Approximation of the Bessel function

A case study

D.S. Karachalios, I.V. Gosea, A.C. Antoulas

February 28, 2018

Affiliations:
MPI Magdeburg, Rice University Houston

12th Elgersburg Workshop

Partners:
Introduction and motivation.

Overview of the approximation methods we apply:

1. The Loewner Framework.
2. The AAA algorithm.
3. The Vector Fitting (VF) method.

Numerical results and comparison among the methods.

Further examples (Physical & Artificial).

Conclusion and further developments.
1. In general, model order reduction (MOR) is used to transform large, complex models \((n)\) of time dependent processes into smaller \((k \ll n)\), simpler models that are still capable of representing accurately the behavior of the original process under a variety of conditions.
1. In general, model order reduction (MOR) is used to transform large, complex models \( (n) \) of time dependent processes into smaller \( (k \ll n) \), simpler models that are still capable or representing accurately the behavior of the original process under a variety of conditions.

2. In particular, interpolatory model reduction methods construct reduced models whose \((\text{rational})\) transfer function matches that of the original system at selected interpolation points.
1. In general, model order reduction (MOR) is used to transform large, complex models \( (n) \) of time dependent processes into smaller \( (k \ll n) \), simpler models that are still capable or representing accurately the behavior of the original process under a variety of conditions.

2. In particular, interpolatory model reduction methods construct reduced models whose (rational) transfer function matches that of the original system at selected interpolation points.

3. **Irrational** transfer function correspond to infinite-dimensional dynamical system. Is it possible to approximate such a function (without performing any spatial discretization)?
1. In general, model order reduction (MOR) is used to transform large, complex models \( n \) of time dependent processes into smaller \( k \ll n \), simpler models that are still capable or representing accurately the behavior of the original process under a variety of conditions.

2. In particular, interpolatory model reduction methods construct reduced models whose (rational) transfer function matches that of the original system at selected interpolation points.

3. **Irrational** transfer function correspond to infinite-dimensional dynamical system. Is it possible to approximate such a function (without performing any spatial discretization)?
   - Main tool: The Loewner framework → data driven MOR method.
1. In general, model order reduction (MOR) is used to transform large, complex models \((n)\) of time dependent processes into smaller \((k \ll n)\), simpler models that are still capable or representing accurately the behavior of the original process under a variety of conditions.

2. In particular, interpolatory model reduction methods construct reduced models whose (rational) transfer function matches that of the original system at selected interpolation points.

3. Irrational transfer function correspond to infinite-dimensional dynamical system. Is it possible to approximate such a function (without performing any spatial discretization)?
   - Main tool: The Loewner framework \(\rightarrow\) data driven MOR method.
   - Computes an independent linear realization \((E,A,B,C)\).
Motivation

Irrational examples: [Curtain/...'09], [Filip/...'17], [Beattie/...'12], [Nakatsukasa/...'16].
Motivation

Irrational examples: [Curtain/...'09], [Filip/...'17], [Beattie/...'12], [Nakatsukasa/...'16].

- Euler - Bernoulli beam

![Frequency response of the original beam model](image-url)

- Hyperbolic sinus function

- Exponential function

- Inverse of the Bessel function
Motivation

Irrational examples: [Curtain/...'09], [Filip/...'17], [Beattie/...'12], [Nakatsukasa/...'16].

- Euler - Bernoulli beam

- Hyperbolic sinus function
Motivation

Irrational examples: [Curtain/...'09], [Filip/...'17], [Beattie/...'12], [Nakatsukasa/...'16].

- **Euler - Bernoulli beam**

- **Exponential function**

- **Hyperbolic sinus function**
Motivation

Irrational examples: [Curtain/...'09], [Filip/...'17], [Beattie/...'12], [Nakatsukasa/...'16].

- Euler - Bernoulli beam

- Exponential function

- Hyperbolic sinus function

- Inverse of the Bessel fct

Dimitris, karachalios@mpi-magdeburg.mpg.de

Irrational Approximation

4/26
Motivation and Methods

What if we don’t have access to the matrix realization or to the explicit form of the transfer function? (Only data provided)

Answer
Motivation and Methods

What if we don’t have access to the matrix realization or to the explicit form of the transfer function? (Only data provided)

Answer → Data Driven approach!
Motivation and Methods

- What if we don’t have access to the matrix realization or to the explicit form of the transfer function? (Only data provided)

  Answer → Data Driven approach!

- Take measurements and use:
Motivation and Methods

What if we don’t have access to the matrix realization or to the explicit form of the transfer function? (Only data provided)

Answer → Data Driven approach!

Take measurements and use:

1. The Loewner framework - [Mayo/Antoulas ’07]
Motivation and Methods

What if we don’t have access to the matrix realization or to the explicit form of the transfer function? (Only data provided)

Answer → Data Driven approach!

Take measurements and use:

1. The Loewner framework - [Mayo/Antoulas ’07]
2. The AAA algorithm - [Nakatsukasa/Sete/Trefethen ’16]
Motivation and Methods

What if we don’t have access to the matrix realization or to the explicit form of the transfer function? (Only data provided)

Answer → Data Driven approach!

Take measurements and use:

1. The Loewner framework - [Mayo/Antoulas '07]
2. The AAA algorithm - [Nakatsukasa/Sete/Trefethen '16]
3. The Vector Fitting method - [Gustavsen/Semlyen '99]
Overview of the methods

A simple SISO example - (spring - mass - damper)

\[ \begin{align*}
\ddot{x}(t) + \dot{x}(t) + kx(t) &= F(t) \\
\dot{x}_1 &= x_2 \\
m\dot{x}_2 &= -kx_1 - dx_2 + F \\
u &= F, \quad y = x_1
\end{align*} \]

\[ \begin{align*}
\dot{x}(t) &= Ax(t) + Bu(t) \\
y(t) &=Cx(t)
\end{align*} \]

Transfer Function

\[ H(s) = \frac{1}{ms^2 + ds + k} \]

We assume that: \(m=1, d=1\) and \(k=1\).
A simple SISO example - (spring - mass - damper)

Spring-mass-damper equation

\[ m\ddot{x}(t) + d\dot{x}(t) + kx(t) = F(t) \]

State variable: \( x_1 = x \),
\( x_2 = \dot{x} \), output \( y = x \).
\( \dot{x}_1 = x_2 \)
\( m\dot{x}_2 = -kx_1 - dx_2 + F \)
\( u = F, \ y = x_1 \)
Overview of the methods

A simple SISO example - (spring - mass - damper)

Spring-mass-damper equation

\[ m\ddot{x}(t) + d\dot{x}(t) + kx(t) = F(t) \]

State variable: \( x_1 = x \),
\( x_2 = \dot{x} \), output \( y = x \).
\( \dot{x}_1 = x_2 \)
\( m\dot{x}_2 = -kx_1 - dx_2 + F \)
\( u = F, \ y = x_1 \)

Compact form - System

\[ \dot{x}(t) = Ax(t) + Bu(t), \ y(t) = Cx(t) \]

Where \( x = \begin{bmatrix} x \\ \dot{x} \end{bmatrix} \),
\( A = \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{d}{m} \end{bmatrix} \),
\( B = \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} \),
\( C = [1 \ 0] \)
Overview of the methods

A simple SISO example - (spring - mass - damper)

**Spring-mass-damper equation**

\[ m \ddot{x}(t) + d \dot{x}(t) + kx(t) = F(t) \]

State variable: \( x_1 = x \), \( x_2 = \dot{x} \), output \( y = x \).

\[ \dot{x}_1 = x_2 \]
\[ m \dot{x}_2 = -kx_1 - dx_2 + F \]
\[ u = F, \ y = x_1 \]

**Compact form - System**

\[ \dot{x}(t) = Ax(t) + Bu(t), \ y(t) = Cx(t) \]

