Journal of Computational Physics 338 (2017) 452–476

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

Adaptive and iterative methods for simulations of nanopores with the PNP–Stokes equations

Gregor Mitscha-Baude *, Andreas Buttinger-Kreuzhuber, Gerhard Tulzer, Clemens Heitzinger

TU Wien, Wiedner Hauptstrasse 8-10, A-1040 Vienna, Austria

ARTICLE INFO

Article history:
Received 16 August 2016
Received in revised form 31 January 2017
Accepted 28 February 2017
Available online 7 March 2017

Keywords:
Nanopore
Poisson–Nernst–Planck
Stokes
Goal-oriented adaptivity
Electrophoresis

ABSTRACT

We present a 3D finite element solver for the nonlinear Poisson–Nernst–Planck (PNP) equations for electrodiffusion, coupled to the Stokes system of fluid dynamics. The model serves as a building block for the simulation of macromolecule dynamics inside nanopore sensors. The source code is released online at github.com/mitschabaude/nanopores.

We add to existing numerical approaches by deploying goal-oriented adaptive mesh refinement. To reduce the computation overhead of mesh adaptivity, our error estimator uses the much cheaper Poisson–Boltzmann equation as a simplified model, which is justified on heuristic grounds but shown to work well in practice. To address the nonlinearity in the full PNP–Stokes system, three different linearization schemes are proposed and investigated, with two segregated iterative approaches both outperforming a naive application of Newton's method. Numerical experiments are reported on a real-world nanopore sensor geometry.

We also investigate two different models for the interaction of target molecules with the nanopore sensor through the PNP–Stokes equations. In one model, the molecule is of finite size and is explicitly built into the geometry; while in the other, the molecule is located at a single point and only modeled implicitly – after solution of the system – which is computationally favorable. We compare the resulting force profiles of the electric and velocity fields acting on the molecule, and conclude that the point-size model fails to capture important physical effects such as the dependence of charge selectivity of the sensor on the molecule radius.

1. Introduction

Nanopore sensors are biotechnological devices designed to mimic the functionality of ion channels that occur in organic cells. They have shown promise as a tool to detect and analyze single molecules in an electrolyte solution. This has enabled fast and cheap DNA sequencing [4,58] – recently put to practice for surveillance of the Ebola virus in West Africa [54] –, and is also finding applications in protein detection and sequencing [30]. The basic principle of a nanopore sensor is illustrated in Fig. 1. Since nanopore technology is in an experimental state, researchers need to be able to investigate new sensor designs rapidly, and simulations play an important role in this process.

* Corresponding author.
E-mail address: gregor.mitscha-baude@gmx.at (G. Mitscha-Baude).

http://dx.doi.org/10.1016/j.jcp.2017.02.072
0021-9991/

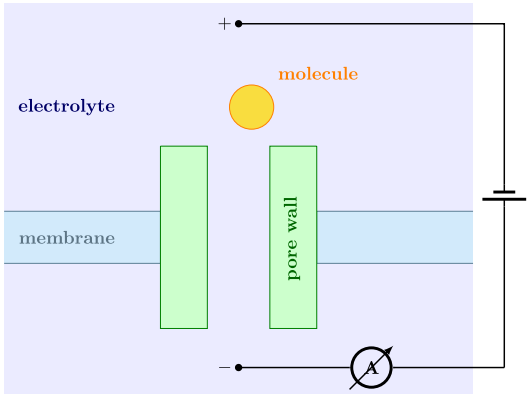


Fig. 1. Schematic of a nanopore sensor. The pore connects two large electrolyte reservoirs (blue) separated by a membrane (cyan). A transmembrane voltage is applied (illustrated symbolically by two electrodes in a circuit diagram), triggering an ionic current through the pore. Target molecules (yellow) can be detected based on modulation of the current as they enter the nanopore. In our simulated geometry, the pore wall (green) is made up of DNA origami [6]. Other possibilities include pore proteins like α-hemolysin [11] or solid-state pores drilled into a silicon nitride membrane [66]. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

The main continuum model for nanopores are the steady-state Poisson–Nernst–Planck (PNP) equations [12,38,45], which capture the electrodiffusion of various ion species in solution. They can be augmented by the Navier–Stokes equation to include effects of electroosmotic flow, resulting in the *Poisson–Nernst–Planck–Stokes (PNPS)* equations, which will be the focus of our work. Pioneered by Rubinstein [56], they have been used to model, for example, lab-on-chip devices [19,18], biological ion channels [65,10] and solid-state nanopores [62,36,26].

Given the importance of the PNPS model in engineering, relatively few papers have addressed the numerical approach in detail. Splitting schemes for the time-dependent formulation have been analyzed in [52,57,49]; but our interest lies in the steady-state distributions which can be obtained more efficiently directly, without time-stepping. Most works related to nanopore modeling rely on a black-box implementation provided by the commercial Comsol Multiphysics package [64,63,43, 40,46]. In the microfluidic community, special-purpose finite difference codes have been shown to give better performance [34]. For our work, however, we prefer the flexibility of unstructured meshes to account for complex real-world nanopore geometries, which are defined, for instance, by the shapes of biological channel proteins like α-hemolysin [60]. Therefore, finite elements will be used for discretization.

Our goal in this work was to create a state-of-the-art PNPS solver that overcomes the limitations of both off-the-shelf commercial solutions and highly structured geometries. The two main contributions to the numerical literature are

- a detailed investigation of linearization schemes for PNPS, and
- a fast adaptive mesh refinement scheme based on goal-oriented error estimation.

The adaptivity part is specifically tailored towards fast and reliable evaluation of the electrophoretic force on particles surrounded by the fluid, which we consider important for nanopore sensor modeling. The novelty compared to previous applications of goal-oriented adaptivity [5,3], besides being the first for PNPS, is that we base the estimator not on the full PDE system but on a simplified model inspired by physics, namely, the linear Poisson–Boltzmann equation. Since solving the latter equation takes only a fraction of the time compared to the full system, this essentially renders the whole iterative mesh refinement procedure a cheap preprocessing step to be performed before the actual simulation. While by computing the estimator for a different equation we necessarily leave the realm of theoretical validity (where sometimes convergence of adaptive methods can be proved rigorously [22,21,8]), numerical results indicate that our approach does work surprisingly well (Section 4.3). The estimator not only qualitatively picks the correct regions for refinement, but also drives down the error in quantities of interest of the full model with the theoretically optimal rate $O(h^2)$.

Our findings on linearization are of general interest for steady-state PNPS related models. To solve the PNPS equations – which form a nonlinear system of seven coupled equations –, we consider three linearization schemes: a monolithic Newton method applied to the whole system, a Gummel-type fixed-point method, and a hybrid scheme where Newton's method is only applied to the PNP part. The fixed-point method uses a novel corrected Poisson equation which lowers iteration count significantly compared to similar approaches in the literature [44]. In our numerical comparisons, the fixed-point and hybrid methods both reveal some strengths and weaknesses, but clearly outperform the Newton method.

A third contribution of this work lies in modeling and is more specifically tied to the simulation of nanopore sensors. For sensor prototyping, solution of the PNPS system is just one building block to obtain the force which drives the transport of target molecules through the sensor. We investigate two different models for the force where the molecule is either finite-sized or point-sized. In the point-size model, the molecule has no influence on its surrounding electric field and fluid flow; this is a strong simplification, since the force on the whole domain can be obtained from a single PNPS solve. Indeed, we find that the two models deviate significantly from each other. However, we show how to calibrate the point-size model using a single evaluation of the finite-size model, to obtain a much better fit.

Accompanying the paper, we release the source code for the solver online.[1] It is written in Python and relies on the finite element package Fenics [41] for various steps in the discretization process, in particular assembling linear systems.

The remainder of the paper is organized as follows. The PNPS equations, including a description of boundary conditions and other modeling aspects are presented in Section 2. We also introduce a 2D version of the system in cylindrical coordinates that can simplify the simulation whenever the geometry is axisymmetric. Section 3 is devoted to numerical methods. Linearization of PNPS is discussed in 3.1, and further details of the numerical solver are given in Section 3.2. In Section 3.3 the goal-oriented adaptivity framework is introduced. We formulate an adaptive algorithm similar to [5,55] which makes use of patch-wise extrapolation of the dual solution. Additionally, we propose a cheaper variant without extrapolation where the error estimator is based directly on the dual solution.

Numerical results are presented in Section 4. The geometry we use for experiments is modeled after the DNA-based nanopore sensor recently published in [6], where we carefully try to replicate the experimental set-up. In the Appendix, additional numerical results can be found which validate our solver against the exact solution on an idealized test problem. We conclude with Section 5.

2. Model

To start with a broad picture of what simulations of a nanopore sensor should achieve, consider the target molecule in Fig. 1, and suppose the sensor is designed to detect exactly this molecule, which is of a certain shape and carries a certain charge. Two basic questions regarding the functionality of our sensor would be, first, whether the molecule can even enter the nanopore, and second, whether it would translocate the pore sufficiently slowly to be detected by measurements of ionic current. Both questions can for instance be addressed using the framework of the Langevin equation [39] for the dynamics of small particles.

In this framework, the specifics of the physical environment (i.e., sensor and molecule) enter through the electrophoretic force $\boldsymbol{F}(x)$ on the target molecule at any given position x. Once the force is available in parametrized form, we can devise algorithms to compute quantities of interest such as mean translocation time and probability. The physical model underlying $\boldsymbol{F}(x)$ is encapsulated in the PNPS equations, which are the focus of our work and are discussed now.

2.1. PNPS equations

The steady-state Nernst–Planck (NP) equations for a 1:1 electrolyte with positive and negative ion concentrations c^+ and c^-, respectively, are given by

$$\nabla \cdot \boldsymbol{j}^{\pm} = 0, \tag{1a}$$

$$\boldsymbol{j}^{\pm} = -D^{\pm}\nabla c^{\pm} \mp \frac{qD^{\pm}}{kT} c^{\pm}\nabla\phi + c^{\pm}\boldsymbol{u}. \tag{1b}$$

The quantity $\boldsymbol{j}^{\pm}$ is called the ionic flux; $D^{\pm}$ is the ion-specific diffusion coefficient; q the elementary charge; ϕ the electric potential; and $\boldsymbol{u}$ the fluid velocity. This differs from the more well-known form of the Nernst–Planck equations [12] by the convective flux term $c^{\pm}\boldsymbol{u}$, which introduces coupling to the Stokes equation; the term describes transport of ions with the surrounding background medium, water. The ion concentrations $c^{\pm}$ are measured in molar units mol/m^3 and will therefore be multiplied by the Faraday constant $F = 96485$ C/mol to give rise to a charge concentration. At present, our solver implements equations (1) for exactly one positive and one negative monovalent ion species. This is, however, a choice made arbitrarily and not a fundamental limitation of the numerical approach; we believe that any number of species and valencies could be handled after small modifications of the implementation.

Notably, we will not assume that the diffusion coefficients $D^{\pm}$ are constant and equal to their bulk value. It has been recognized in the literature that the confining environment of a narrow channel can substantially reduce diffusivity [50,51,17,13,29,37], so we have built this possibility into our solver. In the numerical experiments below, we use a bulk value of $D^{\pm} = 1.9$ nm^2/ns for both ions, which is lowered by a factor of 0.5 inside nanopores. For absolute temperature we use $T = 293$ K.

The Poisson equation for the electric potential ϕ is

$$-\nabla \cdot (\varepsilon\nabla\phi) = \rho_0 + F(c^+ - c^-) \tag{2}$$

with ε denoting the material-dependent electric permittivity and ρ_0 permanent charges present in the system, e.g. in target molecules and on the pore walls. Equivalently, permanent charges located on a surface are incorporated via Neumann boundary or interface conditions.

Finally, the fluid velocity field $\boldsymbol{u}$ and pressure p are determined by the Stokes equations, which read, in conservative form,

[1] https://github.com/mitschabaude/nanopores.

$$-\nabla \cdot \left[\eta \left(\nabla \boldsymbol{u} + \nabla \boldsymbol{u}^T\right) - pI\right] = -F(c^+ - c^-)\nabla\phi, \tag{3a}$$

$$\nabla \cdot \boldsymbol{u} = 0. \tag{3b}$$

Here, $\eta = 10^{-3}$ Pa · s is the viscosity and I is the identity tensor. The right hand side is the force arising from ions which are pushed by the electric field, dragging along nearby water molecules. The nonlinear inertial term $\boldsymbol{u} \cdot \nabla \boldsymbol{u}$ usually present in the Navier–Stokes equations can be neglected, since the Reynolds number in nanopore-related systems is very low.[2]

Taking together (1)–(3), the PNPS system consists of seven equations for the seven unknowns: the electric potential ϕ, the two ion concentrations c^+ and c^-, the three components of the velocity $\boldsymbol{u}$ and the pressure p.

2.2. Geometry and boundary conditions

We solve PNPS on a space domain like that pictured in Fig. 1 on page 453. It consists of an aqueous pore embedded in a solid membrane and possibly a solid sphere inside or near the pore which represents the target molecule. The membrane separates two cylindrical reservoirs of electrolyte which extend the computational domain to about 20 nm in every direction; this has been empirically confirmed large enough to ensure that the physics inside the pore are not disturbed by artificial boundary conditions, by comparing with reservoir sizes of up to 1000 nm.

The Stokes and Nernst–Planck equations for the velocity and ion concentrations are solved only in the fluid part of the domain; the Poisson equation, on the other hand, is solved on the whole domain including membrane and molecule. The pore/membrane may consist of several distinct dielectric materials depending on the exact sensor to be modeled. Details for a concrete application set-up are provided in Section 4.1.

Boundary conditions are specified as follows. For the ions, we fix concentrations on the top and bottom of the computational domain at a bulk value $c^\pm = c_0$ (Dirichlet condition), while on other boundaries we apply a no-flux (Neumann) condition, $n \cdot \boldsymbol{j}^\pm = 0$, which models hard repulsion at the pore walls.

The external electric field is applied by setting the potential to $\phi = 0$ on the top and to a suitable bias such as $\phi = -0.1$ V on the bottom. On the outer barrel of the cylindrical reservoirs, the Neumann condition $\partial_n \phi = 0$ is applied, assuming that sufficiently far from the influence of the pore the potential varies only in the direction perpendicular to the membrane. Interface conditions are used to incorporate charges sitting on the pore walls, i.e. $[\varepsilon \partial_n \phi] = \rho$ where $[\cdot]$ denotes the jump across the interface and ρ the surface charge density. Charges on the target molecule, on the other hand, are incorporated via a volume term ρ_0 on the right hand side of the Poisson equation (2); in this way, the electric force on the molecule can be evaluated consistently by integrating $-\rho_0 \nabla \phi$ over the molecule.

For the fluid velocity, we use the no-slip condition $\boldsymbol{u} = 0$ on fluid–solid interfaces except the molecule and the natural stress-free boundary condition on the outer reservoir boundaries. In addition, we fix the pressure by setting it to zero on the reservoir top. On the molecule boundary, the no-slip condition is $\boldsymbol{u} = \boldsymbol{v}$, where $\boldsymbol{v}$ is the velocity of the moving molecule. Below, we will introduce an approximation of the electrophoretic force that in the PNPS equations allows us to treat the molecule as static, i.e. $\boldsymbol{v} = 0$.

2.3. Axisymmetric reduction

Most nanopores can be modeled to a good approximation as symmetric around the central z axis. In fact, all the 3D geometries we implemented can be generated by rotation of a two-dimensional cross-section about this axis. In such a setting, if the target molecule is either not present at all or sits on the central axis, and has its velocity vector pointing along the central axis, all the equations will possess axial symmetry. By transforming to cylindrical coordinates r, φ, z and using the fact that the solutions do not depend on the angular component φ, the PNPS equations become a system of PDEs in the two variables r, z. This 2D axisymmetric version is much faster to solve and was implemented in addition to the full 3D version to provide validation and rapid calculations for set-ups without off-centered molecules.

