1 The Mathematical Minimum

In this section, we briefly discuss the minimum mathematical background that is required to fully understand this text. Readers who are familiar with the concepts may safely skip this section. Readers who are easily discouraged by even basic math may proceed to the next chapter and refer back here later.

1.1 Complex Numbers

Let us briefly recapitulate complex numbers. A complex number z is of the form

$$z = x + iy.$$

The x is called the real part of z; y is the imaginary part. The imaginary number i is defined as the solution to the equation:

$$x^2 + 1 = 0.$$

In other words, i is defined as the square root of -1. A complex number's conjugate, often denoted by $\bar{z}$ or z^*, is created by simply negating its imaginary part: $i \to -i$. For example, for $z = 5 + 2i$ the conjugate z^* would simply be $z^* = 5 - 2i$.

The conjugate of a product of complex numbers is equal to the product of the conjugates of the complex numbers:

$$(ab)^* = a^* b^*.$$

The *norm* of a complex number, denoted by $|z|$, is computed by multiplying the complex number with its conjugate:

$$|z|^2 = z^* z,$$
$$|z| = \sqrt{z^* z}.$$

Complex numbers can be drawn in the 2D plane with an x- and y-axis according to the definition. If we think of a complex number as a vector originating at (0, 0), the norm of a complex number, which is then the length of the corresponding vector, is a real number and can be computed using Pythagoras' theorem as:

$$|z| = |x + iy| = \sqrt{(x - iy)(x + iy)} = \sqrt{x^2 + y^2}$$

https://doi.org/10.1017/9781009099974.003 Published online by Cambridge University Press

For complex numbers, the norm is commonly referred to as the *modulus*. Note the difference between the *square* of a complex number and its squared *norm*. The square is computed as:

$$z^2 = (x + iy)^2 = (x + iy)(x + iy) = x^2 + 2ixy - y^2.$$

Complex exponentiation is defined by Euler's famous formula:

$$re^{i\phi} = r\big(\cos(\phi) + i\,\sin(\phi)\big).$$

Correspondingly, for complex numbers with norm $|z| = r = 1.0$:

$$z = e^{i\phi} = \cos(\phi) + i\,\sin(\phi).$$

The resulting complex numbers from this exponentiation are on a unit circle around the origin $(0, 0)$.

In Python, complex numbers are, conveniently, part of the language. Note however, that the imaginary i is written as a `j`, which is customarily used in electrical engineering. An example:

```
x = 1.0 + 0.5j
```

To conjugate, you can use the built-in `conjugate()` function for the complex data types or use `numpy`'s `conj()` function. For example:

```
x_conj = x.conjugate()    # or
x_conj = np.conj(x)
```

1.2 Dirac Notation, Bras, and Kets

In quantum computing, we think of qubits and states as column vectors of n complex numbers, where n is typically a power of 2. A vector with n elements is called n-dimensional. In the so-called *Dirac* notation, a column vector is called a *ket* and written as $|x\rangle$:

$$|x\rangle = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{n-1} \end{bmatrix}, \quad \text{with } x_i \in \mathbb{C} \text{ and } |x\rangle \in \mathbb{C}^n.$$

Remember that to *transpose* a matrix A, we take column i of A and make it row i of the transpose A^T, or $A^T_{ij} = A_{ji}$. The *Hermitian conjugate* of a column vector $|x\rangle$,

https://doi.org/10.1017/9781009099974.003 Published online by Cambridge University Press

denoted by a dagger $|x\rangle^\dagger$, is the transpose of the vector with each element conjugated. We write this vector as $\langle x|$, changing the direction of the angle bracket:

$$|x\rangle^\dagger = \langle x| = \begin{bmatrix} x_0^* & x_1^* & \dots & x_{n-1}^* \end{bmatrix}.$$

In Dirac notation, such a row vector $\langle x|$ is called a *bra* or the *dual vector* for a ket $|x\rangle$. Transposition and conjugation goes both ways – applying the transformation twice results in the original ket, a property called *involutivity*.

$$\begin{aligned} |x\rangle^\dagger &= \langle x|, \\ \langle x|^\dagger &= |x\rangle, \\ (|x\rangle^\dagger)^\dagger &= |x\rangle. \end{aligned}$$

There is potential for confusion around the conjugates: should the conjugates be denoted explicitly, via a^* or $a^\dagger$, as in $\langle x_0^* \; x_1^* \; \dots \; x_{n-1}^*|$, or is the fact that a vector has been converted from ket to bra sufficient? Typically, the conjugates are *not* marked explicitly.

