Sparse Interpolation

1. Prony's Method
   - interpolation for coefficients and exponents
2. Singularities of Stewart-Gough platforms
   - computing the determinant of the Jacobian matrix
3. Reformulating Prony's Method
   - deriving a generalized eigenvalue problem
   - the QZ method to solve $A\mathbf{x} = \lambda B\mathbf{x}$
4. The qd Algorithm
   - determining the size of the support

MCS 563 Lecture 29
Analytic Symbolic Computation
Jan Verschelde, 21 March 2014

Sparse Interpolation

1. Prony's Method
   - interpolation for coefficients and exponents
2. Singularities of Stewart-Gough platforms
   - computing the determinant of the Jacobian matrix
3. Reformulating Prony's Method
   - deriving a generalized eigenvalue problem
   - the QZ method to solve $A\mathbf{x} = \lambda B\mathbf{x}$
4. The qd Algorithm
   - determining the size of the support

interpolating exponentials

Consider the exponential function $F(x) = c_1 e^{\mu_1 x} + c_2 e^{\mu_2 x}$ for *unknown* coefficients c_1, c_2, and exponents μ_1, μ_2.

To determine 4 unknowns, evaluate at $x = 0, 1, 2, 3$:

$$\begin{cases} F(0) = c_1 + c_2 = c_1 + c_2 \\ F(1) = c_1 e^{\mu_1} + c_2 e^{\mu_2} = c_1 b_1 + c_2 b_2 \\ F(2) = c_1 e^{2\mu_1} + c_2 e^{2\mu_2} = c_1 b_1^2 + c_2 b_2^2 \\ F(3) = c_1 e^{3\mu_1} + c_2 e^{3\mu_2} = c_1 b_1^3 + c_2 b_2^3 \end{cases}$$

where $b_1 = e^{\mu_1}$ and $b_2 = e^{\mu_2}$.

The matrix of the linear system in c_1 and c_2 is a Vandermonde matrix.

an auxiliary polynomial

We introduce an auxiliary polynomial

$$p(z) = (z - b_1)(z - b_2) = z^2 + \lambda_1 z + \lambda_0.$$

If we know the coefficients λ_1 and λ_0 of p,
then b_1 and b_2 are found by computing the roots of p.

Once b_1 and b_2 are known,
we find c_1 and c_2 by solving a linear system.

Because $p(b_1) = 0$ and $p(b_2) = 0$, we have

$$\begin{cases} c_1 p(b_1) + c_2 p(b_2) = 0 \\ c_1 b_1 p(b_1) + c_2 b_2 p(b_2) = 0. \end{cases}$$

a Hankel system

Recall $p(z) = z^2 + \lambda_1 z + \lambda_0$.

$$\begin{cases} c_1(b_1^2 + \lambda_1 b_1 + \lambda_0) + c_2(b_2^2 + \lambda_1 b_2 + \lambda_0) = 0 \\ c_1(b_1^3 + \lambda_1 b_1^2 + \lambda_0 b_1) + c_2(b_2^3 + \lambda_1 b_2^2 + \lambda_0 b_2) = 0. \end{cases}$$

Collecting terms in λ_0 and λ_1 gives

$$\begin{cases} \lambda_0(c_1 + c_2) + \lambda_1(c_1 b_1 + c_2 b_2) + c_1 b_1^2 + c_2 b_2^2 = 0 \\ \lambda_0(c_1 b_1 + c_2 b_2) + \lambda_1(c_1 b_1^2 + c_2 b_2^2) + c_1 b_1^3 + c_2 b_2^3 = 0. \end{cases}$$

As we recognize $F(0)$, $F(1)$, $F(2)$, and $F(3)$,
we write the system in shortened form as

$$\begin{bmatrix} F(0) & F(1) \\ F(1) & F(2) \end{bmatrix} \begin{bmatrix} \lambda_0 \\ \lambda_1 \end{bmatrix} = - \begin{bmatrix} F(2) \\ F(3) \end{bmatrix}.$$

The matrix is a Hankel matrix.

the method of Prony

Prony's method performs three steps:

1. Solve the Hankel system to find the coefficients λ_1 and λ_0 of the auxiliary polynomial $p(z)$.
2. Solve $p(z) = 0$. The roots of p are b_1 and b_2.
   So the exponents are $\mu_1 = \ln(b_1)$ and $\mu_2 = \ln(b_2)$.
3. Solve the first two interpolating condition for c_1 and c_2.
   For known b_1 and b_2, this system is linear.

This works of course for any number of exponentials.

