

OPEN ACCESS

Collective Langevin dynamics of flexible cytoskeletal fibers

To cite this article: Francois Nedelec and Dietrich Foethke 2007 *New J. Phys.* **9** 427

View the [article online](#) for updates and enhancements.

You may also like

- [Shaping up synthetic cells](#)
Yuval Mulla, Anders Aufderhorst-Roberts and Gijssje H Koenderink
- [Getting around the cell: physical transport in the intracellular world](#)
Saurabh S Mogre, Aidan I Brown and Elena F Koslover
- [Changes in cytoskeletal dynamics and nonlinear rheology with metastatic ability in cancer cell lines](#)
Mark F Coughlin and Jeffrey J Fredberg

Collective Langevin dynamics of flexible cytoskeletal fibers

Francois Nedelec¹ and Dietrich Foethke

European Molecular Biology Laboratory, 69117 Heidelberg, Germany

E-mail: nedelec@embl.de

New Journal of Physics **9** (2007) 427

Received 14 June 2007

Published 30 November 2007

Online at <http://www.njp.org/>

doi:10.1088/1367-2630/9/11/427

Abstract. We develop a numerical method to simulate mechanical objects in a viscous medium at a scale where inertia is negligible. Fibers, spheres and other voluminous objects are represented with points. Different types of connections are used to link the points together and in this way create composite mechanical structures. The motion of such structures in a Brownian environment is described by a first-order multivariate Langevin equation. We propose a computationally efficient method to integrate the equation, and illustrate the applicability of the method to cytoskeletal modeling with several examples.

¹ Author to whom any correspondence should be addressed.

Contents

1. Introduction	2
2. Objects	4
3. Constrained Langevin dynamics	6
4. Numerical integration	7
5. Linear set of points (fiber)	8
5.1. Bending elasticity	9
5.2. Mobility	10
5.3. Projector associated with the constraints	10
5.4. Brownian motion	11
6. Spherical set of points (sphere)	12
6.1. Mobility and Brownian motion	12
7. Non-deformable set of points (solid)	13
7.1. Mobility and constrained motion	13
8. Interactions between objects	13
8.1. Connecting an object to a fixed position	15
8.2. Connecting two objects	15
8.3. Confinement in a convex shape	15
8.4. Connecting two objects at a given distance	15
8.5. Interpolation of forces	16
9. Numerical stability and performance	17
10. Other elements of a cytoskeletal simulation	20
10.1. Modeling molecular motors	20
11. Conclusion	22
Acknowledgments	23
References	23

1. Introduction

The internal architecture of living cells relies largely on microscopic fibers, which form the cytoskeleton with their associated proteins. These fibers have remarkable mechanical properties. Microtubules and actin filaments for instance have persistence lengths of ~ 5 μm and 20 μm , respectively, and can sustain pico-Newtons of force without breaking [1]. Yet these fibers can also be broken down quickly, because they are formed by the non-covalent assembly of protein monomers. Filament ends can grow or shrink, or even alternate between those two states in a remarkable process called dynamic instability [2, 3]. Structurally, the monomers in microtubules and actin filaments assemble head to tail in a regular manner. On the resulting polar lattices, mechano-enzymes called molecular motors (for example kinesin on microtubules or myosin on actin-filaments) use chemical energy to move directionally [1] or to organize the filaments in space [4]. Furthermore, specific enzymes control the filaments by regulating nucleation, assembly/disassembly or even by severing the filaments.

The cytoskeleton is involved in multiple cellular processes such as cytokinesis, motility, polarization and mitosis. These functions are accomplished by many filaments working together.

In this way, a set of dynamic or short-lived filaments may form a stable larger assembly, as exemplified by the mitotic spindle [4]. Many of the enzymes involved in the assembly of these structures are part of multi-functional entities [5–7]. For example, motors form oligomers that can actively connect filaments together [4]; motors may be able to disassemble filaments [6]; nucleation can be controlled such that it occurs on existing filaments [8, 9]; crosslinkers may be polarity-specific [10] and motors are sometimes linked to proteins that track the tips of growing microtubules [7, 11–13]. Generally speaking, modularity allows the cytoskeleton to be reprogrammed, for example, at different stages of the cell cycle. It allows cells to reuse the same functional elements to achieve different tasks and multiplies the number of way in which the organization of fibers can be regulated. This modularity is certainly a consequence of the combinatorial exploration operating during natural selection [14]. In any case, the cytoskeleton in addition to fibers contains a kit of activities which can be combined in many ways.

Biological systems are hard to understand, and theory is necessary to approach the non-intuitive aspects [15]. It is notable that many models in the cytoskeleton field often include the same basic elements (for a recent review on this subject, see [16]). This reflects the inherent modularity of the biological design illustrated briefly in the previous paragraph, and also affects the modeling approach. It implies that it is worthwhile to build a computer simulation to model a few basic elements, if these elements can be combined freely to rapidly model diverse situations. In practice, the elements of the simulation (e.g. a model of kinesin, or a model of a severing enzyme) can even be implemented, tested and benchmarked by different teams of experts for each aspect of the system. Sharing computer code in this way can in fact be a practical means to combine the efforts of the community.

Writing a cytoskeletal simulation is likely to be a collective task also because it is a demanding project, involving multiple aspects: (a) chemical reactions that occur inside cells, (b) transport along fibers, for example, the motion of molecular motors, (c) assembly dynamics of cytoskeletal fibers and (d) motion and deformation of fibers. Fortunately, numerous algorithms are available for certain of these aspects, in particular for reaction–diffusion (see, [17, 18]). Transport along fibers can be modeled with advection equations, or with more details of the motion of the motors [19]. The assembly dynamics of fibers has been the subject of much research and cannot be reviewed here (see [16]). The deformation of the fibers is a classical mechanical problem (see for example [20, 21]). However, the scale of living cells is associated with many specific features. In particular, Brownian motion plays a fundamental role, inertia is negligible [22] and the fibers are dynamic: they can lengthen or shorten by self-assembly. As a consequence, the physics of biological fibers is fundamentally distinct from other mechanical systems. In brief, public or commercial codes are not adapted to simulate the cytoskeleton.

The purpose of this paper is to describe a method to calculate the mechanics of an ensemble of connected fibers and other objects, which is the basis of a cytoskeletal simulation such as *cytosim*. The physics of such a system is described by a Langevin equation (for an introduction, see [23]) that recreates the Brownian motion of the fibers and includes bending elasticity, fiber–fiber interactions and external force fields. Following earlier work [24, 25], we use constraints in order to maintain the length of the fibers. This is an alternative to methods in which potentials are used to represent the longitudinal stiffness of fibers. We extend this approach by introducing an implicit integration scheme. Our method was first used to simulate the effects of motor complexes on two radial arrays of microtubules (asters) [26], and more recently, the assembly of anti-parallel microtubule arrays in *Schizosaccharomyces pombe* [7] and the positioning of the spindle in the *Caenorhabditis elegans* embryo [27]. A major aim of

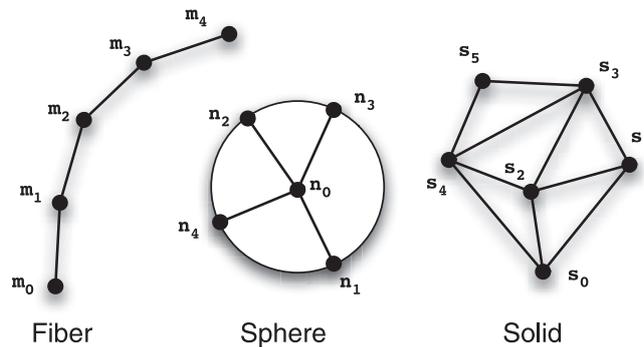


Figure 1. Elementary objects: all objects in the simulation are described by points. The points can move in the viscous medium, but the relative distances between certain points are conserved (lines). Left: a *fiber* is modeled as an equidistributed string of points. Center: a *sphere* is composed of a central point and peripheral points, located at a distance r from the center. The peripheral points can move on the surface, as if they were in a viscous membrane. Right: a *solid* is a set of points that behaves like a solid body. Its shape and size are constants.

these simulations was to reconstitute the system's operation *in silico*, from established physical principles. This offers two major advantages: (i) the assumptions of the model are well defined and can always be modified and (ii) any property of the system can be measured easily. This facilitates further investigations. For example, we could systematically simplify the model in order to identify a minimal set of working properties [7]. In addition, we could identify the parameter range under which the system can operate [27]. However, for these results to be valid, the system's operation needs to be reproduced correctly at the first place! To maximize the chances of success, it is desirable to reconstitute the mechanics in a physically sensible and accurate way. One may otherwise derive conclusions which do not apply to the real system.

In this paper, we focus on the mechanical aspects of the fibers, and explore the numerical resolution of the associated equations. We first describe objects that in addition to fibers are useful for simulating different cellular skeletons. We then present the equation of motion and discuss its numerical integration. We examine the numerical stability of the resulting method and discuss how it affects the simulation speed. Finally, we discuss how other aspects of the cytoskeleton can be added to extend the mechanical calculation.

2. Objects

More accurate mechanics can be achieved if we introduce two new objects in addition to *fibers*: spherical sets of points (*spheres*) and non-deformable sets of points (*solids*). These objects are also described with points but have different morphologies (see figure 1). The mechanical properties are also distinct. While *fibers* may bend, the *solids* do not deform. The *spheres* can represent spherical viscous membranes such as vesicles. Any number of objects can be combined in various ways to build complex cytoskeletons. For example, to simulate interacting microtubule asters [26], *fibers* were positioned around a *solid* using static links (see figure 2A).

