Stable numerical technique to calculate the bending of flexures with extreme aspect ratios

Benjamin Schreyer[1], Lorenz Keck[2], Jon R. Pratt[1], Stephan Schlamminger[1]

[1]National Institute of Standards and Technology, 100 Bureau Drive, Gaithersburg, MD, USA
[2] SLAC National Accelerator Lab, 2575 Sand Hill Rd, Menlo Park, CA 94025, USA

E-mail: stephan.schlamminger@nist.gov

Abstract.

In designing single-sided clamped flexures as part of torsion balances or pendulums for scientific use, structures become thin and semi-analytic calculations of their bending become infeasible with standard double-precision variables. Semi-analytic calculations can be more efficient than finite element methods, allowing faster design optimization. We present straightforward analytical results demonstrating that the failure of the semi-analytic bending simulation, computed using double precision, arises from the exponential growth of the bending angle. To address this issue, we developed a Runge-Kutta 45 integration scheme combined with semi-analytic bending simulation which successfully yields accurate results where standard double-precision bending model implementations fail.

Keywords: arbitrary precision, compliant mechanism, double precision, Euler-Bernoulli beam, Runge-Kutta

1. Introduction

The basic function of compliant mechanisms can be modeled by using shear-free beams. Several authors have sought semi-analytic models of these structures to decrease the need for computational resources [1–4]. We study the numerical solution of a single flexural element. Consider a flexure suspending a weight, $F_\mathrm{w} = mg$, and additionally a deflecting force F_s that acts at the end of the flexure. The flexure angle $\theta(s)$ and moment $M(s)$ given at a distance s from the origin along the neutral axis depend on boundary conditions, and elastic modulus E. We assume the flexure is clamped at $s = 0$ with an initial tangent angle of zero, shown in Figure 1. We can take the initial tangent angle to be zero without loss of generality. The flexure has a length along its neutral axis L. The geometry of deformation can be determined by the coupled differential equations

$$\frac{\mathrm{d}M}{\mathrm{d}s} = F_\mathrm{w}(s)\sin\left(\theta\left(s\right)\right) + F_\mathrm{s}(s)\cos\left(\theta\left(s\right)\right) \tag{1}$$

$$\frac{\mathrm{d}\theta}{\mathrm{d}s} = \frac{M(s)}{E(s)I(s)} \tag{2}$$

for the moment and angle [4]. In the simplest case, $E(s), F_\mathrm{w}(s)$, and $F_\mathrm{s}(s)$ are constant, i.e., independent of s. An example geometry, converted to Cartesian coordinates, is shown in Figure 1. The geometry of the

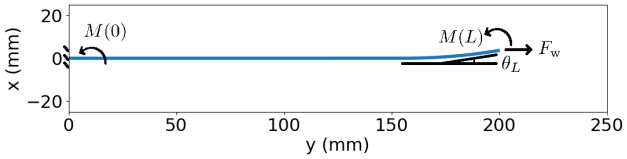


Figure 1: End torque bending to $\theta_L = 10°$ of a flexure with a rectangular cross section with varying thickness $h(s)$. The thickness is shown in Fig. 2. Note, the flexure is drawn lying on its side – gravity is acting on the $+y$ direction. . Add the value of $M(L)$ in caption.

unbent flexure's cross section is captured by $I(s)$, the second moment of area of the plane perpendicular to the neutral axis. The equation for the second moment of area is

$$I_\circ(s) = \frac{\pi}{4}\left(r\left(s\right)\right)^4, \text{ and} \tag{3}$$

$$I_\square(s) = \frac{1}{12}b\left(h\left(s\right)\right)^3, \tag{4}$$

for a circular cross section with radius $r(s)$ and a rectangular cross section with width b and thickness $h(s)$, respectively. In general, the cross-sectional

