

1. [Kincaid and Cheney Exercise 1.3#11a] Give bases consisting of real sequences for the solution space of  $(4I - 3E^2 + E^3)x = 0$ .

This problem maybe reformulated as finding a basis for the solutions to

$$p(E)x = 0 \quad \text{where} \quad p(\lambda) = 4 - 3\lambda^2 + \lambda^3.$$

Factor  $p$  as  $p(\lambda) = (\lambda + 1)(\lambda - 2)^2$ . It follows that  $\lambda_1 = -1$  is a root with multiplicity 1 and  $\lambda_2 = 2$  is a root with multiplicity 2. In this case, the solutions  $u_1$  and  $u_2$  defined by

$$u_1 = (\lambda_1, \lambda_1^2, \lambda_1^3, \dots) \quad \text{and} \quad u_2 = (\lambda_2, \lambda_2^2, \lambda_2^3, \dots)$$

do not span the space of all solutions. However, a third linearly independent solution may be obtained by setting  $x_\lambda = (\lambda, \lambda^2, \lambda^3, \dots)$  and differentiating with respect to  $\lambda$  to obtain

$$u_3 = \left. \frac{dx_\lambda}{d\lambda} \right|_{\lambda=\lambda_2} = (1, 2\lambda_2, 3\lambda_2^2, \dots).$$

The fact that  $u_3$  is linearly independent from  $u_1$  and  $u_2$  is automatic but also may be seen by taking the determinant of the matrix

$$M = \begin{bmatrix} \lambda_1 & \lambda_2 & 1 \\ \lambda_1^2 & \lambda_2^2 & 2\lambda_2 \\ \lambda_1^3 & \lambda_2^3 & 3\lambda_2^2 \end{bmatrix} = \begin{bmatrix} -1 & 2 & 1 \\ 1 & 4 & 4 \\ -1 & 6 & 12 \end{bmatrix} \quad \text{as} \quad \det(M) = -46.$$

That  $u_3$  is a solution follows from

$$\begin{aligned} p(E)u_3 &= p(E) \left. \frac{dx_\lambda}{d\lambda} \right|_{\lambda=\lambda_2} = \left. \frac{d}{d\lambda} (p(E)x_\lambda) \right|_{\lambda=\lambda_2} = \left. \frac{d}{d\lambda} (p(\lambda)x_\lambda) \right|_{\lambda=\lambda_2} \\ &= \left( p(\lambda) \frac{dx_\lambda}{d\lambda} + p'(\lambda)x_\lambda \right) \Big|_{\lambda=\lambda_2} = p(\lambda_2)u_3 + p'(\lambda_2)u_2 = 0 \end{aligned}$$

since both  $p(\lambda_2) = 0$  and  $p'(\lambda_2) = 0$  on account of  $\lambda_2$  being a root of multiplicity 2. It follows that  $\{u_1, u_2, u_3\}$  is a basis for the solution space.

2. [Kincaid and Cheney Exercise 1.3#11c] Give bases consisting of real sequences for the solution space of  $(2E^6 - 9E^5 + 12E^4 - 4E^3)x = 0$ .

This problem maybe reformulated as finding a basis for the solutions to

$$p(E)x = 0 \quad \text{where} \quad p(\lambda) = 2\lambda^6 - 9\lambda^5 + 12\lambda^4 - 4\lambda^3.$$

Factor  $p$  as  $p(\lambda) = \lambda^3(2\lambda - 1)(\lambda - 2)^2$ . It follows that  $\lambda_1 = 0$  is a root with multiplicity 3,  $\lambda_2 = 1/2$  is a root with multiplicity 1 and  $\lambda_3 = 2$  is a root with multiplicity 2. Define

$$x_\lambda = (1, \lambda, \lambda^2, \lambda^3, \lambda^4, \dots)$$

and set

$$\begin{aligned} u_1 &= x_\lambda \Big|_{\lambda=\lambda_1} = (1, \lambda_1, \lambda_1^2, \lambda_1^3, \lambda_1^4, \lambda_1^5, \dots) \\ u_2 &= \frac{dx_\lambda}{d\lambda} \Big|_{\lambda=\lambda_1} = (0, 1, 2\lambda_1, 3\lambda_1^2, 4\lambda_1^3, 5\lambda_1^4, \dots) \\ u_3 &= \frac{d^2x_\lambda}{d\lambda^2} \Big|_{\lambda=\lambda_1} = (0, 0, 2, (3)(2)\lambda_1, (4)(3)\lambda_1^2, (5)(4)\lambda_1^3, \dots) \\ u_4 &= x_\lambda \Big|_{\lambda=\lambda_2} = (1, \lambda_2, \lambda_2^2, \lambda_2^3, \lambda_2^4, \lambda_2^5, \dots) \\ u_5 &= x_\lambda \Big|_{\lambda=\lambda_3} = (1, \lambda_3, \lambda_3^2, \lambda_3^3, \lambda_3^4, \lambda_3^5, \dots) \\ u_6 &= \frac{dx_\lambda}{d\lambda} \Big|_{\lambda=\lambda_3} = (0, 1, 2\lambda_2, 3\lambda_2^2, 4\lambda_2^3, 5\lambda_2^4, \dots). \end{aligned}$$

The fact that the vectors  $\{u_1, \dots, u_6\}$  are independent is automatic but also may be seen by taking the determinant of the matrix

$$M = \begin{bmatrix} u_{1,1} & \cdots & u_{6,1} \\ \vdots & \ddots & \vdots \\ u_{1,6} & \cdots & u_{6,6} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1/2 & 2 & 1 \\ 0 & 0 & 2 & 1/4 & 4 & 4 \\ 0 & 0 & 0 & 1/8 & 8 & 12 \\ 0 & 0 & 0 & 1/16 & 16 & 32 \\ 0 & 0 & 0 & 1/32 & 32 & 80 \end{bmatrix} \quad \text{as} \quad \det(M) = 36.$$

Since  $p(\lambda_1) = 0$ ,  $p'(\lambda_1) = 0$ ,  $p''(\lambda_1) = 0$ ,  $p(\lambda_2) = 0$ ,  $p(\lambda_3) = 0$  and  $p'(\lambda_3) = 0$ , then

$$\begin{aligned} p(E)u_1 &= p(\lambda_1)u_1 = 0 \\ p(E)u_2 &= p(\lambda_1)u_2 + p'(\lambda_1)u_1 = 0 \\ p(E)u_3 &= p(\lambda_1)u_3 + 2p'(\lambda_1)u_2 + p''(\lambda_1)u_1 = 0 \\ p(E)u_4 &= p(\lambda_2)u_4 = 0 \\ p(E)u_5 &= p(\lambda_3)u_5 = 0 \\ p(E)u_6 &= p(\lambda_3)u_6 + p'(\lambda_3)u_5 = 0 \end{aligned}$$

and therefore  $\{u_1, \dots, u_6\}$  is a basis for the solution space.

