Comparison of the MSMS and NanoShaper molecular surface triangulation codes in the TABI Poisson–Boltzmann solver

Leighton Wilson*, Robert Krasny†

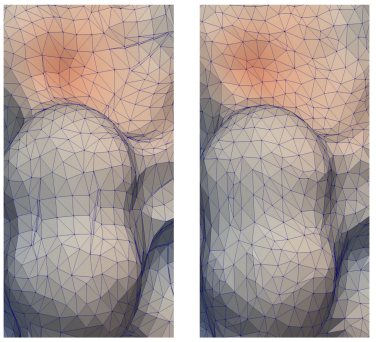
May 11, 2021

Abstract

The Poisson-Boltzmann (PB) implicit solvent model is a popular framework for studying the electrostatics of solvated biomolecules. In this model the dielectric interface between the biomolecule and solvent is often taken to be the molecular surface or solvent-excluded surface (SES), and the quality of the SES triangulation is critical in boundary element simulations of the model. This work compares the performance of the MSMS and NanoShaper surface triangulation codes for a set of 38 biomolecules. While MSMS produces triangles of exceedingly small area and large aspect ratio, the two codes yield comparable values for the SES surface area and electrostatic solvation energy, where the latter calculations were performed using the treecode-accelerated boundary integral (TABI) PB solver. However we found that NanoShaper is computationally more efficient and reliable than MSMS, especially when parameters are set to produce highly resolved triangulations.

Keywords: electrostatics, solvated biomolecule, solvent excluded surface, Poisson–Boltzmann, boundary element method, treecode

*Department of Mathematics, University of Michigan, Ann Arbor, MI 48109
†Department of Mathematics, University of Michigan, Ann Arbor, MI 48109



MSMS **NanoShaper**

The Poisson–Boltzmann (PB) implicit solvent model is a popular framework for studying the electrostatics of solvated biomolecules. This work compares the MSMS and NanoShaper molecular surface triangulation codes for a set of 38 biomolecules. The two codes yield comparable values for the molecular surface area and electrostatic solvation energy, where the latter was computed using the treecode-accelerated boundary integral (TABI) PB solver, although Nanoshaper is found to be computationally more efficient and reliable than MSMS.

INTRODUCTION

Implicit solvent models play an important role in computational modeling of electrostatic interactions between biomolecules and their solvent environment[1–3]. Of particular importance in this work is the Poisson–Boltzmann (PB) implicit solvent model[4,5]. Figure 1 shows the interior domain $\Omega_1 \subset \mathbb{R}^3$ containing the solute biomolecule, the exterior domain $\Omega_2 = \mathbb{R}^3 \setminus \overline{\Omega}_1$ containing the ionic solvent, and the dielectric interface $\Gamma = \overline{\Omega}_1 \cap \overline{\Omega}_2$. In a 1:1 electrolyte at low ionic concentration, the electrostatic potential ϕ satisfies the linear PB equation,

$$-\nabla \cdot (\varepsilon(\mathbf{x})\nabla\phi(\mathbf{x})) + \overline{\kappa}^2(\mathbf{x})\phi(\mathbf{x}) = \sum_{k=1}^{N_a} q_k \delta(\mathbf{x} - \mathbf{y}_k), \quad \mathbf{x} \in \mathbb{R}^3, \tag{1}$$

where $\varepsilon(\mathbf{x})$ is the dielectric constant, $\overline{\kappa}$ is the modified Debye-Hückel inverse length in units of Å^{-1}, N_a is the number of atoms in the solute biomolecule, $\mathbf{y}_k$ is the position of the kth solute atom, and q_k is the associated partial charge in units of fundamental charge e_c. The dielectric interface conditions are

$$\phi_1(\mathbf{x}) = \phi_2(\mathbf{x}), \quad \varepsilon_1 \frac{\partial \phi_1(\mathbf{x})}{\partial n} = \varepsilon_2 \frac{\partial \phi_2(\mathbf{x})}{\partial n}, \quad \mathbf{x} \in \Gamma, \tag{2}$$

where $\phi_1(\mathbf{x})$ and $\phi_2(\mathbf{x})$ are the limiting values approaching the interface Γ from inside and outside the biomolecule, respectively, and n indicates the outward normal direction on the interface. The first condition in Eq. (2) expresses continuity of the potential across the interface and the second condition expresses continuity of the electric flux. The far-field boundary condition is

$$\lim_{|\mathbf{x}| \to \infty} \phi(\mathbf{x}) = 0. \tag{3}$$

The present work assumes that ε and $\overline{\kappa}$ are piecewise constant,

$$\varepsilon(\mathbf{x}) = \begin{cases} \varepsilon_1, & \mathbf{x} \in \Omega_1, \\ \varepsilon_2, & \mathbf{x} \in \Omega_2, \end{cases} \quad , \quad \overline{\kappa}^2(\mathbf{x}) = \begin{cases} 0, & \mathbf{x} \in \Omega_1, \\ \left(\dfrac{8\pi N_A e_c^2}{1000 k_B T} \right) I_s, & \mathbf{x} \in \Omega_2, \end{cases} \tag{4}$$

where N_A is Avogadro's number, k_B is the Boltzmann constant, T is the temperature, and I_s is the molar concentration of the ionic solvent. A key quantity of interest is the electrostatic solvation energy,

$$\Delta G_{\text{solv}} = \frac{1}{2} \sum_{k=1}^{N_a} q_k \phi_{\text{reac}}(\mathbf{y}_k), \tag{5}$$

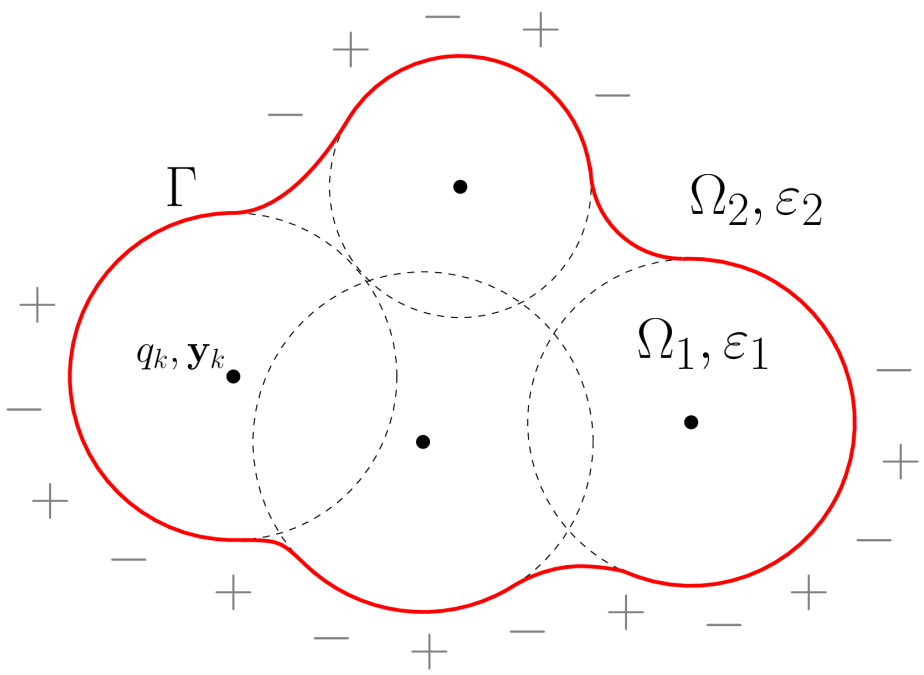


Figure 1: Poisson–Boltzmann implicit solvent model, solute domain Ω_1 with dielectric constant ε_1, atomic charges q_k located at $\mathbf{y_k}$, vdW radii (dashed circles), solvent domain Ω_2 with dielectric constant ε_2, dissolved salt ions $(+,-)$, dielectric interface Γ.

where the reaction potential at an atomic position is defined by the limit

$$\phi_{\text{reac}}(\mathbf{y}_k) = \lim_{\mathbf{x}\to\mathbf{y}_k} \left(\phi(\mathbf{x}) - \sum_{j=1}^{N_a} \frac{q_j}{4\pi|\mathbf{x}-\mathbf{y}_j|} \right), \tag{6}$$

representing the difference between the total potential and the Coulomb potential.

A variety of numerical methods have been applied to the PB model[6], including finite-difference[7–16], finite-element[17–19], and boundary element[20–28] methods. The present work focuses on boundary element methods (BEM) which compute the surface potential on a triangulation of the interface; these schemes benefit from rigorous enforcement of the interface conditions and far-field boundary condition, but they face the difficulty of evaluating singular integrals and the expense of solving a dense linear system. The treecode-accelerated boundary integral PB solver (TABI-PB) addresses these issues using a centroid collocation scheme to discretize the integrals and a treecode algorithm to reduce the cost of solving the linear system from $O(N^2)$ to $O(N \log N)$, where N is the number of triangles representing the interface[27].

MOLECULAR SURFACE MODELS

Several models have been utilized for the dielectric interface between the solute and solvent domains in implicit solvent simulations[29,30]. The simplest of these models, the **van der Waals surface** (vdW), is the union of hard spheres with vdW radii representing the atoms comprising the biomolecule. The **solvent accessible surface** (SAS) is formed by tracing the center of a probe sphere representing a water molecule rolling along the exterior of the vdW surface; the SAS surface is equivalent to a vdW surface in which the vdW radii are increased by the probe sphere radius. The **solvent excluded surface** (SES) is formed by the inward facing surface of the probe sphere rolling along the vdW surface[31,32]. The SES surface is comprised of spherical contact patches where the probe sphere touches the vdW surface, and toroidal reentrant patches formed by the inward facing surface of the probe sphere when it does not touch the vdW surface, i.e., when it is in contact with more than one solute atom. The **skin surface**[33–35] is comprised of spherical and hyperboloid patches constructed from a set of spheres through shrinking and convex combinations. The **Gaussian surface**[36] is the level set of a linear combination of Gaussian functions centered at the solute atoms.

A number of algorithms have been developed to triangulate these surfaces, where the input is the location and radii of the solute atoms and the output is a list of triangles. An alternative approach uses a level-set representation of the surface in an adaptive Cartesian grid[37,38]. Publicly available surface triangulation codes include MSMS[39], EDTSurf[40,41], TMSmesh[29,36], and NanoShaper[42]. Previous work investigated the performance of SES, skin, and Gaussian surfaces in the finite-difference DelPhi code[30], and the performance of Gaussian surfaces relative to SES surfaces in the boundary element fast multipole code AFMPB[43]. The present work focuses on the SES surface and compares the performance of the MSMS and NanoShaper triangulation codes in computations of the surface area and electrostatic solvation energy utilizing the boundary element TABI-PB solver. Next we describe the MSMS and NanoShaper codes, followed by the TABI-PB solver.

MSMS

MSMS, introduced by Sanner in 1995[39], has been widely utilized for generating SES surface triangulations. The algorithm first creates an analytical representation of the surface and then generates a triangulation of specified density by fitting predefined triangulated patches to the surface. The mesh resolution is controlled by the user-specified density parameter d, which sets the number of vertices per Å^2 of surface area in the triangulation.

NanoShaper

NanoShaper, introduced by Decherchi and Rocchia in 2013[42], implements the SES surface as well as several alternatives including the Gaussian and skin surfaces. In constructing an SES surface triangulation, NanoShaper first builds a description of the surface with a set of patches, analytically if possible or else with an approximation. The code then employs a ray-casting algorithm in which rays parallel to the coordinate axes are cast and intersections with the surface are calculated. The vertex positions of intersection are then used by the marching cubes algorithm to obtain the triangulation. The mesh resolution is controlled by the user-specified scale parameter s, which sets the number of grid points per Å in the marching cubes algorithm.