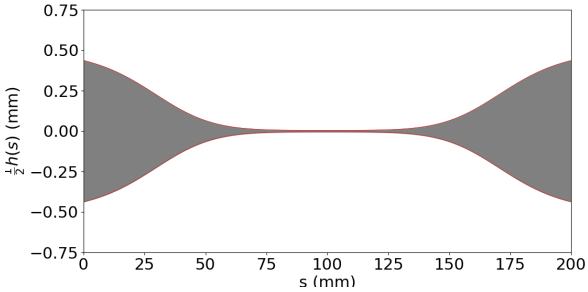


Figure 2: A thin planar flexure profile where for reasonable parameters ($E = 1.31 \times 10^5\,\mathrm{MPa}, F_\mathrm{w} = 1\,\mathrm{N}, b = 0.1\,\mathrm{mm}$) float64 is not sufficient when applying clamp-sided shooting. The flexure becomes 100 times smaller at its waist.

parameters vary as a function of s, necessitating a numerical solution.

A numerical solution of the bending of such a flexure can be obtained by ordinary differential equation (ODE) solvers that consider a single-sided boundary condition. Typical implementation of these solvers, for example in Python, use standard double precision which uses a 64 bits and is also referred to as float64 [5]. The boundary condition at the free end of the flexure, $\theta(s = L) = \theta_L$, is satisfied using the shooting method [6]. For example, a final bending angle of $\theta_L = 10°$ of a flexure is apparent at the curled end in Figure 1. The angle the flexure's tangent makes with the y axis at its end is θ_L, a quantity that is useful as a physically measurable boundary condition. Precision beyond float64 was used to solve for the deformation in Figure 1 because when numerically integrating the geometry $M(0)/M(L) \sim 10^{-596}$, which is outside the capability of float64, which can only represent nonzero numbers that are larger in magnitude than 2×10^{-308}.

This numerical requirement can be easily demonstrated for a flexure with a constant cross section, which can be analytically solved. According to Eq. (1) and Eq. (2) in [7] , we find

$$M(0) = 2M(L)e^{-\alpha L} + F_\mathrm{s}\alpha^{-1} \text{ with } \alpha^2 \coloneqq \frac{F_\mathrm{w}}{EI}. \tag{5}$$

In the absence of F_s, the ratio $M(0)/M(L)$ becomes exponentially smaller for decreasing I, eventually demanding many more exponent bits than float64 can provide.

The purpose of this article is to numerically solve the bending of very thin flexures. As mentioned above, for constant cross-sections, the system of differential equations can be solved analytically. In practice,

however, the flexures are engineered or constrained by specifications to have a non-constant cross-section, and a numerical calculation is the only feasible technique to solve the bending of the flexure. Nonlinear bending can be solved with finite element analysis or numerically, as the system of differential equations shown above with the Runge-Kutta method. The latter is faster because it only requires a linear number of steps in the size of the input, samples of $I(s)$. Hence, an ODE-based solver can speed up the development cycle for compliant mechanisms [8–10]. An alternative approach to solving this problem as an ODE would be to consider relaxation methods [6,11]. For flexures that are not super thin, Runge-Kutta solvers have been used previously. Typically, the ODE integration step in the shooting method starts from the clamped end ($s = 0$) where boundary conditions are well defined [1, 4].

1.1. The shooting method

The shooting method reduces a boundary value problem to an initial value problem, where the end conditions can be iteratively satisfied by varying initial conditions [6]. The iteration step can be formulated as a root-finding problem for an auxiliary function that enforces the boundary condition. For bending we satisfy the end angle boundary condition $\theta(s = L) = \theta(0)$. The end angle boundary condition is not guaranteed for arbitrary $M(0)$ and F_s. In our case, the root finding is on an auxiliary function G,

$$G(M(0), F_\mathrm{s}) = \theta(s = L, M(0), F_\mathrm{s}) - \theta_L. \qquad (6)$$

The root is found by varying $M(0)$ with a fixed F_s or vice versa. For simplicity we always fix exactly one of $M(0)$ and F_s. Evaluating the auxiliary function requires a numerical integration of Eq. 1 and Eq. 2 from $s = 0$ to $s = L$ for each iteration, since $\theta(L, M(0), F_\mathrm{s})$ appears. For bending calculations the shooting method is very effective, but when implemented with float64, it fails in very thin cases. Further, a definition of thin in terms of geometric, loading, and material properties of the flexure is given. Figure 2 shows a flexure geometry where a float64 solver would fail.