3. [Kincaid and Cheney Exercise 1.3#14] Define an operator  $\Delta$  by putting

$$\Delta x = (x_2 - x_1, x_3 - x_2, x_4 - x_3, \dots).$$

Show that  $E = I + \Delta$ . Show that if  $p$  is a polynomial, then

$$p(E) = p(I) + p'(I)\Delta + \frac{1}{2}p''(I)\Delta^2 + \frac{1}{3!}p'''(I)\Delta^3 + \dots + \frac{1}{m!}p^{(m)}(I)\Delta^m.$$

Since

$$\begin{aligned} (I + \Delta)x &= x + \Delta x = (x_1, x_2, x_3, \dots) + (x_2 - x_1, x_3 - x_2, x_4 - x_3, \dots) \\ &= (x_2, x_3, x_4, \dots) = Ex. \end{aligned}$$

for every  $x = (x_1, x_2, x_3, \dots)$ , then  $E = I + \Delta$ . To show the equality in the second part of this homework is valid we prove first the following lemma:

**Lemma 1.** Let  $\pi_n: \mathbf{R}^\infty \rightarrow \mathbf{R}^\infty$  be the projection onto the first  $n$  coordinates defined by

$$\pi_n(x_1, x_2, x_3, \dots) = (x_1, x_2, \dots, x_n, 0, 0, \dots).$$

Suppose  $K: \mathbf{R}^\infty \rightarrow \mathbf{R}^\infty$  and  $L: \mathbf{R}^\infty \rightarrow \mathbf{R}^\infty$  are linear operators such that  $Kx_\lambda = Lx_\lambda$  for every  $\lambda \in \mathbf{N}$  where  $x_\lambda = (\lambda, \lambda^2, \lambda^3, \dots)$ . If there exists  $r \in \mathbf{N}$  such that  $\pi_n Kx = \pi_n K\pi_{n+r}x$  and  $\pi_n Lx = \pi_n L\pi_{n+r}x$  hold for every  $x \in \mathbf{R}^\infty$  and  $n \in \mathbf{N}$ , then  $K = L$ .

**Proof:** Define  $u_i = \pi_{n+r}x_{\lambda_i}$  where  $\lambda_i = i$  for  $i = 1, 2, \dots, n+r$ . The matrix

$$M = \begin{bmatrix} u_{1,1} & \cdots & u_{n+r,1} \\ \vdots & \ddots & \vdots \\ u_{1,n+r} & \cdots & u_{n+r,n+r} \end{bmatrix} = \begin{bmatrix} 1 & 2 & \cdots & n+r \\ 1 & 2^2 & \cdots & (n+r)^2 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 2^{n+r} & \cdots & (n+r)^{n+r} \end{bmatrix} \in \mathbf{R}^{(n+r) \times (n+r)}$$

is a Vandermonde matrix and therefore non-singular. Now, given any  $x \in \mathbf{R}^\infty$  there is a unique  $\beta \in \mathbf{R}^{n+r}$  such that  $\pi_{n+r}x = \beta_1 u_1 + \beta_2 u_2 + \dots + \beta_{n+r} u_{n+r}$ . Let  $z = \beta_1 x_{\lambda_1} + \beta_2 x_{\lambda_2} + \dots + \beta_{n+r} x_{\lambda_{n+r}}$ . Then  $\pi_{n+r}x = \pi_{n+r}z$ . By hypothesis

$$\pi_n K u_i = \pi_n K \pi_{n+r} x_{\lambda_i} = \pi_n K x_{\lambda_i} = \pi_n L x_{\lambda_i} = \pi_n L \pi_{n+r} x_{\lambda_i} = \pi_n L u_i$$

for  $i = 1, 2, \dots, n+r$  and therefore

$$\begin{aligned} \pi_n K x &= \pi_n K \pi_{n+r} x = \pi_n K \pi_{n+r} z = \beta_1 K u_1 + \beta_2 K u_2 + \dots + \beta_{n+r} K u_{n+r} \\ &= \beta_1 L u_1 + \beta_2 L u_2 + \dots + \beta_{n+r} L u_{n+r} = \pi_n L \pi_{n+r} z = \pi_n L \pi_{n+r} x = \pi_n L x. \end{aligned}$$

Since  $\pi_n K x = \pi_n L x$  for any  $x \in \mathbf{R}^\infty$ , then  $\pi_n K = \pi_n L$ . As this holds for every  $n \in \mathbf{N}$ , we conclude that  $K = L$ . This finishes the proof of Lemma 1.

Claim  $\pi_n \Delta^r x = \pi_n \Delta^r \pi_{n+r} x$  for every  $x \in \mathbf{R}^\infty$ ,  $n \in \mathbf{N}$  and  $r \in \mathbf{N}$ . We shall prove this claim by induction. For the base case notice that

$$\pi_n \Delta x = (x_2 - x_1, \dots, x_{n+1} - x_n, 0, 0, \dots) = \pi_n \Delta \pi_{n+1} x \quad \text{for every } n \in \mathbf{N}.$$

Now suppose  $\pi_n \Delta^r x = \pi_n \Delta^r \pi_{n+r} x$  for every  $x \in \mathbf{R}^\infty$  and all  $n \in \mathbf{N}$ . Then

$$\begin{aligned} \pi_n \Delta^{r+1} x &= \pi_n \Delta(\Delta^r x) = \pi_n \Delta \pi_{n+1}(\Delta^r x) = \pi_n \Delta(\pi_{n+1} \Delta^r x) = \pi_n \Delta(\pi_{n+1} \Delta^r \pi_{n+r+1} x) \\ &= \pi_n \Delta \pi_{n+1}(\Delta^r \pi_{n+r+1} x) = \pi_n \Delta(\Delta^r \pi_{n+r+1} x) = \pi_n \Delta^{r+1} \pi_{n+r+1} x. \end{aligned}$$

This completes the induction and proves the claim. Similarly  $\pi_n E^r x = \pi_n E^r \pi_{n+r} x$ .