TABI-PB SOLVER

The TABI-PB solver[27] relies on a reformulation of the linear PB Eq. (1) developed by Juffer et al.[22] as a set of coupled 2nd kind boundary integral equations for the surface potential ϕ_1 and its normal derivative $\partial\phi_1/\partial n$ on the dielectric interface,

$$\frac{1}{2}\left(1+\varepsilon\right)\phi_1\left(\mathbf{x}\right) = \int_\Gamma \left[K_1\left(\mathbf{x},\mathbf{y}\right)\frac{\partial\phi_1\left(\mathbf{y}\right)}{\partial n} + K_2\left(\mathbf{x},\mathbf{y}\right)\phi_1\left(\mathbf{y}\right)\right] dS_\mathbf{y} + S_1(\mathbf{x}), \quad \mathbf{x}\in\Gamma, \quad (7a)$$

$$\frac{1}{2}\left(1+\frac{1}{\varepsilon}\right)\frac{\partial\phi_1\left(\mathbf{x}\right)}{\partial n} = \int_\Gamma \left[K_3\left(\mathbf{x},\mathbf{y}\right)\frac{\partial\phi_1\left(\mathbf{y}\right)}{\partial n} + K_4\left(\mathbf{x},\mathbf{y}\right)\phi_1\left(\mathbf{y}\right)\right] dS_\mathbf{y} + S_2(\mathbf{x}), \quad \mathbf{x}\in\Gamma, \quad (7b)$$

where $\varepsilon = \varepsilon_1/\varepsilon_2$ is the solute/solvent ratio of dielectric constants. The kernels K_1, K_2, K_3, K_4 depend on the Coulomb and screened Coulomb potentials,

$$G_0\left(\mathbf{x},\mathbf{y}\right) = \frac{1}{4\pi\left|\mathbf{x}-\mathbf{y}\right|}, \quad G_\kappa\left(\mathbf{x},\mathbf{y}\right) = \frac{e^{-\kappa|\mathbf{x}-\mathbf{y}|}}{4\pi\left|\mathbf{x}-\mathbf{y}\right|}, \quad (8)$$

where $\kappa^2 = \bar{\kappa}^2/\varepsilon_2$, and the source terms are defined by

$$S_1(\mathbf{x}) = \frac{1}{\varepsilon_1}\sum_{k=1}^{N_a} q_k G_0\left(\mathbf{x}, \mathbf{y}_k\right), \quad S_2(\mathbf{x}) = \frac{1}{\varepsilon_1}\sum_{k=1}^{N_a} q_k \frac{\partial G_0\left(\mathbf{x}, \mathbf{y}_k\right)}{\partial n_{\mathbf{x}}}. \tag{9}$$

In this context the electrostatic solvation energy in Eqs. (5)-(6) is obtained from an equivalent expression involving the surface potential and its normal derivative,

$$\Delta G_{\text{solv}} = \frac{1}{2}\sum_{k=1}^{N_a} q_k \int_\Gamma \left[K_1(\mathbf{y}_k, \mathbf{y})\frac{\partial \phi_1\left(\mathbf{y}\right)}{\partial n} + K_2(\mathbf{y}_k, \mathbf{y})\phi_1(\mathbf{y})\right] dS_{\mathbf{y}}. \tag{10}$$

The TABI-PB solver calculates the surface integrals using a boundary element method on the triangulated SES surface, where the collocation points are the triangle centroids. This yields a linear system for the surface potentials and normal derivatives which is solved by GMRES iteration utilizing a treecode to accelerate the matrix-vector product in each step of the iteration[27]. The boundary element form of the electrostatic solvation energy in Eq. (10) is employed, rather than the limit definition in Eqs. (5)-(6).

METHODOLOGY

To assess the SES surface triangulations produced by MSMS and NanoShaper, we compute the surface area S_{a} and electrostatic solvation energy ΔG_{solv} for 38 biomolecules given in Table 1 comprising peptides, proteins and nucleic acid fragments chosen from a previous comparison study of surface triangulation codes[43]. The PQR file for each biomolecule was generated using PDB2PQR[44] with the CHARMM force field and water molecules removed. For all surfaces a probe radius of 1.4 Å was used. The physical parameter values were ionic concentration $I_s = 0.15$ M, temperature $T = 300$ K, and solute and solvent dielectric constants $\varepsilon_1 = 1$, $\varepsilon_2 = 80$. The treecode parameters were multipole acceptance criterion $\theta = 0.8$, Taylor series order $p = 3$, and maximum number of particles in a leaf $N_0 = 500$. The GMRES tolerance was 1E–4, with 10 iterations between restarts and maximum number of iterations set to 110.

Triangulations were generated for the 38 biomolecules in test set using MSMS density $d = 1, 2, 4, 8, 16$ and NanoShaper scale $s = 1, 2, 3, 4, 5$. As observed in previous work[37], MSMS failed to produce a triangulation in 13 cases involving larger biomolecules and it

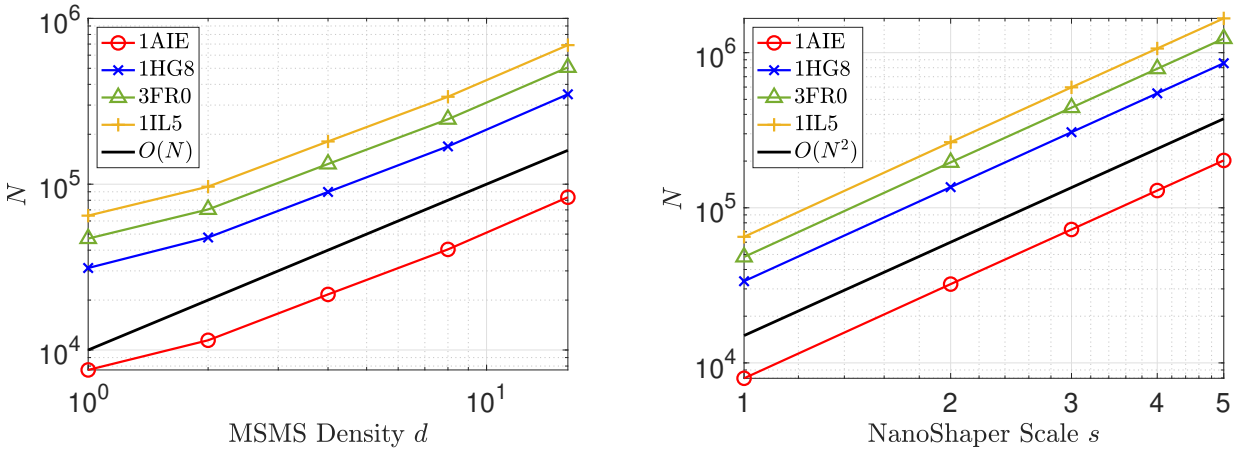
Table 1: Test set of 38 biomolecules[43] with PDB ID and number of atoms N_a.

PDB ID	2LWC	1GNA	1S4J	1CB3	1V4Z	1BTQ	1I2X	1AIE
N_a	75	163	182	183	266	304	513	522
PDB ID	1ZWF	375D	440D	4HLI	3ES0	3IM3	2IJI	1COA
N_a	586	593	629	697	781	851	890	1057
PDB ID	2AVP	1SM5	2ONT	4GSG	3ICB	1DCW	3LDE	1AYI
N_a	1085	1137	1161	1195	1202	1257	1294	1365
PDB ID	2YX5	3DFG	3LOD	1TR4	1RMP	1IF4	4DUT	3SQE
N_a	1385	2198	2246	3423	3478	4071	4217	4647
PDB ID	1HG8	4DPF	3FR0	2H8H	2CEK	1IL5		
N_a	4960	5824	6952	7084	8346	8349		

produced distorted surfaces with spurious solvation energy in 3 more cases. By contrast, NanoShaper failed in only one case, a low resolution mesh with scale $s = 1$ for the smallest molecule in the test set (2LWC, 75 atoms). Figure 2 plots the number of triangles N for four representative proteins (1AIE, 1HG8, 3FR0, 1IL5), showing that N depends linearly on the density d and quadratically on the scale s.

In the results reported below, the SES surface area $S_{\rm a}$ is computed by summing the triangle areas, and the solvation energy $\Delta G_{\rm solv}$ is computed using TABI-PB. In addition to examining the accuracy of the computed $S_{\rm a}$ and $\Delta G_{\rm solv}$, we report the total run time for TABI-PB computations which includes the run time for generating the triangulation and other pre-processing steps. It should be noted that the GMRES iteration run time in TABI-PB is much larger than the other run time components.

The computations were performed in serial on the University of Michigan FLUX cluster, with Intel Xeon CPUs running at either 2.5 or 2.8 GHz. In this system the exact processors could not be specified, so the timing results were averaged over multiple runs. The code was compiled with gfortran using the -O2 optimization flag. Surface visualizations were generated with VTK ParaView[45,46]. The version of TABI-PB used in this work is available at `github.com/lwwilson/TABI-PB`.



(a) (b)

Figure 2: Number of triangles N for different mesh resolution parameters, (a) MSMS density $d = 1, 2, 4, 8, 16$, (b) NanoShaper scale $s = 1, 2, 3, 4, 5$, proteins 1AIE, 1HG8, 3FR0, 1IL5.

RESULTS

First we explain how the triangulations are filtered, followed by discussions of the triangle aspect ratios, surface mesh features, effect of mesh resolution on computed results, and finally the computational efficiency.

Triangulation Filter

Molecular surface triangulation codes typically produce some small or thin triangles that reduce computational accuracy and efficiency. Hence following common practice, the present work deletes triangles if their area is less than 1E–5 Å^2, or if the distance between the centroids of two neighboring triangles is less than 1E–5 Å. Table 2 gives the percent of deleted triangles averaged over all triangulations generated using either MSMS or NanoShaper; among the deleted triangles, some had area less than machine precision and these are designated as zero-area triangles. For MSMS the deleted triangles are 0.064 % of the total, while for NanoShaper the fraction of deleted triangles is more than 100 times smaller. In addition, most of the deleted MSMS triangles were zero-area, while NanoShaper produced none of this type. In the results presented below, the triangulations have been filtered in this manner.

Table 2: Triangulation filter results showing percent of deleted triangles and zero-area triangles averaged over all triangulations generated using either MSMS or NanoShaper.

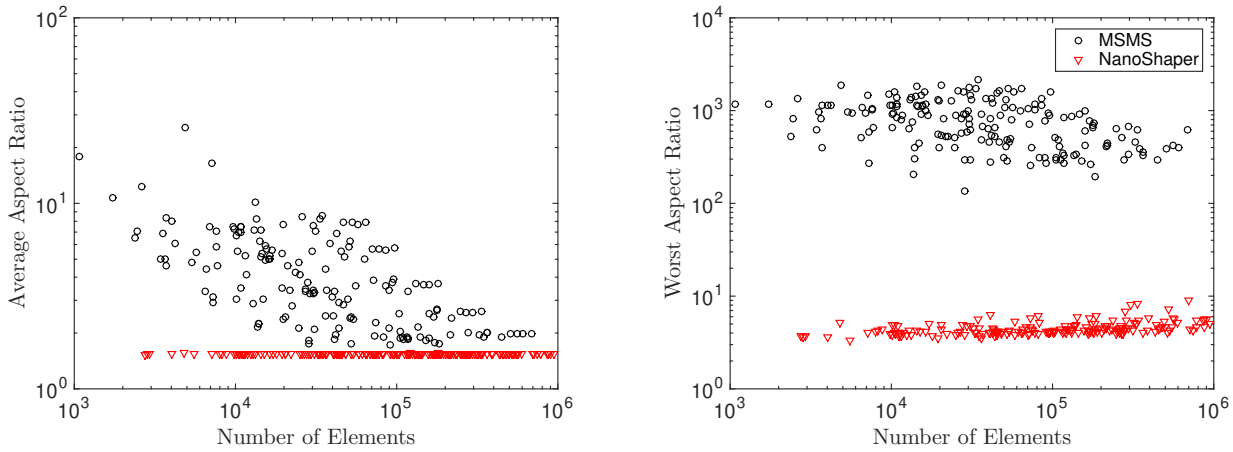
	deleted triangles	zero-area triangles
MSMS	0.064 %	0.054 %
NanoShaper	0.00052 %	0 %

Triangle Aspect Ratios

The aspect ratio of a triangle is defined as the ratio of the longest to shortest side lengths, where an equilateral triangle has ideal aspect ratio 1. Figure 3 plots the average and maximum aspect ratio for each triangulation generated versus the number of triangles N, where N varies from approximately 1E+3 to 1E+6 for the chosen density and scale parameters. Figure 3a shows that the average aspect ratio of MSMS triangles can be as large as 30 for small N and decreases to approximately 2 for large N, while the average aspect ratio of NanoShaper triangles is closer to 1 for all N. Figure 3b shows that the maximum aspect ratio of MSMS triangles varies between approximately 100 and 2000, while the maximum aspect ratio of NanoShaper triangles is less than 10. It should be noted that these large aspect ratio triangles are present even after the filtering described above. In the case of the finite-element method[47], mathematical analysis of model problems shows that computational accuracy and efficiency improves for triangulations with aspect ratio closer to 1. Although we are not aware of similar rigorous results for the boundary element method applied to the PB implicit solvent model, we expect that the discrepancy between MSMS and NanoShaper triangle aspect ratios gives NanoShaper triangulations a computational advantage.