1.2. Floating-point representation

Limitations of numerical methods for differential equations based on fixed-size floating-point are well known. The specific problem of floating-point exponent limitations has been addressed for calculations regarding Legendre polynomials [12, 13]. Other examples such as Abad et al. show a unacceptably greater deviation from exact periodicity of numerical orbits with too few decimals of precision [14]. Double and single precision can be insufficient for numerically determining a choice of Lorenz attractor [15]. Limitations of floating-point representation can also lead to the appearance of anomalous solutions which are only exhibited by the numerically approximated system [16]. In some cases finding such errors can be automated using dynamic program analysis [17, 18].

A floating-point number N is represented on a computer using three binary variables, a sign bit ($\pm$, and two integers d and l as $N = \pm\frac{d}{D} \cdot 2^l$. The multi-bit wide value d and l are referred to as the mantissa and the exponent, respectively. The mantissa is normalized to be less than one with D. As the name implies, float64 uses a total of 64 bits to encode the binary N. According to the IEEE standard [5], the most significant bit, the sign bit, is followed by an 11-bit wide (signed) exponent and a 52-bit wide (unsigned) mantissa. In double precision, the exponent lies between $-1022 \leq l \leq 1023$, limiting the calculations to those which have an order of magnitude between 10^{-308} and 10^{308}. For the calculation of bending the limitations of float64's exponent prevent calculations that appear in practical cases. As we shall discuss below, float64 is not sufficient for the calculations of very thin flexuress. Quadruple precision or float128 extends this range enough for the thin flexures we attempted to model. Float128 uses a 15-bit wide exponent and a 112-bit wide mantissa [5]. Now, $-16382 \leq l \leq 16383$, which limits the exponents to 10^{-4932} and 10^{4932}. One can represent smaller numbers than 10^{-4932} down to 6.6×10^{-4966} by using very small numbers for the mantissa. However, by doing so, one trades precision for magnitude, and full loss of mantissa precision spoils the algorithm we use to solve bending. We use the library mpmath [19] for Python to perform our calculations. It implements floating-point calculations with an exponent range that exceeds the practical application of the bending model.

A common strategy to mitigate numerical rounding errors is to rescale the variables. For instance, rather than computing the moment $M(s)$ in N m, one might calculate it as a dimensionless ratio $M(s)/M_n$, where M_n is a very small moment, for example, 1×10^{-20} N m. However, two key issues limit the effectiveness of this approach. First, as shown in Eq. 5, the ratio $M(L)/M(0)$, comparing the moment at the free and fixed ends of the flexure, can span hundreds of orders of magnitude. A uniform scaling factor does not address this, as it cancels out in the ratio. Second, even if the angle, which may also grow exponentially by Eqn. 2, is rescaled as $\theta(s) = r(s)\theta_n(s)$, numerical instabilities persist. This is because of trigonometric functions which are sensitive to the absolute magnitude of their arguments. For example, for small angles $\theta(s)$, $\cos\theta(s) = \cos(r(s)\theta_n(s)) \approx 1 - 1/2 r(s)^2\theta_n(s)^2$, and, hence, one still needs the full numerical precision

Table 1: Parameters of a circular cross-section flexure used in [20]. The parameters above the single line are the geometric and mechanical properties from which the parameters below the line are calculated.

