Appendix A
Complex Numbers

A.1 Fields

A field is a set F together with two operations, called addition and multiplication and denoted by the usual symbolism, which satisfy the following conditions.

A1. $a + b = b + a$ for all $a, b \in \mathsf{F}$

A2. $(a + b) + c = a + (b + c)$ for all $a, b, c \in \mathsf{F}$

A3. There exists an element denoted by $0 \in \mathsf{F}$ such that $a + 0 = a$ for all $a \in \mathsf{F}$

A4. For each $a \in \mathsf{F}$ there exists an element $-a \in \mathsf{F}$ such that $a + (-a) = 0$

M1. $ab = ba$ for all $a, b \in \mathsf{F}$

M2. $(ab)c = a(bc)$ for all $a, b, c \in \mathsf{F}$

M3. There exists an element denoted by $1 \in \mathsf{F}$ such that $1a = a$ for all $a \in \mathsf{F}$

M4. For each $0 \neq a \in \mathsf{F}$ there exists an element $a^{-1} \in \mathsf{F}$ such that $aa^{-1} = 1$

D. $a(b + c) = ab + ac$ for all $a, b, c \in \mathsf{F}$

It can be shown that the additive identity 0 and the multiplicative identity 1 are unique in F. Also each element a has a unique additive inverse $-a$, and each $a \neq 0$ has a unique multiplicative inverse a^{-1}. The subtraction operation in F is defined in terms of addition as

$$a - b = a + (-b)$$

and the division operation is defined in terms of multiplication as

$$a/b = ab^{-1}\,, \quad b \neq 0$$

Familiar examples of fields are the field of rational numbers and the field of real numbers (denoted R). Another common one is the field of complex numbers explained next.

A.2 Complex Numbers

A complex number is of the form

$$z = a + ib$$

where $a, b \in \mathbf{R}$ and

$$i^2 = -1$$

a and b are called the real and imaginary parts of z, respectively, denoted

$$a = \operatorname{Re} z\,, \quad b = \operatorname{Im} z$$

Two complex numbers $z_1 = a_1 + ib_1$ and $z_2 = a_2 + ib_2$ are called equal if $a_1 = a_2$ and $b_1 = b_2$. The addition and multiplication of z_1 and z_2 are defined as

$$z_1 + z_2 = (a_1 + a_2) + i(b_1 + b_2)$$

and

$$z_1 z_2 = (a_1 a_2 - b_1 b_2) + i(a_1 b_2 + a_2 b_1)$$

Note that multiplication of two complex numbers is performed by the usual rules for algebraic multiplication with $i^2 = -1$.

Defining additive and multiplicative identities as

$$0 = 0 + i0\,, \quad 1 = 1 + i0$$

additive inverse of $z = a + ib$ as

$$-z = (-a) + i(-b)$$

and the multiplicative inverse as

$$z^{-1} = a/(a^2 + b^2) - ib/(a^2 + b^2)$$

it can be shown that the set of all complex numbers $\mathbf{C}$ together with the above addition and multiplication, is a field. Every real number can be considered as a complex number with imaginary part equal to 0, that is $a = a + i0$. Its additive inverse and multiplicative inverse (if $a \neq 0$) as a complex number are the same as its additive and multiplicative inverses as a real number. Thus $\mathbf{R}$, which is itself a field, is a subfield of $\mathbf{C}$ with respect to the same addition and multiplication operations.

The complex conjugate of $z = a + ib$ is defined to be

$$z^* = a - ib$$

Note that

$$zz^* = (a + ib)(a - ib) = a^2 + b^2$$

There is a geometrical representation of complex numbers. To a given complex number $z = a + ib$ we associate the point in a plane with abscissa a and ordinate b, relative to a rectangular coordinate system in the plane, as shown in Figure A.1. In this way there is a one-to-one correspondence between the set of all complex numbers and the set of all points in the plane. The absolute value or modulus of z, is defined as

$$|z| = \sqrt{zz^*} = (a^2 + b^2)^{1/2}$$

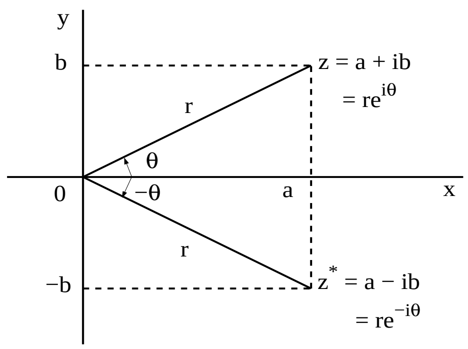

Figure A.1: Representation of a complex number

Geometrically, this is the polar distance r of the point (a, b) from the origin $(0, 0)$, that is, $|z| = r$. We also define the argument of $z \neq 0$, denoted arg z, to be the polar angle θ shown in the figure, that is

$$\arg z = \theta = \tan^{-1}(b/a)$$

Note that

$$z = r(\cos\theta + i\sin\theta)$$

Using the series representations

$$\begin{aligned}\cos\theta &= 1 - \theta^2/2! + \theta^4/4! - \cdots \\ \sin\theta &= \theta - \theta^3/3! + \theta^5/5! - \cdots\end{aligned}$$

and rearranging the terms we observe that

$$\cos\theta + i\sin\theta = 1 + (i\theta) + (i\theta)^2/2! + (i\theta)^3/3! + \cdots$$

By analogy to the series representation of the real quantity

$$e^x = 1 + x + x^2/2! + x^3/3! + \cdots$$

the above series can conveniently be defined as $e^{i\theta}$. Thus we obtain

$$z = r(\cos\theta + i\sin\theta) = re^{i\theta}$$

which is called the polar representation of the complex number z. Polar representation provides simplicity in multiplication and division of complex numbers. If $z_1 = r_1e^{i\theta_1}$ and $z_2 = r_2e^{i\theta_2}$, then

$$z_1z_2 = r_1r_2e^{i(\theta_1+\theta_2)}$$

and if $z_2 \neq 0$ $(r_2 \neq 0)$, then

$$z_1/z_2 = (r_1/r_2)e^{i(\theta_1-\theta_2)}$$

A.3 Complex-Valued Functions

If f and g are real-valued functions of a real variable t, then

$$h(t) = f(t) + ig(t)$$

defines a complex-valued function h of t. If f and g are differentiable on an interval $a < t < b$, then h is also differentiable, and its derivative is given by

$$h'(t) = f'(t) + ig'(t)$$

A useful complex-valued function is e^{zt}, where $z = a + ib$ is a complex number and t is a real variable. Using the polar representation, e^{zt} can be expressed as

$$e^{zt} = e^{(a+ib)t} = e^{at}e^{ibt} = e^{at}(\cos bt + i \sin bt)$$

Differentiating real and imaginary parts of e^{zt}, and rearranging the terms we get

$$\begin{aligned}
\frac{d}{dt}e^{zt} &= ae^{at}(\cos bt + i \sin bt) + e^{at}(-b \sin bt + ib \cos bt) \\
&= e^{at}(a \cos bt - b \sin bt) + ie^{at}(a \sin bt + b \cos bt) \\
&= (a + ib)e^{at}(\cos bt + i \sin bt) \\
&= ze^{zt}
\end{aligned}$$

Thus the usual differentiation property of the real-valued exponential function is generalized to the complex-valued exponential function.

Appendix B
Existence and Uniqueness Theorems

Consider a system of first order ordinary differential equations together with a set of initial conditions:

$$\mathbf{x}' = \mathbf{f}(\mathbf{x}, t)\,, \quad \mathbf{x}(t_0) = \mathbf{x}_o \tag{B.1}$$

where $\mathbf{f} : \mathsf{R}^{n\times 1} \times \mathsf{R} \to \mathsf{R}^{n\times 1}$ is a vector-valued function defined on some interval $\mathcal{I} \subset \mathsf{R}$ containing t_0. We assume that

a) for every fixed $\mathbf{x} \in \mathsf{R}^{n\times 1}$, the function $\mathbf{f}(\mathbf{x}, \cdot) : t \to \mathbf{f}(\mathbf{x}, t)$ is piecewise continuous on $\mathcal{I}$, and

b) $\mathbf{f}$ satisfies a **Lipschitz condition** on $\mathcal{I}$, that is, there exists a constant $K > 0$ such that [1]

$$\| \mathbf{f}(\mathbf{x}_1, t) - \mathbf{f}(\mathbf{x}_2, t) \| \le K \| \mathbf{x}_1 - \mathbf{x}_2 \| \tag{B.2}$$

for all $\mathbf{x}_1, \mathbf{x}_2 \in \mathsf{R}^{n\times 1}$ and $t \in \mathcal{I}$.

Recall that a vector-valued function $\boldsymbol{\phi}$ defined on $\mathcal{I}$ is called a solution of (B.1) if $\boldsymbol{\phi}(t_0) = \mathbf{x}_o$ and

$$\boldsymbol{\phi}'(t) = \mathbf{f}(\boldsymbol{\phi}(t), t) \tag{B.3}$$

at all continuity points of $\mathbf{f}$. Clearly, if $\boldsymbol{\phi}$ is a solution, then integrating both sides of (B.3) from t_0 to t, we obtain

$$\boldsymbol{\phi}(t) = \mathbf{x}_0 + \int_{t_0}^{t} \mathbf{f}(\boldsymbol{\phi}(\tau), \tau)\, d\tau \tag{B.4}$$

Conversely, if $\boldsymbol{\phi}$ satisfies the integral equation in (B.4), then it is a solution of (B.1). We will use this fact in the proof of the following existence and uniqueness theorem.

