CHAPTER

3 Two-Dimensional Systems and Z-Transforms

In this chapter we look at the 2-D Z-transform. It is a generalization of the 1-D Z-transform used in the analysis and synthesis of 1-D linear constant coefficient difference equation-based systems. In two and higher dimensions, the corresponding linear systems are partial difference equations. The analogous continuous parameter systems are partial differential equations. In fact, one big application of partial difference equations is in the numerical or computer solution of the partial differential equations of physics. We also look at LSI stability in terms of its Z-transform system function and present several stability conditions in terms of the zero-root locations of the system function.

3.1 LINEAR SPATIAL OR 2-D SYSTEMS

The spatial or 2-D systems we will mainly be concerned with are governed by difference equations in the two variables n_1 and n_2. These equations can be realized by logical interconnection of multipliers, adders, and shift or delay elements via either software or hardware. For the most part, the coefficients of such equations will be constant, hence the name *linear constant coefficient difference equations* (LCCDEs). The study of 2-D or partial difference equations is much more involved than that of the corresponding 1-D LCCDEs, and much less is known about the general case. Nevertheless, many practical results have emerged, the most basic of which will be presented here. We start with the general input/output equation:

$$\sum_{(k_1,k_2)\in\mathcal{R}_a} a_{k_1,k_2} y(n_1-k_1,n_2-k_2) = \sum_{(k_1,k_2)\in\mathcal{R}_b} b_{k_1,k_2} x(n_1-k_1,n_2-k_2), \quad (3.1\text{–}1)$$

where x is the known input and y is the output to be determined. We consider the coefficients a_{k_1,k_2} and b_{k_1,k_2} to be arrays of real numbers and call b_{k_1,k_2} the *feedforward* coefficients and a_{k_1,k_2} the *feedback* coefficients. We wish to solve (3.1–1) by finding output value y for every point in a prescribed region $\mathcal{R}_y$ given needed input values x plus output values y on the boundary of $\mathcal{R}_y$. We denote this boundary region somewhat imprecisely as $\mathcal{R}_{bc}$. The highest values of k_1 and k_2 on the left-hand side of (3.1–1) determine the order of the difference equation. In general, such equations have to be solved via matrix or iterative methods, but our main interest is *2-D filters*

Multidimensional Signal, Image, and Video Processing and Coding

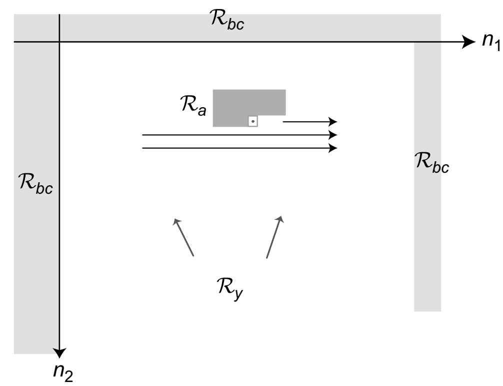

FIGURE 3.1–1

An example of the solution region of a spatial difference equation solution region using a nonsymmetric half-plane (NSHP) coefficient support $\mathcal{R}_a$.

for which the output y can be calculated in a recursive manner from the input x by scanning through the data points (n_1, n_2).

Keeping only the output value $y(n_1, n_2)$ on the left-hand side of (3.1–1), assuming $a_{0,0} \neq 0$ and $(0,0) \in \mathcal{R}_a$, we can write

$$y(n_1,n_2) = -\sum_{(k_1,k_2)\in\mathcal{R}_a} a'_{k_1,k_2} y(n_1-k_1, n_2-k_2) + \sum_{(k_1,k_2)\in\mathcal{R}_b} b'_{k_1,k_2} x(n_1-k_1, n_2-k_2), \tag{3.1–2}$$

where the a'_{k_1,k_2} and b'_{k_1,k_2} are the normalized coefficients (i.e., those divided by $a_{0,0}$). Then, depending on the shape of the region $\mathcal{R}_a$, we may be able to calculate the solution recursively. For example, we would say that the *direction of recursion* of (3.1–2) is "downward and to the right" in Figure 3.1–1, which shows a scan proceeding left-to-right and top-to-bottom.[1] Note that the special shape of the output mask $\mathcal{R}_a$ in Figure 3.1–1 permits such a recursion because of its property of not including any outputs that have not already been scanned and processed in the *past* (i.e., "above and to the left").

