1 Fundamentals

1.0 Preliminaries

The first question we want to answer is: What is "computational mathematics"?

One possible definition is: "The study of algorithms for the solution of computational problems in science and engineering."

Other names for roughly the same subject are *numerical analysis* or *scientific computing*.

What is it we are looking for in these algorithms? We want algorithms that are

- fast,
- stable and reliable,
- accurate.

Note: One could also study hardware issues such as computer architecture and its effects, or software issues such as efficiency of implementation on a particular hardware or in a particular programming language. We will not do this.

What sort of problems are typical?

Example The *Poisson problem* provides the basis for many different algorithms for the numerical solution of differential equations, which in turn lead to the need for many algorithms in numerical linear algebra. Consider

$$\begin{aligned} -\nabla^2 u(x,y) = -\left[u_{xx}(x,y) + u_{yy}(x,y)\right] &= f(x,y), \qquad \text{in } \Omega = [0,1]^2 \\ u(x,y) &= 0, \qquad \text{on } \partial\Omega. \end{aligned}$$

One possible algorithm for the numerical solution of this problem is based on the following discretization of the Laplacian:

$$\nabla^2 u(x_j, y_k) \approx \frac{u_{j-1,k} + u_{j,k-1} + u_{j+1,k} + u_{j,k+1} - 4u_{j,k}}{h^2}, \tag{1}$$

where the unit square is discretized by a set of $(n+1)^2$ equally spaced points (x_j, y_k), $j,k = 0,\ldots,n$, and $h = \frac{1}{n}$. Also, we use the abbreviation $u_{j,k} = u(x_j, y_k)$.

Formula (1) is a straightforward generalization to two dimensions of the linear approximation

$$u'(x) \approx \frac{u(x+h) - u(x)}{h}.$$

If we visit all of the $(n-1)^2$ interior grid points and write down the equation resulting from the discretization of the PDE, then we obtain the following system of linear equations

$$4u_{j,k} - u_{j-1,k} - u_{j,k-1} - u_{j+1,k} - u_{j,k+1} = \frac{f_{j,k}}{n^2}, \qquad j,k = 1,\ldots,n-1$$

along with the discrete boundary conditions

$$u_{j,0} = u_{j,n} = u_{0,k} = u_{n,k} = 0, \qquad j,k = 0,1,\ldots,n.$$

Jacobi	10^{13} operations	1845
Gauss-Seidel	5×10^{12} operations	1832
SOR	10^{10} operations	1950
FFT	1.5×10^{8} operations	1965
multigrid	10^{8} operations	1979

Table 1: Improvements of algorithms for Poisson problem.

Z3	1 flops	1941
Intel Paragon	10 Gflops	1990
NEC Earth Simulator (5120 processors)	40 Tflops	2002
IBM Blue Gene/L (131072 processors)	367 Tflops	2006

Table 2: Improvements of hardware for Poisson problem.

This method is known as the *finite difference method*.

In order to obtain a relative error of 10^{-4} using finite differences one needs about $n = 1000$, i.e., 10^6 points. Therefore, one needs to solve a $10^6 \times 10^6$ sparse system of linear equations. Note that the system is indeed sparse since each row of the system matrix contains at most 5 nonzero entries.

Using the state-of-the-art algorithms and hardware of 1940 it would have taken one of the first computers about 300,000 years to solve this problem with the desired accuracy. In 1990, on the other hand, it took about 1/100 second. In fact, the Earth Simulator (the fastest computer available in 2002) was able to solve a *dense* system of 10^6 linear equations in 10^6 unknowns in less than 6 hours using Fortran and MPI code.

This example is typical and shows – in addition to the huge improvements possible by advances in both software and hardware – that using numerical methods we can usually expect only an approximate solution.

As just noted, errors are introduced in a variety of ways:

- through discretization, i.e., by converting a continuous problem to a discrete one,
- through floating-point representations and roundoff errors,
- through the nature of certain algorithms (e.g., iterative vs. direct).

While other sources of errors also exist (such as measurement errors in experiments), we will focus on the above three sources.

1.1 Fundamentals from Linear Algebra

1.1.1 Basic Definitions

Definition 1.1 *A* vector space *(or* linear space*) V over the field $\mathbb{C}$ of complex numbers consists of a set of elements (or* vectors*) together with two operations "+": $V \times V \to V$ (vector addition) and "$\cdot$": $\mathbb{C} \times V \to V$ (scalar multiplication) such that*

1. *For any $\boldsymbol{u}, \boldsymbol{v} \in V$ we have $\boldsymbol{u} + \boldsymbol{v} \in V$, i.e., V is closed under vector addition.*

2. *Vector addition is associative and commutative, i.e.,* $(\boldsymbol{u}+\boldsymbol{v})+\boldsymbol{w}=\boldsymbol{u}+(\boldsymbol{v}+\boldsymbol{w})$ *and* $\boldsymbol{u}+\boldsymbol{v}=\boldsymbol{v}+\boldsymbol{u}$.

3. *There is a zero vector* $\mathbf{0}$ *such that* $\boldsymbol{u}+\mathbf{0}=\boldsymbol{u}$ *for every* $\boldsymbol{u}\in V$.

4. *For every* $\boldsymbol{u}\in V$ *there is a negative* $-\boldsymbol{u}$ *such that* $\boldsymbol{u}+(-\boldsymbol{u})=\mathbf{0}$.

5. *For every* $\boldsymbol{u}\in V$ *and every scalar* $\alpha\in\mathbb{C}$ *we have* $\alpha\boldsymbol{u}\in V$*, i.e.,* V *is closed under scalar multiplication.*

6. *For every* $\alpha,\beta\in\mathbb{C}$ *and every* $\boldsymbol{u},\boldsymbol{v}\in V$ *we have* $(\alpha+\beta)\boldsymbol{u}=\alpha\boldsymbol{u}+\beta\boldsymbol{u}$*, and* $\alpha(\boldsymbol{u}+\boldsymbol{v})=\alpha\boldsymbol{u}+\alpha\boldsymbol{v}$ *(i.e., distributive laws hold).*

Remark Often we consider V as a vector space over $\mathbb{R}$.

Example Standard examples we will be working with are $V=\mathbb{R}^m$ or $V=\mathbb{C}^m$ with the usual vector addition and scalar multiplication, or $V=\mathbb{R}^{m\times n}$ or $V=\mathbb{C}^{m\times n}$ with the usual addition of matrices and scalar multiplication.

If $A\in\mathbb{C}^{m\times n}$ is an $m\times n$ matrix then A is a *linear map* (or *linear transformation*) since it satisfies

1. $A(\boldsymbol{x}+\boldsymbol{y})=A\boldsymbol{x}+A\boldsymbol{y}$ for every $\boldsymbol{x},\boldsymbol{y}\in\mathbb{C}^n$,
2. $A(\alpha\boldsymbol{x})=\alpha A\boldsymbol{x}$ for every $\boldsymbol{x}\in\mathbb{C}^n$ and $\alpha\in\mathbb{C}$.

Conversely, any linear map from $\mathbb{C}^n$ to $\mathbb{C}^m$ can be associated with matrix multiplication by a matrix in $\mathbb{C}^{m\times n}$.

Example The matrix

$$A=\begin{bmatrix}1&0&0\\0&\cos\theta&-\sin\theta\\0&\sin\theta&\cos\theta\end{bmatrix}$$

maps $\mathbb{R}^3$ into $\mathbb{R}^3$ since it represents counterclockwise rotation about the x-axis by an angle θ.

The matrix

$$A=\begin{bmatrix}1&0&0\\0&1&0\end{bmatrix}$$

maps $\mathbb{R}^3$ into $\mathbb{R}^2$ by projecting into the x-y plane.

