hp-Finite Element Model of Poisson and Nernst-Planck System of Equations

D. Pugal[a,d], P. Solin[b,c,**], K. J. Kim[a,*], A. Aabloo[d]

[a] *Mechanical Engineering Department, University of Nevada, Reno, NV, U.S.A.*
[b] *FEM group, Department of Mathematics and Statistics, University of Nevada, Reno, NV, U.S.A.*
[c] *Institute of Thermomechanics, Prague, Czech Republic*
[d] *Institute of Technology, Tartu University, Estonia*

Abstract

In this paper we present a *hp*-finite element model (*hp*-FEM) of Nernst-Planck and Poisson equation system. The model is implemented in Hermes2D which is a space- and space-time adaptive *hp*-FEM solver. The time dependent adaptivity is used to control the error of the solution. Full mathematical derivation of the weak formulation of the system of equations and the solution comparison with a nonadaptive conventional FEM is presented. Furthermore, we extend the discussion on Hermes3D as a modeling tool that can help in studying and improving the IPMC materials.

Keywords: *hp*-FEM, Nernst-Planck, Poisson, weak form

1. Introduction

The system of Poisson and Nernst-Planck (further denoted PNP) equations has been widely used to describe the charge transport — which includes the ionic migration, diffusion, and convection — in a medium. The charge transport process is a key mechanism for electromechanical transduction in some of the electroactive polymer (EAP) materials, for instance, in ionic polymer-metal composites (IPMCs) [1, 2, 3, 4, 5, 6, 7]. As the system of equations is nonlinear and for a domain with two electrodes, the charge concentration differences occur in a very narrow region near the boundaries, the required computing power for a full scale domain is, especially in 3D, rather significant. This requires a mesh that is optimal in terms of calculation time and calculation error.

The Nernst-Planck equation for a mobile species — in this case for counter ions — and without the convection term is:

$$\frac{\partial C}{\partial t} + \nabla \cdot (-D\nabla C - z\mu F C \nabla \phi) = 0, \tag{1}$$

where C is the counter ion concentration, D diffusion, μ mobility, F Faraday constant, ϕ voltage, and z charge number. The Poisson equation is

$$-\nabla^2 \phi = \frac{F\rho}{\varepsilon}, \tag{2}$$

where ε is an absolute dielectric permittivity and the charge density ρ is expressed

$$\rho = C - C_0 \tag{3}$$

*Corresponding author
**Principal correpsonding auhtor
Email addresses: david.pugal@gmail.com (D. Pugal), solin@unr.edu (P. Solin), kwangkim@unr.edu (K. J. Kim), alvo@ut.ee (A. Aabloo)

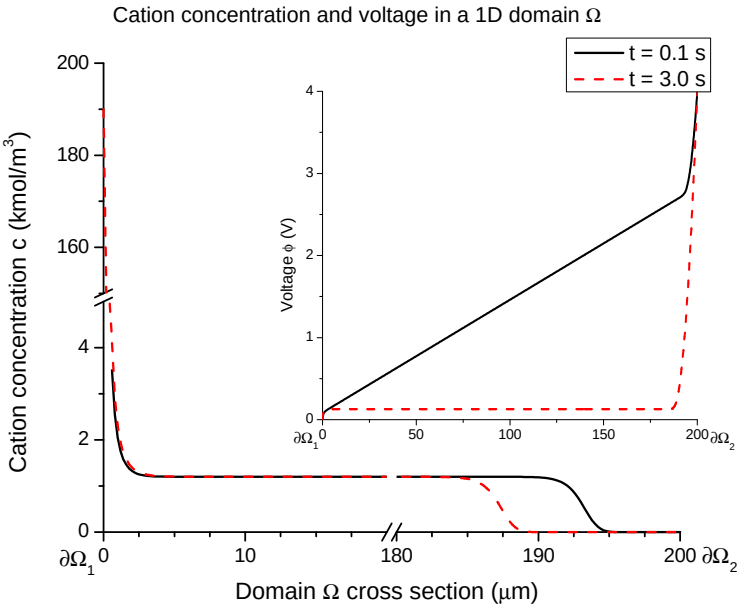


Figure 1: Calculated concentration C and voltage ϕ in a 1D domain $\Omega \subset \mathbb{R}$. Dirichlet boundary conditions ($V_{\partial\Omega_1} = 0\ V$ and $V_{\partial\Omega_2} = 4\ V$) were applied to the Poisson equation Eq. (2) and Neumann boundaries to the Nernst-Planck equation Eq. (1).

with C_0 being a constant anion concentration.

In the following section we give a brief overview of the motivation of the study and a practical application of the results. Thereafter the weak-form [8] derivation of the PNP system of equations and time dependent adaptive *hp*-FEM solutions of the system is presented for different adaptivity algorithms. Also, advantages of a multi-mesh solution over a single-mesh solution are discussed.

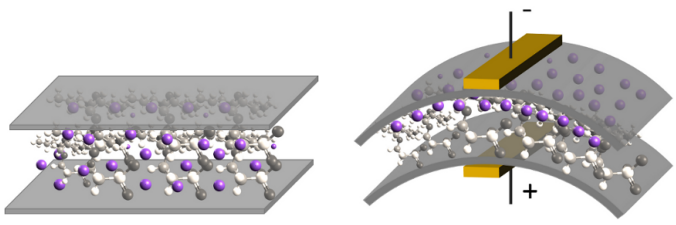
2. Motivation

Fig. 1 shows a solution for C and ϕ in the time dependent NPN system at $t = 0.1\ s$ and $t = 3.0\ s$. Solution constants are shown in Table 1. It can be seen that the solution has two notable characteristics. Firstly, ∇C at $\partial\Omega_2$ is moving in time, whereas ∇C at $\partial\Omega_1$ is very sharp. For the large part of the domain Ω $\nabla C = 0$. At the same time, ϕ is a rather smooth fucntion in Ω. This clearly makes the choice of mesh for the time dependent NPN system difficult. By creating a too fine mesh, the problem gets too big. By choosing a coarser mesh, the calculation error goes up. Furthermore, from the shape of the solution in Fig. 1 can be seen that the polynomial degree of the elements in the middle of the domain Ω and near the boundaries $\partial\Omega_1$, $\partial\Omega_2$ should be different — namely higher degree near the boundaries. Overall shape of the solutions suggest that using different meshes for variables C and ϕ can be beneficial in terms of resources and calculation time.