Example 3.1–1 shows how such a recursion proceeds in the case of a simple first-order 2-D difference equation.

Example 3.1–1: Simple Difference Equation

We now consider the simple LCCDE

$$y(n_1,n_2) = x(n_1,n_2) + \frac{1}{2}[y(n_1-1,n_2) + y(n_1,n_2-1)] \tag{3.1–3}$$

[1] The vertical axis is directed downward, as is common in image processing, where typically the processing proceeds from top to bottom of the image.

to be solved over the first quadrant (i.e., $\mathcal{R}_y = \{n_1 \geq 0, n_2 \geq 0\}$). In this example, we assume that the input x is everywhere zero, but that the boundary conditions given on $\mathcal{R}_{bc} = \{n_1 = -1\} \cup \{n_2 = -1\}$ are nonzero and specified by

$$
\begin{aligned}
&y(-1,1) = y(-1,2) = y(-1,3) = 1, \\
&y(-1,\text{else}) = 0, \\
&y(\text{else},-1) = 0.
\end{aligned}
$$

To calculate the solution recursively, we first determine a *scanning order*. In this case, it is the so-called *raster scan* used in video monitors: first we process the row $n_2 = 0$, starting at $n_1 = 0$ and incrementing by one each time; then we increment n_2 by one, and process the next row. With this scanning order, the difference equation (3.1–3) is seen to only use previous values of y at the "present time," and so is recursively calculable. Proceeding to work out the solution, we obtain

0	0	1	1	1	0	0	$\cdots$
0	0	$\frac{1}{2}$	$\frac{3}{4}$	$\frac{7}{8}$	$\frac{7}{16}$	$\frac{7}{32}$	$\cdots$
0	0	$\frac{1}{4}$	$\frac{1}{2}$	$\frac{11}{16}$	$\frac{18}{32}$	$\frac{25}{64}$	$\cdots$
0	0	$\frac{1}{8}$	$\frac{5}{16}$	$\frac{16}{32}$	$\frac{34}{64}$	$\cdots$	$\cdots$
0	0	$\frac{1}{16}$	$\cdots$				

$n_2 \downarrow$ $\longrightarrow n_1$.

In Example 3.1–1 we have computed the solution to a spatial difference equation by recursively calculating out the values in a suitable scanning order, for a nonzero set of boundary "initial" conditions, but with zero input sequence. In Example 3.1–2 we consider the same 2-D difference equation to be solved over the same output region, but with zero initial boundary conditions and a nonzero input. By linearity of the partial difference equation, the general case of nonzero boundaries and nonzero input follows by superposition of these two *zero-input* and *zero-state* solutions.

Example 3.1–2: Simple Difference Equation (cont'd)

We consider the simple LCCDE

$$y(n_1,n_2) = x(n_1,n_2) + \frac{1}{2}[y(n_1-1,n_2) + y(n_1,n_2-1)] \tag{3.1–4}$$

to be solved over output solution region $\mathcal{R}_y = \{n_1 \geq 0, n_2 \geq 0\}$. The boundary conditions given on $\mathcal{R}_{bc} = \{n_1 = -1\} \cup \{n_2 = -1\}$ are taken as all zeros. The input sequence is $x(n_1,n_2) = \delta(n_1,n_2)$. Starting at $(n_1,n_2) = (0,0)$, we begin to generate the impulse response of the difference equation. Continuing the recursive calculation for the next few columns and rows, we obtain