Theorem B.1 *Under the assumptions (a) and (b) above, the initial-value problem in (B.1) has a unique, continuous solution on $\mathcal{I}$.*

[1] The Lipschitz condition in (B.2) is stronger than continuity of $\mathbf{f}$ in $\mathbf{x}$. For example, the scalar function $f(x, t) = \sqrt{x}$ defined on $\mathcal{I} = [0, \infty)$ is continuous everywhere on $\mathcal{I}$, but does not satisfy a Lipschitz condition. With $x_1 = x$ and $x_2 = 0$, there exists no K that satisfies

$$\sqrt{x} \le Kx$$

for all $x \ge 0$.

Proof Define a sequence of continuous functions recursively as

$$\begin{aligned}
\boldsymbol{\phi}_0(t) &= \mathbf{x}_0 \\
\boldsymbol{\phi}_m(t) &= \mathbf{x}_0 + \int_{t_0}^{t} \mathbf{f}(\boldsymbol{\phi}_{m-1}(\tau), \tau)\, d\tau\,, \quad m = 1, 2, \ldots
\end{aligned} \tag{B.5}$$

Fix $T > 0$ such that $\mathcal{J} = [t_0, t_0 + T] \subset \mathcal{I}$. Since $\mathbf{f}(\mathbf{x}_0, t)$ is piecewise continuous, it is bounded on $\mathcal{J}$. Let

$$B = \max_{t \in \mathcal{J}} \{\, \mathbf{f}(\mathbf{x}_0, t) \,\}$$

We claim that

$$\|\, \boldsymbol{\phi}_m(t) - \boldsymbol{\phi}_{m-1}(t) \,\| \leq \frac{B}{K} \frac{K^m (t - t_0)^m}{m!}\,, \quad m = 1, 2, \ldots \tag{B.6}$$

for all $t \in \mathcal{J}$. The claim is true for $m = 1$ as

$$\|\, \boldsymbol{\phi}_1(t) - \boldsymbol{\phi}_0(t) \,\| \leq \|\, \int_{t_0}^{t} \mathbf{f}(\boldsymbol{\phi}_0(\tau), \tau)\, d\tau \,\| \leq \int_{t_0}^{t} \|\, \mathbf{f}(\mathbf{x}_0, \tau) \,\|\, d\tau \leq B(t - t_0)$$

Suppose it is true for $m = k$. Then for $m = k + 1$

$$\begin{aligned}
\|\, \boldsymbol{\phi}_{k+1}(t) - \boldsymbol{\phi}_k(t) \,\| &\leq \|\, \int_{t_0}^{t} [\, \mathbf{f}(\boldsymbol{\phi}_k(\tau), \tau) - \mathbf{f}(\boldsymbol{\phi}_{k-1}(\tau), \tau) \,]\, d\tau \,\| \\
&\leq \int_{t_0}^{t} K \,\|\, \boldsymbol{\phi}_k(\tau) - \boldsymbol{\phi}_{k-1}(\tau) \,\|\, d\tau \\
&\leq \int_{t_0}^{t} K \frac{B}{K} \frac{K^k (\tau - t_0)^k}{k!}\, d\tau \\
&\leq \frac{B}{K} \frac{K^{k+1} (t - t_0)^{k+1}}{(k+1)!}
\end{aligned}$$

so that it is also true for $m = k + 1$. Hence it is true for all $m \geq 1$. Since $t - t_0 \leq T$ for all $t \in \mathcal{J}$, (B.6) further implies that

$$\|\, \boldsymbol{\phi}_m(t) - \boldsymbol{\phi}_{m-1}(t) \,\| \leq \frac{B}{K} \frac{(KT)^m}{m!}\,, \quad m = 1, 2, \ldots$$

Define

$$\phi_m(t) = \|\, \boldsymbol{\phi}_m(t) - \boldsymbol{\phi}_0(t) \,\|$$

Then

$$\begin{aligned}
\phi_m(t) &= \|\, \sum_{k=1}^{m} [\, \boldsymbol{\phi}_k(t) - \boldsymbol{\phi}_{k-1}(t) \,] \,\| \;\leq\; \sum_{k=1}^{m} \|\, \boldsymbol{\phi}_k(t) - \boldsymbol{\phi}_{k-1}(t) \,\| \\
&\leq \frac{B}{K} \sum_{k=1}^{m} \frac{(KT)^k}{k!} \;\leq\; \frac{B}{K} (e^{KT} - 1)\,, \quad m = 1, 2, \ldots
\end{aligned}$$

for all $t \in \mathcal{J}$. This implies that the sequence of nonnegative-valued continuous functions $\{\, \phi_m \,\}$ converges uniformly on $\mathcal{J}$.[2] Consequently, the sequence of vector-valued continuous functions $\{\, \boldsymbol{\phi}_m \,\}$ converges uniformly to a continuous limit function $\boldsymbol{\phi}$.[3]

---

[2] This is a direct consequence of the comparison test. For details the reader is referred to a book on advanced calculus.

[3] That is, given any $\epsilon > 0$, there exist $M > 0$ such that

$$\|\, \boldsymbol{\phi}(t) - \boldsymbol{\phi}_m(t) \,\| \leq \epsilon$$

for all $m \geq M$ and for all $t \in \mathcal{J}$.

Uniform convergence of $\{ \boldsymbol{\phi}_m \}$, together with the Lipschitz condition on $\mathbf{f}$ further imply that

a) $$\lim_{m \to \infty} \mathbf{f}(\boldsymbol{\phi}_m(t), t) = \mathbf{f}(\boldsymbol{\phi}(t), t)$$

b) $$\lim_{m \to \infty} \int_{t_0}^{t} \mathbf{f}(\boldsymbol{\phi}_m(\tau), \tau)\, d\tau = \int_{t_0}^{t} \mathbf{f}(\boldsymbol{\phi}(\tau), \tau)\, d\tau$$

Thus

$$\begin{aligned}
\boldsymbol{\phi}(t) &= \lim_{m \to \infty} \boldsymbol{\phi}_m(t) \\
&= \mathbf{x}_o + \lim_{m \to \infty} \int_{t_0}^{t} \mathbf{f}(\boldsymbol{\phi}_{m-1}(\tau), \tau)\, d\tau \\
&= \mathbf{x}_o + \int_{t_0}^{t} \mathbf{f}(\boldsymbol{\phi}(\tau), \tau)\, d\tau
\end{aligned}$$

for all $t \in \mathcal{J}$, proving that $\boldsymbol{\phi}$ is a solution of (B.1) on $\mathcal{J}$.

To prove uniqueness of $\boldsymbol{\phi}$, suppose (B.1) has another solution $\boldsymbol{\psi}$ on $\mathcal{J}$. Define

$$g(t) = \| \boldsymbol{\phi}(t) - \boldsymbol{\psi}(t) \| = \| \int_{t_0}^{t} [\, \mathbf{f}(\boldsymbol{\phi}(\tau), \tau) - \mathbf{f}(\boldsymbol{\psi}(\tau), \tau) \,]\, d\tau \, \|$$

Then

$$g(t) \le \int_{t_0}^{t} \| \, \mathbf{f}(\boldsymbol{\phi}(\tau), \tau) - \mathbf{f}(\boldsymbol{\psi}(\tau), \tau) \, \| \, d\tau \le \int_{t_0}^{t} K g(\tau)\, d\tau$$

for all $t \in \mathcal{J}$. Let

$$h(t) = e^{-K(t-t_0)} \int_{t_0}^{t} K g(\tau)\, d\tau$$

Then $h(t_0) = 0$ and

$$h'(t) = K e^{-K(t-t_0)} [\, g(t) - \int_{t_0}^{t} K g(\tau)\, d\tau \,] \le 0$$

so that

$$h(t) \le 0 \quad \text{for all } \ t \in \mathcal{J}$$

Hence

$$0 \le g(t) \le \int_{t_0}^{t} K g(\tau)\, d\tau \le 0 \quad \text{for all } \ t \in \mathcal{J}$$

This implies $g(t) = 0$ for all $t \in \mathcal{J}$, or equivalently,

$$\boldsymbol{\phi}(t) = \boldsymbol{\psi}(t) \quad \text{for all } \ t \in \mathcal{J}$$

contradicting the assumption that $\boldsymbol{\phi}$ and $\boldsymbol{\psi}$ are two different solutions on $\mathcal{J}$. In conclusion, (B.1) has a unique solution on $\mathcal{J}$.

The case $t < t_0$ can be analyzed similarly by considering a closed interval $\mathcal{J} = [t_0 - T, T] \subset \mathcal{I}$. Since T is arbitrary in both cases, it follows that (B.1) has a unique, continuous solution on $\mathcal{I}$.

The functions in (B.5) that converge to the solution of (B.1) are known as the **Picard iterates**, and provide a constructive method for the proof of the existence of a solution. The proof of the uniqueness part of the theorem is a variation of the well-known **Bellman-Gronwal Lemma**.

Proof of Theorem 6.1

The proof follows immediately from Theorem B.1 on noting that

$$\mathbf{f}(\mathbf{x}, t) = A(t)\mathbf{x} + \mathbf{u}(t)$$

is piecewise continuous for every fixed $\mathbf{x} \in \mathbf{R}^{n\times 1}$, and satisfies a Lipschitz condition with

$$K = \sup_{t\in\mathcal{I}} \| A(t) \|$$

Appendix C
The Laplace Transform

C.1 Definition and Properties

The one-sided (or unilateral) **Laplace transform** of a real- or complex-valued function f of a real variable t is a complex-valued function F of a complex variable s, defined by

$$F(s) = \int_0^\infty f(t)e^{-st}\,dt \tag{C.1}$$

provided the integral converges. For convenience, the Laplace transform of f is also denoted by $\mathcal{L}\{f\}$.

