Local Slender Body Theory in the Chebyshev Basis

Ondrej Maxian

May 17, 2019

1 Method formulation

We are interested in modeling inextensible, slender fibers using slender body theory. Thus our task is to solve the PDE

$$\frac{\partial \boldsymbol{X}}{\partial t} = \boldsymbol{M}\left(\boldsymbol{f}^E + \boldsymbol{\lambda}\right), \tag{1}$$

where $\boldsymbol{M}$ is a mobility matrix that comes from slender body theory. $\boldsymbol{f}^E$ are the bending forces on the fiber, and $\boldsymbol{\lambda} = (T\boldsymbol{X}_s)_s$ are tensile forces that act as Lagrange multipliers enforcing inextensibility of the fiber. The fibers are assumed to be free, and are therefore subject to the boundary conditions, $\boldsymbol{X}_{ss}(s=0,L) = \boldsymbol{X}_{sss}(s=0,L) = \boldsymbol{0}$.

1.1 Slender body theory

In slender body theory, the 3D slender fiber with aspect ratio $\epsilon = r(L/2)/L$ is reduced to a one dimensional curve along the fiber centerline via an asymptotic expansion in ϵ. The slender body mobility operator is given by

$$\boldsymbol{M}[\boldsymbol{f}](s) = \frac{1}{8\pi\mu}\left(\boldsymbol{\Lambda}[\boldsymbol{f}](s) + \boldsymbol{J}[\boldsymbol{f}](s)\right), \tag{2}$$

where $\boldsymbol{f}$ is the force per length on the fiber. The operator $\boldsymbol{M}$ is decomposed into a local operator $\boldsymbol{\Lambda}$ given by

$$\boldsymbol{\Lambda} = -\log(\epsilon^2 e)(\boldsymbol{I} + \boldsymbol{X}_s\boldsymbol{X}_s) + 2(\boldsymbol{I} - \boldsymbol{X}_s\boldsymbol{X}_s). \tag{3}$$

Notice that, because the fiber is inextensible, $\boldsymbol{X}_s$ is assumed to be a unit vector, and $\boldsymbol{X}_s\boldsymbol{X}_s$ is an outer product. Using the $\mathcal{O}(\log\epsilon^2)$ term of $\boldsymbol{\Lambda}$ in place of $\boldsymbol{M}$ is synonymous with the *local drag model*.

The $\mathcal{O}(1)$ non-local integral operator $\boldsymbol{J}$ is given by

$$\boldsymbol{J}[f](s) = \int_0^L \left(\frac{\boldsymbol{I} + \hat{\boldsymbol{R}}(s,s')\hat{\boldsymbol{R}}(s,s')}{\|\boldsymbol{R}(s,s')\|}\boldsymbol{f}(s') - \frac{\boldsymbol{I} + \boldsymbol{X}_s(s)\boldsymbol{X}_s(s)}{|s-s'|}\boldsymbol{f}(s)\right) ds', \tag{4}$$

where $\boldsymbol{R}(s,s') = \boldsymbol{x}(s) - \boldsymbol{x}(s')$ and $\hat{\boldsymbol{R}}$ is the corresponding unit vector. We will put off study of the non-local integral until a later time.

For the rest of this report, unless otherwise indicated, we set $M = \Lambda$ to compare with the data received from Floren.

1.2 Fiber mechanics

The bending force is given by

$$\boldsymbol{f}^E = -E\boldsymbol{X}_{ssss} \tag{5}$$

subject to "free fiber" boundary conditions $\boldsymbol{X}_{ss}(s=0,L) = \boldsymbol{X}_{sss}(s=0,L) = \boldsymbol{0}$ [3]. See Section 1.3.1 for details on how we actually enforce these BCs.

It is easy to see that these BCs cause the total force and torque from $\boldsymbol{f}^E$ on the fiber to be zero in the continuum sense

$$\int_0^L \boldsymbol{f}^E \, ds = -E\boldsymbol{X}_{sss}\Big|_0^L = \boldsymbol{0} \tag{6}$$

and

$$\left(\int_0^L \boldsymbol{f}^E \times \boldsymbol{X} \, ds\right)_k = \int_0^L X^i X^j_{ssss} - X^j X^i_{ssss} = \int_0^L X^i_{ss} X^j_{ss} - X^j_{ss} X^i_{ss} = 0, \tag{7}$$

where (i, j, k) is a cyclic permutation of $(1, 2, 3)$. In the torque equation, the free fiber boundary conditions lead to the cancellation of boundary terms.

1.3 Spectral discretization

1.3.1 Rectangular spectral collocation

We use rectangular spectral collocation [1, 2] to solve Eq. (1). Following the convention of [2], we solve the PDE on a type 1 Chebyshev grid with N points (i.e. the grid where the PDE is posed *does not* include the boundary points). The boundary conditions are imposed by upsampling the relevant quantities to a type 2 Chebyshev grid (that includes the endpoints) with $\tilde{N} = N + 4$ points, since there are 4 BCs. This procedure is analogous to using ghost cells in finite difference methods. For the rest of this report, any quantity defined on the type 2 grid is marked with a tilde. Given the values of $\boldsymbol{X}$ on a type 1 Chebyshev grid with N points, there is a unique configuration $\tilde{\boldsymbol{X}}$ on the type 2 grid that satisfies

$$\begin{pmatrix} \boldsymbol{R} \\ \boldsymbol{L} \end{pmatrix} \tilde{\boldsymbol{X}} = \begin{pmatrix} \boldsymbol{I_N} \\ \boldsymbol{0} \end{pmatrix} \boldsymbol{X}. \tag{8}$$

Here $\boldsymbol{R}$ is the downsampling operator that intepolates the data on the type 1 N point grid from the data on an $\tilde{N} = N+4$ type 2 grid, and $\boldsymbol{L}$ is the operator that encodes the boundary conditions $\tilde{\boldsymbol{X}}_{ss}\,(s = 0, L) = \tilde{\boldsymbol{X}}_{sss}\,(s = 0, L)$ on the *type 2 grid*. Because $\boldsymbol{R}$ is rectangular, the right hand side of Eq. (8) is invertible and we can therefore write

$$\tilde{\boldsymbol{X}} = \begin{pmatrix} \boldsymbol{R} \\ \boldsymbol{L} \end{pmatrix}^{-1} \begin{pmatrix} \boldsymbol{I_N} \\ \boldsymbol{0} \end{pmatrix} \boldsymbol{X} = \boldsymbol{E}\boldsymbol{X} \tag{9}$$

Thus at each timestep, there is a unique function $\tilde{\boldsymbol{X}}$ on the type 2 grid that satisfies Eq. (9). In finite difference schemes, there are unique values of the "ghost cells" that allow the boundary stencils to satisfy the BCs to some order. Thus the rectangular spectral collocation method can be thought of as an extension of ghost cell techniques for finite difference methods.