The idea of using axial symmetry to reduce computational effort is not new in the context of nanopores. In fact, [43,40,46,47] – which are the publications most similar in scope to this one – all restrict themselves to axisymmetric settings, so the novelty of our implementation rather lies in the fact that we handle the full 3D case as well.

2.4. Physical quantities of interest

Since we want to describe translocation of target molecules through a nanopore, the primary output of our PNPS solver will be the mean force $\boldsymbol{F}$ acting on such a molecule, which we model as a sum of two contributions

$$\boldsymbol{F} = \boldsymbol{F}_{\text{el}} + \boldsymbol{F}_{\text{drag}}. \tag{4}$$

[2] For the typical length scale of about $L = 10^{-8}$ m, velocity of $u = 10^{-2}$ m/s and fluid density of $\rho = 10^3$ kg/m^3, we have

$$\text{Re} = \frac{\rho u L}{\eta} = 10^{-4}.$$

$\boldsymbol{F}_{\text{el}}$ is the electric force induced by the electric field acting on charges on the molecule, and $\boldsymbol{F}_{\text{drag}}$ is the drag force exerted on the molecule surface by the moving fluid.

Finite-sized molecule. For our main model, we build a spherical target molecule into the geometry by excluding its volume from the fluid domain and ensuring that the mesh aligns with the molecule boundaries. Then, the forces are given by

$$\boldsymbol{F}_{\text{el}} = -\int_M \rho_0 \nabla\phi \tag{5}$$

and

$$\boldsymbol{F}_{\text{drag}} = \int_{\partial M} n \cdot \sigma \tag{6}$$

respectively, where M is the part of the domain occupied by the molecule, ρ_0 the molecule surface charge density and $\sigma = \eta\left(\nabla\boldsymbol{u} + \nabla\boldsymbol{u}^T\right) - pI$ the fluid stress tensor.

Note that the drag force in formula (6) is expressed as a surface integral. In practice, we actually evaluate drag component-wise as the volume integral

$$F^i_{\text{drag}} = \int_{\Omega_F} \sigma : \nabla\boldsymbol{z}_i - \boldsymbol{f}\cdot\boldsymbol{z}_i \tag{7}$$

where Ω_F is the fluid domain (excluding the molecule), $\boldsymbol{f} = -F(c^+ - c^-)\nabla\phi$ the volume force and $\boldsymbol{z}_i$ any vector field that satisfies $\boldsymbol{z}_i|_{\partial M} = \boldsymbol{e}_i$ where $\boldsymbol{e}_i$ is the i-th unit vector. This representation is equivalent to (6) if $\boldsymbol{u}$ and p solve the Stokes equation exactly, as can be shown via integration by parts. Formula (7) was reported by others [5] and observed by ourselves to yield a more accurate numerical approximation. For $\boldsymbol{z}_i$ we use a piece-wise linear interpolation of $\boldsymbol{e}_i$ on the molecule boundary and zero elsewhere.

Point-sized molecule. If the PNPS equations are solved on a geometry *without* target molecule, we can still obtain an estimate for $\boldsymbol{F}_{\text{el}}$ and $\boldsymbol{F}_{\text{drag}}$ on every given position of the computational fluid domain. In this case, we neglect the influence that the charge, velocity and geometrical presence of the molecule has on the physical environment, and assume the molecule to be located at a single point. The electric force is then given by

$$\boldsymbol{F}^*_{\text{el}}(x) = -Q\,\nabla\phi(x) \tag{8}$$

where x is the molecule position and Q the total charge on the molecule. The drag force is more difficult, since it would actually vanish for a point-sized molecule, but we still want a reasonable estimate for a spherical molecule of finite radius $r > 0$. We approximate it by Stokes' law

$$\boldsymbol{F}^*_{\text{drag}}(x, \boldsymbol{v}) = -6\pi\eta r(\boldsymbol{u}(x) + \boldsymbol{v}), \tag{9}$$

where $\boldsymbol{v}$ is the molecule velocity. Because of confinement inside the pore, we anticipate that using equation (9) could lead to an underestimation of the drag force [51]. This is confirmed in Section 4.5, where the discrepancy between the two models $\boldsymbol{F}$ and $\boldsymbol{F}^* := \boldsymbol{F}^*_{\text{el}} + \boldsymbol{F}^*_{\text{drag}}$ (finite-sized and point-sized molecule) is investigated. We shall also elaborate on possible improvements to the point-size model.

2.5. Velocity iteration and linear-velocity approximation

We turn to the question of how our solver and the computation of the electrophoretic force can be used to extract information about target molecule motion. So far, position and velocity of the molecule were assumed to be *given*. A simple approach used in [53,31,1] is to let the velocity be *unknown* and determine it from the condition that the net force on the molecule be zero, $\boldsymbol{F} = \boldsymbol{F}(x, \boldsymbol{v}) = 0$ (*force balance condition*). From this perspective, the electrophoretic force is not a meaningful output of the PNPS solver (being zero); rather, the unknown molecule velocity $\boldsymbol{v}$ becomes the main output quantity.

Note that in our point-size model (8)–(9), the force balance condition is easily solvable as there is a simple linear relationship between force and velocity; i.e., we have $\boldsymbol{F}^*(x, \boldsymbol{v}) = \boldsymbol{F}^*(x, 0) - 6\pi\eta r\boldsymbol{v} = 0$ which gives

$$\boldsymbol{v} = (6\pi\eta r)^{-1}\boldsymbol{F}^*(x, 0).$$

However, in our main finite-size model, $\boldsymbol{v}$ enters the PNPS system in a nonlinear fashion, so it has to be determined iteratively. We implement such an iterative approach but show later that it can be replaced by a simple direct formula, because the dependence of the force on the velocity turns out to be *approximately* linear, even in the finite-size model. The direct formula we propose is also of the form

$$\boldsymbol{v} = \gamma(x)^{-1}\boldsymbol{F}(x, 0),$$

where $\gamma(x)$ is now a more general *friction tensor*.

First, let us introduce an iterative method to solve $\boldsymbol{F} = 0$ for $\boldsymbol{v}$ which only needs evaluations of $\boldsymbol{v} \mapsto \boldsymbol{F}(x, \boldsymbol{v})$, i.e. the PNPS solver, and an even simpler Stokes-only solver. The method hinges on a proposed approximate splitting of the force,

$$\boldsymbol{F}(x, \boldsymbol{v} + \Delta\boldsymbol{v}) \approx \boldsymbol{F}(x, \boldsymbol{v}) - \gamma(x)\Delta\boldsymbol{v}. \tag{10}$$

The second term is the *exact* drag force on the molecule resulting from the pure Stokes equation

$$-\nabla \cdot \left[\eta\left(\nabla\boldsymbol{u} + \nabla\boldsymbol{u}^T\right) - pI\right] = 0, \quad \nabla \cdot \boldsymbol{u} = 0, \quad \boldsymbol{u} = \Delta\boldsymbol{v} \text{ on molecule boundary.} \tag{11}$$

For this equation the drag force depends linearly on the boundary condition, hence can be written in the form $-\gamma(x)\Delta\boldsymbol{v}$ where $\gamma(x)$ is a 3×3 tensor, the friction tensor. The full tensor can be determined by solving (11) for three different (linearly independent) boundary velocities $\Delta\boldsymbol{v}$. Note that the split (10) is not exact, because in the PNPS system a small additional velocity on the molecule introduces a slight change of the convective term in the Nernst–Planck equations (1) which feeds back nonlinearly to the Stokes equation (3) via the force term. The reasoning behind our approximation is that this nonlinear feedback can be neglected.

Using (10) leads to an iterative algorithm for solving $\boldsymbol{F} = 0$.

Algorithm 1 Velocity iteration for force balance condition.

Given a molecule position x, first determine the friction tensor $\gamma(x)$ by solving the pure Stokes system (11) three times with $\Delta\boldsymbol{v} = \boldsymbol{e}_1,\ \boldsymbol{e}_2,\ \boldsymbol{e}_3$. Set the initial velocity to $\boldsymbol{v} = 0$. Then,

1. Calculate $\boldsymbol{F} = \boldsymbol{F}(x, \boldsymbol{v})$ by solving PNPS.
2. If $|\boldsymbol{F}|$ is small enough, stop.
3. Set $\Delta\boldsymbol{v} = \gamma(x)^{-1}\boldsymbol{F}(x, \boldsymbol{v})$.
4. Update $\boldsymbol{v} = \boldsymbol{v} + \Delta\boldsymbol{v}$ and go back to 1.

We find in Section 4.4 that already in the second iteration, the force $|\boldsymbol{F}|$ is very small. This suggests that we even have

$$\boldsymbol{F}(x, \boldsymbol{v}) \approx \boldsymbol{F}(x, 0) - \gamma(x)\boldsymbol{v}, \tag{12}$$

similar to the linear relationship in the point-size model. We call this the *linear velocity approximation*. In Section 4.4 we demonstrate that this approximation is very accurate and can be applied also for relatively large velocities. With this result, the force on a molecule standing still, $\boldsymbol{F}(x, 0)$, turns out to be a meaningful quantity even if the force balance condition $\boldsymbol{F}(x, \boldsymbol{v}) = 0$ is to be applied, because it is directly related to the velocity via the friction tensor: $\boldsymbol{v} \approx \gamma(x)^{-1}\boldsymbol{F}(x, 0)$.

A further reason why the linear split (12) is so desirable comes up when one wants to go beyond the simple force balance condition. As is well known, small particles suspended in water move in an irregular fashion caused by many seemingly random collisions with water molecules. This is not accounted for in the force balance condition, where the velocity is determined solely by the particle position, yielding a mean drift velocity. In reality, the drift velocity is only a statistical property; a target molecule *always* moving at drift velocity could never overcome a non-zero energy barrier, for instance needed to enter a channel with charge of the same sign as the molecule. The appropriate model to account for stochastic variations of the velocity is the Langevin equation [39,9,38], which we write in a general form as

$$m\ddot{x} = \boldsymbol{F}(x, \dot{x}) + \boldsymbol{F}_{\text{rand}}.$$

This is just Newton's second law with the force split into a deterministic force $\boldsymbol{F}(x, \dot{x})$ depending on position x and velocity $\boldsymbol{v} = \dot{x}$, and a stochastic force resulting from collisions. In the overdamped limit the inertial term $m\ddot{x}$ is neglected [38,13], yielding

$$0 = \boldsymbol{F}(x, \dot{x}) + \boldsymbol{F}_{\text{rand}}. \tag{13}$$

Thus, we get a similar force balance condition with an additional random term. In a Brownian dynamics simulation of the target molecule trajectory, (13) would have to be solved for $\dot{x}$ at every timestep (with a randomly drawn force vector), which is computationally prohibitive. However, with the linear velocity approximation $\boldsymbol{F}(x, \dot{x}) = \boldsymbol{F}(x, 0) - \gamma(x)\dot{x}$, equation (13) simplifies to

$$\dot{x} = \gamma(x)^{-1}(\boldsymbol{F}(x, 0) + \boldsymbol{F}_{\text{rand}}).$$

This is the form of the Langevin equation usually employed for Brownian dynamics. The advantage is that $\boldsymbol{F}(x, 0)$ and $\gamma(x)$ depend only on the position and, in particular, *not* on the randomly drawn value of the collision force. Thus, we can store $\boldsymbol{F}(x, 0)$ and $\gamma(x)$ for many points of the domain in a lookup table, prior to the Brownian dynamics simulations that can then proceed rapidly. With this application in mind, we will adopt the linear velocity approximation for most of the remaining paper. The focus will therefore be on accurately evaluating the force on a particle with zero velocity.

2.6. The Poisson–Boltzmann approximation

Close to equilibrium, ion concentrations are well approximated by the Boltzmann factors

$$c^{\pm} = c_0 e^{\mp \frac{q}{kT}\phi}$$

where c_0 is the bulk concentration of both ion species. For zero voltage bias, this is the exact solution to the Nernst–Planck equations (1) without the convective flux term $c^{\pm}\boldsymbol{u}$ (in this case, $\boldsymbol{j}^{\pm} = 0$ and the boundary condition $c^{\pm} = c_0$ is matched exactly if the potential is zero at the boundary). But even for low to moderate biases and with convective flux included, Boltzmann factors provide a useful estimate for the ion distributions near charged walls. By plugging the Boltzmann distribution into the Poisson equation (2), one obtains the Poisson–Boltzmann (PB) equation

$$-\nabla \cdot (\varepsilon \nabla \phi) + \chi_F 2qc_0 \sinh \frac{q\phi}{kT} = \rho_0,$$

where χ_F denotes the characteristic function of the fluid domain. By using the linearization $\sinh(z) \approx z$, we arrive at the *linear* Poisson–Boltzmann equation

$$-\nabla \cdot (\varepsilon \nabla \phi) + \chi_F \frac{2q^2 c_0}{kT}\phi = \rho_0. \tag{14}$$

This gives us a simplified model for the electric potential. In Section 3.3, we construct an error indicator solely based on solutions to the linear PB equation, which is orders of magnitude cheaper than solutions of the full PNPS model. Thus, we use equation (14) for mesh pre-refinement before the actual PNPS simulation. The linearity also makes it fit better into the framework of goal-oriented adaptivity.

In addition, the PB approximation can serve as initial guess for Newton iteration of the PNP equations. This helps with convergence issues caused by high surface charges, as shown in Section 4.2.

3. Numerical method

We use a finite element discretization for all the equations of the PNPS system, which was implemented in Python on top of the open-source finite element package Fenics [41]. It allows us to easily handle unstructured meshes on irregular geometries and a wide variety of variational formulations and boundary conditions.

3.1. Linearization of PNPS

The PNPS system is nonlinear in the potential ϕ, concentrations $c^{\pm}$ and velocity $\boldsymbol{u}$ and therefore has to be solved by some iterative linearization procedure. These procedures can be distinguished by whether they use Newton's method, or fixed-point iteration, or a combination. Along these lines we will describe three different methods for the PNPS system. They will be compared numerically in Section 4.2.

Newton method. The first, black box approach to linearization is to apply Newton's method to the whole system. Let us write our nonlinear equation in abstract variational form,

$$\mathcal{F}(U) = 0,$$

where $U = (\phi, c^+, c^-, \boldsymbol{u}, p)$ represents the finite element solution and $\mathcal{F}(U)$ is a linear functional on the discrete test space. Then, Newton's method consists in successive linear solves of

$$D\mathcal{F}(U)\delta U = -\mathcal{F}(U), \tag{15}$$

followed by updates $U = U + \delta U$. As initial value, for instance, $U = (\phi, c^+, c^-, \boldsymbol{u}, p) = (0, c_0, c_0, 0, 0)$ can be used. Written out, equation (15) amounts to solving the system

$$-\nabla \cdot (\varepsilon \nabla \delta\phi) - F(\delta c^+ - \delta c^-) = \rho_0 + \nabla \cdot (\varepsilon \nabla \phi) + F(c^+ - c^-), \tag{16a}$$

$$\nabla \cdot \left[-D^+ \nabla \delta c^+ - \mu^+ \delta c^+ \nabla \phi + \delta c^+ \boldsymbol{u} - \mu^+ c^+ \nabla \delta\phi + c^+ \delta \boldsymbol{u}\right] = \nabla \cdot \left[D^+ \nabla c^+ + \mu^+ c^+ \nabla \phi - c^+ \boldsymbol{u}\right], \tag{16b}$$

$$\nabla \cdot \left[-D^- \nabla \delta c^- + \mu^- \delta c^- \nabla \phi + \delta c^- \boldsymbol{u} + \mu^- c^- \nabla \delta\phi + c^- \delta \boldsymbol{u}\right] = \nabla \cdot \left[D^- \nabla c^- - \mu^- c^- \nabla \phi - c^- \boldsymbol{u}\right], \tag{16c}$$

$$-\eta \Delta \delta \boldsymbol{u} + \nabla \delta p + F(\delta c^+ - \delta c^-)\nabla \phi + F(c^+ - c^-)\nabla \delta\phi = \eta \Delta \boldsymbol{u} - \nabla p - F(c^+ - c^-)\nabla \phi, \tag{16d}$$

$$\nabla \cdot \delta \boldsymbol{u} = -\nabla \cdot \boldsymbol{u}, \tag{16e}$$

which is a linearized version of PNPS in the unknowns $\delta U = (\delta\phi, \delta c^+, \delta c^-, \delta \boldsymbol{u}, \delta p)$. Here we use $\mu^{\pm} := \frac{qD^{\pm}}{kT}$ for readability. We will refer to this algorithm as the *(full) Newton method*:

Algorithm 2 Newton method for PNPS.