Sparse Interpolation

1. Prony's Method
   - interpolation for coefficients and exponents
2. Singularities of Stewart-Gough platforms
   - computing the determinant of the Jacobian matrix
3. Reformulating Prony's Method
   - deriving a generalized eigenvalue problem
   - the QZ method to solve $A\mathbf{x} = \lambda B\mathbf{x}$
4. The qd Algorithm
   - determining the size of the support

Stewart-Gough Platforms

Think of the design of a flight simulator:

- A moving platform is supported above a stationary base by six legs.
- The ith leg connects to the base at the joint $\mathbf{b}_i$ and to the platform at the joint at $\mathbf{a}_i$, $i = 1, 2, \ldots, 6$.
- The leg length L_i is controlled by a linear actuator.

Understanding the singularities is important for designers.

The condition for singularity can be derived as follows:

- With quaternions we formulate the platform system.
- The determinant of the Jacobian matrix is a polynomial in several variables, which we compute explicitly.

deriving equations

The platform is represented by

- its position $\mathbf{p} \in \mathbb{C}^3$; and
- the orientation by a quaternion $\mathbf{q} \in \mathbb{P}^3$.

In base coordinates, the vector $\mathbf{v}_i$ connects base point $\mathbf{b}_i$ to the corresponding platform point $\mathbf{a}_i$:

$$\mathbf{v}_i := -\mathbf{b}_i + \mathbf{p} + R(\mathbf{q})\mathbf{a}_i = -\mathbf{b}_i + \mathbf{p} + \mathbf{q}\mathbf{a}_i\mathbf{q}'/\mathbf{q}\mathbf{q}',$$

where $R(\mathbf{q})$ is the 3×3 rotation matrix giving the same rotation to a vector $\mathbf{w}$ as the quaternion operation $\mathbf{q}\mathbf{w}\mathbf{q}'/\mathbf{q}\mathbf{q}'$.

relations on leg lengths

The squared length of leg i is $L_i^2 = \mathbf{v}_i \cdot \mathbf{v}_i$, so

$$L_i \dot{L}_i = \mathbf{v}_i \cdot \dot{\mathbf{v}}_i = \mathbf{v}_i \cdot (\dot{\mathbf{p}} + \boldsymbol{\omega} \times (R\mathbf{a}_i)),$$

where "$\cdot$" is the vector inner product, $\times$ is the vector cross product, and $\boldsymbol{\omega}$ is the angular velocity vector.

Rewriting $R = \widehat{R}/Q$ with

$$Q := \mathbf{q}\mathbf{q}' = q_0^2 + q_1^2 + q_2^2 + q_3^2,$$

and substituting from the equation on $\mathbf{v}_i$
we transform the relation on the leg lengths into

$$QL_i \dot{L}_i = (Q(\mathbf{p} - \mathbf{b}_i) + \widehat{R}\mathbf{a}_i) \cdot \dot{\mathbf{p}} + ((\widehat{R}\mathbf{a}_i) \times (\mathbf{p} - \mathbf{b}_i)) \cdot \boldsymbol{\omega}.$$

the Jacobian matrix

Letting $\mathbf{J}$ be the matrix whose i-th column is

$$\mathbf{J}_i = \begin{bmatrix} Q(\mathbf{p} - \mathbf{b}_i) + \widehat{R}\mathbf{a}_i \\ (\widehat{R}\mathbf{a}_i) \times (\mathbf{p} - \mathbf{b}_i) \end{bmatrix}, \qquad i = 1, 2, \ldots, 6,$$

the singularity condition is $0 = \mathbf{Jw}$ for some $\mathbf{w} \neq 0$,
or equivalently, $\det(\mathbf{J}) = 0$.

The expanded $\det(\mathbf{J})$ defines the polynomial we will factor.

Although we start out with quadratic polynomials, $\det(\mathbf{J})$ is a homogeneous polynomial of degree 1728 in 42 variables.

cases studied

Some case studies:

1. General platform, fixed position:
   In this case, det($\mathbf{J}$) is a homogeneous polynomial of degree 12
   with 910 terms in $\mathbf{q} = \{q_0, q_1, q_2, q_3\}$.

   The factorization is $\det(\mathbf{J}) = F_1(\mathbf{q})Q^3$,
   where $F_1(\mathbf{q})$ is a sextic.
   The factor of $Q^3 = 0$ is not of physical significance.
2. Planar base and platform, fixed position:
   det(J) is still homogeneous of degree 12,
   and still factors in two:

   one irreducible single factor of degree six,
   and the quadratic factor having multiplicity three.

third case study

3. Planar base and platform, parallel planes:
   In this case, det($\mathbf{J}$) becomes cubic in $\mathbf{p}$,
   homogeneous of degree 12 in (q_0, q_3),
   and degree 15 in $(\mathbf{p}, \mathbf{q})$ together.

