MUS420/EE367A Supplementary Lecture
Introduction to State Space Models

Julius O. Smith III (jos@ccrma.stanford.edu)
Center for Computer Research in Music and Acoustics (CCRMA)
Department of Music, Stanford University
Stanford, California 94305

January 8, 2007

Outline

- State Space Formulation
- Markov Parameters (Impulse Response)
- Transfer Function
- Difference Equations to State Space Models
- Similarity Transformations
- Modal Representation (Diagonalization)
- Matlab Examples

State Space Models

An important representation for linear systems is the *state-space* formulation

$$\begin{aligned} \underline{x}(n+1) &= A\underline{x}(n) + B\underline{u}(n) \\ \underline{y}(n) &= C\underline{x}(n) + D\underline{u}(n) \end{aligned}$$

where

- $\underline{x}(n) \in \mathbf{R}^N =$ *state vector* at time n
- $\underline{u}(n) = p \times 1$ vector of inputs
- $\underline{y}(n) = q \times 1$ output vector
- $A = N \times N$ *state transition matrix*
- $B = N \times p$ *input coefficient matrix*
- $C = q \times N$ *output coefficient matrix*
- $D = q \times p$ *direct path coefficient matrix*

The state-space representation is especially powerful for

- *multi-input, multi-output* (MIMO) linear systems
- *time-varying* linear systems

We'll be concerned primarily with the single-input, single-output (SISO) case, i.e., $p = q = 1$.

Markov Parameters

The *impulse response* of a state-space model is easily found by direct calculation: Let $\underline{x}(0) \stackrel{\Delta}{=} 0$. Then

$$\begin{aligned}
h(0) &= C\underline{x}(0)B + D = D \\
\underline{x}(1) &= A\,\underline{x}(0) + B\,\underline{\delta}(0) = B \\
h(1) &= CB \\
\underline{x}(2) &= A\,\underline{x}(1) + B\,\underline{\delta}(1) = AB \\
h(2) &= CAB \\
\underline{x}(3) &= A\,\underline{x}(1) + B\,\underline{\delta}(1) = A^2B \\
h(3) &= CA^2B \\
&\vdots \\
h(n) &= CA^{n-1}B, \quad n > 0
\end{aligned}$$

Thus, the impulse response of the state-space model can be summarized as

$$\boxed{h(n) = \begin{cases} D, & n = 0 \\ CA^{n-1}B, & n > 0 \end{cases}}$$

The impulse response terms CA^nB for $n \geq 0$ are known as the *Markov parameters*.

State Space Model Transfer Function

The *transfer function* is the z transform of the impulse response:

$$H(z) \stackrel{\Delta}{=} \sum_{n=0}^{\infty} h(n) z^{-n} = D + \sum_{n=1}^{\infty} \left(C A^{n-1} B\right) z^{-n}$$

$$= D + z^{-1} C \left[\sum_{n=0}^{\infty} \left(z^{-1} A\right)^n\right] B$$

The closed-form sum of a matrix geometric series gives

$$\boxed{H(z) = D + C\left(zI - A\right)^{-1} B}$$

Note that if there are p inputs and q outputs, $H(z)$ is a $p \times q$ *matrix* transfer function.

Example: $(p = 3,\, q = 2)$

$$H(z) = \begin{bmatrix} z^{-1} & \frac{1-z^{-1}}{1-0.5z^{-1}} & 1 + z^{-1} \\ \frac{2+3z^{-1}}{1-0.1z^{-1}} & \frac{1+z^{-1}}{1-z^{-1}} & \frac{(1-z^{-1})^2}{(1-0.1z^{-1})(1-0.2z^{-1})} \end{bmatrix}$$

System Poles

Above, we found the transfer function to be

$$H(z) = D + C\left(zI - A\right)^{-1} B$$

The poles of $H(z)$ are the same as those of

$$H_p(z) \stackrel{\Delta}{=} \left(zI - A\right)^{-1}$$

By *Cramer's rule* for matrix inversion, the denominator polynomial for $\left(zI - A\right)^{-1}$ is given by the *determinant*:

$$D(z) \stackrel{\Delta}{=} |zI - A|$$

where $|Q|$ denotes the *determinant* of the square matrix Q (also written as $\det(Q)$.)