Initialize $U = (\phi, c^+, c^-, \boldsymbol{u}, p)$. Then:

1. Solve (16) for the Newton update $\delta U = (\delta\phi, \delta c^+, \delta c^-, \delta\boldsymbol{u}, \delta p)$.
2. Update $U = U + \delta U$.
3. Let the relative error be defined by

$$\text{error} := \frac{\|\delta U\|_{L^2(\Omega)}}{\|U\|_{L^2(\Omega)}}.$$

Check convergence (with tolerance $\tau > 0$):
(a) If error $< \tau$, stop.
(b) Else, go back to 1.

Hybrid method. The full Newton method yields a single large system for linear solving. We suspected that computational effort could be saved by separating the equations into two smaller subsets, namely the PNP and Stokes systems, and solving them in an alternating fashion until convergence is reached, i.e., by a fixed-point iteration. Thus, the potential ϕ and concentrations $c^\pm$ will be obtained from the PNP part of the system, with the initial velocity set to zero. Then, Stokes will be solved for the velocity $\boldsymbol{u}$ and pressure p, with ϕ, c^+, c^- as input; the velocity will now be plugged into PNP, etc.

In this formulation, the Stokes part of the system is linear, but we still need to address the remaining nonlinearity in the PNP part (namely, the products $c^\pm\nabla\phi$). We could again simply apply Newton's method to the PNP system only, or use an inner fixed-point loop. Both methods are well-established in the context of the PNP equations without velocity term [44,42]; however, the fixed-point method is usually reported to require some form of under-relaxation, resulting in several hundreds of iterations until convergence [44].

In practice, we initially experimented with a naive, unrelaxed fixed-point iteration for PNP but found it to diverge in every case considered. Newton iteration, applied to the PNP part only, was seen to be quite robust, even with the constant initial guess $\phi = 0$, $c^\pm = c_0$; in situations where it does not converge either, resources such as a better initial guess and under-relaxation (damped Newton) can still be used. Regarding the fixed-point iteration between PNP and Stokes, convergence was always observed; the coupling between these two systems, which stems from the convective flux term $c^\pm\boldsymbol{u}$ in the Nernst–Planck equation (1), seems to be sufficiently weak.

Taking this into consideration, the second method we propose consists of an outer fixed-point loop decoupling the system into a PNP and a Stokes part, and an inner Newton iteration to solve the PNP part. As a further refinement, it was found beneficial not to solve the PNP equations to full convergence in every step, but only to perform a single Newton iteration for every linear Stokes solve. We call this version of the algorithm the *hybrid method*:

Algorithm 3 Hybrid method for PNPS.

Initialize $U_{\text{PNP}} = (\phi, c^+, c^-)$ and $U_{\text{Stokes}} = (\boldsymbol{u}, p)$. The Newton update equation for the PNP system is

$$-\nabla\cdot(\varepsilon\nabla\delta\phi) - F(\delta c^+ - \delta c^-) = \rho_0 + \nabla\cdot(\varepsilon\nabla\phi) + F(c^+ - c^-), \tag{17a}$$

$$\nabla\cdot\left[-D^+\nabla\delta c^+ - \mu^+\delta c^+\nabla\phi + \delta c^+\boldsymbol{u} - \mu^+ c^+\nabla\delta\phi\right] = \nabla\cdot\left[D^+\nabla c^+ + \mu^+ c^+\nabla\phi - c^+\boldsymbol{u}\right], \tag{17b}$$

$$\nabla\cdot\left[-D^-\nabla\delta c^- + \mu^-\delta c^-\nabla\phi + \delta c^-\boldsymbol{u} + \mu^- c^-\nabla\delta\phi\right] = \nabla\cdot\left[D^-\nabla c^- - \mu^- c^-\nabla\phi - c^-\boldsymbol{u}\right]. \tag{17c}$$

Now,

1. Solve (17) for the Newton update $\delta U_{\text{PNP}} = (\delta\phi, \delta c^+, \delta c^-)$.
2. Update $U_{\text{PNP}} = U_{\text{PNP}} + \delta U_{\text{PNP}}$.
3. Save $U_{\text{Stokes},0} = U_{\text{Stokes}}$ from the last iteration.
   Solve the Stokes equation (3) for U_{Stokes}.
4. Set $\delta U_{\text{Stokes}} = U_{\text{Stokes}} - U_{\text{Stokes},0}$. Let the relative error be defined by

$$\text{error} := \frac{1}{2}\left(\frac{\|\delta U_{\text{PNP}}\|_{L^2(\Omega)}}{\|U_{\text{PNP}}\|_{L^2(\Omega)}} + \frac{\|\delta U_{\text{Stokes}}\|_{L^2(\Omega)}}{\|U_{\text{Stokes}}\|_{L^2(\Omega)}}\right).$$

Check convergence (with tolerance $\tau > 0$):
(a) If error $< \tau$, stop.
(b) Else, go back to 1.

Fixed-point method. Regardless of initial failure, we still wanted to see if a fast fixed-point solution was possible. We took inspiration from a practice common in the semiconductor community, which is to introduce the so-called Slotboom variables [59]

$$\tilde{c}^\pm = c^\pm e^{\pm\frac{q}{kT}\phi}.$$

For motivation, note that in equilibrium – when the Boltzmann distribution applies – $\tilde{c}^{\pm}$ reduce to constants. Plugging these new variables into the classical Nernst–Planck equations (without velocity) renders them in purely self-adjoint form, which is usually emphasized [44]. But, more importantly for us, it also alters the form of the Poisson equation:

$$-\nabla \cdot (\varepsilon \nabla \phi) = F(\tilde{c}^{+} e^{-\frac{q}{kT}\phi} - \tilde{c}^{-} e^{\frac{q}{kT}\phi}).$$

Note that the potential ϕ now appears also on the right hand side, in the charge distribution term. The Poisson equation becomes nonlinear in the potential. If this would be solved in a fixed-point loop, where $\tilde{c}^{\pm}$ are taken from the last iteration, the new potential would not only give rise to a new charge distribution, but would already reflect part of the impact this change has on itself. This is just the correction that is needed to prevent the fixed-point iteration from exploding. In semiconductor device modeling, this type of iteration is known as *Gummel's method* [28,35].

With this new insight, we can actually change back to the old variables $c^{\pm}$, by using $\tilde{c}^{\pm} = c^{\pm} e^{\pm \frac{q}{kT}\phi_0}$ with ϕ_0 being the potential from the last iteration, to find

$$-\nabla \cdot (\varepsilon \nabla \phi) = F(c^{+} e^{-\frac{q}{kT}(\phi-\phi_0)} - c^{-} e^{\frac{q}{kT}(\phi-\phi_0)}).$$

In the limit of convergence of the fixed-point iteration we have $\phi = \phi_0$ and the exponential terms vanish. Since those are artificial correction terms anyway, we might as well make our life easier and linearize them as

$$e^{\pm \frac{q}{kT}(\phi-\phi_0)} \approx 1 \pm \frac{q}{kT}(\phi - \phi_0).$$

Reordering, we arrive at a corrected Poisson equation which is again linear in the unknown ϕ and features the previous potential ϕ_0 on the right hand side:

$$-\nabla \cdot (\varepsilon \nabla \phi) + \frac{Fq}{kT}\phi(c^{+} + c^{-}) = F(c^{+} - c^{-}) + \frac{Fq}{kT}\phi_0(c^{+} + c^{-}). \tag{18}$$

Alternating the corrected Poisson equation (18) with the unmodified Nernst–Planck and Stokes equations yields our proposed *fixed-point method*, summarized in Algorithm 4. To ensure convergence of the method, we recommend to start with two pure PNP iterations and only afterwards include the Stokes equation in the iteration.

Algorithm 4 Fixed-point method for PNPS.

Initialize ϕ, c^{+}, c^{-} and $U_{\text{Stokes}} = (\boldsymbol{u}, p)$. Set the iteration count to $i = 0$. Then:

1. Save the solutions from the previous iteration: $\phi_0 = \phi$, $c_0^{+} = c^{+}$, etc.
2. Solve the corrected Poisson equation (18) for ϕ.
3. Solve both the Nernst–Planck equations (1) for c^{+}, c^{-}.
4. If $i < 2$, set $i = i + 1$ and go back to 1.
5. Solve the Stokes equation (3) for U_{Stokes}.
6. Define the updates $\delta\phi = \phi - \phi_0$, etc. Let the relative error be defined by

$$\text{error} := \frac{1}{4}\left(\frac{\|\delta\phi\|_{L^2(\Omega)}}{\|\phi\|_{L^2(\Omega)}} + \frac{\|\delta c^{+}\|_{L^2(\Omega)}}{\|c^{+}\|_{L^2(\Omega)}} + \frac{\|\delta c^{-}\|_{L^2(\Omega)}}{\|c^{-}\|_{L^2(\Omega)}} + \frac{\|\delta U_{\text{Stokes}}\|_{L^2(\Omega)}}{\|U_{\text{Stokes}}\|_{L^2(\Omega)}}\right).$$

Check convergence (with tolerance $\tau > 0$):
(a) If error $< \tau$, stop.
(b) Else, go back to 1.

3.2. Details of the numerical solver

Variational formulations for the PNPS related systems were obtained by integration by parts of equations (1)–(3), (11), (14), (16)–(18) respecting their conservative structure; this was done for both the full 3D and 2D axisymmetric cases. The formulation of the Stokes system in cylindrical coordinates can be found in [15]. Dirichlet boundary conditions are enforced strongly, while Neumann conditions are incorporated naturally via boundary terms on the right hand side. For subequations of the PNP system, a standard P^1 finite element discretization is employed, while for the Stokes system, the consistent stabilized $P^1 - P^1$ formulation by Hughes and Franca [32] is used to ensure inf-sup stability; for some of the numerical experiments below, we had to switch to a (more expensive) Taylor–Hood $P^2 - P^1$ formulation for Stokes to enforce higher numerical accuracy.

In the 2D case, we always use a direct algebraic solver based on LU decomposition. This quickly turned out to introduce a memory and speed bottleneck in the 3D case, so we started experimenting with iterative solvers based on Krylov subspace methods for both the PNP and Stokes subsystems. The following choices are used subsequently for the 3D solver with the hybrid method: The PNP system ((17)) and PB equation (14), which are elliptic, are solved with BiCGstab [61] preconditioned by an incomplete LU factorization, and the Stokes system with the transpose-free quasi-minimal residual (TFQMR) method

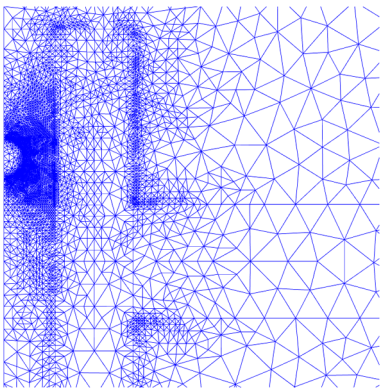


Fig. 2. Refined mesh produced by the goal-oriented adaptive algorithm.

[23] preconditioned by the block preconditioner by Mardal and Winther [48]. For both solvers, the Fenics interface to PETSc [2] was employed, with the ILU implementation taken from the Hypre package [20].

For the PNP system, the iterative approach drastically improves computational speed in 3D even for moderately large meshes. For the Stokes system however, a large number (hundreds to thousands) of iterations are needed which renders the iterative approach quite slow, but at least the memory barrier is removed. Further research will have to shed light on constructing the right preconditioner for the Stokes system in a nanopore context, and explain why an established method [48] seems to fail here.

Meshes were generated with Gmsh [25] and the Python–Gmsh interface written by Schlömer.[3] Curved boundaries, as present in all our pore geometries, were approximated by polygons. For mesh refinement, we modified the implementation in Fenics so that refined meshes respect the curved geometry.

3.3. Goal-oriented adaptivity

Solutions to the PNPS system generally exhibit multiscale features such as thin boundary layers near a charged surface. Overall simulation accuracy depends on whether these layers are resolved by a sufficiently fine mesh. However, since the main output of the simulation are scalar quantities of interest like the force on a target molecule (Section 2.4), the mesh has to be fine only in the locations that directly influence this quantity. Especially in 3D, the simulation time can quickly explode and get intractable if mesh refinement is applied too broadly.

Fig. 2 shows an example mesh on an axisymmetric pore geometry that fulfills these objectives: strong refinement at the boundary layers with emphasis on the positions around the molecule, but a coarse mesh outside the pore and far from the molecule. Clearly, such an approach to mesh refinement would be cumbersome to implement by hand and difficult to generalize. The method which accomplishes this automatically is adaptive refinement based on goal-oriented error estimation [5,3], and will be described in this section.

General framework. To present the general ideas of goal-oriented adaptivity, suppose we want to solve a linear PDE in variational form

$$a(u, v) = L(v). \tag{19}$$

Goal-oriented means that we are interested in a quantity $F(u) \in \mathbb{R}$, where F is a functional on the solution space and assumed to be linear. The Galerkin discretization of (19) yields a solution u_h in a discrete subspace, and we want to refine the mesh such that the error in the goal functional compared to the true solution is small, i.e. decrease $|F(u - u_h)|$. To analyze this error we introduce the *dual solution* w which solves

$$a(v, w) = F(v). \tag{20}$$

Note that the arguments in the bilinear form have been swapped and the right hand side is now the goal functional. Equations (19) and (20) can be combined to give an error representation:

$$F(u - u_h) = a(u - u_h, w) = L(w) - a(u_h, w) =: R(w).$$

The right-hand side is simply the residual of the Galerkin problem, evaluated at the dual solution w, and can in principle be computed once we have an approximation w_h of w. Care must be taken at this point, however, because choosing w_h from the same discrete space as u_h yields $R(w_h) = 0$ due to Galerkin orthogonality, which is hardly a useful error estimate.

[3] https://github.com/nschloe/pygmsh.

One possible solution is to extrapolate w_h patch-wise to a higher-order function space [55,3]. In the case when w_h and u_h are P^1 and the extrapolation Ew_h is P^2, for example, one can expect that [3]

$$F(u-u_h) = R(w) = O(h^2) \quad \text{while} \quad R(Ew_h) = R(w) + O(h^3).$$

This shows that $F(u-u_h) \approx R(Ew_h)$ produces an accurate estimate of the true error. A local error indicator is obtained by expanding the residual into a sum of local contributions,

$$R(w) = \sum_T R_T(w) + R_{\partial T}(w).$$

The local terms can be found by integration by parts on every mesh element T. Their absolute values

$$\eta_T := \left| R_T(\tilde{w}) + R_{\partial T}(\tilde{w}) \right| \tag{21}$$

are the element-wise error indicators. Here we will use $\tilde{w} = Ew_h - w_h$ to make the residuals as small and the upper bound $|F(u-u_h)| \leq \sum \eta_T$ as tight as possible. The error indicators η_T (21) are used to decide which mesh elements will be refined, as outlined in the following classical adaptive algorithm.

Algorithm 5 Adaptive algorithm.