Where \( x = \begin{bmatrix} x \\ \dot{x} \end{bmatrix} \),
\[ A = \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{d}{m} \end{bmatrix}, \ B = \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix}, \ C = [1 \ 0] \]

**Transfer Function**

\[ H(s) = C(sI - A)^{-1}B = \frac{1}{ms^2 + ds + k} \]

We assume that: \( m=1 \), \( d=1 \) and \( k=1 \).
Overview of the methods

Method 1: The Loewner Framework - [Mayo/Antoulas ’07]

Theory
Given: a row array of pairs of complex numbers:

\[ \{(\omega_k, S_k) : k = 1, \ldots, N\} \]

with \( \omega_k \in \mathbb{C}, S_k \in \mathbb{C} \). We can partition the data in two sets:

- left data: \((\mu_j, \nu_j), j = 1, \ldots, p\)
- right data: \((\lambda_i, w_i), i = 1, \ldots, m\)

The objective is to find \( H(s) \in \mathbb{C} \) such that:

\[ H(\lambda_i) = w_i \text{ and } H(\mu_j) = \nu_j \]
Overview of the methods

Method 1: The Loewner Framework - [Mayo/Antoulas '07]

Theory

Given: a row array of pairs of complex numbers:

\[ \{ (\omega_k, S_k) : k = 1, ..., N \} \]

with \( \omega_k \in \mathbb{C}, S_k \in \mathbb{C} \). We can partition the data in two sets:

- left data: \((\mu_j, \nu_j), j = 1, ..., p\)
- right data: \((\lambda_i, w_i), i = 1, ..., m\)

The objective is to find \( H(s) \in \mathbb{C} \) such that:

\[ H(\lambda_i) = w_i \text{ and } H(\mu_j) = \nu_j \]

Example

Sample the transfer function of the spring-mass-damper:

\[ \omega = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \end{bmatrix} \]
\[ s = \begin{bmatrix} 1/3 & 1/7 & 1/13 & 1/21 & 1/31 & 1/43 & 1/57 & 1/73 \end{bmatrix} \]

- left data:
  - \( \mu = \begin{bmatrix} 1 & 3 & 5 & 7 \end{bmatrix} \)
  - \( \mathbf{V} = \begin{bmatrix} 1/3 & 1/13 & 1/31 & 1/57 \end{bmatrix} \)

- right data:
  - \( \lambda = \begin{bmatrix} 2 & 4 & 6 & 8 \end{bmatrix} \)
  - \( \mathbf{W} = \begin{bmatrix} 1/7 & 1/21 & 1/43 & 1/73 \end{bmatrix} \)
Overview of the methods

Method 1: The Loewner Framework - [Mayo/Antoulas '07]

Theory

The **Loewner matrix** $L \in \mathbb{C}^{p \times m}$, is defined as:

$$L = \begin{bmatrix}
\frac{v_1 - w_1}{\mu_1 - \lambda_1} & \cdots & \frac{v_1 - w_m}{\mu_1 - \lambda_m} \\
\vdots & \ddots & \vdots \\
\frac{v_p - w_1}{\mu_p - \lambda_1} & \cdots & \frac{v_p - w_m}{\mu_p - \lambda_m}
\end{bmatrix}$$

The **shifted Loewner matrix** $L_s \in \mathbb{C}^{p \times m}$, is defined as:

$$L_s = \begin{bmatrix}
\frac{\mu_1 v_1 - w_1 \lambda_1}{\mu_1 - \lambda_1} & \cdots & \frac{\mu_1 v_1 - w_m \lambda_m}{\mu_1 - \lambda_m} \\
\vdots & \ddots & \vdots \\
\frac{\mu_p v_p - w_1 \lambda_1}{\mu_p - \lambda_1} & \cdots & \frac{\mu_p v_p - w_m \lambda_m}{\mu_p - \lambda_m}
\end{bmatrix}$$
Overview of the methods

Method 1: The Loewner Framework - [Mayo/Antoulas '07]

Theory

The Loewner matrix \( L \in \mathbb{C}^{p \times m} \), is defined as:

\[
L = \begin{bmatrix}
\frac{v_1 - w_1}{\mu_1 - \lambda_1} & \ldots & \frac{v_1 - w_m}{\mu_1 - \lambda_m} \\
\vdots & \ddots & \vdots \\
\frac{v_p - w_1}{\mu_p - \lambda_1} & \ldots & \frac{v_p - w_m}{\mu_p - \lambda_m}
\end{bmatrix}
\]

The shifted Loewner matrix \( L_s \in \mathbb{C}^{p \times m} \), is defined as:

\[
L_s = \begin{bmatrix}
\frac{\mu_1 v_1 - w_1 \lambda_1}{\mu_1 - \lambda_1} & \ldots & \frac{\mu_1 v_1 - w_m \lambda_m}{\mu_1 - \lambda_m} \\
\vdots & \ddots & \vdots \\
\frac{\mu_p v_p - w_1 \lambda_1}{\mu_p - \lambda_1} & \ldots & \frac{\mu_p v_p - w_m \lambda_m}{\mu_p - \lambda_m}
\end{bmatrix}
\]

Example

The Loewner matrix

\[
L = \begin{bmatrix}
-\frac{4}{21} & -\frac{2}{21} & -\frac{8}{129} & -\frac{10}{219} \\
-\frac{6}{91} & -\frac{8}{273} & -\frac{10}{559} & -\frac{12}{949} \\
-\frac{8}{217} & -\frac{10}{651} & -\frac{12}{1333} & -\frac{14}{2263} \\
-\frac{10}{399} & -\frac{4}{399} & -\frac{14}{2451} & -\frac{16}{4161}
\end{bmatrix}
\]

The shifted Loewner matrix

\[
L_s = \begin{bmatrix}
-\frac{1}{21} & -\frac{1}{21} & -\frac{5}{129} & -\frac{7}{219} \\
-\frac{5}{91} & -\frac{11}{273} & -\frac{17}{559} & -\frac{23}{949} \\
-\frac{9}{217} & -\frac{19}{651} & -\frac{29}{1333} & -\frac{39}{2263} \\
-\frac{13}{399} & -\frac{3}{133} & -\frac{41}{2451} & -\frac{55}{4161}
\end{bmatrix}
\]
Method 1: The Loewner Framework - [Mayo/Antoulas '07]

The following results allow us to construct reduced order models.
Method 1: The Loewner Framework - [Mayo/Antoulas '07]

The following results allow us to construct reduced order models.

**Theorem**

If $(\mathbb{L}, \mathbb{L}_s)$ is regular, then $\{\hat{E} = -\mathbb{L}, \hat{A} = -\mathbb{L}_s, \hat{B} = \mathbb{V}, \hat{C} = \mathbb{W}\}$ is a realization of the data. Hence, $H(z) = \mathbb{W}(\mathbb{L}_s - z\mathbb{L})^{-1}\mathbb{V}$ is the required interpolant.
Overview of the methods

Method 1: The Loewner Framework - [Mayo/Antoulas '07]

The following results allow us to construct reduced order models.

**Theorem**

If \((L, L_s)\) is regular, then \(\{\hat{E} = -L, \hat{A} = -L_s, \hat{B} = V, \hat{C} = W\}\) is a realization of the data. Hence, \(H(z) = W(L_s - zL)^{-1}V\) is the required interpolant.

In the case of redundant data we perform a rank revealing SVD of:

\[
\begin{bmatrix}
L \\
L_s
\end{bmatrix}
\quad \text{or} \quad
\begin{bmatrix}
L \\
L_s
\end{bmatrix}
\]

then \(\begin{bmatrix}
L \\
L_s
\end{bmatrix} = Y\Sigma_\ell \tilde{X}^*\) and \(\begin{bmatrix}
L \\
L_s
\end{bmatrix} = \tilde{Y}\Sigma_r X^*\).
Method 1: The Loewner Framework - [Mayo/Antoulas '07]

The following results allow us to construct reduced order models.