The function $\tilde{\boldsymbol{X}}$ can be used to compute $\boldsymbol{f}^E$ in a way consistent with the boundary conditions. That is, $\tilde{\boldsymbol{f}}^E = -E\tilde{\boldsymbol{X}}_{ssss}$ is computed on the *type 2* grid and then downsampled via the operator $\boldsymbol{R}$. This is so that the bending force is computed in a way consistent with the boundary conditions (analogous to using ghost cells at the boundaries to compute the fourth derivative of $\boldsymbol{X}$). We write the downsampled bending force as

$$\boldsymbol{f}^E = \boldsymbol{R}\tilde{\boldsymbol{f}}^E = \boldsymbol{R}\tilde{\boldsymbol{D}}^4\tilde{\boldsymbol{X}} = \boldsymbol{R}\tilde{\boldsymbol{D}}^4\boldsymbol{E}\boldsymbol{X} := \boldsymbol{L}\boldsymbol{X}. \tag{10}$$

So that we have defined the bending force on the type 1 grid, $\boldsymbol{f}^E = \boldsymbol{L}\boldsymbol{X}$. Noitice that $\boldsymbol{L}$ is a *constant* matrix (i.e. not a function of $\boldsymbol{X}$).

1.3.2 Inextensibility

If the fiber is inextensible, the arclength parameter s is material. Therefore $\boldsymbol{X}_s \cdot \boldsymbol{X}_s = 1$, and, differentiating with respect to time, $\boldsymbol{u}_s \cdot \boldsymbol{X}_s = 0$. Thus in three dimensions, $\boldsymbol{u}_s = \sum_{j=1}^{2} \beta_j \boldsymbol{n}_j$, where $\boldsymbol{n_j}$, $j = 1, 2$ is a vector normal to the fiber centerline.

Therefore, we can represent the velocity in a continuum setting as

$$\boldsymbol{u}(s) = \boldsymbol{U}^r + \int_0^s \sum_{j=1}^{2} \sum_{k=0}^{N-2} \alpha_{jk} T_k(s') \boldsymbol{n_j}(s') \, ds', \tag{11}$$

where T_k is the Chebyshev polynomial of the first kind of degree k and $\boldsymbol{U}^r$ is a constant translation. This definition of velocity ensures that $\boldsymbol{u}_s \cdot \boldsymbol{X}_s = 0$. We discretize Eq. (11) by computing the integral with the pseudo-inverse of the Chebyshev differentiation matrix,

$$\boldsymbol{u} = \boldsymbol{K}\boldsymbol{\alpha} = \boldsymbol{U}^r + \boldsymbol{D}^+\boldsymbol{a} \quad \text{where} \tag{12}$$

$$\boldsymbol{a} = \sum_{j=1}^{2}\sum_{k=0}^{N-2} \alpha_{jk} T_k(s)\boldsymbol{n_j}(s). \tag{13}$$

Here we have defined the matrix $\boldsymbol{K}(\boldsymbol{X})$ which acts on the $2N+1$ vector of $\boldsymbol{\alpha}$ and $\boldsymbol{U}^r$ to give the velocity of the fiber centerline. When we write linear systems, we will always assume that $\boldsymbol{U}^r$ make up the last three entries of $\boldsymbol{\alpha}$.

Uniqueness of representation. If $\boldsymbol{K}\boldsymbol{\alpha}(s_i) = \boldsymbol{0}$ for all $i = 1, \dots N$, we look for a condition that ensures $\boldsymbol{\alpha} = \boldsymbol{0}$. Suppose that we choose $\boldsymbol{n}_1$ to have a zero entry at position p and $\boldsymbol{n}_2$ to have a non-zero entry at position p. Then we can write

$$\left(\boldsymbol{K}\boldsymbol{\alpha}(s_i)\right)_p = \boldsymbol{U}^r + \sum_{k=0}^{N-2} \boldsymbol{D}^+ \left(\alpha_{1k} T_k \cdot 0\right)(s_i) + \boldsymbol{D}^+ \left(\alpha_{2k} T_k n_2^p\right)(s_i) = \boldsymbol{U}^r + \sum_{k=0}^{N-2} \boldsymbol{D}^+ \left(\alpha_{2k} T_k n_2^p\right)(s_i) = 0 \tag{14}$$

for the pth component of velocity, where $n_2^p(s)$ is some nonzero function of s. So if $\left(\boldsymbol{K}\boldsymbol{\alpha}(s_i)\right)_p = 0$ for all i, then $\left(\boldsymbol{K}\boldsymbol{\alpha}(s)\right)_p$ has N zeros. So a necessary condition for an empty null space of $\boldsymbol{K}$ is that $\left(\boldsymbol{K}\boldsymbol{\alpha}(s)\right)_p$ be a polynomial of degree $N-1$ or less. Because of the integration operator $\boldsymbol{D}^+$, this means we can only include Chebyshev modes up to $N-2$. Note that this is a *necessary* condition, not a suficient one, since in practice we cannot know the form of the normal vectors (those could be high order polynomials).

Now, we can write the velocity equation on the type 1 grid as

$$\boldsymbol{M}(\boldsymbol{\lambda} + \boldsymbol{f}^E) = \boldsymbol{K}\boldsymbol{\alpha} \tag{15}$$

Here the $3N$ Lagrange multipliers $\boldsymbol{\lambda}$ are stand-ins for the tensile forces $(T\boldsymbol{X}_s)_s$ required to maintain inextensibility, and $\boldsymbol{f}^E$ is the bending force defined in Eq. (10).

1.3.3 System of equations

We can write a system of equations for $\boldsymbol{\lambda}$ and $\boldsymbol{\alpha}$ as

$$\boldsymbol{A}\boldsymbol{y} = \begin{pmatrix} -\boldsymbol{M} & \boldsymbol{K} \\ \boldsymbol{K}^* & \boldsymbol{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\lambda} \\ \boldsymbol{\alpha} \end{pmatrix} = \begin{pmatrix} \boldsymbol{M}\boldsymbol{f}^E \\ \begin{pmatrix} \boldsymbol{0} \\ -\boldsymbol{I}^*\boldsymbol{f}^E \end{pmatrix} \end{pmatrix} \tag{16}$$

Where we again note that $\boldsymbol{f}^E = \boldsymbol{L}\boldsymbol{X}$ as defined in Eq. (10). The operator $\boldsymbol{A}$ has an obvious saddle point structure. $\boldsymbol{A}$ is not invertible generally because the representation $\boldsymbol{K}\boldsymbol{\alpha}$ is not necessarily unique. We therefore solve the system in the least squares sense with a tolerance of 10^{-6}.