Starting with an initial coarse mesh, do:

1. Solve the primal (19) and dual (20) problems for u_h, w_h.
2. Extrapolate w_h to a higher-order approximation Ew_h.
3. Compute the error indicators (21).
4. Produce a refined mesh by bisecting the elements with the highest indicators.
5. Based on some criterion, either stop or go back to 1.

The patch-wise extrapolation in step 2 is done using the Fenics implementation detailed by Rognes and Logg [55]. The number of elements to be refined in step 4 is determined by Dörfler marking [16]. As for the termination criterion in step 5, we will usually stop after a critical amount of elements is reached corresponding to a maximal work load.

Faster error estimation without extrapolation. In our implementation of the adaptive Algorithm 5, the most computationally expensive step is extrapolation of the dual solution to a higher-order function space. We expect this observation to hold independently of the precise extrapolation method, due to the blow-up of degrees of freedom when switching from a P^1 to a P^2 representation. As discussed, the extrapolation is done to avoid Galerkin orthogonality, improving the useless global estimate $F(u-u_h) \approx R(w_h) = 0$ to an estimate of (sometimes provable) high accuracy $F(u-u_h) \approx R(Ew_h)$. However, that this also leads to local error indicators of high quality remains largely a heuristic, according to current wisdom about adaptive methods [22].

In any case, we will apply error estimation only in an approximate way to a simplified model (explained below). One consequence is that the *global* error estimate we would get from the adaptive algorithm has no relevance to our actual quantities of interest. Once we dispense with a global error estimate and only want to compute local indicators, there is no longer a clear theoretical motivation to use the extrapolated dual solution Ew_h rather than w_h. Consequently, we propose to exchange the indicators (21) for

$$\eta_T^{\text{cheap}} := \left| R_T(w_h) + R_{\partial T}(w_h) \right|. \tag{22}$$

Thus, we simply skip the extrapolation step and calculate our error indicators directly from the dual Galerkin solution. This leads to the following alternative adaptive Algorithm 6, which is both faster and simpler to implement than Algorithm 5.

Algorithm 6 Cheap adaptive algorithm without extrapolation.

Starting with an initial coarse mesh, do:

1. Solve the primal (19) and dual (20) problems for u_h, w_h.
2. Compute the cheap error indicators (22).
3. Produce a refined mesh by bisecting the elements with the highest indicators.
4. Based on some criterion, either stop or go back to 1.

Interestingly, we are not aware of any publication that makes use of this simple idea, although it has also been proposed in the unpublished work [33]. The performance with respect to quantities of interest for our particular problem remains to be examined in Section 4.3.

Adaptivity for the Poisson–Boltzmann equation. Because two systems of equations have to be solved in every iteration of Algorithms 5 and 6, running it for the full PNPS model would be costly. Besides, as a nonlinear system it does not fit neatly

into the framework sketched above (but see [3] for goal-oriented adaptivity in the nonlinear case). However, we expect a good deal of the error information to be already contained in a simplified physical model, namely, the linear PB equation. In our simulations, we will therefore proceed as follows:

Step 1. Refine the mesh several times by goal-oriented adaptivity for the linear PB equation (14).
Step 2. Solve the PNPS equations (1)–(3) once on the final mesh.

When we apply the general adaptivity framework to the linear PB equation, the bilinear form becomes

$$a(\phi, \psi) = \int_\Omega \varepsilon \nabla\phi \cdot \nabla\psi + \int_\Omega \kappa\phi\psi$$

with $\kappa := \chi_F \frac{2q^2 c_0}{k_B T}$ and χ_F the characteristic function of the fluid domain. The right hand side stems from a composition of volume and surface charges

$$L(\psi) = \int_\Omega \rho_0 \psi + \int_\Gamma \rho\psi$$

with Γ denoting all charged interfaces, e.g., pore walls. Both trial and test functions are restricted to vanish on the Dirichlet part of $\partial\Omega$. That is, even if the full PNPS model is specified with non-zero voltage bias, we always solve the auxiliary PB equation with zero voltage bias, since applying a non-zero bias would violate the assumptions underlying the PB model, leading to an unphysical, exponential behavior of the potential at the Dirichlet boundary and an overestimation of error near this boundary.

The important remaining question is how to choose the goal functional. Ideally, our quantity of interest would be the force $\boldsymbol{F}$ on the target molecule (4), but computing it requires the full PNPS solution. Still, we may find a functional of the linear PB potential that roughly preserves the properties of the PNPS force, such as higher relevance of the region near the molecule. We aim to achieve this by defining

$$F(\phi) := -\int_M \rho_0 \partial_z \phi \tag{23}$$

as our goal functional; M is the molecule region and ρ_0 its charge density. This is a direct analogue of the electrical force (5) from the PNPS model; the electric field $-\nabla\phi$ is restricted to the z-direction because we need a scalar-valued functional. The hydrodynamic contribution to the force is neglected for lack of a reasonable analogue in the PB model.

It has to be emphasized that (23) shall *not* provide a quantitative estimate of the actual force $\boldsymbol{F}$ or its electrical part $\boldsymbol{F}_{\text{el}}$. The only purpose of the goal functional is to guide the adaptive mesh refinement process, and we must hope that the parts of the domain where the discretization error $F(\phi - \phi_h)$ is high are the same parts where also the error in the actual quantity of interest $\boldsymbol{F}$ would be high. That this is indeed the case is verified experimentally in Section 4.3.

With these specializations of the general goal-adaptive framework, let w denote the dual solution as before. The local error contributions can be obtained from the residual $R(w)$ via element-wise integration by parts by calculating

$$\begin{aligned} R(w) = -a(\phi_h, w) + L(w) &= \sum_T -\int_T (\varepsilon\nabla\phi \cdot \nabla w + \kappa\phi w) + \int_T \rho_0 w + \frac{1}{2}\int_{\partial T} \rho w \\ &= \sum_T \int_T \left(\varepsilon\nabla^2\phi - \kappa\phi + \rho_0\right) w + \frac{1}{2}\int_{\partial T} \left(\rho - [\varepsilon\partial_n\phi]\right) w \\ &=: \sum_T (r_T, w)_T + (r_{\partial T}, w)_{\partial T}. \end{aligned}$$

The strong cell and facet residuals are given by

$$\begin{aligned} r_T &= \varepsilon\nabla^2\phi - \kappa\phi + \rho_0, \\ r_{\partial T} &= \frac{1}{2}\left(\rho - [\varepsilon\partial_n\phi]\right), \end{aligned}$$

where $[\varepsilon\partial_n\phi]$ denotes the jump over a facet in normal direction. Here, ρ is understood to be zero on uncharged facets, and on boundary facets $[\varepsilon\partial_n\phi] = \varepsilon\partial_n\phi$. The error indicators η_T, defined abstractly in (21) and (22), can now be explicitly written as

$$\eta_T = \left|(r_T, \tilde{w})_T + (r_{\partial T}, \tilde{w})_{\partial T}\right|$$

with either $\tilde{w} = (Ew_h - w_h)$ or $\tilde{w} = w_h$. Goal-oriented adaptivity is implemented according to Algorithms 5 and 6. Once the adaptive loop has produced a mesh sufficiently resolving the regions of interest, the final Poisson–Boltzmann solution is used as an initial guess for the iteration solving the nonlinear PNPS equations.

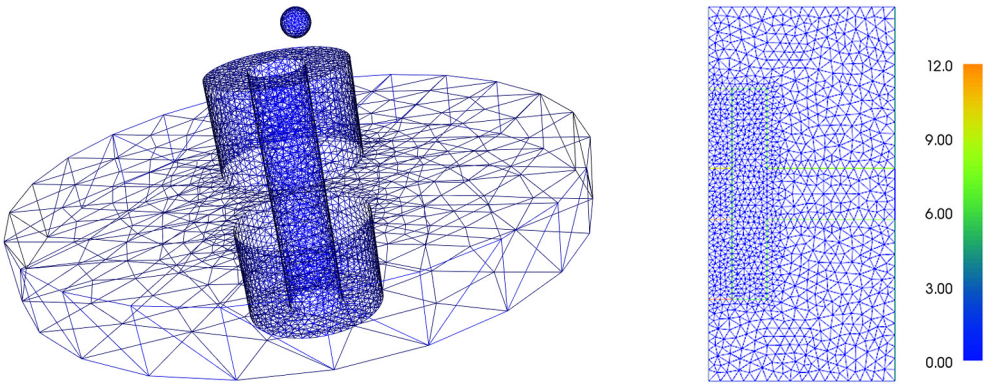


Fig. 3. DNA nanopore model. Shown are the 3D mesh of the pore, membrane and molecule, and the 2D mesh of a half-plane on which the problem is solved in the axisymmetric formulation. The geometry is inspired by [6].

4. Results and discussion

In this section, we want to assess our numerical approach in detail and gain further insights into the modeling and simulation. To embed numerical experiments in a real-world application setting, we introduce a 3D model of the nanopore sensor recently published in [6].

For a basic validation of the correctness of our implementation, we also carried out experiments on an idealized test problem where a semi-analytical solution is possible. They can be found in the Appendix A.1.

4.1. DNA-based nanopore sensor

We describe the DNA-based nanopore sensor model inspired by [6]. The pore consists of six vertically aligned 9 nm short DNA strands arranged in a circle, forming a hole of width approximately 5 nm. As shown in Fig. 3, we modeled the DNA pore wall and hole by two concentric cylinders of radii 2.5 and 1 nm, respectively (thus assuming the DNA thickness to be 1.5 nm on average). The pore resides in a lipid bilayer membrane of thickness 2.2 nm, and the whole system is placed in the center of a cylindrical electrolyte reservoir of height 20 nm and radius 10 nm. We use relative electrical permittivities $\varepsilon_r = 80.2$ for water, $\varepsilon_r = 2$ for the lipid bilayer and $\varepsilon_r = 12$ for DNA. In some of the following numerical experiments, we also include a spherical target molecule, as depicted in Fig. 3, which has $\varepsilon_r = 12$. Without an off-centered molecule, the domain is axisymmetric and simulations can optionally be done with the 2D solver.

Concerning boundary conditions, the DNA surface (inner and outer pore wall, except the outer part which touches the membrane) is equipped with a uniform negative surface charge density ρ_{DNA}, while the membrane was assumed uncharged; a potential bias ΔV is applied on the bottom of the domain; and both ion concentrations are fixed to a bulk value c_0 on the top and bottom. If not stated otherwise, the numerical values used are $\rho_{\text{DNA}} = -0.25q/\text{nm}^2$, $\Delta V = -100$ mV and $c_0 = 300$ mol/m^3. The molecule is equipped with a fixed total charge of $-q$, which is spread uniformly across the volume occupied by the discretized molecule. The molecule velocity is zero, with the exception of Section 4.4 where we take a close look at the relationship between velocity and force.

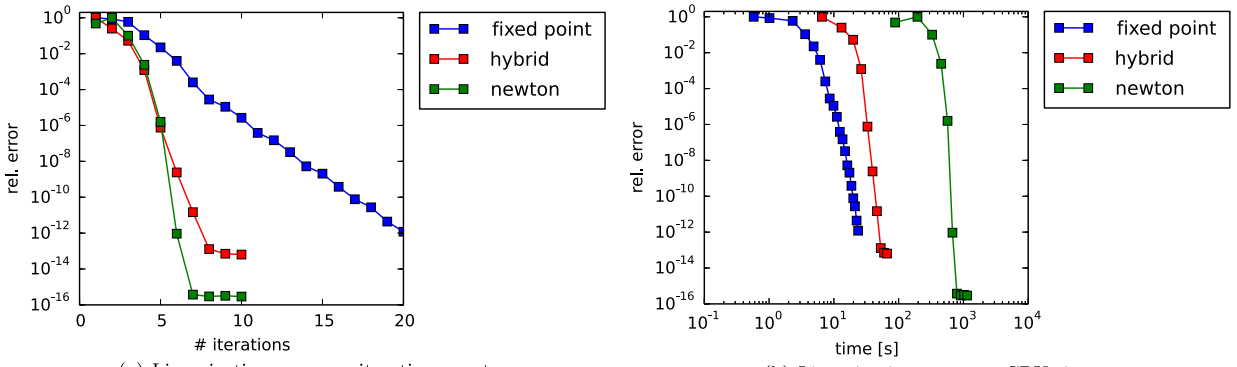
Since our nanopore has such a small (2 nm) diameter, a small note on the applicability of a continuum model may be in order. For DNA translocation, reasonable agreement of the electrophoretic force was demonstrated in several comparisons between the PNPS model and experiments [26,27,62,43], on pores with dimensions similar or slightly larger than this one. Also, PNP theory for ion channels of even smaller diameter has long been known to reproduce the ion distributions and currents of more detailed molecular dynamics simulations at least qualitatively [12,50,38].

4.2. Comparison of linearization schemes

In Section 3.1, we proposed three approaches for linearization of the PNPS system, which we now compare regarding convergence speed and robustness. We consider the axisymmetric (2D) formulation without target molecule as pictured in Fig. 3 (right), with a fixed quasi-uniform mesh size with $h \approx 0.1$ nm.

Comparison of convergence speed. Fig. 4(a) compares convergence speed of the Newton, hybrid and fixed-point methods in terms of the number of iterations. The voltage bias was lowered to -50 mV in this example to ensure convergence of all three schemes. The two Newton-based methods exhibit fast, superlinear convergence at first; the hybrid method deviates from this only in the last few steps and settles for the expected linear asymptotic rate. For the fixed-point method, a slower linear convergence rate can be observed. This indicates that the nonlinearity in the PNP system is far stronger than the coupling between PNP and Stokes, and the iteration between the latter two converges very fast.

In terms of computational time, the picture is turned upside down, because a single iteration of the fixed point scheme is much faster than for the other schemes. All in all, as shown in Fig. 4(b), the fixed point method clearly wins in terms of



(a) Linearization error vs. iteration count

(b) Linearization error vs. CPU time

Fig. 4. Comparison of the linearization error between the Newton, hybrid and fixed-point methods. The error is computed as defined in Algorithms 2, 3 and 4, respectively.

convergence speed. If we compare the times when an accuracy of 4 digits is first reached, fixed-point (with 8.6 s) is 3.8× faster than hybrid (33 s) and 67× faster than Newton (over 9 min) in this particular example. On smaller meshes than shown in the figure, all three methods are closer, meaning the segregated approaches scale better with mesh size.

To put these results into the right perspective, let us note that in Fig. 4(b), computational times are dominated by the algebraic linear solver, and that we used a direct method for solving – which has $O(N^{3/2})$ complexity in the number of variables [24] and thus gives an inherent advantage to decoupled systems, as $2(N/2)^{3/2} < N^{3/2}$. This could be considered an unfair comparison and raises the question whether the advantage remains if an iterative solver of optimal complexity $O(N)$ is used. However, choosing a good preconditioner for the full Newton system seems a more difficult task than for the segregated solvers. For the hybrid and fixed-point methods, the PNP and Stokes subsystems can be treated separately, which makes them more convenient when an iterative solver is used.

In the case that an LU solver is used, like here, there is an additional tweak for the fixed-point and hybrid method that we have not mentioned: Since the bilinear form of the Stokes equations is not changing, the system matrix has to be assembled and factorized only once. This makes the dominance of the fixed point method even more pronounced, but for comparison's sake it was not used in Fig. 4(b).

Comparison of convergence robustness. Next, we compare robustness of convergence with respect to variations of the model parameters. In the speed comparison, parameters where chosen deliberately so that all three linearization methods converged. But, as we will show, either of the methods diverges if some input parameters are made large enough. We want to explore and be aware of these convergence boundaries, as far as they are relevant to realistic models.

