal matrix  $D$ , and a unique upper triangular matrix  $U$  such that  $A = LDU$ .

*Theorem 3.* Let  $A$  be symmetric ( $A = A^t$ ) and satisfy the hypothesis of Theorem 1. Then  $\exists$  a unique unit lower triangular matrix  $L$  and a unique diagonal matrix  $D$  such that  $A = LDL^t$ .

*Theorem 4.* Let  $A$  be symmetric and positive definite ( $x^t Ax > 0$  for all  $x \neq 0$ ). Then  $\exists$  a unique lower triangular matrix  $G$  with positive diagonal elements such that  $A = GG^t$ . ( $A = GG^t$  is the Cholesky factorization of  $A$ .)

*Definition.* A permutation matrix  $P$  is an  $n \times n$  matrix consisting of 0's and 1's, such that every row and column contains exactly one 1. (Intuitively,  $Py$  simply permutes the elements of the vector  $y$ .)

*Theorem 5.* Let  $A$  be an  $n \times n$  matrix. Then  $\exists$  a permutation matrix  $P$ , a unit lower triangular matrix  $L$ , and an upper triangular matrix  $U$  such that  $PA = LU$ .

### DIRECT ALGORITHMS

Suppose it is known beforehand that  $A$  has an  $LU$  factorization without interchanging rows. Then, carefully examining the equation

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \cdots & a_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \cdots & a_{nn} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ \ell_{21} & 1 & 0 & \cdots & 0 \\ \ell_{31} & \ell_{32} & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \ell_{n1} & \ell_{n2} & \ell_{n3} & \cdots & 1 \end{pmatrix} \begin{pmatrix} u_{11} & u_{12} & u_{13} & \cdots & u_{1n} \\ 0 & u_{22} & u_{23} & \cdots & u_{2n} \\ 0 & 0 & u_{33} & \cdots & u_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & u_{nn} \end{pmatrix}$$

shows that  $L$  and  $U$  can be determined recursively. Precisely,

$$\begin{aligned} & \text{for } k = 1, \dots, n \\ & u_{km} = a_{km} - \sum_{j=1}^{k-1} \ell_{kj} u_{jm}, \quad m = k, \dots, n \\ & \ell_{pk} = \left( a_{pk} - \sum_{j=1}^{k-1} \ell_{pj} u_{jk} \right) / u_{kk}, \quad p = k + 1, \dots, n \end{aligned}$$

These formulas (known as the Crout or Doolittle algorithm) require exactly the same computational effort as Gaussian elimination, but are inappropriate unless it is known a priori that pivoting is not necessary.

**Definition.** Let  $u \in C^n$ ,  $u \neq 0$ . A Householder reflection is a matrix of the form

$$P = I - \frac{1}{\beta}uu^*, \quad \text{where } \beta = \frac{\|u\|^2}{2}.$$

$A^*$  denotes the conjugate transpose  $(\bar{A})^t$  of a complex matrix  $A \in C^{m \times n}$ .

**Theorem.** Householder reflections  $P$  are Hermitian ( $P^* = P$ ), unitary ( $P^*P = I$ ), and self-inverse ( $P^{-1} = P$ ).

**Theorem.** Let  $a \in C^n$ ,  $a \neq 0$ ,  $\alpha = e^{i \arg(a_1)}\|a\|$ ,  $u = a + \alpha e_1$ ,  $\beta = u_1 \bar{\alpha}$ . Then  $P = I - \frac{1}{\beta}uu^*$  is a Householder reflection and  $Pa = -\alpha e_1$ . For any  $b \in C^n$ ,  $Pb = b - (u^*b/\beta)u$ .

**Proof.**  $u^*u = (a^* + \bar{\alpha}e_1^t)(a + \alpha e_1) = \|a\|^2 + 2\Re(a_1)\bar{\alpha} + \|a\|^2 = 2(\|a\|^2 + \Re(a_1)\bar{\alpha}) = 2(\|a\|^2 + a_1\bar{\alpha}) = 2(a_1 + \alpha)\bar{\alpha} = 2u_1\bar{\alpha} = 2\beta$  so  $P = I - \frac{1}{\beta}uu^*$  is a Householder reflection. Now

$$Pa = a - (u^*a/\beta)u = a - \frac{(a^* + \bar{\alpha}e_1^t)a}{\beta}(a + \alpha e_1) = a - \frac{(\|a\|^2 + \bar{\alpha}a_1)}{\beta}(a + \alpha e_1) = a - (a + \alpha e_1) = -\alpha e_1.$$

Q. E. D.

**Corollary.** For  $a \in C^n$ ,  $a \neq 0$ , and  $1 \leq r < n \exists$  a Householder reflection  $P$  such that

$$Pa = (a_1, \dots, a_{r-1}, -\alpha, 0, \dots, 0)^t.$$

**Theorem.** Let  $A \in C^{n \times n}$ . Then  $\exists$  a unitary matrix  $Q$  such that  $QA = R$  is upper triangular.

**Proof.** The proof is by induction on  $n$ . There is nothing to prove for  $n = 1$ . Assume the result for  $n - 1$ . By the above theorem, taking  $a = A_{\cdot 1}$ , there is a Householder reflection  $P$  such that

$$PA = \begin{pmatrix} -\alpha & c \\ 0 & \tilde{A} \end{pmatrix}.$$

By the induction hypothesis,  $\exists$  unitary  $\tilde{P} \in C^{(n-1) \times (n-1)}$  such that  $\tilde{P}\tilde{A} = \tilde{R}$  is upper triangular. Then

$$QA = \begin{pmatrix} 1 & 0 \\ 0 & \tilde{P} \end{pmatrix} PA = \begin{pmatrix} -\alpha & c \\ 0 & \tilde{R} \end{pmatrix} = R$$

is upper triangular and  $Q$  is unitary since the product of unitary matrices is unitary. Q. E. D.

**Observation.**  $Q$  may be computed as a product of Householder reflections, and  $Q$  is orthogonal if  $A$  is real.