Now let

$$K = p(E) = \sum_{i=0}^m c_i E^i$$

and

$$L = p(I) + p'(I)\Delta + \frac{1}{2}p''(I)\Delta^2 + \frac{1}{3!}p'''(I)\Delta^3 + \dots + \frac{1}{m!}p^{(m)}(I)\Delta^m.$$

From what we just proved, it follows that

$$\pi_n Kx = \pi_n K \pi_{n+m} x \quad \text{and} \quad \pi_n Lx = \pi_n L \pi_{n+m} x.$$

We finish by noting that

$$Ex_\lambda = \lambda x_\lambda \quad \text{and} \quad \Delta x_\lambda = (E - I)x_\lambda = \lambda x_\lambda - x_\lambda = (\lambda - 1)x_\lambda$$

along with Taylor's theorem

$$p(\lambda) = \sum_{i=0}^m c_i \lambda^i = p(1) + p'(1)(\lambda - 1) + \frac{1}{2!}p''(1)(\lambda - 1)^2 + \dots + \frac{1}{m!}p^{(m)}(1)(\lambda - 1)^m$$

yields

$$\begin{aligned} Lx_\lambda &= p(1)x_\lambda + p'(1)(\lambda - 1)x_\lambda + \frac{1}{2!}p''(1)(\lambda - 1)^2 x_\lambda + \dots + \frac{1}{m!}p^{(m)}(1)(\lambda - 1)^m x_\lambda \\ &= \left( p(1) + p'(1)(\lambda - 1) + \frac{1}{2!}p''(1)(\lambda - 1)^2 + \dots + \frac{1}{m!}p^{(m)}(1)(\lambda - 1)^m \right) x_\lambda \\ &= p(\lambda)x_\lambda = p(E)x_\lambda = Kx_\lambda. \end{aligned}$$

Therefore, the hypothesis of Lemma 1 are satisfied with  $r = m$ . The result  $K = L$  follows.

4. [Kincaid and Cheney Exercise 1.3#15] Prove that if  $x = (\lambda, \lambda^2, \lambda^3, \dots)$  and  $p$  is a polynomial, then  $p(\Delta)x = p(\lambda - 1)x$ . Describe how to solve a difference equation written in the form  $p(\Delta)x = 0$ .

Let  $\mathcal{P}_n$  be the set of all polynomials of degree  $n$  and  $\mathcal{P}$  be the set of all polynomials. We use induction to prove that

$$p(\Delta)x_\lambda = p(\lambda - 1)x_\lambda \quad \text{for all } p \in \mathcal{P}.$$

**Base case:** Let  $p_0 \in \mathcal{P}_0$ . Then  $p_0(\eta) = c$  for some  $c \in \mathbf{C}$ . Therefore

$$p_0(\Delta)x_\lambda = cx_\lambda = p_0(\lambda - 1)x_\lambda.$$

**Induction step:** For induction suppose  $p_n(\Delta)x_\lambda = p_n(\lambda - 1)x_\lambda$  for every  $p_n \in \mathcal{P}_n$ . Let  $p_{n+1} \in \mathcal{P}_{n+1}$ . Write  $p_{n+1}(\eta) = \eta p_n(\eta) + c$  for some  $p_n \in \mathcal{P}_n$  and  $c \in \mathbf{C}$ . To complete the induction we need to show  $p_{n+1}(\Delta)x_\lambda = p_{n+1}(\lambda - 1)x_\lambda$ . By the induction hypothesis we have that  $p_n(\Delta)x_\lambda = p_n(\lambda - 1)x_\lambda$ . Therefore

$$\begin{aligned} p_{n+1}(\Delta)x_\lambda &= (\Delta p_n(\Delta) + cI)x_\lambda = \Delta p_n(\Delta)x_\lambda + cx_\lambda = \Delta(p_n(\lambda - 1)x_\lambda) + cx_\lambda \\ &= (p_n(\lambda - 1))\Delta x_\lambda + cx_\lambda = (p_n(\lambda - 1))(E - I)x_\lambda + cx_\lambda \\ &= (p_n(\lambda - 1))(\lambda - 1)x_\lambda + cx_\lambda = ((\lambda - 1)p_n(\lambda - 1) + c)x_\lambda \\ &= p_{n+1}(\lambda - 1)x_\lambda. \end{aligned}$$

This completes the induction.

To solve the difference equation  $p(\Delta)x = 0$  define  $q(\lambda) = p(\lambda - 1)$ . By Taylor's theorem

$$q(\lambda) = \sum_{i=0}^m \frac{q^{(i)}(1)}{i!} (\lambda - 1)^i.$$

where  $m$  is the degree of the polynomial  $q$ . Therefore

$$p(\eta) = \sum_{i=0}^m \frac{q^{(i)}(1)}{i!} \eta^i.$$

Applying Kincaid and Cheney Exercise 1.3#14 to the polynomial  $q$  we obtain that

$$q(E) = \sum_{i=0}^m \frac{q^{(i)}(I)}{i!} \Delta^i.$$

Since  $q^{(i)}(I) = q^{(i)}(1)I$ , then

$$q(E) = \sum_{i=0}^m \frac{q^{(i)}(1)}{i!} \Delta^i = p(\Delta).$$

Now solve  $p(\Delta)x = 0$  by solving  $q(E)x = 0$  using the method of Kincaid and Cheney chapter 1.3 pages 31–32.