1.1.2 Matrix-Vector Multiplication

For the following we assume $A\in\mathbb{C}^{m\times n}$ and $\boldsymbol{x}\in\mathbb{C}^n$. In fact, our vectors are always to be interpreted as *column vectors* unless noted otherwise.

A first interpretation of a matrix-vector product $\boldsymbol{b}=A\boldsymbol{x}$ is obtained (using Matlab notation) via the representation

$$\begin{aligned}\boldsymbol{b}(i) &= \sum_{j=1}^{n}A(i,j)\boldsymbol{x}(j), \qquad i=1,\ldots,m\\ &= A(i,:)\boldsymbol{x},\end{aligned}$$

i.e., the i-th entry of $\boldsymbol{b}$ is given by the dot product of row i of A with $\boldsymbol{x}$.

A second (vectorized) interpretation is obtained via

$$\begin{aligned} \boldsymbol{b}(:) &= \sum_{j=1}^{n} A(:,j)\boldsymbol{x}(j) \\ &= \sum_{j=1}^{n} \boldsymbol{x}(j)A(:,j), \end{aligned}$$

i.e., the (entire) vector $\boldsymbol{b}$ is given as a linear combination of the columns of A.

Remark Using the first interpretation we need to perform m dot products to calculate $\boldsymbol{b}$. With the second interpretation we compute n scalar products and $n-1$ additions.

Geometrically (based on the second approach above) we can interpret A as a linear transformation, i.e., on the one hand $\boldsymbol{b}$ represents a point in $\mathbb{C}^m$ with coordinates $\boldsymbol{b}(1), \boldsymbol{b}(2), \ldots, \boldsymbol{b}(m)$ with respect to the standard basis $\{\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_m\}$ whereas $A\boldsymbol{x}$ represents the same point in $\mathbb{C}^m$ with coordinates $\boldsymbol{x}(1), \boldsymbol{x}(2), \ldots, \boldsymbol{x}(m)$ with respect to the basis $\{A(:,1), A(:,2), \ldots, A(:,m)\}$ of columns of A.

1.1.3 Matrix-Matrix Multiplication

If we assume that $A \in \mathbb{C}^{\ell \times m}$ and $C \in \mathbb{C}^{m \times n}$ then

$$B = AC \in \mathbb{C}^{\ell \times n}$$

and

$$\begin{aligned} B(i,j) &= \sum_{k=1}^{m} A(i,k)C(k,j), \qquad i = 1, \ldots, \ell,\ j = 1, \ldots, n \\ &= A(i,:)C(:,j), \end{aligned}$$

i.e., the ij entry of B is obtained as the dot product of row i of A with column j of C.

An alternative (vectorized) interpretation of the same matrix-matrix product is given by

$$\begin{aligned} B(:,j) &= \sum_{k=1}^{m} A(:,k)C(k,j) \\ &= \sum_{k=1}^{m} C(k,j)A(:,k), \end{aligned}$$

i.e., the j-th column of B is given as a linear combination of the columns of A.

Example Let's take $C = R \in \mathbb{C}^{n \times n}$ with

$$R(i,j) = \begin{cases} 1 & i \leq j \\ 0 & i > j \end{cases},$$

i.e., an upper triangular matrix of all ones.

Then $B = AR$ leads to

$$\begin{aligned} B(:,j) &= \sum_{k=1}^{n} R(j,k)A(:,k) \\ &= \sum_{k=1}^{j} 1 \; A(:,k) = \sum_{k=1}^{j} A(:,k) \end{aligned}$$

since $R(k,j) = 0$ if $k > j$ by definition and 1 otherwise.

Thus, column j of B is given by the sum of the first j columns of A (which is similar to computation of the integral $\int_0^x f(t)dt = F(x)$.

1.1.4 Range and Nullspace

The *range of A* (range(A) or *column space of A*) is given by the set of all $A\boldsymbol{x}$.

The following theorem is a direct consequence of the vectorized interpretation of matrix-vector multiplication:

Theorem 1.2 *The range(A) is a vector space spanned by the columns of A.*

Remark The columns of A are a basis for range(A) if they are linearly independent.

The *nullspace of A* (null(A) or *kernel of A*) is given by the set of all $\boldsymbol{x}$ such that $A\boldsymbol{x} = \mathbf{0}$.

Remark From the vectorized interpretation of the matrix-vector product $A\boldsymbol{x} = \mathbf{0}$ we know that

$$\mathbf{0} = \boldsymbol{x}(1)A(:,1) + \ldots + \boldsymbol{x}(n)A(:,n),$$

so null(A) characterizes the linear dependence of the columns of A.

In particular, if null(A) contains only the zero vector then the column of A are linearly independent and form a basis for range(A).

The *column rank of A* is given by the *dimension of range(A)* (i.e., number of linearly independent columns of A). The *row rank* is defined analogously.

Remark We always have "row rank = column rank = *rank*". Moreover, for any $m \times n$ matrix we have

$$\dim(\text{null}(A)) + \text{rank}(A) = n.$$

Example Consider the matrix

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 2 \\ 0 & 0 & 0 \end{bmatrix}.$$

The range(A) is given by all vectors in $\mathbb{C}^3$ with third component zero since

$$A\boldsymbol{x} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 2 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{x}(1) \\ \boldsymbol{x}(2) \\ \boldsymbol{x}(3) \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}(2) \\ 2\boldsymbol{x}(3) \\ 0 \end{bmatrix}.$$

Since A has two linearly independent columns (and rows) we have

$$\text{rank}(A) = \dim(\text{range}(A)) = 2.$$

Moreover,

$$\left\{ \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 2 \\ 0 \end{bmatrix} \right\}$$

is a basis for range(A).

The nullspace of A is given by all vectors with last two components zero since

$$A \begin{bmatrix} x \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

for any x. Alternatively, we can see that

$$\boldsymbol{x}(1)A(:,1) + \boldsymbol{x}(2)A(:,2) + \boldsymbol{x}(3)A(:,3) = \boldsymbol{x}(1) \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} + \boldsymbol{x}(2) \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} + \boldsymbol{x}(3) \begin{bmatrix} 0 \\ 2 \\ 0 \end{bmatrix} = \boldsymbol{0}$$

as soon as $\boldsymbol{x}(2) = \boldsymbol{x}(3) = 0$.

Therefore, $\dim(\text{null}(A)) = 1$.

1.1.5 Inverse

Any square matrix $A \in \mathbb{C}^{m \times m}$ with $\text{rank}(A) = m$ (i.e., of *full rank* or *nonsingular*) has an *inverse* A^{-1} such that

$$AA^{-1} = A^{-1}A = I,$$

where $I = [\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_m]$ is the $m \times m$ identity matrix.

Theorem 1.3 *For $A \in \mathbb{C}^{m \times m}$ the following are equivalent:*

1. *A has an inverse A^{-1},*
2. *$\text{rank}(A) = m$,*
3. *$\text{range}(A) = \mathbb{C}^m$,*
4. *$\text{null}(A) = \{\boldsymbol{0}\}$,*
5. *0 is not an eigenvalue of A,*
6. *0 is not a singular value of A,*

7. $\det(A) \neq 0$.

Following our earlier geometric interpretations we can interpret multiplication by A^{-1} as a change of basis transformation:

On the one hand $A\boldsymbol{x} = \boldsymbol{b}$ gives us the coordinates $\boldsymbol{x}$ of a point in $\mathbb{C}^m$ with respect to the basis $\{A(:,1), \ldots, A(:,m)\}$ and on the other hand $\boldsymbol{b}$ can be viewed as the coordinates of the same point with respect to the standard basis $\{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_m\}$. Therefore

$$A\boldsymbol{x} = \boldsymbol{b} \Longleftrightarrow \boldsymbol{x} = A^{-1}\boldsymbol{b}$$

represents a change of basis.