**Definition.** A matrix  $A \in C^{n \times n}$  is *upper Hessenberg* if  $A_{ij} = 0$  for  $i > j + 1$ .

**Theorem.** Let  $A \in C^{n \times n}$ . Then  $\exists$  a unitary matrix  $Q$  such that  $QAQ^* = H$  is upper Hessenberg.  $Q$  may be computed as a product of Householder reflections, and  $Q$  is orthogonal if  $A$  is real.

**Corollary.** If  $A \in C^{n \times n}$  is Hermitian or real symmetric, then  $\exists$  a unitary matrix  $Q$  such that  $QAQ^* = T$  is tridiagonal ( $T_{ij} = 0$  for  $|i - j| > 1$ ).

## CONVERGENCE OF POLYNOMIAL INTERPOLANTS

Let  $f$  be a continuous function on  $[a, b]$ , and let  $P_n(x)$  be the polynomial of degree  $\leq n$  which interpolates  $f$  at  $n + 1$  distinct points  $x_0^{(n)}, \dots, x_n^{(n)}$  of  $[a, b]$ . It is reasonable to ask whether or not  $P_n(x)$  converges to  $f(x) \forall x \in [a, b]$  as  $n \rightarrow \infty$  and the points  $x_0^{(n)}, \dots, x_n^{(n)}$  become dense in  $[a, b]$ . Bernstein in 1912 proved that the polynomials interpolating  $f(x) = |x|$  at equally spaced points in  $[-1, 1]$  converge to  $f(x)$  **only** at  $-1, 0,$  and  $1$ . Runge in 1901 proved that the interpolating polynomials to  $f(x) = 1/(1 + 25x^2)$  at equally spaced points in  $[-1, 1]$  diverge, which is even more surprising since  $1/(1 + 25x^2)$  has infinitely many derivatives. However, Bernstein also proved that if the points

$$x_k^{(n)} = \cos \frac{2k + 1}{2n + 2} \pi$$

are used, then the interpolating polynomials for both  $|x|$  and  $1/(1 + 25x^2)$  converge uniformly to the function. Thus the difficulty seemed to be with equally spaced points. In 1914 Faber shocked everyone by proving that for **any** sequence of interpolation points, there exists a continuous function for which the interpolation polynomials at those points diverge.

The convergence of  $P_n(x)$  to  $f(x)$  has to do with numbers  $\lambda_n$  (called Lebesgue constants) and numbers  $E_n(f)$  (degree of approximation).  $E_n(f) = \inf \|f - P\|_\infty$  taken over all polynomials  $P$  of degree  $\leq n$ . In the 1920's, Jackson proved a series of theorems about  $E_n(f)$ . A corollary of the Jackson Theorems is that  $E_n(f) = \mathcal{O}(1/n)$  if  $f$  satisfies a Lipschitz condition

$$|f(x) - f(y)| \leq L|x - y| \quad \forall x, y \in [a, b].$$

Given points  $x_0^{(n)}, \dots, x_n^{(n)}$ , let

$$L_{n,i}(x) = \prod_{\substack{j=0 \\ j \neq i}}^n \frac{x - x_j^{(n)}}{x_i^{(n)} - x_j^{(n)}}$$

be the Lagrange polynomials. Then

$$\lambda_n(x) = \sum_{i=0}^n |L_{n,i}(x)|, \quad \text{and} \quad \lambda_n = \max_{[a,b]} |\lambda_n(x)|$$

defines the Lebesgue constants. Note that they depend on the interval  $[a, b]$  and the interpolation points, and have nothing to do with the function  $f$ .  $E_n(f)$  depends only on  $[a, b]$  and  $f$ .

**Theorem 1.** Let  $f \in C[a, b]$  and  $P_n$  be the polynomial interpolating at  $x_0^{(n)}, \dots, x_n^{(n)} \in [a, b]$ . If  $x \in [a, b]$  and  $\lim_{n \rightarrow \infty} \lambda_n(x) E_n(f) = 0$ , then  $P_n(x) \rightarrow f(x)$ . If  $\lambda_n E_n(f) \rightarrow 0$ , then  $\|f - P_n\|_\infty \rightarrow 0$ .

**Proof.** Let  $Q(x)$  be the polynomial of degree  $\leq n$  of best approximation to  $f$ , so  $|f(x) - Q(x)| \leq E_n(f)$ .

$$P_n(x) = \sum_{i=0}^n f(x_i^{(n)}) L_{n,i}(x) \quad \text{and} \quad Q(x) = \sum_{i=0}^n Q(x_i^{(n)}) L_{n,i}(x).$$

Then

$$\begin{aligned} |f(x) - P_n(x)| &\leq |f(x) - Q(x)| + |Q(x) - P_n(x)| \\ &\leq E_n(f) + \sum_{i=0}^n \left| Q(x_i^{(n)}) - f(x_i^{(n)}) \right| |L_{n,i}(x)| \\ &\leq E_n(f) + E_n(f) \sum_{i=0}^n |L_{n,i}(x)| \\ &= E_n(f) (1 + \lambda_n(x)) \\ &\leq E_n(f) (1 + \lambda_n). \end{aligned}$$

Since  $E_n(f) \rightarrow 0$  by the Weierstrass theorem, the conclusions follow. Q.E.D.

**Theorem 2 (Bernstein).** For the Chebyshev points  $x_k^{(n)} = \cos \frac{2k+1}{2n+2} \pi$  on  $[-1, 1]$ ,  $\lambda_n = \mathcal{O}(\log n)$ .

**Theorem 3.** Let  $f$  satisfy a Lipschitz condition on  $[-1, 1]$  and  $x_k^{(n)}$  be the Chebyshev points. Then the interpolating polynomials  $P_n(x)$  converge to  $f(x)$  uniformly on  $[-1, 1]$ .

**Proof.** By the Jackson Theorem,  $E_n(f) = \mathcal{O}(1/n)$ . By Theorem 2,  $\lambda_n = \mathcal{O}(\log n)$ . Therefore  $\lambda_n E_n(f) = \mathcal{O}(\log n) \mathcal{O}(1/n) = \mathcal{O}\left(\frac{\log n}{n}\right) \rightarrow 0$ . Therefore by Theorem 1,  $\|f - P_n\|_\infty \rightarrow 0$ . Q.E.D.