1.2.1 Inner Product

The inner product, which is also called the *scalar product* or the *dot product*, is computed as a matrix product of a bra and a ket, which simplifies to the product between a row vector and a column vector – an element-wise vector-vector multiplication and summation. It is written in the following forms, with the dot $(\cdot)$ denoting a scalar product:

$$\langle x| \cdot |y\rangle = \langle x||y\rangle = \langle x|y\rangle.$$

For kets $|x\rangle$ and $|y\rangle$, the inner product is defined as:

$$|x\rangle = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{n-1} \end{bmatrix}, \quad \langle x| = \begin{bmatrix} x_0^* & x_1^* & \dots & x_{n-1}^* \end{bmatrix}, \quad |y\rangle = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{n-1} \end{bmatrix},$$

$$\langle x|y\rangle = x_0^* y_0 + x_1^* y_1 + \dots + x_{n-1}^* y_{n-1}.$$

The inner product is how vectors in this notation get their names. It forms a product of a bra and a ket, a *bra(c)ket*. Naming is difficult in general and quantum computing is no exception.

Note that $\langle x|y\rangle$ does not generally equal $\langle y|x\rangle$. For example, consider two kets $|x\rangle$ and $|y\rangle$:

$$|x\rangle = \begin{bmatrix} -1 \\ 2i \\ 1 \end{bmatrix}, \quad |y\rangle = \begin{bmatrix} 1 \\ 0 \\ i \end{bmatrix}. \tag{1.1}$$

https://doi.org/10.1017/9781009099974.003 Published online by Cambridge University Press

We construct the corresponding bras via transposition and negation of the imaginary parts:

$$\langle x| = \begin{bmatrix} -1 & -2i & 1 \end{bmatrix}, \ \langle y| = \begin{bmatrix} 1 & 0 & -i \end{bmatrix}.$$

We then compute the inner products:

$$\begin{aligned} \langle x|y\rangle &= -1 * 1 + 2i * 0 + i * 1 = -1 + i, \\ \langle y|x\rangle &= 1 * -1 + 0 * 2i - i * 1 = -1 - i. \end{aligned}$$

The second result is the conjugate of the first; the two inner products are different. This points to the important general rule:

$$\langle x|y\rangle^* = \langle y|x\rangle.$$

Two vectors are *orthogonal* if and only if their scalar product is zero. For 2D vectors, we visualize orthogonal vectors as perpendicular to each other:

$$\langle x|y\rangle = 0 \quad \Rightarrow x, y \ \text{ orthogonal.}$$

Similar to the way in which we compute the norm of a complex number, the norm of a vector is the scalar product of the vector with its dual vector. A vector is *normalized* if its norm is 1:

$$\big||x\rangle\big| = \langle x|x\rangle = 1 \quad \Rightarrow |x\rangle \ \text{ normalized.} \tag{1.2}$$

State vectors in quantum computing represent probability distributions that must total 1.0 by definition. Hence, normalized vectors play an important role in quantum computing.

1.2.2 Outer Product

Corresponding to the inner product, we can construct an *outer* product between two kets $|x\rangle$ and $|y\rangle$, denoted as:

$$|x\rangle\langle y| = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{n-1} \end{bmatrix} \begin{bmatrix} y_0^* & y_1^* & \dots & y_{n-1}^* \end{bmatrix} = \begin{bmatrix} x_0 y_0^* & x_0 y_1^* & \dots & x_0 y_{n-1}^* \\ x_1 y_0^* & x_1 y_1^* & \dots & x_1 y_{n-1}^* \\ \vdots & \vdots & \ddots & \vdots \\ x_{n-1} y_0^* & x_{n-1} y_1^* & \dots & x_{n-1} y_{n-1}^* \end{bmatrix}.$$

In the example given by Equation (1.1), $|x\rangle$ is a 3×1 vector and $|y\rangle$ is a 1×3 vector. By the rules of matrix multiplication, their outer product will be a 3×3 matrix. Again, if the vector elements are complex, we conjugate the vector elements when converting from bra to ket and vice versa.

https://doi.org/10.1017/9781009099974.003 Published online by Cambridge University Press

1.3 Tensor Product

To compute the *tensor product*[1] of two vectors, which can be either bras or kets, we use any of these notations:

$$|x\rangle \otimes |y\rangle = |x\rangle|y\rangle = |x,y\rangle = |xy\rangle. \tag{1.3}$$

And correspondingly:

$$\langle x| \otimes \langle y| = \langle x|\langle y| = \langle x,y| = \langle xy|.$$

In a tensor product, each element of the first constituent is multiplied with the whole of the second constituent. Hence, an $n \times m$ matrix tensored with an $k \times l$ matrix will result in an $nk \times ml$ matrix. For example, to compute the tensor products of the following two kets:

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix},$$

$$|0\rangle \otimes |1\rangle = |01\rangle = \begin{bmatrix} 1 & \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\ 0 & \begin{bmatrix} 0 \\ 1 \end{bmatrix} \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}.$$