All the aforementioned difficulties can be solved by using a time dependent adaptive multimesh *hp*-FEM solver. Automatic time dependent adaptivity in conjuction with *hp*-FEM helps to limit the error of the calculations by choosing a suitable mesh and polynomial degrees of the elements at each time step. In this work Hermes2D [9] was used to study the problem size, error convergence and solution time of NPN system with different adaptivity algorithms.

Table 1: Constants used in the Poisson-Nernst-Planck system of equations.

Constant	Value	Unit	Description
D	10×10^{-11}	$\frac{m^2}{s}$	Diffusion constant
z	1	-	Charge number
F	96,485	$\frac{C}{mol}$	Faraday number
R	8.31	$\frac{J}{mol \cdot K}$	The gas constant
$\mu \ \left(= \frac{D}{RT}\right)$	4.11×10^{-14}	$\frac{s}{\frac{mol}{m^3} \cdot K}$	Mobility
C_0	1,200	$\frac{mol}{m^3}$	Anion concentration
ε	0.025	$\frac{F}{m}$	Electric permittivity



(a) (b)

Figure 2: Conceptual model of the actuation of IPMC. Initial counter ion distribution (a) and the distribution and resulting bending after applying a voltage (b).

2.1. Practical application

IPMCs have been studied in past two decades in view of using them as noiseless mechanoelectrical or electromechanical transducers. The advantages of the IPMCs over other electroactive polymer actuators are low voltage bending, high strains ($> 1\%$), and ability to work in wet environments. A typical IPMC consists of a thin sheet of polymer (often Nafion or Teflon) which is sandwiched between noble metal electrodes such as platinum or gold. When fabricated, the polymer membrane is saturated with certain solvent and ions, e.g water and H^+. When a voltage is applied to the eletrodes, the counter ions start migrating due to the imposed electric field. By dragging along the solvent, the osmotic pressure difference near the electrodes occur which in turn results in bending of the material (see Fig. 2). The derived and implemented model helps to predict the actuation of the material. Furthermore, it is expected that the *hp*-FEM implementation results in a smaller problem size, thus likely allowing faster calculation in both 2D and 3D.

3. Model

The model for solving NPN system is implemented in Hermes2D. In this work a rectangular 2D domain $\Omega \subset \mathbb{R}^2$ with boundaries $\partial\Omega_{1\ldots4} \subset \partial\Omega$ is considered (see Fig. 3). As there is now flow considered in or out of the domain and there is no constant concentration at the boundaries, zero Neumann boundary conditions

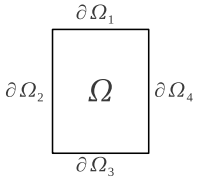


Figure 3: Calculation domain $\Omega \subset \mathbb{R}^2$ with boundaries $\partial\Omega_{1\ldots4} \subset \partial\Omega$.

are used for Eq. (1):

$$-D\frac{\partial C}{\partial n} - z\mu FC\frac{\partial \phi}{\partial n} = 0. \tag{4}$$

As a positive voltage V_{pos} is applied on Ω_1 and $V_{neg} = 0$ is applied on Ω_3, Dirichlet boundary conditions are used for Eq. (2) for boundaries Ω_1 and Ω_3:

$$\phi_{\partial\Omega_1} = V_{pos}, \tag{5}$$
$$\phi_{\partial\Omega_3} = 0, \tag{6}$$

and Neumann boundaries for Ω_2 and Ω_4:

$$\frac{\partial \phi_{\Omega_2}}{\partial n} = \frac{\partial \phi_{\Omega_4}}{\partial n} = 0. \tag{7}$$

3.1. Weak form of Poisson-Nernst-Planck system

To make the derivation of the weak forms more convenient, the following helper constants will be used:

$$K = z\mu F, \tag{8}$$
$$L = \frac{F}{\varepsilon}. \tag{9}$$

So Eq. (1) and Eq. (2) become after substituting Eq. (3) and the constants K and L:

$$\frac{\partial C}{\partial t} - D\nabla^2 C - K\nabla\cdot(C\nabla\phi) = 0, \tag{10}$$
$$-\nabla^2\phi - L\left(C - C_0\right) = 0. \tag{11}$$

The boundary condition Eq. (4) becomes:

$$-D\frac{\partial C}{\partial n} - KC\frac{\partial \phi}{\partial n} = 0. \tag{12}$$

The space for solution is $V = H^1(\Omega)$ where $H^1(\Omega) = \left\{v \in L^2(\Omega);\ \nabla v \in \left[L^2(\Omega)\right]^2\right\}$. Let's choose a test function $v^C \in V$. The weak form the Nernst-Planck equation Eq. (10) is found by multiplying it with the test function v^C and then integrating over the domain Ω:

$$\int_\Omega \frac{\partial C}{\partial t}v^C d\mathbf{x} - \int_\Omega D\nabla^2 C v^C d\mathbf{x} - \int_\Omega K\nabla C\cdot\nabla\phi v^C d\mathbf{x} - \int_\Omega KC\nabla^2\phi v^C d\mathbf{x} = 0. \tag{13}$$

After adding the weak form of the boundary term (Eq. (12)) and applying the Green's first identity to the terms that contain Laplacian ∇^2 we get

$$\int_\Omega \frac{\partial C}{\partial t}v^C d\mathbf{x} + D\int_\Omega \nabla C\cdot\nabla v^C d\mathbf{x} - K\int_\Omega \nabla C\cdot\nabla\phi v^C d\mathbf{x} + K\int_\Omega \nabla\left(Cv^C\right)\cdot\nabla\phi d\mathbf{x} \tag{14}$$
$$-D\int_{\partial\Omega}\frac{\partial C}{\partial n}v^C d\mathbf{S} - \int_{\partial\Omega} K\frac{\partial\phi}{\partial n}Cv^C d\mathbf{S} = 0. \tag{15}$$