Let f be a piece-wise continuous function that is bounded by an exponential, that is, there exist $M, \alpha \in \mathbf{R}$ such that

$$|f(t)| \leq Me^{\alpha t}$$

holds for all t. Such a function is said to be of **exponential order** α. Then for any $s = \sigma + i\omega \in \mathbf{C}$ with $\sigma > \alpha$

$$\begin{aligned} \left|\int_0^\infty f(t)e^{-st}\,dt\right| &\leq \int_0^\infty |f(t)e^{-st}|\,dt \\ &\leq \int_0^\infty Me^{(\alpha-\sigma)t}\,dt \\ &\leq \frac{M}{\sigma-\alpha} \end{aligned}$$

and thus the integral in (C.1) converges. The region

$$\mathcal{C}_\alpha = \{\, s = \sigma + i\omega \mid \sigma > \alpha \,\}$$

is called the **region of convergence** of F.

Let f be a function of exponential order α with a Laplace transform $F(s)$ that exists in a region $\mathcal{C}_\alpha$, and suppose that $f(t) = 0$ for $t < 0$. Then f can be obtained uniquely from F by means of a line integral as

$$f(t) = \lim_{\omega \to \infty} \int_\Gamma F(s)e^{st}\,ds \tag{C.2}$$

where Γ is a vertical straight line in $\mathcal{C}_\alpha$ extending from $s = \sigma - i\omega$ to $s = \sigma + i\omega$. The integral on the right-hand side of (C.2) is called the **inverse Laplace transform** of F, denoted by $\mathcal{L}^{-1}(F)$.[1] We use the notation

$$f(t) \longleftrightarrow F(s)$$

to indicate that f and F are a Laplace transform-inverse Laplace transform pair.

Some useful properties of the Laplace transform are stated below, where it is assumed that the Laplace transforms involved exist.

a) Linearity

$$af(t) + bg(t) \longleftrightarrow aF(s) + bG(s)$$

b) Shift

$$f(t - t_0) \longleftrightarrow e^{-st_0} F(s)\,, \quad t_0 > 0$$

$$e^{s_0 t} f(t) \longleftrightarrow F(s - s_0)\,, \quad s_0 \in \mathbf{C}$$

c) Scaling

$$f(at) \longleftrightarrow \frac{1}{a} F\left(\frac{s}{a}\right), \quad a > 0$$

d) Differentiation

$$f^{(n)}(t) \longleftrightarrow s^n F(s) - s^{n-1} f(0) - \cdots - s f^{(n-2)}(0) - f^{(n-1)}(0)$$

$$t^n f(t) \longleftrightarrow (-1)^n \frac{d^n}{ds^n} F(s)$$

The first three of the properties above are direct consequences of the definition. For example, the Laplace transform of the shifted function $f(t - t_0)$ is

$$\begin{aligned} \int_0^\infty f(t - t_0) e^{-st}\, dt &= \int_{-t_0}^\infty f(\tau) e^{-s(\tau + t_0)}\, d\tau \\ &= e^{-st_0} \int_0^\infty f(\tau) e^{-s\tau}\, d\tau \;=\; e^{-st_0} F(s) \end{aligned}$$

proving the first property in (b).[2] Proofs of the properties in (d) require some manipulations. Evaluating the integral in (C.1) written for f' by parts, we obtain

$$\begin{aligned} \mathcal{L}\{f'\} &= \int_0^\infty f'(t) e^{-st}\, dt \\ &= [\, f(t) e^{-st}\,]_{t=0}^{t=\infty} + \int_0^\infty f(t) s e^{-st}\, dt \\ &= sF(s) - f(0) \end{aligned}$$

[1] In practice, the line integral in (C.2) is seldom used to find the inverse Laplace transform. Instead, Laplace transform tables are used for most of the functions of interest.

[2] The second equality follows from the assumption that $f(t) = 0$ for $t < 0$.

proving the first property in (d) for $n = 1$.[3] The case $n > 1$ and the second property in (d) can be proved similarly.

Example C.1

The Laplace transform of the unit step function

$$u(t) = \begin{cases} 1, & t > 0 \\ 0, & t < 0 \end{cases}$$

is

$$U(s) = \int_0^t e^{-st}\,dt = [\,-se^{-st}\,]_{t=0}^{t=\infty} = \frac{1}{s}\ , \quad \operatorname{Re} s > 0$$

By property (b),

$$u(t - t_0) \;\longleftrightarrow\; \frac{e^{-st_0}}{s}\ , \quad \operatorname{Re} s > 0$$

and, by property (d)

$$tu(t) \;\longleftrightarrow\; -\frac{d}{ds}\,(\,\frac{1}{s}\,) = \frac{1}{s^2}$$

C.2 Some Laplace Transform Pairs

The Laplace transform of the unit step function obtained in Example C.1, together with the properties listed in the previous section, allows us to obtain the Laplace transform of many useful functions. For example, from the second property in (b), we obtain

$$e^{\sigma_0 t}u(t) \;\longleftrightarrow\; \frac{1}{s - \sigma_0}$$

and from property (d),

$$te^{\sigma_0 t}u(t) \;\longleftrightarrow\; \frac{1}{(s - \sigma_0)^2}$$

The Laplace transform of $e^{s_0 t}u(t)$, in turn, can be used to find Laplace transforms of sine and cosine functions. On noting that

$$e^{i\omega_0 t}u(t) = \cos\omega_0 t + i\sin\omega_0 t$$

we obtain

$$(\cos\omega_0 t + i\sin\omega_0 t)u(t) \;\longleftrightarrow\; \frac{1}{s - i\omega_0} = \frac{s + i\omega_0}{s^2 + \omega_0^2}$$

Thus

$$(\cos\omega_0 t)u(t) \;\longleftrightarrow\; \frac{s}{s^2 + \omega_0^2}$$

[3]Since f is exponential order α and $\operatorname{Re} s > \alpha$

$$\lim_{t \to \infty} f(t)e^{-st} = 0$$

and

$$(\sin\omega_0 t)u(t) \;\longleftrightarrow\; \frac{\omega_0}{s^2+\omega_0^2}$$

A list of some Laplace transform pairs, which can be derived similarly, is given in Table C.1.

C.3 Partial Fraction Expansions

A function F that is expressed as a ratio of two polynomials is called a **rational function**. A rational function

$$F(s) = \frac{c(s)}{d(s)} = \frac{c_0 s^m + c_1 s^{m-1} + \cdots + c_m}{s^n + d_1 s^{n-1} + \cdots + d_n} \tag{C.3}$$

is said to be **proper** if $m \le n$ and **strictly proper** if $m < n$.

The Laplace transforms in Table C.1 are simple strictly proper rational functions with denominators being first or second degree polynomials or powers of such polynomials. This observation suggests that if a rational function F can be expressed as a linear combination of such simple rational functions, then by linearity of the Laplace transform, the inverse Laplace transform of F can be obtained as the same linear combination of the inverse Laplace transforms of individual functions, which can be written down directly from the table. For example, the inverse Laplace transform of

$$\frac{s+2}{s^2+s} = \frac{2}{s} - \frac{1}{s+1}$$

can be written down using Table C.1 as

$$\mathcal{L}^{-1}\{\frac{s+2}{s^2+s}\} = (2 - e^{-t})u(t)$$

Consider a strictly proper rational function $F(s)$ expressed as in (C.3). Suppose that the denominator polynomial $d(s)$ is factored out as

$$d(s) = \prod_{i=1}^{k} (s - p_i)^{n_i}$$

where $p_i \in \mathbf{C}$ are distinct zeros of d with multiplicities n_i, $i = 1, \ldots, k$. p_i are called the **poles** of F. Then F can be expressed as

$$F(s) = \sum_{i=1}^{k} \sum_{j=1}^{n_i} \frac{r_{ij}}{(s-p_i)^j} \tag{C.4}$$

where $r_{ij} \in \mathbf{C}$. This expression is known as the **partial fraction expansion** of F. The coefficients r_{ij} can be obtained by collecting the terms on the right-hand side of (C.4) over the common denominator d and equating the coefficients of the resulting numerator polynomial to those of c.

MATLAB provides a built-in function, `residue`, to compute p_i and r_{ij}. The commands

Table C.1: Some Laplace transform pairs

$f(t)$	$F(s)$
1	$\dfrac{1}{s}$
t^n	$\dfrac{n!}{s^{n+1}}$
$e^{\sigma_0 t}$	$\dfrac{1}{s-\sigma_0}$
$t^n e^{\sigma_0 t}$	$\dfrac{n!}{(s-\sigma_0)^{n+1}}$
$\cos\omega_0 t$	$\dfrac{s}{s^2+\omega_0^2}$
$\sin\omega_0 t$	$\dfrac{\omega_0}{s^2+\omega_0^2}$
$t\cos\omega_0 t$	$\dfrac{s^2-\omega_0^2}{(s^2+\omega_0^2)^2}$
$t\sin\omega_0 t$	$\dfrac{2\omega_0 s}{(s^2+\omega_0^2)^2}$
$e^{\sigma_0 t}\cos\omega_0 t$	$\dfrac{s-\sigma_0}{(s-\sigma_0)^2+\omega_0^2}$
$e^{\sigma_0 t}\sin\omega_0 t$	$\dfrac{\sigma_0}{s-\sigma_0)^2+\omega_0^2}$
$te^{\sigma_0 t}\cos\omega_0 t$	$\dfrac{(s-\sigma_0)^2-\omega_0^2}{((s-\sigma_0)^2+\omega_0^2)^2}$
$te^{\sigma_0 t}\sin\omega_0 t$	$\dfrac{2\omega_0(s-\sigma_0)}{((s-\sigma_0)^2+\omega_0^2)^2}$

```
>> c=[c0 c1 ...  cm]; d=[1 d1 ...  dn];
>> [r,p]=residue(c,d);
```

return the poles p_i in the array `p` (with each pole appearing as many times as its multiplicity) and the coefficients r_{ij} in the array `r`.