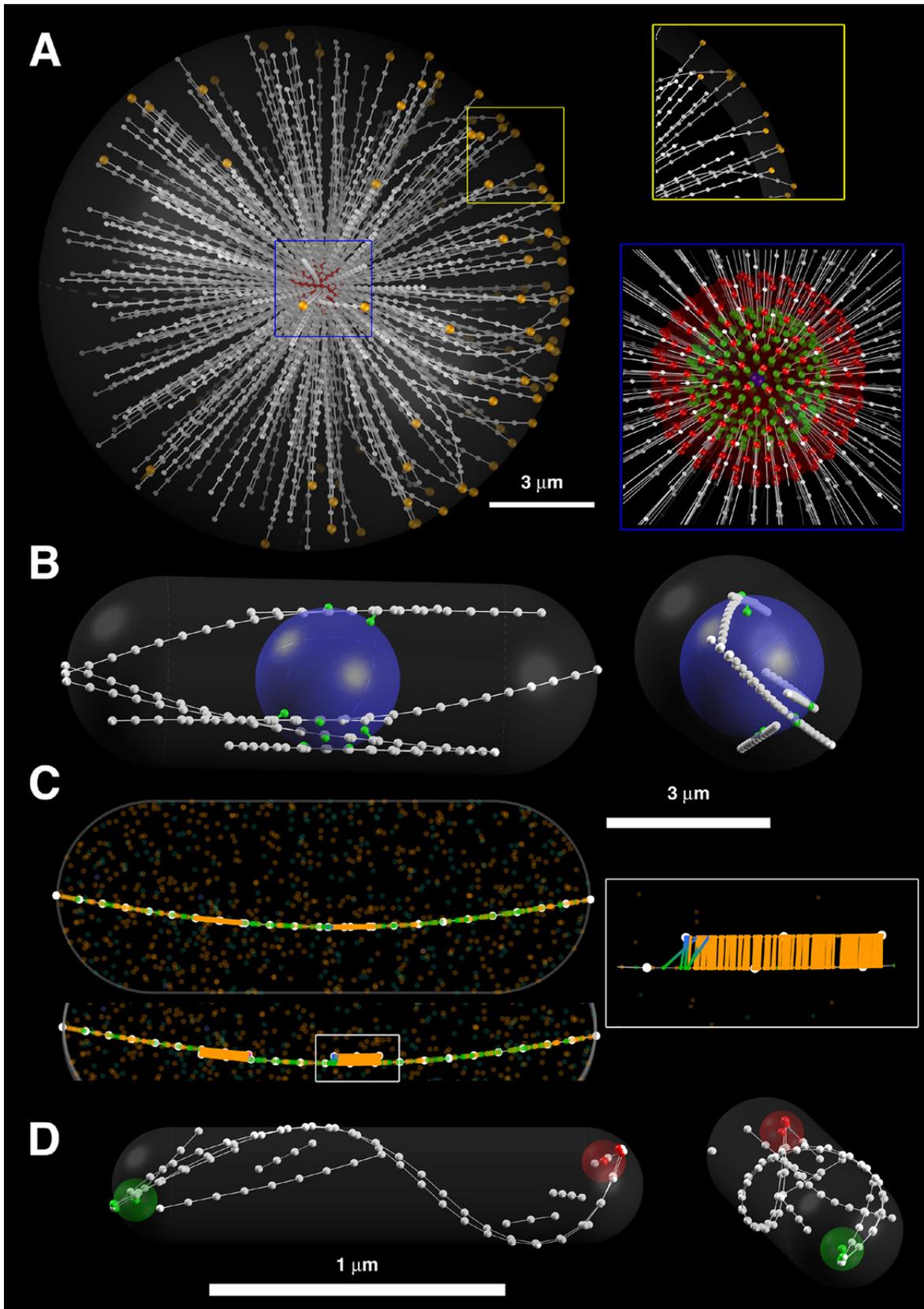


Figure 2. For caption see following page.

Figure 2. Some problems studied with *cytosim*. In all the images, *fibers* are indicated in white, along with their model-points. (A) An aster is constructed by assembling fibers radially around a solid [26, 27]. Right top: interactions of microtubules with the cell cortex. Right bottom: the *solid* is made of a central point (blue) surrounded by two concentric layers of peripheral points (green and red). Only the central point is associated with a viscous drag ($a_i > 0$). The other points are used to attach fibers: the minus-end to one green point, and a distal position on the fiber to one red point. Using a similar simulation with two asters linked by a *solid* spindle, we proposed an original model describing the 3D motions of the spindle in the first cell division of the *C. elegans* embryo [27]. (B) Microtubules in interphase fission yeast and the nucleus, represented by a *sphere* (blue/green). This can be used to study the role of mechanics in regulating the dynamics and organization of microtubules. (C) Self-assembly of interphase microtubules arrays in fission yeast. The simulation contains no steric interaction between the fibers, and they overlap freely. In the display, however, the fibers are shifted in order to visualize the bridging complexes (bottom and right). Using this simulation, we could identify a minimal ‘recipe’ to make stable bundles from dynamic microtubules. This recipe describes how cross-linking, nucleating and motor activities can be associated to obtain the result observed *in vivo*. (D) Self-segregation of plasmids in prokaryotes. Actin-like filaments are simulated, together with two *solids*, representing the plasmids [28]. The efficiency of the segregation is recapitulated in the simulation, and can therefore be analyzed.

The solid represented in this case the organelle (called the centrosome) which in the cell generates microtubules in a radial fashion. *In vivo* as well as in the simulation, the resulting structure is radially symmetric, and the fibers have their ends mechanically joined together. Two such asters were further connected by another *solid*, to model the positioning of the mitotic spindle in *C. elegans* [27]. In this case, the additional solid represented the pole-to-pole mechanical connection achieved by the mitotic spindle. To simulate nuclear positioning in *S. pombe*, *fibers* (microtubules) were attached to a *sphere*, and the ensemble was confined in a cylindrical volume (see figure 2B). The *fibers* and the *sphere* represented microtubules and the cell nucleus, which are attached also in the real cell. To model the formation of anti-parallel microtubule arrays in *S. pombe* [7], *fibers* were connected by motors and other crosslinkers (see figure 2C). Using *fibers* and *solids*, it is also possible to model the segregation of parM plasmids in *Escherichia coli* (see figure 2D), a process which depends on actin-like filaments [28]. The objects can naturally be combined in many more ways than illustrated here. This enables diverse cellular mechanics to be reproduced, and consequently widens the application scope of the method. This freedom is intimately linked to the structure of the master equation that will be examined below, and to the way it is integrated numerically.

3. Constrained Langevin dynamics

In the simulation, fibers and other objects are described by points. The coordinates of the points are collected in a vector \mathbf{x} of size Nd , for a system of N points in dimension d . Following

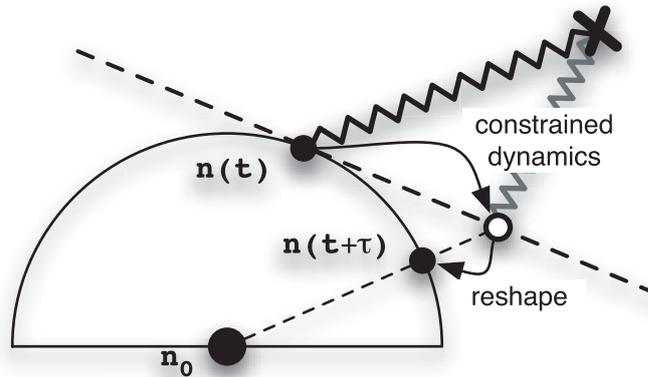


Figure 3. Dynamics with constraints. The principle of the algorithm is illustrated here for a point n_i constrained to stay at a fixed distance from n_0 . The point is first moved on the tangent to the circle (this is the plane associated with the constraint) using an implicit integration scheme. The constraint is then re-established exactly by projecting on the sphere. We call this last operation ‘reshaping an object’.

Langevin (for a simple introduction, see [23]) the equation of motion reads:

$$d\mathbf{x} = \mu F(\mathbf{x}, t) dt + dB(t) \quad (1)$$

$F(\mathbf{x}, t)$ of size Nd contains the forces acting on the points at time t . It includes object-specific forces such as bending elasticity, and all the links between different objects. $dB(t)$ of size Nd summarizes the random molecular collisions leading to Brownian motions; it is a stochastic non-differentiable function of time. The matrix μ contains the mobility coefficients of the object-points, which will be defined later for each object.

In addition, certain distances between points inside the objects ($|a_i - a_j| = \lambda_{ij}$) must be conserved during the motion. To satisfy these constraints, we perform a step of the dynamics in a subspace tangent to the manifold defined by the constraints, and project the result on the manifold. The procedure can be explained simply for a point n constrained to move at a distance r from a fixed position n_0 (see figure 3). To calculate the motion of n , we first write its dynamics in the plane tangent to the sphere at the current position (this is the plane allowed by the constraint $|n - n_0| = r$). The restricted dynamics is integrated implicitly, and the result projected on the sphere to restore the constraint exactly. This approach can be generalized as described next.