This explains the convergence of the interpolating polynomials at the Chebyshev points for  $F(x) = |x|$  and  $f(x) = 1/(1 + 25x^2)$ .

### CS/MATH 3414 Lab # 5

- (3) 1. Let  $F(x) = |x|$ ,  $x_j = -1 + 2j/n$ ,  $j = 0, \dots, n$ . Compute the polynomial  $P_n$  interpolating  $F$  at the  $x_j$  for  $n = 6, 8, 10, 12, 14$ . On  $[-1, 1]$  graph the polynomials  $P_n$  and  $F$  (either by hand or using Mathematica).
- (3) 2. Same as 1. except use the Chebyshev points  $x_j = \cos \frac{2j+1}{2n+2} \pi$ ,  $j = 0, \dots, n$ .
- (3) 3. The problem here is to compute the Hermite interpolating polynomial. For any  $f \in C[-1, 1]$ , Féjer proved that the Hermite polynomial  $P_n(x)$  which satisfies  $P_n(x_k) = f(x_k)$ ,  $P_n'(x_k) = 0$  at the Chebyshev points converges uniformly to  $f$  on  $[-1, 1]$  as  $n \rightarrow \infty$ . Verify this by graphing  $P_n(x)$  and  $f(x)$  on  $[-1, 1]$  for  $n = 4, 6, 8, 10, 12$  and  $f(x) = |x|$ .
- (3) 4. Find the Lagrange form of the polynomial interpolating the data points  $(-2, -35)$ ,  $(-1, -4)$ ,  $(0, -3)$ ,  $(1, -2)$ ,  $(2, 29)$ ,  $(3, 240)$ .
- (3) 5. Find the Newton form of the polynomial interpolating the data:  $f(0) = -3$ ,  $f(1) = -2$ ,  $f'(1) = 5$ ,  $f''(1) = 20$ ,  $f(2) = 29$ ,  $f(-1) = -4$ .

---

Extra Credit

- (12) 6. Let  $P(x) = \sum_{k=0}^n a_k \prod_{i=0}^{k-1} (x - x_i)$  and consider the divided difference table below with  $n + 2$  distinct points  $z, x_0, \dots, x_n$ :

|           |              |       |          |           |       |
|-----------|--------------|-------|----------|-----------|-------|
| $z$       | $b_0$        |       |          |           |       |
| $x_0$     | $a_0$        | $b_1$ |          |           |       |
| $x_1$     | $P(x_1)$     | $a_1$ | $b_2$    |           |       |
| $x_2$     | $P(x_2)$     |       | $\ddots$ |           |       |
| $\vdots$  | $\vdots$     |       |          | $b_{n-1}$ |       |
| $x_{n-1}$ | $P(x_{n-1})$ |       |          | $\ddots$  | $b_n$ |
| $x_n$     | $P(x_n)$     |       |          | $a_{n-1}$ |       |
|           |              |       |          |           | $a_n$ |

- 1) If  $b_0 = P(z)$ , write the relationship between the  $a$ 's and  $b$ 's in this divided difference table.
- 2) Given any point  $z$ , show that an algorithm to evaluate  $P(z)$  is

```

 $\tilde{b}_n := a_n;$ 
for  $k := n - 1$  step  $-1$  until  $0$  do
     $\tilde{b}_k := \tilde{b}_{k+1} * (z - x_k) + a_k;$ 
comment  $\tilde{b}_0 = P(z);$ 

```

- 3) Using polynomial division, show that

$$\frac{P(x)}{x - z} = \hat{b}_1 + \hat{b}_2(x - x_0) + \dots + \hat{b}_n(x - x_0) \cdots (x - x_{n-2}) + \frac{\hat{b}_0}{x - z}$$

and write the relationship between the  $a$ 's and  $\hat{b}$ 's. Conclude that

$$b_0 = \tilde{b}_0 = \hat{b}_0 = P(z), \quad a_n = b_n = \tilde{b}_n = \hat{b}_n, \quad \text{and } b_i = \tilde{b}_i = \hat{b}_i \text{ for } i = 0, \dots, n$$

and that

$$P(x) = b_0 + b_1(x - z) + b_2(x - z)(x - x_0) + \dots + b_n(x - z)(x - x_0) \cdots (x - x_{n-2}).$$

- 4) Use the algorithm in 2) to express the Newton polynomial from 5. in the Newton form

$$P(x) = \sum_{j=0}^5 d_j x^j.$$

## CS/MATH 3414 – Lab # 6

- (2) 1. The purpose of this part is to understand the effect of increasing the number of knots in a spline approximation. The Mathematica function

$$s = \text{Interpolation}[list, \text{InterpolationOrder} \rightarrow k]$$

generates a spline of degree  $k$  interpolating the points in  $list = \{\{x_1, f_1\}, \{x_2, f_2\}, \dots\}$ . Using the function

$$f(x) = \frac{1}{1 + 100x^2}$$

and  $n + 1$  equally spaced points in the interval  $[-1, 1]$ , plot  $f(x)$  and the splines  $s[x]$  of degree 3 interpolating  $f(x)$  at  $n + 1$  equally spaced points for  $n = 6, 10, 14$ .

- (2) 2. Consider the data

|        |      |      |      |      |      |      |     |      |      |     |      |     |      |      |      |
|--------|------|------|------|------|------|------|-----|------|------|-----|------|-----|------|------|------|
| $x$    | 0.9  | 1.3  | 1.9  | 2.1  | 2.6  | 3.0  | 3.9 | 4.4  | 4.7  | 5.0 | 6.0  | 7.0 | 8.0  | 9.2  | 10.5 |
| $f(x)$ | 1.3  | 1.5  | 1.85 | 2.1  | 2.6  | 2.7  | 2.4 | 2.15 | 2.05 | 2.1 | 2.25 | 2.3 | 2.25 | 1.95 | 1.4  |
| $x$    | 11.3 | 11.6 | 12.0 | 12.6 | 13.0 | 13.3 |     |      |      |     |      |     |      |      |      |
| $f(x)$ | 0.9  | 0.7  | 0.6  | 0.5  | 0.4  | 0.25 |     |      |      |     |      |     |      |      |      |

from the top profile of a ruddy duck in flight. Plot the original data (using `ListPlot` and `PlotStyle`  $\rightarrow \{\{\text{PointSize}[.02]\}\}$ ), the interpolating polynomial  $P(x)$  of degree 20, and the interpolating spline  $s(x)$  of degree 3. Comment on the results.