   This polynomial is much sparser: only 24 terms.

   The computed factorization is

$$\det(\mathbf{J}) = a p_3^3 (q_0 + b q_3)(q_0 + c q_3)(q_0 + i q_3)^5 (q_0 - i q_3)^5,$$

   where a, b, c are constants that depend on the choice of $\mathbf{a}_i, \mathbf{b}_i$.

Sparse Interpolation

1. Prony's Method
   - interpolation for coefficients and exponents
2. Singularities of Stewart-Gough platforms
   - computing the determinant of the Jacobian matrix
3. Reformulating Prony's Method
   - deriving a generalized eigenvalue problem
   - the QZ method to solve $A\mathbf{x} = \lambda B\mathbf{x}$
4. The qd Algorithm
   - determining the size of the support

numerical problems

The three steps in Prony's algorithm
all suffer numerical problems:

1. a Hankel matrix might be very ill-conditioned;
2. the roots are often very sensitive to small changes
   in the coefficients;
3. Vandermonde matrices are also typically ill-conditioned.

Therefore, we reformulate the problem.

matrix decomposition

As in our two exponentials as before:

$$\begin{bmatrix} 1 & 1 \\ b_1 & b_2 \end{bmatrix} \underbrace{\begin{bmatrix} c_1 & 0 \\ 0 & c_2 \end{bmatrix} \begin{bmatrix} 1 & b_1 \\ 1 & b_2 \end{bmatrix}} = \underbrace{\begin{bmatrix} F(0) & F(1) \\ F(1) & F(2) \end{bmatrix}}$$

$$\begin{bmatrix} 1 & 1 \\ b_1 & b_2 \end{bmatrix} \overbrace{\begin{bmatrix} c_1 & c_1 b_1 \\ c_2 & c_2 b_2 \end{bmatrix}} = \overbrace{\begin{bmatrix} c_1 + c_2 & c_1 b_1 + c_2 b_2 \\ c_1 b_1 + c_2 b_2 & c_1 b_1^2 + c_2 b_2^2 \end{bmatrix}},$$

abbreviated as $VDV^T = H_0$.

matrix multiplications

Multiply the diagonal matrix D with another diagonal matrix that has b_1 and b_2 on its diagonal:

$$\begin{bmatrix} 1 & 1 \\ b_1 & b_2 \end{bmatrix} \underbrace{\begin{bmatrix} c_1 b_1 & 0 \\ 0 & c_2 b_2 \end{bmatrix} \begin{bmatrix} 1 & b_1 \\ 1 & b_2 \end{bmatrix}} = \underbrace{\begin{bmatrix} F(1) & F(2) \\ F(2) & F(3) \end{bmatrix}}$$

$$\begin{bmatrix} 1 & 1 \\ b_1 & b_2 \end{bmatrix} \overbrace{\begin{bmatrix} c_1 b_1 & c_1 b_1^2 \\ c_2 b_2 & c_2 b_2^2 \end{bmatrix}} = \overbrace{\begin{bmatrix} c_1 b_1 + c_2 b_2 & c_1 b_1^2 + c_2 b_2^2 \\ c_1 b_1^2 + c_2 b_2^2 & c_1 b_1^3 + c_2 b_2^3 \end{bmatrix}},$$

abbreviated as $VDBV^T = H_1$.

generalized eigenvalue problem

The two Hankel matrices H_0 and H_1 are diagonalized simultaneously:

$$\begin{aligned} V^{-1} H_0 V^{-T} &= D \\ V^{-1} H_1 V^{-T} &= DB. \end{aligned}$$

Therefore, we obtained the generalized eigenvalue problem

$$(H_1 - \lambda H_0)\mathbf{v} = \mathbf{0}.$$

This reformulation of Prony's method as a generalized eigenvalue problem avoids the computation of the coefficients of the auxiliary polynomial and the corresponding root finding problem.

Sparse Interpolation

1. Prony's Method
   - interpolation for coefficients and exponents
2. Singularities of Stewart-Gough platforms
   - computing the determinant of the Jacobian matrix
3. Reformulating Prony's Method
   - deriving a generalized eigenvalue problem
   - the QZ method to solve $A\mathbf{x} = \lambda B\mathbf{x}$
4. The qd Algorithm
   - determining the size of the support

generalized eigenvalue problem

A generalizated eigenvalue problem is

$$A\mathbf{x} = \lambda B\mathbf{x}, \quad A, B \in \mathbb{C}^{n \times n},$$

where λ is a scalar and $\mathbf{x}$ an n-vector.