$n_2\downarrow$

0	0	0	0	0	0	0	$\cdots$
0	1	$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{8}$	$\frac{1}{16}$	$\cdots$	$\cdots$
0	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{3}{8}$	$\frac{4}{16}$	$\frac{5}{32}$	$\cdots$	$\cdots$
0	$\frac{1}{4}$	$\frac{3}{8}$	$\frac{6}{16}$	$\frac{10}{32}$	$\cdots$	$\cdots$	$\cdots$
0	$\frac{1}{8}$	$\frac{4}{16}$	$\frac{10}{32}$	$\cdots$	$\cdots$	$\cdots$	$\cdots$

$\longrightarrow n_1.$

It turns out that this spatial impulse response has a closed-form analytic solution [1, 2],

$$y(n_1,n_2) = h(n_1,n_2) = \binom{n_1+n_2}{n_1} 2^{-(n_1+n_2)} u_{++}(n_1,n_2),$$

where $\binom{n_1+n_2}{n_1}$ is the combinatorial symbol for "n_1+n_2 things taken n_1 at a time,"

$$\binom{n_1+n_2}{n_1} = \frac{(n_1+n_2)!}{n_1!\,n_2!}, \quad \text{for} \quad n_1 \geq 0, n_2 \geq 0,$$

with $0!$ taken as 1, and where $u_{++}(n_1,n_2) = u(n_1,n_2)$ is the first quadrant unit step function. ■

Though it is usually the case that 2-D difference equations do not have a closed-form impulse response, the first-order difference equation of Example 3.1–2 is one of the few exceptions. From these two examples, we can see it is possible to write the general solution to a spatial linear difference equation as a sum of a zero-input solution given rise by the boundary values plus a zero-state solution driven by the input sequence

$$y(n_1,n_2) = y_{ZI}(n_1,n_2) + y_{ZS}(n_1,n_2).$$

This generalizes the familiar 1-D systems theory result. To see this, consider a third example with both nonzero input and nonzero boundary conditions. Then note that the sum of the two solutions from these examples will solve this new problem.

In general, and depending on the output coefficient support region $\mathcal{R}_a$, there can be different recursive directions for (3.1–1), which we can obtain by bringing other terms to the left-hand side and recursing in other directions. For example, we can take (3.1–4) from Example (3.1–2) and bring $y(n_1,n_2-1)$ to the left-hand side to yield

$$y(n_1,n_2-1) = -2y(n_1,n_2) + 2x(n_1,n_2) + y(n_1-1,n_2),$$

or equivalently,

$$y(n_1,n_2) = -2y(n_1,n_2+1) + y(n_1-1,n_2+1) + 2x(n_1,n_2+1),$$

with direction of recursion upwards, to the right or left. So the direction in which a 2-D difference equation can be solved recursively, or recursed, depends on the support of the output or feedback coefficients (i.e., $\mathcal{R}_a$). For a given direction of recursion, we can calculate the output points in particular orders that are constrained by the shape of the coefficient support region $\mathcal{R}_a$, resulting in an *order of computation*. In fact, there are usually several such orders of computation that are consistent with a given direction of recursion. Further, usually several output points can be calculated in parallel to speed the recursion.

Such recursive solutions are appropriate when the boundary conditions are only imposed "in the past" of the recursion—i.e., not on any points that must be calculated. In particular, with reference to Figure 3.1–1, we see no boundary conditions on the bottom of the solution region. In the more general case where there are both "initial" and "final" conditions, we can fall back on the general matrix solution for a finite region.

To solve LCCDE (3.1–1) in a finite solution region, we can use linear algebra and form a vector of the solution $\boldsymbol{y}$ scanned across the region in any prespecified manner. Doing the same for the input $\boldsymbol{x}$ and the boundary conditions $\boldsymbol{y}_{bc}$, we can write all the equations with one very large dimensioned vector equation,

$$\boldsymbol{x} = \boldsymbol{A}\boldsymbol{y} + \boldsymbol{B}\boldsymbol{y}_{bc},$$

for appropriately defined coefficient matrices $\boldsymbol{A}$ and $\boldsymbol{B}$. For a 1000×1000 image, the dimension of $\boldsymbol{y}$ would be 1,000,000. Here, $\boldsymbol{A}\boldsymbol{y}$ provides the terms of the equations where $\boldsymbol{y}$ is on the region, and $\boldsymbol{B}\boldsymbol{y}_{bc}$ provides the terms when $\boldsymbol{y}$ is on the boundary. A problem at the end of the chapter asks you to prove this fact.