Par.	Eq.	Value	
E		7.3×10^{10}	N/m^2
L		6.00×10^{-1}	m
r		2.00×10^{-4}	m
F_w		1.47×10^{2}	N
I	$\pi r^4/4$	1.26×10^{-15}	m^4
α	$\sqrt{F_\mathrm{w}/(EI)}$	1.27×10^{3}	m^{-1}
αL		7.60×10^{2}	
$\lfloor 2^{l-1} \ln 2 \rfloor_{50}$	for $l = 12$	1.40×10^{3}	
$e^{\alpha L}$		1.16×10^{330}	
σ_w	$F_\mathrm{w}/(r^2\pi)$	1.176×10^{9}	N/m^2
$\sqrt{E/\sigma_\mathrm{W}}$		7.90	
L/r		3.00×10^{3}	

to obtain the correct result.

So, rescaling the variables with common factors does not lead to a satisfactory solution, but increasing the floating-point exponent size does without adding too much complexity.

2. Analytical solutions

2.1. Large exponents in bending

The problem that arises with exponent limitations in the outlined bending calculations is unintuitive because it involves scales smaller than the float64 minimum 10^{-308}. If something is so small it goes unmeasured. Numerically, this scale is of practical use because the growth of the flexure angle is exponential when the angle is small. Without this small nonzero initial $M(0)$ or a side force F_s, the numerical approximation problem is symmetric and would not favor a leftward or rightward bend geometry. The solution shown in Figure 1 was determined with a small initial value $M(0)$. With unbroken symmetry ($\theta(0) = 0, M(0) = 0\,\mathrm{N\,m}$), Equations 1 and 2 have null right-hand side. The result is zero bending for all s, which carries over to numerical solutions.

To achieve a bending solution, we may introduce a small but non-negligible moment at the clamp. We must represent such small values in our ODE solver, which is not possible with float64. For the flexure defined by the parameters in Table 1, achieving a final bending angle on the order of one radian requires an initial moment $M(0)$ of order $1 \times 10^{-330}\,\mathrm{N\,m}$. Even when $F_\mathrm{s} \neq 0$, the system exhibits similar exponential growth and corresponding limitations due to floating-point precision.

Equation 2 entails that when the geometry becomes extremely thin, i.e., $I(s)$ is small, the rate of change of $\theta(s)$ will become extremely large. We will explicitly show that the initial condition must be exponentially small in terms of the physical parameters of the flexure to achieve a measurable end angle. This, in turn, explains why the initial values in semi-analytic bending are extremely small numbers below the capability of float64.

2.2. Small angle solutions

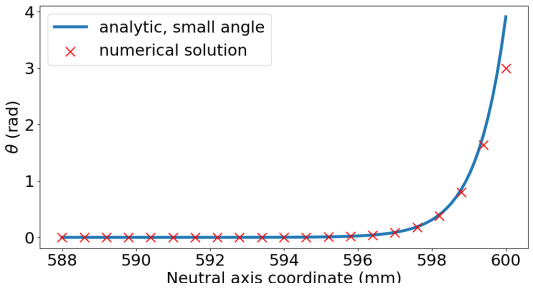
Hyperbolic trigonometric functions as solutions to beam deformation have been well documented in textbooks and are often used in practical research [7, 21]. Here, we investigate these approximate solutions in a form that is amenable to exposing exponent limitations. Any solution with $\theta(0) = 0$ will closely match the behavior of small angle solutions in some portion of the flexure length. Thus limitations we derive in the small angle affect large angle solutions.

To illustrate beyond a qualitative understanding that there is an exponential increase present in bending, consider the solutions where θ is small. For $F_\mathrm{s} = 0$, the second order equation obtained by combining Equation 1 and Equation 2 yields an exponential solution.