- (2) 3. This part shows the effect of a few well placed knots. Consider

$$y = \frac{x}{1/4 + x^2},$$

which is  $x = (\cot t)/2$ ,  $y = \sin 2t$ ,  $0 < t < \pi$ , in parametric form. Generate the knots by

$$t = \text{Sort}[N[\text{Table}[\{\text{Cot}[i \text{ Pi}/14.0]/2, \text{Sin}[i \text{ Pi}/7.0]\}, \{i, 1, 13, 1\}]]]$$

Plot both the Newton form of the interpolating polynomial and the interpolating cubic spline to this data.

- (4) 4. The Bernstein basis functions for  $\mathcal{P}_n$  are

$$B_i^n(x) = \binom{n}{i} \frac{(b-x)^{n-i}(x-a)^i}{(b-a)^n}, \quad i = 0, 1, \dots, n,$$

which can be computed recursively from

$$B_0^0(x) = 1, \quad B_{-1}^k = B_{k+1}^k(x) \equiv 0,$$

$$B_i^n(x) = \frac{(b-x)B_i^{n-1}(x) + (x-a)B_{i-1}^{n-1}(x)}{b-a}, \quad 0 \leq i \leq n, \quad n > 0.$$

Taking  $a = 0$ ,  $b = 1$ , a Bézier curve with control points (vectors)  $\{\vec{p}_i\}_{i=0}^n$  is

$$\vec{P}(x) = \sum_{i=0}^n \vec{p}_i B_i^n(x), \quad 0 \leq x \leq 1.$$

The ability to draw circles quickly is important to a graphics kernel or a hardware implementation. The first quadrant of the unit circle could be generated by  $x_i = \cos(i\pi/180)$ ,  $y_i = \sin(i\pi/180)$ ,  $i = 0, 1, \dots, 90$ , but this requires both a sine and cosine (or a sine and square root) for each point on the circle. As an alternative consider the four Bézier control points:  $\vec{p}_0 = (1, 0)$ ,  $\vec{p}_1 = (1, 0.552)$ ,  $\vec{p}_2 = (0.552, 1)$ ,  $\vec{p}_3 = (0, 1)$ .

- (a) Show that the associated vector Bézier curve is

$$\vec{P}(t) = (1 - 1.344t^2 + 0.344t^3, 1.656t - 0.312t^2 - 0.344t^3).$$

- (b) As  $t$  varies from 0 to 1,  $\vec{P}(t)$  traces a curve in the plane which approximates a quarter circle. Using Horner's rule evaluate  $\vec{P}(t)$  at 100 points in  $[0, 1]$  and compare the values with those on an exact circle. What is the maximum error? Since the parametric value  $t$  is not proportional to angle some thought is required to decide how to measure error. Is this method faster than using trigonometric functions? Why? (See sections 4.13, 4.14, and P4-9 in Kahaner, Moler, and Nash for more information on Bézier curves.)

**CS/MATH 3414 Lab # 7**

- (2) 1. For order  $k = 4$  and knot sequence  $t = (0, 0, 0, 0, 1, 2, 3, 4, 4, 4, 6, 6, 6, 6)$ , plot the ten B-splines  $B_1(x), \dots, B_{10}(x)$ . Note the curve shapes at the multiple knots 0, 4, 6. Evaluate the B-splines using the recursive formula

$$B_{i,k}(x) = \frac{x - t_i}{t_{i+k-1} - t_i} B_{i,k-1}(x) + \frac{t_{i+k} - x}{t_{i+k} - t_{i+1}} B_{i+1,k-1}(x),$$

$$B_{j,1}(x) = \begin{cases} 1, & t_j \leq x < t_{j+1}, \\ 0, & \text{otherwise.} \end{cases}$$

- (5) 2. Using the knot sequence  $t = (0, 0, 0, 0, 1, 2, 3, 4, 4, 4, 6, 6, 6, 6)$ , construct the cubic spline  $F(x) = \sum_{i=1}^{10} \alpha_i B_{i,4}(x)$  interpolating the data  $(.5,1), (1,1), (1.5,1), (2,1), (3,1), (3.5,1), (4,1), (4.5,.75), (5,.5), (5.5,.25)$ . Plot  $F(x)$  and the data.
- (5) 3. Using the knot sequence  $\tilde{t} = (0, 0, 0, 0, 1, 2, 3, 3.5, 4, 5, 6, 6, 6, 6)$ , construct the cubic spline  $G(x) = \sum_{i=1}^{10} \beta_i B_{i,4}(x)$  interpolating the same data points as in 2. Plot  $G(x)$  and the data.
- (2) 4. Which of  $F$  or  $G$  is the better fit to the data? Explain why, in terms of the knot sequences.

## CS/MATH 3414 Lab # 8

- (5) 1. (Problem 4.12, page 156 in Shampine, Allen, and Pruess.) For turbulent flow of fluid in a smooth pipe, the equation

$$1 = \sqrt{c_f}(-0.4 + 1.74 \ln(\text{Re}\sqrt{c_f}))$$

models the relationship between the friction factor  $c_f$  and the Reynold's number  $\text{Re}$ . Compute  $c_f$  for  $\text{Re} = 10^4, 10^5, 10^6$ . Solve for all values of the Reynold's number in the same run. Do this by communicating the parameter  $\text{Re}$  to the function subprogram using a MODULE in Fortran 90. Use DFZERO (kahaner.ar) or ZEROIN (cs3410.ar) to find the friction factor  $c_f$  for the pipe.