In other words, if we multiply $\boldsymbol{b}$ (the coordinates of our point with respect to the standard basis) by A^{-1} then we obtain the coordinates $\boldsymbol{x}$ with respect to the column space of A. Conversely, if we multiply $\boldsymbol{x}$ by A, then we transform back to the standard basis.

Remark The solution $\boldsymbol{x}$ of the linear system $A\boldsymbol{x} = \boldsymbol{b}$ gives us the vector of coefficients of $\boldsymbol{b}$ expanded in terms of the columns of A, i.e., we "rotate $\boldsymbol{b}$ back into the column space of A via A^{-1}".

1.2 Orthogonal Vectors and Matrices

1.2.1 Adjoint

If $A \in \mathbb{C}^{m\times n}$ is an $m \times n$ matrix then $A^* \in \mathbb{C}^{n\times m}$ is called the *adjoint of* A provided $A^*(i,j) = \overline{A(j,i)}$, $i = 1, \ldots, m$, $j = 1, \ldots, n$.

If A is real then A^* is called the *transpose of* A and usually denoted by A^T (or A' in Matlab).

If $A = A^*$ then A is called *Hermitian* (or *symmetric* if real).

Remark Note that only square matrices can be Hermitian (or real symmetric).

Applied to vectors this means that $\boldsymbol{x}$ denotes a *column vector* and $\boldsymbol{x}^*$ a *row vector*.

1.2.2 Inner Product

If $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{C}^m$ then

$$\boldsymbol{x}^*\boldsymbol{y} = \sum_{i=1}^{m} \overline{\boldsymbol{x}(i)}\boldsymbol{y}(i)$$

is called the *inner product* (or *dot product*) of $\boldsymbol{x}$ and $\boldsymbol{y}$.

The *length* of a vector $\boldsymbol{x}$ is given by

$$\|\boldsymbol{x}\| = \sqrt{\boldsymbol{x}^*\boldsymbol{x}} = \left(\sum_{i=1}^{m} |\boldsymbol{x}(i)|^2\right)^{1/2}.$$

This is the same as the (Euclidean) or 2-norm of the vector. We will say more about norms later.

Once an inner product is available we can define an angle α between two vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ by

$$\cos\alpha = \frac{\boldsymbol{x}^*\boldsymbol{y}}{\|\boldsymbol{x}\|\|\boldsymbol{y}\|}.$$

We summarize the following properties of inner products.

Lemma 1.4 *Let $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z} \in \mathbb{C}^m$ and $\alpha, \beta \in \mathbb{C}$. Then*

1. $(\boldsymbol{x}+\boldsymbol{y})^*\boldsymbol{z} = \boldsymbol{x}^*\boldsymbol{z} + \boldsymbol{y}^*\boldsymbol{z}$,
2. $\boldsymbol{x}^*(\boldsymbol{y}+\boldsymbol{z}) = \boldsymbol{x}^*\boldsymbol{y} + \boldsymbol{x}^*\boldsymbol{z}$,
3. $(\alpha\boldsymbol{x})^*(\beta\boldsymbol{y}) = \overline{\alpha}\beta\boldsymbol{x}^*\boldsymbol{y}$.

Together, we say that an inner product is bilinear.

Moreover, the definition of the inner product implies that $\boldsymbol{y}^*\boldsymbol{x} = \overline{\boldsymbol{x}^*\boldsymbol{y}}$. In the real case, however, the inner product is symmetric.

We can use the inner product and its properties to obtain some other properties of matrix multiplication.

Lemma 1.5 *Assume A and B are $m \times m$ matrices. Then*

1. $(AB)^* = B^*A^*$,
2. $(AB)^{-1} = B^{-1}A^{-1}$.

Proof We will prove item 1. Let $C = AB$ so that $C(i,j) = A(i,:)B(:,j)$, the inner product of row i of A with column j of B. Now

$$\begin{aligned}
(AB)^*(i,j) &= C^*(i,j) \\
&= \overline{C(j,i)} \\
&= \overline{A(j,:)}\ \overline{B(:,i)} \\
&= \sum_{k=1}^{m} \overline{A(j,k)}\ \overline{B(k,i)} \\
&= \sum_{k=1}^{m} B^*(i,k)A^*(k,j) \\
&= B^*(i,:)A^*(:,j) \\
&= (B^*A^*)(i,j).
\end{aligned}$$

■

Remark We will use the notational convention $A^{-*} = (A^{-1})^* = (A^*)^{-1}$.

1.2.3 Orthogonal Vectors

Since we defined angles earlier as

$$\cos\alpha = \frac{\boldsymbol{x}^*\boldsymbol{y}}{\|\boldsymbol{x}\|\|\boldsymbol{y}\|},$$

the vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ are *orthogonal* if and only if $\boldsymbol{x}^*\boldsymbol{y} = 0$.

- If X, Y are two sets of vectors then X *is orthogonal to* Y if $\boldsymbol{x}^*\boldsymbol{y} = 0$ for every $\boldsymbol{x} \in X$ and every $\boldsymbol{y} \in Y$.
- X is an *orthogonal set* (or simply orthogonal) if $\boldsymbol{x}^*\boldsymbol{y} = 0$ for every $\boldsymbol{x}, \boldsymbol{y} \in X$ with $\boldsymbol{x} \neq \boldsymbol{y}$.
- X is *orthonormal* if X is orthogonal and $\|\boldsymbol{x}\| = 1$ for all $\boldsymbol{x} \in X$.

Theorem 1.6 *Orthogonality implies linear independence, i.e., if X is orthogonal then X is linearly independent.*

Corollary 1.7 *If $X \in \mathbb{C}^m$ is orthogonal and consists of m vectors then X is a basis for $\mathbb{C}^m$.*

1.2.4 Orthogonal Decomposition of a Vector

In analogy to the Cartesian decomposition of a vector into its coordinates, i.e.,

$$\boldsymbol{v} = \boldsymbol{v}(1)\boldsymbol{e}_1 + \ldots + \boldsymbol{v}(m)\boldsymbol{e}_m$$

we can find the components of an arbitrary vector $\boldsymbol{v}$ with respect to any given orthogonal set.

Assume $Q = \{\boldsymbol{q}_1, \ldots, \boldsymbol{q}_n\}$ is an orthonormal set and $\boldsymbol{v} \in \mathbb{C}^m$ is an arbitrary vector with $m \geq n$.
Claim: We can decompose $\boldsymbol{v}$ as

$$\boldsymbol{v} = \boldsymbol{r} + \sum_{i=1}^{n} (\boldsymbol{q}_i^*\boldsymbol{v})\boldsymbol{q}_i$$

with $\{\boldsymbol{r}, \boldsymbol{q}_1, \ldots, \boldsymbol{q}_n\}$ an orthogonal set.

Remark The vectors $(\boldsymbol{q}_i^*\boldsymbol{v})\boldsymbol{q}_i$ are the projections of $\boldsymbol{v}$ onto the (basis) vectors $\boldsymbol{q}_i$.