5. [Kincaid and Cheney Exercise 1.3#16] Show that

$$\Delta^n = (-1)^n \left( E^0 - nE + \frac{1}{2}n(n-1)E^2 - \frac{1}{3!}n(n-1)(n-2)E^3 + \cdots + (-1)^n E^n \right).$$

Define  $p(\eta) = \eta^n$  and  $q(\lambda) = p(\lambda - 1)$ . By the binomial theorem

$$\begin{aligned} q(\lambda) &= (\lambda - 1)^n = \sum_{i=0}^n \binom{n}{i} \lambda^i (-1)^{n-i} = (-1)^n \sum_{i=0}^n \binom{n}{i} \lambda^i (-1)^i \\ &= (-1)^n \left( \lambda^0 - n\lambda + \frac{1}{2}n(n-1)\lambda^2 - \frac{1}{3!}n(n-1)(n-2)\lambda^3 + \cdots + (-1)^n \lambda^n \right). \end{aligned}$$

From Kincaid and Cheney Exercise 1.3#15 we have that  $p(\Delta) = q(E)$ . Therefore

$$\begin{aligned} \Delta^n &= p(\Delta) = q(E) \\ &= \left( E^0 - nE + \frac{1}{2}n(n-1)E^2 - \frac{1}{3!}n(n-1)(n-2)E^3 + \cdots + (-1)^n E^n \right). \end{aligned}$$

as desired.

6. [Kincaid and Cheney Computer Exercise 1.3#4abc] Compute the particular solution to the recurrence relation  $x_n = 2(x_{n-1} + x_{n-2})$  in Problem 1.3.27 for  $n = 1, 2, \dots, 100$  in the following three ways and compare the results.
- $x_n$  directly from the recurrence relation.
  - $y_n = \beta(1 - \sqrt{3})^n$
  - $z_n = \alpha(1 + \sqrt{3})^n + \beta(1 - \sqrt{3})^n$  where  $\alpha$  is the computer's unit roundoff error.

We will be using a computer which implements the IEEE 754 standard for double precision floating point arithmetic. For this hardware the mantissa consists of 52 binary digits with an implicit leading bit for 53 binary digits of precision. Equation (6) on page 45 of Kincaid and Cheney defines the unit roundoff error by

$$\alpha = \max \{ |\delta| : \text{fl}(x) = x(1 + \delta) \}.$$

In the case of IEEE 754 double precision  $\alpha = 2^{-54}$ . The following program performs the desired calculations.

```

1 /*
2   Kincaid and Cheney Computer Exercise 1.3#4abc
3   Written October 15, 2012 by for Math 701 Eric Olson
4 */
5
6 #include <stdio.h>
7 #include <math.h>
8
9 #define N 100
10 double x[N],y[N],z[N];
11
12 main(){
13     double lambda1=1.0+sqrt(3.0);
14     double lambda2=1.0-sqrt(3.0);
15     double alpha=pow(2.0,-54.0);
16     double beta=1/lambda2;
17     printf(
18         "Kincaid and Cheney Computer Exercise 1.3#4abc\n"
19         "Written October 15, 2012 by for Math 701 Eric Olson\n");
20     x[0]=y[0]=z[0]=1.0;
21     x[1]=y[1]=z[1]=lambda2;
22     int n;
23     for(n=2;n<N;n++){
24         x[n]=2.0*(x[n-1]+x[n-2]);
25         y[n]=beta*pow(lambda2,(double)(n+1));
26         z[n]=alpha*pow(lambda1,(double)(n+1))+y[n];
27     }
28     printf("\n%4s %24s %24s %24s\n",
29         "n", "x[n]", "y[n]", "z[n]");
30     for(n=0;n<N;n++){
31         printf("%4d %24.15e %24.15e %24.15e\n",
32             n+1,x[n],y[n],z[n]);
33     }
34     return 0;
35 }
```

The output of this program is

Kincaid and Cheney Computer Exercise 1.3#4abc  
Written October 15, 2012 by for Math 701 Eric Olson

n	x[n]	y[n]	z[n]
1	1.000000000000000e+00	1.000000000000000e+00	1.000000000000000e+00
2	-7.320508075688772e-01	-7.320508075688772e-01	-7.320508075688772e-01
3	5.358983848622456e-01	5.358983848622453e-01	5.358983848622464e-01
4	-3.923048454132632e-01	-3.923048454132636e-01	-3.923048454132605e-01
5	2.871870788979649e-01	2.871870788979631e-01	2.871870788979716e-01
6	-2.102355330305965e-01	-2.102355330306008e-01	-2.102355330305777e-01
7	1.539030917347368e-01	1.539030917347247e-01	1.539030917347877e-01
8	-1.126648825917194e-01	-1.126648825917522e-01	-1.126648825915798e-01
9	8.247641828603491e-02	8.247641828594490e-02	8.247641828641564e-02
10	-6.037692861136890e-02	-6.037692861161448e-02	-6.037692861032840e-02
11	4.419897934933203e-02	4.419897934866083e-02	4.419897935217446e-02
12	-3.235589852407372e-02	-3.235589852590728e-02	-3.235589851630785e-02
13	2.368616165051662e-02	2.368616164550707e-02	2.368616167173320e-02
14	-1.733947374711420e-02	-1.733947376080042e-02	-1.733947368914929e-02
15	1.269337580680485e-02	1.269337576941330e-02	1.269337596516782e-02
16	-9.292195880618692e-03	-9.292195982774224e-03	-9.292195447962918e-03
17	6.802359852372319e-03	6.802359573278148e-03	6.802361034409808e-03
18	-4.979672056492745e-03	-4.979672818992151e-03	-4.979668827106218e-03
19	3.645375591759148e-03	3.645373508571991e-03	3.645384414607177e-03
20	-2.668592929467195e-03	-2.668598620840318e-03	-2.668568824998080e-03
21	1.953565324583906e-03	1.953549775463346e-03	1.953631179218194e-03
22	-1.430055209766579e-03	-1.430097690753941e-03	-1.429875291559770e-03
23	1.047020229634654e-03	1.046904169418809e-03	1.047511775316849e-03
24	-7.660699602638488e-04	-7.663870426702639e-04	-7.647270324858406e-04
25	5.619005387416109e-04	5.610342534970903e-04	5.655694856620164e-04
26	-4.083388430444757e-04	-4.107055783463471e-04	-3.983150936476483e-04
27	3.071233913942706e-04	3.006573503014861e-04	3.345087840287361e-04
28	-2.024309033004101e-04	-2.200964560897217e-04	-1.276126192378242e-04
29	2.093849761877209e-04	1.611217884235287e-04	4.137923295818237e-04
30	1.390814577462152e-05	-1.179493353323860e-04	5.723594206879990e-04
31	4.465862439246848e-04	8.634490618228544e-05	1.972303500539645e-03
32	9.209887793986127e-04	-6.320885830020099e-05	5.089325842455288e-03
33	2.735150046646595e-03	4.627209576416886e-05	1.412325868598987e-02
34	7.312277652090415e-03	-3.387352507206424e-05	3.842516905689031e-02
35	2.009485539747402e-02	2.479714138420923e-05	1.050968554857603e-01
36	5.481426609912887e-02	-1.815276737570999e-05	2.870440490853013e-01
37	1.498182429932058e-01	1.328874801699847e-05	7.842818091421232e-01
38	4.092650181846693e-01	-9.728038717423045e-06	2.142651716454849e+00
39	1.118166522355750e+00	7.121418599150844e-06	5.853867051193944e+00
40	3.054863081080839e+00	-5.213240236544398e-06	1.599303753529759e+01
41	8.346059206873178e+00	3.816356725212891e-06	4.369380917298306e+01
42	2.280184457590804e+01	-2.793767022663012e-06	1.193736934165613e+02
43	6.229580756556243e+01	2.045179405099756e-06	3.261350051790887e+02
44	1.701953042829409e+02	-1.497175235126512e-06	8.910173971912998e+02
45	4.649822236970067e+02	1.096008339946487e-06	2.434304804740777e+03
46	1.270355055959895e+03	-8.023337903600500e-07	6.650644403864154e+03
47	3.470674559313804e+03	5.873490991728730e-07	1.816989841720986e+04
48	9.482059230547398e+03	-4.299693823743542e-07	4.964108564214802e+04
49	2.590546757972240e+04	3.147594335970373e-07	1.356219681187158e+05