- (5) 2. (Problem 4.20, page 156 in Shampine, Allen, and Pruess.) The following problem concerns the cooling of a sphere. Suppose the sphere is of radius  $a$  and is initially at a temperature  $V$ . It cools by Newton's law of cooling with thermal conductivity  $k$ , thalpance  $\epsilon$ , and diffusivity  $h^2$  after being suddenly placed in air at  $0^\circ\text{C}$ . It can be shown that the temperature  $\theta(r, t)$  at time  $t > 0$  and radius  $r$  is

$$\theta(r, t) = \sum_{n=1}^{\infty} \frac{A_n}{r} e^{-\gamma_n^2 h^2 t} \sin(\gamma_n r).$$

Here the  $\gamma_n$  are the (positive) roots of

$$\gamma_n \cos(\gamma_n a) - \left(\frac{1}{a} - \frac{\epsilon}{k}\right) \sin(\gamma_n a) = 0$$

and

$$A_n = \frac{2\gamma_n V}{[\gamma_n a - \cos(\gamma_n a) \sin(\gamma_n a)]} \int_0^a r \sin(\gamma_n r) dr.$$

For a steel sphere cooling in air at  $0^\circ\text{C}$ , suppose the initial temperature is  $V = 100^\circ\text{C}$  and the radius is  $a = 0.30$  meters. Appropriate physical constants are  $h^2 = 1.73 \times 10^{-5}$ ,  $\epsilon = 20$ , and  $k = 60$ . Find the three smallest values of  $\gamma_n a$  and use them to compute  $A_1$ ,  $A_2$ , and  $A_3$ . Approximate the temperature at  $r = 0.25$  for  $t = 10^i$  seconds,  $i = 2, 3, 4, 5$ .

- (5) 3. The polynomial

$$P(x) = x^{10} - 15x^9 + 95x^8 - 330x^7 + 685x^6 - 873x^5 + 685x^4 \\ - 330x^3 + 95x^2 - 15x + 1$$

has a unique zero between 2 and 3. Using any technique or combination of techniques you want (other than symbolic factorization or extended precision arithmetic), find that zero as accurately as you can. Don't forget to evaluate  $P(x)$  using Horner's rule.

- (5) 4. Find the fixed point of  $f(x) = \tan x$  closest to  $x = 100$  (radians) as accurately as you can.

## CS/MATH 3414 Lab # 9

A special case of Poisson's integral formula (R. V. Churchill, *Complex Variables and Applications*, McGraw-Hill, 1948) is

$$I(r, \theta) = 2\pi r^2 \cos 2\theta = \int_0^{2\pi} \frac{(1 - r^2) \cos 2\phi}{1 - 2r \cos(\phi - \theta) + r^2} d\phi \quad (|r| < 1, 0 \leq \theta \leq 2\pi).$$

(15) 1. Plot the function

$$f(r, \phi) = \frac{(1 - r^2) \cos 2\phi}{1 - 2r \cos(\phi - \pi) + r^2}, \quad 0 \leq \phi \leq 2\pi,$$

for each of the three cases  $r = .1, .5, .9$ . Compute  $I(r, \theta)$ ,  $\theta = \pi$ ,  $r = .1, .5, .9$ , by Simpson's rule ( $h = 2\pi/64$ ), the adaptive code SIMP ( $ACC = h^4$ ), the adaptive code QUANC8 ( $ABSERR = 0$ ,  $RELERR = h^4$ ), Mathematica, and Gaussian quadrature with 20 points. Report your results in a table like

|                     | $r = .1$       |                           | $r = .5$       |                           | $r = .9$       |                           |
|---------------------|----------------|---------------------------|----------------|---------------------------|----------------|---------------------------|
|                     | absolute error | # of function evaluations | absolute error | # of function evaluations | absolute error | # of function evaluations |
| Simpson's rule      |                |                           |                |                           |                |                           |
| SIMP                |                |                           |                |                           |                |                           |
| QUANC8              |                |                           |                |                           |                |                           |
| Gaussian quadrature |                |                           |                |                           |                |                           |
| Mathematica         |                |                           |                |                           |                |                           |

Give a half page explanation and discussion of your results.

(3) 2. Find an integration formula of the form

$$\int_0^h f(x) dx \approx \alpha_0 f(0) + \alpha_1 f'(0) + \alpha_2 f(h)$$

which is exact if  $f$  is a polynomial of degree  $\leq 2$ . [Hint: Let  $L(f) = \alpha_0 f(0) + \alpha_1 f'(0) + \alpha_2 f(h)$ . First show that  $L(af + bg) = aL(f) + bL(g)$  for constants  $a, b$  and functions  $f, g$ . Then find  $\alpha_0, \alpha_1, \alpha_2$  such that  $L(1) = \int_0^h 1 dx$ ,  $L(x) = \int_0^h x dx$ ,  $L(x^2) = \int_0^h x^2 dx$ . It now follows that  $L(a + bx + cx^2) = aL(1) + bL(x) + cL(x^2) = a \int_0^h 1 dx + b \int_0^h x dx + c \int_0^h x^2 dx = \int_0^h a + bx + cx^2 dx$ , i.e.,  $L$  is exact for polynomials of degree  $\leq 2$ .]