Example C.2

The strictly proper rational function

$$F(s) = \frac{2s^2 + 4s + 1}{s^3 + 4s^2 + 5s + 2} = \frac{2s^2 + 4s + 1}{(s+2)(s+1)^2}$$

has the poles $p_1 = -2$ with $n_1 = 1$ and $p_2 = -1$ with $n_2 = 2$. Hence F has a partial fraction expansion of the form

$$F(s) = \frac{r_1}{s+2} + \frac{r_{21}}{s+1} + \frac{r_{22}}{(s+1)^2}$$

Reorganizing the above expression, we get

$$\begin{aligned} F(s) &= \frac{r_1(s+1)^2 + r_{21}(s+1)(s+2) + r_{22}(s+2)}{(s+2)(s+1)^2(s+2)} \\ &= \frac{(r_1 + r_{21})s^2 + (2r_1 + 3r_{21} + r_{22})s + (r_1 + 2r_{21} + 2r_{22})}{s^3 + 4s^2 + 5s + 2} \\ &= \frac{2s^2 + 4s + 1}{s^3 + 4s^2 + 5s + 2} \end{aligned}$$

Equating the coefficients of the numerators of the last two expressions we obtain a system of three equatins in three unknows

$$\begin{bmatrix} 1 & 1 & 0 \\ 2 & 3 & 1 \\ 1 & 2 & 2 \end{bmatrix} \begin{bmatrix} r_1 \\ r_{21} \\ r_{22} \end{bmatrix} = \begin{bmatrix} 2 \\ 4 \\ 1 \end{bmatrix}$$

the unique solution of which is easily computed as $r_1 = r_{12} = 1, r_{22} = -1$.

Alternatively, the MATLAB commands

```
>> c=[2 4 1]; d=[1 4 5 2];
>> [r,p]=residue(c,d);
```

produce

$$r = [1 \quad 1 \quad -1], \quad p = [-2 \quad -1 \quad -1]$$

Note that the coefficients r_{ij} associated with a multiple pole p_i appear in the order of increasing j in the array `r`.

Thus

$$F(s) = \frac{1}{s+2} + \frac{1}{s+1} - \frac{1}{(s+1)^2}$$

and its inverse Laplace transform is

$$f(t) = (e^{-2t} + e^{-t} - te^{-t})u(t)$$

Example C.3

To find the inverse Laplace transform of

$$G(s) = \frac{s^2 + 9s + 16}{s^3 + 5s^2 + 9s + 5}$$

we execute the MATLAB commands

```
>> c=[1 9 16]; d=[1 5 9 5];
>> [r,p]=residue(c,d)
```

which compute

$$r = [\,-1.5 - i \quad -1.5 + i \quad 4\,]\,, \quad p = [\,-2 + i \quad -2 - i \quad -1\,]$$

Thus

$$G(s) = \frac{-1.5 - i}{s + 2 - i} + \frac{-1.5 + i}{s + 2 + i} + \frac{4}{s + 1}$$

and

$$\begin{aligned} g(t) &= (-1.5 - i)\, e^{(-2+i)t} + (-1.5 + i)\, e^{(-2-i)t} + 4\, e^{-t} \\ &= 2\,\mathrm{Re}\,\{\, (-1.5 - i)\, e^{(-2+i)t}\,\} + 4\, e^{-t} \\ &= e^{-2t}(2 \sin t - 3 \cos t) + 4\, e^{-t} \end{aligned}$$

C.4 Solution of Differential Equations by Laplace Transform

Consider an nth order linear differential equation with constant coefficients

$$y^{(n)} + a_1 y^{(n-1)} + \cdots + a_{n-1} y' + a_n y = u(t) \tag{C.5}$$

together with a set of n initial conditions

$$y(0) = y_0\,, \quad y'(0) = y_1\,, \quad \ldots\,, \quad y^{(n-1)}(0) = y_{n-1}$$

specified at $t_0 = 0$. Taking the Laplace transform of both sides of (C.5) and using the differentiation property, we obtain

$$\begin{array}{rcl} s^n Y(s) - s^{n-1} y_0 - \cdots - s y_{n-2} - y_{n-1} & + & \\ s^{n-1} Y(s) - \cdots - s y_{n-3} - y_{n-2} & + & \\ & \vdots & \\ sY(s) - y_0 & + & \\ Y(s) & = & U(s) \end{array}$$

Rearranging the terms, the above expression can be written as

$$Y(s) = \frac{b(s)}{a(s)} + \frac{1}{a(s)}\, U(s) \tag{C.6}$$

where

$$\begin{aligned} a(s) &= s^n + a_1 s^{n-1} + \cdots + a_n \\ b(s) &= y_0 s^{n-1} + (y_0 + y_1) s^{n-2} + \cdots + (y_0 + y_1 + \cdots + y_{n-1}) \end{aligned}$$

Taking the inverse Laplace transform of (C.6), the solution of the given initial-value problem is obtained as

$$y = y_o(t) + y_u(t)\,, \quad t \geq 0 \tag{C.7}$$

where

$$y_o(t) = \mathcal{L}^{-1}\{\, \frac{b(s)}{a(s)} \,\}$$

is part of the solution due to the initial conditions, and

$$y_u(t) = \mathcal{L}^{-1}\{\, \frac{1}{a(s)}\, U(s) \,\}$$

is the part due to the forcing term. Note the similarity between the expressions in (2.15) and (C.7).

Example C.4

Consider the differential equation

$$y'' + 2y' + 26y = 26u(t)\,, \quad y(0) = y'(0) = 0$$

where $u(t)$ is the unit step function.

Taking the Laplace transforms of both sides of the given differential equation and rearranging the terms, we get

$$Y(s) \;=\; \Phi(s) = \frac{26}{s(s^2 + 2s + 26)} \tag{C.8}$$

Expanding $Y(s)$ into partial fractions, we obtain

$$\begin{aligned} Y(s) &= \frac{26}{s(s+1-5i)(s+1+5i)} \\ &= \frac{1}{s} + \frac{-0.5+0.1\,i}{s+1-5i} + \frac{-0.5-0.1\,i}{s+1+5i} \end{aligned}$$

Thus

$$\begin{aligned} y \;=\; \phi(t) &= 1 + (-0.5+0.1\,i)\,e^{(-1+5i)t} + (-0.5-0.1\,i)\,e^{(-1-5i)t} \\ &= 1 + 2\,\mathrm{Re}\,\{\,(-0.5+0.1\,i)\,e^{(-1+5i)t}\,\} \\ &= 1 - e^{-t}(\cos 5t + 0.2\,\sin 5t)\,, \quad t \geq 0 \end{aligned} \tag{C.9}$$

the plot of which is shown in Figure C.1.

If the initial conditions were specified as $y(0) = 1, y'(0) = 0$, then the Laplace transform would yield

$$s^2 Y(s) - s + 2sY(s) - 2 + 26Y(s) = \frac{26}{s}$$

or equivalently,

$$Y(s) = \frac{1}{s}$$

Then the solution would be

$$y = 1\,, \quad t \geq 0$$

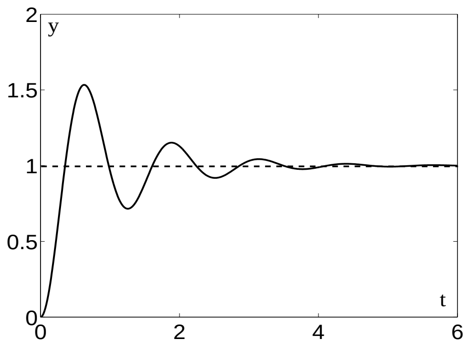

Figure C.1: Solution of the DE in Example C.4

Example C.5

Consider the same differential equation in the previous example with a different forcing function:

$$y'' + 2y' + 26y = 26f(t)\,, \quad y(0) = y'(0) = 0$$

where

$$f(t) = \begin{cases} 1, & 0 < t < 2 \\ 0, & t < 0 \text{ or } t > 2 \end{cases}$$

is a pulse of unit strength extending from $t = 0$ to $t = 2$.

Observing that

$$f(t) = u(t) - u(t-2)$$

we have

$$F(s) = U(s) - e^{-2s}U(s) = \frac{1 - e^{-2s}}{s}$$

Then the Laplace transform of the solution is obtained as

$$Y(s) = \frac{26(1 - e^{-2s})}{s(s^2 + 2s + 26)} = (1 - e^{-2s})\Phi(s)$$

where $\Phi(s)$ is given by (C.8). Taking the inverse Laplace transform, we compute the solution as

$$\begin{aligned} y &= \phi(t)u(t) - \phi(t-2)u(t-2) \\ &= \begin{cases} \phi(t), & 0 < t < 2 \\ \phi(t) - \phi(t-2), & t > 2 \end{cases} \\ &= \begin{cases} 1 - e^{-t}(\cos 5t + 0.2\,\sin 5t), & 0 < t < 2 \\ -e^{-t}(\cos 5t + 0.2\,\sin 5t) + \\ \qquad e^{-(t-2)}(\cos 5(t-2) + 0.2\,\sin 5(t-2)), & t > 2 \end{cases} \end{aligned}$$

The solution is plotted in Figure C.2.