50	7.077505362053961e+04	-2.304198975546335e-07	3.705261075217275e+05
51	1.933610424005240e+05	1.686790720848074e-07	1.012296151280887e+06
52	5.282721920421273e+05	-1.234816509396521e-07	2.765644517605228e+06
53	1.443266468885303e+06	9.039484229031054e-08	7.555881337772229e+06
54	3.943077321854860e+06	-6.617361729868312e-08	2.064305171075491e+07
55	1.077268758148032e+07	4.844244998325481e-08	5.639786609705429e+07
56	2.943152980667037e+07	-3.546233463085662e-08	1.540818356156184e+08
57	8.040843477630138e+07	2.596023070479635e-08	4.209594034253453e+08
58	2.196799291659435e+08	-1.900420785212053e-08	1.150082478081927e+09
59	6.001767278844898e+08	1.391204570535163e-08	3.142083763014545e+09
60	1.639713314100867e+09	-1.018432429353779e-08	8.584332482192945e+09
61	4.479780083970713e+09	7.455442823627675e-09	2.345283249041498e+10
62	1.223898679614316e+10	-5.457762939820230e-09	6.407432994521584e+10
63	3.343753376022774e+10	3.995359767614888e-09	1.750543248712616e+11
64	9.135304111274179e+10	-2.924806344410680e-09	4.782573096329550e+11
65	2.495811497459391e+11	2.141106846408414e-09	1.306623269008433e+12
66	6.818683817173617e+11	-1.567398996004532e-09	3.569761157282776e+12
67	1.862899062926602e+12	1.147415700807764e-09	9.752768852582418e+12
68	5.089534889287927e+12	-8.399665903935332e-10	2.664506001973039e+13
69	1.390486790442906e+13	6.148982208284623e-10	7.279565774462559e+13
70	3.798880558743397e+13	-4.501367391301416e-10	1.988814355287120e+14
71	1.037873469837260e+14	3.295229633966412e-10	5.433541865466751e+14
72	2.835523051423200e+14	-2.412275514670007e-10	1.484471244150774e+15
73	7.746793042520921e+14	1.765908238592808e-10	4.055650861394898e+15
74	2.116463218788824e+15	-1.292734552154398e-10	1.108024421109134e+16
75	5.782285046081833e+15	9.463473728768180e-11	3.027179014497248e+16
76	1.579749652974131e+16	-6.927743585551600e-11	8.270406871212765e+16
77	4.315956315164630e+16	5.071460286433157e-11	2.259517177142003e+17
78	1.179141193627752e+17	-3.712566598236882e-11	6.173115728526557e+17
79	3.221473650288430e+17	2.717787376392549e-11	1.686526581133712e+18
80	8.801229687832365e+17	-1.989558443688666e-11	4.607676307972735e+18
81	2.404540667624159e+18	1.456457865407766e-11	1.258840577821289e+19
82	6.569327272814792e+18	-1.066201156561798e-11	3.439216417237126e+19
83	1.794773588087790e+19	7.805134176919353e-12	9.396113990116830e+19
84	4.903412630738538e+19	-5.713754777397256e-12	2.567066081470791e+20
85	1.339637243765266e+20	4.182758799044191e-12	7.013354960964948e+20
86	3.659957013678239e+20	-3.061991956706127e-12	1.916084208487148e+21
87	9.999188514887010e+20	2.241533684676127e-12	5.234839409167285e+21
88	2.731829105713050e+21	-1.640916544060000e-12	1.430184723530886e+22
89	7.463495914403501e+21	1.201234281232254e-12	3.907337328895229e+22
90	2.039065004023310e+22	-8.793645256554909e-13	1.067504410485223e+23
91	5.570829190927321e+22	6.437395111535248e-13	2.916476286749492e+23
92	1.521978838990126e+23	-4.712500290039321e-13	7.967961394469430e+23
93	4.158123516165716e+23	3.449789642991852e-13	2.176887536243784e+24
94	1.136020471031169e+24	-2.525421294094934e-13	5.947367351381454e+24
95	3.103665645295480e+24	1.848736697793836e-13	1.624850977525048e+25
96	8.479372232653297e+24	-1.353369192602197e-13	4.439175425326385e+25
97	2.316607575589755e+25	9.907350103832773e-14	1.212805280570287e+26
98	6.329089597710171e+25	-7.252683644378381e-14	3.313445646205851e+26
99	1.729139434659985e+26	5.309332918908782e-14	9.052501853552273e+26
100	4.724096788862005e+26	-3.886701450939198e-14	2.473189499951625e+27

Note that  $x_n$  and  $z_n$  exhibit exponential growth by the  $n = 100$  term whereas  $y_n$  continues to decay. Rounding error in the early calculations of  $x_n$  introduce errors that are approximately equal to  $\alpha$ . These errors grow exponentially with order  $\alpha(1 + \sqrt{3})^n$ .