- In linear algebra, the polynomial $D(z) = |zI - A|$ is called the *characteristic polynomial* for the matrix A.
- The roots of the characteristic polynomial are called the *eigenvalues* of A.
- Thus, *the eigenvalues of the state transition matrix A are the system poles*.

Difference Equation to State Space Form

A digital filter is often specified by its *difference equation* (Direct Form I). Second-order example:

$$y(n) = u(n) + 2u(n-1) + 3u(n-2) - \frac{1}{2}y(n-1) - \frac{1}{3}y(n-2)$$

Every nth order *difference equation* can be reformulated as a *first order vector* difference equation called the "state space" (or "state variable") representation:

$$\begin{aligned} \underline{x}(n+1) &= A\underline{x}(n) + Bu(n) \\ y(n) &= C\underline{x}(n) + Du(n) \end{aligned}$$

For the above example, we have, as we'll show,

$$A \stackrel{\Delta}{=} \begin{bmatrix} -\frac{1}{2} & -\frac{1}{3} \\ 1 & 0 \end{bmatrix} \quad \text{(state transition matrix)}$$

$$B \stackrel{\Delta}{=} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \text{(matrix routing input to state variables)}$$

$$C \stackrel{\Delta}{=} \begin{bmatrix} 3/2 \\ 8/3 \end{bmatrix} \quad \text{(output linear-combination matrix)}$$

$$D \stackrel{\Delta}{=} 1 \quad \text{(direct feedforward coefficient)}$$

Converting to State-Space Form by Hand

1. First, determine the filter transfer function $H(z)$. In the example, the transfer function can be written, by inspection, as

$$H(z) = \frac{1 + 2z^{-1} + 3z^{-2}}{1 + \frac{1}{2}z^{-1} + \frac{1}{3}z^{-2}}$$

2. If $h(0) \neq 0$, we must "pull out" the parallel delay-free path:

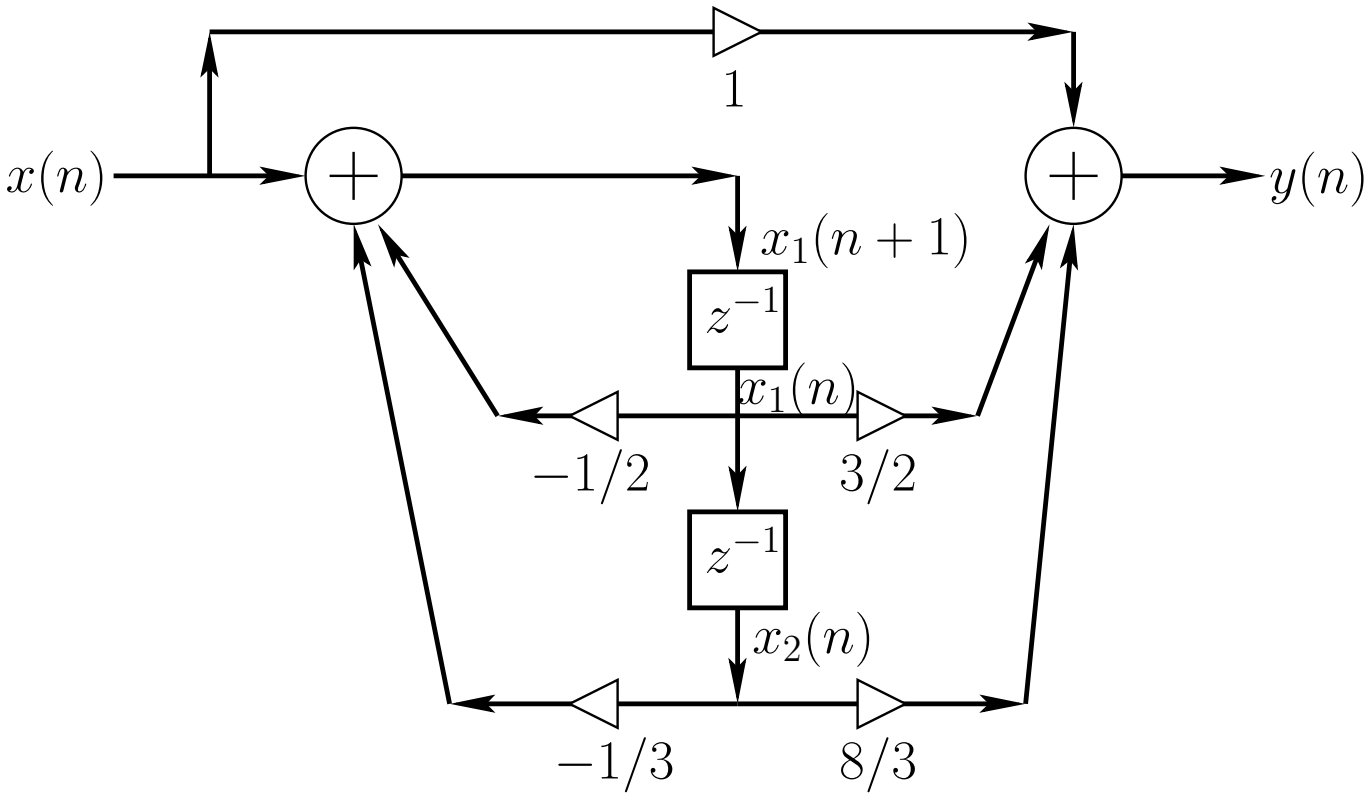
$$H(z) = d_0 + \frac{b_1 z^{-1} + b_2 z^{-2}}{1 + \frac{1}{2}z^{-1} + \frac{1}{3}z^{-2}}$$

