

# Designing IIR filters with the Pade method

## Laboratory 2, SDP

### Table of contents

<b>1</b>	<b>Objective</b>	<b>1</b>
<b>2</b>	<b>Theoretical notions</b>	<b>1</b>
2.1	The Pade method . . . . .	2
2.2	Mathematical algorithm . . . . .	2
<b>3</b>	<b>Matlab notions</b>	<b>3</b>
3.1	Solve system of equations with <code>linsolve()</code> . . . . .	3
3.2	Calculate impulse response with <code>impz()</code> . . . . .	3
<b>4</b>	<b>Theoretical exercises</b>	<b>4</b>
<b>5</b>	<b>Practical exercises</b>	<b>4</b>
<b>6</b>	<b>Final questions</b>	<b>5</b>

## 1 Objective

Understanding the Pade method for designing IIR filters.

## 2 Theoretical notions

Designing a filter means finding the values of the coefficients from the numerator and denominator of the system function,  $b_0, b_1, \dots, b_M$  and  $a_1, a_2, \dots, a_N$ , so that the filter will have the desired characteristics.

$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2} + \dots + b_M z^{-M}}{1 + a_1 z^{-1} + a_2 z^{-2} + \dots + b_N z^{-N}}$$

## 2.1 The Pade method

In the Pade method, we want to find a system  $H(z)$  whose impulse response  $h[n]$  is similar to a desired impulse response  $h_d[n]$ . The desired impulse response  $h_d[n]$  can be, for example, the impulse response of an ideal low-pass, band-pass or high-pass filter.

The idea of the Pade method is as follows: find  $b_0, b_1, \dots, b_M$  and  $a_1, a_2, \dots, a_N$  so that the **first samples** of  $h[n]$  are **identical** to those of  $h_d[n]$ .

## 2.2 Mathematical algorithm

1. Start from the general difference equation of the system:

$$y[n] = -a_1 y[n-1] - \dots - a_N y[n-N] + b_0 x[n] + b_1 x[n-1] + \dots + b_M x[n-M]$$

2. If the input is  $\delta[n]$ , the output will be  $h[n]$ , and therefore:  $h[n] = -a_1 h[n-1] - \dots - a_N h[n-N] + b_0 \delta[n] + b_1 \delta[n-1] + \dots + b_M \delta[n-M]$
3. We match the first  $M + N + 1$  samples to those of  $h_d[n]$ . The number of samples is the same as the total number of coefficients to be obtained. Assume initial conditions  $h[-1] = h[-2] = \dots = 0$ , unless otherwise specified.

We obtain the following system:

$$\begin{aligned} h[0] &= b_0 & &= h_d[0] \\ h[1] &= -a_1 h_d[0] + b_1 & &= h_d[1] \\ &\dots & & \\ h[M] &= -a_1 h_d[M-1] - \dots - a_M h_d[0] + b_M & &= h_d[M] \\ h[M+1] &= -a_1 h_d[M] - \dots - a_M h_d[M+1-N] & &= h_d[M+1] \\ &\dots & & \\ h[M+N] &= -a_1 h_d[M+N-1] - \dots - a_M h_d[N] & &= h_d[M+N] \end{aligned}$$

This results in a system with  $M + N + 1$  equations, enough to find a total  $M + N + 1$  number of unknowns ( $a_i$  and  $b_i$ ).

4. Solve the system. When solving by hand, we can use the following observation:
  - The last  $N$  equations do not depend on  $b_i$ , only on  $a_i$ . These are solved as a separate, smaller system from which the  $a_i$  values are obtained.

- Then we move to the first  $M + 1$  equations. Replace  $a_i$  with the values found above and get  $b_i$ . Each equation provides a value for a  $b_i$ .

### 3 Matlab notions

#### 3.1 Solve system of equations with `linsolve()`

To solve a system of equations in Matlab, the `linsolve()` function can be used.

$$\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 5 \\ 7 & 8 & 9 \end{bmatrix} \cdot \begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix}$$

Matlab solution:

```
% Construct left vector and the system matrix
y = [1; 1; 1];
A = [1 2 3; 4 5 5; 7 8 9];

% Solve the system
x = linsolve(A, y);
```

Result:

```
x =

-1.0000
 1.0000
 0.0000
```

#### 3.2 Calculate impulse response with `impz()`

The `impz(b, a, n)` function is used, where:

- **b** = vector with coefficients from the numerator
- **a** = vector with coefficients from the denominator
- **n** = desired number of samples from  $h[n]$

$$H(z) = \frac{2 + 0.3z^{-1} - 0.4z^{-2}}{1 - 0.2z^{-1} + 0.1z^{-2}}$$

```
% Construct the coefficient vectors
b = [2, 0.3, -0.4];
a = [1, -0.2, 0.1];

% Find first 20 samples of the impulse response
h = impz(b, a, n);
```

## 4 Theoretical exercises

1. Use the Pade method to find out the parameters of the system with the following system function

$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}},$$

considering the desired impulse response:

$$h_d[n] = \left(\frac{1}{3}\right)^n \cos(n\pi) u[n] + u[n-3]$$

## 5 Practical exercises

1. Use Matlab to solve numerically the Pade system for the previous exercise, using the `linsolve()` function.

$$\begin{bmatrix} h_d[0] \\ h_d[1] \\ h_d[2] \\ h_d[3] \\ h_d[4] \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ -h_d[0] & 0 & 0 & 1 & 0 \\ -h_d[1] & -h_d[0] & 0 & 0 & 1 \\ -h_d[2] & -h_d[1] & 0 & 0 & 0 \\ -h_d[3] & -h_d[2] & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} a_1 \\ a_2 \\ b_0 \\ b_1 \\ b_2 \end{bmatrix}$$

2. Compute the first 10 samples of the impulse response, using `impz()`, for the filter designed in the previous exercise.

Check that the first 5 samples are the same as  $h_d[n]$ . How about the next ones?

3. Implement in Matlab a generic function for the Pade method, for any filter order and any  $h_d[n]$ :

```
function [b,a] = pademethod(order, hd)
...
end
```

The function shall have the following arguments:

- **order**: order of the desired filter
- **hd**: a vector with the desired impulse response (first samples)

The function shall return the coefficients of the designed filter:

- **b**: coefficients from the numerator
- **a**: the coefficients from the denominator

4. Check the result of Exercise 1 using this function.
5. Use the `impz()` function to find the impulse response of a system with

$$H(z) = \frac{1 - 1.7z^{-1} + 0.7z^{-2}}{1 + 1.3z^{-1} + 0.4z^{-2}}.$$

Then use the `pademethod()` function to approximate a 2nd order filter based on this impulse response. Do we find back the coefficients of the original filter?

How about if we approximate a 3rd order filter instead?

6. Load an audio signal into Matlab and filter it with the filter designed above. Play the filtered signal to the audio output of the system.

## 6 Final questions

1. TBD