After expanding the nonlinear term and given that the boundary terms do not contribute, the weak form becomes

$$\int_\Omega \frac{\partial C}{\partial t}v^C d\mathbf{x} + D\int_\Omega \nabla C\cdot\nabla v^C d\mathbf{x} - K\int_\Omega \nabla C\cdot\nabla\phi v^C d\mathbf{x} + K\int_\Omega \nabla\phi\cdot\nabla C v^C d\mathbf{x} + K\int_\Omega C\left(\nabla\phi\cdot\nabla v^C\right)d\mathbf{x} = 0. \tag{16}$$

As the second and third term cancel out, the final weak from of the Nernst-Planck equation is

$$\int_\Omega \frac{\partial C}{\partial t}v^C d\mathbf{x} + D\int_\Omega \nabla C\cdot\nabla v^C d\mathbf{x} + K\int_\Omega C\left(\nabla\phi\cdot\nabla v^C\right)d\mathbf{x} = 0. \tag{17}$$

Similarly the weak form of Poisson equation Eq. 11 with a test function $v^\phi \subset V$ is:

$$-\int_\Omega \nabla^2\phi v^\phi d\mathbf{x} - \int_\Omega LCv^\phi d\mathbf{x} + \int_\Omega LC_0 v^\phi d\mathbf{x} = 0. \tag{18}$$

After expanding the ∇^2 terms, the final form becomes

$$\int_\Omega \nabla\phi\cdot\nabla v^\phi d\mathbf{x} - \int_\Omega LCv^\phi d\mathbf{x} + \int_\Omega LC_0 v^\phi d\mathbf{x} = 0. \tag{19}$$

3.2. Implementation in Hermes2D

To implement the system of equations Eq. (17) and Eq. (19), the residuals and the Jacobian matrix must be derived. For that, Crank-Nicolson time stepping was used

$$\frac{\partial C}{\partial t} \approx \frac{C^{n+1}-C^n}{\tau}, \tag{20}$$

where τ is a time step. For the variables C^{n+1} and ϕ^{n+1} the following notation will be used:

$$C^{n+1} = \sum_{k=1}^{N^C} y_k^C v_k^C, j \tag{21}$$

$$\phi^{n+1} = \sum_{k=1}^{N^\phi} y_k^\phi v_k^\phi, \tag{22}$$

where v_k^C and v_k^ϕ are piecewise polynomial functions in V. Considering the Crank-Nicolson time stepping and the notation (21), the time discretized Eq. (17) becomes

$$\begin{aligned} F_i^C(Y) = &\int_\Omega \frac{C^{n+1}}{\tau} v_i^C d\mathbf{x} - \int_\Omega \frac{C^n}{\tau} v_i^C d\mathbf{x} \\ &+\frac{1}{2}\left[D\int_\Omega \nabla C^{n+1}\cdot\nabla v_i^C d\mathbf{x} + D\int_\Omega \nabla C^n\cdot\nabla v_i^C d\mathbf{x}\right] \\ &+\frac{1}{2}\left[K\int_\Omega C^{n+1}\left(\nabla\phi^{n+1}\cdot\nabla v_i^C\right)d\mathbf{x} + K\int_\Omega C^n\left(\nabla\phi^n\cdot\nabla v_i^C\right)d\mathbf{x}\right], \end{aligned} \tag{23}$$

and in the notation (22), Eq. (19) becomes

$$F_i^\phi(Y) = \int_\Omega \nabla\phi^{n+1}\cdot\nabla v_i^\phi d\mathbf{x} - \int_\Omega LC^{n+1} v_i^\phi d\mathbf{x} + \int_\Omega LC_0 v_i^\phi d\mathbf{x}. \tag{24}$$

For the implementation the 2×2 Jacobian matrix DF/DY with elements corresponding to

$$\frac{\partial F_i^C}{\partial y_j^C},\ \frac{\partial F_i^C}{\partial y_j^\phi},\ \frac{\partial F_i^\phi}{\partial y_j^C},\ \frac{\partial F_i^\phi}{\partial y_j^\phi}, \tag{25}$$

must be derived:

$$\frac{\partial F_i^C}{\partial y_j^C} = \int_\Omega \frac{1}{\tau} v_j^C v_i^C d\mathbf{x} + \frac{1}{2}D\int_\Omega \nabla v_j^C\cdot\nabla v_i^C d\mathbf{x} + \frac{1}{2}K\int_\Omega v_j^C\left(\nabla\phi^{n+1}\cdot\nabla v_i^C\right)d\mathbf{x}, \tag{26}$$

$$\frac{\partial F_i^C}{\partial y_j^\phi} = \frac{1}{2}K\int_\Omega C^{n+1}\left(\nabla v_j^\phi\cdot\nabla v_i^C\right)d\mathbf{x}, \tag{27}$$

$$\frac{\partial F_i^\phi}{\partial y_j^C} = -\int_\Omega L v_j^C v_i^\phi d\mathbf{x}, \tag{28}$$

$$\frac{\partial F_i^\phi}{\partial y_j^\phi} = \int_\Omega \nabla v_j^\phi\cdot\nabla v_i^\phi d\mathbf{x}. \tag{29}$$

In Hermes2D Eq. (23) and (24) define the residuum F and Eq. (26)—(29) define the Jacobian matrix J.