You can see that the tensor product of two kets is a ket. Similarly, the tensor product of two bras is a bra, and the tensor product of two diagonal matrices is a diagonal matrix. Of course, tensor products are also defined for general matrices:

$$\begin{bmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{bmatrix} \otimes \begin{bmatrix} b_{00} & b_{01} \\ b_{10} & b_{11} \end{bmatrix} = \begin{bmatrix} a_{00}\begin{bmatrix} b_{00} & b_{01} \\ b_{10} & b_{11} \end{bmatrix} & a_{01}\begin{bmatrix} b_{00} & b_{01} \\ b_{10} & b_{11} \end{bmatrix} \\ a_{10}\begin{bmatrix} b_{00} & b_{01} \\ b_{10} & b_{11} \end{bmatrix} & a_{11}\begin{bmatrix} b_{00} & b_{01} \\ b_{10} & b_{11} \end{bmatrix} \end{bmatrix}$$

$$= \begin{bmatrix} a_{00}b_{00} & a_{00}b_{01} & a_{01}b_{00} & a_{01}b_{01} \\ a_{00}b_{10} & a_{00}b_{11} & a_{01}b_{10} & a_{01}b_{11} \\ a_{10}b_{00} & a_{10}b_{01} & a_{11}b_{00} & a_{11}b_{01} \\ a_{10}b_{10} & a_{10}b_{11} & a_{11}b_{10} & a_{11}b_{11} \end{bmatrix}.$$

For multiplication of scalars α and β with a tensor product, these rules apply:

$$\alpha(x \otimes y) = (\alpha x) \otimes y = x \otimes (\alpha y), \tag{1.4}$$

$$(\alpha + \beta)(x \otimes y) = \alpha x \otimes y + \beta x \otimes y. \tag{1.5}$$

A key property of the the tensor product is the following – it is used in many derivations in this text:

$$(A \otimes B)(a \otimes b) = (A \otimes a)(B \otimes b). \tag{1.6}$$

[1] Here, we are ignoring differences between the tensor product and the Kronecker product.

https://doi.org/10.1017/9781009099974.003 Published online by Cambridge University Press

The next rule is also very important for this text. Given two *composite* kets:

$$|\psi_1\rangle = |\phi_1\rangle \otimes |\chi_1\rangle \quad \text{and} \quad |\psi_2\rangle = |\phi_2\rangle \otimes |\chi_2\rangle,$$

the inner product between $|\psi_1\rangle$ and $|\psi_2\rangle$ is computed as:

$$\begin{aligned}\langle\psi_1|\psi_2\rangle &= \big(|\phi_1\rangle \otimes |\chi_1\rangle\big)^\dagger \big(|\phi_2\rangle \otimes |\chi_2\rangle\big) \\ &= \big(\langle\phi_1| \otimes \langle\chi_1|\big)\big(|\phi_2\rangle \otimes |\chi_2\rangle\big) \\ &= \langle\phi_1|\phi_2\rangle\langle\chi_1|\chi_2\rangle. \end{aligned} \tag{1.7}$$

1.4 Unitary and Hermitian Matrices

A square matrix A is *Hermitian* if it is equal to its transposed complex conjugate $A^\dagger$. Hence the diagonal elements must be real numbers, and the elements mirrored along the main diagonal are the complex conjugates of each other. For example:

$$A = A^\dagger = \begin{bmatrix} 1 & 3 + i\sqrt{2} \\ 3 - i\sqrt{2} & 0 \end{bmatrix}.$$

A square matrix A is *unitary* if its conjugate transpose is equal to its inverse, with $A^\dagger A = I$. Unitary matrices are *norm preserving* – multiplying a unitary matrix with a vector might change the vector's orientation but will not change its norm. For example, here the matrix Y is both unitary and Hermitian. The matrix S is unitary but *not* Hermitian:

$$Y = Y^\dagger = \begin{bmatrix} 0 & i \\ -i & 0 \end{bmatrix}, \quad \text{and} \quad S = \begin{bmatrix} 1 & 0 \\ 0 & e^i \end{bmatrix} \neq \begin{bmatrix} 1 & 0 \\ 0 & e^{-i} \end{bmatrix} = S^\dagger.$$

Similar to the way we computed Hermitian conjugates for vectors in Section 1.2, to construct the Hermitian *conjugate* of a square matrix, you have to transpose the matrix and conjugate its elements. A Hermitian conjugate is also called a Hermitian *adjoint*, or just adjoint for short. The terms adjoint and Hermitian conjugate are synonyms.