The two parameters we identified as critical for convergence are surface charge and voltage bias. Interestingly, the hybrid method is mainly sensitive to surface charge but not to voltage bias, while for the fixed-point method the opposite is true. In Fig. 5, we illustrate the region of convergence for these two methods in the 2D plane of DNA surface charge and voltage bias parameters, with the (negative) surface charge density ranging up to $2q/\mathrm{nm}^2$ and the (negative) voltage[4] up to 2 V. We disregard the full Newton method at this point because of its unfavorable speed performance, and because we observed its robustness properties to be similar to the hybrid method.

The parameter values for which convergence is achieved are colored in Fig. 5 in bright blue and red for the fixed-point and hybrid method, respectively. As can be seen, the fixed-point method has problems even for a moderate voltage bias of -0.1 V, while the hybrid method converges for all voltage biases. On the other hand, the hybrid method has trouble with a (realistic) DNA charge density of $-q/\mathrm{nm}^2$.

In light of these observations, we propose a simple modification to either of the two methods to mitigate convergence issues. For the fixed-point method, the idea is to slowly ramp up the voltage during the iteration, beginning with zero and increasing the voltage bias by a fixed small amount per step until the desired value is reached. Until the iteration has arrived at the full voltage, only the PNP equations are solved. A step-wise increase of 0.025 V has been found to work well; higher values can cause the iteration to become unstable and diverge. With this modification, the fixed-point method converges for every considered voltage bias, but the number of iterations can become unsatisfyingly high if a large number of steps is needed to divide the voltage by 0.025 V. Therefore, in Fig. 5, we only count the iteration as converged if the desired tolerance of 10^{-3} was reached during a specified number of iterations, 20 or 100 respectively. The modified fixed-point algorithm is denoted as *fixed point + voltage schedule* in the figure.

For the hybrid method, which has problems with high surface charge, we modify it by first solving the nonlinear Poisson–Boltzmann equation and using that as an initial guess. This makes sense because the stationary exponential distribution of

[4] For DNA-based nanopore models, we expect ±0.2 V to be sufficient, since for larger voltages the pore is not stable anyway [7]. The surface charge of DNA is about $-q/\mathrm{nm}^2$, which is already high compared to other materials.

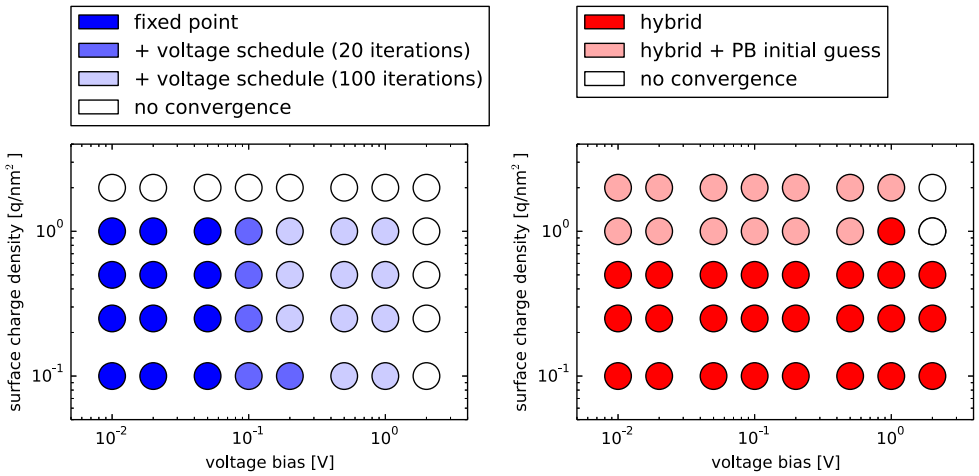


Fig. 5. Robustness of convergence of hybrid and fixed-point schemes with respect to voltage bias/surface charge. The colored circles represent values of voltage and surface charge where the respective iteration converges. Convergence is defined as the linearization error (see Fig. 4) falling below $\mathrm{tol} = 10^{-3}$; in case of non-convergence the solution either exploded or, for the fixed-point methods with voltage schedule, an allowed number of iterations was exceeded. Lighter colors are meant to subsume darker colors; i.e. the *hybrid method + PB initial guess* converges for same values as the *hybrid method* (red) plus additional values which are colored in light red. A white circle indicates that none of the respective methods converged. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

ions inside the Debye layer is well captured by the PB model, and is exactly what causes the Newton method for PNP to diverge. The PB equation is solved with zero voltage bias; the resulting potential ϕ and the ion concentrations $c^{\pm} = c_0 e^{\mp \frac{q}{kT}\phi}$ provide the initial guess; the Stokes variables $\boldsymbol{u}$ and p are initialized to zero as before. The additional nonlinear PB solver adds only small computation overhead to the much larger PNPS system. As we can see in Fig. 5, this modification greatly enhances the robustness of the hybrid method: the *hybrid method + PB initial guess* converges for almost all parameter values considered. It is more robust than any variant of the fixed-point method and is arguably the method of choice if a large range of voltage biases is needed.

Both linearization methods could be made even more robust by using some form of damping or under-relaxation, which we avoided here because damping inherently increases the number of iterations. An adaptive scheme, for example in case of the Newton iteration that is part of the hybrid method, to using damping only when required and the solution is in danger of diverging would definitely be useful to further increase robustness.

In the remaining numerical experiments, the hybrid method is always used.

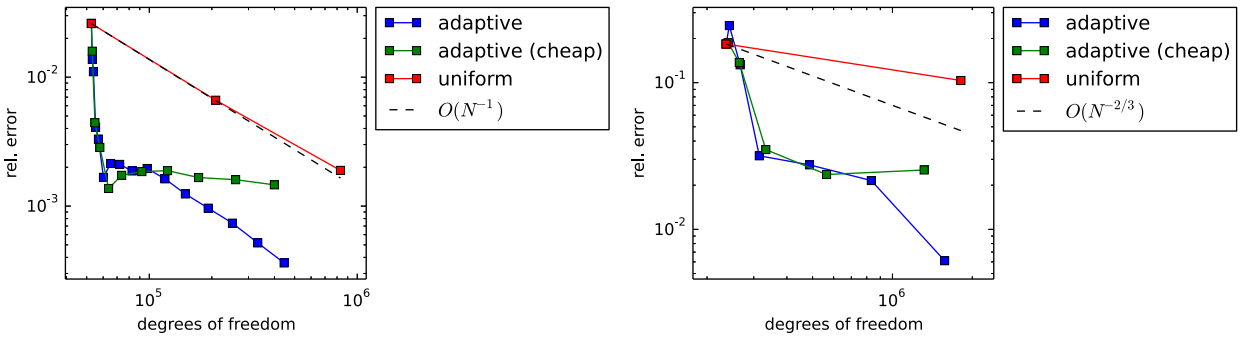
4.3. Assessment of goal-adaptive refinement strategy

Next, we investigate the goal-oriented adaptive scheme from Section 3.3. We use the same geometry as in the last subsection, but with a spherical target molecule of radius 0.5 nm inserted at $z = 2$ nm, inside the pore, slightly above the center. The initial mesh is coarse and quasi-uniform, similar to the ones depicted in Fig. 3.

We are interested if the force on the target molecule from the PNPS system is computed with increasing accuracy on the meshes produced by our adaptive algorithms. *A priori*, this is not clear because the estimator is tailored towards the auxiliary linear PB model with an unphysical pseudo-force as goal functional; it does not necessary yield good results when applied to the full PNPS model. For computation of the error, we created a very fine reference solution with the axisymmetric 2D solver; the actual experiments were carried out with both the 2D and 3D solvers.

Since our geometry has curved parts (spherical molecule and, in 3D, cylindrical pore) whose polygonal discretization influences the numerical values of goal functionals, we adapt the mesh in such a way that new vertices created on boundary facets lie on the actual curved boundaries rather than on the initial discrete polygons. In other words, after mesh bisection the geometry of the problem is slightly altered in every step of the adaptive loop. This is necessary to ensure convergence of a 3D discretization to an axisymmetric 2D reference solution (and, finally, to the actual continuum solution). On the other hand, the geometric error cannot be captured by our error estimator, so performance of the adaptive algorithm may be worse than predicted by theory. Another remark concerning the adaptation process is that we always choose volume and surface charge densities so that the *total charge* on molecule and pore wall are the same as they would be for the exact curved geometry (i.e., we change the densities in every step), because this makes more sense from a physical point of view.

For the PNPS force in 3D, it turned out that we could not exhibit satisfying convergence when using the stabilized $P^1 - P^1$ formulation of the Stokes system (results not shown). This was remedied by switching to a Taylor–Hood ($P^2 - P^1$) formulation; the additional error introduced by the stabilization seems to dominate the pure discretization error on coarse meshes. The results shown in Fig. 6 were obtained with the Taylor–Hood formulation.



(a) Convergence of force: 2D (b) Convergence of force: 3D

Fig. 6. Performance of different refinement strategies with respect to the force (4) in axisymmetric 2D (left) and 3D (right).

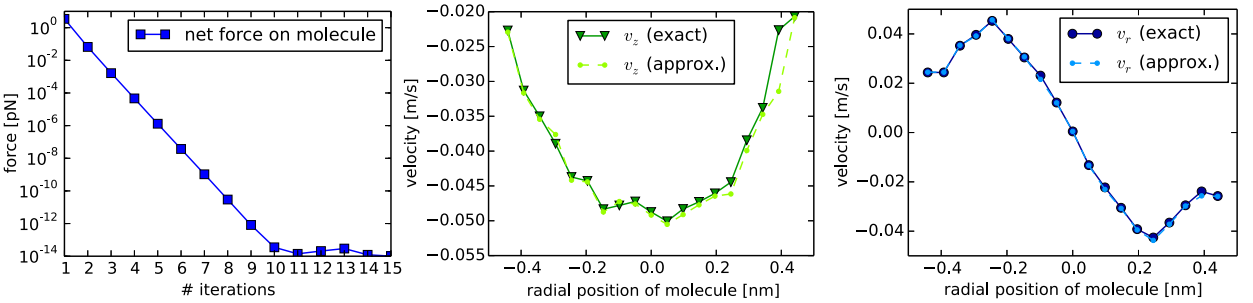


Fig. 7. Determining molecule velocity $\boldsymbol{v}$ from force balance $\boldsymbol{F} = 0$. Left: Magnitude of force $|\boldsymbol{F}|$ after every iteration of Algorithm 1. Center/right: Comparison of exact $\boldsymbol{v}$ determined by iterative method (solid lines) and approximation $\boldsymbol{v} = \gamma(x)^{-1}\boldsymbol{F}(x, 0)$ (dashed lines). The molecule is at different radial positions along the pore cross-section $z = 0$ (halfway through the pore); we show the z (center) and r (right) components of the velocity.

In Fig. 6, we compare the different refinement strategies proposed in Section 3.3. Starting from the same initial mesh, we compare the classical goal-adaptive Algorithm 5 and the proposed cheaper variant Algorithm 6 without extrapolation, as well as uniform refinement as a baseline. Shown is the error relative to the 2D reference solution, defined as

$$\text{error} := \frac{|\boldsymbol{F}_{\text{el},z} - \boldsymbol{F}^{\text{ref}}_{\text{el},z}|}{|\boldsymbol{F}^{\text{ref}}_{\text{el},z}|} + \frac{|\boldsymbol{F}_{\text{drag},z} - \boldsymbol{F}^{\text{ref}}_{\text{drag},z}|}{|\boldsymbol{F}^{\text{ref}}_{\text{drag},z}|}.$$

We compare only the z-components of the forces since the radial components are zero by design in the axisymmetric formulation (the molecule sits on the central axis). The reason we sum absolute values of electrical and drag error components is to avoid a "lucky cancellation" that was observed in several data points and would make the errors shown in this figure unreliable.

Both in 2D (left) and 3D (right), the adaptive schemes are clearly superior to uniform refinement, especially in the first couple of steps when the error gets quickly reduced by about one order of magnitude. After that, however, the cheap estimator degrades in performance, while the classical estimator seems to enter an asymptotic phase where a well-defined convergence rate of $O(h^2)$ is assumed. See Appendix A.3 for a derivation of this convergence rate using *a priori* estimates.

Practically speaking, we see that both adaptive schemes make it possible to drive the approximation error of our main quantities of interest down to a few percent, in the full 3D problem, with a single CPU core on a desktop machine as used for creating Fig. 6. In case the axisymmetric 2D formulation can be applied, even much lower tolerances can be achieved more quickly. Regarding the choice of adaptive algorithm, from our results we would recommend to stick with the more expensive Algorithm 5 (which is still cheap compared to solving the PNPS system). This algorithm does surprisingly well given that it only uses information from solutions to the linear Poisson–Boltzmann equation, and despite inexact discretization of the curved geometry. The cheaper variant (Algorithm 6) seems to capture the rough regions of interest during the first few refinement steps but may not be valid in the asymptotic limit.

4.4. Velocity iteration and linear velocity approximation

We turn to the evaluation of the velocity iteration scheme described in Section 2.5, for determining the molecule velocity leading to a zero net force. In this algorithm, the Stokes and PNPS systems are solved several times, but mesh refinement has to be applied only once, at the beginning, because all instances of the PNPS solver share the same geometry (only the no-slip boundary condition at the molecule is changed). In Fig. 7 (left), we show how the magnitude of the net force is decreased

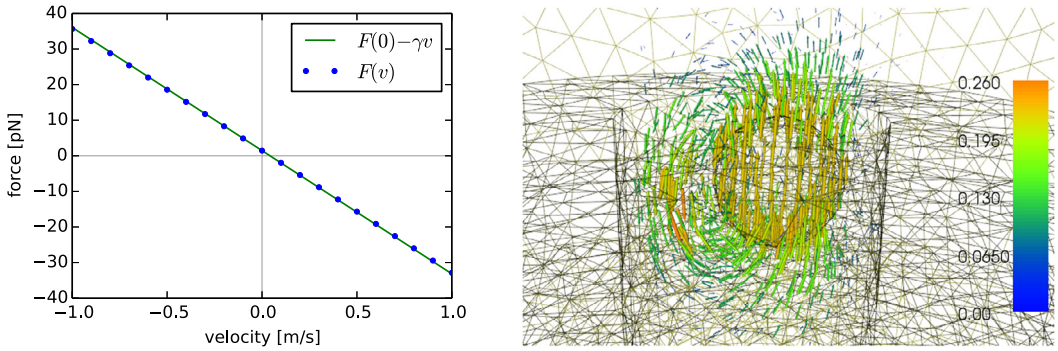
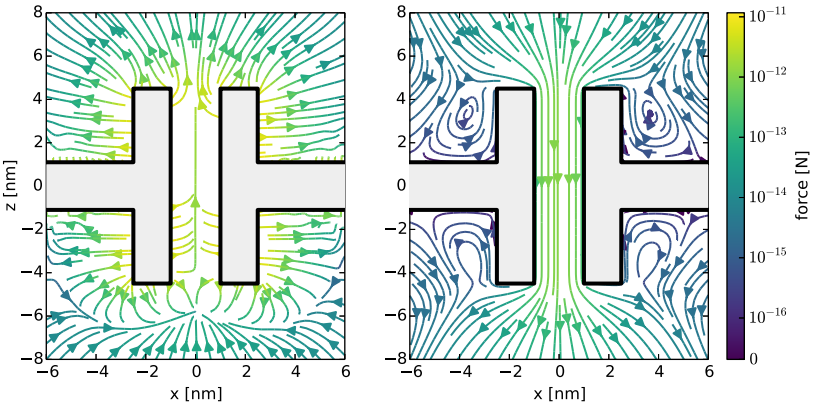


Fig. 8. Left: Velocity-dependence of force for fixed molecule position $x = (0,\ 0,\ 4.5\ \text{nm})$. Velocity and force only point in the z direction and are therefore treated as scalars. The linear relationship $F(x, v) \approx F(x, 0) - \gamma(x)v$ holds almost exactly. Right: Picture of velocity field obtained from the 3D PNPS solver.