**Theorem**

If \((L, L_s)\) is regular, then \(\{\hat{E} = -L, \hat{A} = -L_s, \hat{B} = V, \hat{C} = W\}\) is a realization of the data. Hence, \(H(z) = W(L_s - zL)^{-1}V\) is the required interpolant.

In the case of redundant data we perform a rank revealing SVD of:

\[
\begin{bmatrix}
L \\
L_s
\end{bmatrix} \quad \text{or} \quad \begin{bmatrix}
L \\
L_s
\end{bmatrix} = Y\Sigma_\ell\hat{X}^* \quad \text{and} \quad \begin{bmatrix}
L \\
L_s
\end{bmatrix} = \tilde{Y}\Sigma_rX^*.
\]

**Theorem**

The quadruple \(\{\hat{E} = -Y^*LX, \hat{A} = -Y^*L_sX, \hat{B} = Y^*V, \hat{C} = WX\}\), is the realization of an approximate data interpolant.
Overview of the methods

Method 1: The Loewner Framework - [Mayo/Antoulas '07]

The following results allow us to construct reduced order models.

**Theorem**

If \((\mathbb{L}, \mathbb{L}_s)\) is regular, then \(\{\hat{E} = -\mathbb{L}, \hat{A} = -\mathbb{L}_s, \hat{B} = \mathbb{V}, \hat{C} = \mathbb{W}\}\) is a realization of the data. Hence, \(H(z) = \mathbb{W}(\mathbb{L}_s - z\mathbb{L})^{-1}\mathbb{V}\) is the required interpolant.

In the case of redundant data we perform a rank revealing SVD of:

\[
\begin{bmatrix}
\mathbb{L} \\
\mathbb{L}_s
\end{bmatrix}
\]

then \(\begin{bmatrix}
\mathbb{L} \\
\mathbb{L}_s
\end{bmatrix} = \mathbb{Y}\Sigma_\ell \mathbb{X}^*\) and \(\begin{bmatrix}
\mathbb{L} \\
\mathbb{L}_s
\end{bmatrix} = \tilde{\mathbb{Y}}\Sigma_r \mathbb{X}^*\).

**Theorem**

The quadruple \(\{\hat{E} = -\mathbb{Y}^*\mathbb{L}\mathbb{X}, \hat{A} = -\mathbb{Y}^*\mathbb{L}_s\mathbb{X}, \hat{B} = \mathbb{Y}^*\mathbb{V}, \hat{C} = \mathbb{W}\mathbb{X}\}\), is the realization of an approximate data interpolant.

**Remark:** Above is the SISO case. Moreover, the Loewner Framework can be applied also for MIMO case via tangential interpolation.
Overview of the methods

Method 1: The Loewner Framework - spring-mass-damper

Example

We compute the singular values for the augmented matrix \([L \ L_s]\):

\[
\sigma(L \ L_s) = \begin{bmatrix}
0.27197 & 0.063812 \\
4.522 \cdot 10^{-18} & 3.3768 \\
4.522 \cdot 10^{-17} & 0.27197 \\
\end{bmatrix},
\]

\[\rightarrow \text{rank} = 2.\]

Reduce the dimension of the Loewner model from 4 to dimension 2;

The reduced model \((\hat{C}, \hat{E}, \hat{A}, \hat{B})\) is obtained by projecting the raw model \((W, L, L_s, V)\):

\[
\hat{C} = WX, \quad \hat{E} = -Y^*LX, \\
\hat{A} = -Y^*L_sX, \quad \hat{B} = Y^*V.
\]

\[
\hat{\dot{z}} = \hat{A}z + \hat{B}u \\
y = \hat{C}z
\]

\[
H_r(s) = \hat{C}(s\hat{E} - \hat{A})^{-1}\hat{B} \left(1 + s + s^2\right)
\]

Remark: Poles \(= \text{eig}(\hat{A}, \hat{E}) = (-0.5 + 0.86603i, -0.5 - 0.86603i)\),

Zeros \(= \text{eig}(\begin{bmatrix} \hat{A} & \hat{B} \\ \hat{E} & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}) \approx \inf(3, 1)\).
Overview of the methods

Method 1: The Loewner Framework - spring-mass-damper

Example

We compute the singular values for the augmented matrix \([\mathbb{L} \quad \mathbb{L}_s]\):

\[
\sigma([\mathbb{L} \quad \mathbb{L}_s]) = \begin{pmatrix}
0.27197 \\
0.063812 \\
3.3768 \cdot 10^{-17} \\
4.522 \cdot 10^{-18}
\end{pmatrix}, \quad \rightarrow \text{rank} = 2.
\]
Overview of the methods

Method 1: The Loewner Framework - spring-mass-damper

Example

We compute the singular values for the augmented matrix $[\mathbb{L} \quad \mathbb{L}_s]$:

$$\sigma([\mathbb{L} \quad \mathbb{L}_s]) = \begin{pmatrix} 0.27197 \\ 0.063812 \\ 3.3768 \cdot 10^{-17} \\ 4.522 \cdot 10^{-18} \end{pmatrix}, \quad \rightarrow \quad \text{rank} = 2.$$ 

Reduce the dimension of the Loewner model from 4 to dimension 2;
Overview of the methods

Method 1: The Loewner Framework - spring-mass-damper

Example

We compute the singular values for the augmented matrix \([\mathbb{L} \quad \mathbb{L}_s]\):

\[
\sigma([\mathbb{L} \quad \mathbb{L}_s]) = \begin{pmatrix}
0.27197 \\
0.063812 \\
3.3768 \times 10^{-17} \\
4.522 \times 10^{-18}
\end{pmatrix}, \quad \rightarrow \quad \text{rank} = 2.
\]

Reduce the dimension of the Loewner model from 4 to dimension 2;

The reduced model \((\hat{\mathbb{C}}, \hat{\mathbb{E}}, \hat{\mathbb{A}}, \hat{\mathbb{B}})\) is obtained by projecting the raw model \((\mathbb{W}, \mathbb{L}, \mathbb{L}_s, \mathbb{V})\):

\[
\hat{\mathbb{C}} = \mathbb{W}X, \quad \hat{\mathbb{E}} = -Y^*\mathbb{L}X, \quad \hat{\mathbb{A}} = -Y^*\mathbb{L}_sX, \quad \hat{\mathbb{B}} = Y^*\mathbb{V}.
\]
Overview of the methods

Method 1: The Loewner Framework - spring-mass-damper

Example

We compute the singular values for the augmented matrix $[L \quad L_s]$:

$$\sigma([L \quad L_s]) = \begin{pmatrix} 0.27197 \\ 0.063812 \\ 3.3768 \cdot 10^{-17} \\ 4.522 \cdot 10^{-18} \end{pmatrix}, \quad \rightarrow \quad \text{rank} = 2.$$ 

Reduce the dimension of the Loewner model from 4 to dimension 2;

The reduced model $(\hat{C}, \hat{E}, \hat{A}, \hat{B})$ is obtained by projecting the raw model $(W, L, L_s, V)$: $\hat{C} = WX, \hat{E} = -Y^*LX, \hat{A} = -Y^*L_sX, \hat{B} = Y^*V$.

$$\begin{cases} \hat{E}\dot{z} = \hat{A}z + \hat{B}u \\ y = \hat{C}z \end{cases} \quad \rightarrow \quad H_r(s) = \hat{C}(s\hat{E} - \hat{A})^{-1}\hat{B} \left( \frac{1}{s^2 + s + 1} \right)$$
Method 1: The Loewner Framework - spring-mass-damper

Example

We compute the singular values for the augmented matrix $[L \quad L_s]$:

$$\sigma([L \quad L_s]) = \begin{pmatrix} 0.27197 \\ 0.063812 \\ 3.3768 \cdot 10^{-17} \\ 4.522 \cdot 10^{-18} \end{pmatrix}, \quad \rightarrow \quad \text{rank} = 2.$$