4. Numerical integration

From an initial configuration, the system is calculated by discrete time steps τ (see [29] for a general discussion on numerical integration). To calculate $\mathbf{x}_{t+\tau}$ from \mathbf{x}_t , the equation (1) is integrated implicitly. We will discuss the advantages of using an implicit rather than an explicit integration in section 9, and concentrate here on the practical issues. For an implicit integration, we need to express $F(\mathbf{x}, t)$ linearly as $A_t \mathbf{x} + G_t$, where the square matrix A_t contains the stiffness coefficients associated with the interactions, and the vector G_t contains the constant

forces. This linearization is obtained by summing over all the interactions present at time t (see figure 6). In our simulations, many of the interactions were modeled as harmonic potentials for simplicity, and are therefore already linear. Non-linear interactions simply need to be linearized at this point. In particular, the linearization of the constraints leads to an orthogonal projection $P(\mathbf{x})$, which will be defined later for each object. To obtain a finite difference scheme for the interval $[t, t + \tau]$, P and A are used at time t , but \mathbf{x} is used at $t + \tau$ (using $x_{t+\tau}$ instead of x_t is the basis of implicit integration):

$$\mathbf{x}_{t+\tau} - \mathbf{x}_t = P_t [\tau \mu (A_t \mathbf{x}_{t+\tau} + G_t) + \delta B_t],$$

leading to a system of linear equations:

$$[I - \tau P_t \mu A_t] (\mathbf{x}_{t+\tau} - \mathbf{x}_t) = P_t [\tau \mu (A_t \mathbf{x}_t + G_t) + \delta B_t], \quad (2)$$

where $A_t = A(t)$, $G_t = G(t)$, $P_t = P(x_t)$. The ‘simulated Brownian’ $\delta B_t = \int_t^{t+\tau} dB$ is a vector $\{\beta_i \theta_{t,i}\}_{i \in [1, Nd]}$, where $\theta_{t,i} \sim N(0, 1)$ are Nd independent normally distributed numbers (derived from uniformly distributed pseudo-random numbers [29]). The factors $\beta_i \sim \tau^{1/2}$ represent the magnitude of the Brownian motion during a lapse of time τ . We will see later how they are obtained by calibrating the diffusive motion for the objects. The equation can be solved to obtain $\mathbf{x}_{t+\tau}$, since both the right-hand side and the matrix $[I - \tau P_t \mu A_t]$ are known. It would be inefficient to invert the matrix, because the system is sparse (it only has few non-zero coefficients). This is true of matrix A_t , as long as objects are only connected to few others. This is also true of P_t which is block-diagonal: it has one block for each object on the diagonal, but the rest of the coefficients are null. This is because the constraints never involve points from different objects, and the projection can thus be done independently for each object. In this situation, it is advantageous to solve the linear system using an iterative method [29]. Different iterative solvers are adapted to different matrices. Because $P_t A_t$ is non-symmetric, we have used the biconjugate gradient stabilized (<http://www.netlib.org>). This method iteratively converges toward the solution of the linear system, and can be stopped when the difference with the exact solution is below a certain threshold. We set this threshold to $\psi \min(\beta_i)$, with $\psi = 1/10$. In this way, the numerical error on \mathbf{x} remains below 10% of the Brownian motion, and the approximate solution of (2) is practically indistinguishable from the real one. In practice, it is wise to systematically vary ψ and τ for each application to check the convergence of the method. It is easy to verify, for example, that more stringent values of ψ produce the same results.

Finally, since equation (2) is obtained by linearization, an additional correction is necessary to re-establish the constraints. The result of equation (2) is projected back on the manifold associated with the constraints [26]. This introduces corrections which are second-order in τ . In the following sections, we will call this procedure ‘reshaping’ the objects. We now survey how *fibers*, *spheres* and *solids* are represented in space, their mobility coefficients, projection operators and ‘reshaping’ procedure. The interactions between objects (which contribute to A_t and G_t) will be described subsequently.

5. Linear set of points (fiber)

Fibers are modeled as infinitely thin linear objects behaving like elastic, non-extensible rods [26]. Each fiber is represented by $p + 1$ equidistant model-points m_i , for $i \in [0, p]$, separated by a distance L/p . A fiber is polar: m_0 is the minus-end and m_p the plus-end. The

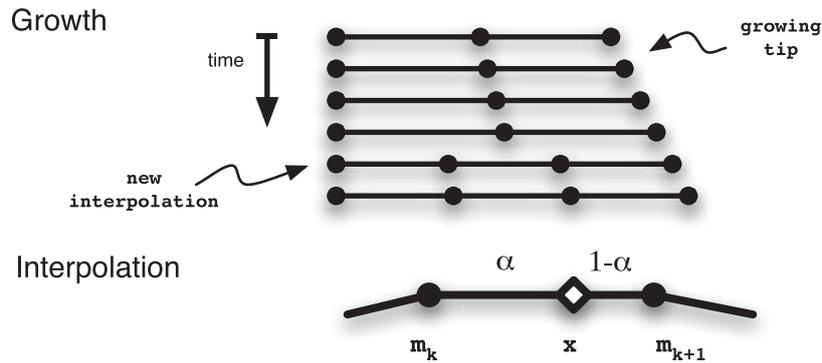


Figure 4. Dynamic fibers. Top: the model-points of a *fiber* are updated when the tips grow, but they are always equally distributed over the fiber. Points are added or removed as necessary to ensure an optimal coverage (see section 5). Bottom: an intermediate position x along the fiber is interpolated from the model-points located on each side: $x = (1 - \alpha)m_k + \alpha m_{k+1}$ (see section 5).

number of segments p is adjusted as a function of the total length L of the fiber. Points are added or removed, in order to always minimize $|\rho - L/p|$, for each fiber as it grows or shrinks (see figure 4). The desired segment length ρ is a parameter affecting the precision of the simulation. To set ρ , one may run a representative case with various values (for microtubules, $\rho < 0.5 \mu\text{m}$ is usually appropriate).

It is often necessary to interpolate between the model-points, when, for example, calculating the position x of a molecule attached to the fiber. If m_k and m_{k+1} are the model-points on each side of x , we use $x = (1 - \alpha)m_k + \alpha m_{k+1}$. The interpolation coefficient $\alpha \in [0, 1]$ is calculated from the known relative positions of the three points along the fiber: $\alpha = |m_k x| / |m_k m_{k+1}|$. The model-points are themselves updated using this interpolation procedure at every time-step if the length of the fiber has changed (see figure 4).

5.1. Bending elasticity

Fibers can bend under external forces and resist these forces elastically. The standard formula for bending elasticity [20] can be applied to strings of points. For any set of three consecutive points $m_k, k \in \{i - 1; i; i + 1\}$, we approximate it linearly as a triplet of forces $\{-F; 2F; -F\}$. Each triplet corresponds to the torque generated between two consecutive segments (see figure 5). Furthermore, we have $F = \alpha(m_{i-1} - 2m_i + m_{i+1})$, with $\alpha = \kappa(p/L)^3$, where κ is the bending modulus of the fiber, and L/n the length of each segment. The result was verified by comparing the buckling threshold in the simulation with Euler's formula $\pi^2 \kappa / L^2$. The procedure is appropriate if ρ is such that the angles between consecutive segments remain small during the simulation (not shown). Physically, the forces are isotropic, i.e. they can be written as a reduced matrix of size $p \times p$ (and not $pd \times pd$), obtained by adding several times the 3×3 matrix $E = -(1, -2, 1) \otimes (1, -2, 1)$ (\otimes is the tensor product). The final result is simple because points are distributed regularly over the length of the fiber (see figure 5).

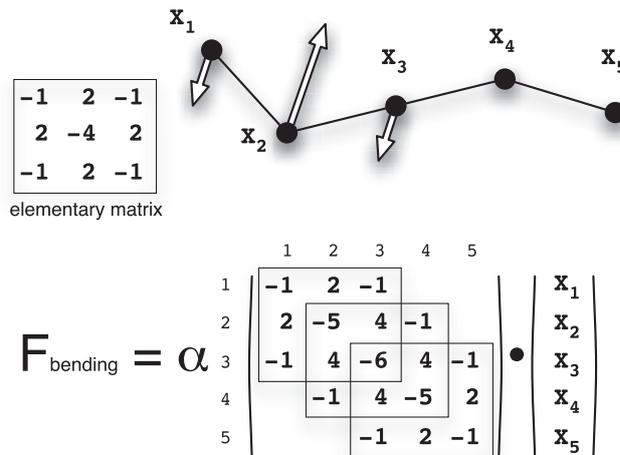


Figure 5. Matrix elements associated with bending elasticity. The stiffness matrix A_t contains the bending elasticity of fibers. The contributions are obtained by adding a 3×3 elementary matrix for each consecutive triplets of points (see section 5.1). The sum of all rows and columns is zero, since the matrix should only generate an internal torque. The forces associated with the first triplet (points x_1 , x_2 and x_3) are depicted. The resulting matrix for 5 points is also shown, and the generalization is straightforward. For any fiber, the result is a symmetric banded matrix multiplied by a scalar α that depends on the bending elasticity modulus and on the distance between the points.

5.2. Mobility

The motion of an object at low Reynolds number is characterized by a mobility. This is defined by factors which link speed and force (speed = mobility \times force). These factors depend on the size and shape of the object, and on the viscosity η of the surrounding fluid. For instance, a straight cylinder has two mobility factors, because it is twofold easier to move in the longitudinal direction than in a transverse direction. This anisotropy could not be implemented simply, because fibers in the simulation may bend and adopt arbitrary shapes. An exact calculation would require finding the hydrodynamic interactions between all the points in the system. This can be done in the future, but for simplicity, we have so far used the averaged mobility of a straight rod of length L and diameter δ : $\mu = \log(L_h/\delta)/3\pi\eta L$ [30]. The logarithmic term is an effective hydrodynamic correction on the scale L_h , which is either the length of the fiber, or a hydrodynamic cut-off, whatever is smallest. We derive a single mobility factors for the $p + 1$ points representing a fiber: $\mu_p = (p + 1) \mu$.