   Obtaining a common denominator and equating numerator coefficients yields

$$\begin{aligned} d_0 &= 1 \\ b_1 &= 2 - \frac{1}{2} = \frac{3}{2} \\ b_2 &= 3 - \frac{1}{3} = \frac{8}{3} \end{aligned}$$

   The same result is obtained using long or synthetic division.

3. Next, draw the strictly causal part in *direct form II*, as shown below:



It is important that the filter representation be *canonical with respect to delay*, i.e., the number of delay elements equals the order of the filter.

4. Assign a state variable to the output of each delay element (see figure).

5. Write down the state-space representation by inspection. (Try it and compare to answer above.)

Matlab Conversion from Direct-Form to State-Space Form

Matlab has extensive support for state-space models, such as

- `tf2ss` - transfer-function to state-space conversion
- `ss2ttf` - state-space to transfer-function conversion

Note that these utilities are documented primarily for continuous-time systems, but they are also used for discrete-time systems.

Let's repeat the previous example using Matlab:

Previous Example Using Matlab

```
>> num = [1 2 3]; % transfer function numerator
>> den = [1 1/2 1/3]; % denominator coefficients
>> [A,B,C,D] = tf2ss(num,den)

A =
   -0.5000   -0.3333
    1.0000         0

B =
     1
     0

C =  1.5000    2.6667

D =  1

>> [N,D] = ss2tf(A,B,C,D)

N = 1.0000    2.0000    3.0000

D = 1.0000    0.5000    0.3333
```

Matlab Documentation

The `tf2ss` and `ss2tf` functions are documented at

`http://www.mathworks.com/access/helpdesk/help/toolbox/signal/tf2ss.shtml`

as well as within Matlab itself (e.g., `help tf2ss`).

Related Signal Processing Toolbox functions include

- `tf2sos` — Convert digital filter transfer function parameters to second-order sections form.
- `sos2ss` — Convert second-order filter sections to state-space form.
- `tf2zp` — Convert transfer function filter parameters to zero-pole-gain form.
- `zp2ss` — Convert zero-pole-gain filter parameters to state-space form.

Similarity Transformations

A *similarity transformation* of a state-space system is a *linear change of state variable coordinates*:

$$\underline{x}(n) \stackrel{\Delta}{=} \mathbf{E}\underline{\tilde{x}}(n)$$

where

- $\underline{x}(n) =$ original state vector
- $\underline{\tilde{x}}(n) =$ state vector in *new coordinates*
- $\mathbf{E} =$ any *invertible* (one-to-one) matrix (linear transformation)

Substituting $\underline{x}(n) = \mathbf{E}\underline{\tilde{x}}(n)$ into the general state-space model gives

$$\begin{aligned} \mathbf{E}\underline{\tilde{x}}(n+1) &= A\,\mathbf{E}\underline{\tilde{x}}(n) + B\underline{u}(n) \\ \underline{y}(n) &= C\mathbf{E}\underline{\tilde{x}}(n) + D\underline{u}(n) \end{aligned}$$