Reduce the dimension of the Loewner model from 4 to dimension 2;

The reduced model $(\hat{C}, \hat{E}, \hat{A}, \hat{B})$ is obtained by projecting the raw model $(W, L, L_s, V)$: $\hat{C} = WX$, $\hat{E} = -Y^*LX$, $\hat{A} = -Y^*L_sX$, $\hat{B} = Y^*V$.

$$\begin{cases} \hat{E}z = \hat{A}z + \hat{B}u \\ y = \hat{C}z \end{cases} \quad \rightarrow \quad H_r(s) = \hat{C}(s\hat{E} - \hat{A})^{-1}\hat{B} \left( \frac{1}{s^2 + s + 1} \right)$$

Remark: Poles $= \text{eig}(\hat{A}, \hat{E}) = \begin{pmatrix} -0.5 + 0.86603i \\ -0.5 - 0.86603i \end{pmatrix}$, Zeros $= \text{eig} \left( \begin{bmatrix} \hat{A} & \hat{B} \\ \hat{C} & 0 \end{bmatrix}, \begin{bmatrix} \hat{E} & 0 \\ 0 & 0 \end{bmatrix} \right) \approx \inf_{3,1}$
Method 2: The AAA algorithm - [Nakatsukasa/Sete/Trefethen '16]

- The algorithm uses Barycentric representation of interpolants.

\[
Z = [s_1, ..., s_n]^T, \quad F = [f_1, ..., f_n]^T
\]
Method 2: The AAA algorithm - [Nakatsukasa/Sete/Trefethen '16]

- The algorithm uses **Barycentric representation** of interpolants.

\[
Z = [s_1, \ldots, s_n]^T, \quad F = [f_1, \ldots, f_n]^T \quad \Rightarrow \quad r(s) = \frac{n(s)}{d(s)} = \frac{\sum_{k=1}^{r} \frac{w_k f_k}{s-s_k}}{\sum_{k=1}^{r} \frac{w_k}{s-s_k}}.
\]
Method 2: The AAA algorithm - [Nakatsukasa/Sete/Trefethen '16]

- The algorithm uses **Barycentric representation** of interpolants.

\[
Z = [s_1, \ldots, s_n]^T, \quad F = [f_1, \ldots, f_n]^T \Rightarrow r(s) = \frac{n(s)}{d(s)} = \frac{\sum_{k=1}^{r} \frac{w_k f_k}{s-s_k}}{\sum_{k=1}^{r} \frac{w_k}{s-s_k}}.
\]

- \( R = \text{mean}(F), \quad e = [1, \ldots, 1], \quad J = [1, \ldots, j, \ldots, n] \)
Method 2: The AAA algorithm - [Nakatsukasa/Sete/Trefethen '16]

- The algorithm uses **Barycentric representation** of interpolants.

\[
Z = [s_1, ..., s_n]^T, \quad F = [f_1, ..., f_n]^T \Rightarrow r(s) = \frac{n(s)}{d(s)} = \sum_{k=1}^{r} \frac{w_k f_k}{s-s_k}.
\]

- \( R = \text{mean}(F), \quad e = [1, ..., 1], \quad J = [1, ..., j, ..., n] \)
- for \( m = 1, ..., r \ll n \)
Method 2: The AAA algorithm - [Nakatsukasa/Sete/Trefethen ’16]

- The algorithm uses **Barycentric representation** of interpolants.

\[
Z = [s_1, ..., s_n]^T, \quad F = [f_1, ..., f_n]^T \Rightarrow r(s) = \frac{n(s)}{d(s)} = \frac{\sum_{k=1}^{r} w_k f_k}{\sum_{k=1}^{r} w_k (s-s_k)}.
\]

- \( R = \text{mean}(F), \quad e = [1, ..., 1], \quad J = [1, ..., j, ..., n] \)

- **for** \( m = 1, ..., r \ll n \)
  1. \([\sim, j] = \text{max}|F - Re^T| \& J(J == j) = []\)
Overview of the methods

Method 2: The AAA algorithm - [Nakatsukasa/Sete/Trefethen ’16]

- The algorithm uses **Barycentric representation** of interpolants.

\[
Z = [s_1, \ldots, s_n]^T, \quad F = [f_1, \ldots, f_n]^T \quad \Rightarrow r(s) = \frac{n(s)}{d(s)} = \frac{\sum_{k=1}^{r} \frac{w_k f_k}{s-s_k}}{\sum_{k=1}^{r} w_k s-s_k} .
\]

- \( R = \text{mean}(F) \), \( e = [1, \ldots, 1] \), \( J = [1, \ldots, j, \ldots, n] \)
- for \( m = 1, \ldots, r \ll n \)

1. \( [\sim, j] = \max |F - Re^T| \) & \( J(J==j) = [] \)
2. \( z = [z, Z(j)], \quad f = [f, F(j)] \rightarrow \text{update supports points and data} \)
Method 2: The AAA algorithm - [Nakatsukasa/Sete/Trefethen '16]

- The algorithm uses **Barycentric representation** of interpolants.

\[
Z = [s_1, ..., s_n]^T, \quad F = [f_1, ..., f_n]^T \implies r(s) = \frac{n(s)}{d(s)} = \frac{\sum_{k=1}^{r} \frac{w_k f_k}{s-s_k}}{\sum_{k=1}^{r} \frac{w_k}{s-s_k}}.
\]

- \( R = \text{mean}(F), \quad e = [1, ..., 1], \quad J = [1, ..., j, ..., n] \)

- **for** \( m = 1, ..., r \ll n \)
  1. \([\sim, j] = \max|F - R e^T| & J(J==j) = []\)
  2. \( z = [z, Z(j)], \quad f = [f, F(j)] \rightarrow \text{update supports points and data} \)
  3. \( C = [C, \frac{1}{Z-Z(j)}] \rightarrow \text{next column vector of the Cauchy matrix} \)
Method 2: The AAA algorithm - [Nakatsukasa/Sete/Trefethen ’16]

The algorithm uses **Barycentric representation** of interpolants.

\[ Z = [s_1, \ldots, s_n]^T, \quad F = [f_1, \ldots, f_n]^T \Rightarrow r(s) = \frac{n(s)}{d(s)} = \frac{\sum_{k=1}^{r} \frac{w_k f_k}{s - s_k}}{\sum_{k=1}^{r} \frac{w_k}{s - s_k}}. \]

\[ R = \text{mean}(F), \quad e = [1, \ldots, 1], \quad J = [1, \ldots, j, \ldots, n] \]

**for** \( m = 1, \ldots, r \ll n \)

1. \( [\sim, j] = \max|F - Re^T| \& J(J == j) = [] \)
2. \( z = [z, Z(j)], \quad f = [f, F(j)] \rightarrow \text{update supports points and data} \)
3. \( C = [C, \frac{1}{Z - Z(j)}] \rightarrow \text{next column vector of the Cauchy matrix} \)
4. \( A = SF \ast C - C \ast Sf \rightarrow \text{Loewner matrix via scaling matrices } SF, Sf \)
Overview of the methods

Method 2: The AAA algorithm - [Nakatsukasa/Sete/Trefethen '16]

- The algorithm uses **Barycentric representation** of interpolants.

\[ Z = [s_1, ..., s_n]^T, \quad F = [f_1, ..., f_n]^T \Rightarrow r(s) = \frac{n(s)}{d(s)} = \frac{\sum_{k=1}^{r} \frac{w_k f_k}{s - s_k}}{\sum_{k=1}^{r} \frac{w_k}{s - s_k}}. \]

\[ R = \text{mean}(F), \quad e = [1, ..., 1], \quad J = [1, ..., j, ..., n] \]