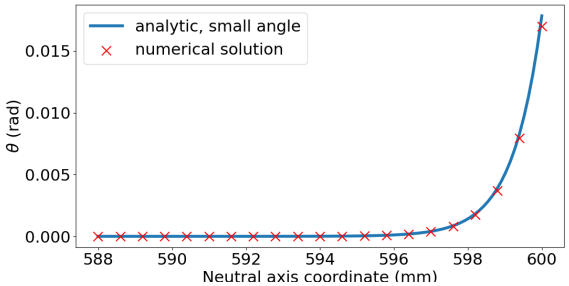
Consider a constant width geometry $I(s) = I_0$ and $\theta(0) = 0$. In this case, we can draw upon the solutions provided by Speake [7]. A minor but notable difference lies in the coordinate system: Speake formulates the problem using x and y, whereas we employ θ and arc length s. Within the framework of linear bending theory, the approximation $x \approx s$ holds simplifying change of coordinates. These are sufficiently accurate to capture and quantify the numerical instabilities under discussion. The compliance matrix given in [7] can be used to find the end bending angle, $\theta(L)$. To expose exponent limitations, we consider large growth in $M(s)$ and $\theta(s)$. For the analytical solutions [7], this means $\alpha L \gg 1$, with $\alpha^2 = F_\mathrm{w}/(EI_0)$. In this case, $\sinh \alpha L \approx \cosh \alpha L \approx \frac{1}{2} e^{\alpha L}$ and, consequently, $\tanh \alpha L \approx 1$. Using Eq. (1) and Eq. (2) in [7] together with the above approximations, we find that

$$M(L) \approx \frac{1}{2} \left(M\,(0) - \frac{F_\mathrm{s}}{\alpha} \right) e^{\alpha L}. \tag{7}$$

Since the term in parentheses is constant along the flexure, the exponential growth is due to the exponential. Further, $\theta(s)$ can be obtained by integrating $M(s)$ according to Eq. 2. Hence, exponential growth in $M(s)$ leads to the same growth in $\theta(s)$, see Fig. 3.



(a) Large angle bend



(b) Small angle bend

Figure 3: Bending angle near the end of a torque bending flexure with parameters as in Table 1. Insets (a) and (b) show small and large deflections, respectively. For (a), the small-angle solution deviates from the numerical solver since the bend angle is sufficiently large.

Consequently, the final bending angle is given using Eq. (3) in [7] by

$$\theta(L) \approx \frac{F_\mathrm{s}}{F_\mathrm{w}}(1-2e^{-\alpha L}) + \frac{\alpha M(L)}{F_\mathrm{w}}. \tag{8}$$

Finally we can substitute $M(L)$ and ignore terms that are not exponentially large in αL. We find a form that lets us study the floating-point exponent range needed to achieve a small target end angle θ_L,

$$\theta(L) \approx \frac{1}{2F_\mathrm{w}}\left(\alpha M\left(0\right) - F_\mathrm{s}\right)e^{\alpha L} \quad \text{for} \quad \alpha L \gg 1. \tag{9}$$

2.2.1. Definition of a long thin flexure

The exponential analytical solution given above allows us to formally define what we mean by a thin flexure for a calculation with a certain floating-point exponent size. Assume a circular cross-section flexure is loaded with force F_w, which provides a stress σ_w. Further, we have a larger exponent than the l bit floating-point exponent in the scaling of the initial condition $M(0)$, assuming $F_\mathrm{s} = 0$ for simplicity. We have two conditions up to small constants,

$$\sigma_\mathrm{w} = \frac{F_\mathrm{w}}{\pi r^2} \tag{10}$$

$$2L\sqrt{\frac{F_\mathrm{w}}{E\pi r^4}} > 2^{l-1}\ln 2. \tag{11}$$

Combining Eq. 10 and Eq. 11 produces a succinct approximate criterion of an extreme aspect ratio,

$$\frac{L}{r} > \underbrace{\frac{\ln 2}{4}}_{0.17}\sqrt{\frac{E}{\sigma_\mathrm{w}}}2^l. \tag{12}$$