Proof We need to show that $\boldsymbol{r} = \boldsymbol{v} - \sum_{i=1}^{n}(\boldsymbol{q}_i^*\boldsymbol{v})\boldsymbol{q}_i$ is orthogonal to Q. This can be done by considering the inner product with an arbitrary member $\boldsymbol{q}_j$ of Q:

$$\begin{aligned} \boldsymbol{q}_j^*\boldsymbol{r} &= \boldsymbol{q}_j^*\boldsymbol{v} - \boldsymbol{q}_j^*\left(\sum_{i=1}^{n}(\boldsymbol{q}_i^*\boldsymbol{v})\boldsymbol{q}_i\right) \\ &= \boldsymbol{q}_j^*\boldsymbol{v} - \sum_{i=1}^{n}(\boldsymbol{q}_i^*\boldsymbol{v})\boldsymbol{q}_j^*\boldsymbol{q}_i. \end{aligned}$$

Now, since Q is orthonormal we have

$$\boldsymbol{q}_j^* \boldsymbol{q}_i = \delta_{ij} = \begin{cases} 0, & i \neq j \\ 1, & i = j, \end{cases}$$

where δ_{ij} is referred to as the *Kronecker delta*. Therefore only one term in the summation survives (namely if $i = j$) and we have

$$\boldsymbol{q}_j^* \boldsymbol{r} = \boldsymbol{q}_j^* \boldsymbol{v} - (\boldsymbol{q}_j^* \boldsymbol{v}) 1 = 0.$$

Since $\boldsymbol{q}_j$ was arbitrary we have established that $\boldsymbol{r}$ is orthogonal to the entire set Q. ■

Remark If Q is a basis for $\mathbb{C}^m$ then $n = m$ and (since $\boldsymbol{v} \in \mathbb{C}^m$) $\boldsymbol{r} = \boldsymbol{0}$. Therefore we get the coordinates of $\boldsymbol{v}$ with respect to Q:

$$\boldsymbol{v} = \sum_{i=1}^{m} (\boldsymbol{q}_i^* \boldsymbol{v}) \boldsymbol{q}_i.$$

We now take a closer look at the projection idea. This will be very important later on. First $(\boldsymbol{q}_i^* \boldsymbol{v}) \boldsymbol{q}_i = \boldsymbol{q}_i (\boldsymbol{q}_i^* \boldsymbol{v})$ since $(\boldsymbol{q}_i^* \boldsymbol{v})$ is a scalar. Next, by the associativity of vector multiplication, this is also equal to $(\boldsymbol{q}_i \boldsymbol{q}_i^*) \boldsymbol{v}$. This latter expression is the product of a (rank-1) matrix and a column vector. The matrix $(\boldsymbol{q}_i \boldsymbol{q}_i^*)$ is known as a *projection matrix*.

Now we can represent the projections occurring in the decomposition of $\boldsymbol{v}$ either via a *sum of vector projections*

$$\sum_i (\boldsymbol{q}_i^* \boldsymbol{v}) \boldsymbol{q}_i$$

or by a sum of matrix projections

$$\sum_i (\boldsymbol{q}_i \boldsymbol{q}_i^*) \boldsymbol{v}.$$

Example Compute the orthogonal decomposition of $\boldsymbol{v} = [1, 1, 1]^*$ with respect to $\boldsymbol{q}_1 = [1/\sqrt{2}, 0, 1/\sqrt{2}]^*$, $\boldsymbol{q}_2 = [1/\sqrt{2}, 0, -1/\sqrt{2}]^*$, and $\boldsymbol{q}_3 = [0, 1, 0]^*$.

Since $\boldsymbol{v} \in \mathbb{R}^3$ and $\boldsymbol{q}_1, \boldsymbol{q}_2, \boldsymbol{q}_3$ are orthonormal they form a basis for $\mathbb{R}^3$ and we know that $\boldsymbol{r} = \boldsymbol{0}$.

1. In terms of vector projections we have

$$\begin{aligned} \boldsymbol{v} &= \sum_{i=1}^{3} (\boldsymbol{q}_i^* \boldsymbol{v}) \boldsymbol{q}_i \\ &= 2 \frac{1}{\sqrt{2}} \boldsymbol{q}_1 + 0 \boldsymbol{q}_2 + 1 \boldsymbol{q}_3 \\ &= \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}. \end{aligned}$$

Thus, the coordinates of $\boldsymbol{v}$ with respect to $\boldsymbol{q}_1, \boldsymbol{q}_2, \boldsymbol{q}_3$ are $(\sqrt{2}, 0, 1)$.

2. In terms of matrix projections we get

$$\begin{aligned}
\boldsymbol{v} &= \sum_{i=1}^{3} (\boldsymbol{q}_i \boldsymbol{q}_i^*)\boldsymbol{v} \\
&= \begin{bmatrix} 1/2 & 0 & 1/2 \\ 0 & 0 & 0 \\ 1/2 & 0 & 1/2 \end{bmatrix} \boldsymbol{v} + \begin{bmatrix} 1/2 & 0 & -1/2 \\ 0 & 0 & 0 \\ -1/2 & 0 & 1/2 \end{bmatrix} \boldsymbol{v} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \boldsymbol{v} \\
&= \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}.
\end{aligned}$$

1.2.5 Unitary and Orthogonal Matrices

A square matrix $Q \in \mathbb{C}^{m \times m}$ is called *unitary* (or *orthogonal* in the real case) if its columns are orthonormal, i.e., if $Q^*Q = I$. Equivalently, $Q^{-1} = Q^*$.

In particular, this implies

$$\boldsymbol{q}_i^* \boldsymbol{q}_j = \delta_{ij} = \begin{cases} 0, & i \neq j \\ 1, & i = j. \end{cases}$$

Recall that earlier we observed that if we multiply a given vector $\boldsymbol{b}$ (the coordinates of some point with respect to the standard basis) by A^{-1} then we obtain the coordinates $\boldsymbol{x}$ with respect to the column space of A. Conversely, if we multiply $\boldsymbol{x}$ by A, then we transform back to the standard basis.

Now, if we assume that A is a unitary matrix, i.e., $A = Q$ and $A^{-1} = Q^*$, then multiplication of $\boldsymbol{b}$ by Q^* yields the coordinates $\boldsymbol{x}$ with respect to the basis $\{Q(:,1), \ldots, Q(:,m)\} = \{\boldsymbol{q}_1, \ldots, \boldsymbol{q}_m\}$. Conversely, multiplication of $\boldsymbol{x}$ by Q transforms back to the standard basis $\{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_m\}$.

Example Take

$$\boldsymbol{b} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \quad \text{and} \quad Q = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 0 & 0 & 1 \\ 1/\sqrt{2} & -1/\sqrt{2} & 0 \end{bmatrix}$$

and note that $\boldsymbol{b} = \boldsymbol{v}$ and the columns of Q are given by $\boldsymbol{q}_1, \boldsymbol{q}_2, \boldsymbol{q}_3$ from the previous example.

Clearly,

$$Q^* = \begin{bmatrix} 1/\sqrt{2} & 0 & 1/\sqrt{2} \\ 1/\sqrt{2} & 0 & -1/\sqrt{2} \\ 0 & 1 & 0 \end{bmatrix},$$

and therefore

$$Q^*\boldsymbol{b} = \begin{bmatrix} \sqrt{2} \\ 0 \\ 1 \end{bmatrix}$$

are the coordinates with respect to the columns of Q.

Some properties of unitary matrices are collected in

Lemma 1.8 *Assume $Q \in \mathbb{C}^{m\times m}$ is unitary and $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{C}^m$. Then*

1. $(Q\boldsymbol{x})^*(Q\boldsymbol{y}) = \boldsymbol{x}^*\boldsymbol{y}$*, that is angles are preserved under unitary (orthogonal) transformations.*
2. $\|Q\boldsymbol{x}\| = \|\boldsymbol{x}\|$*, that is lengths are preserved under unitary (orthogonal) transformations.*
3. *All eigenvalues λ of Q satisfy $|\lambda| = 1$, and therefore* $\det Q = \pm 1$.

Remark The "+" in item 3 corresponds to rotations, and the "−" to reflections.