- **for** \( m = 1, ..., r \ll n \)
  1. \([\sim, j] = \max |F - Re^T| \& J(J == j) = []\)
  2. \( z = [z, Z(j)], \quad f = [f, F(j)] \rightarrow \) update supports points and data
  3. \( C = [C, \frac{1}{z - Z(j)}] \rightarrow \) next column vector of the Cauchy matrix
  4. \( A = SF \ast C - C \ast Sf \rightarrow \) Loewner matrix via scaling matrices \( SF, Sf \)
  5. \([\sim, \sim, V] = \text{svd}(A(J,:)), \quad w = V(:, m) \rightarrow \) weight vector - min sv
Method 2: The AAA algorithm - [Nakatsukasa/Sete/Trefethen '16]

- The algorithm uses **Barycentric representation** of interpolants.

\[ Z = [s_1, \ldots, s_n]^T, \quad F = [f_1, \ldots, f_n]^T \Rightarrow r(s) = \frac{n(s)}{d(s)} = \sum_{k=1}^{r} \frac{w_k f_k}{s-s_k} / \sum_{k=1}^{r} \frac{w_k}{s-s_k} \]

- \( R = \text{mean}(F), \quad e = [1, \ldots, 1], \quad J = [1, \ldots, j, \ldots, n] \)

- for \( m = 1, \ldots, r \ll n \)
  1. \( [\sim, j] = \max |F - Re^T| \& J(J == j) = [] \)
  2. \( z = [z, Z(j)], \quad f = [f, F(j)] \rightarrow \) update supports points and data
  3. \( C = [C, \frac{1}{z-Z(j)}] \rightarrow \) next column vector of the Cauchy matrix
  4. \( A = SF \ast C - C \ast SF \rightarrow \) Loewner matrix via scaling matrices \( SF, Sf \)
  5. \( [\sim, \sim, V] = \text{svd}(A(J,:)), \quad w = V(:,m) \rightarrow \) weight vector - min sv
  6. \( N = C \ast (w \ast f), \quad D = C\ast w, \quad R = F, \quad R(J) = N(J)./D(J) \)
Overview of the methods

Method 2: The AAA algorithm - [Nakatsukasa/Sete/Trefethen '16]

- The algorithm uses **Barycentric representation** of interpolants.

\[ Z = [s_1, \ldots, s_n]^T, \quad F = [f_1, \ldots, f_n]^T \Rightarrow r(s) = \frac{n(s)}{d(s)} = \frac{\sum_{k=1}^r w_k f_k}{\sum_{k=1}^r w_k} \cdot \]

- \( R = \text{mean}(F) \), \( e = [1, \ldots, 1] \), \( J = [1, \ldots, j, \ldots, n] \)
- **for** \( m = 1, \ldots, r \ll n \)
  1. \([\sim, j] = \max|F - Re^T| \& J(J == j) = []\)
  2. \( z = [z, Z(j)], \quad f = [f, F(j)] \rightarrow \text{update supports points and data} \)
  3. \( C = [C, \frac{1}{z-Z(j)}] \rightarrow \text{next column vector of the Cauchy matrix} \)
  4. \( A = SF \ast C - C \ast Sf \rightarrow \text{Loewner matrix via scaling matrices } SF, Sf \)
  5. \([\sim, \sim, V] = \text{svd}(A(J,:)), \quad w = V(:,m) \rightarrow \text{weight vector - min sv} \)
  6. \( N = C \ast (w \ast f), \quad D = C \ast w, \quad R = F, \quad R(J) = N(J)/D(J) \)
  7. \( \text{Error} = ||F - R||_{\text{inf}} \rightarrow \text{tolerance (reached?)}\)
Overview of the methods

Method 2: The AAA algorithm - [Nakatsukasa/Sete/Trefethen '16]

- The algorithm uses **Barycentric representation** of interpolants.

\[
Z = [s_1, ..., s_n]^T, \quad F = [f_1, ..., f_n]^T \Rightarrow r(s) = \frac{n(s)}{d(s)} = \frac{\sum_{k=1}^{r} \frac{w_k f_k}{s-s_k}}{\sum_{k=1}^{r} \frac{w_k}{s-s_k}}.
\]

- \(R = \text{mean}(F), \quad e = [1, ..., 1], \quad J = [1, ..., j, ..., n]\)

- **for** \(m = 1, ..., r \ll n\)
  1. \([\sim, j] = \text{max} |F - Re^T| \& J(J == j) = []\)
  2. \(z = [z, Z(j)], \quad f = [f, F(j)] \rightarrow \text{update supports points and data}\)
  3. \(C = [C, \frac{1}{Z-Z(j)}] \rightarrow \text{next column vector of the Cauchy matrix}\)
  4. \(A = SF \ast C - C \ast Sf \rightarrow \text{Loewner matrix via scaling matrices } SF, Sf\)
  5. \([\sim, \sim, V] = \text{svd}(A(J,:)), \quad w = V(:, m) \rightarrow \text{weight vector - min sv}\)
  6. \(N = C \ast (w \ast f), \quad D = C \ast w, \quad R = F, \quad R(J) = N(J)/D(J)\)
  7. \(\text{Error} = ||F - R||_{\text{inf}} \rightarrow \text{tolerance (reached?)}\)

- **end**
Method 3: Vector Fitting (VF) - [Gustavsen/Semlyen '99]

VF aims at finding an approximant expressed in pole-residue form, as

\[ f(s) = \sum_{n=1}^{r} c_n s^{-a_n} + d + sh. \]

VF solves the above problem as a linear problem in two stages.

1. Stage: Pole identification

Specify the starting poles \( \bar{a}_n, n = 1, \ldots, r \).

Then multiply with an unknown function \( \sigma(s) \).

\[ \sigma(s) f(s) = \sum_{n=1}^{r} c_n s^{-\bar{a}_n} + d + sh, \]

\[ \sigma(s) = \sum_{n=1}^{r} \bar{c}_n s^{-\bar{a}_n} + \sum_{n=1}^{r} (\bar{c}_n s^{-\bar{a}_n} - \bar{a}_n + 1) f(s) = \sum_{n=1}^{r} c_n s^{-\bar{a}_n} + d + sh. \]

Overdetermined \( Ax = b \) with unknowns: \( c_n, d, h, \bar{c}_n \).

2. Stage: Residue identification

We can solve the original problem with the zeros of \( \sigma(s) \) as a new poles for \( f(s) \).

Overdetermined \( Ax = b \) with unknowns: \( c_n, d, h \).

Repeat until converge.
Method 3: Vector Fitting (VF) - [Gustavsen/Semlyen '99]

VF aims at finding an approximant expressed in pole-residue form, as

\[ f(s) = \sum_{n=1}^{r} \frac{c_n}{s - a_n} + d + sh. \]

1. Stage: Pole identification
   - Specify the starting poles \( \bar{a}_n, n = 1, \ldots, r \).
   - Then multiply with an unknown function \( \sigma(s) \).

\[ \sigma(s)f(s) = \sum_{n=1}^{r} c_n s - \bar{a}_n + d + sh, \]

\[ \sigma(s) = \sum_{n=1}^{r} \bar{c}_n s - \bar{a}_n + 1 \sum_{n=1}^{r} (\bar{c}_n s - \bar{a}_n + 1) f(s) = \sum_{n=1}^{r} c_n s - \bar{a}_n + d + sh. \]

2. Stage: Residue identification
   - We can solve the original problem with the zeros of \( \sigma(s) \) as new poles \( a_n \) for \( f(s) \).

\[ \text{Overdetermined } \mathbf{A} \mathbf{x} = \mathbf{b} \text{ with unknowns: } c_n, d, h. \]

Repeat until converge.
Method 3: Vector Fitting (VF) - [Gustavsen/Semlyen '99]

- VF aims at finding an approximant expressed in pole-residue form, as
  \[ f(s) = \sum_{n=1}^{r} \frac{c_n}{s - a_n} + d + sh. \]
- VF solves the above problem as a linear problem in two stages.
Overview of the methods

Method 3: Vector Fitting (VF) - [Gustavsen/Semlyen '99]

- VF aims at finding an approximant expressed in pole-residue form, as

\[ f(s) = \sum_{n=1}^{r} \frac{c_n}{s - a_n} + d + sh. \]

- VF solves the above problem as a linear problem in two stages.
  1. **Stage: Pole identification**: Specify the starting poles \( \bar{a}_n, n = 1..., r \).
Method 3: Vector Fitting (VF) - [Gustavsen/Semlyen '99]

VF aims at finding an approximant expressed in pole-residue form, as

\[ f(s) = \sum_{n=1}^{r} \frac{c_n}{s - a_n} + d + sh. \]

VF solves the above problem as a linear problem in two stages.