Mathematica wizardry: define

```

Lop[f_] := alpha0 * Derivative[0][f][0] + alpha1 * Derivative[1][f][0] +
alpha2 * Derivative[0][f][h]; Attributes[Lop] = {HoldFirst}
eqn1 = Lop[(1)&] == Integrate[1, {x, 0, h}]
eqn2 = Lop[(#1)&] == Integrate[x, {x, 0, h}]
eqn3 = Lop[((#1)^2)&] == Integrate[x^2, {x, 0, h}]
Solve[{eqn1, eqn2, eqn3}, {alpha0, alpha1, alpha2}]

```

- (5) 3. Find an explicit expression (in terms of  $h$  and  $f$ ) for the error in the integration formula of  
 2. [Hint:  $L(f) = \int_0^h$  (some interpolant to  $f$ )  $dx$ . Find that interpolant.]
- (3) 4. Compute the first four orthogonal polynomials  $\psi_0, \dots, \psi_3$  with respect to the inner product

$$\langle f, g \rangle = \int_{-1}^1 f(x)g(x) \cos(\pi x/2) dx.$$

The relevant Mathematica package is `LinearAlgebra`Orthogonalization``, containing the functions:

```
GramSchmidt[{v1, v2, ...}, InnerProduct -> pure function, Normalized -> False],
  Normalize[v, InnerProduct -> pure function],
  Projection[v1, v2, InnerProduct -> pure function],.
```

**CS/MATH 3414 Lab # 10**

- (6) 1. Compute  $w_i$  and  $x_i$  ( $i = 0, 1, 2$ ) such that the formula

$$\int_{-1}^1 f(x) \cos(\pi x/2) dx \approx \sum_{i=0}^2 w_i f(x_i)$$

is exact if  $f$  is a polynomial of degree  $\leq 5$ .

- (4) 2. Compute coefficients such that the formula

$$\int_0^\infty f(x)e^{-x} dx \approx w_0 f(0) + \tilde{w}_0 f'(0) + w_2 f(3) + \tilde{w}_2 f'(3)$$

is exact if  $f$  is a polynomial of degree  $\leq 3$ . [Hint:  $\int_0^\infty x^n e^{-x} dx = \Gamma(n+1) = n!$ ]

- (2) 3. Compute the first three orthogonal polynomials  $\Omega_0, \Omega_1, \Omega_2$  with respect to the inner product

$$\langle f, g \rangle = \int_0^1 f(x)g(x) \frac{1}{\sqrt{x}} dx.$$

- (3) 4. Express  $x^5$  as a linear combination of the Chebyshev polynomials  $T_0, T_1, \dots, T_5$ :

$$x^5 = \underline{\hspace{2cm}} T_0 + \underline{\hspace{2cm}} T_1 + \underline{\hspace{2cm}} T_2 + \underline{\hspace{2cm}} T_3 + \underline{\hspace{2cm}} T_4 + \underline{\hspace{2cm}} T_5.$$

- (5) 5. Find the polynomial  $p^*(x)$  of degree  $\leq 3$  which minimizes

$$\langle x^5 - p(x), x^5 - p(x) \rangle = \|x^5 - p(x)\|^2 = \int_{-1}^1 \frac{(x^5 - p(x))^2}{\sqrt{1-x^2}} dx$$

over  $\mathcal{P}_3 = \{\text{polynomials } p(x) \text{ of degree } \leq 3\}$ . (Hint: with respect to the above inner product  $\langle \cdot, \cdot \rangle$ , what is an orthogonal basis for  $\mathcal{P}_3$ ? Use Problem 4 to find  $p^*$  without any arithmetic at all.)

## LEAST SQUARES APPROXIMATION EXAMPLE

Problem: Find the polynomial  $p^*(x)$  of degree  $\leq 2$  which minimizes

$$\int_{-1}^1 (p(x) - \sin x)^2 dx.$$

Solution: Define  $\langle r, s \rangle = \int_{-1}^1 r(x)s(x) dx$ ,  $f(x) = \sin x$ . Then the problem is equivalent to

$$\min_p \int_{-1}^1 (p(x) - \sin x)^2 dx = \min_p \langle p - f, p - f \rangle = \min_p \|p - f\|^2 \quad (1)$$

or

$$\min_p \|p - f\| \quad (2)$$

where the minimization is done over all polynomials  $p$  of degree  $\leq 2$ . The solution to (2) is

$$p^* = \sum_{i=0}^2 \frac{\langle f, \phi_i \rangle}{\langle \phi_i, \phi_i \rangle} \phi_i \quad (3)$$

where  $\phi_0, \phi_1, \phi_2$  are orthogonal polynomials with respect to the inner product  $\langle \cdot, \cdot \rangle$ . Now, working out the details,  $\phi_0 = 1$ ,  $\phi_1 = x$ ,  $\phi_2 = (3/2)(x^2 - 1/3)$ ,  $\langle \phi_0, \phi_0 \rangle = 2$ ,  $\langle \phi_1, \phi_1 \rangle = 2/3$ ,  $\langle \phi_2, \phi_2 \rangle = 2/5$ ,  $\langle f, \phi_0 \rangle = 0$ ,  $\langle f, \phi_1 \rangle = 2(\sin 1 - \cos 1)$ ,  $\langle f, \phi_2 \rangle = 0$ . Thus

$$p^*(x) = 0 \cdot 1 + \frac{2(\sin 1 - \cos 1)}{2/3} x + 0 \cdot \frac{3}{2}(x^2 - 1/3) = 3(\sin 1 - \cos 1)x.$$

## CS/MATH 3414 Lab # 11

The purpose of this lab is to study least squares approximation, both discrete and continuous, using the statistical error function

$$\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$$

(`Erf[x]` in Mathematica). Generate 11 data points by taking  $t_i = (i - 1)/10$  and  $b_i = \operatorname{erf}(t_i)$ ,  $i = 1, \dots, 11$ . Make a table showing all the results from the following four parts.

- (5) 1. Fit the data in a least-squares sense with polynomials of degrees from 1 to 10. Compare the fitted polynomial with  $\operatorname{erf}(t)$  for 100 values of  $t$  between the data points, and see how the maximum (absolute) error depends on  $n$ , the number of coefficients in the polynomial.
- (5) 2. Since  $\operatorname{erf}(t)$  is an odd function of  $t$ , that is,  $\operatorname{erf}(t) = -\operatorname{erf}(-t)$ , it is reasonable to fit the same data by a linear combination of odd powers of  $t$ ,

$$\operatorname{erf}(t) \approx c_1 t + c_2 t^3 + \dots + c_n t^{2n-1}.$$

Again, see how the error between data points depends on  $n$ . Since  $t$  varies over  $[0, 1]$  in this problem, it is not necessary to consider using other basis polynomials.