Premultiplying the first equation above by $\mathbf{E}^{-1}$, we get

$$\begin{aligned} \underline{\tilde{x}}(n+1) &= \left(\mathbf{E}^{-1}A\mathbf{E}\right)\underline{\tilde{x}}(n) + \left(\mathbf{E}^{-1}B\right)\underline{u}(n) \\ \underline{y}(n) &= (C\mathbf{E})\,\underline{\tilde{x}}(n) + D\underline{u}(n). \end{aligned}$$

Define the *transformed system matrices* by

$$\begin{aligned}\tilde{A} &= \mathbf{E}^{-1}A\mathbf{E}\\ \tilde{B} &= \mathbf{E}^{-1}B\\ \tilde{C} &= C\mathbf{E}\\ \tilde{D} &= D\end{aligned}$$

We can now write

$$\begin{aligned}\underline{\tilde{x}}(n+1) &= \tilde{A}\underline{\tilde{x}}(n) + \tilde{B}\underline{u}(n)\\ \underline{y}(n) &= \tilde{C}\underline{\tilde{x}}(n) + D\underline{u}(n)\end{aligned}$$

The transformed system describes the *same system* in new state-variable coordinates.

Let's verify that the transfer function has not changed:

$$\begin{aligned}\tilde{H}(z) &= \tilde{D} + \tilde{C}(zI - \tilde{A})^{-1}\tilde{B}\\ &= D + (C\mathbf{E})\left(zI - \mathbf{E}^{-1}A\mathbf{E}\right)^{-1}(\mathbf{E}^{-1}B)\\ &= D + C\left[\mathbf{E}\left(zI - \mathbf{E}^{-1}A\mathbf{E}\right)\mathbf{E}^{-1}\right]^{-1}B\\ &= D + C\left(zI - A\right)^{-1}B = H(z)\end{aligned}$$

- Since the eigenvalues of A are the poles of the system, it follows that the eigenvalues of $\tilde{A} = \mathbf{E}^{-1}A\mathbf{E}$ are the same. In other words, eigenvalues are unaffected by a similarity transformation.
- The transformed Markov parameters, $\tilde{C}\tilde{A}^n\tilde{B}$, are also unchanged since they are given by the inverse z

transform of the transfer function $\tilde{H}(z)$. However, it is also easy to show this by direct calculation.

State Space Modal Representation

Diagonal state transition matrix = *modal representation*:

$$\begin{bmatrix} x_1(n+1) \\ x_2(n+1) \\ \vdots \\ x_{N-1}(n+1) \\ x_N(n+1) \end{bmatrix} = \begin{bmatrix} \lambda_1 & 0 & 0 & \cdots & 0 \\ 0 & \lambda_2 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \lambda_{N-1} & 0 \\ 0 & 0 & 0 & 0 & \lambda_N \end{bmatrix} \begin{bmatrix} x_1(n) \\ x_2(n) \\ \vdots \\ x_{N-1}(n) \\ x_N(n) \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_{N-1} \\ b_N \end{bmatrix} u(n)$$

$$y(n) = C\underline{x}(n) + Du(n)$$

The N complex modes are *decoupled*:

$$\begin{aligned} x_1(n+1) &= \lambda_1 x_1(n) + b_1 u(n) \\ x_2(n+1) &= \lambda_2 x_2(n) + b_2 u(n) \\ &\vdots \\ x_N(n+1) &= \lambda_N x_N(n) + b_N u(n) \\ y(n) &= c_1 x_1(n) + c_2 x_2(n) + \cdots + c_N x_N(n) + Du(n) \end{aligned}$$

That is, diagonal state-space system consists of N *parallel one-pole systems*.

Diagonalizing a State-Space Model

To obtain the *modal representation*, we must *diagonalize the state-space model*.

The *similarity transformation* which diagonalizes the system is given by the *matrix of eigenvectors* of the state transition matrix A.

An eigenvector $\underline{e}_i$ of A satisfies, by definition,

$$A\underline{e}_i = \lambda_i \underline{e}_i$$

where $\underline{e}_i$ and λ_i may be complex.