5.3. Projector associated with the constraints

In this section, we calculate the projection P derived from the constraint that the length of the fiber should remain constant during the resolution of equation (1). For each fiber, the coordinates of the $p + 1$ model-points m_k are stored in a vector of dimension $(p + 1)d$ (for $d = 3$, $\{x_0, x_1, x_2\}$ correspond to m_0 , and $\{x_3, x_4, x_5\}$ to m_1 , etc). The motions of these points are determined by

external forces $\mathbf{f} = \{f_k\}$, and additionally by internal forces $\hat{\mathbf{f}} = \{\hat{f}_k\}$. The speeds resulting from $\hat{\mathbf{f}} + \mathbf{f}$ should be compatible with the constraints $C_k = (m_{k+1} - m_k)^2 - (L/p)^2 = 0$ for $k \in [0, p]$. To calculate $\hat{\mathbf{f}}$ from \mathbf{f} , we first define the $p \times d(p+1)$ Jacobian matrix $J_{ij} = \partial C_i / \partial x_j$. In 3D, it reads:

$$J = 2 \begin{pmatrix} x_0 - x_3 & x_1 - x_4 & x_2 - x_5 & x_3 - x_0 & x_4 - x_1 & x_5 - x_2 & 0 & 0 & 0 & \cdots \\ 0 & 0 & 0 & x_3 - x_6 & x_4 - x_7 & x_5 - x_8 & x_6 - x_3 & x_7 - x_4 & x_8 - x_5 & \cdots \\ & & & & \vdots & & & & & \ddots \end{pmatrix}.$$

Because the mobility coefficients are the same for all the points (μ_p , see section 5.2), the speed of the points is $\mathbf{v} = \mu_p (\mathbf{f} + \hat{\mathbf{f}})$. This motion maintains the constraints if $J \mathbf{v} = 0$. Therefore $\hat{\mathbf{f}}$ must be such that $J(\mathbf{f} + \hat{\mathbf{f}}) = 0$. Furthermore, internal forces should not contribute to the global motion or the rotation of the object. This imposes that their work should be null for any motion compatible with the constraints: $\hat{\mathbf{f}} \cdot \mathbf{u} = 0$ for any \mathbf{u} such that $J \mathbf{u} = 0$. This implies that $\hat{\mathbf{f}} = J^t \lambda$, where λ is a vector of size p (the Lagrange multipliers). We derive $J(\mathbf{f} + J^t \lambda) = 0$, and since $J J^t$ of size $p \times p$ is non-singular, $\lambda = -(J J^t)^{-1} J \mathbf{f}$, and finally $\hat{\mathbf{f}} = -J^t (J J^t)^{-1} J \mathbf{f}$. This shows that the total force can be obtained linearly as $\mathbf{f} + \hat{\mathbf{f}} = P \mathbf{f}$, with $P = I - J^t (J J^t)^{-1} J$. From this result, it is clear that P is an orthogonal projection (P is symmetric and idempotent $P P = P$). Notice that $J J^t$ is banded symmetric, and therefore easy to invert, which means that P can be computed fast. P (which depends solely on \mathbf{x}) is one block of the operator P_t used in equation (2).

Fibers are ‘reshaped’ to restore the constraints exactly after the model-points have been moved. This is done sequentially for $k \in [0, p]$, by moving the points m_0, \dots, m_k in the direction of $m_{k+1} - m_k$ and m_{k+1}, \dots, m_p in the opposite direction, to restore $|m_{k+1} - m_k| = L/p$ while conserving the center of gravity of the fiber.

5.4. Brownian motion

To simulate Brownian motion, a term δB_t is attributed to each fiber coordinate x_t (equation (2)). This term is most simply calibrated by considering diffusion in the absence of bending or external forces ($A = 0$ and $G = 0$). If we first assume $P_t = I$ in equation (2), we get $x_{t+h} - x_t = \delta B_t$. To produce a pure diffusion with a coefficient D , one needs:

$$\langle x_{t+\tau} - x_t \rangle = 0, \quad \langle (x_{t+\tau} - x_t)^2 \rangle = 2 D \tau.$$

This holds true, if δB_t is normally distributed, of mean zero and variance $2D\tau$. We can use $\delta B_t = \beta \theta$, where $\theta \sim N(0, 1)$ is a random number generated for each time step, and $\beta = \sqrt{2D\tau}$, as mentioned in section 4. From Einstein’s relation, we set $D = \mu_p k_B T$, where μ_p is the mobility, k_B the Boltzmann constant, and T the absolute temperature. For a fiber with $p+1$ points, we use $(p+1)d$ random numbers, independent and all normally distributed of variance β^2 . Projecting these numbers with P produces the appropriate diffusion for the fiber, as well as thermally-driven deformations. For example, the translation x of the center of gravity depends on the sum of all the terms in δB corresponding to the fiber, leading to a diffusion $D = \mu k_B T$ (with μ and not μ_p).

6. Spherical set of points (sphere)

To simulate the nucleus of *S. pombe* and attach microtubules on its surface (see figure 2B), we implemented a ‘spherical set of points’ of radius r . Such object is composed of a point n_0 in the center, and q additional points n_i on the periphery. If we define $r_k = n_k - n_0$, the constraints are $|r_k| = r$. A *sphere* moves as a rigid body, and the peripheral points behave as if they were embedded in a viscous surface (see figure 1). If f_k is the force applied at point k , the motion of the set reads:

$$\begin{aligned} dn_o &= \mu^T F dt + dB^T, \\ dr_k &= (\mu^R M dt + dB^R) r_k + P_k (\mu^S f_k dt + dB_k^S), \end{aligned} \quad (3)$$

where $F = \sum_{i=0}^q f_i$ is the total force on the sphere, $M = \sum_{i=1}^q r_i \times f_i$ is the total torque calculated from the center, and where

$$P_k = I - \frac{r_k \otimes r_k}{r_k^2},$$

is the projection on the plane tangent to the sphere in r_k . dB^R , dB^T and dB_k^S are the Brownian terms. Note that these equations would not describe a set of peripheral points articulated around a central node. For example, the motion of the center n_0 depends on the sum of all the forces applied to the object, and not only on the force applied in n_0 . This in fact corresponds to a sphere with points on its surface. To keep track of the orientation of the sphere, we also included three reference points \tilde{n}_k on the surface, which form with n_0 a reference frame associated to the sphere. The motion of these reference points is entirely determined by the total torque on the sphere: $d\tilde{r}_k = (\mu^R M dt + dB^R) \times \tilde{r}_k$, where as before $\tilde{r}_k = \tilde{n}_k - n_0$. When the object needs to be ‘reshaped’, the peripheral points are simply projected on the surface (n_0 is not moved).

6.1. Mobility and Brownian motion

The equations involve three mobility factors: the translation and rotational mobility of the sphere μ^T and μ^R , and the mobility of the points in the surface μ^S . Stokes’ law can be used to set μ^T and μ^R , if the sphere is surrounded by a large volume of fluid. The mobility coefficients for the points in the surface can also be calculated [31]. As described above, points undergo three different types of motion, and a random number δB_t in equation (2) is associated with each of these motions. The parameters are calculated by considering diffusion in the absence of other forces ($A = 0$ and $G = 0$). For the translational diffusion of the sphere, the result from equation (3) is obtained as previously for the fiber: $\beta^T = \sqrt{2\mu^T \tau k_B T}$. Rotational diffusion is calibrated using equation (3). If r_t is fixed on the surface, we get $r_{t+\tau} - r_t = \delta B_t^R \times r_t$. This should be a rotational diffusion of a point on a sphere:

$$\langle r_{t+\tau} - r_t \rangle = 0, \quad \langle (r_{t+\tau} - r_t)^2 \rangle = 4 k_B T \mu^R r^2 \tau.$$

Since $|r_t| = r$, we can use for δB_t^R a random vector with d independent components of mean zero and variance $2\tau \mu^R k_B T / r^2$. A peripheral point r_t also diffuses on the surface, which in equation (3) is described by $r_{t+\tau} - r_t = P_k \delta B_{k,t}^S$. The projection p_t of r_t should diffuse in 2D:

$$\langle p_{t+\tau} - p_t \rangle = 0, \quad \langle (p_{t+\tau} - p_t)^2 \rangle = 4 k_B T \mu^S \tau.$$

Since P_k is the identity in the tangent plane, we used for $\delta B_{k,t}^S$ a vector with d independent components of mean zero, and variance $2 \tau \mu^S k_B T$.

7. Non-deformable set of points (solid)

We also implemented non-deformable objects called *solids* (see figure 1) in which the points move together in such a way that the shape and size of the set is conserved. The number of points p in a *solid*, and their positions s_i can be chosen arbitrarily, and each point is associated with a radius $a_i \geq 0$. The mobility of the *solid* is derived from Stokes's result for the spheres of center s_i and radius a_i , neglecting for simplicity the hydrodynamic interactions between the spheres. It is possible to include points with $a_i = 0$ provided that $\sum_i a_i > 0$. In our previous work, we have actually used *solids* where only one a_i was non-zero. These solids moved like isolated spheres, and the points a_i where positions to which forces could be applied.