Surface Mesh Features

Figure 4 displays the SES triangulation and surface potential for a representative protein (1AIE) using MSMS and NanoShaper with similar resolution ($N \approx$ 3E+4 in each case). The surfaces appear similar at first glance, although the NanoShaper surface is slightly smoother than the MSMS surface. Figure 5 displays a zoom of the triangulations, where



(a) average aspect ratio

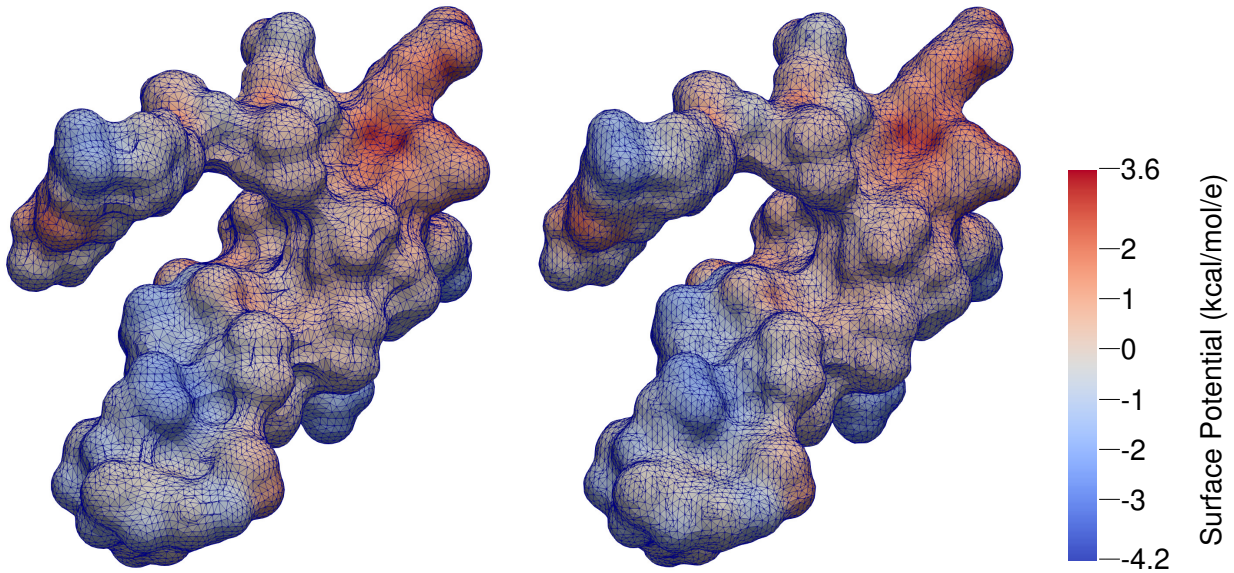
(b) maximum aspect ratio

Figure 3: Triangle aspect ratio for each triangulation generated versus number of elements N, (a) average, (b) maximum, MSMS ($\circ$, black), NanoShaper ($\triangledown$, red).

some small-scale irregular features are highlighted; in the MSMS mesh, green boxes enclose *stitches* formed by high aspect ratio triangles, and a yellow box encloses a *cusp* formed by neighboring triangles that meet at an acute angle, while in the NanoShaper mesh, a white box encloses a possible irregular feature, which could in fact simply be an artifact of the surface lighting. It should be noted that these irregular features are present in the MSMS mesh even after filtering, while the NanoShaper mesh is relatively free of them. Similar results were observed for other mesh resolutions and biomolecules in the test set. We expect that in TABI-PB computations, these irregular features diminish the efficiency of MSMS meshes relative to NanoShaper meshes.

Effect of Mesh Resolution

Next we examine the effect of mesh resolution for four representative proteins (1AIE, 1HG8, 3FR0, 1IL5). Figure 6 plots the computed surface area S_a versus N^{-1}, where N is the number of triangles in a given triangulation, and increasing resolution corresponds to the limit $N^{-1} \to 0$. The MSMS and NanoShaper results converge to almost the same value in Fig. 6a,d (1AIE, 1IL5), but in Fig. 6b,c (1HG8, 3FR0), the NanoShaper surface area is 2-3% larger than the MSMS surface area. In all cases the convergence with N^{-1} is smooth.



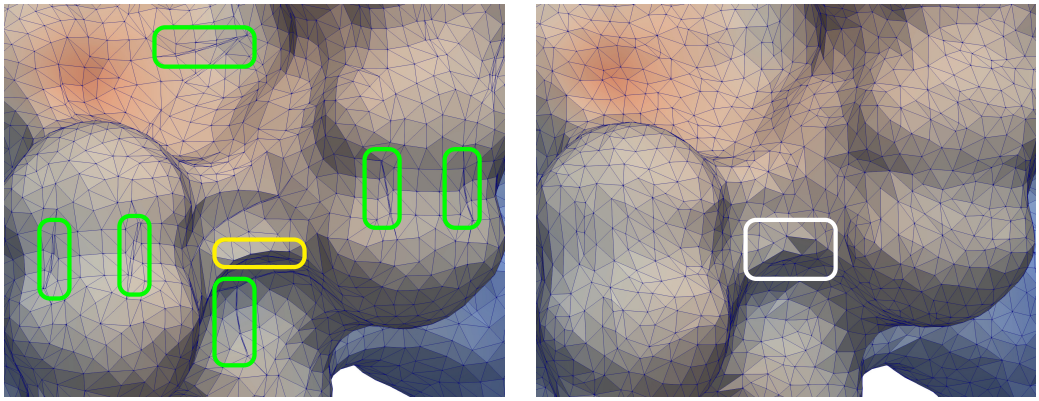
(a) MSMS (b) NanoShaper

Figure 4: Protein 1AIE, SES triangulation and surface potential, (a) MSMS density $d = 6$, $N = 31480$, (b) NanoShaper scale $s = 2$, $N = 32208$.

The MSMS results approach their limit somewhat faster, but MSMS was unable to generate reliable meshes with larger N; either it fails to produce a mesh or the generated mesh is distorted. The largest MSMS triangulation size obtained here was $N \approx$ 2e+6, whereas NanoShaper had no such limitation. Hence if it is necessary to generate a very dense mesh, or even a less dense mesh for a biomolecule with a large surface area, then NanoShaper has an advantage.

Figure 7 plots the computed solvation energy ΔG_{solv} versus N^{-1}. In this case the MSMS and NanoShaper results converge to almost the same value. The MSMS results again approach their limiting value somewhat faster than the NanoShaper results, but the NanoShaper dependence on N^{-1} is generally smoother than the MSMS dependence.

We used linear extrapolation to obtain highly accurate converged values of the surface area S_a and solvation energy ΔG_{solv}, i.e. the converged result is obtained by extrapolating the computed S_{a} and ΔG_{solv} values to the limit $N^{-1} \to 0$ using the two highest resolution meshes, MSMS density $d = 8, 16$ and NanoShaper scale $s = 4, 5$. In cases for which MSMS



(a) MSMS, zoomed (b) NanoShaper, zoomed

Figure 5: Protein 1AIE, zoom of SES triangulation, (a) MSMS, density $d = 6$, $N = 31480$, green boxes enclose *stitches* formed by high aspect ratio triangles, yellow box encloses a *cusp* formed by neighboring triangles that meet at an acute angle, (b) NanoShaper, scale $s = 2$, $N = 32208$, white box encloses a possible irregular feature.

produced spurious or null results, the extrapolation used the highest resolution meshes for which MSMS did not fail.

Figure 8 displays the extrapolated values of the surface area S_a and solvation energy ΔG_{solv} for the 38 biomolecules in the test set, where the NanoShaper results are plotted versus MSMS results. The correspondence between MSMS and NanoShaper results is very good, except in two cases (1I2X, 375D) indicated by the two markers furthest away from the diagonal line in Figs. 8a,b. These two cases each consist of a complex of two domains separated by more than the water probe radius, and MSMS returned a triangulation of only one domain. In principle, MSMS could be run on each domain, but this would require dividing the input XYZR file into two separate files and processing them individually. On the other hand, NanoShaper successfully triangulated both domains in the complex with no special processing required.

Computational Efficiency

Figure 9 compares the efficiency of MSMS and NanoShaper triangulations for computing the solvation energy ΔG_{solv} using TABI-PB. Figure 9a plots the run time versus the number

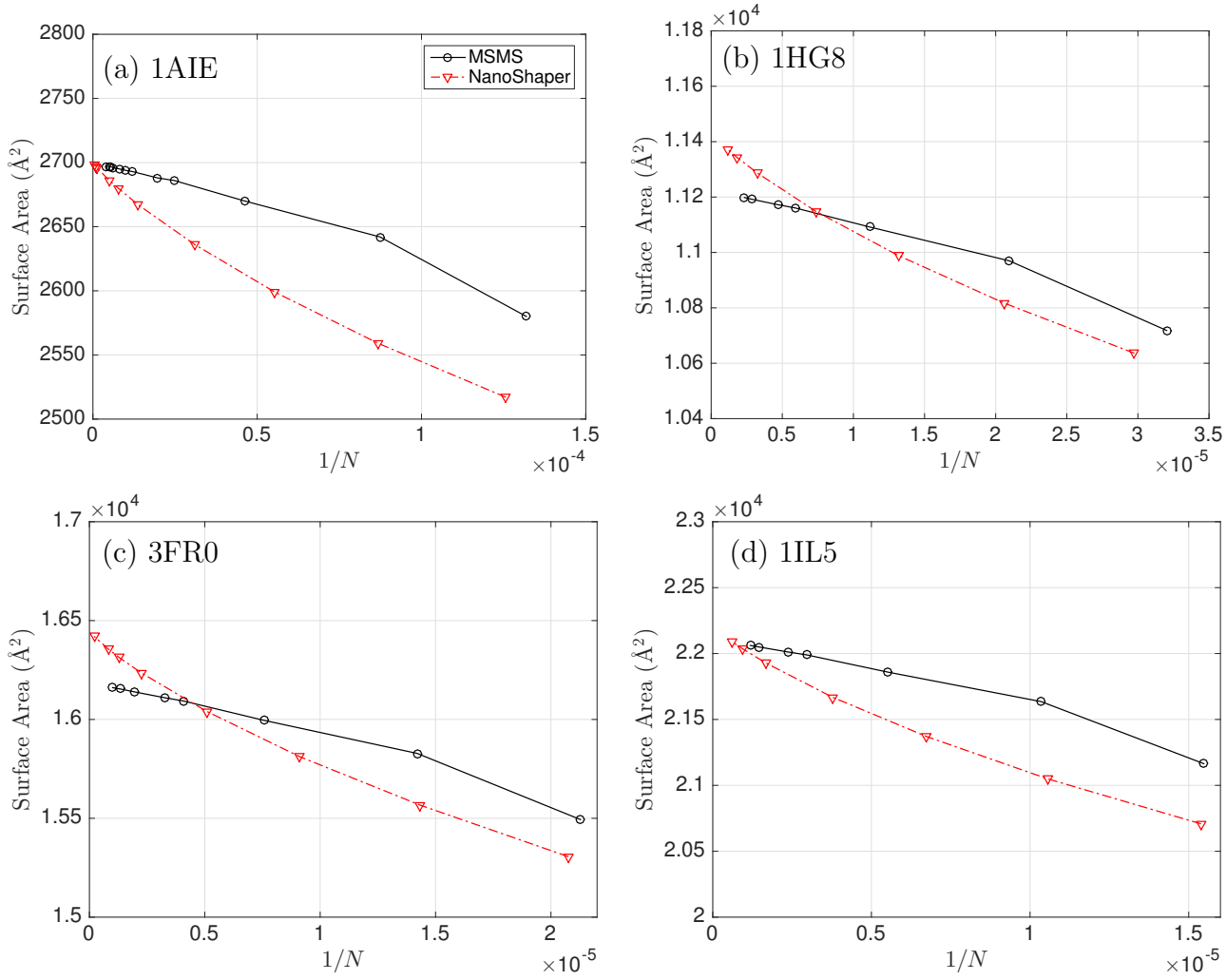


Figure 6: Computed surface area S_a versus N^{-1} for four proteins, N is the number of triangles, MSMS (black, solid line, $\circ$), NanoShaper (red, dashed line, $\triangledown$).

of triangles N for all 38 biomolecules and triangulations generated, where the solid lines are least squares fits to the data. The run time for generating and filtering the meshes is negligible compared to the TABI-PB run time. The results show that NanoShaper meshes generally require less run time than MSMS meshes. This is supported by Fig. 9b showing the number of GMRES iterations in each case, where the maximum number of iterations was set to 110. The results show that NanoShaper meshes generally require fewer GMRES iterations than MSMS meshes. Note that in the case of MSMS, the iteration maximum was reached in 23 out of 177 meshes, while in the case of NanoShaper, the iteration maximum was never reached.