- (5) 3. Polynomials are not particularly good approximants for  $\operatorname{erf}(t)$  because they are unbounded for large  $t$ , whereas  $\operatorname{erf}(t)$  approaches 1 for large  $t$ . So, using the same data points, fit a model of the form

$$\operatorname{erf}(t) \approx c_1 + e^{-t^2} (c_2 + c_3 z + c_4 z^2 + c_5 z^3)$$

where  $z = 1/(1 + t)$ . How does the error between data points compare with the polynomial models?

- (5) 4. Fit  $\operatorname{erf}(t)$ ,  $0 \leq t \leq 1$ , with a model of the form

$$\operatorname{erf}(t) \approx \sum_{i=0}^n c_i T_i(2t - 1)$$

where the  $T_i$  are Chebyshev polynomials and  $\sum c_i T_i$  is the best continuous least squares approximation with respect to the inner product

$$\langle f, g \rangle = \int_{-1}^1 \frac{f(x)g(x)}{\sqrt{1-x^2}} dx.$$

(Translate  $\operatorname{erf}(t)$ ,  $0 \leq t \leq 1$ , to the interval  $[-1, 1]$ ,  $\operatorname{erf}((x + 1)/2)$ ,  $-1 \leq x \leq 1$ , so that the Chebyshev polynomials apply.) Compare the error here to that of the previous models. To get `NIntegrate` to work with an improper integral, make the change of variable  $x = \cos \theta$ , and use the option `PrecisionGoal`  $\rightarrow$  `9` to speed up the integration.

The relevant Mathematica functions are `Fit`, `Projection`, `NIntegrate`, and `Normalize`.

## CS/MATH 3414 Lab # 12

- (12) 1. (Problem P7-2, page 260 in Kahaner, Moler, and Nash.) The purpose of this problem is to remind you that sometimes explicit solutions are not as useful as numerical methods. Kepler's equation in orbit determination is

$$M = E - e \sin E.$$

The symbols  $M$ ,  $E$ , and  $e$  are referred to as the Mean anomaly, Eccentric anomaly, and eccentricity of the orbit, respectively.

- (a) If  $M = 24.851090$  and  $e = 0.1$  use a root finding algorithm to find  $E$ .  
 (b) A formula for  $E$  is

$$E = M + 2 \sum_{m=1}^{\infty} \frac{1}{m} J_m(me) \sin(mM),$$

where  $J_m(x)$  is the Bessel function of the first kind of order  $m$ . Use this formula to find  $E$  for the values in (a).

- (c) Find  $E$  from the formula in (b) using

$$J_m(me) = \sum_{n=0}^{\infty} \frac{(-1)^n (me/2)^{2n+m}}{n!(m+n)!}.$$

In (b) and (c) compare the effort and accuracy of your results with those in (a).

Suggestion: use the root finding routine ZEROIN (cs3410 archive) and DBESJ (kahaner archive) for the Bessel functions. The entire exercise can also be done in Mathematica.

SUBROUTINE DBESJ(X,ALPHA,N,Y,NZ)

C ABSTRACT

C BESJ COMPUTES AN N MEMBER SEQUENCE OF J BESSEL FUNCTIONS  
 C J/SUB(ALPHA+K-1)/(X), K=1,...,N FOR NON-NEGATIVE ALPHA AND X.  
 C A COMBINATION OF THE POWER SERIES, THE ASYMPTOTIC EXPANSION  
 C FOR X TO INFINITY AND THE UNIFORM ASYMPTOTIC EXPANSION FOR  
 C NU TO INFINITY ARE APPLIED OVER SUBDIVISIONS OF THE (NU,X)  
 C PLANE. FOR VALUES OF (NU,X) NOT COVERED BY ONE OF THESE  
 C FORMULAE, THE ORDER IS INCREMENTED OR DECREMENTED BY INTEGER  
 C VALUES INTO A REGION WHERE ONE OF THE FORMULAE APPLY. BACKWARD  
 C RECURSION IS APPLIED TO REDUCE ORDERS BY INTEGER VALUES EXCEPT  
 C WHERE THE ENTIRE SEQUENCE LIES IN THE OSCILLATORY REGION. IN  
 C THIS CASE FORWARD RECURSION IS STABLE AND VALUES FROM THE  
 C ASYMPTOTIC EXPANSION FOR X TO INFINITY START THE RECURSION  
 C WHEN IT IS EFFICIENT TO DO SO. LEADING TERMS OF THE SERIES  
 C AND UNIFORM EXPANSION ARE TESTED FOR UNDERFLOW. IF A SEQUENCE  
 C IS REQUESTED AND THE LAST MEMBER WOULD UNDERFLOW, THE RESULT  
 C IS SET TO ZERO AND THE NEXT LOWER ORDER TRIED, ETC., UNTIL A  
 C MEMBER COMES ON SCALE OR ALL MEMBERS ARE SET TO ZERO.  
 C OVERFLOW CANNOT OCCUR.

```

C
C      INPUT
C      X      - X .GE. 0.0E0
C      ALPHA  - ORDER OF FIRST MEMBER OF THE SEQUENCE,
C              ALPHA .GE. 0.0E0
C      N      - NUMBER OF MEMBERS IN THE SEQUENCE, N .GE. 1
C
C      OUTPUT
C      Y      - A VECTOR WHOSE FIRST N COMPONENTS CONTAIN
C              VALUES FOR J/SUB(ALPHA+K-1)/(X), K=1,...,N
C      NZ     - NUMBER OF COMPONENTS OF Y SET TO ZERO DUE TO
C              UNDERFLOW,
C              NZ=0   , NORMAL RETURN, COMPUTATION COMPLETED
C              NZ .NE. 0, LAST NZ COMPONENTS OF Y SET TO ZERO,
C                    Y(K)=0.0E0, K=N-NZ+1,...,N.