7.1. Mobility and constrained motion

Because the set of points should not deform, its elementary motion during a time-step can be written as $(s_i^{t+\tau} - s_i^t)/\tau = v + \omega \times s_i^t$, where v and ω are instantaneous translation and rotation speeds. The spheres of radius a_i in a medium with viscosity η have a translational drag coefficient $\xi_i = 6\pi\eta a_i$, and a rotational drag coefficient $\xi_i^\omega = 8\pi\eta a_i^3$ [30]. The forces and torques resulting from the friction of the fluid on the sphere thus read:

$$\tilde{f}_i = \xi_i (v + \omega \times s_i), \quad \tilde{M}_i = \xi_i^\omega \omega,$$

and should match the externally applied forces f_i :

$$\sum_i \tilde{f}_i = \sum_i f_i, \quad \sum_i s_i \times \tilde{f}_i + \tilde{M}_i = \sum_i s_i \times f_i.$$

This set of four equations can be solved algebraically in both 2D and 3D, to express v and ω as a function of the external forces f_i . The result always fits in the format of equation (1). It is actually not necessary to calculate the matrix P to run a simulation. It is more efficient to calculate v and ω when the product $P\mu f$ is needed. To 'reshape' a *solid*, one may restore a reference configuration in the current position and orientation. For this, the best translation and rotation which brings the reference points on to the current points is calculated [32]. The current points are then replaced by the transformed reference configuration. The Brownian components are calibrated as described before.

8. Interactions between objects

The three objects defined previously can be linked together using elementary interactions. By adding the contributions of all these interactions in the system, we obtain the linearized force $F(\mathbf{x}, t) = A_t \mathbf{x} + G_t$, which enters equation (2). In practice, each elementary interaction leads to a small matrix, which needs to be added to the matrix A_t and vector G_t , at the right rows

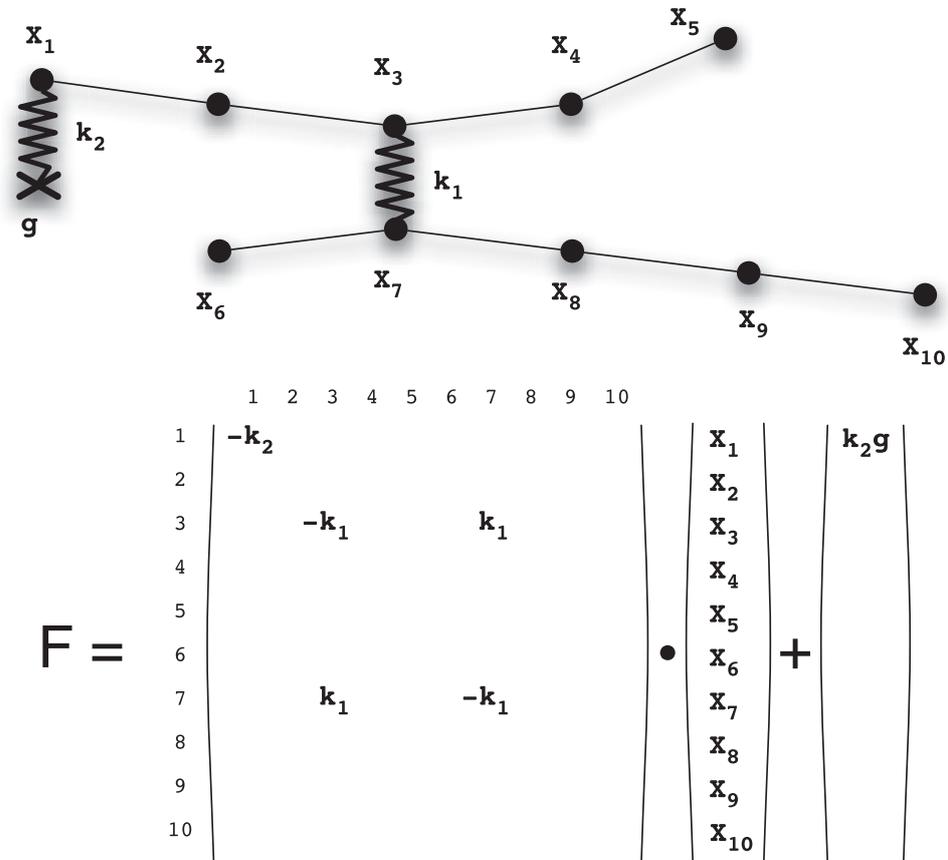


Figure 6. Stiffness matrix and force vector. The stiffness matrix A_t and the force vector G_t in equation (2) are set by considering all the interactions present at time t . For each interaction, the appropriate formula (section 8) is first expanded algebraically. The factors associated with the coordinates of the points are added to A , and the coefficients which are independent of the coordinates are added to G . At the end of the procedure, one obtains a (sparse) symmetric matrix A and a vector G that provide the forces on the points $F = Ax + G$. Here, we illustrate how a connection of stiffness k_1 (section 8.2) contribute to factors k_1 and $-k_1$ at the rows and columns of A corresponding to the points connected. For a connection to a fixed position g (section 8.1), a stiffness coefficient $-k_2$ is added in A , while k_2g is added in G . In this example, the connections are attached exactly to points of the system, but this is not always the case. Section 8 explains the general procedure. In addition, the matrix represented here corresponds to a 1D system. It needs to be duplicated for a 2D simulation, and triplicated in 3D (section 8.5).

and columns to correspond to the appropriate points (see example on figure 6). It is necessary to repeat the procedure at every time step, because the position of the interactions may change with respect to the model-points. We define four interactions in the case where they connect model-points of the objects. We later explain the procedure to connect intermediate positions

between the model-points. This approach can be generalized to more complicated interactions if necessary. For example, it is possible to implement a ring which is able to slide along a fiber with viscous resistance [33].

8.1. Connecting an object to a fixed position

The simplest way to immobilize an object is to attach a point a within the object to a fixed position g . If the stiffness of the link is k , the resulting force is $f_a = k(g - a)$. In practice, this means adding $-k$ at one diagonal position in matrix A_t , and kg to the vector G_t (see figure 6). Such interactions are used to model gliding assays (see figure 8) in which motors immobilized on a surface propel fibers in solution. Each attached molecular motor leads to an elementary interaction where g corresponds to the place of immobilization, and a corresponds to the position on the fiber at which the motor domain is attached.

8.2. Connecting two objects

Points from two different objects can be connected by a link of stiffness k . The forces between the points are $f_a = -f_b = k(b - a)$. These elementary interactions are effective to model oligomeric motors [26] and more generally any entity which is able to connect two fibers together (see figure 2C). In the case of an oligomeric motor, a and b are the positions to which the two motor domains are attached on the fibers.

8.3. Confinement in a convex shape

To confine the objects inside a convex shape, we use a harmonic potential that is flat inside the allowed region, and rises quadratically away from its edge. Hence, a point a outside the cell volume is subject to a force $f(a) = k(p(a) - a)$, where $p(a)$ is the closest point to a on the edge of the allowed volume. Because p is also the orthogonal projection of a , the force corresponds to a friction-less edge. We linearized f as $x \rightarrow k(e_a \cdot (p(a) - x))e_a$, where e_a is a unit vector in the direction of $p(a) - a$. This linearization corresponds to the tangent plane in $p(a)$, and usually gives a good approximation of $f(a)$ as long as the curvature is small. To confine a *fiber*, it is sufficient to follow the procedure for its model-points, if the volume is convex, which is the case for example of the cylindrical yeast *S. pombe* (see figure 2B). To confine the nucleus of radius r in the same volume, we used a cell volume reduced by r . In this way only the center of the *sphere* needs to be tested.

8.4. Connecting two objects at a given distance

A Hookean spring of stiffness k with a non-zero resting length r between two points a and b corresponds to:

$$f_a = -f_b = -k \left(1 - \frac{r}{|\delta|} \right) \delta,$$

with $\delta = a - b$. This force should be linearized for $|\delta| \approx r$, leading for a to a term $kr\delta/|\delta|$ in G_t and a contribution in A_t which is:

$$-k \frac{\delta \otimes \delta}{\delta^2}, \quad \text{if } |\delta| \leq r, \quad \text{and} \quad -k \left[I - \frac{r}{|\delta|} \left[I - \frac{\delta \otimes \delta}{\delta^2} \right] \right], \quad \text{otherwise,}$$

and the opposite contributions for b . This interaction can be useful to introduce a repulsion between the points. It can, for example, represent the physical interaction between the nuclear membrane and the microtubules in *S. pombe* (see figure 2B).

8.5. Interpolation of forces

We have discussed connections which were attached to model-points. However, in the case of a *fiber*, a molecule may bind at any position x , which is likely to be between the two model-points m_k and m_{k+1} . When this happens, a is interpolated from the flanking model-points using a coefficient $\alpha = |m_k x|/|m_k m_{k+1}|$ in $[0, 1]$. In the same way, a force f applied in x can be distributed to the model-points as $f_k = (1 - \alpha)f$ and $f_{k+1} = \alpha f$. Since this procedure preserves any linearity in the relationship between force and coordinates, the different matrix elements mentioned previously can be used with interpolated points, provided they are multiplied left and right by an appropriate weight matrix. We can illustrate the procedure for the simplest connection $f_a = -f_b = k(b - a)$ of stiffness k between two points a and b (section 8.2), which reads:

$$\begin{pmatrix} f_a \\ f_b \end{pmatrix} = \begin{pmatrix} -k & k \\ k & -k \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix}.$$

When a and b are model-points, this 2×2 matrix is a reduction of A , corresponding to the x , y or z -subspaces. This is sufficient in this case because a Hookean spring of null resting length is *isotropic*, that is to say it does not mix x -, y - and z -coordinates, and applies similarly to each subspace. This is not the case for all interactions discussed in this section, and it is often necessary to calculate a full matrix. Moreover, when a and b are intermediate positions between the model-points, we have two indices k, l and two interpolation coefficients α, β such that $a = (1 - \alpha)m_k + \alpha m_{k+1}$ and $b = (1 - \beta)m_l + \beta m_{l+1}$. If we define $\bar{\alpha} = 1 - \alpha$ and $\bar{\beta} = 1 - \beta$, and

$$w = \begin{pmatrix} \bar{\alpha} & \alpha & 0 & 0 \\ 0 & 0 & \bar{\beta} & \beta \end{pmatrix},$$

we get:

$$\begin{pmatrix} f_k \\ f_{k+1} \\ f_l \\ f_{l+1} \end{pmatrix} = w^t \begin{pmatrix} f_x \\ f_y \end{pmatrix} = -k w^t \begin{pmatrix} -1 & 1 \\ 1 & -1 \end{pmatrix} w \begin{pmatrix} m_k \\ m_{k+1} \\ m_l \\ m_{l+1} \end{pmatrix}.$$

The resulting 4×4 matrix is $\bar{w}(-k)\bar{w}^t$, with $\bar{w}^t = (\bar{\alpha}, \alpha, -\bar{\beta}, -\beta)$. We derive that a matrix made by adding multiple such interactions is symmetric negative-semidefinite ($x^t A x \leq 0$, for any x). The fact that this is true for any configuration of the connections guarantees the numerical stability of the method, as explained in the next section.