(a) Electric force (b) Drag force

Fig. 9. Streamline plots of PNPS force fields from the point-sized molecule model. The molecule has a radius $r = 0.5$ nm and a negative total charge of $-q$.

by subsequent iterations of Algorithm 1. The molecule in this figure is positioned off-centered at $x = (0.2\ \text{nm},\ 0,\ 0)$, i.e. we use the full 3D solver. We see that the algorithm leads to rapid, linear convergence $|\boldsymbol{F}| \to 0$.

Already after a *single* update of the velocity (second iteration in Fig. 7, left), the force is reduced by two orders of magnitude compared to its initial value. This corresponds to the linear velocity approximation $\boldsymbol{F}(x, \boldsymbol{v}) \approx \boldsymbol{F}(x, 0) - \gamma(x)\boldsymbol{v} = 0$, i.e. $\boldsymbol{v} = \gamma(x)^{-1}\boldsymbol{F}(x, 0)$. In Fig. 7 (center, right), we compare this approximate formula for the velocity by with the *exact* velocity after 6 iterations. The molecule varies along the radius of a pore cross-section. Clearly, the direct formula captures the physical situation very well, and is in fact hardly distinguishable to the exact velocity.

Next, we want to test the validity of the linear velocity approximation not only in case $\boldsymbol{v}$ is the drift velocity, but also for larger velocities. In Fig. 8 (left) we plot results from a 2D simulation where the molecule position was fixed at the center, and different velocities of the form $\boldsymbol{v} = (0,\ 0,\ v)$ are applied. In this case, the force is also of the form $\boldsymbol{F} = (0,\ 0,\ F)$ and the friction tensor reduces to a single number $\gamma(x) \in \mathbb{R}$ which was computed by solving the Stokes equation (11) in axisymmetric variables. The figure compares the exact force $F(x, v)$ with the linear approximation $F(x, 0) - \gamma(x)v$. The difference, stemming from nonlinear effects in the PNPS equations, is negligible. Physically speaking, this means that the molecule velocity has almost no impact on the charge distribution and electric potential, at least not one that affects the electrophoretic force. These results confirm that the force on a molecule with zero velocity, together with the friction tensor, can be viewed as the main output of our solver, since they can be used to treat *any* velocity via the linear approximation.

4.5. Finite-sized vs. point-sized target molecule

After having obtained a good picture of the numerical accuracy and having established that the target molecule can be assumed to have zero velocity, we want to address the modeling question put forward in Section 2.4: whether to model the molecule as *finite-sized* and explicitly part of the geometry or only implicitly (*point-sized*), i.e., simulate without molecule and only afterwards calculate the forces on a virtual molecule from the electric and flow fields.

The point-sized molecule approach – albeit less accurate – is computationally attractive, since the whole force field is obtained from a single simulation, while in the finite-sized approach one has to compute the force on a different mesh for every distinct molecule position. In Fig. 9, we show the electric and hydrodynamic force fields obtained from the point-size model and an axisymmetric simulation.

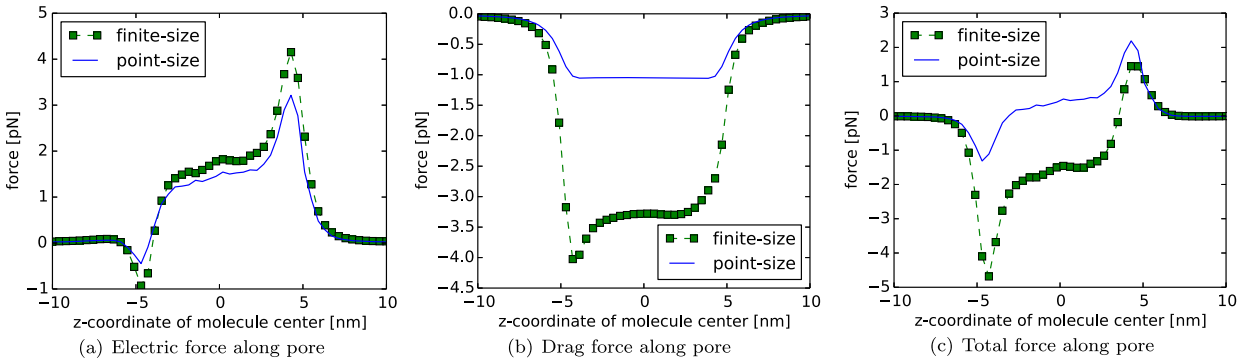


Fig. 10. Force profiles along DNA nanopore, finite-size vs. point-size model. The z-component of forces is plotted along the central pore axis, where for the finite-size approach every data point corresponds to a distinct geometry and thus simulation; for the point-sized approach the whole curve is obtained from a single simulation. A negative total force in some region means that the target molecule is, on average, driven downward through the pore. In the point-size model, the magnitude of drag force is underestimated to such an extent that the net total force has a different direction than in the finite-size model.

Qualitatively, the following physical phenomena can be observed in Fig. 9: As the molecule is negatively charged, and the applied electric field is pointing downwards, the net electrical force $\boldsymbol{F}_{\text{el}}$ inside the pore is in the *upper* direction (Fig. 9, left). On the other hand, the negative surface charge of DNA (which makes up the pore wall) leads to crowding of the pore by positive ions; the positive ions move downwards, causing electroosmotic drag $\boldsymbol{F}_{\text{drag}}$ in the *downward* direction (Fig. 9, right). From these two competing effects, it is *a priori* not clear whether the net force on the molecule will be upwards or downwards. Outside the pore, the molecule is repelled away by $\boldsymbol{F}_{\text{el}}$ due to the equally signed charges, which leads to an energy barrier for entering the pore from either direction.

Force profiles. In Fig. 10, force profiles along the pore are compared for the finite- and point-size formulation. For the point-sized molecule, we calculate drag using Stokes' law $\boldsymbol{F}^*_{\text{drag}}(x) = 6\pi\eta r\boldsymbol{u}(x)$ which is valid for a particle in bulk water. As we can see (Fig. 10, center), this leads to a strong underestimation of drag force inside the pore. In the finite-size model, the drag force actually dominates the electric force by a factor 2, but is weakened by a factor 3–4 in the point-size model; so the net total force comes out in the wrong direction. For the electric force, agreement between both models looks more reasonable, yet the point-size model fails to capture an additional repelling force at the edges which can possibly be attributed to the dielectric self-energy of the molecule [14].

Improving the point-size model. Looking at Fig. 10, it is clear that the point-size model could be improved a lot by simply scaling the drag force by a constant factor inside the pore. More generally, we propose to modify the calculation of the force on a point-sized molecule by using

$$\boldsymbol{F}^*(x) = \alpha(x)\boldsymbol{F}^*_{\text{el}}(x) + \beta(x)\boldsymbol{F}^*_{\text{drag}}(x).$$

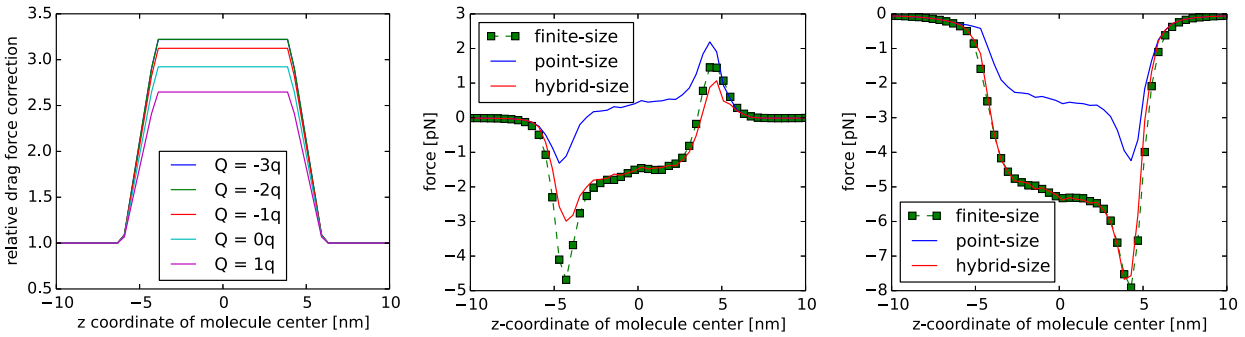
Here, $\alpha(x)$ and $\beta(x)$ are corrective factors yet to be determined and $\boldsymbol{F}^*_{\text{el}}(x)$, $\boldsymbol{F}^*_{\text{drag}}(x)$ are the point-sized molecule forces given by (8) and (9). The corrections α and β will be obtained by some parametrization of the finite-size model, for instance by evaluating the finite-size forces $\boldsymbol{F}_{\text{el}}(x)$ and $\boldsymbol{F}_{\text{drag}}(x)$ on a small number of points $x_1, \dots, x_n$ and constructing α and β by interpolation to satisfy

$$\alpha(x_i) = \frac{\boldsymbol{F}_{\text{el}}(x_i)}{\boldsymbol{F}^*_{\text{el}}(x_i)}, \qquad \beta(x_i) = \frac{\boldsymbol{F}_{\text{drag}}(x_i)}{\boldsymbol{F}^*_{\text{drag}}(x_i)}, \qquad i = 1, \dots, n.$$

In this way we ensure $\boldsymbol{F}^*(x_i) = \boldsymbol{F}(x_i)$ at the interpolation points. The idea is that the expensive finite-size model has to be evaluated only at a few points instead of hundreds or thousands, since the rough overall shape of $\boldsymbol{F}^*$ is correctly determined by the point-size model. Thus we hope to get the best of both models, efficiency as well as accuracy.

For demonstration, we apply this approach – which we dub the *hybrid-size model* – using only a *single* interpolation point at the pore center. The resulting correction factor $\beta(x)$ to the drag force is shown in Fig. 11(a), for molecules of different charges. We constructed β so that is has the same (interpolated) value throughout the pore; outside the pore, we set $\beta = 1$ (assuming that the point-size model is correct under bulk conditions), and in the transition region near the pore edges, we interpolate linearly between those two values. For $\alpha(x)$ the procedure is the same.

By using only one interpolation point, we deliberately ignore possible edge effects captured by the finite- but not the point-size model. Still, as shown in Fig. 11(b), (c), this approach significantly improves on the pure point-size model for molecules of both positive and negative charge, with only twice the computational cost. This may be in part due to our specific nanopore geometry, which has a constant radius along the entire pore; but the approach is general enough to be applicable to irregularly shaped pore proteins such as α-hemolysin, by using more than one interpolation point.



(a) Correction of drag force $\beta(x)$ for different molecule charges

(b) Total force, molecule charge $-q$

(c) Total force, molecule charge $+q$

Fig. 11. Demonstration of the hybrid-size model for approximating the finite-size model with minimal computational effort. Left: Correction factor computed for the drag force, which measures by how much the finite-size model is underestimated by the point-size model. Center/right: Corrected force profiles for molecules with negative and positive unit charge, respectively.

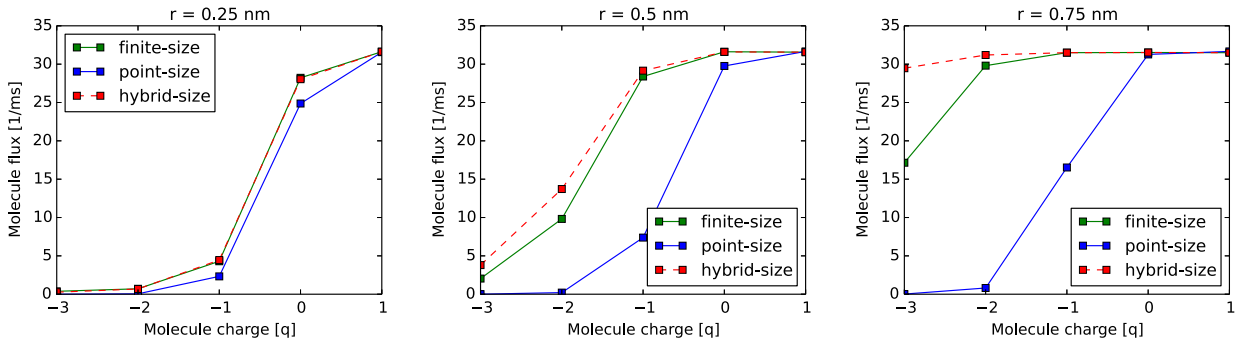


Fig. 12. Molecule flux after 1 µs for different molecule charges and sizes and different models of the force. From left to right, the radii are $r = 0.25$, 0.5, 0.75 nm. The point-size model does not match the results of the (expensive and more accurate) finite-size model. For most setups however, the (cheap) hybrid-size model comes relatively close. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Comparison of molecule flux. Finally, we want to provide a more succinct way of relating our three models (finite-, point- and hybrid-size). In the end, we are interested in how the PNPS force influences selectivity between different molecule species. Therefore, the single number that represents each model best is the flux of target molecules, i.e., the number of molecules translocating the pore in a given time interval. We model transport along the z-direction by a 1D diffusion equation $\frac{dc}{dt} = \frac{d}{dz}(-D\frac{dc}{dz} + \frac{D}{kT}Fc)$; the molecules start out in the upper reservoir and diffuse through the force field in the pore region. Details on the equation and how we solve it are given in Appendix A.4. The term inside the brackets $(-D\frac{dc}{dz} + \frac{D}{kT}Fc)$ is the flux density; multiplied by the cross-sectional area of the pore, it gives the total molecule flux through the pore. We have computed the flux after a time of 1 µs for many different molecule charges and sizes, with results shown in Fig. 12.

In Fig. 12, looking at molecule fluxes from the finite-size model (green), we see that our sensor exhibits non-trivial selectivity between molecules of different charges. Strongly negatively charged molecules struggle to make their way through the pore against the direction of voltage bias. This is more pronounced for smaller molecules; for large molecules of radius $r = 0.75$ nm, there is almost no differentiation, as the charge-agnostic drag force dominates. The latter effect is captured well by the hybrid-size model, but not by the uncorrected point-size model. However, the hybrid-size model as implemented with a single interpolation point underestimates the energy barrier at the pore entrance for large molecules of charge $-3q$, and therefore overestimates the flux there (Fig. 12, right). This effect vanishes for larger timescales, which is why we stopped after 1 µs to reveal the limitations of our proposed hybrid-size model.

5. Conclusion

We have implemented a 2D/3D finite element solver for the steady-state Poisson–Nernst–Planck–Stokes system for the simulation of nanopore sensors. To solve the nonlinear equations, we proposed three different schemes for linearization in an attempt to clarify the amount of system segregation that leads to the most efficient and robust solver. Two methods emerged as roughly equal candidates, with the pure fixed-point method being faster for small voltages and the hybrid method, if initialized from the Poisson–Boltzmann equation, better suited to work across a large range of voltages. For the fixed-point method, a new correction to the Poisson equation was proposed that prevents blow-up and – for small voltages – enables convergence in less than 10 iterations without the need for damping.

The goal-adaptive mesh refinement schemes we proposed can efficiently allocate computational resources towards the prediction of output functionals. This proved indispensable to evaluate the electrophoretic force on particles in 3D with accuracies of over 99% on non-HPC architectures. We experimented with a cheaper alternative to the goal-oriented error estimation procedure established in [3] by skipping the extrapolation step, but conclude that the extrapolation-based algorithm remains the more robust choice. Interestingly, we have shown that adaptivity can work in practice even if a radically simplified model is used to compute the error indicators. This gives a total speedup of roughly 50%, because we essentially save the effort to compute all the solutions on coarser meshes that are not needed in the end. The idea should be applicable to a variety of computational problems where model simplifications are available.

To determine the target molecule's velocity from a force balance condition, we showed how the iterative approach from [53,31,1] can be replaced by a simple direct formula. The formula relies on an approximate linear relationship between force and velocity, which we found to be close to exact. The linear approximation could also be used to great effect for simulation of the molecule trajectory with a more refined model – the Langevin equation – which takes into account random collisions.