We now find a computational success criterion applicable for arbitrary flexure geometry. We compare the scale factor $e^{\alpha L}$ with the maximum scaling representable with a floating-point number

$$e^{\alpha L} < 2^{2^{l-1}}. \tag{13}$$

Taking a logarithm and rounding down to the nearest lower number that is divisible by N, written as $\lfloor\cdots\rfloor_N$, we preserve the inequality to get

$$\alpha L < \beta := \lfloor 2^{l-1}\ln 2\rfloor_N. \tag{14}$$

We chose $N = 50$ somewhat arbitrarily, but it is reasonable to do heavy rounding to ensure numerical stability. Now we make the bound applicable to simulating flexures with variable cross-section geometry, $I(s)$ not constant. To overestimate the left hand side assume the characteristic exponential growth length scale $\alpha^{-1} = (\frac{F_\mathrm{w}}{EI(s)})^{-\frac{1}{2}}$ is at a minimum along the flexure's entire length L. To achieve this maximize the inverse length scale as $max_s(\sqrt{\frac{F_\mathrm{w}}{EI(s)}})$. We now have a condition that determines the floating-point type needed for bending

$$\max_s(\sqrt{\frac{F_\mathrm{w}}{EI(s)}})L < \beta. \tag{15}$$

We provide Table 2, which informs one of the required floating-point precision for solving bending. For bending parameters from Table 1 we have $\alpha L = 760 > \beta_\mathrm{double} = 700$ so double precision is inadequate to calculate bending.

3. Numerical solution

We assume the target bending angle θ_L is less than $\frac{\pi}{2}$ rad (extended to π rad for $F_\mathrm{s} = 0$) to avoid an oscillating regime on $\theta(s)$ and $M(s)$ which could make shooting for solutions more difficult. The shooting method alone is sufficient to find initial conditions which bend to θ_L. We conceptually separate shooting

Table 2: Lookup table for bending solver IEEE754 precision. $\max_s(\sqrt{\frac{F_w}{EI(s)}})L < \beta$ means the precision is sufficient. The column ϵ gives the smallest full-precision positive value in the corresponding number format.

name	total bits	exp. bits, l	ϵ	β $\lfloor 2^{l-1}\ln 2 \rfloor_{50}$
single	32	8	1×10^{-38}	50
double	64	11	2×10^{-308}	700
quadruple	128	15	3.3×10^{-4932}	11 350
octuple	256	19	$1.5 \times 10^{-78\,913}$	181 700

for initial conditions which result in $\theta(L) = \theta_L$ into two stages. First, we provide an initial guess based on linearity under small-angle bending. Then we perform nonlinear shooting. Roughly speaking, for bending the first stage will yield the exponent of the initial condition, the second yields the mantissa.

3.1. Small angle step

We set the varied initial condition $M(0)$ or F_s to be as small as possible without sacrificing mantissa precision. We call this the trial initial condition $M^\star(0)$ or $F_s^\star$. We then integrate this solution with our arbitrary precision Runge-Kutta 45(APRK45) solver. We call this the trial solution. We determine an ancillary end bending angle $\theta^\star$ for the trial solution. Then the guess for the shooting stage is determined as $M(0) = M^\star(0)\theta_L/\theta^\star$. If the desired θ_L is not small, we rescale $M(0) = M^\star(0)\theta_{\text{small}}/\theta^\star$ to achieve $\theta_{\text{small}} = 0.1$ rad. We must first aim for a small bending angle to surely avoid nonlinear shooting on bending to angles larger than $\pi/2$ rad. Above this limit trigonometric functions change sign complicating search for initial conditions. The single ODE integration and direct scaling step are simply leveraging the linear nature of the approximate ODE to rescale the trial solution to better achieve bending to an angle with order of magnitude close to θ_L. This can be seen as a trivial case of shooting where the ODE is linear [6]. In studying nonlinear bending, we must proceed further.

3.2. Shooting

A shooting step is done starting with the trial solution derived guess for either F_s or $M(0)$ with the APRK45 solver for side force significant and side force insignificant cases respectively. We have found that the Anderson-Björck root finding method [22] was best applied to the shooting step, rapidly converging on the correct initial condition for a wide range of cases. We give pseudocode for the algorithm we have described in this section as Algorithm 1. Shooting for F_s instead of $M(0)$ is identical except in which parameter is modified.