We look for eigenvalues $\lambda \in \mathbb{C}$ so that $(A - \lambda B)\mathbf{x} = \mathbf{0}$ admits an eigenvector $\mathbf{x} \in \mathbb{C}^n$, $\mathbf{x} \neq \mathbf{0}$.

If B has an inverse B^{-1}, then we could solve $B^{-1}A\mathbf{x} = \lambda\mathbf{x}$.

An ill conditioned matrix B prevents accurate computation of well conditioned eigenvalue-eigenvector pairs.

the Schur decomposition

We first compute the Schur decomposition of A and B:

$$\begin{cases} Q^H A Z = T, & Q^H Q = I,\ Z^H Z = I, \\ Q^H B Z = S, & T \text{ and } S \text{ are upper triangular.} \end{cases}$$

I represents the identity matrix.
By $\cdot^H$ we denote the conjugate transposition: $(a_{ij})^H = \bar{a}_{ji}$.

Applying the transformations to $A\mathbf{x} = \lambda B\mathbf{x}$ gives
$Q^H(A - \lambda B)Z = Q^H A Z - \lambda Q^H B Z = T - \lambda S$,
reducing $A\mathbf{x} = \lambda B\mathbf{x}$ to $T\mathbf{x} = \lambda S\mathbf{x}$.

Givens rotations introduce zeroes:

$$\begin{bmatrix} c & -s \\ s & c \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} z \\ 0 \end{bmatrix}, \quad z = \sqrt{x^2 + y^2},\ c = \frac{x}{z},\ s = -\frac{y}{z},$$

where $c = \cos(\theta)$ and $s = \sin(\theta)$ for the rotation angle θ.

Hessenberg matrices

The first step to obtain a Schur decomposition is
to compute a Hessenberg-triangular reduction.

A matrix $A = (a_{ij})$ is upper Hessenberg
if $a_{ij} = 0$ for all $i > j + 1$.

An upper Hessenberg matrix is almost upper triangular,
except of the elements just below the diagonal.

Hessenberg-triangular reduction of (A, B)

Algorithm Hessenberg-Triangular Reduction of (A, B)

Input: $A, B \in \mathbb{C}^{n \times n}$.

Output: $Q, Z, T, S \in \mathbb{C}^{n \times n}$: $Q^H Q = I$, $Z^H Z = I$,
$Q^H A Z = T$, T is upper Hessenberg,
$Q^H B Z = S$, S is upper triangular.

$[Q^H, B] := QR(B)$;
$A := Q^H A$; $Z := I$;
for j from 1 to $n - 1$ do
  for i from $n + 1 - j$ to $j + 2$ do
    $A := Q_{ij}^H A$; $B := Q_{ij}^H B$; $Q := Q_{ij}^H Q$;
    $A := A Z_{ij}$; $B := B Z_{ij}$; $Z := Z Z_{ij}$;
return (Q, Z, A, B).

Sparse Interpolation

1. Prony's Method
   - interpolation for coefficients and exponents
2. Singularities of Stewart-Gough platforms
   - computing the determinant of the Jacobian matrix
3. Reformulating Prony's Method
   - deriving a generalized eigenvalue problem
   - the QZ method to solve $A\mathbf{x} = \lambda B\mathbf{x}$
4. The qd Algorithm
   - determining the size of the support

the size of the support

The problem of sparse interpolation is to determine
the number of monomials in the polynomial

$$f(\mathbf{x}) = \sum_{\mathbf{a} \in A} c_{\mathbf{a}} \mathbf{x}^{\mathbf{a}}, \quad m = \#A.$$

Given is a blackbox function to evaluate f at roots of unity ω_k yielding values $v_k = f(\omega_k)$, $k = 0, 1, \ldots, 2m-1$.

The problem is to determine m.