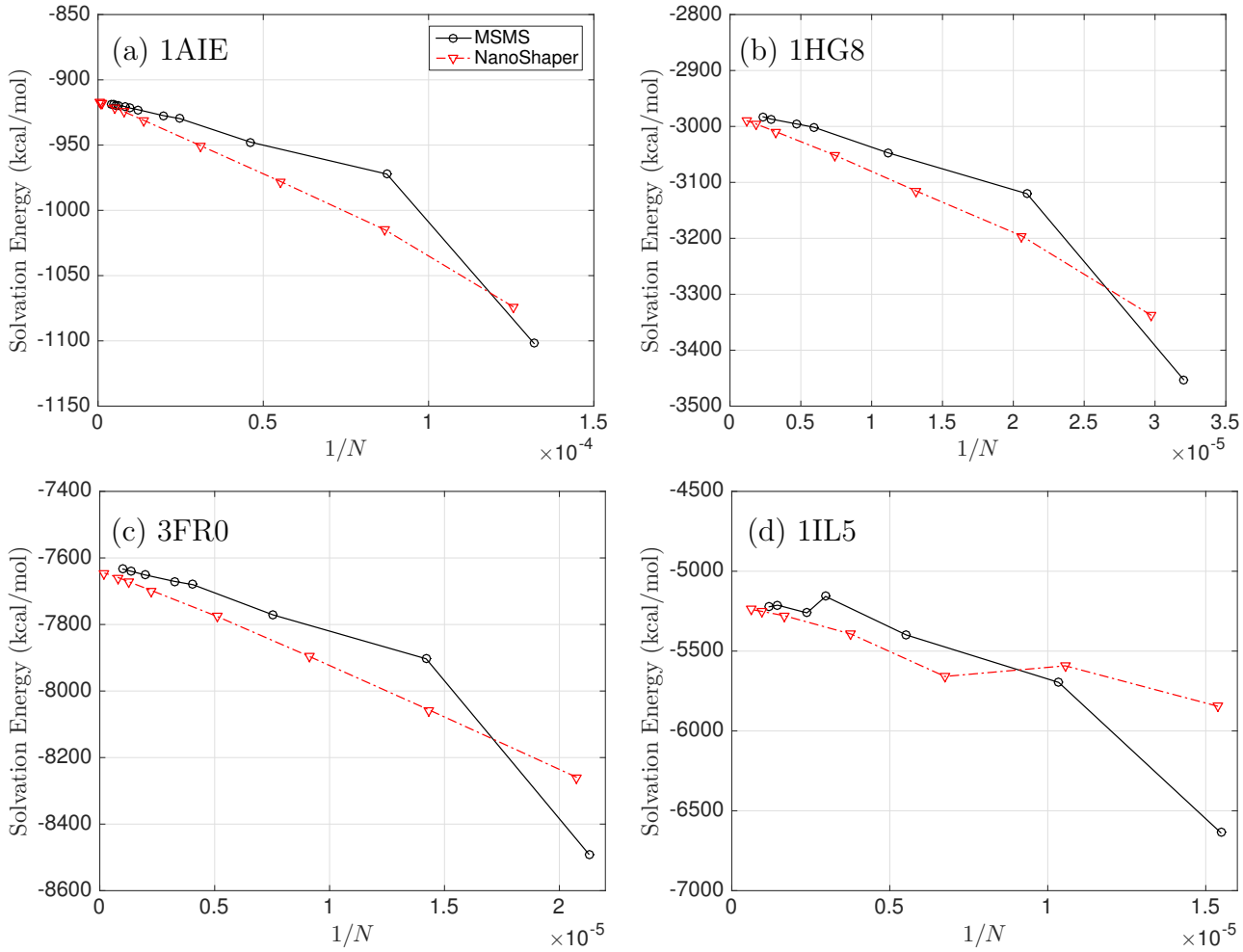
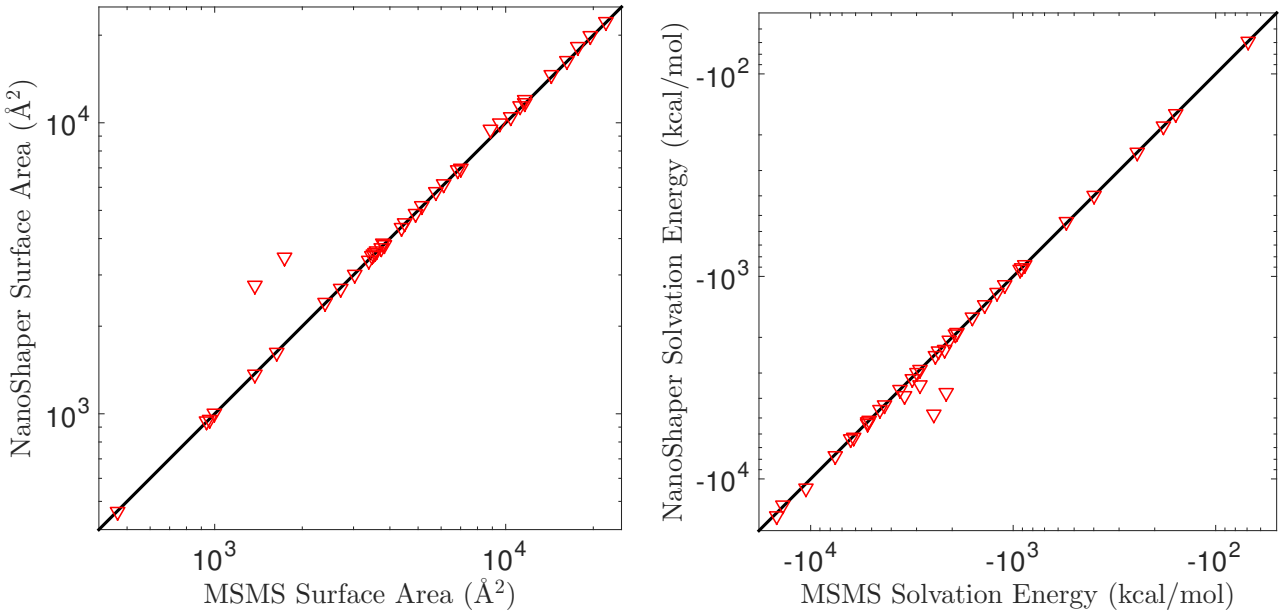


Figure 7: Computed solvation energy ΔG_{solv} versus N^{-1} for four proteins, N is the number of triangles, MSMS (black, solid line, $\circ$), NanoShaper (red, dashed line, $\triangledown$).

Table 3 quantifies the average run time and average number of GMRES iterations per triangle for each mesh type over the 38 biomolecules, showing that NanoShaper meshes require less run time and fewer iterations than MSMS meshes. The slower convergence of GMRES for MSMS meshes is attributed to the presence of large aspect ratio triangles (Fig. 3) and irregular small-scale features in the triangulation (Fig. 5). While the 2nd kind boundary integral formulation in Eq. (7) is well-conditioned[22], a low quality triangulation can result in an ill-conditioned discrete linear system. In such cases one can apply preconditioning to alleviate the problem and the present work used the simple diagonal preconditioner for this purpose. Recently a more effective block preconditioner was developed to further accelerate the convergence of GMRES in TABI-PB calculations[48], and it was shown that the block



(a) (b)

Figure 8: NanoShaper versus MSMS results for the 38 biomolecules in the test set using values extrapolated to the limit $N \to \infty$, (a) surface area S_{a}, (b) solvation energy ΔG_{solv}, black lines indicate perfect correspondence.

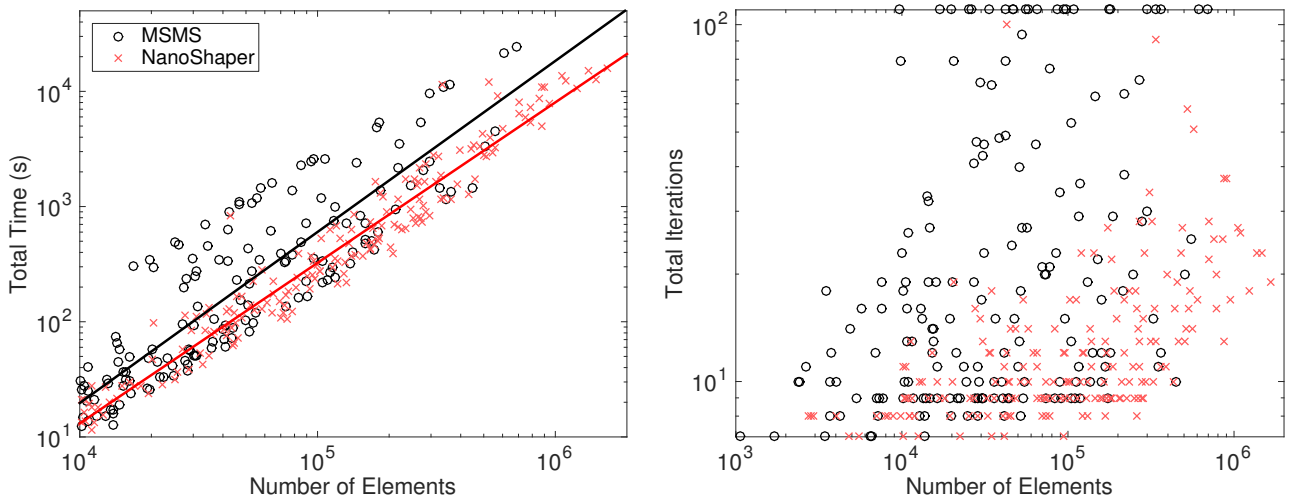
preconditioner reduces the number of GMRES iterations not only for MSMS meshes, but also in some cases for NanoShaper meshes.

Table 3: Average run time (s) and average number of GMRES iterations per triangle for MSMS and NanoShaper meshes over all 38 biomolecules and triangulations generated.

	average run time (s)/triangle	average iterations/triangle
MSMS	6.67E–3	1.17E–3
NanoShaper	4.19E–3	2.92E–4

CONCLUSIONS

We compared the performance of MSMS and NanoShaper, two widely used codes for triangulating the solvent excluded surface (SES) in Poisson–Boltzmann simulations of solvated



(a) (b)

Figure 9: Efficiency of triangulations generated using MSMS (∘, black) and NanoShaper (×, red) in computing solvation energy ΔG_{solv} using TABI-PB, (a) total run time (s), solid lines are least squares fits, (b) total number of GMRES iterations (maximum 110), versus number of triangles N for all 38 biomolecules and triangulations.

biomolecules. Comparisons were made of the surface area S_{a} and electrostatic solvation energy ΔG_{solv}, where the latter calculations were performed using the treecode-accelerated boundary integral (TABI-PB) solver utilizing a well-conditioned boundary integral formulation and centroid collocation on the SES triangulation. Calculations were carried out for a set of 38 biomolecules over a range of mesh resolutions. The triangulations produced by the two codes are qualitatively similar, although the MSMS meshes contain some triangles of exceedingly small area and high aspect ratio. The computed values of the surface area and solvation energy produced by MSMS and NanoShaper meshes often agree to within several percent, but NanoShaper meshes were more efficient, generally requiring less run time and fewer GMRES iterations than MSMS meshes. Furthermore, NanoShaper was consistently able to produce higher resolution meshes than MSMS, and NanoShaper solvation energies exhibited smoother convergence with increasing mesh resolution. A version of TABI-PB using NanoShaper was recently installed as an option in the APBS software suite maintained at the Pacific Northwest National Laboratory[49]. A future release of TABI-PB will incorporate the recently developed block preconditioner[48] and several other improvements.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are available from the corresponding author upon request.

ACKNOWLEDGMENTS

We thank the reviewer for constructive comments that helped improve the article. This work was supported by NSF grant DMS-1819094 and a grant from the Michigan Institute for Computational Discovery and Engineering (MICDE). Leighton Wilson was supported by the Department of Defense (DoD) through the National Defense Science & Engineering Graduate Fellowship (NDSEG) Program.

References

1. B. Roux and T. Simonson, Biophys. Chem. **78**, 1 (1999).

2. J. Tomasi, Theor. Chem. Acc. **112**, 184 (2004).

3. Z. Zhang, S. Witham, and E. Alexov, Phys. Biol. **8**, 035001 (2011).

4. N. A. Baker, in *Numerical Computer Methods, Part D*, edited by L. Brand and M. L. Johnson (Academic Press, 2004), vol. 383 of *Methods in Enzymology*, chap. 5, pp. 94–118, 1st ed.

5. J. Wang, C. Tan, Y.-H. Tan, Q. Lu, and R. Luo, Commun. Comput. Phys. **3**, 1010 (2008).

6. B. Lu, Y. C. Zhou, M. J. Holst, and J. A. McCammon, Commun. Comput. Phys. **3**, 973 (2008).

7. J. Warwicker and H. C. Watson, J. Mol. Biol. **157**, 671 (1982).

8. I. Klapper, R. Hagstrom, R. Fine, K. Sharp, and B. Honig, Proteins **1**, 47 (1986).

9. M. J. Holst and F. Saied, J. Comput. Chem. **16**, 337 (1995).

10. N. A. Baker, D. Sept, S. Joseph, M. J. Holst, and J. A. McCammon, Proc. Natl. Acad. Sci. U.S.A. **98**, 10037 (2001).

11. R. Luo, L. David, and M. K. Gilson, J. Comput. Chem. **23**, 1244 (2002).

12. J. Wang and R. Luo, J. Comput. Chem. **31**, 1689 (2010).

13. D. Chen, Z. Chen, C. Chen, W. Geng, and G. W. Wei, J. Comput. Chem. **32**, 756 (2011).

14. A. H. Boschitsch and M. O. Fenley, J. Chem. Theory Comput. **7**, 1524 (2011).

15. W. Geng and S. Zhao, Mol. Based Math. Biol. **1**, 109 (2012).

16. L. Wilson and S. Zhao, Int. J. Numer. Anal. Model. **13**, 852 (2016).

17. M. J. Holst, N. A. Baker, and F. Wang, J. Comput. Chem. **21**, 1319 (2000).

18. N. A. Baker, M. J. Holst, and F. Wang, J. Comput. Chem. **21**, 1343 (2000).

19. Y. Jiang, J. Ying, and D. Xie, Mol. Based Math. Biol. **2**, 2299 (2014).

20. R. J. Zauhar and R. S. Morgan, J. Comput. Chem. **9**, 171 (1988).

21. B. J. Yoon and A. M. Lenhoff, J. Comput. Chem. **11**, 1080 (1990).

22. A. H. Juffer, E. F. F. Botta, B. A. M. van Keulen, A. van der Ploeg, and H. J. C. Berendsen, J. Comput. Phys. **97**, 144 (1991).