Note that in Eq. (16), the matrices $\boldsymbol{M}$ and $\boldsymbol{K}$ are functions of $\boldsymbol{X}$. That is, $\boldsymbol{M} = \boldsymbol{M}(\boldsymbol{X})$ and $\boldsymbol{K} = \boldsymbol{K}(\boldsymbol{X})$.

The action of the operators is defined as follows. We have already discussed the velocity equation. The adjoint condition of $\boldsymbol{\lambda}$ is defined as

$$\boldsymbol{K}^*\boldsymbol{\lambda} = \sum_{i=1}^{N} \boldsymbol{\lambda}(s_i) \cdot \boldsymbol{D}^+ \left(\boldsymbol{n}_j T_k\right)(s_i)\, w_i = 0, \text{together with} \tag{17}$$

$$\sum_{i=1}^{N} \boldsymbol{\lambda}(s_i) w_i = -\sum_{i=1}^{N} \boldsymbol{f}^E(s_i) w_i, \tag{18}$$

for all Chebyshev degrees $k = 0, \dots N-2$ and normal vectors $\boldsymbol{n}_j$, $j = 1, 2$.

Physically, Eq. (17) states that the constraint forces $\boldsymbol{\lambda}$ can do no work, since the power is given by

$$\mathcal{P} = \int_0^L \boldsymbol{u}(s) \cdot \boldsymbol{\lambda}(s)\, ds \approx \tag{19}$$

$$= \int_0^L \boldsymbol{U}^r \cdot \boldsymbol{\lambda}(s)\, ds + \int_0^L \boldsymbol{D}^+ \left(\sum_{j=1}^{2} \sum_{k=0}^{N-2} \alpha_{jk} T_k \boldsymbol{n_j} \right)(s) \cdot \boldsymbol{\lambda}(s)\, ds = 0 \tag{20}$$

Since this equation must hold for every feasible motion $\boldsymbol{U}^r$ and $\boldsymbol{\alpha}$, each term in Eq. (20) must be zero. Let us focus on the last term for now. Discretizing the integral using Clenshaw-Curtis quadrature,

$$\mathcal{P}_3 = \sum_{i=1}^{N} \sum_{j=1}^{2} \sum_{k=0}^{N-2} \alpha_{jk} \boldsymbol{D}^+ \left(T_k \boldsymbol{n_j}\right)(s_i) \cdot \boldsymbol{\lambda}(s_i) w_i \tag{21}$$

$$= \sum_{k=0}^{N-2} \sum_{j=1}^{2} \alpha_{jk} \sum_{i=1}^{N} \boldsymbol{D}^+ \left(T_k \boldsymbol{n_j}\right)(s_i) \cdot \boldsymbol{\lambda}(s_i) w_i = 0, \tag{22}$$

which must hold for any α_{jk}. In particular, it must hold for $\alpha_{jk} = \delta_{1j}\alpha_k$ and $\alpha_{jk} = \delta_{2j}\alpha_k$, and so we have that *for each j and k*,

$$\sum_{i=1}^{N} \boldsymbol{D}^+ \left(\boldsymbol{n}_j T_k\right)(s_i) \cdot \boldsymbol{\lambda}(s_i) w_i = 0, \tag{23}$$

which is Eq. (17).

Since $\boldsymbol{U}^r$ is an arbitrary constant, the first term in Eq. (20) can be discretized as

$$\boldsymbol{0} = \int \boldsymbol{\lambda}(s)\, ds \approx \sum_{i=1}^{N} \boldsymbol{\lambda}(s_i) w_i := \boldsymbol{I}^* \boldsymbol{\lambda}, \tag{24}$$

where we have defined the discrete integration matrix $\boldsymbol{I}^*$. In Eq. (16), we enforce Eq. (24) up to discretization errors in $\int \boldsymbol{f}^E(s)\, ds \approx \boldsymbol{I}^* \boldsymbol{f}^E$. Although $\int \boldsymbol{f}^E(s)\, ds = \boldsymbol{0}$ in the continuous case, this does not necessarily hold discretely. We therefore keep the term $\boldsymbol{I}^* \boldsymbol{f}^E$ in our discretization to enforce the condition that the *total force on the fiber is zero exactly in the discrete setting*. Thus Eq. (18) enforces the constraint that $\boldsymbol{\lambda}$ does no work on the type 1 grid (up to discretization errors in integrating $\boldsymbol{f}^E$).

1.3.4 Temporal update

At each time-step, we solve Eq. (16) for $\boldsymbol{\alpha}$ *in the least squares sense*, set $\boldsymbol{u} = \boldsymbol{K}\boldsymbol{\alpha}$, and evolve the positions $\boldsymbol{X}$ via temporal integration. We experiment with different temporal integrators in Section 2.3.

2 Results

In this section, we compare the output of our algorithm directly with that of Floren for a fiber with initial tangent vector

$$\boldsymbol{X}_s(s, t=0) = \frac{1}{\sqrt{2}} \begin{pmatrix} \cos\left(s^3(s-L)^3\right) \\ \sin\left(s^3(s-L)^3\right) \\ 1 \end{pmatrix}. \tag{25}$$

This choice of tangent vector satisfies the boundary conditions $\boldsymbol{X}_{ss}\,(s=0,L) = \boldsymbol{X}_{sss}\,(s=0,L) = \boldsymbol{0}$ and the inextensibility constraint. Because there is no analytical solution for its configuration, we integrate Eq. (25) forward in s with $\boldsymbol{X}(s=0, t=0) = \boldsymbol{0}$ to arrive at a configuration of the form $(x(s), y(s), s/\sqrt{2})$. The 2D projection of the fiber (and therefore the form of the functions $x(s)$ and $y(s)$) is shown in Fig. 1 as a blue curve. We also show in Fig. 1 the fiber configuration at $t = 0.01$ for $N = 8$ for both our code