If the solution region of the LCCDE is infinite, then as in the 1-D case, it is often useful to express the solution in terms of a Z-transform, which is our next topic.

3.2 Z-TRANSFORMS

Definition 3.2–1: Z-Transform

The 2-D Z-transform of a two-sided sequence $x(n_1,n_2)$ is defined as follows:

$$X(z_1,z_2) \triangleq \sum_{n_1=-\infty}^{+\infty} \sum_{n_2=-\infty}^{+\infty} x(n_1,n_2) z_1^{-n_1} z_2^{-n_2}, \tag{3.2–1}$$

where $(z_1,z_2) \in \mathcal{C}^2$, the "2-D" (really 4-D) complex Cartesian product space. In general, there will be only some values of $(z_1,z_2)^T \triangleq \boldsymbol{z}$ for which this double sum will converge. Only *absolute convergence*,

$$\sum_{n_1=-\infty}^{+\infty} \sum_{n_2=-\infty}^{+\infty} |x(n_1,n_2) z_1^{-n_1} z_2^{-n_2}|$$

$$= \sum_{n_1=-\infty}^{+\infty} \sum_{n_2=-\infty}^{+\infty} |x(n_1,n_2)|\ |z_1|^{-n_1}\ |z_2|^{-n_2} < \infty,$$

is considered in the theory of complex variables [3, 4], so we look for joint values of $|z_1|$ and $|z_2|$ that will yield absolute convergence. The set of $\boldsymbol{z}$ for which this occurs is called the *region of convergence*, denoted $\mathcal{R}_x$. In summary, a 2-D Z-transform is specified by its functional form $X(z_1, z_2)$ and its convergence region $\mathcal{R}_x$. ■

Similar to the 1-D case, the Z-transform is simply related to the Fourier transform, when both exist:

$$X(z_1, z_2)|_{\substack{z_1=e^{jw_1}\\ z_2=e^{jw_2}}} = X(\omega_1, \omega_2),$$

with the customary abuse of notation.[2]

A key difference from the 1-D case is that the 2-D complex variable z exists in a 4-D space and is hard to visualize. The familiar unit circle becomes something a bit more abstract, the *unit bi-circle* in $\mathcal{C}^2$ [4]. The unit disk then translates over to the *unit bi-disk,* $\{|z_1|^2 + |z_2|^2 \leq 1\} \in \mathcal{C}^2$. Another key difference for two and higher dimensions is that the zeros of the Z-transform are no longer isolated. Two different *zero loci* can intersect.

Example 3.2–1: Zero Loci

Consider the following signal $x(n_1, n_2)$:

$n_2 \backslash n_1$	0	1
0	1	2
1	2	1

with assumed support $\{0,1\} \times \{0,1\}$. This simple four-point signal could serve, after normalization, as the impulse response of a simple directional spatial averager, giving an emphasis to structures at 45°. Proceeding to take the Z-transform, we obtain

$$X(z_1, z_2) = 1 + 2z_1^{-1} + 2z_2^{-1} + z_1^{-1}z_2^{-1}.$$

This Z-transform X is seen to exist for all $\mathcal{C}^2$ except for $z_1 = 0$ or $z_2 = 0$. Factoring X, we obtain

$$X(z_1, z_2) = 1 + 2z_2^{-1} + z_1^{-1}(2 + z_2^{-1}),$$

which upon equating to zero gives the zero (z_1, z_2) locus

$$z_1 = -\frac{2z_2 + 1}{z_2 + 2}, \quad \text{for} \quad z_2 \neq -2,$$

$$z_1 = +\infty, \text{ otherwise.}$$

[2]To avoid confusion, when the same symbol X is being used for two different functions, we note that the Fourier transform $X(\omega_1, \omega_2)$ is a function of real variables, while the Z-transform $X(z_1, z_2)$ is a function of complex variables. A pitfall, for example $X(1, 0)$, can be avoided by simply writing either $X(\omega_1, \omega_2)|_{\substack{\omega_1=1\\ \omega_2=0}}$ or $X(z_1, z_2)|_{\substack{z_1=1\\ z_2=0}}$, whichever is appropriate, in cases where confusion could arise.