23. H.-X. Zhou, Biophys. J. **65**, 955 (1993).

24. J. Liang and S. Subramaniam, Biophys. J. **73**, 1830 (1997).

25. A. H. Boschitsch, M. O. Fenley, and H.-X. Zhou, J. Phys. Chem. B **106**, 2741 (2002).

26. B. Lu, X. Cheng, J. Huang, and J. A. McCammon, Comput. Phys. Commun. **181**, 1150 (2010).

27. W. Geng and R. Krasny, J. Comput. Phys. **247**, 62 (2013).

28. C. D. Cooper, J. P. Bardhan, and L. A. Barba, Comput. Phys. Commun. **185**, 720 (2014).

29. M. Chen and B. Lu, J. Chem. Theory Comput. **7**, 203 (2011).

30. S. Decherchi, J. Colmenares, C. E. Catalano, M. Spagnuolo, E. Alexov, and W. Rocchia, Commun. Comput. Phys. **13**, 61 (2013).

31. F. M. Richards, Annu. Rev. Biophys. Bioeng. **6**, 151 (1977).

32. M. L. Connolly, J. Appl. Crystallogr. **18**, 499 (1985).

33. W. Rocchia, S. Sridharan, A. Nicholls, E. G. Alexov, A. Chiabrera, and B. Honig, J. Comput. Chem. **23**, 128 (2002).

34. W. Rocchia, Math. Comput. Model. **41**, 1109 (2005).

35. H.-L. Cheng and X. Shi, Comput. Geom. **42**, 192 (2009).

36. M. Chen, B. Tu, and B. Lu, J. Mol. Graph. Model. **38**, 411 (2012).

37. T. Can, C.-I. Chen, and Y.-F. Wang, J. Mol. Graph. Model. **25**, 442 (2006).

38. R. Egan and F. Gibou, J. Comput. Phys. **374**, 91 (2018).

39. M. F. Sanner, A. J. Olson, and J.-C. Spehner, in *Proceedings of the 11th ACM Symposium on Comput. Geom.* (Association for Computing Machinery, 1995), pp. C6–C7.

40. D. Xu and Y. Zhang, PLoS ONE **4**, e8140 (2009).

41. D. Xu, H. Li, and Y. Zhang, J. Comput. Biol. **20**, 805 (2013).

42. S. Decherchi and W. Rocchia, PLoS ONE **8**, e59744 (2013).

43. T. Liu, M. Chen, and B. Lu, J. Mol. Model. **21**, 113 (2015).

44. T. J. Dolinsky, J. E. Nielsen, J. A. McCammon, and N. A. Baker, Nucleic Acids Res. **32**, W665 (2004).

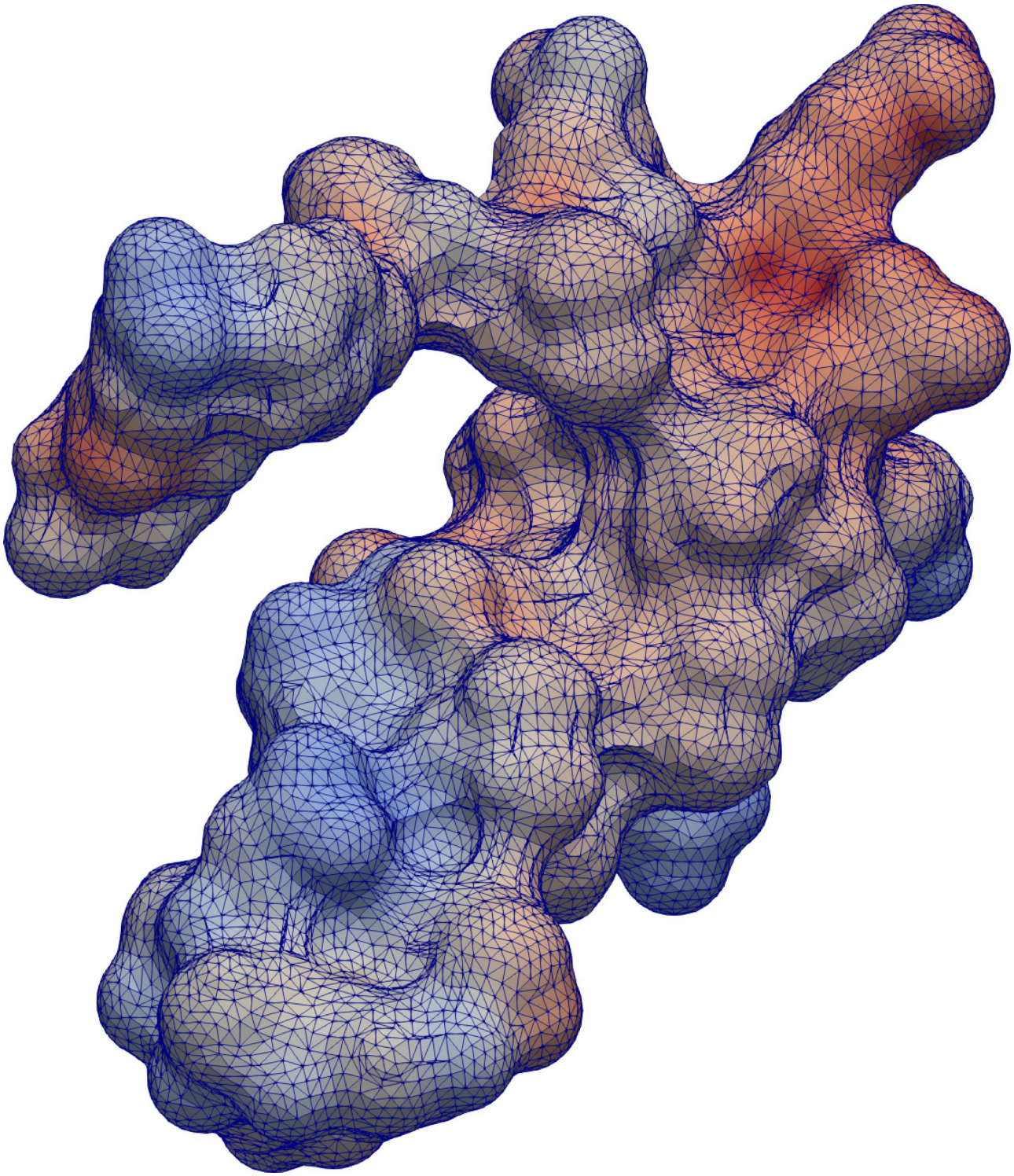
45. J. Ahrens, B. Geveci, and C. Law, in *The Visualization Handbook*, edited by C. D. Hansen and C. R. Johnson (Butterworth–Heinemann, Burlington, MA, USA, 2005), pp. 717–731.

46. U. Ayachit, *The ParaView Guide: A Parallel Visualization Application* (Kitware, Inc., Clifton Park, NY, USA, 2015).

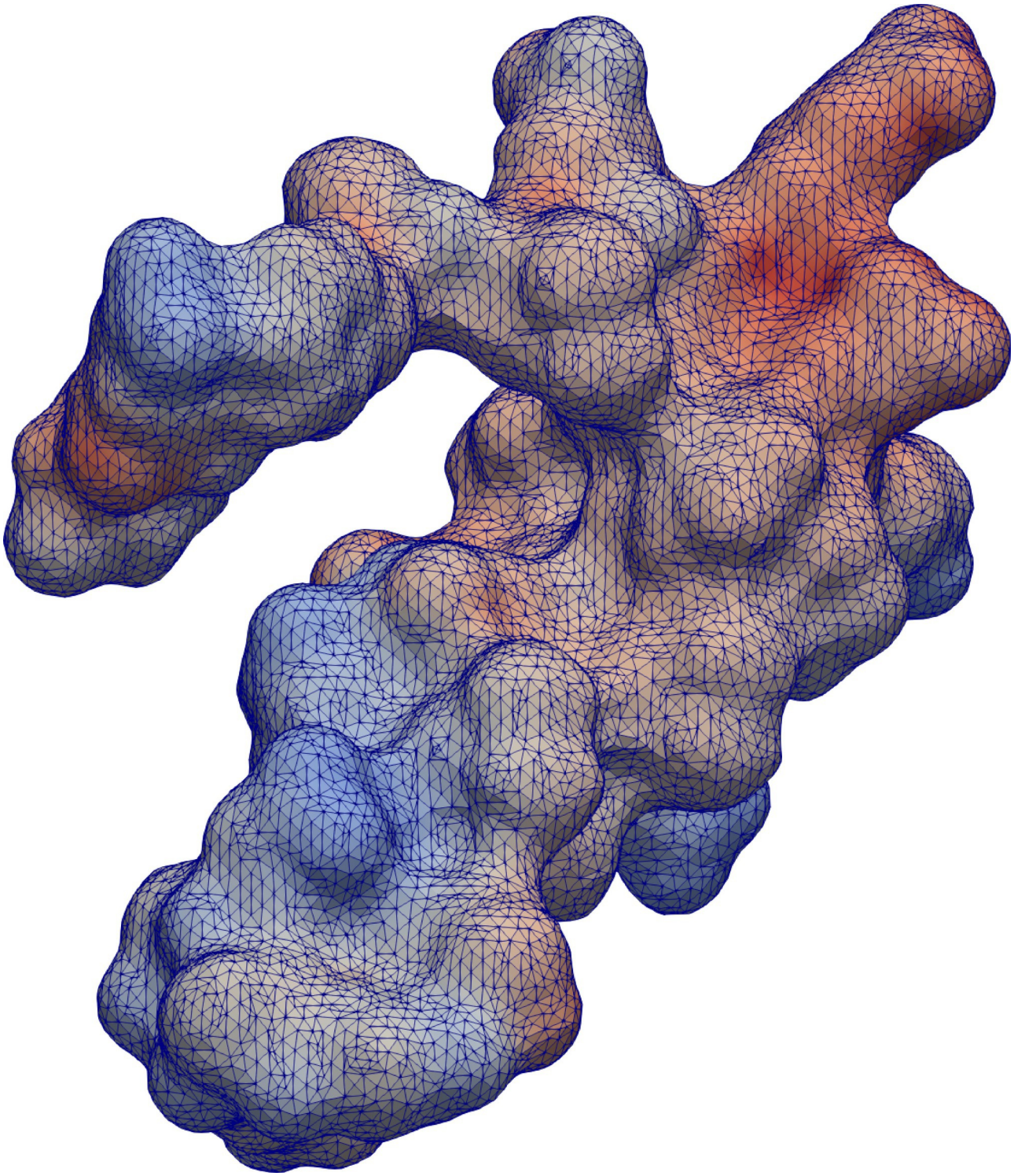
47. C. Johnson, *Numerical Solution of Partial Differential Equations by the Finite Element Method* (Studentlitteratur, 1987).

48. J. Chen and W. Geng, J. Comput. Phys. **373**, 750 (2018).

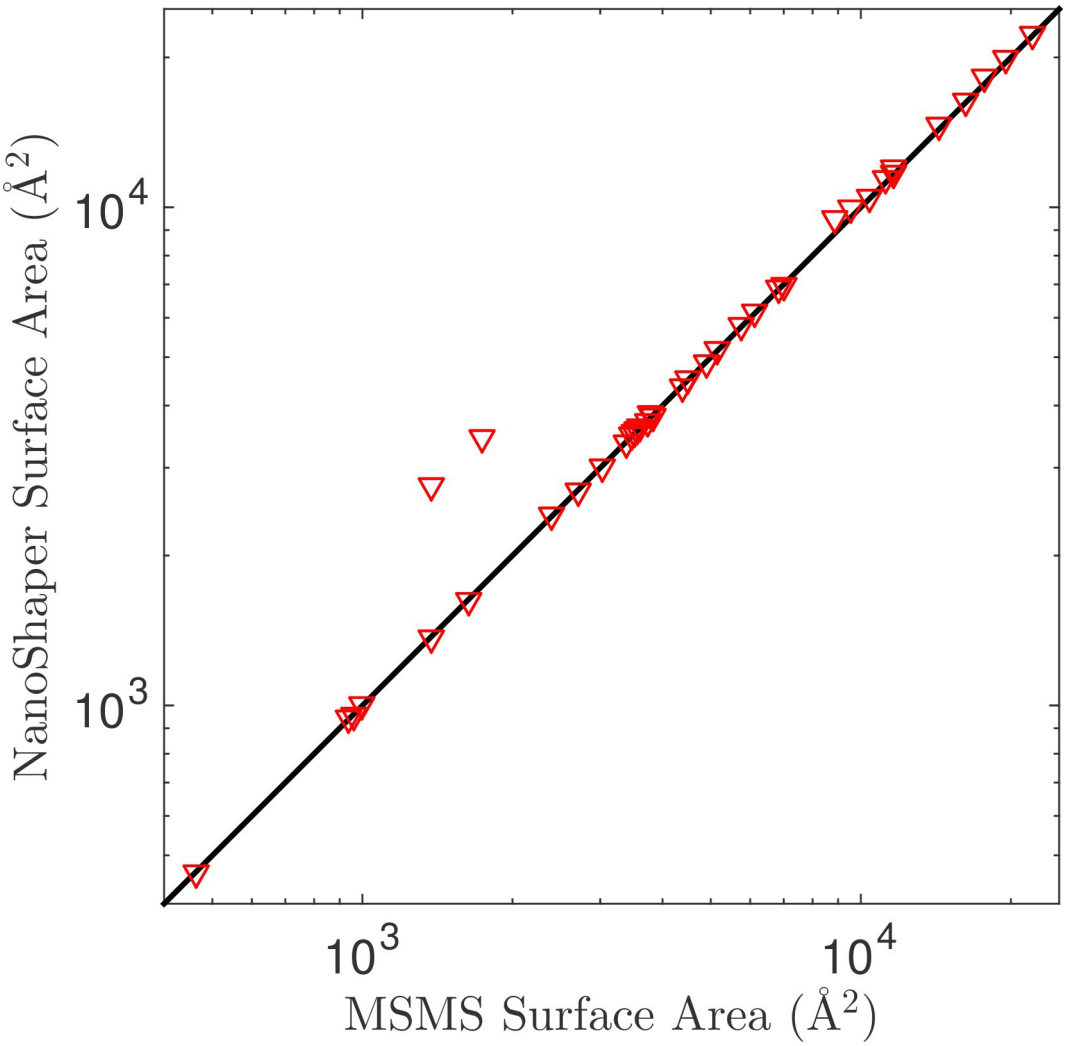
49. E. Jurrus, D. Engel, K. Star, K. Monson, J. Brandi, L. E. Felberg, D. H. Brookes, L. Wilson, J. Chen, K. Liles, et al., Protein Sci. **27**, 112 (2018).