Proof We prove item 1:

$$(Q\boldsymbol{x})^*(Q\boldsymbol{y}) = \boldsymbol{x}^* \underbrace{Q^*Q}_{=I} \boldsymbol{y} = \boldsymbol{x}^*\boldsymbol{y}.$$

■

1.3 Norms

1.3.1 Vector Norms

Definition 1.9 *Let V be a vector space over $\mathbb{C}$. A* norm *is a function $\|\cdot\| : V \to \mathbb{R}_0^+$ which satisfies*

1. $\|\boldsymbol{x}\| \geq 0$ *for every $\boldsymbol{x} \in V$, and $\|\boldsymbol{x}\| = 0$ only if $\boldsymbol{x} = \boldsymbol{0}$.*
2. $\|\alpha\boldsymbol{x}\| = |\alpha|\|\boldsymbol{x}\|$ *for every $\boldsymbol{x} \in V$, $\alpha \in \mathbb{C}$.*
3. $\|\boldsymbol{x} + \boldsymbol{y}\| \leq \|\boldsymbol{x}\| + \|\boldsymbol{y}\|$ *for all $\boldsymbol{x}, \boldsymbol{y} \in V$ (triangle inequality).*

Example

1. $\|\boldsymbol{x}\|_1 = \sum_{i=1}^{m} |\boldsymbol{x}(i)|$, ℓ_1-norm.
2. $\|\boldsymbol{x}\|_2 = \left(\sum_{i=1}^{m} |\boldsymbol{x}(i)|^2 \right)^{1/2}$, ℓ_2-norm or Euclidean norm.
3. $\|\boldsymbol{x}\|_\infty = \max_{1\leq i\leq m} |\boldsymbol{x}(i)|$, ℓ_∞-norm, maximum norm or Chebyshev norm.
4. $\|\boldsymbol{x}\|_p = \left(\sum_{i=1}^{m} |\boldsymbol{x}(i)|^p \right)^{1/p}$, ℓ_p-norm.

It is interesting to consider the corresponding unit "spheres" for these three norms, i.e., the location of points in $\mathbb{R}^m$ whose distance to the origin (in the respective norm) is equal to 1. Figure 1 illustrates this for the case $m = 2$.

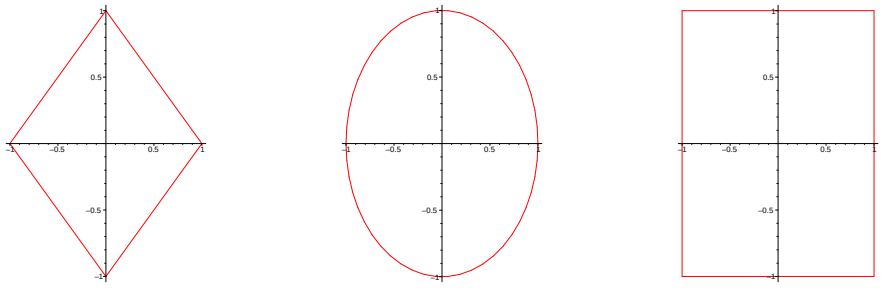

Figure 1: Unit "circles" in $\mathbb{R}^2$ for the ℓ_1, ℓ_2 and ℓ_∞ norms.

Sometimes one also wants to work with *weighted norms*. To this end one takes a diagonal weight matrix

$$W = \begin{bmatrix} \boldsymbol{w}(1) & 0 & \cdots & 0 \\ 0 & \boldsymbol{w}(2) & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \boldsymbol{w}(m) \end{bmatrix}$$

and then defines

$$\|\boldsymbol{x}\|_W = \|W\boldsymbol{x}\|.$$

Example A weighted p-norm is of the form

$$\|\boldsymbol{x}\|_{W,p} = \left(\sum_{i=1}^{m} |\boldsymbol{w}(i)\boldsymbol{x}(i)|^p \right)^{1/p}.$$

1.3.2 Matrix Norms

Definition 1.10 *If $\|\cdot\|_{(m)}$ and $\|\cdot\|_{(n)}$ are vector norms on $\mathbb{C}^m$ and $\mathbb{C}^n$, respectively, and $A \in \mathbb{C}^{m\times n}$, then the* induced matrix norm *(or* associated *or* subordinate matrix norm*) is defined by*

$$\|A\| = \sup_{\substack{\boldsymbol{x}\in\mathbb{C}^n \\ \|\boldsymbol{x}\|_{(n)}=1}} \|A\boldsymbol{x}\|_{(m)} = \sup_{\substack{\boldsymbol{x}\in\mathbb{C}^n \\ \boldsymbol{x}\neq\boldsymbol{0}}} \frac{\|A\boldsymbol{x}\|_{(m)}}{\|\boldsymbol{x}\|_{(n)}}.$$

Remark 1. The notation sup in Definition 1.10 denotes the *supremum* or least upper bound.

2. Often we can use the maximum instead of the supremum so that

$$\|A\| = \max_{\substack{\boldsymbol{x}\in\mathbb{C}^n \\ \boldsymbol{x}\neq\boldsymbol{0}}} \frac{\|A\boldsymbol{x}\|_{(m)}}{\|\boldsymbol{x}\|_{(n)}}.$$

3. One can show that $\|\cdot\|$ satisfies items (1)–(3) in Definition 1.9, i.e., it is indeed a norm.

4. $\|A\|$ can be interpreted as the maximum factor by which A can "stretch" $\boldsymbol{x}$.

Example The "stretch" concept can be understood graphically in $\mathbb{R}^2$. Consider the matrix

$$A = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$$

which maps $\mathbb{R}^2$ to $\mathbb{R}^2$.

1. By mapping the 1-norm unit circle under A we can see that the point that is maximally stretched is $(0, 1)$ which gets mapped into $(1, 1)$. Thus, a vector of 1-norm length 1 is mapped to a vector with 1-norm length 2, and $\|A\|_1 = 2$.

2. By mapping the 2-norm unit circle under A we can see (although this is much harder and requires use of the singular value decomposition) that the point that is maximally stretched is $(0.5257, 0.8507)$ which gets mapped into $(1.3764, 0.8507)$. Thus, a vector of 2-norm length 1 is mapped to a vector with 2-norm length 1.6180, and $\|A\|_2 = 1.6180$.

3. By mapping the ∞-norm unit circle under A we can see that the point that is maximally stretched is $(1, 1)$ which gets mapped into $(2, 1)$. Thus, a vector of ∞-norm length 1 is mapped to a vector with ∞-norm length 2, and $\|A\|_\infty = 2$.

How to Compute the Induced Matrix Norm

We now discuss how to compute the matrix norms induced by the popular p-norm vector norms. Strictly speaking we now would have to use two different subscripts on the vector norms (in addition to the index p also the (m) and (n) indicating the length of the vectors). In order to simplify notation we omit the second subscript which can be inferred from the context.

Consider an $m \times n$ matrix A. The most popular matrix norms can be computed as follows:

1.
$$\begin{aligned} \|A\|_1 &= \max_{1 \le j \le n} \|A(:, j)\|_1 \\ &= \max_{1 \le j \le n} \sum_{i=1}^{m} |A(i, j)|. \end{aligned}$$

   This gives rise to the name *maximum column sum norm.*

2. $\|A\|_2 = \max\limits_{1 \le j \le n} |\sigma_j|$, where σ_j is the j-th *singular value* of A (more later, fairly difficult to compute).

3.
$$\begin{aligned} \|A\|_\infty &= \max_{1 \le i \le m} \|A(i, :)\|_1 \\ &= \max_{1 \le i \le m} \sum_{j=1}^{n} |A(i, j)|. \end{aligned}$$

This gives rise to the name *maximum row sum norm.*

Example Let

$$A = \begin{bmatrix} 1 & 1 & 2 \\ 0 & 1 & 1 \\ 1 & 0 & 2 \end{bmatrix}.$$

Then $\|A\|_1 = 5$, $\|A\|_2 = 3.4385$, $\|A\|_\infty = 4$.