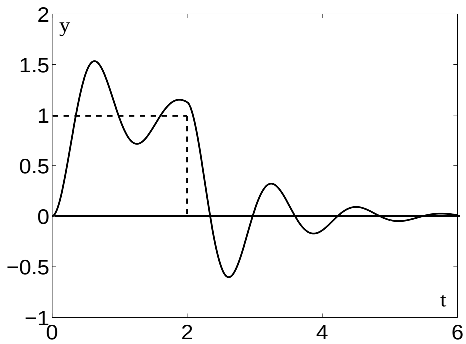

Figure C.2: Solution of the DE in Example C.5

The Laplace transform can also be used to solve systems of differential equations provided we properly define the Laplace transform of a vector-valued function. This is fairly straightforward: We define the Laplace transform of

$$\mathbf{f}(t) = \mathrm{col}\,[\, f_1(t), \, \ldots \, , f_n(t)\,]$$

element-by-element as

$$\mathbf{F}(s) = \mathcal{L}\{\,\mathbf{f}(t)\,\} = \mathrm{col}\,[\, F_1(s), \, \ldots \, , F_n(s)\,]$$

With this definition it is easy to prove that all the properties of the Laplace transform are also valid for the vector case. For example

$$A\mathbf{f}(t) + B\mathbf{g}(t) \;\longleftrightarrow\; A\mathbf{F}(s) + B\mathbf{G}(s)$$

and

$$\mathbf{f}'(t) \;\longleftrightarrow\; s\mathbf{F}(s) - \mathbf{f}(0)$$

Consider a SLDE with a constant coefficient matrix:

$$\mathbf{x}' = A\mathbf{x} + \mathbf{u}(t)\,, \quad \mathbf{x}(0) = \mathbf{x}_o \tag{C.10}$$

Taking the Laplace transform of both sides, we obtain

$$s\mathbf{X}(s) - \mathbf{x}_o = A\mathbf{X}(s) + \mathbf{U}(s)$$

which can be solved for $\mathbf{X}(s)$ as

$$\mathbf{X}(s) = (sI - A)^{-1}\mathbf{x}_o + (sI - A)^{-1}\mathbf{U}(s) \tag{C.11}$$

Then the solution is

$$\mathbf{x} = \mathcal{L}^{-1}\{\,(sI - A)^{-1}\,\}\mathbf{x}_o + \mathcal{L}^{-1}\{\,(sI - A)^{-1}\mathbf{U}(s)\,\} = \Phi_o(t) + \Phi_u(t)$$

When $u(t) = 0$, i.e., (C.10) is homogeneous, the solution expression above reduces to

$$\mathbf{x} = \mathcal{L}^{-1}\{\,(sI - A)^{-1}\,\}\mathbf{x}_o$$

Comparing the above expression with (6.19) we observe that

$$\mathcal{L}^{-1}\{\,(sI-A)^{-1}\,\}=e^{At}$$

We thus obtain an alternative formula to compute the matrix exponential function e^{At}.

Example C.6

Consider again Example 6.4, where

$$(sI-A)^{-1}=\begin{bmatrix} s+3 & 2 \\ 1 & s+2 \end{bmatrix}^{-1}=\frac{1}{(s+1)(s+4)}\begin{bmatrix} s+2 & -2 \\ -1 & s+3 \end{bmatrix}$$

We can compute e^{At} by taking the inverse Laplace transform of the elements of $(sI-A)^{-1}$ after expanding each of them into its partial fractions. However, a more elegant approach is to expand the matrix rational function $(sI-A)^{-1}$ into its partial fractions as

$$\begin{aligned}(sI-A)^{-1} &= \frac{1}{s+1}R_1+\frac{1}{s+4}R_2 \\ &= \frac{1}{(s+1)(s+4)}\left[(s+4)R_1+(s+1)R_2\right] \\ &= \frac{1}{(s+1)(s+4)}\left[(R_1+R_2)s+(4R_1+R_2)\right]\end{aligned}$$

Comparing the numerator polynomial matrices of the two expressions for $(sI-A)^{-1}$, we get

$$\begin{aligned} R_1+R_2 &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \\ 4R_1+R_2 &= \begin{bmatrix} 2 & -2 \\ -1 & 3 \end{bmatrix}\end{aligned}$$

from which we obtain

$$R_1=\frac{1}{3}\begin{bmatrix} 1 & -2 \\ -1 & 2 \end{bmatrix},\quad R_2=\frac{1}{3}\begin{bmatrix} 2 & 2 \\ 1 & 1 \end{bmatrix}$$

Thus

$$e^{At}=e^{-t}R_1+e^{-4t}R_2=\frac{1}{3}\begin{bmatrix} e^{-t}+2e^{-4t} & -2e^{-t}+2e^{-4t} \\ -e^{-t}+e^{-4t} & 2e^{-t}+e^{-4t} \end{bmatrix}$$

and the solution corresponding to the given initial condition $\mathbf{x}_o=\mathrm{col}\,[\,1,2\,]$ is

$$\mathbf{x}=e^{At}\mathbf{x}_o=\begin{bmatrix} -e^{-t}+2e^{-4t} \\ e^{-t}+e^{-4t} \end{bmatrix}$$

which is the same as the one obtained in Example 6.4.

Of course, we can obtain the solution corresponding to a given initial condition directly without computing e^{At}. By computing $\mathbf{X}(s)$ and expanding it into partial fractions as

$$\begin{aligned}\mathbf{X}(s) &= (sI-A)^{-1}\mathbf{x}_o = \frac{1}{(s+1)(s+4)}\begin{bmatrix} s+2 & -2 \\ -1 & s+3 \end{bmatrix}\begin{bmatrix} 1 \\ 2 \end{bmatrix} \\ &= \frac{1}{(s+1)(s+4)}\begin{bmatrix} s-2 \\ 2s+5 \end{bmatrix} = \frac{1}{s+1}\begin{bmatrix} -1 \\ 1 \end{bmatrix}+\frac{1}{s+4}\begin{bmatrix} 2 \\ 1 \end{bmatrix}\end{aligned}$$

we get the same solution.

Appendix D
A Brief Tutorial on MATLAB

MATLAB is an interactive system and a programming language for general scientific and technical computation. When it is invoked (either by clicking on the Matlab icon or by typing the command `matlab` from the keyboard) the command prompt `>>` appears indicating that MATLAB is ready to accept command from the keyboard. Commands are terminated by "return" or "enter" keys. The `exit` or `quit` command ends MATLAB.

D.1 Defining Variables

The basic data element of MATLAB is a matrix that does not require dimensioning. Scalars and arrays (vectors) are treated as special matrices. A variable is a data element with a name, which can be any combination of upper and lowercase letters, digits and underscores, starting with a letter and length not exceeding 19. Variables are case sensitive, so `A` and `a` are different variables.

Variables are assigned numerical values by typing an expression or a formula or a function that utilizes arithmetic operations on numerical data or previously defined variables. For example, the commands

```
>> a=2+7; b=4*a;
>> c=sqrt(b);
```

assign the values 9, 36 and 6 to the variables `a`, `b` and `c`, respectively; and the command

```
>> c=c/3;
```

reassigns `c` the value 2. Note that more than one command, separated by commas or semicolons, may appear on a single line. When a command is not terminated by a semicolon the result of the operation is echoed on the screen:

```
>> d=sin(pi/c)
d =
     1
```

If the result of an operation is not assigned to a variable, it is assigned to a default variable `ans` (short for "answer"):

```
>> a+sqrt(-b)
ans =
  9.0000+6.0000i
```

The last example shows that MATLAB requires no special handling of complex numbers. In fact, the imaginary unit `i` is one of the special variables of MATLAB. Some others are `j` (same as `i`), `ans` (default variable name), `pi` (π), `eps` (smallest number such that when added to 1 creates a floating-point number greater than 1), `inf` (∞) and `NaN` (not a number, e.g., 0/0). It is recommended that these variables should not be used as variable names to avoid changing their values.

A matrix is defined by entering its elements row by row as

```
>> A=[1 2 3 4; 3 4 5 6; 5 6 7 8]
A =
     1     2     3     4
     3     4     5     6
     5     6     7     8
```

where elements in each row are separated by spaces (or commas), and the rows by semicolons. Thus the commands

```
>> x=[2 4 6 8]; y=[-3; 2; -1];
```

define a row vector `x`, and a column vector `y`. In particular, the command

```
start:increment:end
```

generates a row vector (an array) of equally spaced values with the values of `start` and `end` specifying the first and the last elements of the array. If the increment is omitted, it is assumed to be 1. Thus

```
>> arry=-3:2:9
arry =
    -3    -1     1     3     5     7     9
```

Note also:

```
>> B(1,2)=7, B(2,4)=2
B =
     0     7
B =
     0     7     0     0
     0     0     0     2
```

The command `size(A)` returns a 1×2 row vector consisting of the number of rows and the number of columns of `A`. For a row or column vector `x`, the command `length(x)` returns the number of elements of the vector. Thus

```
>> size(A), size(arry), length(arry)
ans =
     3     4
ans =
     1     7
ans =
     7
```

A particular element of a matrix (or a row or column vector) can be extracted as

```
>> a23=A(2,3), x3=x(3), x3new=x(1,3), y2=y(2)
a23 =
     5
x3 =
     6
```

```
x3new =
     6
y2 =
     2
```

To extract a submatrix of a matrix, the rows and columns of the submatrix are specified:

```
>> sub1=A(2,3:4), sub2=A([1 3],[2 3]), sub3=A(2,:)
sub1 =
     5     6
sub2 =
     2     3
     6     7
sub3 =
     3     4     5     6
```

Thus `sub1` consists of row 2 and columns 3 and 4 of `A`, `sub2` rows 1 and 3 and columns 2 and 3, and `sub3` row 2 and all columns. Similarly,

```
>> part=arry(2:5)
part =
    -1     1     3     5
```

Conversely, a matrix can be constructed from smaller blocks:

```
>> A1=[x;0:3], A2=[A y A(:,[3 1])]
A1 =
     2     4     6     8
     0     1     2     3
A2 =
     1     2     3     4    -3     3     1
     3     4     5     6     2     5     3
     5     6     7     8    -1     7     5
```

MATLAB has special commands for generating special matrices: `eye(n)` generates an identity matrix of order `n`, `zeros(m,n)` an $m \times n$ zero matrix, `ones(m,n)` an $m \times n$ matrix with all elements equal to 1. If `d` is a row or column matrix of length n, the command `diag(d)` generates an $n \times n$ diagonal matrix with the elements of `d` appearing on the diagonal; and if `A` is an $m \times n$ matrix, `diag(A)` gives a column vector of the diagonal elements of `A`.