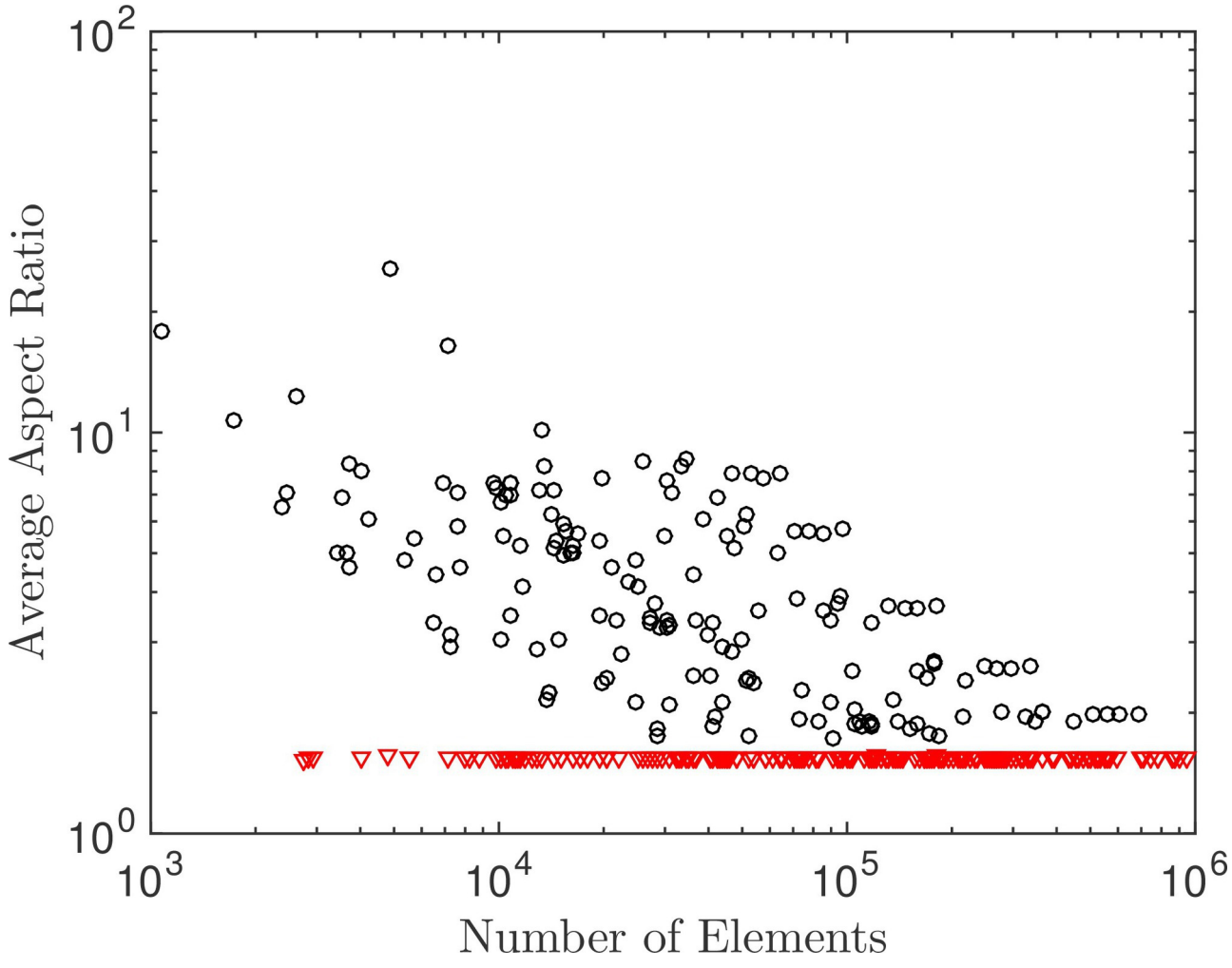
jcc_26692_1aie_msms_6.eps



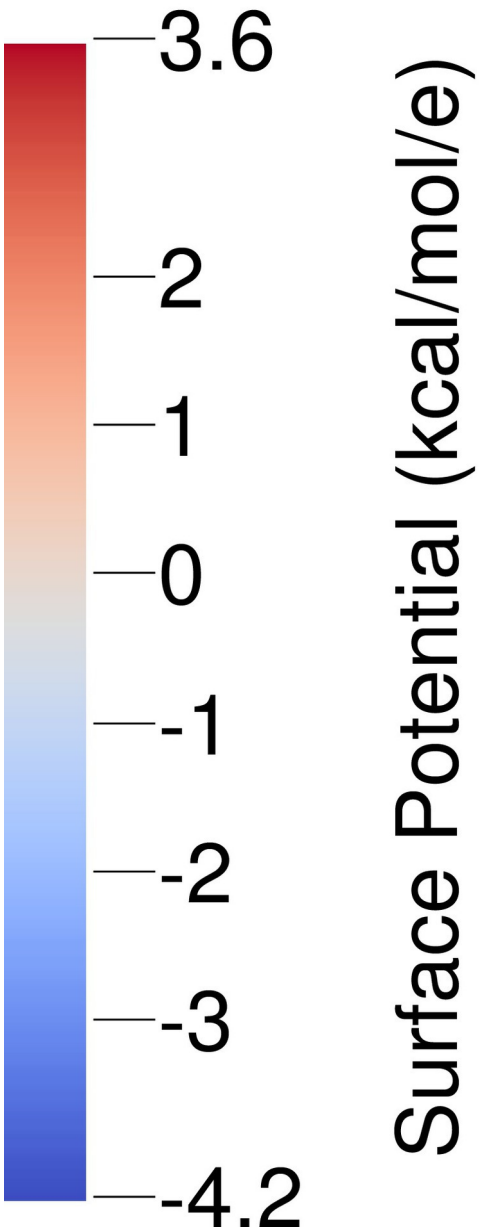
jcc_26692_1aie_nanoshaper_ses_2.eps



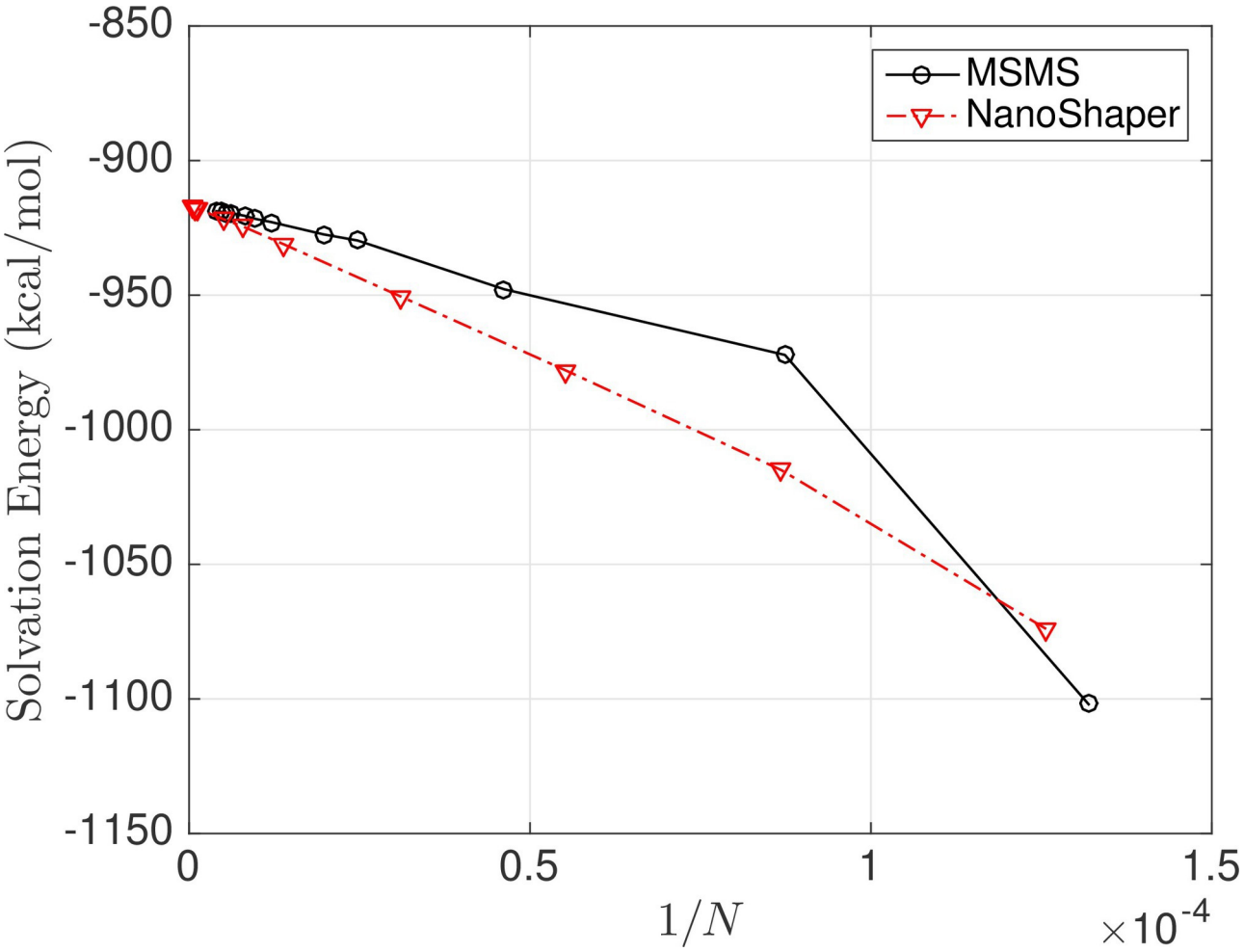
jcc_26692_areacomp2.eps



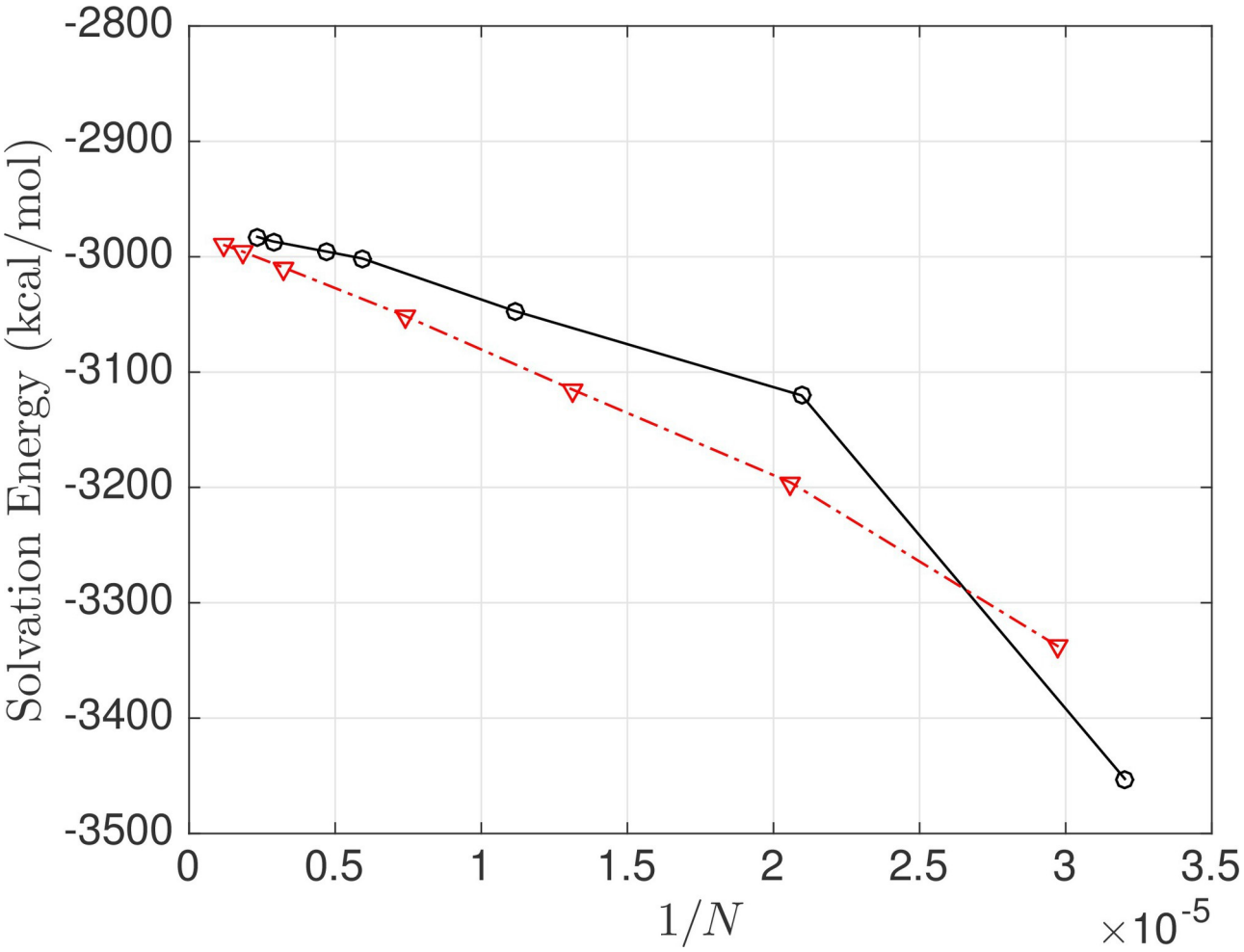
jcc_26692_average_aspect_ratio.eps