We now verify that the matrix norm induced by the ℓ_∞-norm is indeed given by the formula stated in item 3:

$$\|A\|_\infty = \sup_{\|\boldsymbol{x}\|_\infty=1} \|A\boldsymbol{x}\|_\infty = \sup_{\|\boldsymbol{x}\|_\infty=1} \max_{1\le i\le m} |(A\boldsymbol{x})(i)|.$$

By interchanging the supremum and maximum we obtain

$$\|A\|_\infty = \max_{1\le i\le m} \sup_{\|\boldsymbol{x}\|_\infty=1} |(A\boldsymbol{x})(i)|.$$

Next, we rewrite the matrix-vector product to get

$$\|A\|_\infty = \max_{1\le i\le m} \sup_{\|\boldsymbol{x}\|_\infty=1} |\sum_{j=1}^{n} A(i,j)\boldsymbol{x}(j)|.$$

Finally, using formula (2) below we obtain the desired result, i.e.,

$$\|A\|_\infty = \max_{1\le i\le m} \sum_{j=1}^{n} |A(i,j)|.$$

We now derive formula (2). Consider

$$|\sum_{j=1}^{n} A(i,j)\boldsymbol{x}(j)| = |A(i,1)\boldsymbol{x}(1) + A(i,2)\boldsymbol{x}(2) + \ldots + A(i,n)\boldsymbol{x}(n)|.$$

The supremum over all unit vectors (in the maximum norm) is attained if all terms in the above sum are positive. This can be ensured by picking $\boldsymbol{x}(j) = \text{sign}(A(i,j))$. But then we have

$$\sup_{\|\boldsymbol{x}\|_\infty=1} |\sum_{j=1}^{n} A(i,j)\boldsymbol{x}(j)| = |A(i,1)| + |A(i,2)| + \ldots + |A(i,n)| = \sum_{j=1}^{n} |A(i,j)|. \qquad (2)$$

1.3.3 Cauchy-Schwarz and Hölder Inequalities

Theorem 1.11 *Any two vectors $\boldsymbol{x}, \boldsymbol{y} \in V$ equipped with an inner product such that $\|\boldsymbol{x}\|^2 = \boldsymbol{x}^*\boldsymbol{x}$ satisfy the Cauchy-Schwarz inequality*

$$|\boldsymbol{x}^*\boldsymbol{y}| \le \|\boldsymbol{x}\|\|\boldsymbol{y}\|.$$

Proof The Cauchy-Schwarz inequality in a *real* vector space can be proved geometrically by starting out with the projection $\boldsymbol{p} = \frac{\boldsymbol{x}^*\boldsymbol{y}}{\|\boldsymbol{y}\|^2}\boldsymbol{y}$ of $\boldsymbol{x}$ onto the $\boldsymbol{y}$. Since $\boldsymbol{y}$ is in general not a unit vector we need to normalize by $\|\boldsymbol{y}\|$ here.

Now $\|\boldsymbol{x}-\boldsymbol{p}\|^2 \geq 0$ for any $\boldsymbol{x}, \boldsymbol{y}$ and we can use the definition of the norm properties of the inner product (in particular $\boldsymbol{x}^*\boldsymbol{y} = \boldsymbol{y}^*\boldsymbol{x}$ if $\boldsymbol{x}, \boldsymbol{y}$ real) to compute

$$\begin{aligned}
\|\boldsymbol{x}-\boldsymbol{p}\|^2 &= \left\|\boldsymbol{x} - \frac{\boldsymbol{x}^*\boldsymbol{y}}{\|\boldsymbol{y}\|^2}\boldsymbol{y}\right\|^2 \\
&= \left(\boldsymbol{x} - \frac{\boldsymbol{x}^*\boldsymbol{y}}{\|\boldsymbol{y}\|^2}\boldsymbol{y}\right)^* \left(\boldsymbol{x} - \frac{\boldsymbol{x}^*\boldsymbol{y}}{\|\boldsymbol{y}\|^2}\boldsymbol{y}\right) \\
&= \boldsymbol{x}^*\boldsymbol{x} - 2\frac{\boldsymbol{x}^*\boldsymbol{y}}{\|\boldsymbol{y}\|^2}\boldsymbol{x}^*\boldsymbol{y} + \left(\frac{\boldsymbol{x}^*\boldsymbol{y}}{\|\boldsymbol{y}\|^2}\right)^2 \boldsymbol{y}^*\boldsymbol{y} \\
&= \|\boldsymbol{x}\|^2 - 2\frac{\boldsymbol{x}^*\boldsymbol{y}}{\|\boldsymbol{y}\|^2}\boldsymbol{x}^*\boldsymbol{y} + \left(\frac{\boldsymbol{x}^*\boldsymbol{y}}{\|\boldsymbol{y}\|^2}\right)^2 \|\boldsymbol{y}\|^2 \\
&= \frac{\|\boldsymbol{x}\|^2\|\boldsymbol{y}\|^2 - 2\left(\boldsymbol{x}^*\boldsymbol{y}\right)^2 + \left(\boldsymbol{x}^*\boldsymbol{y}\right)^2}{\|\boldsymbol{y}\|^2} \\
&= \frac{\|\boldsymbol{x}\|^2\|\boldsymbol{y}\|^2 - \left(\boldsymbol{x}^*\boldsymbol{y}\right)^2}{\|\boldsymbol{y}\|^2}.
\end{aligned}$$

Remembering that this quantity is non-negative we get

$$\left(\boldsymbol{x}^*\boldsymbol{y}\right)^2 \leq \|\boldsymbol{x}\|^2\|\boldsymbol{y}\|^2$$

and taking square roots

$$|\boldsymbol{x}^*\boldsymbol{y}| \leq \|\boldsymbol{x}\|\|\boldsymbol{y}\|.$$

■

As a generalization of the Cauchy-Schwarz we have the *Hölder inequality*:

$$|\boldsymbol{x}^*\boldsymbol{y}| \leq \|\boldsymbol{x}\|_p\|\boldsymbol{y}\|_q,$$

where we allow any $1 \leq p, q \leq \infty$ such that $\frac{1}{p} + \frac{1}{q} = 1$.

Example We can apply the Cauchy-Schwarz inequality to compute the 2-norm of a rank-1 matrix $A = \boldsymbol{u}\boldsymbol{v}^*$ (cf. the projection matrices that came up earlier).

First, we note that

$$\begin{aligned}
\|A\boldsymbol{x}\|_2 &= \|(\boldsymbol{u}\boldsymbol{v}^*)\boldsymbol{x}\|_2 = \|\boldsymbol{u}\underbrace{(\boldsymbol{v}^*\boldsymbol{x})}_{\text{scalar}}\|_2 \\
&= |\boldsymbol{v}^*\boldsymbol{x}|\|\boldsymbol{u}\|_2 \\
&\leq \|\boldsymbol{u}\|_2\|\boldsymbol{v}\|_2\|\boldsymbol{x}\|_2 \qquad (3)
\end{aligned}$$

by the Cauchy-Schwarz inequality.