All commands entered and variables defined in a session are stored in MATLAB's workspace, and can be recalled at any time. Typing the name of a variable returns its value:

```
>> A1
A1 =
     2     4     6     8
     0     1     2     3
```

The command `who` gives a list of all variables defined.

```
>> who
Your variables are:
```

```
A       a       c       sub2    x3new
A1      a23     d       sub3    y
A2      arry    part    x       y2
B       b       sub1    x3
```

The command `clear v_name_1 v_name_2` clears the variables `v_name_1` and `v_name_2` from the workspace, and `clear` clears all variables.

The command `save fn` saves the workspace in the binary file `fn.mat`, which can later be retrieved by the `load fn` command. Menu items *Save Workspace As...* and *Load Workspace* in the *File* menu serve the same purpose.

D.2 Arithmetic Operations

MATLAB utilizes the following arithmetic operators: `+` (addition), `-` (subtraction), `*` (multiplication), `^` (power operator), `'` (transpose), `/` and `\` (right and left division).

These operators work on scalars or matrices:

```
>> i*sub1, (sub1-[8 5])*sub2
ans =
       0+5.0000i        0+6.0000i
ans =
     0    -2
```

where `*` denotes a scalar multiplication in the first command and matrix multiplication in the second. However,

```
>> sub2*sub1
???  Error using ==> *
Inner matrix dimensions must agree.
```

which indicates that the matrices are not compatible for the product.

Although addition of matrices requires that the matrices be of the same order, for convenience MATLAB also allows for addition of a scalar and a matrix by first enlarging the scalar to the size of the matrix. Thus

```
>> sub2+2
ans =
     4     5
     8     9
```

that is, `sub2+2` is equivalent to `sub2+2*ones(2,2)`.

The power operator requires a square matrix as operand:

```
>> C=[0 i; -i 0]^3
C =
       0                0+1.0000i
       0-1.0000i        0
```

The transpose operator takes the Hermitian adjoint of a matrix, which reduces to transpose when the matrix is real. Thus

```
>> [1-i;2+i]'
ans =
  1.0000+1.0000i   2.0000-1.0000i
```

Right division operator / works as usual when both operands or the divisor is a scalar:

```
>> C/5
ans =
       0                  0+0.2000i
       0-0.2000i          0
```

However, care must be taken when "dividing" two matrices: The command A/B calculates a matrix Y such that A=YB. Obviously, this requires that A and B have the same number of columns. If the equation is inconsistent then Y is a least-squares solution. Thus

```
>> [0 2]/[1 2; 3 4]
ans =
     3    -1
```

calculates the exact solution of the consistent equation

$$[\,0 \;\; 2\,] = Y \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

and

```
>> [2 4]/[6 2]
ans =
   0.5000
```

calculates a least-squares solution of the inconsistent equation

$$[\,2 \;\; 4\,] = Y\,[\,6 \;\; 2\,]$$

Similarly, the command A\B (left division) calculates a matrix X such that AX=B, provided that A and B have the same number of rows. Again, if the equation is inconsistent then X is a least-squares solution. Thus

```
>> [1 2; 3 4]\[0 2]
ans =
   2.0000
  -1.0000
```

Note that A\B = (B'/A')'.

MATLAB also provides array versions of the above arithmetic operators that allow for element-by-element operations on arrays (row or column vectors). If x and y are arrays of the same length, then x.*y generates an array whose elements are obtained by multiplying corresponding elements of x and y. Array versions of right and left division and the power operator are defined similarly. For example,

```
>> x=[1 2 3]; y=[4 5 6];
>> x.*y
ans =
     4    10    18
>> x./y, x.\y
ans =
   0.2500  0.4000  0.5000
```

```
ans =
  4.0000  2.5000  2.0000
>> x./2, 2./x, x.\2
ans =
  0.5000  1.0000  1.5000
ans =
  2.0000  1.0000  0.6667
ans =
  2.0000  1.0000  0.6667
>> x.^y, y.^x
ans =
     1    32   729
ans =
     4    25   216
```

Array version of transpose operator takes the transpose (without conjugate) so that

```
>> [1-i; 2+3i].'
ans =
  1.0000-1.0000i   2.0000+3.0000i
```

D.3 Built-In Functions

MATLAB provides a number of elementary math functions that operate on individual elements of matrices. Among them are trigonometric, inverse trigonometric, hyperbolic, inverse hyperbolic, exponential, natural and common logarithmic functions, square root, absolute value, angle, conjugate, real and imaginary parts of complex numbers. For example,

```
>> u=(pi/4)*[1 -3];
>> v=sin(u)
v =
  0.7071 -0.7071
>> w=sqrt(v)
w =
  0.8409                0+0.8409i
>> z=exp(w)
z =
  2.3184           0.6668+0.7452i
>> ang=(180/pi)*angle(w)
ang =
     0    90
```

MATLAB also provides many useful matrix functions, some of which are summarized below.

```
>> A=[1 2 3 4; 2 3 4 5; 3 5 7 9];
```

```
>> rank(A)
ans =
     2
>> rref(A)            % reduced row echelon form
ans =
     1     0    -1    -2
     0     1     2     3
     0     0     0     0
>> norm(A,1), norm(A,2), norm(A,inf)      % p norms
ans =
    18
ans =
 15.7403
ans =
    24
>> % singular value decomposition:  A=USV'
>> [U,S,V]=svd(A)
U =
  0.3472  0.7390  0.5774
  0.4664 -0.6702  0.5774
  0.8136  0.0688 -0.5774
S =
 15.7403       0       0       0
       0  0.4921       0       0
       0       0  0.0000       0
V =
  0.2364 -0.8026  0.3025 -0.4566
  0.3914 -0.3831 -0.0629  0.8343
  0.5465  0.0364 -0.7815 -0.2987
  0.7016  0.4558  0.5420 -0.0790
```

The following matrix functions operate on square matrices:

```
>> A=[0 -3 1; 1 4 -2; 1 2 0];
>> det(A)
ans =
     4
>> inv(A)
ans =
  1.0000  0.5000  0.5000
 -0.5000 -0.2500  0.2500
 -0.5000 -0.7500  0.7500
>> % LU decomposition:  L*U=P*A
>> [L,U,P]=lu(A)
L =
```

```
  1.0000       0       0
       0  1.0000       0
  1.0000  0.6667  1.0000
U =
  1.0000  4.0000 -2.0000
       0 -3.0000  1.0000
       0       0  1.3333
P =
     0     1     0
     1     0     0
     0     0     1
>> % modal matrix and diagonal form:  A*P=P*D
>> [P,D]=eig(A)
P =
  0.5000+0.5000i   0.5000-0.5000i   0.7845
       0-0.5000i        0+0.5000i  -0.5883
       0-0.5000i        0+0.5000i  -0.1961
D =
  1.0000+1.0000i        0               0
       0           1.0000-1.0000i       0
       0                0          2.0000
```

Note that `eig` command computes the linearly independent eigenvectors of `A` but not the generalized eigenvectors. If A is not diagonalizable the `P` matrix will not be a modal matrix.

D.4 Programming in MATLAB

D.4.1 Flow Control

MATLAB commands that control flow of execution based on decision making are similar to those of most programming languages and are briefly summarized below.

For–End Structure

The general structure of a `for` loop is

```
for x=matrix
   commands
end
```

where the `commands` between the `for` and `end` statements are executed once for each column of the `matrix` with `x` assigned the value of the corresponding column. Usually `matrix` is an array, and `x` is a scalar. For example,

```
n=input('Enter n = ')
fact=1;
for k=1:n
   fact=fact*k;
```

```
end
```

calculates the factorial of `n`.

`For` loops can be nested as desired.

While–End Structure

The general structure of a `while` loop is

```
while expression
   commands
end
```

The `commands` between the `while` and `end` statements are executed as long as the `expression` is `True`. The `expression` may include the relational operators `>`, `<`, `>=`, `<=`, `==` (equal) and `~=` (not equal), and/or logical operators `&` (AND), `|` (OR) and `~` (NOT). As an example,

```
>> n=1; x=1; series=1;
>> while x>0.000001
      series=series+x;
      n=n+1;
      x=x/n;
   end
>> format long
>> series
series =
   2.71828152557319
```

calculates e using the McLaurin series

$$\exp(x) = \sum_{n=0}^{\infty} \frac{x^n}{n!}$$

for $x = 1$, truncated when $x^n/n! < 0.000001$.

A mistake in the expression controlling a `while` loop may result in a never-ending loop. For example, if the second command above is mistakenly typed as `while x>0` then the loop never ends. Such a run-away loop can be broken by [CTRL-C] keys.

If–End Structure and Variations

`If` structures allow for control of the flow of execution based on simple decision making. The basic structure of the `if` command is

```
if expression
   commands
end
```

where `commands` are executed if `expression` is `True` and skipped otherwise. The variation

```
if expression_1
   commands_1
```

```
elseif expression_2
   commands_2
...
elseif expression_k
   commands_k
else
   commands_last
end
```

allows for a choice among several sets of commands.