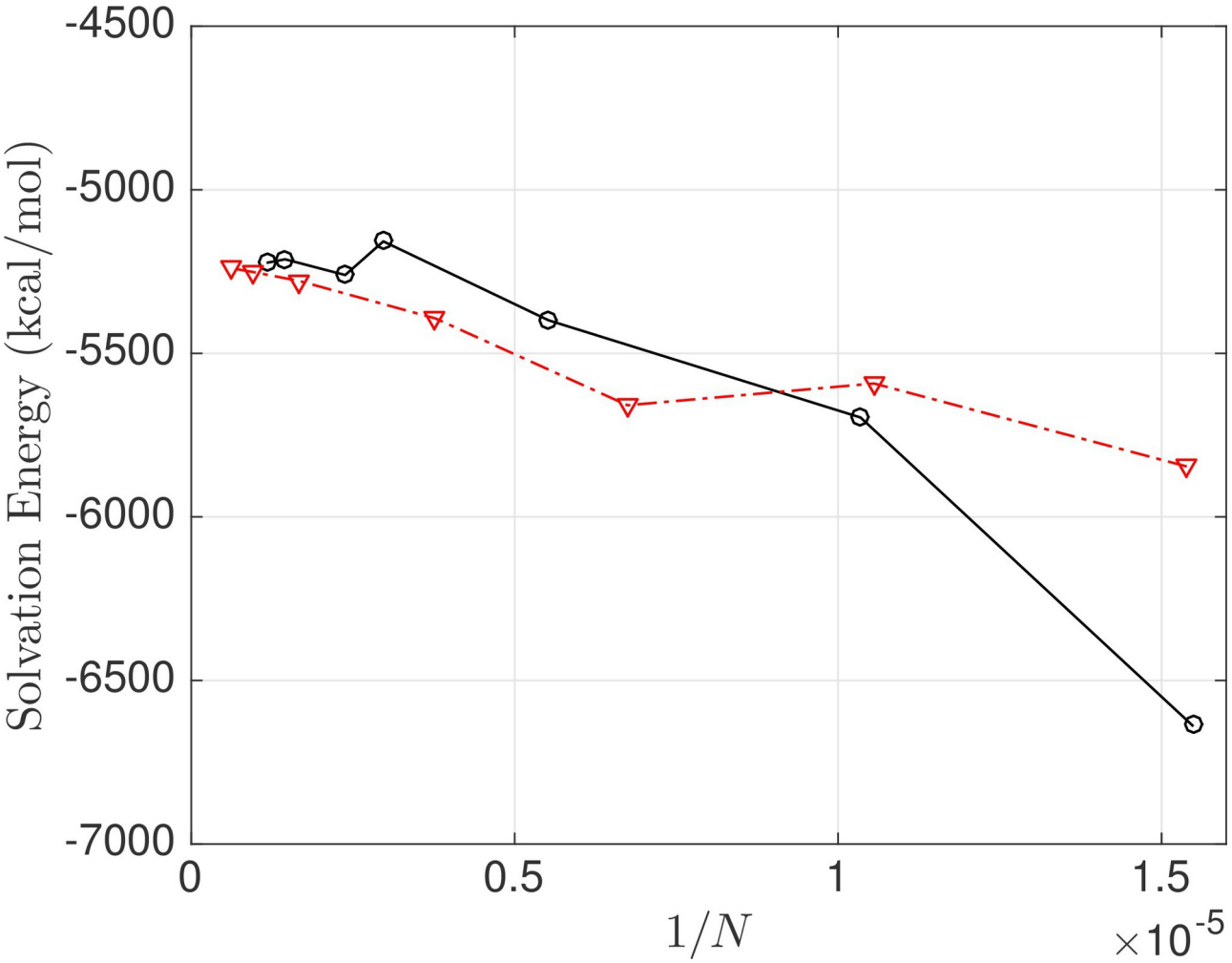
jcc_26692_colorlegend.eps



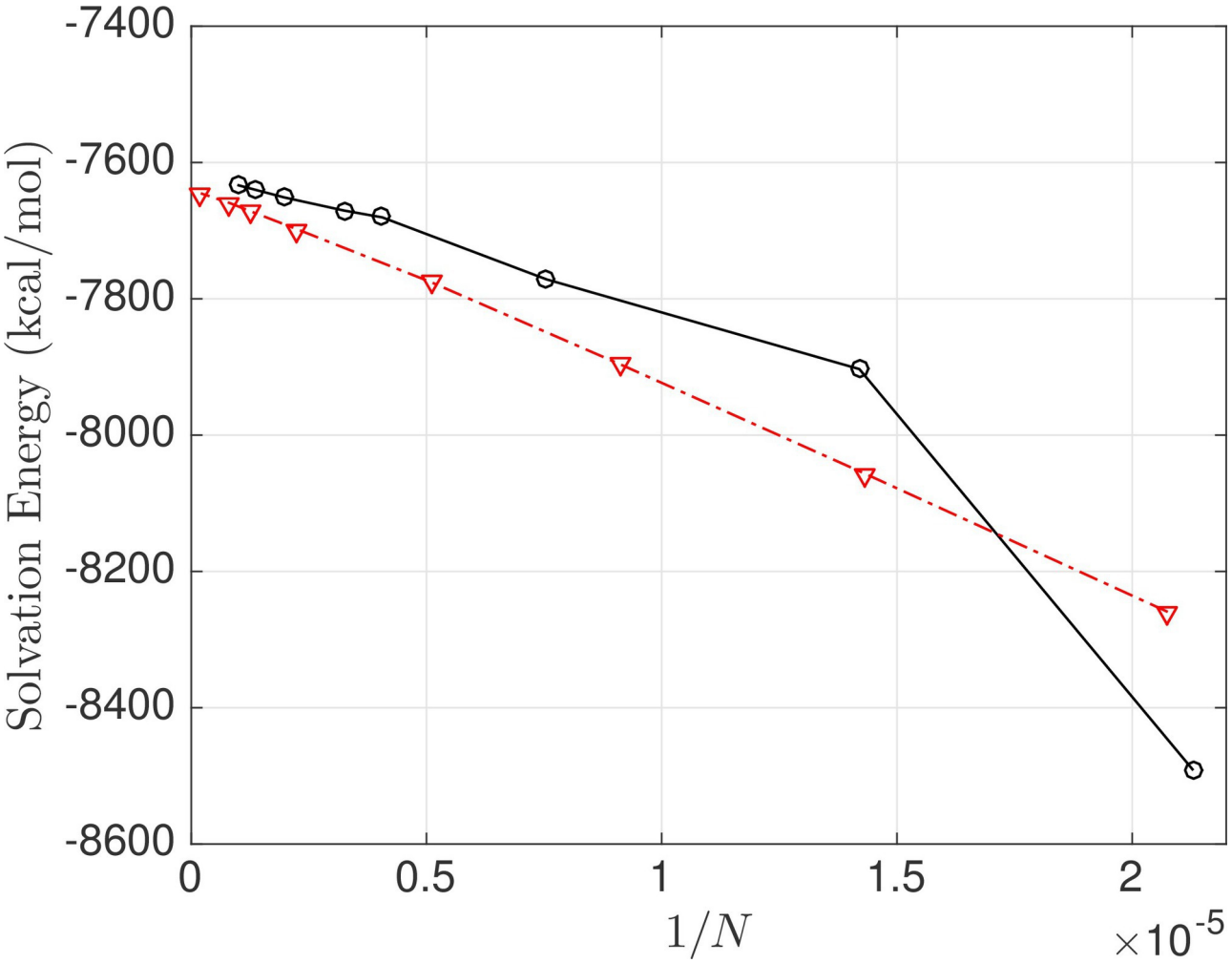
jcc_26692_converge_faces_1aie.eps



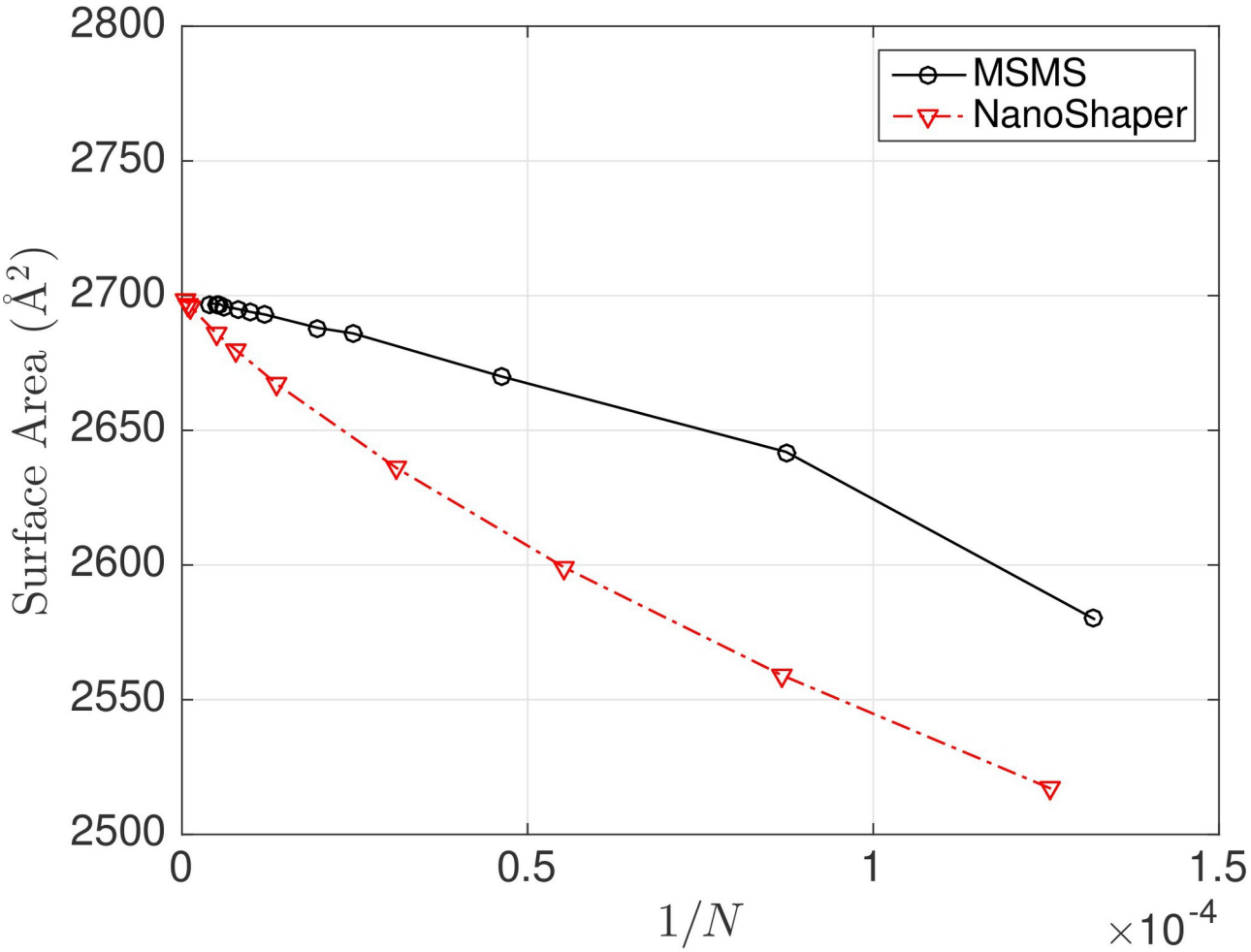
jcc_26692_converge_faces_1hg8.eps



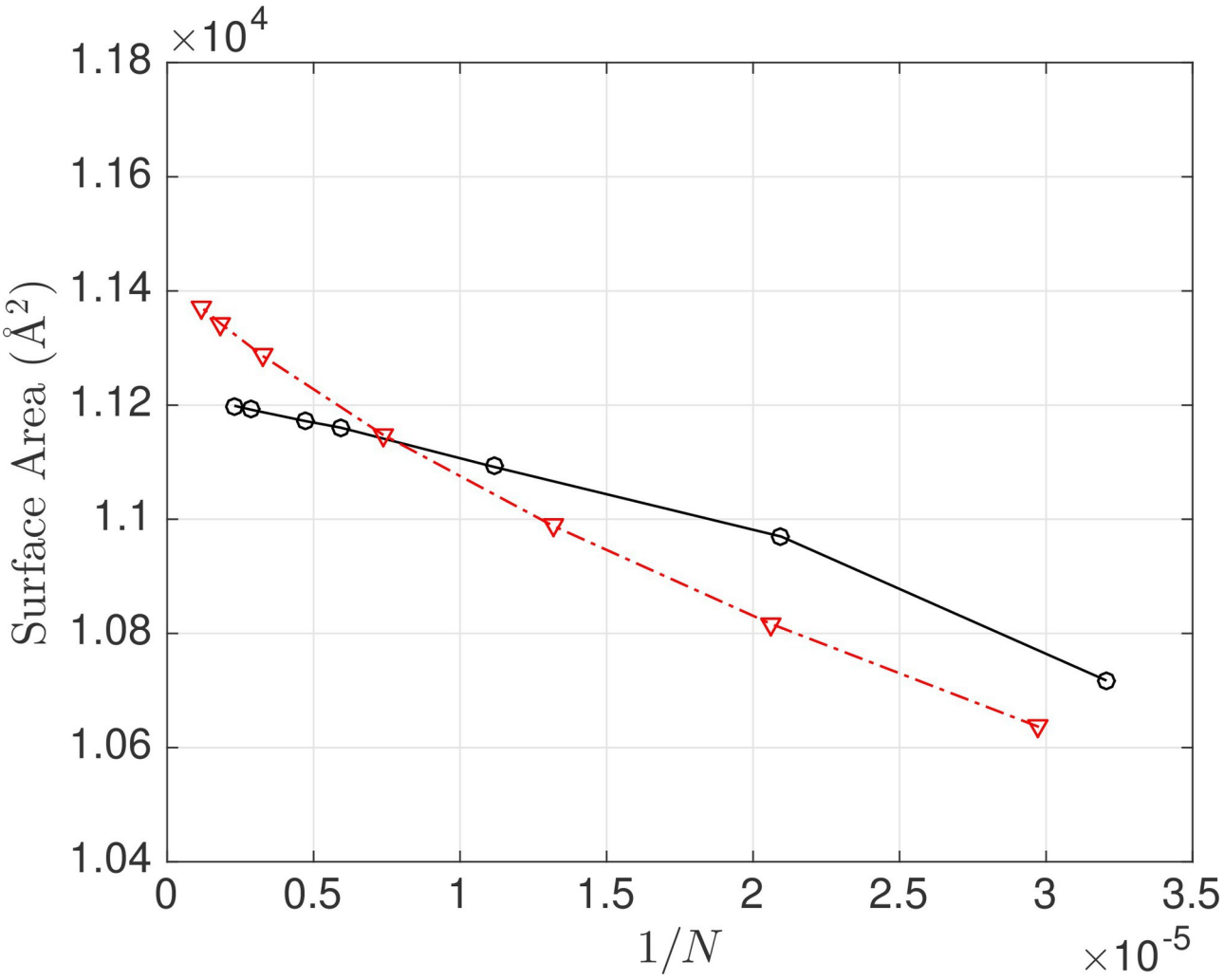