1.5 Hermitian Adjoint of Expressions

Here are the rules for how to conjugate expressions of matrices and vectors. We have already learned how to convert between bras and kets:

$$\begin{aligned} |\psi\rangle^\dagger &= \langle\psi|, \\ \langle\psi|^\dagger &= |\psi\rangle. \end{aligned}$$

To compute the adjoint of a matrix scaled by a complex factor:

$$(\alpha A)^\dagger = \alpha^* A^\dagger. \tag{1.8}$$

https://doi.org/10.1017/9781009099974.003 Published online by Cambridge University Press

For matrix-matrix products, the order reverses (this is an important rule used in this book):

$$(AB)^\dagger = B^\dagger A^\dagger. \tag{1.9}$$

And similarly, to compute the adjoint for products of matrices and vectors:

$$(A|\psi\rangle)^\dagger = \langle\psi|A^\dagger, \tag{1.10}$$
$$(AB|\psi\rangle)^\dagger = \langle\psi|B^\dagger A^\dagger. \tag{1.11}$$

For matrices in outer product notation, this rule is easy to derive:

$$A = |\psi\rangle\langle\phi| \quad \Rightarrow \quad A^\dagger = |\phi\rangle\langle\psi|. \tag{1.12}$$

And finally:

$$(A+B)^\dagger = A^\dagger + B^\dagger. \tag{1.13}$$

1.6 Eigenvalues and Eigenvectors

There is a special case of matrix-vector multiplication, where the following equation holds. Here, A is a square matrix, $|\psi\rangle$ is a ket, and λ is a simple (complex) scalar:

$$A|\psi\rangle = \lambda|\psi\rangle.$$

Applying A to the special vector $|\psi\rangle$ only *scales* the vector with a complex number, it does not change its orientation. We call λ an *eigenvalue* of A. There can be multiple eigenvalues for a given operator. The corresponding vectors for which this equation holds are called *eigenvectors*. In quantum mechanics, the synonym *eigenstates* is also used. Eigenvalues are allowed to be 0 by definition, but a null vector is not considered an eigenvector.

Diagonal matrices are a case for which finding the eigenvalues is trivial. Given a diagonal matrix of this form:

$$\begin{bmatrix} \lambda_0 & & & \\ & \lambda_1 & & \\ & & \ddots & \\ & & & \lambda_{n-1} \end{bmatrix},$$

we can pick the eigenvalues right off the diagonal. The corresponding eigenvectors are the computational bases $(1,0,0,\ldots)^T$, $(0,1,0,\ldots)^T$, and so on. For Hermitian matrices the eigenvalues are necessarily real.

https://doi.org/10.1017/9781009099974.003 Published online by Cambridge University Press

1.7 Trace of a Matrix

The *trace* of an $n \times n$ matrix A is defined as the sum of its diagonal elements:

$$\operatorname{tr}(A) = \sum_{i=0}^{n-1} a_{ii} = a_{00} + a_{11} + \cdots + a_{n-1n-1}.$$

Basic properties of the trace are the following, where c is a scalar, and A and B are square matrices:

$$\operatorname{tr}(A+B) = \operatorname{tr}(A) + \operatorname{tr}(B), \tag{1.14}$$

$$\operatorname{tr}(cA) = c\operatorname{tr}(A), \tag{1.15}$$

$$\operatorname{tr}(AB) = \operatorname{tr}(BA). \tag{1.16}$$

For tensor products, this important relation holds:

$$\operatorname{tr}(A \otimes B) = \operatorname{tr}(A)\operatorname{tr}(B). \tag{1.17}$$

The trace of a Hermitian matrix is real because the diagonal elements of a Hermitian are real. The trace of a matrix A is the sum of its n eigenvalues λ_i:

$$\operatorname{tr}(A) = \sum_{i=0}^{n-1} \lambda_i. \tag{1.18}$$

This next relation is important for measurements. Suppose we have two kets $|x\rangle$ and $|y\rangle$, such that

$$|x\rangle = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{n-1} \end{bmatrix} \quad \text{and} \quad |y\rangle = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{n-1} \end{bmatrix}.$$

The trace of the outer product of $|x\rangle$ and $\langle y|$ is equal to their inner product:

$$\operatorname{tr}(|x\rangle\langle y|) = \langle y|x\rangle. \tag{1.19}$$

This is easy to see from the outer product:

$$\begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{n-1} \end{bmatrix} \begin{bmatrix} y_0^* & y_1^* & \ldots & y_{n-1}^* \end{bmatrix} = \begin{bmatrix} x_0 y_0^* & x_0 y_1^* & \ldots & x_0 y_{n-1}^* \\ x_1 y_0^* & x_1 y_1^* & \ldots & x_1 y_{n-1}^* \\ \vdots & \vdots & \ddots & \vdots \\ x_{n-1} y_0^* & x_{n-1} y_1^* & \ldots & x_{n-1} y_{n-1}^* \end{bmatrix}$$

$$\implies \operatorname{tr}(|x\rangle\langle y|) = \sum_{i=0}^{n-1} x_i y_i^* = \langle y|x\rangle.$$

https://doi.org/10.1017/9781009099974.003 Published online by Cambridge University Press