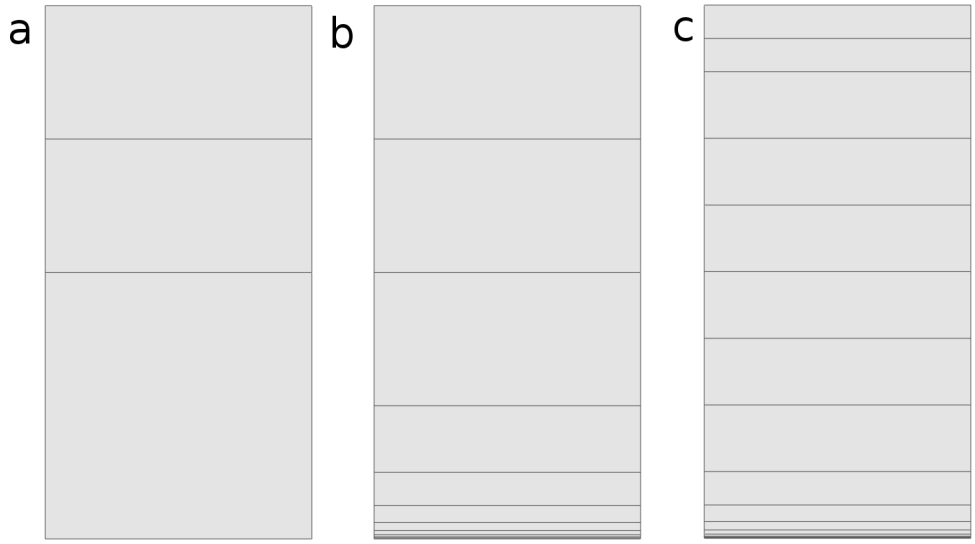


Figure 4: Initial coarse mesh (a), half refined mesh (b) and refined mesh (c). The coarse mesh and refined mesh were used in the initial calcualtions, the latter one in case of p-adaptivity (including HP_ANISO_P). The half-refined mesh was used later to optimize hp-adaptive mode solutions.

3.3. Adaptive multi-mesh solution

The defined Jacobian J and residuum F can be simply solved in Hermes2D by using Newton's iteration. However, we were more interested in the automatic adaptivity and the calculation differences with a single-mesh and multi-mesh.

In case of the single mesh, the same basis function base was used, i.e. $v_i^\phi = v_i^C$. For multi mesh, the meshes and therefore the basis functions were different. Regardless of the meshing, automatic adaptivity was applied during each time step, till the error converged to the acceptable level. In analogy to the most successful adaptive ODE solvers, Hermes2D uses a pair of approximations with different orders of accuracy to obtain this information: coarse mesh solution and fine mesh solution. The initial coarse mesh is read from the mesh file, and the initial fine mesh is created through its global refinement both in h and p. For more information, see [9]. In the calculations, the coarse mesh solution was not calculated at each time step, but the fine mesh solution was projected to the coarse mesh by using orthogonal projection. This yielded better convergence and also faster calculation.

Hermes2D supports 8 different refinement modes, namely, 3 isotropic and 5 anisotropic refinements. The isotropic refinements are h-isotropic (H_ISO), p-isotropic (P_ISO), hp-isotropic (HP_ISO). Anisotropic refinement modes are h-anisotropic (H_ANISO), hp-anisotropic-h (HP_ANISO_H), p-anisotropic (P_ANISO), hp-anisotropic-p (HP_ANISO_P), and hp-anisotropic (HP_ANISO). Once the refinement mode is selected (by user), Hermes2D selects a particular refinement scheme from several candiates based on the score. More information can be found in [9].

4. Results

We ran the calculation for all the adaptivity types in both single-mesh and multi-mesh configurations. The following numerical results were recorded for each adaptivity type: converged relative error, cumulative CPU time, and the problem size in terms of number of degrees of freedoms (NDOFs) at each time step. Two types of initial meshes were used — in case of only p-adaptivity, more refined mesh was used (see Fig. 4 (b)) to ensure the error convergence. When also the element size refinement was enabled (all h adaptivity types), very coarse initial mesh was used (see Fig. 4 (a)) to let the adaptivity algorithm find the most optimal mesh. In both case, the initial mesh was loaded at each time step of the calculation.