9. Numerical stability and performance

We have described all the components of equation (2) which describes the collective mechanics of cellular fibers and other objects. The necessary steps of the calculation are summarized in figure 7. It is useful at this stage to examine the method mathematically. This is usually done by looking at two properties: precision and numerical stability [29]. The precision is a measure of how the typical error behaves when the time-step τ becomes small. The numerical stability is a measure of how large τ can be, before the calculation fails. Numerical precision is important for deterministic equations, for example, to predict the trajectories of celestial bodies. However, this is not so critical at the cellular scale. In fact, to simulate the Brownian motion present in the cell, a random term $\delta B \sim \sqrt{\tau}$ was included in equation (2). The presence of this ‘noise’ indicates that the physics itself limits the precision at which the position of an object can be predicted. This fact undermines the usefulness of high precision schemes. The implicit method that we have described is of order one: the step’s error scales like $O(\tau^2)$, which is better than the physical ‘noise’ in $\sqrt{\tau}$. We found that it was not practically useful to use higher order numerical schemes.

In contrast, the numerical stability of the method is most important. Indeed, explicit schemes usually converge only if the time-step is small. In general, a condition like $\tau \mu k < 1$ must be fulfilled, where μ is the mobility of a point in the system, and k the stiffness of the interaction potential. For example, we looked at a test-case in which a microtubule is pushed by immobilized motors (see [25] and figure 8). It can be simulated explicitly only if $\tau < 1 \mu s$, but the implicit method can use larger time-steps. To achieve this stability, we treated the repulsive and attractive interactions in the system differently. Compressive forces in the fibers (which are repulsive in nature) were replaced by constraints. All the other forces were attractive. This ensured that A_t would be negative-semidefinite (this result was proven in section 8.2 for Hookean interactions of null resting length). Mathematically, because P_t is an orthogonal projection, we can show that the eigenvalues of $I - \tau \mu P_t A_t$ are always greater than 1, for any value of τ . This implies that our integration scheme is unconditionally stable. For the other elementary interactions, some instabilities may appear, but only for very high values of the time step (not shown).

Beyond stability, other considerations naturally limit the choice of τ . In particular, the iterative solver might not converge when τ is large. The optimal time-step generally depends on the problem studied, and it is best to perform systematic trials to find it. For the test-case (see figure 8), the results are consistent for $\tau < 20$ ms. This means that a value of 5 or 10 ms would be appropriate. The computational requirements depend on the total number of steps (total time/time-step), but also on the cost of individual steps. An implicit step of integration is always more costly than an explicit step, because a linear system must be solved. However, the use of sparse matrix techniques reduces the additional work. In practice the considerable reduction in the number of steps makes implicit simulations faster (in the test-case, this gain is 10^4 , using $\tau = 10$ ms instead of $1 \mu s$). Increasing the execution speed is essential if many simulations need to be performed. Implicit methods require increased numerical labor, of which we have illustrated the main difficulties. Using the method described here, we can simulate the examples shown in figure 2(B–D) much faster than the real time using one processor (<http://www.cytosim.org>).

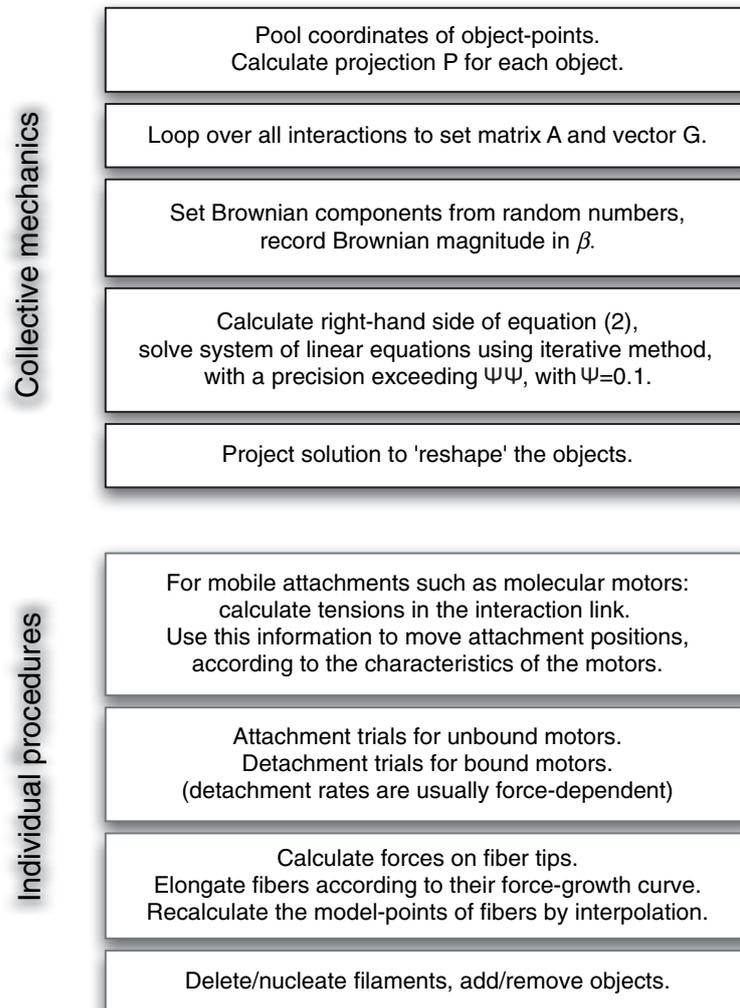


Figure 7. Synopsis of a simulation time-step. Sub-steps necessary to simulate a system of molecular motors and dynamic fibers. The collective mechanics corresponds to the algorithm described in the paper. As a byproduct of calculating the mechanics, one gets the tensions in the fibers and the forces connecting the fibers. With this information, simulation sub-steps can be performed for the objects independently. Events such as the binding and the unbinding of motors and the nucleation of new filaments will most likely be modeled stochastically. Depending on the level of details required, less-discrete events may be simulated in a deterministic manner. For example, the active motion of molecular motors and the assembly dynamics of cytoskeletal fibers can be simulated as non-random processes characterized by a force-velocity curve.