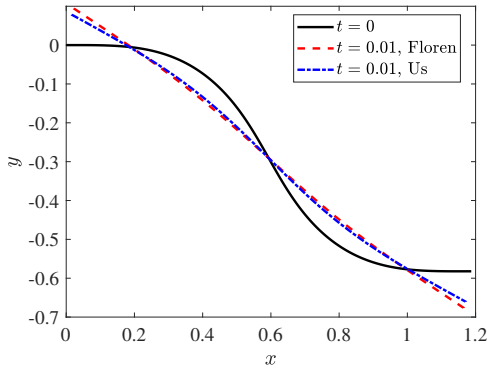


Figure 1: Fiber configuration for code comparison (2D projection). The black line shows the initial configuration at $t = 0$. The initial z coordinate is simply $z(s) = s/\sqrt{2}$. The final position of the fiber for $N = 8, \tilde{N} = 12$ points is shown as a dashed red line (for Floren's data) and a dashed/dotted blue line (for our data). Details: Floren's data has $\tilde{N} = 12$, $\Delta t = 10^{-6}$, our data has $N = 8$, $\Delta t = 10^{-4}$ (forward Euler update).

and Floren's code ($\tilde{N} = 12$ - Floren uses a type 2 grid, so all of his data is marked with a tilde). Note the differences at the boundaries; we will establish which is more accurate in subsequent sections.

In general, we will use an L^2 function norm to compute the differences between configurations throughout this section. Given two configurations $\boldsymbol{X}$ and $\boldsymbol{Y}$, the L^2 norm of their difference is defined as

$$E[\boldsymbol{X}, \boldsymbol{Y}](t) = \left(\int_0^L \|\boldsymbol{X}(t) - \boldsymbol{Y}(t)\|^2 \, ds \right)^{1/2} = \left(\sum_{i=1}^{1000} \|\boldsymbol{X}(s_i, t) - \boldsymbol{Y}(s_i, t)\|^2 \, w_i \right)^{1/2}. \tag{26}$$

The final term denotes the fact that we upsample each configuration to a common type 2 grid of 1000 points.

2.1 Convergence

2.1.1 Fiber tension at $t = 0$

We compare the values of $\boldsymbol{\lambda}$ with $(T\boldsymbol{X}_s)_s$ at $t = 0$. The latter quantity is obtained by mutiplying the analytically computed (from Eq. (25)) $\boldsymbol{X}_s$ by Floren's provided values of tension $T(t = 0)$ at each point, and then applying the spectral differentiation matrix. We measure the error of each against the "exact solution" of Floren's tensile force for $\tilde{N} = 24$. Fig. 2 shows that $\boldsymbol{\lambda}$ is indeed converging to $(T\boldsymbol{X}_s)_s$. Indeed, the L^2 error in $\boldsymbol{\lambda}$ for $N = 8$ is 2.81, whereas for $N = 20$ the L^2 error in $\boldsymbol{\lambda}$ is 8.3×10^{-3}.

2.2 Spatial error

We received 2 simulations from Floren: one with $\tilde{N} = 12$ points and one with $\tilde{N} = 24$. In this section, we compare the results from the algorithms for both $N = 8$ and $N = 20$. In order to isolate the *spatial* error, we compare our solution with a strong stability preserving third order Runge Kutta (SSP RK3) time integrator to the solution Floren sent with the lowest timestep. The SSP RK3 solutions are accurate to 10 or more digits, as verified by refining the timestep by a factor of 2 and comparing solutions (for $N = 8$, $\Delta t = 10^{-6}$, for $N = 20$, $\Delta t = 10^{-7}$). The "exact" solution is defined to be the one Floren sent with $\tilde{N} = 24$ and smallest timestep $\Delta t = 10^{-6}$.

Fig. 3(a) shows the L^2 errors of 3 simulations: both ours and Floren's when $N = 8, \tilde{N} = 12$, and ours when $N = 20$. Our algorithm gives 3 digits of *absolute* accuracy in the fiber position for $N = 8$ and 4 digits for $N = 20$.

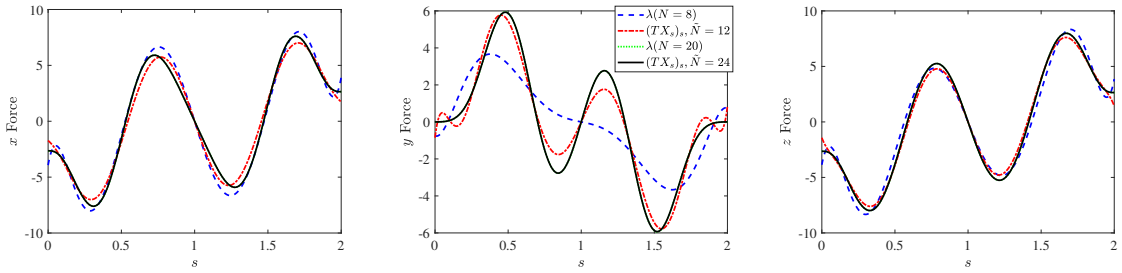
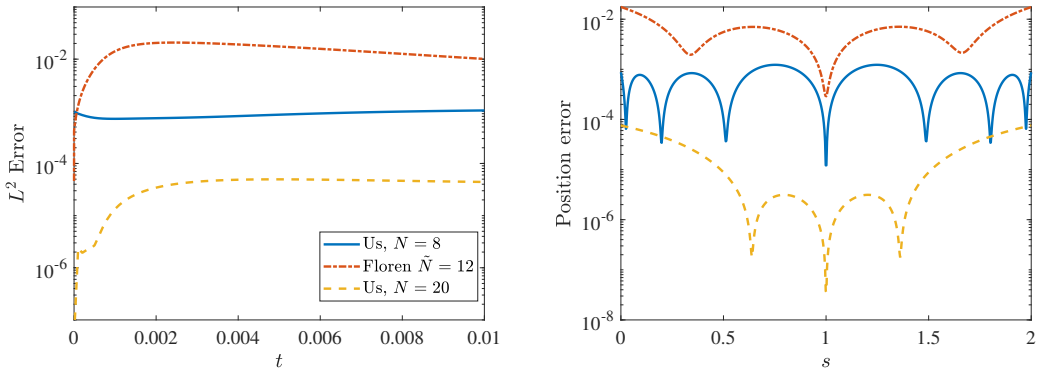


Figure 2: Measuring the convergence of $\boldsymbol{\lambda}$ against $(T\boldsymbol{X}_s)_s$. Each direction in the force (x, y, z) is a column from left to right. The dashed blue line shows $\boldsymbol{\lambda}$ for $N = 8$, which has an L^2 error of 2.81. For comparison, Floren's tensile force for $\tilde{N} = 12$ is shown as a dashed-dotted red line. It has an L^2 error of 1.22. $\boldsymbol{\lambda}$ for $N = 20$ is shown as a dotted green line, it has L^2 error 8.3×10^{-3} and is completely indistinguishable from the "exact" solution of Floren for $\tilde{N} = 24$.