Now

$$\|A\|_2 = \sup_{\substack{\boldsymbol{x}\in\mathbb{C}^n \\ \boldsymbol{x}\neq\boldsymbol{0}}} \frac{\|A\boldsymbol{x}\|_2}{\|\boldsymbol{x}\|_2}$$

so that (3) gives us

$$\|A\|_2 \leq \|\boldsymbol{u}\|_2 \|\boldsymbol{v}\|_2.$$

However, in the special case $\boldsymbol{x} = \boldsymbol{v}$ we get

$$\|A\boldsymbol{x}\|_2 = \|A\boldsymbol{v}\|_2 = \|(\boldsymbol{u} \underbrace{\boldsymbol{v}^*)\boldsymbol{v}}_{\text{scalar}} \|_2 = |\boldsymbol{v}^*\boldsymbol{v}| \|\boldsymbol{u}\|_2 = \|\boldsymbol{v}\|_2^2 \|\boldsymbol{u}\|_2,$$

and therefore actually

$$\|A\|_2 = \sup_{\substack{\boldsymbol{x} \in \mathbb{C}^n \\ \boldsymbol{x} \neq \boldsymbol{0}}} \frac{\|A\boldsymbol{x}\|_2}{\|\boldsymbol{x}\|_2} = \|\boldsymbol{u}\|_2 \|\boldsymbol{v}\|_2.$$

1.3.4 Other Matrix Norms

There are also matrix norms that are not induced by vector norms.

Example The *Frobenius norm* of an $m \times n$ matrix A is given by

$$\|A\|_F = \left(\sum_{i=1}^{m} \sum_{j=1}^{n} |A(i,j)|^2 \right)^{1/2},$$

i.e., we interpret the matrix in $\mathbb{C}^{m \times n}$ as a vector in $\mathbb{C}^{mn}$ and compute its 2-norm.

Other formulas for the Frobenius norm are

$$\|A\|_F = \sqrt{\mathrm{tr}(A^*A)} = \sqrt{\mathrm{tr}(AA^*)},$$

where the *trace* $\mathrm{tr}(A)$ is given by the sum of the diagonal entries of A.

Finally,

Theorem 1.12 *Let $A \in \mathbb{C}^{m \times n}$ and $Q \in \mathbb{C}^{m \times m}$ be unitary. Then*

1. $\|QA\|_2 = \|A\|_2$,
2. $\|QA\|_F = \|A\|_F$,

i.e., both the 2-norm and the Frobenius norm are invariant under unitary transformation.

Proof The invariance of the matrix 2-norm is a direct consequence of the invariance of lengths of vectors under unitary transformations discussed earlier, i.e., $\|Q\boldsymbol{x}\|_2 = \|\boldsymbol{x}\|_2$. In particular, $\|QA\boldsymbol{x}\|_2 = \|A\boldsymbol{x}\|_2$, and so

$$\|QA\|_2 = \sup_{\substack{\boldsymbol{x} \in \mathbb{C}^n \\ \boldsymbol{x} \neq \boldsymbol{0}}} \frac{\|QA\boldsymbol{x}\|_2}{\|\boldsymbol{x}\|_2} = \sup_{\substack{\boldsymbol{x} \in \mathbb{C}^n \\ \boldsymbol{x} \neq \boldsymbol{0}}} \frac{\|A\boldsymbol{x}\|_2}{\|\boldsymbol{x}\|_2} = \|A\|_2.$$

For the Frobenius norm we have

$$\begin{aligned} \|QA\|_F &= \sqrt{\mathrm{tr}\left((QA)^*(QA)\right)} \\ &= \sqrt{\mathrm{tr}\left(A^*Q^*QA\right)} \\ &= \sqrt{\mathrm{tr}\left(A^*A\right)} = \|A\|_F \end{aligned}$$

since $Q^*Q = I$. ■

1.4 Computer Arithmetic

1.4.1 Floating-Point Arithmetic

We will use *normalized scientific notation* to represent real numbers $x \neq 0$, i.e., in decimal representation we write

$$x = \pm r \times 10^n, \qquad \frac{1}{10} \leq r < 1,$$

and in binary representation (which of course will matter on the computer) we write

$$x = \pm q \times 2^m, \qquad \frac{1}{2} \leq q < 1.$$

Both of these representations consist of the *sign* "$\pm$", the *mantissa* (either r or q), the *base* (either 10 or 2), and the *exponent* (either n or m).

Example Some examples of various floating-point numbers in normalized scientific notation with base 10:

$$\begin{aligned} 0.0000747 &= 0.747 \times 10^{-4} \\ 31.4159265 &= 0.314159265 \times 10^2 \\ 9,700,000,000 &= 0.97 \times 10^{10} \\ 1K &= 0.1024 \times 10^4 \quad \text{for computer stuff.} \end{aligned}$$

In order to study the kinds of errors that we can make when we represent real numbers as machine numbers we will use a *hypothetical binary computer*. We will assume that this computer can represent only positive numbers of the form

$$(0.d_1d_2d_3d_4)_2 \times 2^n$$

with $n \in \{-3, -2, -1, 0, 1, 2, 3, 4\}$. Our representation will actually have a 3-bit mantissa (i.e., the digit is always assumed to be 1, and therefore never stored). The set of choices for the exponent n comes from using a 3-bit exponent which allows us to generate $0, 1, \ldots, 7$, so that the n is actually determined as the values of $4 -$ exponent.

With this configuration we are able to generate $2^3 = 8$ different mantissas:

$$\begin{aligned} (0.1000)_2 &= (0.5)_{10} \\ (0.1001)_2 &= (0.5625)_{10} \\ (0.1010)_2 &= (0.625)_{10} \\ (0.1011)_2 &= (0.6875)_{10} \\ (0.1100)_2 &= (0.75)_{10} \\ (0.1101)_2 &= (0.8125)_{10} \\ (0.1110)_2 &= (0.875)_{10} \\ (0.1111)_2 &= (0.9375)_{10} \end{aligned}$$

for a total of 64 machine numbers (obtained by combining the 8 mantissa with the 8 possible exponents).

	$n=-3$	$n=-2$	$n=-1$	$n=0$	$n=1$	$n=2$	$n=3$	$n=4$
$(0.1000)_2$	0.0625	0.125	0.25	0.5	1	2	4	8
$(0.1001)_2$	0.0703125	0.140625	0.28125	0.5625	1.125	2.25	4.5	9
$(0.1010)_2$	0.078125	0.15625	0.3125	0.625	1.25	2.5	5	10
$(0.1011)_2$	0.0859375	0.171875	0.34375	0.6875	1.375	2.75	5.5	11
$(0.1100)_2$	0.09375	0.1875	0.375	0.75	1.5	3	6	12
$(0.1101)_2$	0.1015625	0.203125	0.40625	0.8125	1.625	3.25	6.5	13
$(0.1110)_2$	0.109375	0.21875	0.4375	0.875	1.75	3.5	7	14
$(0.1111)_2$	0.1171875	0.234375	0.46875	0.9375	1.875	3.75	7.5	15

Table 3: List of all 64 machine numbers for hypothetical computer.

Remark Any computer has finite word length (usually longer than that of our hypothetical computer) and can therefore represent only a discrete set on finite numbers exactly. Moreover, these numbers are distributed unevenly.

Example How does the computation of $\left(\frac{1}{10}+\frac{1}{5}\right)+\frac{1}{6}=\frac{7}{15}=0.4\overline{6}$ work out in our hypothetical computer?

First, we notice that we will be committing a number of *representation errors*. The closest machine number to $\frac{1}{10}$ is $0.1015625=(0.1101)_2\times 2^{-3}$. Similarly, the closest machine number to $\frac{1}{5}$ is $0.203125=(0.1101)_2\times 2^{-2}$.

We can add these two numbers (by shifting the mantissa of the first number) and obtain $(1.00111)_2\times 2^{-2}$. This, however, is not in normalized scientific notation. The "correct" representation of the intermediate calculation of $\frac{1}{10}+\frac{1}{5}$ is therefore $(0.100111)_2\times 2^{-1}$.