Algorithm 1 Semi-analytic bending

procedure SHOOT $M(0)$
  $\theta_{\text{small}} \leftarrow 0.1$ rad
  $M(0) \leftarrow$ smallest magnitude floating-point value
  $\theta^\star \leftarrow \text{APRK45}(M(0))$
  if $\theta_L < \theta_{\text{small}}$ **then**
    $M(0) \leftarrow \frac{\theta_L}{\theta^\star} M(0)$
  else
    $M(0) \leftarrow \frac{\theta_{\text{small}}}{\theta^\star} M(0)$
  $S \leftarrow (\frac{M(0)}{32}, 32M(0))$
  return $\text{root}(\theta_L - \text{APRK45}(x), S)$

3.3. Speed and error

The fixed step-size APRK45 solver we implemented leverages fourth- and fifth-order coefficients to provide calculable and sufficiently small error. The geometry $I(s)$ is sampled at a fixed interval and cubic spline interpolated to feed the solver. We generate stepwise error estimates. Errors for our parameters were much smaller than our tolerance with APRK45 as shown in Figure 4 when run on a flexure with parameters as in Figure 2. These parameters can only be solved by shooting from the clamped end if we can represent $M(0) \sim 1 \times 10^{-596}\,\text{N m}$. The time to do shooting for a single bend for the provided plots was about $5 - 50$ seconds on a personal computer with Python implementation. Our implementation has been shared on github: usnistgov/BeamBending.

To validate our numerical solver, we compare it against the analytical solution for a constant cross section with ($F_s = 0$), where the solution involves only the hyperbolic sine function. These solutions exhibit exponential growth and, in practical scenarios, cannot be numerically approximated with double precision arithmetic. We choose the test case of a constant circular cross-section with the parameters shown in Table 1. This calculation requires a larger floating-point format than double. We calculate bending to

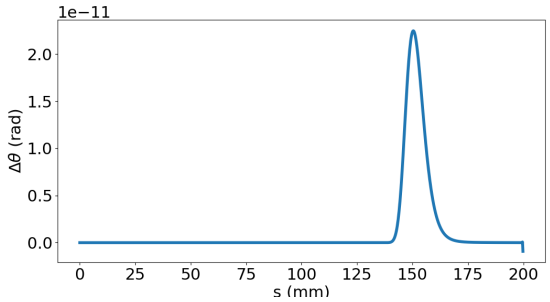


Figure 4: Point-wise error estimates for $\theta(s)$ solving $F_s = 0$ bending with 1000 uniform Runge-Kutta steps for a ribbon as in Figure 2 with bending geometry shown in Figure 1.

final angles of 3 rad and 1°. Practically 3 rad is an extreme bend but serves to well exhibit the small angle approximation. When the angle of bending is still small the numerical result and the approximate analytic solution should coincide. Figure 3 shows this test case.

4. Conclusions

For very thin flexures, we have determined that it is feasible to calculate their deformation using semi-analytic methods. This was shown using approximate analytic solutions and arbitrary precision floating-point to exhibit limits of the double precision exponent. To integrate with existing tools in order to accommodate thin flexures double precision values can be exchanged for higher precision values according to Table 2.

References

[1] L. Keck, S. Schlamminger, R. Theska, F. Seifert, and D. Haddad. Flexures for kibble balances: minimizing the effects of anelastic relaxation. *Metrologia*, 61(4):045006, Jul 2024.

[2] S. Henning and L. Zentner. Analytical characterization of spatial compliant mechanisms using beam theory. In *Microactuators, Microsensors and Micromechanisms*, pages 61–76. Springer International Publishing, 2023.

[3] V. Platl and L. Zentner. An analytical method for calculating the natural frequencies of spatial compliant mechanisms. *Mech. Mach. Theory*, 175:104939, 2022.

[4] S. Henning and L. Zentner. Analysis of planar compliant mechanisms based on non-linear analytical modeling including shear and lateral contraction. *Mech. Mach. Theory*, 164:104397, 2021.

[5] Ieee standard for floating-point arithmetic. *IEEE Std 754-2019 (Revision of IEEE 754-2008)*, pages 1–84, 2019.