(a) L^2 norm

(b) Error by position

Figure 3: Errors from 3D fiber simulation. (a) L^2 norm over time for $N = 8$ (solid blue), $N = 20$ (dashed yellow). The norm is defined in Eq. (26). The L^2 error in Floren's data for $\tilde{N} = 12$ is shown as a dashed-dotted red line. (b) The absolute error in the positions along the fiber (at $t = 0.01$) for $N = 8$, $N = 20$ (same color scheme as (a)). The spatial errors are isolated by using an SSP RK3 scheme for our method and using the lowest timestep provided for Floren's method.

Fig. 3(b) gives the absolute error at $t = 0.01$ as a function of the arclength position along the fiber for $N = 8, 20$ for both Floren's and our simulation. We observe about 3 digits of absolute accuracy for $N = 8$ and 4 digits for $N = 20$. In general, the relative accuracy is 10 times the absolute accuracy (i.e. 10 times worse), since the points move approximately $\mathcal{O}(0.1)$ throughout the simulation.

Note also that the absolute accuracy of Floren's solution for $\tilde{N} = 12$ is 1-2 digits. This translates to a relative accuracy of 0-1 digit. That is, there is an error of 10-100% in the motion of the points (especially near to the boundaries). **Thus the spatial error for our discretization is 10^{-3} for $N = 8$ points, but for Floren's it is 10^{-2}, or even larger near the boundary. Thus, at least for this spatial discretization, our algorithm is more accurate.**

2.2.1 Inextensibility

We consider how inextensible the fiber truly is for $t > 0$. In [4], a penalty parameter β is added to the line tension equation to preserve inextensibility. In this section, we compare this formulation to our formulation, where the inextensibility is constrained by the allowed fiber motion, Eq. (11).

To measure the inextensibility error, we take the fiber positions $\boldsymbol{X}$, upsample them to a fine grid,

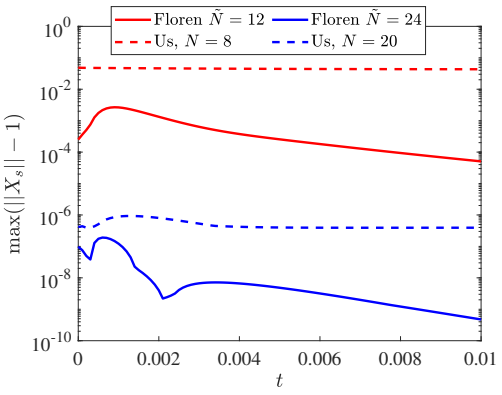


Figure 4: Error in enforcing the inextensibility constraint, $\max |\|\boldsymbol{X}_s\| - 1|$ for $N = 8$, $\tilde{N} = 12$ (red lines) and $N = 20$, $\tilde{N} = 24$ (blue lines). Solid lines show the results from Floren's code with the penalty parameter, while dashed lines show the results for our algorithm with constrained motion.

apply the spectral differentiation matrix on the fine grid to compute $\boldsymbol{X}_s$, and then measure the maximum difference of its norm from 1 along the fiber at every timestep. Fig. 4 shows the results for simulations with negligible temporal error. While Floren's code with the penalty parameter (solid lines) generally performs better, we are still able to enforce inextensibility to 2 digits for $N = 8$ and 6 digits for $N = 20$.

2.2.2 Boundary conditions

One of the features of our method is that boundary conditions are enforced in a weak sense with rectangular matrices (as opposed to a strong sense with square matrices). To verify this, we consider the fiber configuration $\boldsymbol{X}$ at time t. We perform differentiation on the type 1 grid and upsample the result to a type 2 grid with 1000 points (this is numerically preferable to upsampling and then differentiating). We consider the norms $\|\boldsymbol{X}_{ss}(s = 0)\|$ and $\|\boldsymbol{X}_{sss}(s = 0)\|$ of the upsampled derivative (these quantities are the same at $s = L$ by symmetry and should both be zero).

Fig. 5 shows the error in the boundary conditions as a function of time for $N = 8, 12, 16$ and 20. As in Section 2.2.1, the configurations are generated using RK3 temporal integration with $\Delta t = 10^{-6}$ to isolate the spatial error. For both $\boldsymbol{X}_{ss}$ and $\boldsymbol{X}_{sss}$, we observe spectral convergence.

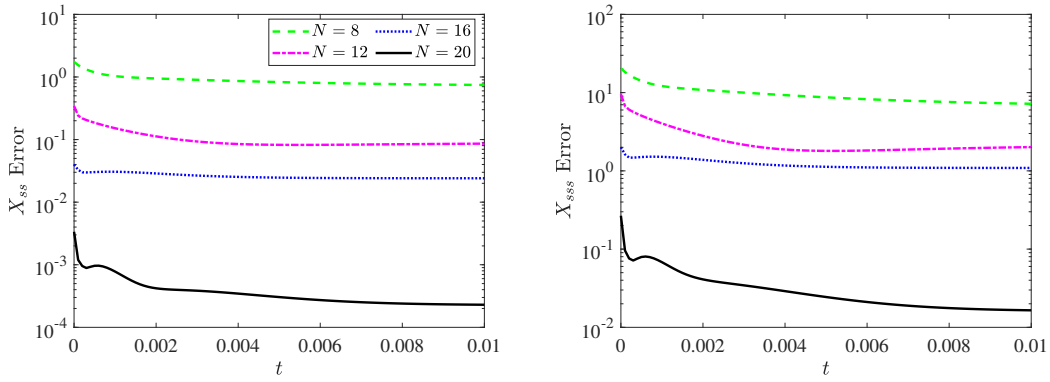
2.3 Temporal error

Having studied the spatial error in detail, we next study the temporal error and the requisite timestep to achieve a certain accuracy across both algorithms.