In other words, a state-space model is diagonalized by a similarity transformation matrix $\mathbf{E}$ whose columns are given by the eigenvectors of A:

$$\mathbf{E} = [\underline{e}_1 \ \cdots \ \underline{e}_N]$$

- A system can be diagonalized whenever the eigenvectors of A are *linearly independent*.
  - This always holds when the system poles are *distinct*.
  - It may or may not hold when poles are *repeated*

State Space Diagonalization

- Suppose we solve the equation $A\underline{e}_i = \lambda_i \underline{e}_i$ and find N linearly independent eigenvectors of A
- Form the $N \times N$ matrix $\mathbf{E} = [\underline{e}_1 \ldots \underline{e}_N]$ having these eigenvectors as columns.
- Since the eigenvectors are linearly independent, $\mathbf{E}$ is *full rank* and can be *inverted*. This means it is *one-to-one* and qualifies as a *linear coordinate transformation matrix*.
- As derived above, the *transformed* state transition matrix is given by

$$\tilde{A} = \mathbf{E}^{-1} A \mathbf{E}$$

- Since $A\underline{e}_i = \lambda_i \underline{e}_i$, we have

$$A\mathbf{E} = \mathbf{E}\Lambda$$

  where Λ is a diagonal matrix having the (complex) eigenvalues of A along its diagonal.
- It follows that

$$\tilde{A} = \mathbf{E}^{-1} A \mathbf{E} = \mathbf{E}^{-1}\mathbf{E}\Lambda = \Lambda$$

  Thus, the new state transition matrix Λ is *diagonal* consisting of the eigenvalues of A.

- The transfer function of the diagonalized system is

$$\begin{aligned} H(z) &= \tilde{D} + \tilde{C}\left(zI - \Lambda\right)^{-1}\tilde{B} \\ &= \tilde{D} + \frac{\tilde{c}_1 b_1 z^{-1}}{1-\lambda_1 z^{-1}} + \frac{\tilde{c}_2\tilde{b}_2 z^{-1}}{1-\lambda_2 z^{-1}} + \cdots + \frac{\tilde{c}_N\tilde{b}_N z^{-1}}{1-\lambda_N z^{-1}} \\ &= \tilde{D} + \sum_{i=1}^{N} \frac{\tilde{c}_i\tilde{b}_i z^{-1}}{1-\lambda_i z^{-1}} \end{aligned}$$

  We see again that the diagonalized system (modal representation) consists of N *parallel one-pole systems*.

- Dynamic modes λ_i are *decoupled*

- Closely related to *partial-fraction expansion* of $H(z)$:
  - *Residue* of the ith pole is $c_i b_i$
  - Complex-conjugate poles may be combined to form real second-order sections

Finding the Eigenvalues of A in Practice

Small problems may be solved by hand by solving the system of equations

$$A\mathbf{E} = \mathbf{E}\Lambda$$

The Matlab built-in function `eig()` may be used to find the eigenvalues of A (system poles).

Example of State-Space Diagonalization

For the previous example

$$A \triangleq \begin{bmatrix} -\frac{1}{2} & -\frac{1}{3} \\ 1 & 0 \end{bmatrix} \qquad B \triangleq \begin{bmatrix} 1 \\ 0 \end{bmatrix} \qquad C \triangleq \begin{bmatrix} 3/2 \\ 8/3 \end{bmatrix} \qquad D \triangleq 1$$

we obtain the following in Matlab:

```
>> eig(A) % eigenvalues of state transition matrix

ans =
  -0.2500 + 0.5204i
  -0.2500 - 0.5204i

>> roots(den) % poles of transfer function H(z)

ans =
  -0.2500 + 0.5204i
  -0.2500 - 0.5204i

% They are the same, as they must be.
>> abs(roots(den)) % check stability

ans =
    0.5774
    0.5774
```

The system is stable.

Complex-conjugate poles are typically combined to produce real, second-order (2×2) parallel sections in the modal representation. Thus, our second-order example is already in *real* modal form. However, to illustrate the computations, let's obtain the eigenvectors and compute

the *complex* modal representation:

```
>> % Initial state space model from example above:
>> A = [-1/2, -1/3; 1, 0];
>> B = [1; 0];
>> C = [2-1/2, 3-1/3];
>> D = 1;

>> % Diagonalizing similarity transformation:
>> [E,L] = eig(A)  % [Evects, Evals] = eig(A)

E =

  -0.4507 - 0.2165i  -0.4507 + 0.2165i
        0 + 0.8660i        0 - 0.8660i


L =

  -0.2500 + 0.5204i        0
        0            -0.2500 - 0.5204i

>> A * E - E * L   % should be zero

ans =
```

```
1.0e-016 *
      0 + 0.2776i          0 - 0.2776i
      0                    0
```