A `break` command within an if structure can be used to terminate a loop prematurely. As an example

```
>> x=1; series=1;
>> for n=1:1000
      if x<0.000001
         break
      end
      x=x/n; series=series+x;
   end
```

is equivalent to the sequence of commands in the `while` loop example.

D.4.2 M-Files

Rather than being typed on the keyboard, a sequence of MATLAB commands can be placed in a text file with an extension `.m`, which are then executed upon typing the name of the file at command prompt. Such a file is called a script file, or an m-file referring to its extension. A script file can be created by selecting the *M-file* option of the menu item *New* under the *File* menu, or by using any text editor. When a valid variable name is typed at command prompt, MATLAB first checks if it is the name of a current variable or a built-in command, and if not, looks for an m-file with that name. If such a file exists, the commands in it are executed as if they were typed in response to `>>` prompts.

The `input` command in an M-file allows the user to type a value from the keyboard to be assigned to a variable. As an example, suppose that the following set of commands are stored in the M-file `myfactorial.m`

```
n=input('Enter n = ')
fact=1;
for k=1:n
   fact=fact*k;
end
fact
```

When the command `myfactorial` is typed at the `>>` prompt, MATLAB starts executing the commands in the file starting with the first command, which types the prompt

```
Enter n =
```

and waits for the user to type an integer, which is assigned to the variable `n`. The program ends after the value of `fact`, computed by the `for` loop, is echoed on the screen. A typical session would be

```
>> myfactorial
Enter n = [5]
fact =
   120
```

where `[5]` denotes the number entered by the user (followed by a return). Of course, the program can be refined to provide suitable error messages when the keyboard entry is not an admissible input.

D.4.3 User Defined Functions

Each of MATLAB's built-in functions is a sequence of commands which operate on the variables passed to it, compute the required results, and pass those results back. For example, the function

```
[L,U,P]=lu(A)
```

accepts as input a square matrix `A`, computes its LU decomposition, and passes back the results in the matrices `L`, `U` and `P`. The commands executed by the function as well as any intermediate variables created by those commands are hidden.

MATLAB provides a structure for creating user-defined functions in the form of a text M-file. The general structure of a user-defined function is

```
[vo_1,...,vo_k]=fname(vi_1,...,vi_m)
commands
```

where `fname` is a user given name of the function and `commands` is a set of MATLAB commands evaluated to compute the output variables `vo_1,...,vo_k` using the input variables `vi_1,...,vi_m`. A single output variable need not be enclosed in brackets. The text of the function must be saved with the same name as the function itself and with an extension .m, i.e., as `fname.m`.

As an example, the following function finds the largest `k` elements of an array `v` and returns them in an array `u`.

```
function u=mymax(v,k)
for p=1:k
   [w,ind]=max(v);
   u(p)=w;
   v(ind)=-inf;
end
```

Its use is illustrated below:

```
>> u=[-7:3:5];
>> x=mymax(u,3)
x =
     5     2    -1
```

Note that the function `mymax` uses the built-in MATLAB function `max`, which finds the maximum element in an array and its position in the array. It should also be noted that unlike an M-file, a function does not interfere with MATLAB's workspace; it has its own separate workspace.

D.5 Simple Plots

The `plot` command of MATLAB plots an array against another of the same length:

```
>> t=0:0.01:2; x=cos(2*pi*t);
>> plot(t,x)
```

produces the graph in Figure D.1.

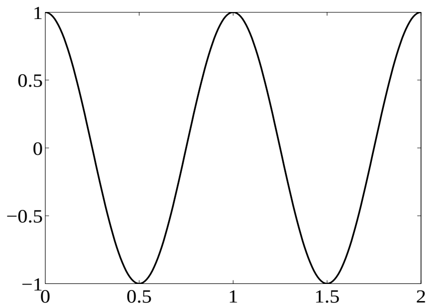

Figure D.1: A simple graph produced by MATLAB

More than one graphs may be plotted on the same graph, with different line characteristics. Lines may be added, axes and tick marks may be redefined, axis labels and a title may be added as shown in D.2:

```
>> newt=0:0.02:1; newx=sin(2*pi*newt);
>> plot(t,x,newt,newx,'o')
>> axis([-0.5 2.5 -1.25 1.25])
>> set(gca,'XTick',0:0.5:2,'YTick',-1:0.5:1)
>> line([-0.5 2.5],[0 0]), line([0 0],[-1.25 1.25])
>> xlabel('t'), ylabel('cos 2\pit (-) and sin 2\pit (o)')
>> title('A Simple MATLAB Plot')
```

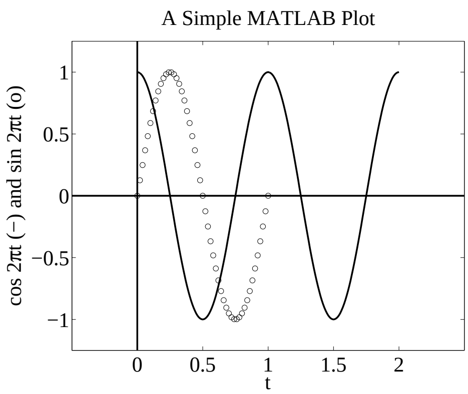

Figure D.2: A more complicated graph produced by MATLAB

D.6 Solving Ordinary Differential Equations

MATLAB provides two functions, `ode23` and `ode45`, for solving systems of first-order differential equation of the form

$$\mathbf{x}' = \mathbf{f}(t, \mathbf{x})\,, \quad \mathbf{x}(t_0) = \mathbf{x}_0$$

Although they use different numerical techniques, their formats are exactly the same:

```
[t,x]=ode23('myfnc', tspan, x0);
```

where `myfunc` is the name of a user-defined function that evaluates $\mathbf{f}(t, \mathbf{x})$ for a pair $(t, \mathbf{x})$ and returns it with name `xdot`, `tspan` is an array of strictly increasing or decreasing values of t_k at which the solution is to be found, and `x0` is a vector containing the initial value $\mathbf{x}_0$. The output array `t` contains a set of discrete points $t_k, k = 0, 1, \ldots, m$ in the range specified by `tspan`, and each column of the output matrix `x` contains the values of the corresponding component of the solution at t_k. If `tspan=[ti tf]`, then `ode23` use a variable step size to generate `t` with `t(1)=ti` and `t(m)<tf`.

As an example, the first order differential equation

$$y' = -2ty^2\;, \quad y(0) = 1$$

has the exact solution (see Example 2.15)

$$y = \frac{1}{t^2 + 1}$$

The following set of commands evaluate the exact solution and plot it together with its difference from the solution obtained by the `ode23` function.

```
>> t=0:0.01:5;
>> y_e=1./(t.*t+1); [t,y_a]=ode23('myrhs',t,1);
>> subplot(211),plot(t,y_e)
>> Xlabel('t'),Ylabel('y_e')
>> subplot(212),plot(t,y_e-y_a')
>> Xlabel('t'),Ylabel('y_e-y_a')
```

where the MATLAB function

```
function xdot=myrhs(t,x)
xdot=-2*t.*x.*x;
```

which is saved as a text file with name `myrhs.m`, evaluates $f(t, y) = -2ty^2$. This example also illustrates the use of the `subplot(rcn)` command, which divides a figure area into an `r-by-c` array with `n` referring to the nth cell on which the current figure is to be plotted.

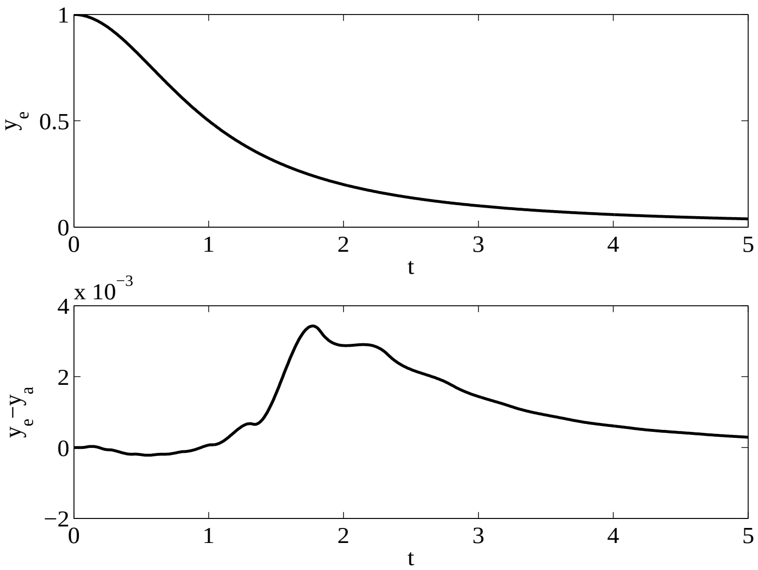

Figure D.3: Illustration of the `subplot` command

Index

n-space, 87

adjugate matrix, 162
algebraic sum, 124
angle between vectors, 255, 269
augmented matrix, 17

basic column, 20
basic variables, 22
basis, 98, 99
  canonical, 99
  change of, 107
  orthogonal, 251
  orthonormal, 251
Bellman-Gronwal Lemma, 302
Bernoulli equation, 76
Bessel's inequality, 268
boundary value problem, 80, 174