2.3.1 Floren's data

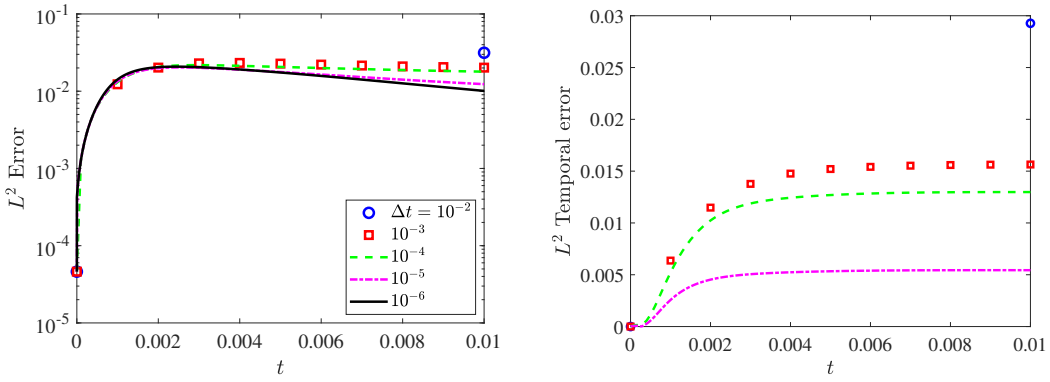
We begin with an analysis of Floren's data. Again, the "exact" solution is Floren's fiber evolution for $\tilde{N} = 24$, $\Delta t = 10^{-6}$ (the finest spatial and temporal discretization provided). Fig. 6(a) shows the L^2 errors for $\tilde{N} = 12$ and a variety of timesteps for Floren's algorithm. We observe the L^2 errors in Floren's algorithm for $\tilde{N} = 12$ are more or less saturated by the spatial error. Even for a timestep as small as 10^{-6}, there is still an L^2 error of $\mathcal{O}(10^{-2})$.

Given the large spatial error, it remains to be seen what timestep Floren has to use to get a temporal error less than or equal to the spatial error for $\tilde{N} = 12$. As shown in Fig. 6(b), a timestep of $\Delta t = 10^{-3}$ gives an L^2 difference of ≈ 0.015 from the evolution when $\Delta t = 10^{-6}$. Given that the spatial error is ≈ 0.01, we conclude that **Floren can use a $\Delta t = 10^{-3}$ for $\tilde{N} = 12$ and get a temporal error less than or equal to the spatial error**.



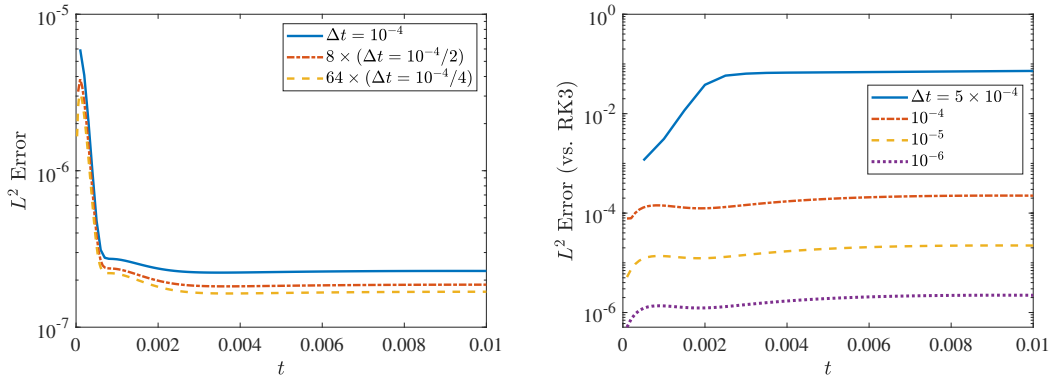
(a) $\|\boldsymbol{X}_{ss}(s=0)\|$ (b) $\|\boldsymbol{X}_{sss}(s=0)\|$

Figure 5: Spectral convergence of boundary conditions (a) $\|\boldsymbol{X}_{ss}(s=0)\|_2$, (b) $\|\boldsymbol{X}_{sss}(s=0)\|_2$, both of which should be 0. Shown are solutions for $N = 8$ (dashed green), $N = 12$ (dashed-dotted pink), $N = 16$ (dotted blue), and $N = 20$ (solid black), all simulated with RK3 so that the temporal integration is accurate to 10 digits. Values at the boundaries are obtained by differentiation on the N grid, then upsampling to a type 2 grid (that includes the endpoints).



(a) "Error" wrt $\tilde{N} = 24$ (b) "Error" wrt $\tilde{N} = 12$

Figure 6: (a) L^2 errors over time when comparing to Floren's "exact" solution with $\tilde{N} = 24$, $\Delta t = 10^{-6}$ to his solutions when $\tilde{N} = 12$. (b) L^2 *temporal* errors in Floren's algorithm for $\tilde{N} = 12$ using $\tilde{N} = 12$ $\Delta t = 10^{-6}$ as the "exact" solution. Timesteps are color coded as blue circles (10^{-2}), red squares (10^{-3}), dashed green lines (10^{-4}), dashed-dotted pink lines (10^{-5}), and solid black lines (10^{-6}).



(a) SSP RK3

(b) Forward Euler

Figure 7: Temporal accuracy for a fixed spatial discretization $N = 8$. (a) L^2 error for SSP RK3 on $N = 8$ (with respect to successive refinements in Δt by a factor of 2). The error for $\Delta t = 10^{-4}$ (blue line) is approximately 8 times the error for $\Delta t = 10^{-4}/2$ (dashed/dotted red line) and 64 times the error for $\Delta t = 10^{-4}/4$. The scheme is therefore third-order accurate. (b) Accuracy of the forward Euler method when compared to the "exact solution" from SSP RK3. $\Delta t = 5 \times 10^{-4}$ (blue line) is clearly inaccurate; it is likely too close to the stability limit. $\Delta t = 10^{-4}$ (dashed/dotted red), $\Delta t = 10^{-5}$ (dashed yellow), and $\Delta t = 10^{-6}$ (dotted purple) demonstrate first order convergence starting at an L^2 error of 10^{-4}. We conclude that the first-order method is sufficient since temporal error is less than spatial error for any $\Delta t \leq 2 \times 10^{-4}$.