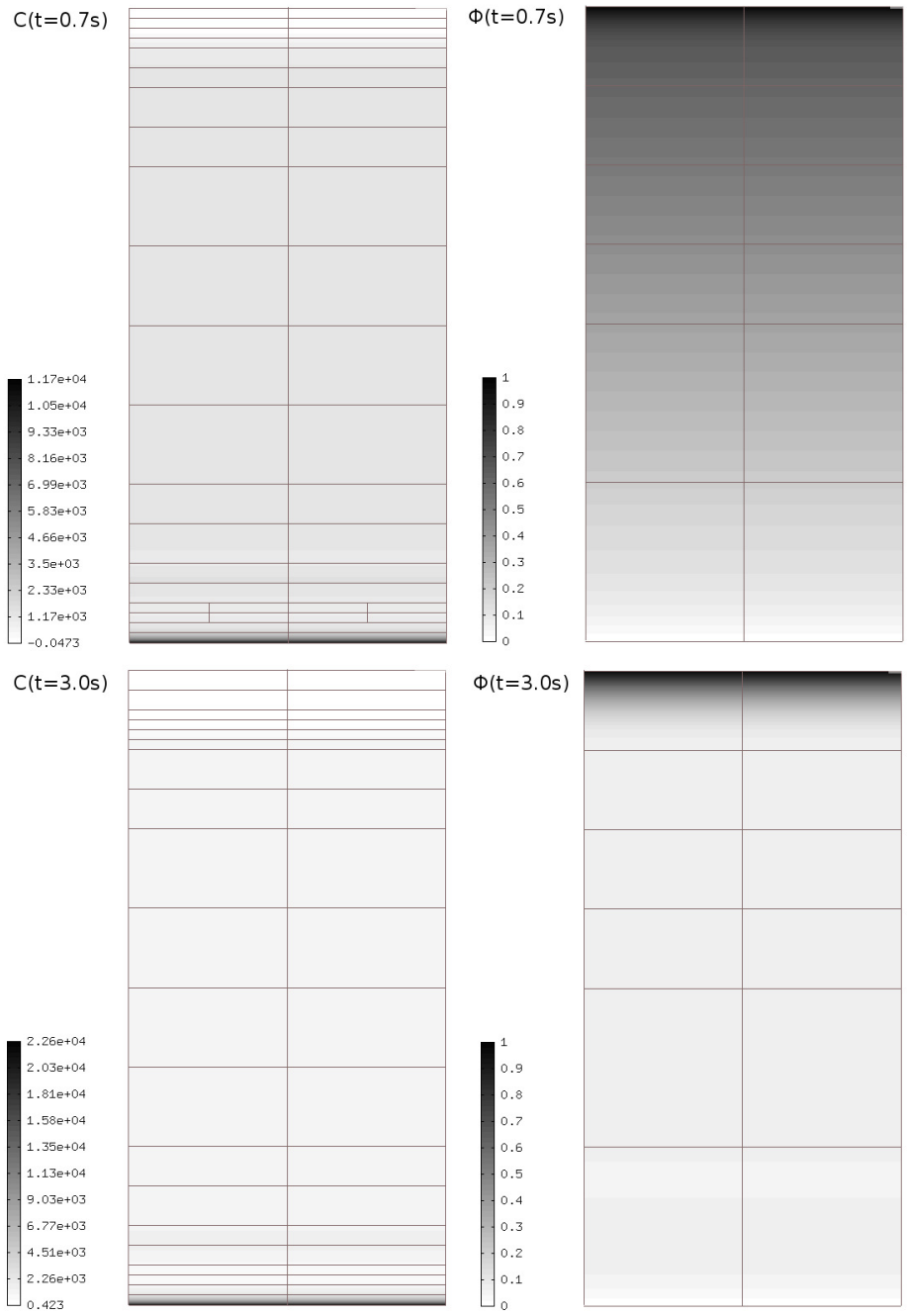


Figure 5: Concentration C and voltage ϕ at two different time steps (HP_ANISO adaptivity was used).

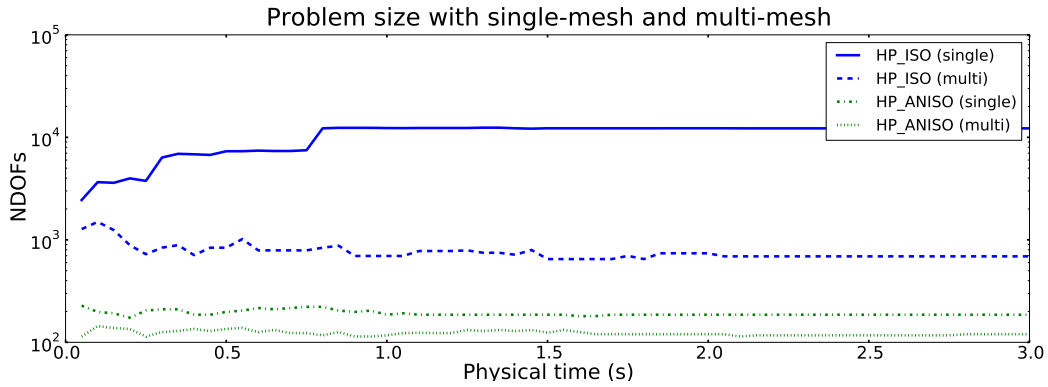


Figure 6: NDOFs in case of single-mesh and multi-mesh solutions with HP_ISO and HP_ANISO adaptivities. (Notice log Y scale)

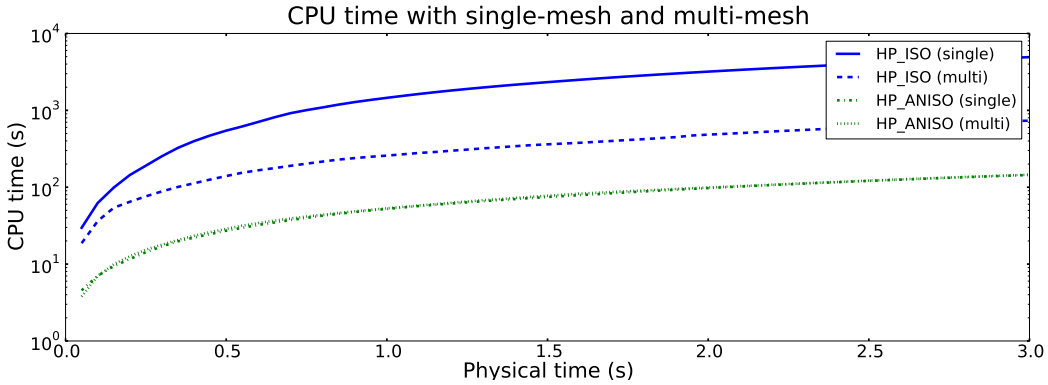


Figure 7: CPU times in case of single-mesh and multi-mesh solutions with HP_ISO and HP_ANISO adaptivities. (Notice log Y scale)

Example of the solution at $t = 0.7\ s$ and $t = 3.0\ s$ with adaptivity type HP_ANISO is shown in Fig. 5. The time $t = 0.7\ s$ was chosen because by that time, some ionic migration has already taken place, i.e. concentration gradient near the $\partial\Omega_1$ and $\partial\Omega_3$ has formed. The automatic mesh refinements at different time steps are clearly visible — especially near the top boundary, where the concentration gradient is moving in time (as seen in Fig. 1).

The following subsections provide a detailed comparison of different adaptivity types and estimate the most suitable one for the given problem.

4.1. Optimal adaptivity types

Running the simulation with different adaptivity types and meshes showed that the multi-mesh configuration generally results in smaller problem size, faster calculation, and better or similar error convergence compared to the single-mesh configuration. This is well illustrated in Fig. 6 and Fig. 7. This can be well understood from Fig. 1 — in the boundary regions $\partial\Omega_1$ and $\partial\Omega_3$ the concentration gradient is greater than voltage gradient, namely $\nabla C >> \nabla\phi$. Therefore refining the mesh for both variables is not reasonable. For instance, the solution space with corresponding mesh in case of HP_ANISO at $t = 0.7\ s$ is shown in Fig. 5. The corresponding polynomial degree space is shown in Fig. 8. Notice that the adaptive algorithm has increased the maximum polynomial degree for C space to 7, however, at the same time, maximum polynomial degree for ϕ is one. Furthermore, the mesh is more refined for C. For most cases using the multi-mesh results in similar or better CPU time, i.e. multi-mesh takes less computational resources. Only

