ZJU-ZCCC

Solve an Eular beam with InverseDQM

Andy Jado

December 25, 2021

Abstract

A method to calculate the lower order of a differential equation numerically with a series of discreted functional values from higher order, derived directly from differential quadrature method(DQM), provides a linearly independent way to impose boundary conditions (B.C) on the govern equation from any order. Example is given by solving an Eular beam with different B.C.s

1 Introduction

DQM(Differential Quadrature Method) approximate the derivative value $\vec{f^n}(x)$ of a function at the gird points with $\vec{f^n}(x) \cong A^n\vec{f}(x)$ where A^n is a matrix filled with weighting coefficients that only concern the location of the grid points and $\vec{f}(x)$ be the vector of the functional value at the gird points. InverseDQM I call it becasue here we reconstruct the above equation into something like $\vec{f}(x) = (A^n)^{-1}\vec{f^n}(x) + \vec{e}$. An elgent intepretation for the error term $\vec{e}$ is realized in engineering sense.
There is a two page proof you can find in the attachment showing how I was driven into this resolution, along with the source code that covers everything in this report written in Julia.

2 Illustration

The idea is quite simple.
If you impose A^1 on a discreted function value vector from $f(x) = 0x + c_0$, you get a zero vector. If you impose A^n on a discreted function value vector from $f(x) = c_{n-1}x^{n-1} + ... + c_0$, you get a zero vector (a numerically tolerable zero). namely, the DQ matrix A doesn't recogonize that specific function value vector that you have from many others, Therefore, when you impose $(A^n)^{-1}$, the information your result vector missed comes from $c_{n-1}x^{n-1} + ... + c_0$. you can also impose $(A^1)^{-1}$ for n times and it converges with same amount of unknown coefficients.
Imagine climbing a ladder, combine DQM and inverseDQM, it allows us to set foot on any order of the ladder. Climb up, climb down,take a big step and skip some of the order, but we always set foot on the order where lies the boundary conditions, so we can impose the boundary conditions in a simple and direct way.
I'll show with examples.

3 Examples

The DE of an Eular beam is described as follow, where EI for material properties, m for distributed moments, p for distributed lateral force, and w denotes the deflection of the beam.

$$EI\frac{d^4w}{dx^4} = p - \frac{dm}{dx} \tag{1}$$

M and V for moment and shear force on the cross section.

$$M = -EI\frac{d^2w}{dx^2}$$

$$V = \frac{d^M}{dx} - m \tag{2}$$

$$\frac{dV}{dx} = -p$$

3 type of beams are investigated, and the boundary conditions are listed in the table.

Table 1: B.C.s for the beams

Type	x=0	x=L	force b.c.
Hinged-Hinged	$w = 0; w'' = 0$	$w = 0; w'' = 0$	concentrated unit force in middle span
Fixed-Free	$w = 0; w' = 0$	$w''' = 0; w'' = 0$	concentrated unit force at the end
Fixed-Hinged	$w = 0; w' = 0$	$w = 0; w'' = 0$	$p = 1N/m$

We assume we have sensors placed along the beam and the locations fullfill *Gauss-Lobatto-Chebyshev* points distribution, and the sensor tells us the moment at its location. namely, $[M(x_i)$ for $i = 1 : N]$.

Let me post the result first. It can be observed that the boundary conditions are well fit and the residual are 1% for the worst case.

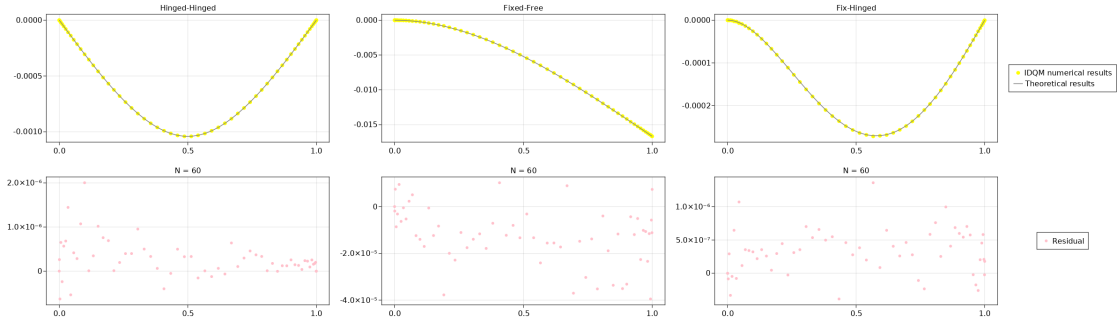

Figure 1: Deflection of an Eular beam solved by InverseDQM and the residual comparing to theoretical value (with 60 sampling points)

Now I show you the essential part of InverseDQM, I hope that'll explain why we always set foot on the order where lies the boundary conditions and the ability that DQM gives us to skip those orders with free ends.