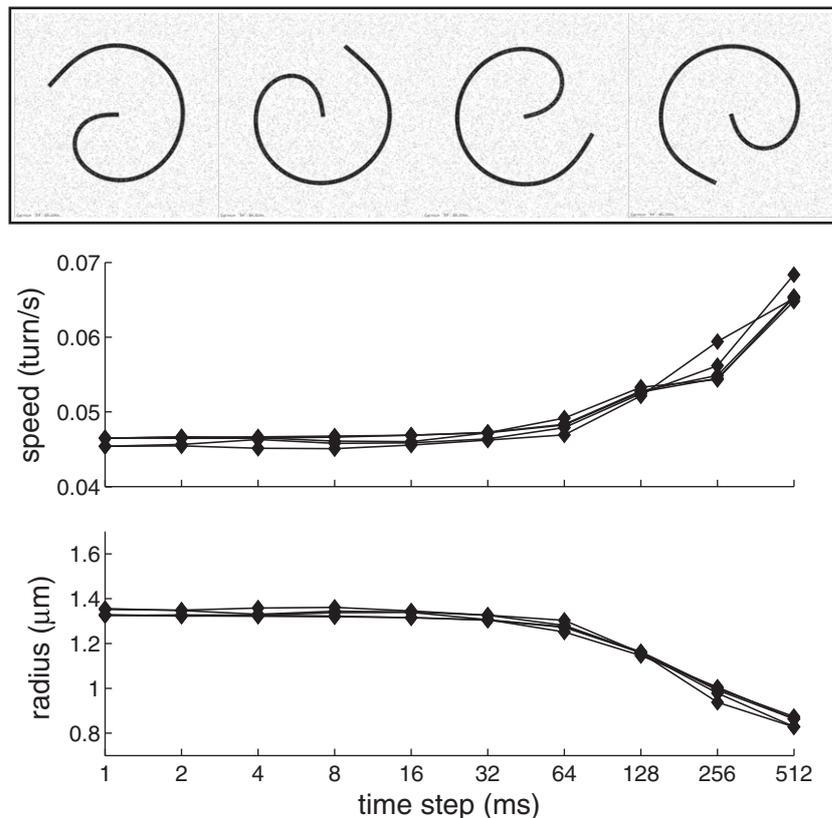


Figure 8. Numerical stability of the integration scheme. Top: a gliding assay where a filament is attached at its end (time-intervals of 5 s). The motors pushing the fiber lead to the formation of a rotating spiral, as observed experimentally [25]. The rotation speed and maximum radius of the spiral can be calculated from the parameters of the system: 16 000 motors cover an area of $2 \times 2 \mu\text{m}$, and have the characteristics of kinesin: stall force $f_{\text{max}} = 5 \text{ pN}$, unloaded speed $0.4 \mu\text{m s}^{-1}$, binding rate 10 s^{-1} , unbinding rate $p_{\text{off}} = 0.5 \text{ s}^{-1} \exp(\text{force}/2.5 \text{ pN})$, maximum binding distance 10 nm and stiffness $200 \text{ pN } \mu\text{m}^{-1}$. The microtubule of length $8 \mu\text{m}$ has a rigidity of $20 \text{ pN } \mu\text{m}^2$. It is constrained at the minus end by a link of stiffness $4000 \text{ pN } \mu\text{m}$. The effective viscosity is $0.02 \text{ pN s } \mu\text{m}^{-2}$. Bottom: the configuration is simulated for different values of the time-step τ , with accurate results for $\tau < 20 \text{ ms}$. The algorithm is numerically stable, and even produces a spiral with $\tau \sim 0.5 \text{ s}$. However, the radius is then underestimated, and the rotation speed overestimated. Another critical parameter, the distance ρ between the points on the fiber was also varied. The results shown for $\rho = 0.1, 0.2, 0.4$ and $0.5 \mu\text{m}$ (different lines) are similar, because all these values are appropriate. The calculations were inaccurate however with $\rho = 0.8 \mu\text{m}$ (data not shown). This is expected considering that the radius of the spiral is $\sim 1.4 \mu\text{m}$.

10. Other elements of a cytoskeletal simulation

In addition to mechanics, a cytoskeletal simulation such as **cytosim** must include additional aspects such as the motion of molecular motors, their binding/unbinding dynamics, as well as the transitions between growth and shrinkage of dynamic fibers. These processes can be modeled most simply by executing small sub-routines after the Brownian mechanics has been calculated, because they correspond to independent operations (see figure 7). However, two particularly important aspects of cytoskeletal physics need to be mentioned. Firstly, only in very particular cases can we approximate the system as a well-mixed reactor. At least some of the molecules should be spatially resolved. Secondly, the mechanics commonly affects the chemistry. For instance, the rates of certain key reactions are force-dependent. This is the case for the unbinding rates of molecular motors and for their stepping rate (see below). Because these elements are essential for modeling the system accurately, it will be rarely possible to apply algorithms developed for purely chemical systems (e.g. the Gillespie algorithms [34] or even spatially resolved methods [35]) without extensive modifications. We can however use simple and robust simulation strategies, as illustrated below in the case of molecular motors.

10.1. Modeling molecular motors

In **cytosim**, a motor is characterized by a position, when it is not attached, and by a pointer to a fiber and a curvilinear abscissa, when it is attached (see figure 9). The abscissa is the distance, measured along the fiber, between a reference and the attachment position. It is necessary to use a reference which is fixed with respect to the physical lattice, because the model-points of a *fiber* are themselves updated as the fiber grows (see figure 4). This description neatly separates the details of how the mechanics is implemented from the routines simulating the motors *per se*. This means that the interface with the rest of the program can be very simple, with only two procedures: *step(f)* and *attach(m)*.

10.1.1. Active motion. The first procedure *step(f)* simulates the possible actions of a bound motor. The argument f is the load of the motor calculated during the collective mechanics. The procedure should decide to detach the motor, or to update the abscissa a according to a microscopic model for the interval τ . For a well-characterized motor like kinesin, a classical model is based on the measured characteristics of the motion: the abscissa is increased by $\delta a = \tau v_{\max}(1 - f/f_{\text{stall}})$. In addition, a force-dependent unbinding rate $p_{\text{off}} = p_0 \exp(|f|/f_0)$ is used to model the dissociation from the fiber. v_{\max} , p_0 , f_0 and f_{stall} are characteristics of the motor that have been measured for kinesin [1]. With this model, the fibers are continuous tracks along which motors may be located anywhere. Alternatively, we may model the motion of a motor as a succession of discrete stochastic steps. In this case, the motor does one of four things: stay immobile, detach, take a step toward the minus-end or take a step toward the plus-end. This means that if the motor does not detach, the abscissa is either unchanged, or it is increased or decreased by the step size (8 nm). The procedure *step(f)* calculates the probabilities of these events as a function of the force f for the interval τ , and selects one of them. This model is quite attractive, because these probabilities are actually available for kinesin [36]. Most models describing the movement of motors [19] can be summarized similarly with a function *step(f)*.

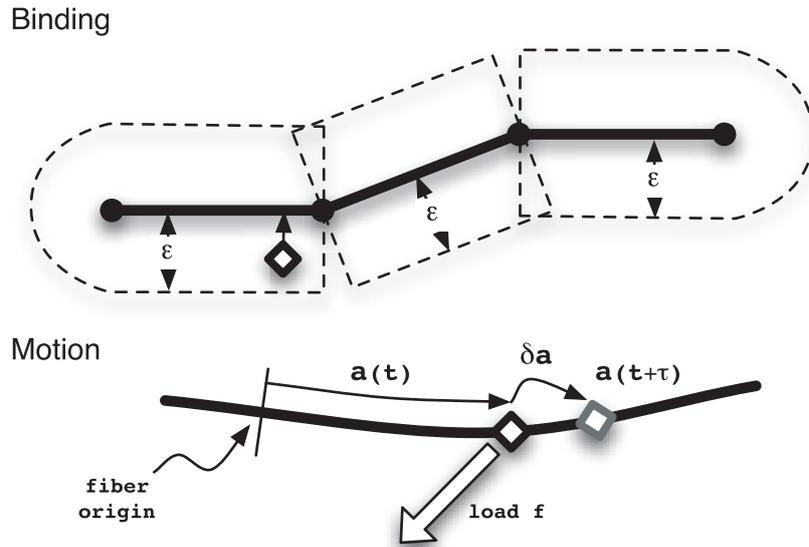


Figure 9. Molecular motors. Top: an unbound motor (diamond) is represented by a position. Attachment occurs on the closest site on the fiber-segment, provided this site is within a distance ϵ (dashed lines). The capture regions of the segments are truncated such that they cover exactly the region located at a distance ϵ from a straight fiber. When the fiber is not straight, the gaps and overlaps exactly compensate each other. Bottom: a bound motor is represented by a pointer to a fiber, and by a curvilinear abscissa $a(t)$ measured from a fixed origin on the fiber. This defines the position of the motor along the fiber independently of the mathematical representation of the fiber. The motor sub-model needs to decide whether the motor should detach during the interval of time τ , or it needs to calculate the displacement δa during the same interval. For this, it can use the load f calculated during the collective mechanics, and other properties associated with the fiber, such as the proximity of the ends, or information on the crowdedness of the binding sites on the fiber.

10.1.2. Attachment to fibers. The second procedure necessary to model motors, $attach(m)$ simply decides if a unbound motor binds or not to a site m . Usually the model would specify ϵ , a maximum distance at which a motor may bind from its current position (see figure 9). In addition, the molecule would bind at the closest site on the fiber (the orthogonal projection) with a certain molecular binding rate k_{on} (s^{-1}). To simulate attachments, one therefore needs to first find the fiber-segments which are closer than ϵ , typically from all the positions x at which molecular motors are located. For each point x , the list of candidates should then be shuffled, to ensure a random ordering of the segments. The molecular binding rate can finally be tested sequentially for each segment in the list, for example, by comparing τk_{on} with a random number θ in $[0,1]$. The first successful trial is followed by attachment. If done naively, the first step of the operation may require calculating the distance of all points to all fiber-segments, and thus a great deal of computation for many motors. To avoid this bottleneck in *cytosim*, a divide and conquer algorithm was developed (see figure 10). Its goal is to limit the number of segments

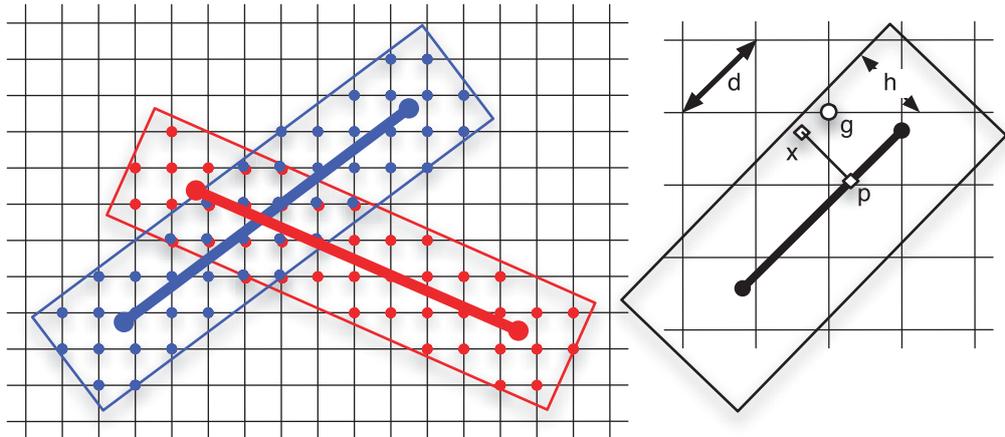


Figure 10. Divide and conquer algorithm. To simulate the attachments to fibers, we must be able to find all the fiber-segments which are within a distance ϵ from an arbitrary position x . We can proceed according to the following two steps method. Divide (left): a grid is set in space, each node of the grid being associated with a list of segments. The segments are recorded on the grid, at the nodes located at a distance h or less (h will be defined later). This operation is performed in 2D using standard rasterizer codes derived from computer graphics, which are optimized to scan all points with integer coordinates located inside an arbitrary polygon. We rasterize the rectangles built around the segments at a distance h . For example, on this diagram, the blue segment is recorded at the blue points, and the red segment at the red points. In 3D, the rectangular volume can be rasterized following the same principles as in 2D. Conquer (right): after the segments have been distributed over the grid, one can quickly find which ones are near x : one needs to check only the segments recorded at the grid point g closest to x . One will find all segments located at distance $h - d/2$ or less from x , since $|gx| < d/2$, where d is the diagonal of the grid. Hence, to find all the segments closer than ϵ , one sets $h = \epsilon + d/2$ during the rasterizing operation. Note: the grid does not need to be square (the unit cell can be rectangular) and it can be adjusted for optimal performance. If the grid is too fine, it will use a lot of memory and rasterizing will be slow. If the grid is coarse (d large), the number of candidates returned for a point x will be larger. Experimentation may be necessary to optimize the grid, but the procedure provides exact results for any cell size.