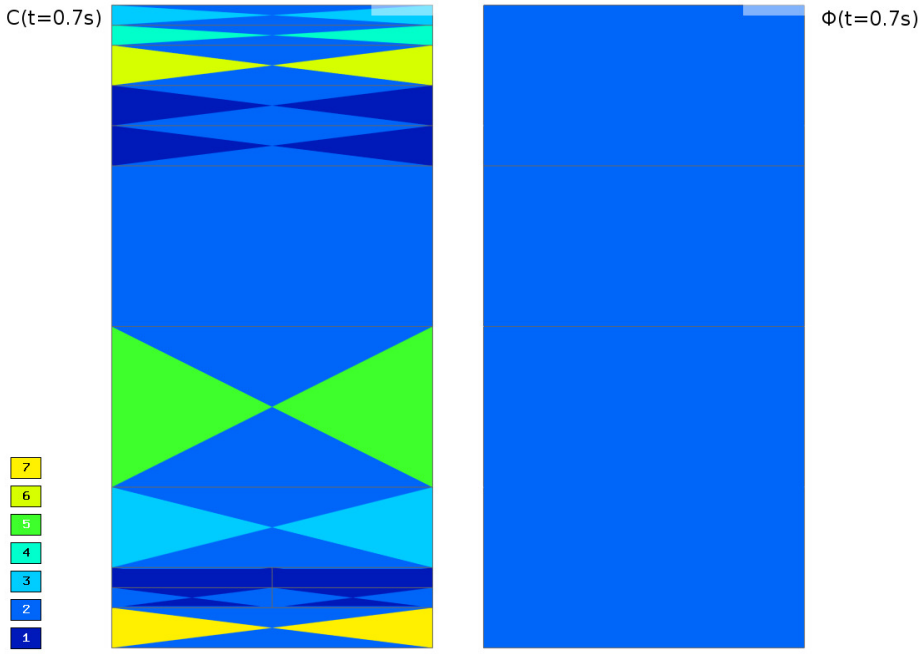


Figure 8: Polynomial degree space for C and ϕ at $t = 0.7\ s$ The color indicates the maximum polynomial degree of the corresponding element.

excpetion was HP_ANISO_H adaptivity type for which the single mesh configuration resulted slightly faster calculation time. Based on the results only multi-mesh configurations will be considered in the following comparisons.

To narrow down the list of adaptivity modes for the given problem, first the isotropic and anisotropic adaptivities were compared. Here p-adaptivity modes are compared separately as they use more refined mesh. Fig. 9 and Fig. 10 show the comparison of H_ISO, H_ANISO, HP_ISO, HP_ANISO, and HP_ANISO_H modes in terms of CPU time and problem size. Fig. 11 and Fig. 12 show the similar comparsion for the P_ISO, P_ANISO, and HP_ANISO_P modes. Here we see that the anisotropic adaptivities result in a reasonable problem size and the problems solve within a reasonable calculation time. It is interesting to note that HP_ANISO results in the smallest problem size. Also, in p-adaptivity group, HP_ANISO_P results in the smallest problem size at each time step, whereas P_ISO and P_ANISO have a very largy problem size during the first time steps of the solution. Here the term "reasonable problem size" means here that the number of degrees of freedom in time converges to so that $N_{dof} < 500$, and the term "reasonable calculation time" means that the calculation (step $\tau = 0.01\ s$, physical time $t_{end} = 3.0\ s$) time t on a givem system was $t < 500\ s$. Although these parameters are empirical, they serve as an upper limit, given that the most adaptivity modes gave significantly smaller results, e.g. $t << 500\ s$ and $N_{dof} << 500$.

4.2. Quantitative analysis of different adaptivity types

Based on the results of the previous subsection, H_ANISO, HP_ANISO, and HP_ANISO_H from the *hp/h* adaptivity group and HP_ANISO_P from p adaptivity group will be compared in terms of problem size and cumulative CPU time. In all of the cases, the relative error at each time step remained below the threshold which were set to $e_{th} = 0.5\%$ between the coarse mesh and fine mesh solutions, therefore the error vs. time plot will not be considered.

Fig. 13 shows the cumulative CPU time for different adaptivity modes at each time step. All the calculations were done on the same computer. Here we see that HP_ANISO_H and HP_ANISO require the most resources which can be understood from the fact that these adaptivity modes have the largest

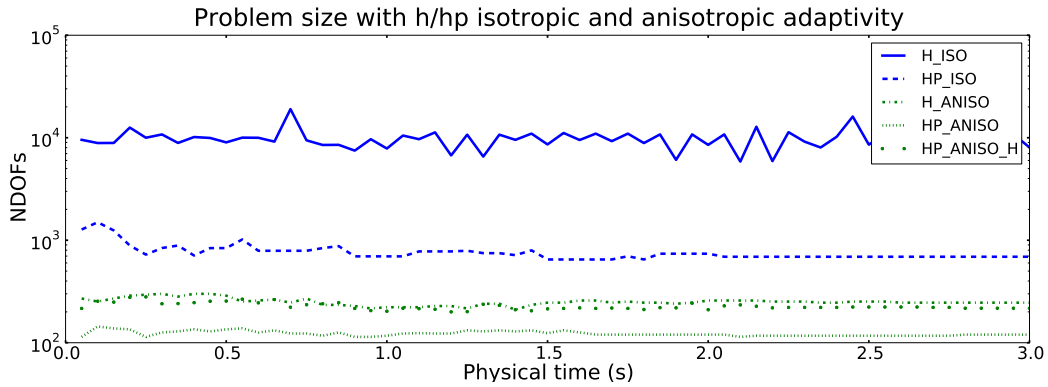


Figure 9: NDOFs in case of multi-mesh solutions with H_ISO, H_ANISO, HP_ISO, HP_ANISO, and single-mesh solution with HP_ANISO_H adaptivities. (Notice log Y scale)