1. Stage: **Pole identification**: Specify the starting poles \( \bar{a}_n, n = 1..., r \). Then multiply with an unknown function \( \sigma(s) \).

\[ \sigma(s)f(s) = \sum_{n=1}^{r} \frac{c_n}{s - \bar{a}_n} + d + sh \]

\[ \sigma(s) = \sum_{n=1}^{r} \frac{c_n}{s - \bar{a}_n} + 1 \]
Method 3: Vector Fitting (VF) - [Gustavsen/Semlyen '99]

■ VF aims at finding an approximant expressed in pole-residue form, as

\[ f(s) = \sum_{n=1}^{r} \frac{c_n}{s - a_n} + d + sh. \]

■ VF solves the above problem as a linear problem in two stages.
  1. Stage: **Pole identification**: Specify the starting poles \( \bar{a}_n, n = 1..., r \). Then multiply with an unknown function \( \sigma(s) \).

\[ \sigma(s)f(s) = \sum_{n=1}^{r} \frac{c_n}{s - \bar{a}_n} + d + sh \quad , \quad \sigma(s) = \sum_{n=1}^{r} \frac{\bar{c}_n}{s - \bar{a}_n} + 1 \]

\[ \sum_{n=1}^{r} \left( \frac{\bar{c}_n}{s - \bar{a}_n} + 1 \right)f(s) = \sum_{n=1}^{r} \frac{c_n}{s - \bar{a}_n} + d + sh \]
Overview of the methods

Method 3: Vector Fitting (VF) - [Gustavsen/Semlyen '99]

VF aims at finding an approximant expressed in pole-residue form, as

\[ f(s) = \sum_{n=1}^{r} \frac{c_n}{s-a_n} + d + sh. \]

VF solves the above problem as a linear problem in two stages.

1. Stage: **Pole identification**: Specify the starting poles \( \bar{a}_n, n = 1..., r \). Then multiply with an unknown function \( \sigma(s) \).

   \[ \sigma(s)f(s) = \sum_{n=1}^{r} \frac{c_n}{s-a_n} + d + sh \quad , \quad \sigma(s) = \sum_{n=1}^{r} \frac{\bar{c}_n}{s-a_n} + 1 \]

   \[ \sum_{n=1}^{r} \left( \frac{\bar{c}_n}{s-a_n} + 1 \right) f(s) = \sum_{n=1}^{r} \frac{c_n}{s-a_n} + d + sh \]

2. Stage: **Residue identification**: We can solve the original problem with the zeros of \( \sigma(s) \) as a new poles \( a_n \) for \( f(s) \).

   Overdetermined \( Ax = b \) with unknowns: \( c_n, d, h, \bar{c}_n \).
Method 3: Vector Fitting (VF) - [Gustavsen/Semlyen '99]

- VF aims at finding an approximant expressed in pole-residue form, as

\[ f(s) = \sum_{n=1}^{r} \frac{c_n}{s - a_n} + d + sh. \]

- VF solves the above problem as a linear problem in two stages.
  
  1. Stage: **Pole identification**: Specify the starting poles \( \bar{a}_n, n = 1..., r \). Then multiply with an unknown function \( \sigma(s) \).
   
   \[ \sigma(s)f(s) = \sum_{n=1}^{r} \frac{\bar{c}_n}{s - \bar{a}_n} + d + sh \], \( \sigma(s) = \sum_{n=1}^{r} \frac{\bar{c}_n}{s - \bar{a}_n} + 1 \)
   
   \[ \sum_{n=1}^{r} \left( \frac{\bar{c}_n}{s - \bar{a}_n} + 1 \right)f(s) = \sum_{n=1}^{r} \frac{c_n}{s - a_n} + d + sh \]
   
   Overdetermined \( Ax = b \) with unknowns: \( c_n, d, h, \bar{c}_n \).

  2. Stage: **Residue identification**: We can solve the original problem with the zeros of \( \sigma(s) \) as a new poles \( a_n \) for \( f(s) \).
Method 3: Vector Fitting (VF) - [Gustavsen/Semlyen '99]

- VF aims at finding an approximant expressed in pole-residue form, as

\[ f(s) = \sum_{n=1}^{r} \frac{c_n}{s - a_n} + d + sh. \]

- VF solves the above problem as a linear problem in two stages.
  1. **Stage: Pole identification**: Specify the starting poles \( \bar{a}_n, n = 1..., r \). Then multiply with an unknown function \( \sigma(s) \).
    - \( \sigma(s) f(s) = \sum_{n=1}^{r} \frac{\bar{c}_n}{s - \bar{a}_n} + d + sh \), \( \sigma(s) = \sum_{n=1}^{r} \frac{\bar{c}_n}{s - \bar{a}_n} + 1 \)
    - \( \sum_{n=1}^{r} \left( \frac{\bar{c}_n}{s - \bar{a}_n} + 1 \right) f(s) = \sum_{n=1}^{r} \frac{c_n}{s - a_n} + d + sh \)
    - Overdetermined \( Ax = b \) with unknowns: \( c_n, d, h, \bar{c}_n \).
  2. **Stage: Residue identification**: We can solve the original problem with the zeros of \( \sigma(s) \) as a new poles \( a_n \) for \( f(s) \).
    - Overdetermined \( Ax = b \) with unknowns: \( c_n, d, h \).
Overview of the methods

Method 3: Vector Fitting (VF) - [Gustavsen/Semlyen '99]

Vector Fitting (VF) aims at finding an approximant expressed in pole-residue form, as

\[ f(s) = \sum_{n=1}^{r} \frac{c_n}{s-a_n} + d + sh. \]

VF solves the above problem as a linear problem in two stages.

1. Stage: **Pole identification**: Specify the starting poles \( \bar{a}_n, n = 1...r \). Then multiply with an unknown function \( \sigma(s) \).

   \[ \sigma(s)f(s) = \sum_{n=1}^{r} \frac{c_n}{s-\bar{a}_n} + d + sh \quad , \quad \sigma(s) = \sum_{n=1}^{r} \frac{\bar{c}_n}{s-\bar{a}_n} + 1 \]

   \[ \sum_{n=1}^{r} \left( \frac{\bar{c}_n}{s-\bar{a}_n} + 1 \right)f(s) = \sum_{n=1}^{r} \frac{c_n}{s-\bar{a}_n} + d + sh \]

   Overdetermined \( Ax = b \) with unknowns: \( c_n, d, h, \bar{c}_n \).

2. Stage: **Residue identification**: We can solve the original problem with the zeros of \( \sigma(s) \) as a new poles \( a_n \) for \( f(s) \).