We notice that for each value of z_2 there is a corresponding value of z_1 for which the Z-transform X takes on the value of zero. Notice also that, with the possible exception of $z_2 = -2$, the zero locus value $z_1 = f(z_2)$ is a *continuous function* of the complex variable z_2. This first-order 2-D system thus has one zero locus. ■

We next look at a more complicated second-order case where there are two root loci that intersect, but without being identical; therefore, we cannot just cancel the factors out. In the 1-D case that we are familiar with, the only way there can be a pole and zero at the same z location is when the numerator and denominator have a common factor. Example 3.2–2 shows that this is not true in general for higher dimensions.

Example 3.2–2: Intersecting Zero Loci

Consider the Z-transform

$$X(z_1, z_2) = (1 + z_1)/(1 + z_1 z_2),$$

for which the zero locus is easily seen to be $(z_1, z_2) = (-1, *)$, and the pole locus is $(z_1, z_2) = (\alpha, -1/\alpha)$, where $*$ represents an arbitrary complex number and α is any nonzero complex number. These two distinct zero sets are seen to intersect at $(z_1, z_2) = (-1, 1)$. One way to visualize these root loci is *root mapping*, which we will introduce later when we study the stability of 2-D filters (see Section 3.5). ■

Next, we turn to the topic of convergence for the 2-D Z-transform. As in the 1-D case, we expect that knowledge of the region in $\mathbf{z}$ space where the series converges will be essential to the uniqueness of the transform, and hence to its inversion.

3.3 REGIONS OF CONVERGENCE

Given a 2-D Z-transform $X(z_1, z_2)$, its *region of convergence* (ROC) is given as the set of $\mathbf{z}$ for which

$$= \sum_{n_1=-\infty}^{+\infty} \sum_{n_2=-\infty}^{+\infty} |x(n_1, n_2)| \, |z_1|^{-n_1} \, |z_2|^{-n_2}$$

$$= \sum_{n_1=-\infty}^{+\infty} \sum_{n_2=-\infty}^{+\infty} |x(n_1, n_2)| \, r_1^{-n_1} \, r_2^{-n_2} < \infty, \tag{3.3–1}$$

where $r_1 \triangleq |z_1|$ and $r_2 \triangleq |z_2|$ are the moduli of the complex numbers z_1 and z_2. The ROC can then be written in terms of such moduli values as

$$\mathcal{R}_x \triangleq \{(z_1, z_2) | \, |z_1| = r_1, \; |z_1| = r_2, \text{ and } (3.3-1) \text{ holds}\}.$$

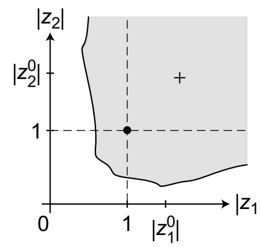

FIGURE 3.3–1

The 2-D complex magnitude plane. Here, (•) denotes the unit bi-circle and (+) denotes an arbitrary point at (z_1^0, z_2^0).

Since this specification only depends on magnitudes, we can plot ROCs in the convenient magnitude plane (Figure 3.3–1).