Cauchy sequence, 266
Cayley-Hamilton theorem, 175
change–of–basis matrix, 107, 112
characteristic equation, 43, 53, 170, 231
characteristic polynomial, 170, 231
codomain, 109
coefficient matrix, 12
cofactor, 158
column equivalence, 35, 148
column representation of vectors, 104
column space, 135
companion matrix, 196
complementary solution, 22, 27, 45, 57, 216, 230
condition number, 289
conic section, 278
convergence, 265
convolution, 111
Cramer's rule, 160

determinant, 154
  column expansion of, 154
  Laplace expansion of, 158
  row expansion of, 154
diagonal dominance, 205
diagonal form, 179

difference equations, 122
differential equation(s), 41
  exact, 65
  implicit solution of, 66, 68
  linear, 43, 53, 211, 226
  numerical solution of, 71
  order of, 41
  ordinary, 41
  partial, 41
  separable, 68
  solution curve of, 42
  solution of, 42
  system of, 70, 211
differential operator, 60
  linear, 61, 110
direct sum, 124
  orthogonal, 284
discrete Fourier series, 106
domain, 109

echelon form
  column, 35, 136
  reduced column, 35
  reduced row, 20
  row, 20, 135
eigenfunction, 174
eigenspace, 171
  generalized, 194
eigenvalue, 170
  algebraic multiplicity of, 171
  geometric multiplicity of, 171
eigenvector, 170
  generalized, 194
elementary matrix, 141
elementary operations, 17, 35, 95
equivalence
  of linear systems, 18
  of matrices, 17, 35, 148
  of norms, 264
Euclidean norm, 244
Euler method, 72
exact differential equation, 65
existence and uniqueness theorem, 299
exponential order, 303

Fibonacci sequence, 133
field, 1, 86, 295
Fourier series, 106, 259
Frobenius norm, 246
function of a matrix, 198
function space, 88
fundamental matrix, 213

Gauss-Jordan algorithm, 22
Gaussian elimination, 21
general solution, 27, 48, 57, 121, 132, 216, 230
generalized eigenvector, 194
generalized inverse, 146
Gersgorin's theorem, 205
Gram matrix, 251
Gram-Schmidt process, 255

Hölder's inequality, 263
Hermitian adjoint, 5
Hermitian matrix, 5, 272, 277
    indefinite, 277
    positive (negative) definite, 277
    positive (negative) semi-definite, 277
Hilbert matrix, 163, 268

idempotent matrix, 126
identity matrix, 8
image, 115
implicit solution, 66, 68
impulse response, 51
infinity norm, 244
initial conditions, 46, 57, 211
initial-value problem, 46, 57, 211
inner product, 248
inner product space, 248
integrating factor, 67
interpolating polynomial, 198
invariant subspace, 186, 194
inverse Laplace4 transform, 303
inverse of a matrix, 140
    generalized, 146
    left, 140
    pseudo, 286
    right, 140
inverse transformation, 119, 140
isomorphism, 119

Jordan form, 191

kernel, 115

Laplace transform, 303
least–squares problem, 257, 285
left inverse, 117, 140
linear combination, 91
linear dependence, 92
linear differential equation(s), 43, 211, 226
    nth order, 226
    characteristic equation of, 43, 53, 231
    characteristic polynomial of, 231
    complementary solution of, 45, 57, 230
    first order, 43
    general solution of, 48, 57, 230
    homogeneous, 43, 53, 227
    non-homogeneous, 44, 56, 229
    particular solution of, 44, 57, 230
    second order, 53
    system of, 211
linear differential operator, 61, 110
linear equations, 119
    general solution of, 121, 132
    homogeneous, 120
linear independence, 27, 55, 92, 94, 212, 227, 232
linear operator, 109
linear system(s), 12
    complementary solution of, 22, 27
    consistent, 12
    equivalence of, 18
    general solution of, 27
    homogeneous, 12
    ill-conditioned, 31
    inconsistent, 12
    particular solution of, 22, 27
    solution of, 12
linear transformation(s), 108
    codomain of, 109
    domain of, 109
    image of, 115
    inverse of, 119, 140
    kernel of, 115
    left inverse of, 117, 140
    matrix representation of, 112
    null space of, 115
    nullity of, 115
    one-to-one, 116
    onto, 118
    range space of, 115
    rank of, 115
    right inverse of, 118, 140
Lipschitz condition, 299

Lorentz transformation, 130
LU decomposition, 150

Markov matrix, 205
matrices
  addition of, 3
  column equivalence of, 35, 148
  commutative, 7
  equality of, 3
  equivalence of, 148
  multiplication of, 6
  row equivalence of, 17, 148
  similarity of, 149, 178
matrix, 1
  adjugate, 162
  augmented, 17
  change-of-basis, 107, 112
  characteristic equation of, 170
  characteristic polynomial of, 170
  coefficient, 12
  cofactor of, 158
  column, 1
  column space of, 135
  companion, 196
  condition number of, 289
  determinant of, 154
  diagonal, 2, 8
  diagonal form of, 179
  diagonally dominant, 205
  echelon form of, 20, 35, 135
  eigenspace of, 171
  eigenvalue of, 170
  eigenvector of, 170
  element of, 1
  elementary, 141
  function of, 198
  fundamental, 213
  generalized eigenspace of, 194
  generalized eigenvector of, 194
  generalized inverse of, 146
  Gram, 251
  Hermitian, 5, 272, 277
  Hermitian adjoint of, 5
  Hilbert, 163, 268
  idempotent, 126
  identity, 8
  image of, 115
  inverse of, 140
  invertible, 141
  Jordan form of, 191
  kernel of, 115
  left inverse of, 140
  Markov, 205
  minimum polynomial of, 176, 205
  modal, 179, 191
  nilpotent, 128
  nonsingular, 138, 160
  norm of, 246
  normal, 291
  normal form of, 146
  null, 3
  null space of, 115
  order of, 1
  orthogonal, 269
  partitioned, 9
  permutation, 142
  projection, 126
  pseudoinverse of, 286
  range space of, 115
  rank of, 136
  right inverse of, 140
  rotation, 270, 290
  row, 1
  row space of, 135
  scalar multiplication of, 3
  semi-diagonal form of, 189
  singular, 138
  skew-Hermitian, 5
  skew-symmetric, 5
  square, 2
  state transition, 213, 217
  symmetric, 5, 272
  trace of, 2
  transpose of, 4
  triangular, 2
  unitary, 269
  Vandermonde, 166
  Wronski, 229
  zero, 3
matrix representation of linear transformations, 112
method of undetermined coefficients, 233
method of variation of parameters, 44, 56, 215, 230
minimum polynomial, 176, 205
Minkowski's inequality, 263
minor, 158
modal matrix, 179, 191
  orthogonal, 271, 273
  real, 189
  unitary, 270, 272
mode, 219
Moore-Penrose generalized inverse, 286

nilpotent matrix, 128
nonsingular matrix, 138, 160
norm
  defined by an inner product, 249
  equivalence of, 264
  Euclidean, 244
  Frobenius, 246
  infinity, 244
  of a function, 244
  of a matrix, 246
  of a vector, 243
  subordinate, 246
  uniform, 243, 244
normal form, 146
normal matrix, 291
normed vector space, 243
null space, 115
nullity, 115
numerical solution, 71

order
  of a differential equation, 41
  of a matrix, 1
orthogonal
  basis, 251
  complement, 252
  direct sum, 284
  matrix, 269
  projection, 253
  set, 250
  trajectories, 80
  vectors, 250
orthonormal
  basis, 251
  set, 250
  vectors, 250

partial fraction expansion, 306
partial pivoting, 30, 151
particular solution, 22, 27, 44, 57, 216, 230
partitioned matrix, 9
  block of, 9
Pauli spin matrices, 207
permutation, 153
permutation matrix, 142
Picard iterates, 302
pivot element, 22
projection, 126
projection matrix, 126
projection theorem, 253
pseudoinverse, 286
Pythagorean theorem, 250

quadratic form, 274, 277
  indefinite, 274
  positive (negative) definite, 274
  positive (negative) semi-definite, 274
quadric surface, 280

range space, 115
rank
  column, 35, 136
  of a generalized eigenvector, 194
  of a linear transformation, 115
  of a matrix, 136
  row, 20, 22, 135
rational function, 306
recursion equation, 72, 122
right inverse, 118, 140
rotation matrix, 270, 290
row equivalence, 17, 148
row space, 135

scalar, 3, 86
scalar multiplication, 3, 86
Schur's theorem, 290
Schwarz Inequality, 249
semi-diagonal form, 189
separable differential equation, 68
similarity, 149, 178
singular matrix, 138
singular value decomposition, 282
singular values, 283
singular vectors, 283
solution
  by Laplace transform, 309
  of a differential equation, 42
  of a linear equation, 120
  of a linear system, 12
span, 91
standard inner product on $\mathbf{R}^{n\times 1}, \mathbf{C}^{n\times 1}$, 248
state transition matrix, 213, 217, 313
step response, 50
submatrix, 9
subordinate matrix norm, 246
subspace, 89
  complement of, 126
  invariant, 186, 194
  orthogonal complement of, 252
symmetric matrix, 5, 272
  indefinite, 274
  positive (negative) definite, 274
  positive (negative) semi-definite, 274
system of differential equations, 70, 211
system of linear differential equations, 211

complementary solution of, 216
fundamental matrix of, 213
general solution of, 216
modes of, 219
particular solution of, 216
state transition matrix of, 213
system of linear equations, 12

trace, 2
transpose, 4
triangle inequality, 243

uniform norm, 243, 244
unit impulse, 51
unit step function, 48, 305
unit vector, 243
unitary matrix, 269

Vandermonde's matrix, 166
vector space(s), 86
algebraic sum of, 124
basis of, 99
dimension of, 101
direct sum of, 124
finite dimensional, 100
infinite dimensional, 100
isomorphic, 119
normed, 243
subspace of, 89
vector(s), 1, 86
addition of, 86
angle between, 255, 269
column, 1
column representation of, 104
linear combination of, 91
linear dependence of, 92
linear in dependence of, 92
norm of, 243
orthogonal, 250
orthonormal, 250
row, 1
scalar multiplication of, 86
span of, 91
unit, 243

Wronski matrix, 229
Wronskian, 229

zero matrix, 3