[6] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes: The Art of Scientific Computing*. Cambridge University Press, 3rd edition, 2007.

[7] C. C. Speake. Anelasticity in flexure strips revisited. *Metrologia*, 55(1):114, Jan 2018.

[8] L. Keck, G. Shaw, R. Theska, and S. Schlamminger. Design of an electrostatic balance mechanism to measure optical power of 100 kW. *IEEE Transactions on Instrumentation and Measurement*, 70:1–9, 2021.

[9] Lorenz Keck. *Flexure-based mechanism for a Kibble Balance*. PhD thesis, Ilmenau, Feb 2025. Dissertation, Technische Universität Ilmenau, 2024.

[10] L. Keck, K. Arumugam, L. Chao, Z. Comden, F. Seifert, D.B. Newell, D. Haddad, and S. Schlamminger. Thoughts on the Kibble-Robinson theory. *Metrologia*, 62(2):025012, mar 2025.

[11] N. Perrone and R. Kao. A general nonlinear relaxation iteration technique for solving nonlinear problems in mechanics. *J. Appl. Math. Mech.*, 38(2):371–376, Jun 1971.

[12] J. M. Smith, Olver F. W. J., and D. W. Lozier. Extended-range arithmetic and normalized legendre polynomials. *ACM Trans. Math. Softw.*, 7:93–105, 1981.

[13] T. Fukushima. Numerical computation of spherical harmonics of arbitrary degree and order by extending exponent of floating point numbers. *J. Geod.*, 86(4):271–285, Apr 2012.

[14] A. Abad, R. Barrio, and Á. Dena. Computing periodic orbits with arbitrary precision. *Phys. Rev. E*, 84:016701, Jul 2011.

[15] P. Wang, G. Huang, and Z. Wang. Analysis and application of multiple-precision computation and round-off error for nonlinear dynamical systems. *Adv. Atmos. Sci.*, 23:758–766, 2006.

[16] E. Allen, J. Burns, D. Gilliam, J. Hill, and V. Shubov. The impact of finite precision arithmetic and sensitivity on the numerical solution of partial differential equations. *Math. Comput. Model.*, 35(11):1165–1195, 2002.

[17] F. Benz, A. Hildebrandt, and S. Hack. A dynamic program analysis to find floating-point accuracy problems. *SIGPLAN Not.*, 47(6):453–462, 2012.

[18] E. T. Barr, T. Vo, V. Le, and Z. Su. Automatic detection of floating-point exceptions. *SIGPLAN Not.*, 48(1), 2013.

[19] Fredrik Johansson et al. mpmath: a Python library for arbitrary-precision floating-point arithmetic, 2023. Accessed: 2025-05-28.

[20] S. M. Aston, M. A. Barton, A. S. Bell, N. Beveridge, B. Bland, A. J. Brummitt, G. Cagnoli, C. A. Cantley, L. Carbone, A. V. Cumming, L. Cunningham, R. M. Cutler, R. J. S. Greenhalgh, G. D. Hammond, K. Haughian, T. M. Hayler, A. Heptonstall, J. Heefner, D. Hoyland, J. Hough, R. Jones, J. S. Kissel, R. Kumar, N. A. Lockerbie, D. Lodhia, I. W. Martin, P. G. Murray, J. O'Dell, M. V. Plissi, S. Reid, J. Romie, N. A. Robertson, S. Rowan, B. Shapiro, C. C. Speake, K. A. Strain, K. V. Tokmakov, C. Torrie, A. A. van Veggel, A. Vecchio, and I. Wilmut. Update on quadruple suspension design for Advanced LIGO. *Class. Quantum Gravity*, 29(23):235004, December 2012.

[21] T. J. Quinn, C. C. Speake, and R. S. Davis. A 1 kg mass comparator using flexure-strip suspensions: Preliminary results. *Metrologia*, 23(2):87, Jan 1986.

[22] N. Anderson and Å. Björck. A new high order method of regula falsi type for computing a root of an equation. *BIT Numer. Math.*, 13:253–264, 1973.