Example 3.3–1: Z-Transform Calculation

We consider the spatial, first quadrant step function

$$x(n_1,n_2) = u_{++}(n_1,n_2) = u(n_1,n_2).$$

Taking the Z-transform, we have the following from (3.2–1):

$$\begin{aligned}
X(z_1,z_2) &= \sum_{n_1=-\infty}^{+\infty}\sum_{n_2=-\infty}^{+\infty} u(n_1,n_2) z_1^{-n_1} z_2^{-n_2} \\
&= \sum_{n_1=0}^{+\infty}\sum_{n_2=0}^{+\infty} z_1^{-n_1} z_2^{-n_2} \\
&= \sum_{n_1=0}^{+\infty} z_1^{-n_1} \cdot \sum_{n_2=0}^{+\infty} z_2^{-n_2} \\
&= \frac{1}{1-z_1^{-1}}\frac{1}{1-z_2^{-1}} \quad \text{for} \quad |z_1|>1 \quad \text{and} \quad |z_2|>1, \\
&= \frac{z_1}{z_1-1}\frac{z_2}{z_2-1} \quad \text{with} \quad \mathcal{R}_x = \{|z_1|>1, |z_2|>1\}.
\end{aligned}$$

We can plot this ROC on the complex $\mathbf{z}$-magnitude plane as in Figure 3.3–2. Note that we have shown the ROC as the gray region and moved it slightly outside the lines $|z_1| = 1$ and $|z_2| = 1$ in order to emphasize that this open region does not include these lines. The zero loci for this separable signal are the manifold $z_1 = 0$ and the manifold $z_2 = 0$. These two distinct loci intersect at the complex point $z_1 = z_2 = 0$. The pole loci are also two in number and occur at the manifold $z_1 = 1$ and the manifold $z_2 = 1$. We note that these two pole loci intersect at the single complex point $z_1 = z_2 = 1$.

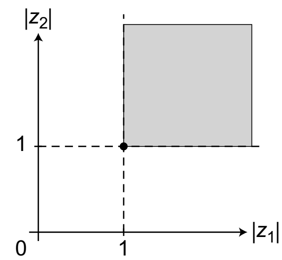

FIGURE 3.3–2

The gray area illustrates the ROC for the Z-transform of the first quadrant unit step function $u(n_1, n_2) = u_{++}(n_1, n_2)$.

Next, we consider how the Z-transform changes when the unit step switches to another quadrant.

Example 3.3–2: Unit Step Function in the Fourth Quadrant

Here, we consider a unit step function that has support on the fourth quadrant. We denote it as $u_{+-}(n_1, n_2)$:

$$u_{+-}(n_1, n_2) \triangleq \begin{cases} 1, & n_1 \geq 0, n_2 \leq 0, \\ 0, & \text{else.} \end{cases}$$

So, setting $x(n_1, n_2) = u_{+-}(n_1, n_2)$, we next compute

$$\begin{aligned} X(z_1, z_2) &= \sum_{n_1=-\infty}^{+\infty} \sum_{n_2=-\infty}^{+\infty} u_{+-}(n_1, n_2) z_1^{-n_1} z_2^{-n_2} \\ &= \sum_{n_1=0}^{+\infty} z_1^{-n_1} \cdot \sum_{n_2=-\infty}^{0} z_2^{-n_2} \\ &= \sum_{n_1=0}^{+\infty} z_1^{-n_1} \cdot \sum_{n_2'=0}^{+\infty} z_2^{n_2'} \quad \text{with} \quad n_2' \triangleq -n_2 \\ &= \frac{1}{1 - z_1^{-1}} \frac{1}{1 - z_2} \quad \text{for} \quad |z_1| > 1 \quad \text{and} \quad |z_2| < 1, \\ &= -\frac{z_1}{z_1 - 1} \frac{1}{z_2 - 1} \quad \text{with} \quad \mathcal{R}_x = \{|z_1| > 1, |z_2| < 1\}. \end{aligned}$$

The ROC is shown as the gray area in Figure 3.3–3.

All four quarter-plane support, unit step sequences have the special property of separability. Since the Z-transform is a separable operator, this makes the calculation split into the product of two 1-D transforms in the n_1 and n_2 directions, as we have

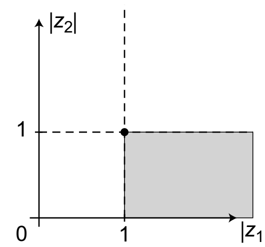

FIGURE 3.3–3

The ROC (gray area) for the fourth quadrant unit step function $u_{+-}(n_1,n_2)$.

just seen. The ROC then factors into the Cartesian product of the two 1-D ROCs. We look at a more general case next.