7. [Kincaid and Cheney Exercise 2.3#2] Find analytically the solution of this difference equation with the given initial values

$$\begin{cases} x_0 = 1 & x_1 = 0.9 \\ x_{n+1} = -0.2x_n + 0.99x_{n-1} \end{cases}$$

Without computing the solution recursively, predict whether such a computation would be stable.

The characteristic polynomial for this difference equation is

$$p(\lambda) = \lambda^2 + 0.2\lambda - 0.99 = (\lambda - 0.9)(\lambda + 1.1).$$

Solving  $p(\lambda) = 0$  we obtain  $\lambda_1 = 0.9$  and  $\lambda_2 = -1.1$ . It follows that any solution  $x_n$  of this difference equation may be written in the form

$$x_n = \alpha(0.9)^n + \beta(-1.1)^n$$

for a suitable choice of  $\alpha$  and  $\beta$ . Solving for  $\alpha$  and  $\beta$  to satisfy the initial conditions

$$\begin{cases} \alpha + \beta = 1 \\ 0.9\alpha - 1.1\beta = 0.9 \end{cases}$$

yields

$$\alpha = 1 \quad \text{and} \quad \beta = 0.$$

It follows that  $x_n = (0.9)^n$  and so  $x_n \rightarrow 0$  as  $n \rightarrow \infty$ .

Any perturbation to the initial conditions  $x_0$  and  $x_1$  leads to a perturbation in the constants  $\alpha$  and  $\beta$ . As the set of perturbations which leave  $\beta = 0$  is a one-dimensional subset of the two-dimensional space of perturbations, it follows that  $\beta \neq 0$  after a generic perturbation. In particular, since it is not possible to represent the condition  $x_1 = 0.9$  exactly using a base-2 floating point number, any computation made from the recurrence relation will involve rounding errors that lead to a perturbation where  $\beta \neq 0$ . In this case

$$|x_n| \geq |\beta|(1.1)^n - |\alpha|(0.9)^n \rightarrow \infty \quad \text{as} \quad n \rightarrow \infty,$$

which shows that the recursive computation is unstable.

8. [Kincaid and Cheney Exercise 2.3#5b] Compute the condition number for  $f(x) = \ln x$  and determine where the condition number is large.

The condition number  $\kappa_f$  is

$$\kappa_f = \frac{xf'(x)}{f(x)} = \frac{1}{\ln x}.$$

Since  $|\kappa_f| \rightarrow \infty$  as  $x \rightarrow 1$ , the condition number is large when  $x \approx 1$ .

9. [Kincaid and Cheney Exercise 2.3#5c] Compute the condition number for  $g(x) = \sin x$  and determine where the condition number is large.

The condition number  $\kappa_g$  is

$$\kappa_g = \frac{xg'(x)}{g(x)} = \frac{x \cos x}{\sin x}.$$

Since  $|\sin x| \leq |x| \leq |\tan x|$  for  $|x| < \pi/2$  it follows that  $|k_g| \leq 1$  for  $|x| < \pi/2$ . On the other hand  $|\kappa_g| \rightarrow \infty$  as  $x \rightarrow k\pi$  for any integer  $k \neq 0$ . Therefore the condition number is large when  $x \approx k\pi$  for any interger  $k \neq 0$ .

10. [Kincaid and Cheney Computer Exercise 2.3#1] Let sequences  $a_n$  and  $b_n$  be generated as follows

$$\begin{cases} a_0 = 0 & a_1 = 1 \\ a_n = na_{n-1} + a_{n-2} \end{cases} \quad \begin{cases} b_0 = 1 & b_1 = 1 \\ b_n = nb_{n-1} + b_{n-2} \end{cases}.$$

What is  $\lim_{n \rightarrow \infty} (a_n/b_n)$ ?

I wrote a program to compute this limit. The program was

```

1 /*
2   Kincaid and Cheney Computer Exercise 2.3#1
3   Written October 15, 2012 by for Math 701 Eric Olson
4 */
5
6 #include <stdio.h>
7 #include <math.h>
8
9 #define N 20
10 double a[N],b[N];
11
12 main(){
13     printf(
14         "Kincaid and Cheney Computer Exercise 2.3#1\n"
15         "Written October 15, 2012 by for Math 701 Eric Olson\n");
16     a[0]=0.0;
17     a[1]=1.0;
18     b[0]=1.0;
19     b[1]=1.0;
20     int n;
21     double aob=-1.0,aobo;
22     for(n=2;n<N;n++){
23         a[n]=n*a[n-1]+a[n-2];
24         b[n]=n*b[n-1]+b[n-2];
25         aobo=aob;
26         aob=a[n]/b[n];
27         if(aobo==aob) break;
28     }
29     printf("\n%4s %24s %24s %24s\n",
30         "n", "a[n]", "b[n]", "a[n]/b[n]");
31     int k;
32     for(k=0;k<=n;k++){
33         printf("%4d %24.15e %24.15e %24.15e\n",
34             k,a[k],b[k],a[k]/b[k]);
35     }
36     return 0;
37 }

```

and the output was

```

Kincaid and Cheney Computer Exercise 2.3#1
Written October 15, 2012 by for Math 701 Eric Olson

n          a[n]          b[n]          a[n]/b[n]
0  0.000000000000000e+00  1.000000000000000e+00  0.000000000000000e+00
1  1.000000000000000e+00  1.000000000000000e+00  1.000000000000000e+00
2  2.000000000000000e+00  3.000000000000000e+00  6.666666666666666e-01