Now form the complete diagonalized state-space model (complex):

```
>> Ei = inv(E); % matrix inverse
>> Ab = Ei*A*E  % diagonalized state xition mtx

Ab =
  -0.2500 + 0.5204i   0.0000 + 0.0000i
  -0.0000            -0.2500 - 0.5204i

>> Bb = Ei*B   % new input "routing vector"

Bb =
   -1.1094
   -1.1094

>> Cb = C*E    % new output linear combination

Cb =
  -0.6760 + 1.9846i  -0.6760 - 1.9846i

>> Db = D      % feed-through term unchanged

Db =
     1
```

Verify that we still have the same transfer function:

```
>> [numb,denb] = ss2tf(Ab,Bb,Cb,Db)

numb =
   1       2 + 0i   3 + 0i

denb =
   1    0.5 - 0i   0.3333

>> num = [1, 2, 3]; % original numerator
>> norm(num-numb)

ans =
  1.5543e-015

>> den = [1, 1/2, 1/3]; % original denominator
>> norm(den-denb)

ans =
  1.3597e-016
```

Close enough.

Properties of the Modal Representation

- The modal representation is not *unique* since B and C may be scaled in compensating ways to produce the same transfer function. Also, the diagonal elements of A may be permuted.
- For oscillatory systems, the λ_i are *complex*.
- If mode i is oscillatory and *undamped* (lossless), the state variable $x_i(n)$ oscillates *sinusoidally* at some frequency ω_i, where

$$\lambda_i = e^{j\omega_i T}$$

- In the damped oscillatory case, we have

$$\lambda_i = R_i e^{j\omega_i T}$$

  where R_i is the pole (eigenvalue) radius. For stability, we must have $|R_i| < 1$.
- In practice, we often prefer to combine complex-conjugate pole-pairs to form a real, "block-diagonal" system in which A has two-by-two real matrices along its diagonal.
- Matlab function `cdf2rdf()` can be used to convert complex diagonal form to real block-diagonal form.

- The input vector $\tilde{B}$ in the modal representation specifies *how the modes are excited* by the input signal $u(n)$:
$$x_i(n) = \tilde{b}_i u(n)$$
- The output vector $\tilde{C}$ in the modal representation specifies *how the modes are mixed* in the output signal $y(n)$:

$$y(n) = \tilde{C}\underline{\tilde{x}}(n) = \tilde{c}_1\tilde{x}_1(n) + \tilde{c}_2\tilde{x}_2(n) + \cdots + \tilde{c}_N\tilde{x}_N(n)$$