More General Case

In general, we have the Z-transform

$$X(z_1,z_2) = \frac{B(z_1,z_2)}{A(z_1,z_2)},$$

where both B and A are polynomials in coefficients of some partial difference equation,

$$B(z_1,z_2) = \sum_{n_1=-N_1}^{+N_1}\sum_{n_2=-N_2}^{+N_2} b(n_1,n_2)z_1^{-n_1}z_2^{-n_2} \quad \text{and}$$

$$A(z_1,z_2) = \sum_{n_1=-N_1}^{+N_1}\sum_{n_2=-N_2}^{+N_2} a(n_1,n_2)z_1^{-n_1}z_2^{-n_2}.$$

To study the existence of this Z-transform, we focus on the denominator and rewrite A as

$$A(z_1,z_2) = z_1^{-N_1}z_2^{-N_2}\widetilde{A}(z_1,z_2),$$

where $\widetilde{A}$ is a *strict-sense polynomial* in z_1 and z_2 (i.e., no negative powers of z_1 or z_2). Grouping together terms in z_1^n, we can write

$$\widetilde{A}(z_1,z_2) = \sum_{n=0}^{\widetilde{N}_1} a_n(z_2)z_1^n,$$

yielding $\widetilde{N}_1$ poles (N_1 at most!) for each value of z_2,

$$z_1^i = f_i(z_2), \quad i = 1,\ldots,\widetilde{N}_1.$$

A sketch of such a pole surface is plotted in Figure 3.3–4. Note that we are only plotting the magnitude of one surface here, and this plot therefore does not tell the

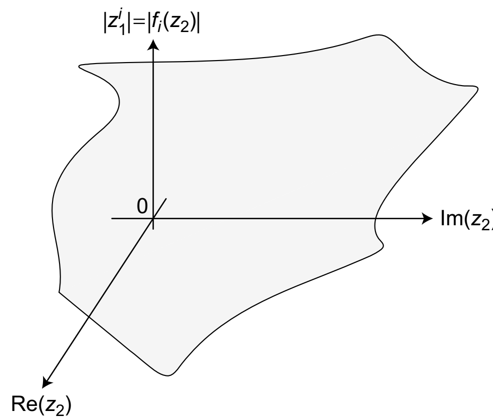

FIGURE 3.3–4

Sketch of pole magnitude $|z_1^i|$ surface as a function of a point in the z_2 complex plane.

whole story. Also there are $\widetilde{N}_1$ such sheets. Of course, there will be a similar number of zero loci or surfaces that come about from the numerator

$$\widetilde{B}(z_1, z_2) = \sum_{n=0}^{\widetilde{N}_1} b_n(z_2) z_1^n,$$

where $B(z_1, z_2) = z_1^{-N_1} z_2^{-N_2} \widetilde{B}(z_1, z_2)$. Note that these zero surfaces can intersect the pole surfaces (as well as each other) without being identical. Thus indeterminate $\frac{0}{0}$ situations can arise that cannot be simply canceled out. One classic example is [5]

$$\frac{z_1 + z_2 - 2}{(z_1 - 1)(z_2 - 1)},$$

which evaluates to $\frac{0}{0}$ at the point $(z_1, z_2) = (1, 1)$, and yet has no cancelable factors.

3.4 SOME Z-TRANSFORM PROPERTIES

Here, we list some useful properties of the 2-D Z-transform that we will use in the sequel. Many are easy extensions of known properties of the 1-D Z-transform, but some are essentially new. In listing these properties, we introduce the symbol Z for the 2-D Z-transform operator.

Linearity property:

$$Z\{ax(n_1, n_2) + by(n_1, n_2)\} = aX(z_1, z_2) + bY(z_1, z_2), \quad \text{with ROC} = \mathcal{R}_x \cap \mathcal{R}_y.$$