Hadamard polynomials

For $m = 3$, $p(x) = x^3 + \lambda_2 x^2 + \lambda_1 x + \lambda_0$ where

$$\begin{bmatrix} v_0 & v_1 & v_2 \\ v_1 & v_2 & v_3 \\ v_2 & v_3 & v_4 \end{bmatrix} \begin{bmatrix} \lambda_0 \\ \lambda_1 \\ \lambda_2 \end{bmatrix} = - \begin{bmatrix} v_3 \\ v_4 \\ v_5 \end{bmatrix}.$$

Moving the righthandside of the above equation to the left and adding $\lambda_0 + \lambda_1 x + \lambda_2 x^2 + x^3 = 0$ gives

$$\begin{bmatrix} v_0 & v_1 & v_2 & v_3 \\ v_1 & v_2 & v_3 & v_4 \\ v_2 & v_3 & v_4 & v_5 \\ 1 & x & x^2 & x^3 \end{bmatrix} \begin{bmatrix} \lambda_0 \\ \lambda_1 \\ \lambda_2 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

The determinant of this matrix equation is the polynomial denoted by $H_3^{(0)}(x)$, where its leading coefficient is $\det H_3^{(0)}$. Observe: if $m = 2$, then $\det H_3^{(0)} = 0$.

Hankel matrices

A k-by-k Hankel matrix $H_k^{(j)}$ is defined by

$$H_k^{(j)} = \begin{bmatrix} v_j & v_{j+1} & \cdots & v_{j+k-1} \\ v_{j+1} & v_{j+2} & \cdots & v_{j+k} \\ \vdots & \vdots & \ddots & \vdots \\ v_{j+k-1} & v_{j+k} & \cdots & v_{j+2k-2} \end{bmatrix}.$$

Neighboring Hankel determinants satisfy

$$\left(\det H_k^{(j)}\right)^2 = \det H_k^{(j-1)} \det H_k^{(j+1)} - \det H_{k+1}^{(j-1)} \det H_{k-1}^{(j+1)}.$$

The qd algorithm bypasses the computation of Hankel determinants.

the qd algorithm

The initialization of the qd algorithm goes as

$$e_0^{(j)} = 0,\ j = 0, 1, \ldots, \quad q_1^{(j)} = \frac{v_{j+1}}{v_j},\ j = 1, 2, \ldots$$

and for $k = 1, 2, \ldots$:

$$q_k^{(0)} = \frac{\det H_{k-1}^{(0)} \det H_k^{(1)}}{\det H_k^{(0)} \det H_{k-1}^{(1)}}, \quad e_k^{(0)} = \frac{\det H_{k+1}^{(0)} \det H_{k-1}^{(1)}}{\det H_k^{(0)} \det H_k^{(1)}}.$$

Then the progressive qd algorithm uses the formulas

$$q_k^{(j+1)} = e_{k-1}^{(j+1)} - e_k^{(j)} + q_k^{(j)}, \quad e_k^{(j+1)} = \left(\frac{q_{k+1}^{(j)}}{q_k^{(j+1)}} \right) e_k^{(j)},$$

for $k = 1, 2, \ldots$ and $j = 0, 1, \ldots$.

Properties

The q and the d of the qd algorithm
stand for quotient and difference.

The qd algorithm computes both the roots z_k of the polynomial p and its degree m.

The key properties are:

1. $\lim_{j \to \infty} q_k^{(j)} = z_k$; and
2. $e_m^{(j)} = 0$ for all $j = 1, 2, \ldots$.

Summary + Exercises

A numerically stable version of Prony's method uses the qd algorithm and the QZ method for sparse interpolation.

Exercises:

1. Suppose a polynomial in one variable is given as a blackbox function: evaluation is cheap, but we do not know its algebraic representation. Explain by example(s) how divided differences and Newton interpolation lead to the degree of the polynomial.

more exercises

3. Generate Vandermonde matrices of increasing dimension (using MATLAB or Octave) and compute their condition number.
   Take interpolation points in $[-1, +1]$ and compare the condition numbers for (1) equidistant interpolation points; (2) randomly chosen points, from a uniform distribution; and (3) for interpolation points as the roots of the Chebyshev polynomials.
4. Verify $H\mathbf{c} = \pm||\mathbf{c}||_2\mathbf{e}_1$, for $\mathbf{e}_1 = [1\ 0\ 0\ \cdots\ 0]^T$, $\mathbf{u} = \mathbf{c} \pm ||\mathbf{c}||_2\mathbf{e}_1$, and $H = \left(I - 2\frac{\mathbf{u}\mathbf{u}^T}{\mathbf{u}^T\mathbf{u}}\right)$.

and more exercises

5. Use the QZ algorithm as implemented in MATLAB or Octave to verify an instance of $VDV^T = H_0$.
6. Consider $f(x,y) = x^5y + 0.1xy^{13} - 0.5xy + 2.2x^4y^4$. Describe how to recover the exponents and coefficients, knowing that f has four terms.