4 Resolution

With equation (2) we write our $(EI)^{-1}\vec{M}$ into $\vec{w}''$ and we say $\Lambda^n = (A^n)^{-1}$.LU decomposition is applied when doing the inverse.

4.1 Hinged-Hinged

We have $w''(0) = 0; w''(L) = 0$ that is already revealed in the input, we know nothing about w' so we want to skip that order, so we apply Λ^2, thus we have $w_{notail} = \Lambda^2 w''$ and

$$w_{idqm} = w_{notail} + c_0\vec{1} + c_1\vec{x} \tag{3}$$

With

$$c_0 = w(0) - w_{notail}(0); c_1 = \{[(w(L) - w_{notail}(L)) - (w(0) - w_{notail}(0))]/L\} \tag{4}$$

Since we know every element in vector $\vec{w}_{notail}$ we can calculate $c_0; c_1$ with $w(0) = 0; w(L) = 0$ imposed on equation(4).

4.2 Fixed-Free

We have $w''(L) = 0$ that is already revealed in the input, however we have B.C. in an even higher order w''', therefore we have to apply A^1 to $\vec{w}''$ and impose $w'''(L) = 0$, and in each order of the ladder there is a B.C before we reach w, so we impose Λ^1 for 3 times.

$$\begin{aligned} \vec{w}'''_{dqm} &= A^1\vec{w}'' \\ \vec{w}'''_{dqm}(end) &= 0 \end{aligned} \tag{5}$$

It should be noticed that we modified vector $\vec{w}'''_{dqm}$ simply by trimming its last element into 0. It's a bold move. But let's just climb with this modified $\vec{w}'''_m$.

$$\begin{aligned} \vec{w}''_{notail} &= \Lambda^1\vec{w}'''_m \\ \vec{w}''_{idqm} &= \vec{w}''_{notail} + [\vec{w}''(0) - \vec{w}''_{notail}(0)] \end{aligned} \tag{6}$$

And now we climb the next order.

$$\begin{aligned} \vec{w}'_{notail} &= \Lambda^1\vec{w}''_{idqm} \\ \vec{w}'_{idqm} &= \vec{w}'_{notail} + [\vec{w}'(0) - \vec{w}'_{notail}(0)] \end{aligned} \tag{7}$$

And finally we impose all our boundary conditions.

$$\begin{aligned} \vec{w}_{notail} &= \Lambda^1\vec{w}'_{idqm} \\ \vec{w}_{idqm} &= \vec{w}_{notail} + [\vec{w(0)} - \vec{w}_{notail}(0)] \end{aligned} \tag{8}$$

Well you can check the result in figure 1.

4.3 Fixed-Hinged

Now we have an overstatic system, namely we have more boundary conditions than we need, it allows us to climb the ladder with both schemes we used in 4.1 and 4.2. So I just post the result with both schemes.

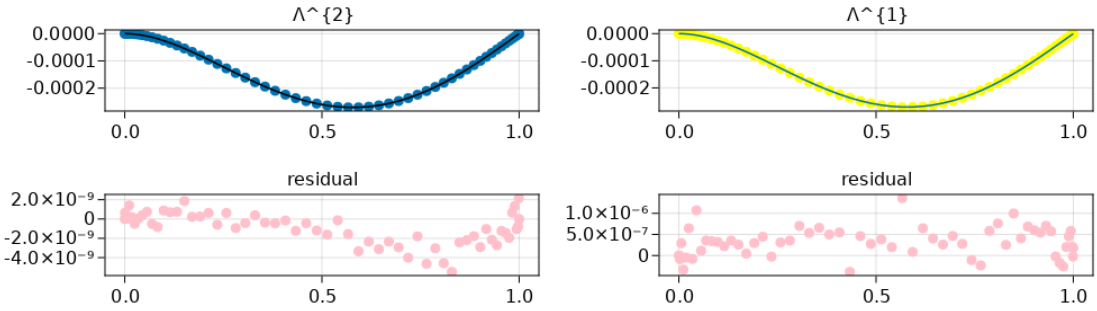

Figure 2: Solve a Fixed-Hinged beam with 2 schemes. (with 60 sampling points)

Yeah they both work and it appears we should use bigger steps when climbing ladders if B.C. allows.

5 Appendix

Let's see how mesh density affects the residual.