2.3.2 Accuracy of temporal integration for fixed spatial discretization

We next focus on our algorithm with a fixed spatial discretization. For simplicity and ease of running multiple simulations, we choose $N = 8$. We first check the order of accuracy of the SSP RK3 scheme

$$\boldsymbol{X}^{n+1} = \boldsymbol{X}^n + \Delta t \left(\frac{1}{6}\boldsymbol{k}_1 + \frac{1}{6}\boldsymbol{k}_2 + \frac{2}{3}\boldsymbol{k}_3 \right) \tag{27}$$

$$\boldsymbol{k}_1 = \boldsymbol{u}\left(\boldsymbol{X}^n\right) \qquad \boldsymbol{k}_2 = \boldsymbol{u}\left(\boldsymbol{X}^n + \Delta t \boldsymbol{k}_1\right) \qquad \boldsymbol{k}_3 = \boldsymbol{u}\left(\boldsymbol{X}^n + \frac{\Delta t}{4}\boldsymbol{k}_1 + \frac{\Delta t}{4}\boldsymbol{k}_2\right). \tag{28}$$

Importantly, for this test we will use successive refinements to measure convergence (rather than comparing to Floren's results). As shown in Fig. 7(a), SSP RK3 is a third order accurate scheme, as the errors scale as Δt^3. Importantly, the temporal error for $\Delta t = 10^{-4}$ using RK3 is 10^{-5}, which is 2 orders of magnitude less than the spatial error of 10^{-3} (see Fig. 3(a)). Furthermore (this data is not shown on the plot), using $\Delta t = 2 \times 10^{-4}$ gives a maximum L^2 error of 9×10^{-5} and $\Delta t = 4 \times 10^{-4}$ gives a maximum L^2 error of 2×10^{-2} in RK3.

Next, we take the solution from SSP RK3 with $\Delta t = 10^{-6}$ and treat this (which has converged to 12 digits) as an exact solution for $N = 8$. We seek the error from forward Euler. The timestep $\Delta t = 10^{-3}$ is unstable. As shown in Fig. 7(b), using a timestep half of this unstable Δt yields a large error. However, for $\Delta t = 10^{-4}$ and lower, we see the expected first order convergence. Furthermore, 4 digits of accuracy are observed for $\Delta t = 10^{-4}$. Finally (this data is not shown on the plot), using $\Delta t = 2 \times 10^{-4}$ gives a maximum L^2 error of 6×10^{-4} and $\Delta t = 4 \times 10^{-4}$ gives a maximum L^2 error of 4×10^{-2} in forward Euler. **Therefore, the conclusion is that there is no reason to use RK3 in this scheme. While the temporal accuracy is slightly improved in RK3, the 10^{-3} spatial error still dominates for every $\Delta t \leq 2 \times 10^{-4}$, regardless of whether forward Euler or RK3 is used. Put another way, for any Δt for which the temporal error is less than or equal to the spatial error, forward Euler is sufficient.**

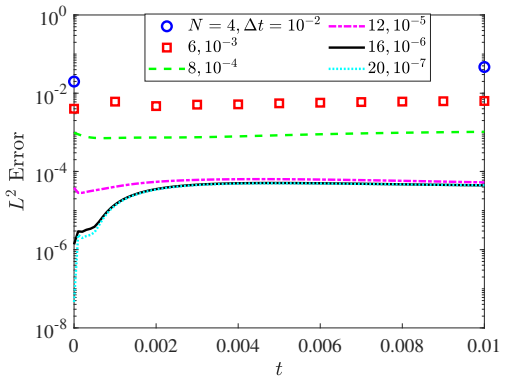


Figure 8: Overall spatio-temporal accuracy of our scheme. Timestep is the first stable power of 10 using forward Euler integration for a given spatial discretization. Recall that the relative error in displacement is approximately 10 times the L^2 error.

2.3.3 Spatio-temporal accuracy of our scheme

Since we cannot choose any timestep, we seek some kind of comparison between our method and Floren's. That is, a way to compare to Fig. 6. We therefore set up the following experiment: consider spatial discretizations from $N = 6, 8, \ldots 20$. We run the *forward Euler* temporal integrator with *the first stable timestep that is a power of 10*. The accuracy of this temporal integration is verfied by ensuring the maximum L^2 error in time is of the same order of magnitude with a Δt that is half the relevant power of 10.

For this section, we return to using Floren's maximally refined evolution as the "exact" solution (i.e. the same "exact" solution as in Fig. 6(a)).

Shown in Fig. 8 is the overall accuracy of our scheme under spatio-temporal refinement. **In our case, a timestep of 10^{-4} is stable up to $N = 8$ points, and we get an accuracy of 10^{-3}. This is an order of magnitude better than Floren, whose errors are spatially dominated for this Δt. Furthermore, for $\Delta t = 10^{-3}$, (when Floren's spatial and temporal errors are comparable) we still get higher accuracy with $N = 6$ than Floren with $\tilde{N} = 12$!**

2.4 Implicit timestepping

2.4.1 Backward Euler

We recall that we can define $\boldsymbol{f}^E = \boldsymbol{L}\boldsymbol{X}$ on the type 1 grid. With this in mind, we can write the equations for a backward Euler update as

$$\boldsymbol{X}^{n+1} = \boldsymbol{X}^n + \Delta t \boldsymbol{K}\boldsymbol{\alpha} \tag{29}$$

$$\boldsymbol{K}\boldsymbol{\alpha} = \boldsymbol{M}^n \left(\boldsymbol{\lambda} + \left(\boldsymbol{f}^E \right)^{n+1} \right) \tag{30}$$

$$\boldsymbol{K}^* \boldsymbol{\lambda} = \begin{pmatrix} \boldsymbol{0} \\ -\boldsymbol{I}^* \left(\boldsymbol{f}^E \right)^{n+1} \end{pmatrix} \tag{31}$$

Eq. (29) can be used to rewrite these equations in block form as

$$\boldsymbol{A}\boldsymbol{y} = \begin{pmatrix} -\boldsymbol{M} & \boldsymbol{K} - \Delta t \boldsymbol{M}\boldsymbol{L}\boldsymbol{K} \\ \boldsymbol{K}^* & \begin{pmatrix} \boldsymbol{0} \\ \Delta t \boldsymbol{I}^* \boldsymbol{L}\boldsymbol{K} \end{pmatrix} \end{pmatrix}^n \begin{pmatrix} \boldsymbol{\lambda} \\ \boldsymbol{\alpha} \end{pmatrix} = \begin{pmatrix} \boldsymbol{M}\boldsymbol{L}\boldsymbol{X}^n \\ \begin{pmatrix} \boldsymbol{0} \\ -\boldsymbol{I}^* \boldsymbol{L}\boldsymbol{X}^n \end{pmatrix} \end{pmatrix} \tag{32}$$

N	Spatial Er	FE Δt	FE Temp Er	BE Δt	BE Temp Er	CN Δt	CN Temp Er
4	2.23e-2	5×10^{-3}	1.35e-2	10^{-2}	1.98e-2	10^{-2}	1.28e-2
8	1.04e-3	2×10^{-4}	5.60e-4	10^{-3}	2.29e-3	2×10^{-3}	1.58e-3
12	3.83e-5	10^{-5}	2.23e-5	10^{-5}	2.47e-5	2×10^{-4}	2.69e-5
16	2.32e-6	10^{-6}	2.23e-6	10^{-6}	2.48e-6	5×10^{-5}	1.72e-6

Table 1: Comparing the required timestep to match the spatial error (measured against $N = 20$ RK3) with the temporal error (measured against the RK3 spatial solution for that N) using forward Euler (FE), backward Euler (BE), and Crank-Nicolson (CN). All errors are maximum L^2 errors over time.