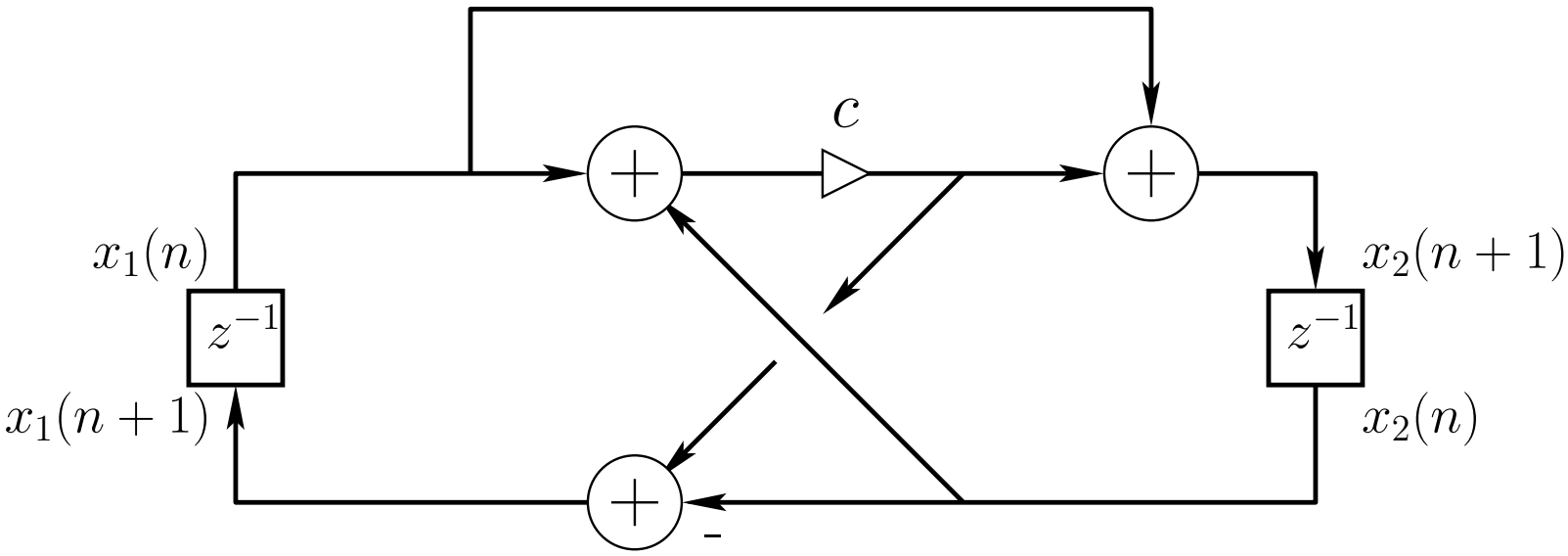
Repeated Poles

For repeated poles λ_i. we have two cases:

- If the corresponding eigenvectors are *linearly independent*, the modes are independent and can be decoupled (system can be diagonalized)
- Otherwise, if λ_i corresponds to k linearly *dependent* eigenvectors, the diagonalized system will contain a *Jordan block* of order k corresponding to that mode.
- Same as repeated roots in a partial-fraction expansion
- Impulse response looks like $n\lambda^n$, $n^2\lambda^n$, etc.

State-Space Analysis Example: The Digital Waveguide Oscillator

Let's use state-space analysis to determine the frequency of oscillation of the following system:



The second-order digital waveguide oscillator.

Note the assignments of unit-delay *outputs* to state variables $x_1(n)$ and $x_2(n)$.

We have

$$x_1(n+1) = c[x_1(n)+x_2(n)]-x_2(n) = c\,x_1(n)+(c-1)x_2(n)$$

and

$$x_2(n+1) = x_1(n)+c[x_1(n)+x_2(n)] = (1+c)x_1(n)+c\,x_2(n)$$

In matrix form, the state time-update can be written as

$$\left[\begin{array}{c} x_1(n+1) \\ x_2(n+1) \end{array}\right] = \underbrace{\left[\begin{array}{cc} c & c-1 \\ c+1 & c \end{array}\right]}_{A} \left[\begin{array}{c} x_1(n) \\ x_2(n) \end{array}\right]$$

or, in vector notation,

$$\underline{x}(n+1) = A\,\underline{x}(n)$$

We have two natural choices of output, corresponding to the choices $C = [1, 0]$ and $C = [0, 1]$:

$$\begin{aligned} y_1(n) &\stackrel{\Delta}{=} x_1(n) = [1, 0]\underline{x}(n) \\ y_2(n) &\stackrel{\Delta}{=} x_2(n) = [0, 1]\underline{x}(n) \end{aligned}$$

The determinant of a matrix is equal to the product of its eigenvalues. As a quick check, we find that the determinant of A is

$$|A| = c^2 - (c+1)(c-1) = c^2 - (c^2 - 1) = 1.$$

We expect this of any second-order real sinusoidal oscillator.

Oscillation Frequency

If this system is to generate a real sampled sinusoid at radian frequency ω, we must have modes of the form

$$\begin{aligned} \lambda_1 &= e^{j\omega T} \\ \lambda_2 &= e^{-j\omega T} \end{aligned}$$

Thus, we can determine the frequency of oscillation ω from eigenvalues λ_i of A.

Eigenstructure of A

Starting with

$$A\underline{e}_i = \lambda_i \underline{e}_i, \quad i = 1, 2$$

we get

$$\begin{bmatrix} c & c-1 \\ c+1 & c \end{bmatrix} \begin{bmatrix} 1 \\ \eta_i \end{bmatrix} = \begin{bmatrix} \lambda_i \\ \lambda_i \eta_i \end{bmatrix}.$$

- The first element of $\underline{e}_i$ is normalized arbitrarily to 1
- We have two equations in two unknowns:

$$\begin{aligned} c + \eta_i(c-1) &= \lambda_i \\ (1+c) + c\eta_i &= \lambda_i \eta_i \end{aligned}$$