that need to be tested to find those which are close to x . The geometrical distance between x and these segments is calculated using the vector cross-product to exactly determine which ones are closer than ϵ . Reducing the number of tested segments is sufficient to accelerate the simulation.

11. Conclusion

The method described here is efficient to simulate sparsely connected networks of filaments. It applies to many *in vivo* situations, because the connections between fibers are usually mediated by proteins that are small compared to the fibers, and consequently the fibers are only locally

connected. We have modeled fibers as oriented lines, which is sufficient to calculate the extent of bending. It may be necessary in the future to include more details such as writhe, since cytoskeletal fibers also have a torsional rigidity. The method can be extended in several other ways. One could, for instance, easily model discrete binding sites on the fibers. This may be important if the fibers are highly covered and molecules compete or interact while bound to the lattice. It is also possible to extend the overdamped mechanics by adding hydrodynamic effects. It will be very exciting to integrate fiber mechanics with membrane dynamics, since membranes and cytoskeleton contribute synergistically to cellular architecture, but this might take some time. Cellular chemistry, reaction–diffusion of the components in the cell, gene expression networks, can be added more simply. This can be done by interfacing our software with other tools (e.g. the Virtual Cell project), which already cover some of these aspects of physiology. We did not discuss here implementation issues, but the scale of the task should remind us of their importance. Software modularity is essential to divide the development effort into separate projects of manageable size. Sub-models or algorithms should be developed and tested separately, in such a way that they can be added or removed from the integrative software easily. Dividing the work among different groups is the best way to produce the high-quality cellular simulations that biology needs.

Acknowledgments

This method was designed in 2001 [26] and extended by Dietrich Foethke to *spheres*. We thank the members of our laboratory, and in particular Rose Loughlin and Cleopatra Kozlowski for their help in developing Cytosim, and for critically reading this paper. We thank Jonathan Ward for his critical reading, Tony Lelievre and Rafe Mazzeo for mathematical insights. We acknowledge support from IBM, BioMS (<http://www.bioms.de>), the Volkswagenstiftung initiative ‘new conceptual approaches to modeling and simulation of complex systems’ and HFSP grant RGY84.

References

- [1] Howard J 2001 *Mechanics of Motor Proteins and the Cytoskeleton* (Sunderland, MA: Sinauer)
- [2] Desai A and Mitchison T 1997 Microtubule polymerization dynamics *Annu. Rev. Cell Dev. Biol.* **13** 83–117
- [3] Garner E, Campbell C and Mullins R 2004 Dynamic instability in a dna-segregating prokaryotic actin homolog *Science* **306** 1021–5
- [4] Wittmann T, Hyman A and Desai A 2001 The spindle: a dynamic assembly of microtubules and motors *Nat. Cell Biol.* **3** E28–E34
- [5] Blaineau C, Tessier M, Dubessay P, Tasse L, Crobu L, Pagès M and Bastien P 2007 A novel microtubule-depolymerizing kinesin involved in length control of a eukaryotic flagellum *Curr. Biol.* **17** 778–82
- [6] Varga V, Helenius J, Tanaka K, Hyman A, Tanaka T and Howard J 2006 Yeast kinesin-8 depolymerizes microtubules in a length-dependent manner *Nat. Cell Biol.* **8** 957–62
- [7] Janson M, Loughlin R, Loïodice I, Fu C, Brunner D, Nedelec F and Tran P 2007 Crosslinkers and motors organize dynamic microtubules to form stable bipolar arrays in fission yeast *Cell* **128** 357–68
- [8] Janson M, Setty T, Paoletti A and Tran P 2005 Efficient formation of bipolar microtubule bundles requires microtubule-bound gamma-tubulin complexes *J. Cell Biol.* **169** 297–308
- [9] Mahoney N, Goshima G, Douglass A and Vale R 2006 Making microtubules and mitotic spindles in cells without functional centrosomes *Curr. Biol.* **16** 564–9

- [10] Loiodice I, Staub J, Setty T, Nguyen N, Paoletti A and Tran P 2005 Ase1p organizes antiparallel microtubule arrays during interphase and mitosis in fission yeast *Mol. Biol. Cell* **16** 1756–68
- [11] Ambrose J C, Li W, Marcus A, Ma H and Cyr R 2005 A minus-end-directed kinesin with plus-end tracking protein activity is involved in spindle morphogenesis *Mol. Biol. Cell* **16** 1584–92
- [12] Goshima G, Nedelec F and Vale R 2005 Mechanisms for focusing mitotic spindle poles by minus end-directed motor proteins *J. Cell Biol.* **171** 229–40
- [13] Morrison E E 2007 Action and interactions at microtubule ends *Cell. Mol. Life Sci.* **64** 307–17
- [14] Gerhart J and Kirschner M 1997 *Cells, Embryos, and Evolution* (Oxford: Blackwell Science)
- [15] Mogilner A, Wollman R, Civelekoglu-Scholey G and Scholey J 2006 Modeling mitosis *Trends Cell Biol.* **16** 88–96
- [16] Karsenti E, Nedelec F and Surrey T 2006 Modelling microtubule patterns *Nat. Cell Biol.* **8** 1204–11
- [17] Lemerle C, Di Ventura B and Serrano L 2005 Space as the final frontier in stochastic simulations of biological systems *FEBS Lett.* **579** 1789–94
- [18] Dobrzynski M, Rodríguez J V, Kaandorp J A and Blom J G 2007 Computational methods for diffusion-influenced biochemical reactions *Bioinformatics* **23** 1969–77
- [19] Kolomeisky A B and Fisher M E 2007 Molecular motors: a theorist's perspective *Annu. Rev. Phys. Chem.* **58** 675–95
- [20] Landau L and Lifshitz E 1986 *Theory of Elasticity* (Oxford: Butterworth-Heinemann)
- [21] Leighton R, Feynman R and Sands M 1989 *The Feynman Lectures on Physics* (Reading, MA: Addison-Wesley)
- [22] Purcell E 1977 Life at low Reynolds number *Am. J. Phys.* **45** 3–11
- [23] Lemons D S 1949 *An Introduction to Stochastic Processes in Physics* (Baltimore, MD: Johns Hopkins University Press)
- [24] Deutsch J 1988 Theoretical studies of DNA during gel electrophoresis *Science* **240** 922–4
- [25] Bourdieu L, Duke T, Elowitz M B, Winkelmann D A, Leibler S and Libchaber A 1995 Spiral defects in motility assays: a measure of motor protein force *Phys. Rev. Lett.* **75** 176–9
- [26] Nedelec F 2002 Computer simulations reveal motor properties generating stable antiparallel microtubule interactions *J. Cell Biol.* **158** 1005–15
- [27] Kozlowski C, Srayko M and Nedelec F 2007 Cortical microtubule contacts position the spindle in *C. elegans* embryos *Cell* **129** 499–510
- [28] Moller-Jensen J, Borch J, Dam M, Jensen R, Roepstorff P and Gerdes K 2003 Bacterial mitosis: parm of plasmid r1 moves plasmid DNA by an actin-like insertional polymerization mechanism *Mol. Cell* **12** 1477–87
- [29] Press W, Teukolsky S, Vetterling W and Flannery B 2002 *Numerical Recipes* 2nd edn (Cambridge: Cambridge University Press)
- [30] Berg H 1993 *Random Walk in Biology* (Princeton, NJ: Princeton University Press)
- [31] Saffman P and Delbrück M 1975 Brownian motion in biological membranes *Proc. Natl Acad. Sci. USA* **72** 3111–3
- [32] Horn B 1987 Closed-form solution of absolute orientation using unit quaternions *J. Opt. Soc. Am. A* **4** 629
- [33] Westermann S, Wang H-W, Avila-Sakar A, Drubin D, Nogales E and Barnes G 2006 The dam1 kinetochore ring complex moves processively on depolymerizing microtubule ends *Nature* **440** 565–9
- [34] Gillespie D T 2007 Stochastic simulation of chemical kinetics *Annu. Rev. Phys. Chem.* **58** 35–55
- [35] Hattne J, Fange D and Elf J 2005 Stochastic reaction-diffusion simulation with mesord *Bioinformatics* **21** 2923–4
- [36] Carter N J and Cross R A 2005 Mechanics of the kinesin step *Nature* **435** 308–12