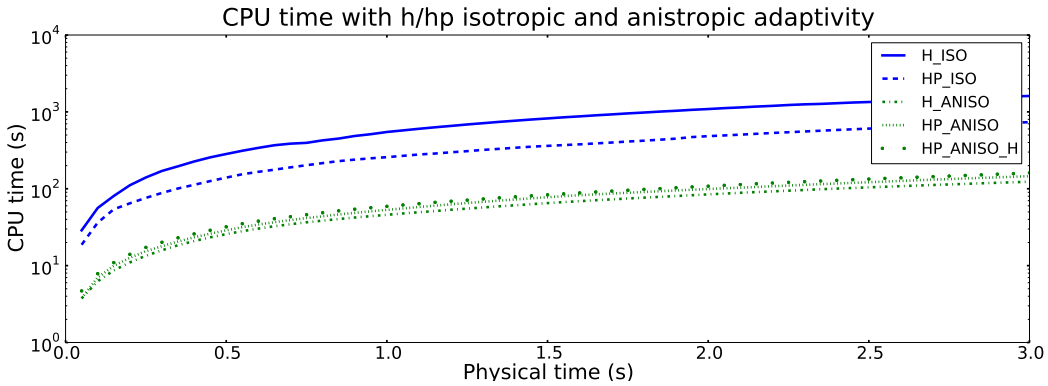


Figure 10: CPU times in case of multi-mesh solutions with H_ISO, H_ANISO, HP_ISO, HP_ANISO, and single-mesh solution with HP_ANISO_H adaptivities. (notice log Y scale)

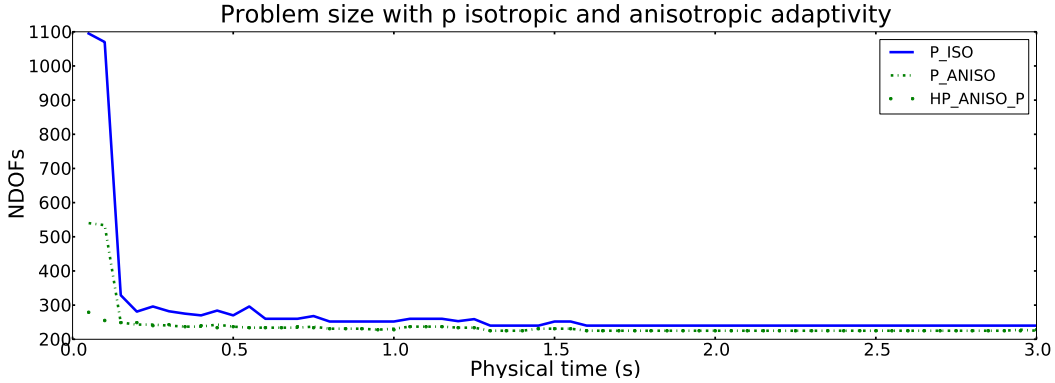


Figure 11: NDOFs in case of multi-mesh solutions with P_ISO, P_ANISO, and HP_ANISO_P adaptivities.

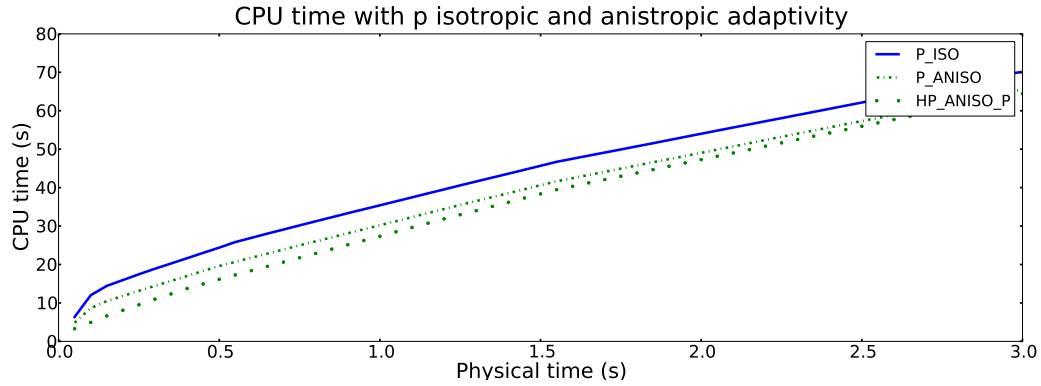


Figure 12: CPU times in case of multi-mesh solutions with P_ISO, P_ANISO, and HP_ANISO_P adaptivities.

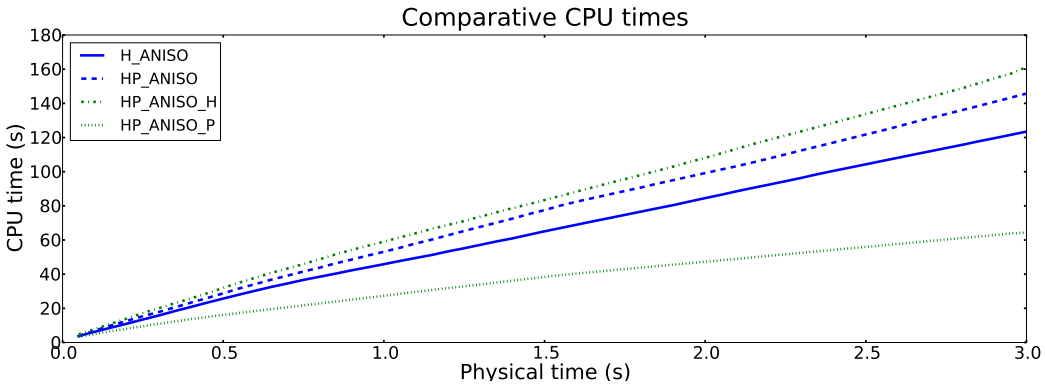


Figure 13: Comparative CPU time for different adaptivity modes.

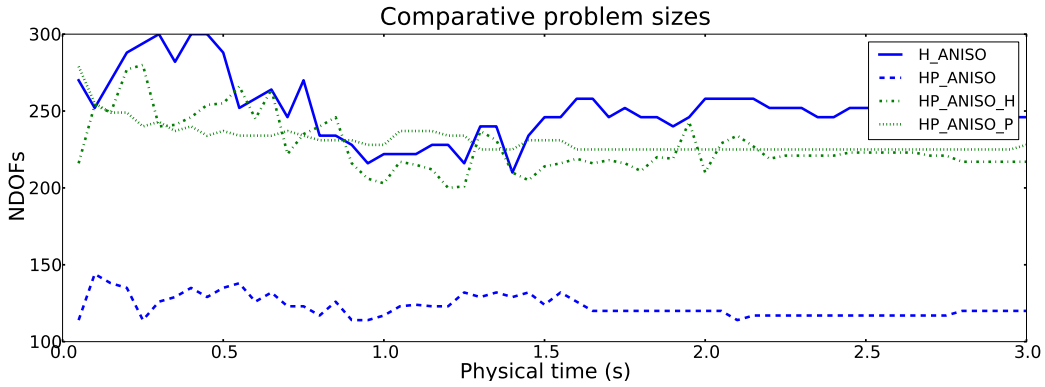


Figure 14: Comparative NDOFs at each time step for different adaptivity modes.