- Substitute the first into the second to eliminate λ_i:

$$\begin{aligned} 1 + c + c\eta_i &= [c + \eta_i(c-1)]\eta_i = c\eta_i + \eta_i^2(c-1) \\ \Longrightarrow \quad 1 + c &= \eta_i^2(c-1) \\ \Longrightarrow \quad \eta_i &= \pm\sqrt{\frac{c+1}{c-1}} \end{aligned}$$

- We have found both eigenvectors:

$$\underline{e}_1 = \begin{bmatrix} 1 \\ \eta \end{bmatrix}, \quad \underline{e}_2 = \begin{bmatrix} 1 \\ -\eta \end{bmatrix}, \quad \text{where } \eta \stackrel{\Delta}{=} \sqrt{\frac{c+1}{c-1}}$$

They are linearly independent provided
$\eta \neq 0 \Leftrightarrow c \neq -1$ and finite provided $c \neq 1$.

- The eigenvalues are then

$$\lambda_i = c + \eta_i(c-1) = c \pm \sqrt{\frac{c+1}{c-1}(c-1)^2} = c \pm \sqrt{c^2-1}$$

- Assuming $|c| < 1$, they can be written as

$$\lambda_i = c \pm j\sqrt{1-c^2}$$

- With $c \in (-1,1)$, define $\theta = \arccos(c)$, i.e., $c \stackrel{\Delta}{=} \cos(\theta)$ and $\sqrt{1-c^2} = \sin(\theta)$.

- The eigenvalues become

$$\begin{aligned}\lambda_1 &= c + j\sqrt{1-c^2} = \cos(\theta) + j\sin(\theta) = e^{j\theta} \\ \lambda_2 &= c - j\sqrt{1-c^2} = \cos(\theta) - j\sin(\theta) = e^{-j\theta}\end{aligned}$$

  as expected.

We now have an explicit formula for the frequency of oscillation:

$$\omega = \theta/T = f_s \arccos(c),$$

where f_s denotes the sampling rate. Or,

$$\boxed{c = \cos(\omega T)}$$

The coefficient range $c \in (-1,1)$ corresponds to frequencies $f \in (-f_s/2, f_s/2)$.

We have shown that the example system oscillates sinusoidally at any desired digital frequency ω when $c = \cos(\omega T)$, where T denotes the sampling interval.

Choice of Output Signal and Initial Conditions

The two most natural choices of output signal are

$$\begin{aligned} y_1(n) &= [1,0]\underline{x}(n) = [1,0]\begin{bmatrix} 1 & 1 \\ \eta & -\eta \end{bmatrix}\underline{\tilde{x}}(n) \\ &= [1,1]\underline{\tilde{x}}(n) = \lambda_1^n\,\tilde{x}_1(0) + \lambda_2^n\,\tilde{x}_2(0) \\ y_2(n) &= [0,1]\underline{x}(n) = [0,1]\begin{bmatrix} 1 & 1 \\ \eta & -\eta \end{bmatrix}\underline{\tilde{x}}(n) \\ &= [\eta,-\eta]\underline{\tilde{x}}(n) = \eta\lambda_1^n\tilde{x}_1(0) - \eta\lambda_2^n\,\tilde{x}_2(0) \end{aligned}$$

The output signal from the first state variable $x_1(n)$ is

$$\begin{aligned} y_1(n) &= \lambda_1^n\,\tilde{x}_1(0) + \lambda_2^n\,\tilde{x}_2(0) \\ &= e^{j\omega nT}\tilde{x}_1(0) + e^{-j\omega nT}\tilde{x}_2(0) \end{aligned}$$

The *initial condition* $\underline{x}(0) = [1,0]^T$ corresponds to modal initial state

$$\underline{\tilde{x}}(0) = \mathbf{E}^{-1}\begin{bmatrix} 1 \\ 0 \end{bmatrix} = \frac{-1}{2e}\begin{bmatrix} -e & -1 \\ -e & 1 \end{bmatrix}\begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1/2 \\ 1/2 \end{bmatrix}$$

For this initialization, the output y_1 from the first state variable x_1 is simply

$$y_1(n) = \frac{e^{j\omega nT} + e^{-j\omega nT}}{2} = \cos(\omega nT)$$

Similarly $y_2(n)$ is proportional to $\sin(\omega nT)$
("phase quadrature" output).

On-Line Reader

A superset of these overheads, recommended for printing, exists online at

`http://ccrma.stanford.edu/~jos/filters/State_Space_Models.html`