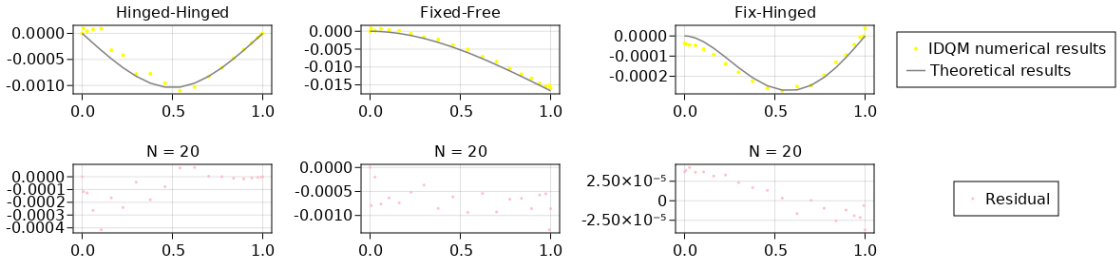

Figure 3: Solve a Fixed-Hinged beam with 2 schemes. (with 20 sampling points)

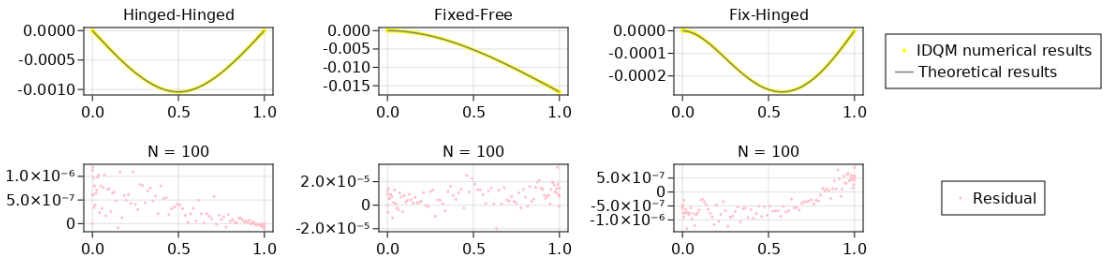

Figure 4: Solve a Fixed-Hinged beam with 2 schemes. (with 100 sampling points)

And IDQM doesn't care how messy your DE looks like, discreted function value vector and B.C. are all he need.

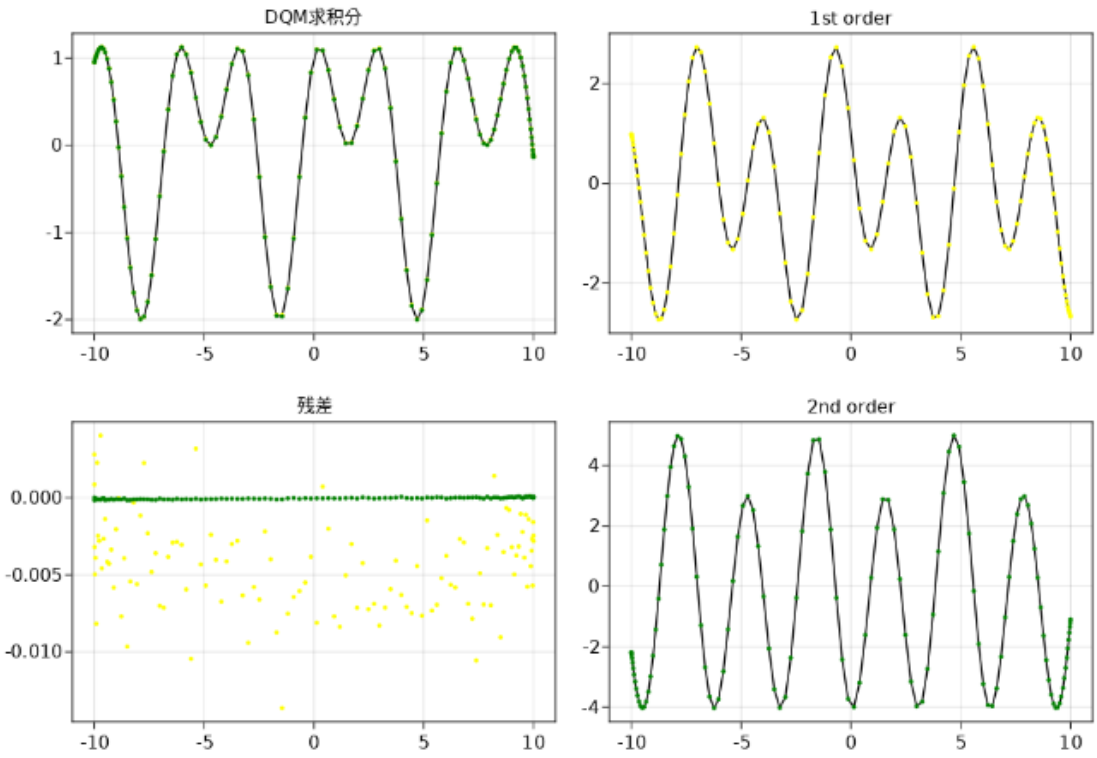

Figure 5: a messy function with 2 schemes(N=100)