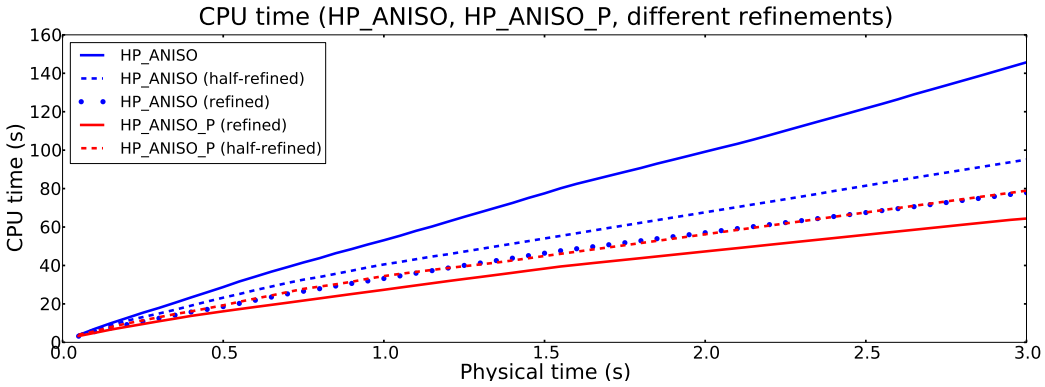


Figure 15: Cumulative CPU time for HP_ANISO and HP_ANISO_P with different initial meshes.

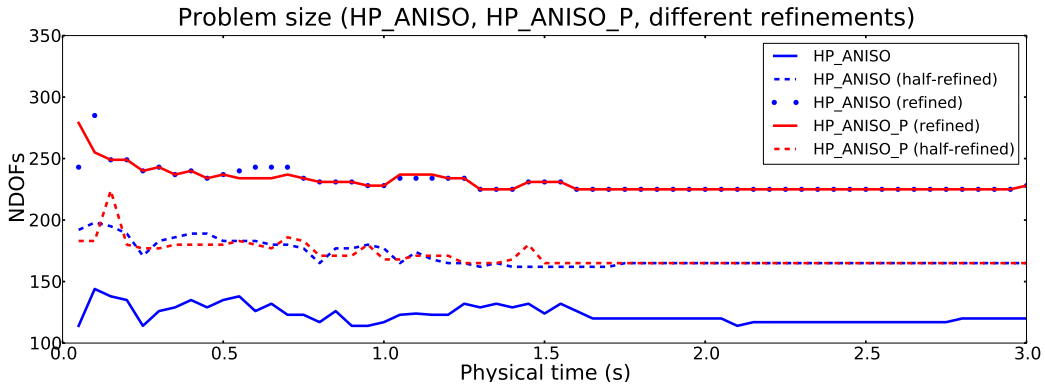


Figure 16: NDOFs at each time step for HP_ANISO and HP_ANISO_P with different meshes.

number of candidates (see XXX) from which the refinement methode is chosen. Fig. 14 shows the NDOFs at each time step for different adaptivity modes. There we see that the HP_ANISO results in the smallest problem size — $N_{dof} \approx 125$. All the other adaptivity modes result in the problem size of approximately ($N_{dof} \approx 250$),

So far we have seen that HP_ANISO results in the smallest problem size, but requires quite a lot CPU time. At the same time, HP_ANISO_P is the fastest, whereas it does not result in that small problem. When it comes to a large domain or 3D modeling the problem size becomes the most important factor. Therefore, we consider HP_ANISO the most suitable adaptivity type to the given problem. Thus some ways to optimize the time factor will be considered.

The performance of HP_ANISO_P with refined mesh and half-refined mesh, and performance of HP_ANISO with unrefined, half-refined, and refined meshes was calculated (see Fig. 4). The results are shown in Fig. 15 and Fig. 16.

TODO(more explanation) It can be seen that some initial refinemts help to reduce the CPU time, however, it increases the problem size.

5. Conclusion, future works

[1] S. Basu, M. Sharma, An improved space-charge model for flow through charged microporous membranes, Journal of Membrane Science 124 (1) (1997) 77–91.

[2] M. Shahinpoor, K. J. Kim, Ionic polymer-metal composites: I. fundamentals, Smart Materials and Structures 10 (4) (2001) 819.

[3] S. Nemat-Nasser, Micromechanics of actuation of ionic polymer-metal composites, Journal of Applied Physics 92 (5) (2002) 2899–2915.
[4] K. Newbury, D. Leo, Linear electromechanical model of ionic polymer transducers-Part I: Model Development, Journal of Intelligent Material Systems and Structures 14 (6) (2003) 333.
[5] T. Wallmersperger, D. J. Leo, C. S. Kothera, Transport modeling in ionomeric polymer transducers and its relationship to electromechanical coupling, Journal of Applied Physics 101 (2) (2007) 024912.
[6] D. Pugal, K. J. Kim, A. Punning, H. Kasemagi, M. Kruusmaa, A. Aabloo, A self-oscillating ionic polymer-metal composite bending actuator, Journal of Applied Physics 103 (8) (2008) 084908.
[7] D. Pugal, K. Jung, A. Aabloo, K. Kim, Ionic polymer-metal composite mechanoelectrical transduction: review and perspectives, Polymer international 59 (3) (2010) 279–289.
[8] P. Solin, K. Segeth, I. Dolezel, Higher-Order Finite Element Methods, Chapman & Hall / CRC Press, 2003.
[9] P. Solin, et al., Hermes - Higher-Order Modular Finite Element System (User's Guide).
URL `http://hpfem.org`