```

3	7.000000000000000e+00	1.000000000000000e+01	7.000000000000000e-01
4	3.000000000000000e+01	4.300000000000000e+01	6.976744186046512e-01
5	1.570000000000000e+02	2.250000000000000e+02	6.977777777777778e-01
6	9.720000000000000e+02	1.393000000000000e+03	6.977745872218234e-01
7	6.961000000000000e+03	9.976000000000000e+03	6.977746591820368e-01
8	5.666000000000000e+04	8.120100000000000e+04	6.977746579475622e-01
9	5.169010000000000e+05	7.407850000000000e+05	6.977746579641867e-01
10	5.225670000000000e+06	7.489051000000000e+06	6.977746579640064e-01
11	5.799927100000000e+07	8.312034600000000e+07	6.977746579640079e-01
12	7.012169220000000e+08	1.004933203000000e+09	6.977746579640079e-01

Thus  $\lim_{n \rightarrow \infty} (a_n/b_n) \approx 0.6977746579640079$ .

11. [Kincaid and Cheney Computer Exercise 2.3#2] The Bessel functions  $Y_n$  satisfy the recurrence formula

$$Y_n(x) = 2(n-1)x^{-1}Y_{n-1}(x) - Y_{n-2}(x).$$

For  $Y_n(1)$  the starting values are

$$\begin{cases} Y_0(1) \approx 0.0882569642 \\ Y_1(1) \approx -0.7812128213. \end{cases}$$

Compute  $Y_2(1), Y_3(1), \dots, Y_{20}(1)$  using the recurrence formula. Try to decide whether the results are reliable.

The program

```

1 /*
2   Kincaid and Cheney Computer Exercise 2.3#2
3   Written October 15, 2012 by for Math 701 Eric Olson
4 */
5
6 #include <stdio.h>
7 #include <math.h>
8
9 #define N 21
10 double y[N];
11
12 void computeYn(double *y){
13     int n;
14     for(n=2;n<N;n++){
15         y[n]=2.0*(n-1)*y[n-1]-y[n-2];
16     }
17 }
18
19 main(){
20     printf(
21         "Kincaid and Cheney Computer Exercise 2.3#2\n"
22         "Written October 15, 2012 by for Math 701 Eric Olson\n");
23     y[0]= 0.0882569642;
24     y[1]=-0.7812128213;
25     computeYn(y),
26     printf("\n%4s %24s\n", "n", "y[n]");
27     int n;
28     for(n=0;n<N;n++){
29         printf("%4d %24.15e\n", n, y[n]);
30     }
31     return 0;
32 }

```

computes the values of  $Y_n(1)$  for  $n = 0, 1, \dots, 20$  as

```

Kincaid and Cheney Computer Exercise 2.3#2
Written October 15, 2012 by for Math 701 Eric Olson

```

n	y[n]
0	8.825696420000000e-02
1	-7.812128213000000e-01
2	-1.650682606800000e+00
3	-5.821517605900000e+00
4	-3.327842302860000e+01
5	-2.604058666229000e+02
6	-2.570780243200400e+03
7	-3.058895705178190e+04
8	-4.256746184817462e+05
9	-6.780204938656158e+06
10	-1.216180142773291e+08
11	-2.425580080607925e+09
12	-5.324114375909703e+10
13	-1.275361870137721e+12
14	-3.310616747982164e+13
15	-9.256973275648682e+14
16	-2.773781365946622e+16
17	-8.866843397753542e+17
18	-3.011952973870258e+19
19	-1.083416386253517e+21
20	-4.113970314789496e+22

To determine whether the results are reliable perturbed starting values given by

$$\begin{cases} Z_0 = Y_0(1)(1 + \alpha) \\ Z_1 = Y_1(1)(1 + \beta) \end{cases}$$

where  $\alpha$  and  $\beta$  are uniformly distributed random variables on the interval  $[-0.1, 0.1]$  were used and the results compared to the results with the unperturbed starting values. The program

```

1 /*
2   Kincaid and Cheney Computer Exercise 2.3#2 part 2
3   Written October 15, 2012 by for Math 701 Eric Olson
4 */
5
6 #include <stdio.h>
7 #include <stdlib.h>
8 #include <math.h>
9
10 #define N 21
11 double x[N],y[N],z[N];
12
13 double relerr(double *z,double *y){
14     int n;
15     double x1=0.0;
16     double x2=0.0;
17     for(n=0;n<N;n++){
18         double t2=y[n];
19         double t1=z[n]-t2;
20         x1+=t1*t1;
21         x2+=t2*t2;

```



```

22     }
23     return sqrt(x1/x2);
24 }
25
26 void computeYn(double *y){
27     int n;
28     for(n=2;n<N;n++){
29         y[n]=2.0*(n-1)*y[n-1]-y[n-2];
30     }
31 }
32
33 main(){
34     printf(
35         "Kincaid and Cheney Computer Exercise 2.3#2 part 2\n"
36         "Written October 15, 2012 by for Math 701 Eric Olson\n");
37     y[0]= 0.0882569642;
38     y[1]=-0.7812128213;
39     computeYn(y),
40
41     printf("\n%4s %24s %24s %24s\n",
42         "k", "alpha[k]", "beta[k]", "E_rel[k]");
43     int k;
44     for(k=0;k<20;k++){
45         double alpha=0.2*random()/RAND_MAX-0.1;
46         double beta=0.2*random()/RAND_MAX-0.1;
47         z[0]=y[0]*(1.0+alpha);
48         z[1]=y[1]*(1.0+beta);
49         computeYn(z);
50         printf("%4d %24.15e %24.15e %25.15e\n",
51             k+1,alpha,beta,relerr(z,y));
52     }
53     return 0;
54 }

```

compares the relative errors

$$E_{\text{rel}} = \sqrt{\sum_{n=0}^{20} |Z_n - Y_n(1)|^2} / \sqrt{\sum_{n=0}^{20} |Y_n(1)|^2}$$

for 20 different perturbations. The output

```

Kincaid and Cheney Computer Exercise 2.3#2 part 2
Written October 15, 2012 by for Math 701 Eric Olson