For the modeling of selectivity in nanopore sensors, we find that assuming a point-sized target molecule, although computationally attractive, severely reduces the predictiveness of our simulations. For practical applications, it is therefore recommended to model the molecule explicitly and calculate the force for each possible molecule position separately. However, if this is too expensive, a carefully calibrated hybrid model that we proposed may also capture most features of the force field and give results similar to the full finite-size model. These results are, of course, not only applicable to nanopore sensors, but also to related systems such as ion channels.

Acknowledgements

We acknowledge support through the Austrian Science Fund (FWF) START Project No. Y660 *PDE Models for Nanotechnology*. We also thank the anonymous reviewers for their valuable suggestions.

Appendix A

A.1. Semi-analytical test problem

To validate our 2D and 3D solvers, we look at a simple model problem where the whole PNPS system can be reduced analytically to the solution of a scalar 1D boundary value problem. Since the 1D problem can be solved to high accuracy with negligible effort, it serves our purpose just as well as an exact analytical solution would. Unlike manufactured test problems often employed in academia, we do not introduce artificial right hand side terms. The problem actually describes, albeit simplified, a physically relevant situation: that of a nanopore where the pore length is large compared to the radius and the applied voltage is sufficiently low to have negligible influence on the ion distribution.

The domain is a plain cylindrical tube of radius $R = 1$ nm and length $L = 4$ nm and is meant to represent an inner part of the nanopore (far away from the pore entrances on either side); the only dielectric material is water. Using axial symmetry, we write the PNPS solutions as functions of the radial component r and the height z. The exact potential is prescribed as

$$\phi(r, z) = U_T \phi^*(r) - E_0 z,$$

where E_0 is a given constant external field strength, $U_T = kTq^{-1}$ is the thermal voltage and the dimensionless potential $\phi^*(r)$ solves a 1D radial Poisson–Boltzmann equation

$$\frac{1}{r}\frac{d}{dr}\left(r\frac{d\phi^*}{dr}\right) = \frac{1}{\lambda_D^2}\sinh(\phi^*).$$

Here $\lambda_D = \sqrt{\frac{\varepsilon kT}{2Fc_0 q}}$ is the Debye length; the potential is subject to the boundary condition $U_T \frac{d\phi^*}{dr}(R) = \rho$ for a given constant surface charge density ρ. This equation is readily solved e.g. by 1D FEM.

The other unknowns depend only on r and can all be obtained from $\phi^*(r)$: Ion concentrations are given by the Boltzmann factors

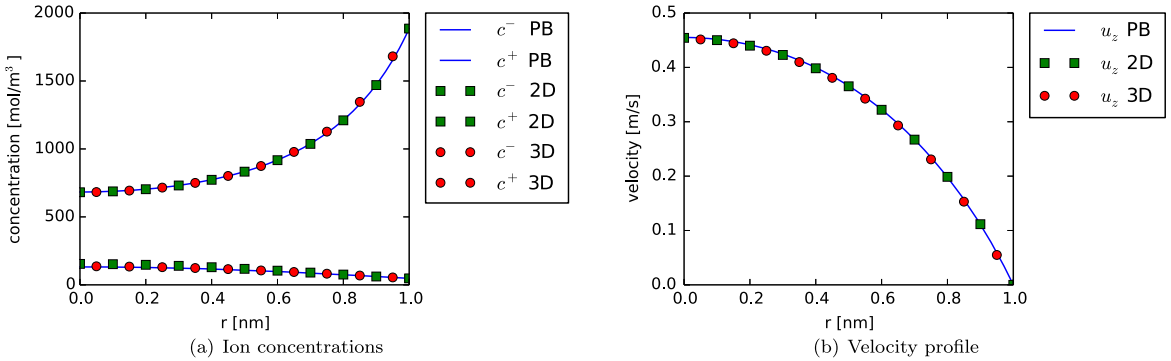
$$c^{\pm}(r) = c_0 \exp\left(\mp\phi^*(r)\right)$$

for any given bulk concentration c_0; the velocity is of the form $\boldsymbol{u} = (0, 0, u)^T$ with

$$u(r) = \frac{\varepsilon E_0}{\eta} U_T \left[\phi^*(r) - \phi^*(R)\right];$$

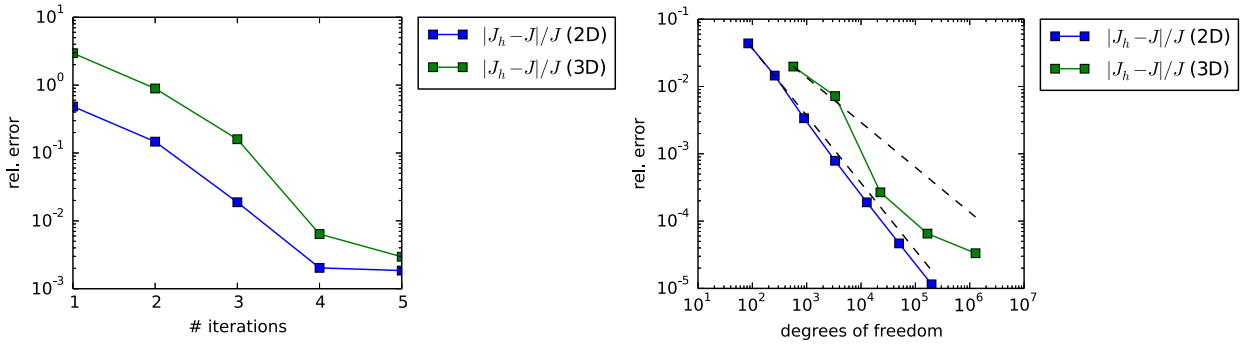
and the pressure is

$$p(r) = -2Fc_0 U_T \cosh(\phi^*(r)) + p_0$$



(a) Ion concentrations (b) Velocity profile

Fig. 13. Cross-sectional plots for the analytical test problem with three different numerical methods: the 1D PB equation and the 2D and 3D PNPS solvers.



(a) Convergence with nonlinear iteration (b) Convergence with mesh refinement

Fig. 14. Convergence to exact solution. The relative error is defined as $|J_h - J|/|J|$ where J is the exact current from the 1D model and J_h is the output of the PNPS solvers. (a) Error against number of iterations in the hybrid linearization scheme. (b) Error against degrees of freedom N for uniform mesh refinement. The theoretical N dependency is $O(N^{-1})$ in 2D and $O(N^{-2/3})$ in 3D, indicated by dashed lines.

with an arbitrary constant p_0, which we fix by requiring $p(0) = 0$. A mathematical derivation of this exact solution can be found in Appendix A.2.

Numerical results. We solve the problem both in 3D and in axisymmetric 2D variables. Dirichlet or Neumann boundary conditions are applied to match the exact solutions; we have tried to encode as little information about the exact solutions as possible into the boundary conditions. For instance, for the Stokes equation it suffices to enforce the usual no-slip condition $\boldsymbol{u} = 0$ on the wall, the Neumann-type condition $n \cdot \nabla \boldsymbol{u} = 0$ at the upper and lower ends, and the no pressure condition $p = 0$ somewhere in the center. The numerical values used for the free parameters were $E_0 = -100$ mV, $\rho = -0.05$ C/m^2 and $c_0 = 300$ mol/m^3, respectively. The Debye length, which determines the width of the double layer and thus the multiscale nature of the problem, is $\lambda_D = 0.56$ nm in this case. Comparing this with the pore radius of $R = 1$ nm, we see that we have posed ourselves a rather easy problem numerically (in the sense that a coarse mesh will suffice to resolve all features). Our test problem can be turned into a challenging benchmark problem by increasing the pore radius such that R/λ_D is large.

In Fig. 13 we compare cross-sectional plots of ion concentrations and the velocity profile coming from the three different numerical formulations: the 1D Poisson–Boltzmann equation – which we take as the *exact solution* –, the axisymmetric 2D model and the full 3D PNPS system. We used a uniform mesh width of $h = 0.1$ nm in these calculations, which corresponds to about 1k mesh elements in 2D and 50k in 3D. It is reassuring that the three results in Fig. 13 are almost indistinguishable.

For a quantitative assessment of the solver accuracy, we calculate the total ion current through the pore, defined as

$$J = F \int \left(j_z^+ - j_z^-\right),$$

where $j_z^\pm$ is the z component of the ion flux defined in (1); the integral is taken over any horizontal cross-section of the pore. In practice, it is more robust to average over many cross-sections by taking a volume integral, which is what we do. The ion current involves the potential, both ion concentrations and the velocity and therefore provides a suitable measure of accuracy. In Fig. 14(a) we show the relative error of J compared to the exact solution against number of iterations of our hybrid Newton/fixed-point scheme, with fixed mesh size $h = 0.2$ nm. After about five iterations a converged state is reached where the error stops changing; at this point, the linearization error is dominated by the discretization error.

To study the discretization error, in Fig. 14(b) the mesh is refined uniformly starting from a coarse initial mesh, and the resulting error curve compared to the number of degrees of freedom is shown. Note that every data point in the

latter figure corresponds to the final converged state of Fig. 14(a). Different asymptotic slopes for 2D and 3D arise as derived in Section 4.3, confirming that the error in J is $O(h^2)$. This provides a strong validation for the correctness of our implementation. More generally speaking, the results also indicate that for the current we should attain high numerical accuracy quite easily in simulations similar to this test problem, without any goal-specific adaptivity.

A.2. Derivation of test problem

First, assume that the potential $\phi(r,z)$ is a superposition of a part resulting from a given constant external field E_0 in the z-direction and a part which depends only on the radius. Thus,

$$\phi(r,z) = \psi(r) - E_0 z.$$

Second, the ion concentrations are assumed to depend only on r and to be given by Boltzmann factors

$$c^{\pm}(r) = c_0 \exp\left(\mp \frac{q\psi(r)}{kT}\right).$$

Third, for the Stokes equation we assume the velocity is of the form $\boldsymbol{u} = (0,0,u_z)^T$ with $u_z = u_z(r)$ and pressure $p = p(r)$, i.e., the velocity field points only in the z-direction but is constant in this direction.

These three assumptions, which can be encoded in the boundary conditions, determine a solution to the PNPS system as follows. Since the second derivatives of the external part of the potential $-E_0 z$ vanish and because of the assumed form of the ion concentrations, the Poisson equation reduces to a 1D Poisson–Boltzmann equation in radial direction

$$\varepsilon \frac{1}{r}\partial_r(r\partial_r\psi) = 2Fc_0 \sinh\left(\frac{q\psi}{kT}\right)$$

with Neumann boundary condition $\partial_r\psi(R) = \rho$ for some given (constant) surface charge density ρ. The permittivity ε is constant since our domain only consists of fluid.

The Stokes equation splits into a x, y part only for the pressure (since $u_x = u_y = 0$) and a z part only for the velocity (since $\partial_z p = 0$),

$$\partial_r p = -F(c^+ - c^-)\partial_r\psi, \tag{24}$$

$$-\eta \frac{1}{r}\partial_r(r\partial_r u_z) = -F(c^+ - c^-)E_0. \tag{25}$$

Equation (24) can be solved by noting that the right hand side can be written as a derivative

$$-F(c^+ - c^-)\partial_r\psi = -2Fc_0 \sinh\left(\frac{q\psi}{kT}\right)\partial_r\psi = \partial_r\left[-2Fc_0\frac{kT}{q}\cosh\left(\frac{q\psi}{kT}\right)\right],$$

which is equal to $\partial_r p$, thus yielding

$$p(r) = -2Fc_0\frac{kT}{q}\cosh\left(\frac{q\psi(r)}{kT}\right) + p_0$$

for some arbitrary constant p_0. For equation (25), we write on the other hand

$$-F(c^+ - c^-) = \varepsilon\frac{1}{r}\partial_r(r\partial_r\psi)$$

and obtain

$$\partial_r(r\partial_r u_z) = -\frac{\varepsilon E_0}{\eta}\partial_r(r\partial_r\psi),$$

which combined with the no-slip condition $u(R) = 0$ yields the solution

$$u(r) = \frac{\varepsilon E_0}{\eta}\left[\psi(R) - \psi(r)\right].$$

Finally, the form of these solutions can be simplified by writing the potential as $\psi(r) = \frac{kT}{q}\phi^*(r)$, where $\phi^*(r)$ is dimensionless.

A.3. Derivation of force convergence rates

To derive the theoretical convergence rates for adaptive mesh-refinement, consider a goal functional $F(u)$, a bilinear form $a(\cdot,\cdot)$ and a discrete solution u_h. Let w and w_h be the continuous resp. discrete dual solutions as in Section 3.3. The error can be estimated as

$$F(u-u_h) = a(u-u_h, w-w_h) \lesssim \|u-u_h\|\,\|w-w_h\|,$$

where $\|\cdot\|$ is the $H^1(\Omega)$ norm of the computational domain Ω. For the subequations of the PNPS problem, which are elliptic and whose primal and dual right hand sides are continuous functionals on H^1, the *a priori* estimate we expect to hold is $\|u-u_h\| = \|w-w_h\| = O(h^1)$ locally in the mesh size h; hence the product is $O(h^2)$. In 3D, the local mesh size h scales like $N^{-1/3}$, where N are the degrees of freedom. In summary, we expect

$$F(u-u_h) = O(h^2) = O(N^{-2/3}).$$

Similarly, in 2D we derive $F(u-u_h) = O(N^{-1})$.

A.4. Diffusion equation for molecule flux

We show how to compute the flux of target molecules for the results in Fig. 12, Section 4.5. For simplicity, we model flux only in the z-direction. Given a force profile $F(z)$ which is obtained from the PNPS model (e.g. Fig. 10), the time-dependent diffusion equation for target molecules reads

$$j = -D\frac{dc}{dz} + \frac{D}{kT}Fc, \tag{26a}$$

$$\frac{dj}{dz} = \frac{dc}{dt}. \tag{26b}$$

Here, $c(z,t)$ is the molecule concentration; $j(z,t)$ is the molecule flux density; and $D(z)$ is the position-dependent diffusion coefficient. We solve this equation on the symmetric interval $|z| \le 1$ µm, with $|z| \le 4.5$ nm defining the pore region. For the initial condition, we use

$$c(z,0) = \begin{cases} c_0, & z > 4.5 \text{ nm}, \\ 0, & \text{elsewhere}, \end{cases}$$

with $c_0 = 1$ mol/m^3. Thus, the molecules start out uniformly distributed in the upper reservoir, and will diffuse downwards through the pore region. At the reservoir boundaries $z = \pm 1$ µm, Neumann conditions $\frac{dj}{dz} = 0$ are applied, modeling hard walls. The diffusion coefficient is modeled by a generalized Stokes-Einstein relation

$$D(z) = \frac{kT}{6\pi\eta r}d(z) \qquad \text{where } d(z) = \begin{cases} d_r < 1 & \text{in the pore}, \\ 1 & \text{in bulk}. \end{cases}$$

Here, r denotes the molecule radius. The reduced relative diffusivity d_r inside the pore depends on r and is taken from the tables in [51].

We solve (26) using backward Euler in time applied to a P^1 finite element space discretization on a uniform grid. For the results in Fig. 12, we used a grid spacing of $h = 0.2$ nm (10k nodes), a timestep of $dt = 10$ ns and stopped after 100 steps or $t = 1$ µs. The final-time downwards flux density $-j(\cdot,t)$ was averaged over the pore region and multiplied by the cross-sectional pore area $A = \pi(1\text{ nm})^2$ and the Avogadro constant N_A to yield the molecule flux in numbers per second.