Now, however, our computer has only a 4-bit mantissa, and therefore we need to commit a *rounding error*, i.e., we represent the intermediate result by $(0.1010)_2\times 2^{-1}$. Note that this is indeed (fortunately so) the closest machine number to $\frac{3}{10}$.

For the final step of the calculation we need to represent $\frac{1}{6}$ as a machine number. We again commit a representation error by using $0.171875=(0.1011)_2\times 2^{-2}$. Adding this to the intermediate result from above (again by shifting the mantissa) we get $(0.11111)_2\times 2^{-1}$. Once more we need to round this result to the nearest machine number, so that the final answer of our calculation is $(0.1000)_2\times 2^0=0.5$ which is a rather poor representation of the true answer $\frac{7}{15}=0.4\overline{6}$.

In fact, the *absolute error* of our calculation is

$$|0.5-0.4\overline{6}|=0.0\overline{3},$$

and the *relative error* is

$$\left|\frac{0.5-0.4\overline{6}}{0.4\overline{6}}\right|\approx 0.0714 \quad \text{or } 7.14\%.$$

Example Another important observation is the fact that the *order of operations matters*, i.e., even though addition (or multiplication) is commutative and associative for real numbers, this may not be true for machine numbers.

Consider the problem of adding $1+\frac{1}{16}+\frac{1}{16}=\frac{9}{8}=1.125$ on our hypothetical computer. Note that all of these numbers are machine numbers (so we will be committing no representation errors).

a) We first represent 1 by $(0.1000)_2 \times 2^1$ and $\frac{1}{16}$ by $(0.1000)_2 \times 2^{-3}$. Addition of these 2 numbers leads to $(0.10001)_2 \times 2^1$ which now has to be rounded to a machine number, i.e., $(0.1000)_2 \times 2^1$. Clearly, adding another $\frac{1}{16}$ will not change the answer. Thus, $1 + \frac{1}{16} + \frac{1}{16} = 1$.

b) If we start by adding the smaller numbers first, then $\frac{1}{16} + \frac{1}{16}$ is represented by $(0.1000)_2 \times 2^{-3} + (0.1000)_2 \times 2^{-3} = (0.1000)_2 \times 2^{-2}$ (exactly), and adding $1 = (0.1000)_2 \times 2^0$ to this yields the correct answer of $(0.1001)_2 \times 2^1 = 1.125$

More common word lengths for the representation of floating-point numbers are 32-bit (single precision) and 64-bit (double precision). In the 32-bit representation the first bit is used to represent the sign, the next 8 bits represent the exponent, and the remaining 23 bits are used for the mantissa. This implies that the largest possible exponent is $(11111111)_2 = 2^8 - 1 = 255$ (or $-126, \ldots, 127$ where the two extreme cases 0 and 255 are reserved for special purposes). This means that we can roughly represent numbers between $2^{-126} \approx 10^{-38}$ and $2^{127} \approx 10^{38}$. Since the mantissa has 23 bits we can represent numbers with an accuracy (machine ε) of $2^{-23} \approx 0.12 \times 10^{-6}$, i.e., we can expect 6 accurate digits (which is known as single precision).

In the 64-bit system we use 11 bits for the exponent and 52 for the mantissa. This leads to machine numbers between $2^{-1022} \approx 10^{-308}$ and $2^{1023} \approx 10^{308}$. The resolution possible with the 52-bit mantissa is $2^{-52} \approx 0.22 \times 10^{-15}$. Thus double precision has 15 accurate digits.

1.4.2 Rounding and Chopping

Machine numbers can be obtained by either *rounding* (up or down to the nearest machine number), or by simply *chopping off* any extra digits.

Depending on the representation method used (rounding or chopping) any number will be represented internally with relative accuracy δ, where

$$\delta = \frac{fl(x) - x}{x} \quad \text{or } fl(x) = x(1 + \delta).$$

Here $|\delta| \leq \frac{1}{2}\beta^{1-n}$ for rounding and $|\delta| \leq \beta^{1-n}$ for chopping with representing the base (usually 2 for digital computers) and n the length of the mantissa.

Example A nice (actually quite disastrous) illustration of the difference of chopping versus rounding is given by the first few months of operations of the Vancouver Stock Exchange. In 1982 its index was initialized to a value of 1000. After that, the index was updated after every transaction. After 22 months the index had fallen to 520, and everyone was stumped, since "common sense" indicated *mild growth*.

The explanation was found when it was discovered that the updated values were *truncated* instead of rounded. A corrected update using *rounding* yielded an index values of 1098.892.

Some other examples of disasters due to careless use of computers can be found on the web at `http://www.ima.umn.edu/~arnold/disasters/`.

In order to have a "good" computer we would like

$$fl(x \odot y) = [x \odot y](1 + \delta)$$

for any basic arithmetic operation $\odot \in \{+, -, \cdot, /\}$. This is usually accomplished by using higher precision internally.

1.4.3 Loss of Significance

Consider the function $f(x) = x\left(\sqrt{x+1} - \sqrt{x}\right)$. We want to accurately compute $f(500)$. The true solution is $11.174755300747198\ldots$. This problem is studied in the Maple worksheet `477_577_loss_of_significance.mws`.

Using 6 digits (single precision) the detailed computations are

$$\begin{aligned} f(500) &= 500\left(\sqrt{501} - \sqrt{500}\right) \\ &= 500\,(22.3830 - 22.3607) \\ &= 500\,(0.0223) \\ &= 11.1500 \end{aligned}$$

with a relative error of

$$\left|\frac{11.15 - 11.1748}{11.1748}\right| \approx 0.22\%.$$

On the other hand, if we choose a better method to solve the problem (without disastrous subtractions), i.e., rewrite the function as

$$\begin{aligned} g(x) &= x\left(\sqrt{x+1} - \sqrt{x}\right)\frac{\sqrt{x+1} + \sqrt{x}}{\sqrt{x+1} + \sqrt{x}} \\ &= \frac{x(x+1-x)}{\sqrt{x+1} + \sqrt{x}} \\ &= \frac{x}{\sqrt{x+1} + \sqrt{x}} \end{aligned}$$

then

$$g(500) = \frac{500}{\sqrt{501} + \sqrt{500}} = \frac{500}{22.3830 + 22.3607} = \frac{500}{44.7437} = 11.1748,$$

which is exact up to the precision used.

We end with a theorem quantifying the loss of significant digits in subtractions.

Theorem 1.13 *If $x > y$ are positive normalized floating-point numbers in binary representation and $2^{-q} \leq 1 - \frac{y}{x} \leq 2^{-p}$ then the number ℓ of significant binary bits lost when computing $x - y$ satisfies $p \leq \ell \leq q$.*

Proof We show the lower bound, i.e., $\ell \geq p$ (the upper bound can be shown similarly).

Let $x = r \times 2^n$ and $y = s \times 2^m$ with $\frac{1}{2} \leq r, s < 1$. In order to perform the subtraction we rewrite (shift) y such that $y = (s \times 2^{m-n}) \times 2^n$. Then

$$x - y = \left[r - s \times 2^{m-n}\right] \times 2^n,$$

where

$$r - s \times 2^{m-n} = r\left(1 - \frac{s \times 2^m}{r \times 2^n}\right) = r\left(1 - \frac{y}{x}\right).$$

Now $r < 1$ and $1 - \frac{y}{x} \leq 2^{-p}$ by assumption, so that the mantissa satisfies $r - s \times 2^{m-n} < 2^{-p}$.

Finally, we need to shift at least p bits to the left in order to normalize the representation (we need $\frac{1}{2} \leq$ mantissa < 1). This introduces at least p (binary) zeros at the right end of the number, and so at least p bits are lost. ■

Remark If the theorem is formulated in base 10, i.e., if the relative error is between 10^{-q} and 10^{-p}, then between p and q digits are lost.