```

k	alpha[k]	beta[k]	E_rel[k]
1	6.803754343094190e-02	-2.112341463618140e-02	1.568407481919753e-02
2	5.661984475172118e-02	5.968800669521467e-02	5.950083087537189e-02
3	8.232947158735687e-02	-6.048972614132321e-02	5.177691993470489e-02
4	-3.295544885702219e-02	5.364591896238081e-02	4.836272911789027e-02
5	-4.444505783936244e-02	1.079399115908611e-02	7.424086156009646e-03
6	-4.520589627567954e-03	2.577418495238488e-02	2.392602655221173e-02
7	-2.704310544163133e-02	2.680182039123113e-03	8.668876938113203e-04
8	9.044594503494256e-02	8.323901360074012e-02	8.367867856125882e-02

9	2.714234559198020e-02	4.345938588653663e-02	4.246395101354367e-02
10	-7.167948892883932e-02	2.139377525141173e-02	1.571576188404672e-02
11	-9.673988567513409e-02	-5.142264587405261e-02	5.418726249762213e-02
12	-7.255368464279627e-02	6.083535084539808e-02	5.269783967897831e-02
13	-6.866418214918310e-02	-1.981112115076330e-02	2.279144354333087e-02
14	-7.404191064370885e-02	-7.823823959484615e-02	7.798223899050180e-02
15	9.978490360071181e-02	-5.634861893781862e-02	4.682356106136112e-02
16	2.586478880879692e-03	6.782244693852144e-02	6.384267141515562e-02
17	2.252796651913224e-02	-4.079367646053139e-02	3.693068596107059e-02
18	2.751045354060384e-02	4.857438013356849e-03	6.239404425599178e-03
19	-1.283402601854598e-03	9.455500477671391e-02	8.870830150488451e-02
20	-4.149664311739460e-02	5.427153955878296e-02	4.842912040444308e-02

indicates that the relative error  $E_{\text{rel}}$  has magnitude similar to  $\alpha$  and  $\beta$ . This is evidence that the results are reliable.

12. [Kincaid and Cheney Computer Exercise 2.3#6] The Bessel functions  $J_0(x), J_1(x), \dots$  can be computed by the recurrence relation

$$J_n(x) = 2(n-1)x^{-1}J_{n-1}(x) - J_{n-2}(x)$$

if it is used with descending  $n$ . Thus, we start with  $n = N$  and proceed downward by the formula

$$J_{n-1}(x) = 2nx^{-1}J_n(x) - J_{n+1}(x) \quad \text{for } n = N, N-1, \dots, 1.$$

To get started, we set  $J_{N+1}(x) = 0$  and  $J_N(x) = 1$ . These are tentative values. After computing  $J_{N-1}(x), J_{N-2}(x), \dots, J_0(x)$  we scale them by using the identity

$$J_0(x)^2 + 2 \sum_{n=1}^{\infty} J_n(x)^2 = 1.$$

Obviously  $N$  must be large enough so taking  $J_{N+1}(x) = 0$  is reasonable for the precision expected. Use  $N = 51$  to compute  $J_0(1), J_1(1), \dots, J_{50}(1)$ .

The program was

```

1 /*
2   Kincaid and Cheney Computer Exercise 2.3#5
3   Written October 15, 2012 by for Math 701 Eric Olson
4 */
5
6 #include <stdio.h>
7 #include <math.h>
8
9 #define N 51
10 double J[N+2];
11
12 main(){
13     printf(
14         "Kincaid and Cheney Computer Exercise 2.3#5\n"
15         "Written October 15, 2012 by for Math 701 Eric Olson\n");
16     J[N+1]=0.0;
17     J[N]=1.0;
18     int n;
19     for(n=N;n>0;n--) J[n-1]=2*n*J[n]-J[n+1];
20     double s=0;
21     for(n=1;n<=N;n++) s+=J[n]*J[n];
22     s=1.0/sqrt(J[0]*J[0]+2*s);
23     for(n=0;n<=N;n++) J[n]*=s;
24     printf("\n%4s %24s\n", "n", "J[n]");
25     for(n=0;n<N;n++){
26         printf("%4d %24.15e\n", n, J[n]);
27     }
28     return 0;
29 }
```

and the output was

Kincaid and Cheney Computer Exercise 2.3#5  
Written October 15, 2012 by for Math 701 Eric Olson

n	J[n]
0	7.651976865579665e-01
1	4.400505857449335e-01
2	1.149034849319005e-01
3	1.956335398266840e-02
4	2.476638964109955e-03
5	2.497577302112344e-04
6	2.093833800238927e-05
7	1.502325817436808e-06
8	9.422344172604498e-08
9	5.249250179911873e-09
10	2.630615123687452e-10
11	1.198006746303137e-11
12	4.999718179448403e-13
13	1.925616764480172e-14
14	6.885408200044221e-16
15	2.297531532210343e-17
16	7.186396586807489e-19
17	2.115375568053260e-20
18	5.880344573595755e-22
19	1.548478441211653e-23
20	3.873503008524656e-25
21	9.227621982096665e-27
22	2.098223955943776e-28
23	4.563424055950103e-30
24	9.511097932712489e-32
25	1.902951751891381e-33
26	3.660826744416801e-35
27	6.781552053554108e-37
28	1.211364502417112e-38
29	2.089159981718167e-40
30	3.482869794251480e-42
31	5.618948327215678e-44
32	8.781686222395142e-46
33	1.330855117212921e-47
34	1.957551210137318e-49
35	2.797056804555225e-51
36	3.885530513390645e-53
37	5.251650860388840e-55
38	6.911232970971623e-57
39	8.861975495926604e-59
40	1.107915851128632e-60
41	1.351313102452444e-62
42	1.608928823721760e-64
43	1.871094738345342e-66
44	2.126512552341710e-68
45	2.363077153624461e-70
46	2.568859203053248e-72
47	2.733131845271327e-74
48	2.847315017992931e-76
49	2.905720018862307e-78
50	2.906004921305573e-80

I checked the output with the values  $J_0(1) \approx 0.7651976866$  and  $J_1(1) \approx 0.4400505857$  appearing on page 36 in Kincaid and Cheney as well as with Maple.