   Overdetermined \( Ax = b \) with unknowns: \( c_n, d, h \).

   repeat until converge.
The Bessel function of the first kind:

\[ J_n(s) = \frac{1}{2\pi i} \oint e^{\left(\frac{s}{2}\right)(t-\frac{1}{t})} t^{-n-1} \, dt, \quad s \in \mathbb{C} \]
The Bessel function of the first kind:

\[ J_n(s) = \frac{1}{2\pi i} \oint e^{(s/2)(t^{-1})} t^{-n-1} dt, \quad s \in \mathbb{C} \]

The aim is to approximate \( \frac{1}{J_0(s)} \) over \( \Omega = [0, 10] \times [-1, 1] \).
The Bessel function of the first kind:

\[ J_n(s) = \frac{1}{2\pi i} \oint e^{\left(\frac{s}{2}\right)(t-\frac{1}{t})} t^{-n-1} dt, \ s \in \mathbb{C} \]

The aim is to approximate \( \frac{1}{J_0(s)} \) over \( \Omega = [0, 10] \times [-1, 1] \).

Irrational Approximation
Choose interpolation points in two different ways.

1. Structured grid with 2121 conjugate points.
Choose interpolation points in two different ways.

1. Structured grid with 2121 conjugate points.

2. Random uniformly distributed points as 2000 conjugate points.
Choose interpolation points in two different ways.

1. Structured grid with 2121 conjugate points.

2. Random uniformly distributed points as 2000 conjugate points.

- **Remark:** In both cases, conjugates pairs are under consideration in order to built a real model approximant.
Method 1: The Loewner Framework

- **Singular values and superimposed graphs** - $H(s)$, $H_r(s)$.

- **Error** $O(10^{-11})$ and the $2 \times 11$ compressed points from the initial 2121.

---

Dimitris, karachalios@mpi-magdeburg.mpg.de
Method 1: The Loewner Framework

Poles and Zeros diagram

\[
Poles = \begin{pmatrix}
-8.32213293322054 - 1.4252i \\
-8.32213289862456 + 1.4252i \\
-5.51461491999547 \\
-2.40481847965605 \\
2.40482555769577 \\
5.52007811028631 \\
8.65372791291101 \\
11.7915356008908 \\
14.9135964357538 \\
17.6548692348549 - 1.561i \\
17.654869354827 + 1.561i
\end{pmatrix}
\]

Bessel original roots = \[
\begin{pmatrix}
2.40482555769577 \\
5.52007811028631 \\
8.65372791291101 \\
11.7915344390142 \\
14.9309177084877 \\
18.0710639679109
\end{pmatrix}
\]
Structured grid 2121 points compressed to 22.
Structured grid 2121 points compressed to 22.

Random Uniformly grid 2000 points compressed to 24.

Remark: If we directly use those compressed points together with the corresponding values as the interpolation data set (points/values), the interpolant constructed this way will coincide with the interpolant computed from the initial 2000 points. Hence, those are optimal points.

Dimitris, karachalios@mpi-magdeburg.mpg.de
Left & Right Projected Points

- Structured grid **2121** points compressed to **22**.

- Random Uniformly grid **2000** points compressed to **24**.

**Remark:** If we directly use those compressed points together with the corresponding values as the interpolation data set (points/values), the interpolant constructed this way will coincide with the interpolant computed from the initial \( \sim 2000 \) points. Hence, those are optimal points.
Method 2: The AAA algorithm

- The AAA approximant $H_r(s)$ and the original function.

- Absolute error over the $\Omega$ domain: $O(10^{-11})$ + support points.
Method 3: The VF method

- The VF approximant \( H_r(s) \) and the original function.

- Absolute error over the \( \Omega \) domain: \( O(10^{-6}) \).

- After the pole/zero cancellation, obtain an order \( r=11 \) approximant.

- The largest error appears in the vicinity of the 3rd pole.
### Methods comparison

<table>
<thead>
<tr>
<th>Case/Method</th>
<th>Loewner</th>
<th>AAA</th>
<th>VF</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st: 2121 structured points</td>
<td>$O(10^{-11})$</td>
<td>$O(10^{-11})$</td>
<td>$O(10^{-6})$</td>
</tr>
<tr>
<td>2nd: 2000 uniformly points</td>
<td>$O(10^{-11})$</td>
<td>$O(10^{-13})$</td>
<td>$O(10^{-6})$</td>
</tr>
</tbody>
</table>

**Table:** Error comparison
Bessel Approximation

Methods comparison

<table>
<thead>
<tr>
<th>Case/Method</th>
<th>Loewner</th>
<th>AAA</th>
<th>VF</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st: 2121 structured points</td>
<td>$O(10^{-11})$</td>
<td>$O(10^{-11})$</td>
<td>$O(10^{-6})$</td>
</tr>
<tr>
<td>2nd: 2000 uniformly points</td>
<td>$O(10^{-11})$</td>
<td>$O(10^{-13})$</td>
<td>$O(10^{-6})$</td>
</tr>
</tbody>
</table>

Table: Error comparison

1. The Loewner Framework:
   - builds: $(r - 1, r)$ real approximant.
   - is a direct method.
   - main complexity is due to SVD.

Dimitris, karachalios@mpi-magdeburg.mpg.de
Methods comparison

<table>
<thead>
<tr>
<th>Case/Method</th>
<th>Loewner</th>
<th>AAA</th>
<th>VF</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st: 2121 structured points</td>
<td>$O(10^{-11})$</td>
<td>$O(10^{-11})$</td>
<td>$O(10^{-6})$</td>
</tr>
<tr>
<td>2nd: 2000 uniformly points</td>
<td>$O(10^{-11})$</td>
<td>$O(10^{-13})$</td>
<td>$O(10^{-6})$</td>
</tr>
</tbody>
</table>

Table: Error comparison

1. The Loewner Framework:
   - builds: $(r - 1, r)$ real approximant.
   - is a direct method.
   - main complexity is due to SVD.

2. The AAA algorithm:
   - builds: $(r, r)$ complex approximant.
   - is an iterative method.
   - main complexity is due to SVDs of an incremental dimension.
### Methods comparison

<table>
<thead>
<tr>
<th>Case/Method</th>
<th>Loewner</th>
<th>AAA</th>
<th>VF</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st: 2121 structured points</td>
<td>$O(10^{-11})$</td>
<td>$O(10^{-11})$</td>
<td>$O(10^{-6})$</td>
</tr>
<tr>
<td>2nd: 2000 uniformly points</td>
<td>$O(10^{-11})$</td>
<td>$O(10^{-13})$</td>
<td>$O(10^{-6})$</td>
</tr>
</tbody>
</table>

**Table:** Error comparison

1. **The Loewner Framework:**
   - builds: $(r - 1, r)$ real approximant.
   - is a direct method.
   - main complexity is due to SVD.

2. **The AAA algorithm:**
   - builds: $(r, r)$ complex approximant.
   - is an iterative method.
   - main complexity is due to SVDs of an incremental dimension.

3. **The VF method:**
   - builds: $(r + 1, r)$ real approximant.
   - is an iterative method.
   - main complexity is due to: solving 2 least squares.
An Euler - Bernoulli Beam

Further examples treated with the Loewner Framework - [R. Curtain/K. Morris '09]

PDE

\[
\frac{\partial^2 w(x,t)}{\partial t^2} + \frac{\partial^2}{\partial x^2} \left[ EI \frac{\partial^2 w(x,t)}{\partial x^2} + cdI \frac{\partial^3 w(x,t)}{\partial x^2 \partial t} \right] = 0
\]

Boundary Conditions and Input - Output

- \( w(0, t) = 0, \frac{\partial w}{\partial x}(0, t) = 0, EI \frac{\partial^2 w(L,t)}{\partial x^2} + cdI \frac{\partial^3 w(L,t)}{\partial x^2 \partial t} = 0 \)
- \(-EI \frac{\partial^3 w(L,t)}{\partial x^3} - cdI \frac{\partial^4 w(L,t)}{\partial x^3 \partial t} = u(t), \quad y(t) = \frac{\partial w(L,t)}{\partial t}\)

\[
H(s) = \frac{sN(s)}{(EI + sCdI)m^3(s)D(s)}
\]

- \( m(s) = \left[ -\frac{s^2}{EI + cdIs} \right]^{\frac{1}{4}}, \quad N(s) = \cosh(Lm(s))\sin(Lm(s)) - \sinh(Lm(s))\cos(Lm(s)), \)
- \( D(s) = 1 + \cosh(Lm(s))\cos(Lm(s)) \)

Dimitris, karachalios@mpi-magdeburg.mpg.de
Further examples treated with the Loewner Framework

- **Approximant:** sample domain $\rightarrow [10, 10^4] \text{Hz}$ and Error curve.