```

### CS/MATH 3414 Lab # 13

- (10) 1. A second order accurate finite difference approximation to the solution  $u(x)$  of the linear two-point boundary value problem

$$\begin{aligned} u'' - 7\pi \cos(7\pi x)u' + (7\pi)^2 \sin(7\pi x)u &= -(7\pi)^2 \sin(7\pi x), & 0 \leq x \leq 1, \\ u(0) = u(1) &= 0, \end{aligned}$$

is given by the solution  $U$  to the linear system  $AU = D$ :

$$\begin{aligned} [-7\pi \cos(7\pi x_i)h/2 - 1]U_{i-1} + [2 - (7\pi h)^2 \sin(7\pi x_i)]U_i \\ + [7\pi \cos(7\pi x_i)h/2 - 1]U_{i+1} = (7\pi h)^2 \sin(7\pi x_i), \quad i = 1, \dots, n, \end{aligned}$$

where

$$h = \frac{1}{n+1}, \quad x_i = ih, \quad U_0 = U_{n+1} = 0, \quad u_i = u(x_i), \quad U_i = u_i + \mathcal{O}(h^2).$$

Solve  $AU = D$  for  $n+1 = 25, 50, 100, 200, 400, 800$ , and plot the solution vectors  $U$ . The exact solution satisfies  $u(.5) = 1/e - 1$ . Comment (one well written paragraph) on these plots and what you believe is the accuracy of your computed solutions.

- (10) 2. (**Extrapolated finite differences.**) Consider again the two-point boundary value problem above. An analysis of the finite difference approximation  $U$  shows that

$$u_i = U_i + a_2 h^2 + a_4 h^4 + a_6 h^6 + \dots,$$

where the coefficients  $a_i$  are independent of  $h$ . With  $n+1 = 50, 100, 200, 400$ , solve  $AU = D$  and estimate  $u(.5)$  using extrapolation to the limit. Compare the four finite difference approximations, the extrapolated value  $\hat{u}(.5)$ , and the exact value  $u(.5)$ . As before, use DGTSV (LAPACK) to factor and solve  $AU = D$ .

- (10) 3. (**Shooting.**) Consider the initial value problem (IVP)

$$\begin{aligned} u'' - 7\pi \cos(7\pi x)u' + (7\pi)^2 \sin(7\pi x)u &= -(7\pi)^2 \sin(7\pi x), & 0 \leq x \leq 1, \\ u(0) = 0, \quad u'(0) &= \alpha. \end{aligned}$$

Denote the solution by  $u(x; \alpha)$ . Observe that the two-point boundary value problem is equivalent to finding a value of  $\alpha$  such that

$$f(\alpha) = u(1; \alpha) = 0.$$

$f(\alpha)$  is a highly nonlinear function defined only implicitly by solving an IVP. Using root finding and IVP software, find  $\alpha$  such that  $|f(\alpha)| \leq 10^{-10}$ . Now compute  $u(.5)$  with this  $\alpha$  and compare it with the values obtained in problem 1. Compare the time (human and computer) required by extrapolated finite differences and shooting.

Use GAMS to retrieve one of the ODE IVP software modules in search class I1a1a or I1a1b. You will also need to use a root solving software package such as ROOT (HOMPACK) or ZEROIN (CS3410). A brief description of GAMS is included with this homework assignment.

GAMS existed before the World Wide Web, and was available to run in both X-windows and non-X-windows environments. For historical interest, here is a description of how the preWWW X-windows version of GAMS operated.

To retrieve software modules, use search to select the desired problem class. Before browsing through the modules, set the search filters so that access is free and precision is double. This is accomplished by issuing the following gams commands:

```
filter access free
filter precision double
```

The filters need only be set once while gams is active. Browse through the software modules until you reach a module you wish to retrieve. The commands

```
get documentation
get fullsource
```

retrieve the software module's documentation and software files to your home directory. If you are using xgams, substitute the xgams versions of these commands.

GAMS is reached through the World Wide Web at

`http://gams.nist.gov`.

## The GAMS Virtual Software Repository

The Guide to Available Mathematical Software (GAMS) project of the National Institute of Standards and Technology (NIST) studies techniques to provide scientists and engineers with improved access to reusable computer software which is available to them for use in mathematical modeling and statistical analysis. One of the products of this work is an on-line cross-index of available mathematical software. This system also operates as a virtual software repository. That is, it provides centralized access to such items as abstracts, documentation, and source of software modules that it catalogs; however, rather than operate a physical repository of its own, GAMS provides transparent access to multiple repositories operated by others.

Currently GAMS indexes four software repositories: NIST Cray (GRANTA), NIST Convex (TIBER), NIST Suns (CAMSUN), and NETLIB, a public-domain software collection of Oak Ridge National Laboratory and AT&T Bell Labs. Among the packages cataloged in GAMS are : the IMSL, NAG, PORT, and SLATEC libraries; the BLAS, EISPACK, FISHPAK, FNLIB, FFTPACK, LAPACK, LINPACK, and STARPACK packages; the DATAPLOT and SAS statistical analysis systems; as well as other collections such as the Collected Algorithms of the ACM. Note that although GAMS catalogs both public-domain and proprietary software, source code of proprietary software products are not available through GAMS, although related items such as documentation and example programs often are.

All problem-solving software modules in GAMS are assigned one or more problem classifications from a tree-structured taxonomy of mathematical and statistical problems. Users can browse through modules in any given problem class. To find an appropriate class, one can utilize the taxonomy as a decision tree, or enter keywords which are then mapped to problem classes. Search filters can be declared which allow users to specify preferences such computing precision, programming language and access type (i.e., free or proprietary). In addition, users can browse through all modules in a given package, or all modules with a given name.

Complete documentation on using the GAMS system is available within the system itself. Please report any problems to [gams@cam.nist.gov](mailto:gams@cam.nist.gov). We are always interested in hearing suggestions for improving the system.

Computing and Applied Mathematics Laboratory  
National Institute of Standards and Technology  
Gaithersburg, MD 20899 USA

Disclaimers : (1) GAMS catalogs mathematical and statistical software that has been made available for use by NIST staff. Identification of commercial products in GAMS does not imply recommendation or endorsement by NIST. (2) Not all software available at the repositories indexed by GAMS is retrievable using GAMS.