References

[1] Y. Ai, J. Liu, B.K. Zhang, S. Qian, Field effect regulation of DNA translocation through a nanopore, Anal. Chem. 82 (19) (2010) 8217–8225, http://dx.doi.org/10.1021/ac101628e.
[2] S. Balay, et al., PETSc Users Manual, Revision 3.5, Argonne National Laboratory (ANL), 2014.
[3] W. Bangerth, R. Rannacher, Adaptive Finite Element Methods for Differential Equations, Birkhäuser, 2003, http://www.springer.com/birkhauser/mathematics/book/978-3-7643-7009-1.
[4] H. Bayley, Nanopore sequencing: from imagination to reality, Clin. Chem. 61 (1) (2015) 25–31, http://www.clinchem.org/content/61/1/25.full.
[5] R. Becker, R. Rannacher, An optimal control approach to a posteriori error estimation in finite element methods, Acta Numer. 10 (January 2003) 1–102, http://www.journals.cambridge.org/abstract_S0962492901000010, 2003.
[6] J.R. Burns, A. Seifert, N. Fertig, S. Howorka, A biomimetic DNA-based channel for the ligand-controlled transport of charged molecular cargo across a biological membrane, Nat. Nanotechnol. 11 (2016) 152–156, http://dx.doi.org/10.1038/nnano.2015.279.
[7] J.R. Burns, E. Stulz, S. Howorka, Self-assembled DNA nanopores that span lipid bilayers, Nano Lett. 13 (6) (Jul. 2013) 2351–2356, http://dx.doi.org/10.1021/nl304147f.
[8] C. Carstensen, M. Feischl, M. Page, D. Praetorius, Axioms of adaptivity, Comput. Math. Appl. 67 (6) (2014) 1195–1253.
[9] S. Chandrasekhar, Stochastic problems in physics and astronomy, 1943.

[10] D.P. Chen, R.S. Eisenberg, J.W. Jerome, C.W. Shu, Hydrodynamic model of temperature change in open ionic channels, Biophys. J. 69 (6) (Dec. 1995) 2304–2322, http://www.sciencedirect.com/science/article/pii/S0006349595801013.
[11] J. Clarke, H.-c. Wu, L. Jayasinghe, A. Patel, S. Reid, H. Bayley, Continuous base identification for single-molecule nanopore DNA sequencing, Nat. Nanotechnol. 4 (April 2009) 265–270.
[12] R.D. Coalson, M.G. Kurnikova, Poisson–Nernst–Planck theory approach to the calculation of current through biological ion channels, IEEE Trans. Nanobiosci. 4 (1) (2005) 81–92.
[13] J. Comer, A. Aksimentiev, Predicting the DNA sequence dependence of nanopore ion current using atomic-resolution Brownian dynamics, J. Phys. Chem. C 116 (5) (2012) 3376–3393.
[14] B. Corry, S. Kuyucak, S.-H. Chung, Dielectric self-energy in Poisson–Boltzmann and Poisson–Nernst–Planck models of ion channels, Biophys. J. 84 (6) (2003) 3594–3606.
[15] S. Deparis, Numerical Analysis of Axisymmetric Flows and Methods for Fluid–Structure Interaction Arising in Blood Flow Simulation, Ph.D. thesis, École Polytechnique Fédérale de Lausanne, 2004.
[16] W. Dörfler, A convergent adaptive algorithm for Poisson's equation, SIAM J. Numer. Anal. 33 (3) (1996) 1106–1124, http://epubs.siam.org/doi/abs/10.1137/0733054.
[17] B. Egwolf, Y. Luo, D.E. Walters, B. Roux, Ion selectivity of alpha-hemolysin with beta-cyclodextrin adapter. II. Multi-ion effects studied with grand canonical Monte Carlo/Brownian dynamics simulations, J. Phys. Chem. B 114 (8) (Mar. 2010) 2901–2909, http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=2843906&tool=pmcentrez&rendertype=abstract.
[18] D. Erickson, D. Li, Influence of surface heterogeneity on electrokinetically driven microfluidic mixing, Langmuir 18 (5) (Mar. 2002) 1883–1892, http://dx.doi.org/10.1021/la015646z.
[19] D. Erickson, D. Li, Microchannel flow with patchwise and periodic surface heterogeneity, Langmuir 18 (23) (Nov. 2002) 8949–8959, http://dx.doi.org/10.1021/la025942r.
[20] R.D. Falgout, U.M. Yang, hypre: a library of high performance preconditioners, in: Computational Science – ICCS 2002, 2002, pp. 632–641, http://link.springer.com/10.1007/3-540-47789-6.
[21] M. Feischl, T. Führer, G. Mitscha-Eibl, D. Praetorius, E.P. Stephan, Convergence of adaptive BEM and adaptive FEM-BEM coupling for estimators without h-weighting factor, Comput. Methods Appl. Math. 14 (4) (2014) 485–508.
[22] M. Feischl, D. Praetorius, K.G. van der Zee, An abstract analysis of optimal goal-oriented adaptivity, SIAM J. Numer. Anal. 54 (3) (2016) 1423–1448, http://dx.doi.org/10.1137/15M1021982.
[23] R.W. Freund, A transpose-free quasi-minimal residual algorithm for non-hermitian linear systems, SIAM J. Sci. Comput. 14 (2) (1993) 470–482, http://epubs.siam.org/doi/abs/10.1137/0914029.
[24] A. George, J. Liu, E. Ng, Computer Solution of Sparse Linear Systems, Prentice–Hall Series in Computational Mathematics, Prentice–Hall, 1994.
[25] C. Geuzaine, J.F. Remacle, Gmsh: a 3-D finite element mesh generator with built-in pre- and post-processing facilities, Int. J. Numer. Methods Eng. 79 (11) (Sep. 2009) 1309–1331, http://doi.wiley.com/10.1002/nme.2579.
[26] S. Ghosal, Electrophoresis of a polyelectrolyte through a nanopore, Phys. Rev. E 74 (4 Pt 1) (Oct. 2006) 041901, http://journals.aps.org/pre/abstract/10.1103/PhysRevE.74.041901.
[27] S. Ghosal, Effect of salt concentration on the electrophoretic speed of a polyelectrolyte through a nanopore, Phys. Rev. Lett. 98 (23) (Jun. 2007) 238104, http://link.aps.org/doi/10.1103/PhysRevLett.98.238104.
[28] H.K. Gummel, A self-consistent iterative scheme for one-dimensional steady state transistor calculations, IEEE Trans. Electron Devices 11 (10) (Oct. 1964) 455–465, http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1473752.
[29] C. Heitzinger, C. Ringhofer, A transport equation for confined structures derived from the Boltzmann equation, Commun. Math. Sci. 9 (3) (2011) 829–857, http://intlpress.com/site/pub/files/_fulltext/journals/cms/2011/0009/0003/CMS-2011-0009-0003-a008.pdf.
[30] S. Howorka, Z.S. Siwy, Nanopores as protein sensors, Nat. Biotechnol. 30 (6) (Jun. 2012) 506–507, http://dx.doi.org/10.1038/nbt.2264.
[31] J.P. Hsu, Z.S. Chen, S. Tseng, Effect of electroosmotic flow on the electrophoresis of a membrane-coated sphere along the axis of a cylindrical pore, J. Phys. Chem. B 113 (21) (2009) 7701–7708.
[32] T.J.R. Hughes, L.P. Franca, A new finite element formulation for computational fluid dynamics: VII. The Stokes problem with various well-posed boundary conditions: symmetric formulations that converge for all velocity/pressure spaces, Comput. Methods Appl. Mech. Eng. 65 (1) (Nov. 1987) 85–96, http://www.sciencedirect.com/science/article/pii/0045782587901848.
[33] J. Jansson, J. Hoffman, C. Degirmenci, Adaptive Error Control in Finite Element Methods Using the Error Representation as Error Indicator, Tech. rep., KTH Royal Institute of Technology, 2013.
[34] E. Karatay, C.L. Druzgalski, A. Mani, Simulation of chaotic electrokinetic transport: performance of commercial software versus custom-built direct numerical simulation codes, J. Colloid Interface Sci. 446 (May 2015) 67–76, http://www.sciencedirect.com/science/article/pii/S0021979714010510.
[35] T. Kerkhoven, On the effectiveness of Gummel's method, SIAM J. Sci. Stat. Comput. 9 (1) (1988) 48–60, http://epubs.siam.org/doi/abs/10.1137/0909005.
[36] U.F. Keyser, S. van Dorp, S.G. Lemay, Tether forces in DNA electrophoresis, Chem. Soc. Rev. 39 (3) (Mar. 2010) 939–947, http://www.ncbi.nlm.nih.gov/pubmed/20179816.
[37] A. Khodadadian, C. Heitzinger, A transport equation for confined structures applied to the OprP, Gramicidin A, and KcsA channels, J. Comput. Electron. 14 (2) (2015) 524–532, http://link.springer.com/article/10.1007/s10825-015-0680-6.
[38] S. Kuyucak, O.S. Andersen, S.-H. Chung, Models of permeation in ion channels, Rep. Prog. Phys. 64 (11) (Nov. 2001) 1427–1472, http://iopscience.iop.org/article/10.1088/0034-4885/64/11/202.
[39] P. Langevin, Sur la theorie du mouvement Brownien, C. R. Acad. Sci. (Paris) 146 (530–533) (1908) 530–533.
[40] N. Laohakunakorn, U.F. Keyser, Electroosmotic flow rectification in conical nanopores, Nanotechnology 26 (27) (Jul. 2015) 275202, http://www.ncbi.nlm.nih.gov/pubmed/26087132.
[41] A. Logg, K.-A. Mardal, G.N. Wells, Automated Solution of Differential Equations by the Finite Element Method, Lecture Notes in Computational Science and Engineering, vol. 84, Springer Science & Business Media, 2011.
[42] B. Lu, M.J. Holst, J. Andrew McCammon, Y.C. Zhou, Poisson–Nernst–Planck equations for simulating biomolecular diffusion–reaction processes I: finite element solutions, J. Comput. Phys. 229 (19) (Sep. 2010) 6979–6994, http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=2922884&tool=pmcentrez&rendertype=abstract.
[43] B. Lu, D.P. Hoogerheide, Q. Zhao, D. Yu, Effective driving force applied on DNA inside a solid-state nanopore, Phys. Rev. E 86 (1) (Jul. 2012) 011921, http://journals.aps.org/pre/abstract/10.1103/PhysRevE.86.011921.
[44] B. Lu, Y.C. Zhou, G.A. Huber, S.D. Bond, M.J. Holst, J.A. McCammon, Electrodiffusion: a continuum modeling framework for biomolecular systems with realistic spatiotemporal resolution, J. Chem. Phys. 127 (13) (Oct. 2007) 135102, http://scitation.aip.org/content/aip/journal/jcp/127/13/10.1063/1.2775933.
[45] C. Maffeo, S. Bhattacharya, J. Yoo, D. Wells, A. Aksimentiev, Modeling and simulation of ion channels, Chem. Rev. 112 (12) (Dec. 2012) 6250–6284, http://dx.doi.org/10.1021/cr3002609.
[46] M. Mao, S. Ghosal, G. Hu, Hydrodynamic flow in the vicinity of a nanopore induced by an applied voltage, Nanotechnology 24 (24) (Jun. 2013) 245202, http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3738177&tool=pmcentrez&rendertype=abstract.
[47] M. Mao, J.D. Sherwood, S. Ghosal, Electro-osmotic flow through a nanopore, J. Fluid Mech. 749 (May 2014) 167–183, http://www.journals.cambridge.org/abstract_S0022112014002146.

[48] K.-A. Mardal, R. Winther, Preconditioning discretizations of systems of partial differential equations, Numer. Linear Algebra Appl. 18 (1) (Jan. 2011) 1–40, http://eprints.maths.ox.ac.uk/1499/.
[49] M.S. Metti, J. Xu, C. Liu, Energetically stable discretizations for charge transport and electrokinetic models, J. Comput. Phys. 306 (Feb. 2016) 1–18, http://www.sciencedirect.com/science/article/pii/S0021999115007305.
[50] S.Y. Noskov, W. Im, B. Roux, Ion permeation through the alpha-hemolysin channel: theoretical studies based on Brownian dynamics and Poisson–Nernst–Plank electrodiffusion theory, Biophys. J. 87 (4) (Oct. 2004) 2299–2309, http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=1304654&tool=pmcentrez&rendertype=abstract.
[51] P.L. Paine, P. Scherr, Drag coefficients for the movement of rigid spheres through liquid-filled cylindrical pores, Biophys. J. 15 (10) (Oct. 1975) 1087–1091, http://dx.doi.org/10.1016/S0006-3495(75)85884-X.
[52] A. Prohl, M. Schmuck, Convergent finite element discretizations of the Navier–Stokes–Nernst–Planck–Poisson system, ESAIM: Math. Model. Numer. Anal. 44 (3) (Feb. 2010) 531–571, http://www.esaim-m2an.org/10.1051/m2an/2010013.
[53] S. Qian, A. Wang, J.K. Afonien, Electrophoretic motion of a spherical particle in a converging-diverging nanotube, J. Colloid Interface Sci. 303 (2) (2006) 579–592.
[54] J. Quick, et al., Real-time, portable genome sequencing for Ebola surveillance, Nature 530 (7589) (2016) 228–232, http://www.nature.com/doifinder/10.1038/nature16996.
[55] M.E. Rognes, A. Logg, Automated goal-oriented error control I: stationary variational problems, SIAM J. Sci. Comput. 35 (3) (Apr. 2013) C173–C193, http://epubs.siam.org/doi/abs/10.1137/10081962X.
[56] I. Rubinstein, Electro-Diffusion of Ions, SIAM, 1990.
[57] R. Sacco, P. Airoldi, A.G. Mauri, J.W. Jerome, Three-dimensional simulation of biological ion channels under mechanical, thermal and fluid forces, arXiv preprint, http://arxiv.org/abs/1509.07301, Sep. 2015, pp. 1–38.
[58] G.F. Schneider, C. Dekker, DNA sequencing with nanopores, Nat. Biotechnol. 30 (4) (Apr. 2012) 326–328, http://dx.doi.org/10.1038/nbt.2181.
[59] J.W. Slotboom, Computer-aided two-dimensional analysis of bipolar transistors, IEEE Trans. Electron Devices 20 (8) (Aug. 1973) 669–679, http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1477384.
[60] L. Song, M.R. Hobaugh, C. Shustak, S. Cheley, H. Bayley, J.E. Gouaux, Structure of staphylococcal alpha-hemolysin, a heptameric transmembrane pore, Science 274 (5294) (Dec. 1996) 1859–1866, http://www.ncbi.nlm.nih.gov/pubmed/8943190.
[61] H.A. van der Vorst, Bi-CGStab: a fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems, SIAM J. Sci. Stat. Comput. 13 (2) (1992) 631–644.
[62] S. van Dorp, U.F. Keyser, N.H. Dekker, C. Dekker, S.G. Lemay, Origin of the electrophoretic force on DNA in solid-state nanopores, Nat. Phys. 5 (5) (2009) 347–351, http://dx.doi.org/10.1038/nphys1230.
[63] I. Vlassiouk, S. Smimov, Z. Siwy, Nanofluidic ionic diodes. Comparison of analytical and numerical solutions, ACS Nano 2 (8) (Aug. 2008) 1589–1602, http://dx.doi.org/10.1021/nn800306u.
[64] I. Vlassiouk, S. Smimov, Z. Siwy, Ionic selectivity of single nanochannels, Nano Lett. 8 (7) (Jul. 2008) 1978–1985, http://dx.doi.org/10.1021/nl800949k.
[65] G.-W.W. Wei, Q. Zheng, Z. Chen, K. Xia, Variational multiscale models for charge transport, SIAM Rev. 54 (4) (Jan. 2012) 699–754.
[66] R. Wei, V. Gatterdam, R. Wieneke, R. Tampé, U. Rant, Stochastic sensing of proteins with receptor-modified solid-state nanopores, Nat. Nanotechnol. 7 (4) (Apr. 2012) 257–263, http://www.ncbi.nlm.nih.gov/pubmed/22406921.