- **Pole/Zero diagram.**

---

**Parameter values:**
- $E = 69, \ GPa = 6.9 \cdot 10^{10} \text{ N/m}^2$ - Young’s modulus elasticity constant,
- $I = (1/12) \cdot 7 \cdot 8.5^3 \cdot 10^{-11} \text{ m}^4$ - moment of inertia,
- $c_d = 5 \cdot 10^{-4}$ - damping constant,
- $L = 0.7m, \ b = 7cm, \ h = 8.5\text{ mm}$ - length, base, height of the rectangular cross section.
Example from [Filip/Nakatsukasa/Trefethen/Beckermann '17]

\[ H(x) = \frac{100\pi(x^2 - 0.36)}{\sinh(100\pi(x^2 - 0.36))} , \quad x \in [-1, 1] \]

Pole/Zero diagram ("Far" - "Zoom")
Transfer function from 1D Heat equation

Example from [Beattie/Gugergin 12’]

- \( H(s) = e^{-\sqrt{s}} \) with \( s \in I = \{ j\omega : \omega \in \mathbb{R}_+ \} \).

Poles/Zeros diagram and the impulse response of the system.
We investigated the practical applicability of three rational approximation methods for fitting irrational transfer functions.
We investigated the practical applicability of three rational approximation methods for fitting irrational transfer functions.

The Loewner framework computes a full SVD. We are able to optimize the computational cost:

- By investigating the theoretical upper bounds of the singular values of $L$, we can use a theoretical bound as a “seed” for a shorter version of SVD (or rSVD where r stands for randomized SVD). This could be accessible with the Zolotarev bounds. [Beckermann & Townsend'16]
- Another approach is to substitute SVD with “pseudoskeleton” approximation - CUR decomposition:
  1. Max volume/Cross approximation [B.Kramer & A. Gorodetsky '16]
  2. DEIM - CUR [D.C. Sorensen & M. Embree '16]

Left and Right compressed projected points are special points!

1. Aim is to analyse the compressed information.
2. Connection with potential theory and Zolotarev numbers.
We investigated the practical applicability of three rational approximation methods for fitting irrational transfer functions.

The Loewner framework computes a full SVD. We are able to optimize the computational cost:

- By investigating the theoretical upper bounds of the singular values of $\mathbb{L}$, we can use a theoretical bound as a "seed" for a shorter version of SVD (or rSVD where $r$ stands for randomized SVD). This could be accessible with the Zolotarev bounds. [Beckermann & Townsend'16]

Another approach is to substitute SVD with "pseudoskeleton" approximation - CUR decomposition:

1. Max volume/Cross approximation [B.Kramer & A. Gorodetsky '16]
2. DEIM - CUR [D.C. Sorensen & M. Embree '16]

Left and Right compressed projected points are special points!

1. Aim is to analyse the compressed information.
2. Connection with potential theory and Zolotarev numbers.
We investigated the practical applicability of three rational approximation methods for fitting irrational transfer functions.

The Loewner framework computes a full SVD. We are able to optimize the computational cost:

- By investigating the theoretical upper bounds of the singular values of $\mathbb{L}$, we can use a theoretical bound as a "seed" for a shorter version of SVD (or rSVD where $r$ stands for randomized SVD). This could be accessible with the Zolotarev bounds. [Beckermann & Townsend'16]
- Another approach is to substitute SVD with "pseudoskeleton" approximation - CUR decomposition:
We investigated the practical applicability of three rational approximation methods for fitting irrational transfer functions.

The Loewner framework computes a full SVD. We are able to optimize the computational cost:

- By investigating the theoretical upper bounds of the singular values of $\mathbb{L}$, we can use a theoretical bound as a "seed" for a shorter version of SVD (or rSVD where $r$ stands for randomized SVD). This could be accessible with the Zolotarev bounds. [Beckermann & Townsend’16]
- Another approach is to substitute SVD with "pseudoskeleton" approximation - CUR decomposition:
  1. Max volume/Cross approximation [B.Kramer & A. Gorodetsky ’16]
Conclusions and further developments

- We investigated the practical applicability of three rational approximation methods for fitting irrational transfer functions.

- The Loewner framework computes a full SVD. We are able to optimize the computational cost:
  - By investigating the theoretical upper bounds of the singular values of $L$, we can use a theoretical bound as a "seed" for a shorter version of SVD (or rSVD where r stands for randomized SVD). This could be accessible with the Zolotarev bounds. [Beckermann & Townsend’16]
  - Another approach is to substitute SVD with "pseudoskeleton" approximation - CUR decomposition:
    1. Max volume/Cross approximation [B.Kramer & A. Gorodetsky '16]
    2. DEIM - CUR [D.C. Sorensen & M. Embree '16]
We investigated the practical applicability of three rational approximation methods for fitting irrational transfer functions.

The Loewner framework computes a full SVD. We are able to optimize the computational cost:

- By investigating the theoretical upper bounds of the singular values of $\mathbb{L}$, we can use a theoretical bound as a "seed" for a shorter version of SVD (or rSVD where $r$ stands for randomized SVD). This could be accessible with the Zolotarev bounds. [Beckermann & Townsend'16]
- Another approach is to substitute SVD with "pseudoskeleton" approximation - CUR decomposition:
  1. Max volume/Cross approximation [B.Kramer & A. Gorodetsky '16]
  2. DEIM - CUR [D.C. Sorensen & M. Embree '16]
- Left and Right compressed projected points are special points!
Conclusions and further developments

- We investigated the practical applicability of three rational approximation methods for fitting irrational transfer functions.

- The Loewner framework computes a full SVD. We are able to optimize the computational cost:
  - By investigating the theoretical upper bounds of the singular values of $L$, we can use a theoretical bound as a "seed" for a shorter version of SVD (or rSVD where r stands for randomized SVD). This could be accessible with the Zolotarev bounds. [Beckermann & Townsend'16]
  - Another approach is to substitute SVD with "pseudoskeleton" approximation - CUR decomposition:
    1. Max volume/Cross approximation [B.Kramer & A. Gorodetsky '16]
    2. DEIM - CUR [D.C. Sorensen & M. Embree '16]

- Left and Right compressed projected points are special points!
  1. Aim is to analyse the compressed information.

Dimitris, karachalios@mpi-magdeburg.mpg.de

Irrational Approximation
Conclusions and further developments

- We investigated the practical applicability of three rational approximation methods for fitting irrational transfer functions.
- The Loewner framework computes a full SVD. We are able to optimize the computational cost:
  - By investigating the theoretical upper bounds of the singular values of \( \mathbb{L} \), we can use a theoretical bound as a "seed" for a shorter version of SVD (or rSVD where \( r \) stands for randomized SVD). This could be accessible with the Zolotarev bounds. [Beckermann & Townsend’16]
  - Another approach is to substitute SVD with "pseudoskeleton" approximation - CUR decomposition:
    1. Max volume/Cross approximation [B.Kramer & A. Gorodetsky ’16]
    2. DEIM - CUR [D.C. Sorensen & M. Embree ’16]
- Left and Right compressed projected points are special points!
  1. Aim is to analyse the compressed information.
  2. Connection with potential theory and Zolotarev numbers.
A.C. Antoulas, *Approximation of Large-Scale Dynamical Systems, Advances in Design and Control*, https://doi.org/10.1137/1.9780898718713, SIAM, Philadelphia, 2005,


THANK YOU VERY MUCH FOR YOUR ATTENTION!
ANY QUESTIONS...?