The matrices $\boldsymbol{M}$ and $\boldsymbol{K}$ depend on $\boldsymbol{X}$, and these are treated explicitly. The bending force $\boldsymbol{f}^E = \boldsymbol{L}\boldsymbol{X}$ is always treated implicitly. Note that this results in an $(5N+1)^2$ linear system, which is exactly the same size as in the explicit case, Eq. (16).

Now, we saw in Section 2.2 that the L^2 spatial error of our discretization is $\approx 10^{-3}$. However, in Fig. 7(b), we saw that temporal error near the boundary of the stability region is 10^{-4}. If we used a larger timestep than $\Delta t = 1 - 2 \times 10^4$, we saw ballooning errors. In this section, we ask whether we can "fix" this with implicit timestepping.

Table 1 shows the "savings" when using backward Euler for different N. All of the errors are the maximal L^2 error over time. The first column is the spatial error, which is the difference between an RK3 simulation (accurate to 10 digits) of the given N and an RK3 simulation of $N = 20$ (which we take as the exact solution). Our goal is for the temporal error, which is the difference between a given (N, Δt) pair and the "exact" RK3 solution for the same N, to be on the same order as the spatial error. Table 1 shows what timestep is necessary for this.

Considering $N = 4$ and $N = 8$, we see that the forward Euler method is limited by stability and we can get a savings by switching to backward Euler. For $N > 8$, the spatial discretization is converging so rapidly that we are limited by accuracy; and forward Euler and backward Euler get the same result. So it appears that backward Euler is not particularly advantageous (at least for this very smooth problem), except for $N = 8$ when we get a factor of 5.

2.4.2 Crank-Nicolson

We combine Crank-Nicolson with a linear multi-step method to obtain a second-order scheme with one solver per timestep. Specifically, we set

$$\boldsymbol{X}^{n+1/2,*} = \frac{3}{2}\boldsymbol{X}^n - \frac{1}{2}\boldsymbol{X}^{n-1}. \tag{33}$$

We then solve

$$\boldsymbol{A}\boldsymbol{y} = \begin{pmatrix} -\boldsymbol{M} & \boldsymbol{K} - \frac{1}{2}\Delta t \boldsymbol{M}\boldsymbol{L}\boldsymbol{K} \\ \boldsymbol{K}^* & \begin{pmatrix} \boldsymbol{0} \\ \frac{1}{2}\Delta t \boldsymbol{I}^*\boldsymbol{L}\boldsymbol{K} \end{pmatrix} \end{pmatrix}^{n+1/2,*} \begin{pmatrix} \boldsymbol{\lambda} \\ \boldsymbol{\alpha} \end{pmatrix} = \begin{pmatrix} \boldsymbol{M}\boldsymbol{L}\boldsymbol{X}^n \\ \begin{pmatrix} \boldsymbol{0} \\ -\boldsymbol{I}^*\boldsymbol{L}\boldsymbol{X}^n \end{pmatrix} \end{pmatrix} \tag{34}$$

Eq. (34) is now the system to be solved at every timestep, with the operators $\boldsymbol{M}$ and $\boldsymbol{K}$ evaluated using Eq. (33). Because we saw that backward Euler gave little to no savings over forward Euler, our goal in this section is to determine whether a *second-order* implicit scheme (which requires one solve per timestep) gives a savings over an explicit one.

Fig. 9 shows that the Crank-Nicolson scheme is second-order accurate and gives a much lower temporal error for $\Delta t \leq 10^{-3}$ than backward Euler for $N = 8$. Furthermore, we see in Table 1 that the Crank-Nicolson temporal update can give a temporal error comparable to the spatial error for much larger timesteps than forward/backward Euler. In particular, for $N = 4, 8, 12$, and 16, we see a savings from Crank-Nicolson (over forward Euler) that is a factor of 1, 10, 20, and 50. This comes at no extra cost (other than a possibly dangerous extrapolation step), since we are doing a $(5N + 1)^2$ least squares solve once per timestep in all cases.

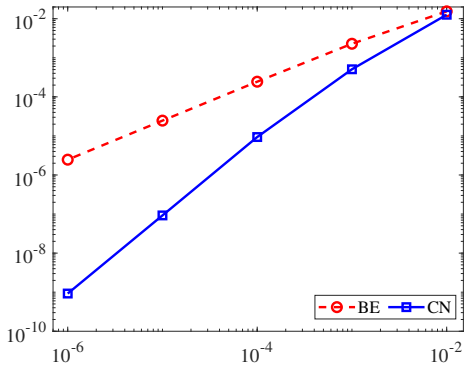


Figure 9: Comparing the temporal error from backward Euler (red circles) and Crank-Nicolson (blue squares) for $N = 8$. Shown is the maximum L^2 error (wrt the RK3 explicit solution accurate to 10 digits) over time for different Δt. First order convergence is observed for backward Euler, and second-order convergence is observed for Crank-Nicolson. Table 1 has more details on the savings that can be obtained for different N.

References

[1] Jared L Aurentz and Lloyd N Trefethen. Block operators and spectral discretizations. *SIAM Review*, 59(2):423–446, 2017.

[2] Tobin A Driscoll and Nicholas Hale. Rectangular spectral collocation. *IMA Journal of Numerical Analysis*, 36(1):108–132, 2015.

[3] Ehssan Nazockdast, Abtin Rahimian, Denis Zorin, and Michael Shelley. A fast platform for simulating semi-flexible fiber suspensions applied to cell mechanics. *Journal of Computational Physics*, 329:173–209, 2017.

[4] Anna-Karin Tornberg and Michael J Shelley. Simulating the dynamics and interactions of flexible fibers in stokes flows. *Journal of Computational Physics*, 196(1):8–40, 2004.