jcc_26692_converge_faces_1il5.eps



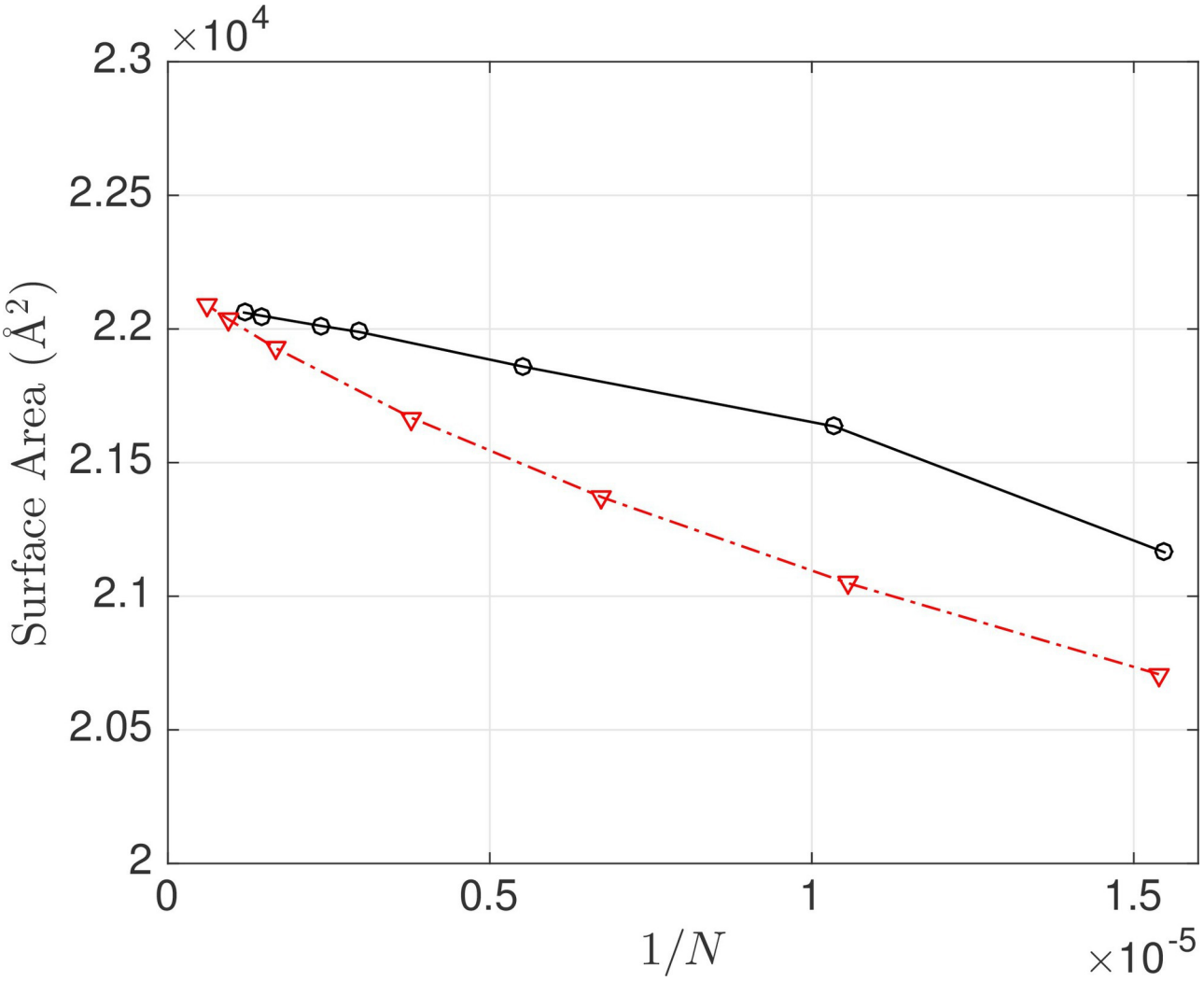
jcc_26692_converge_faces_3fr0.eps



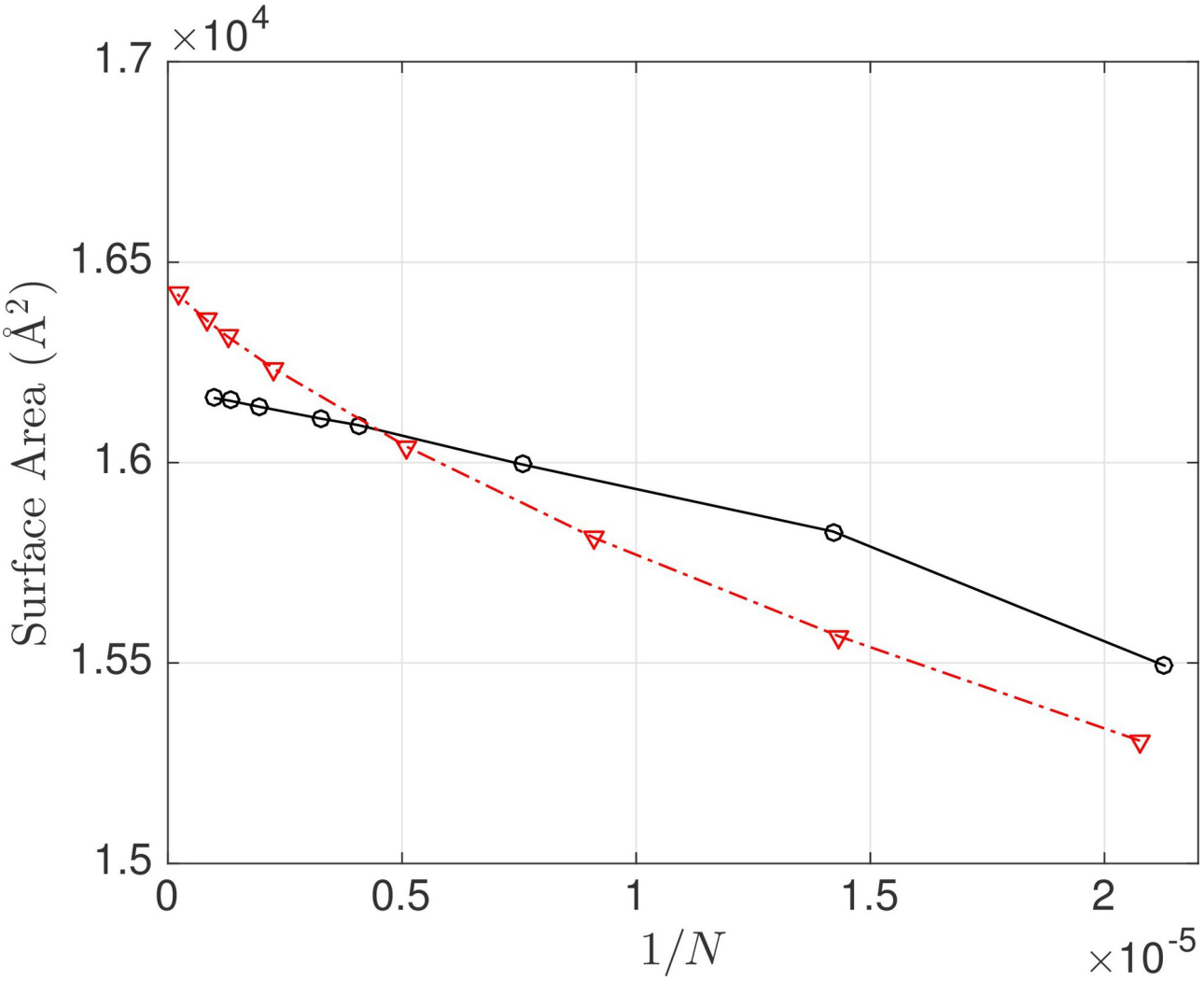
jcc_26692_converge_faces_area_1aie.eps



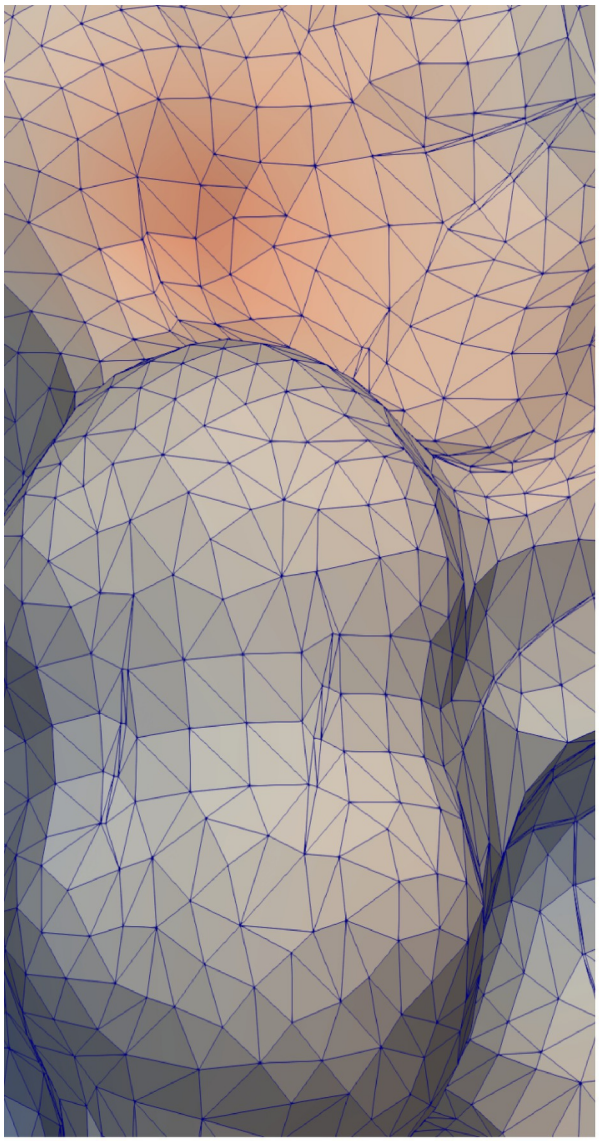
jcc_26692_converge_faces_area_1hg8.eps



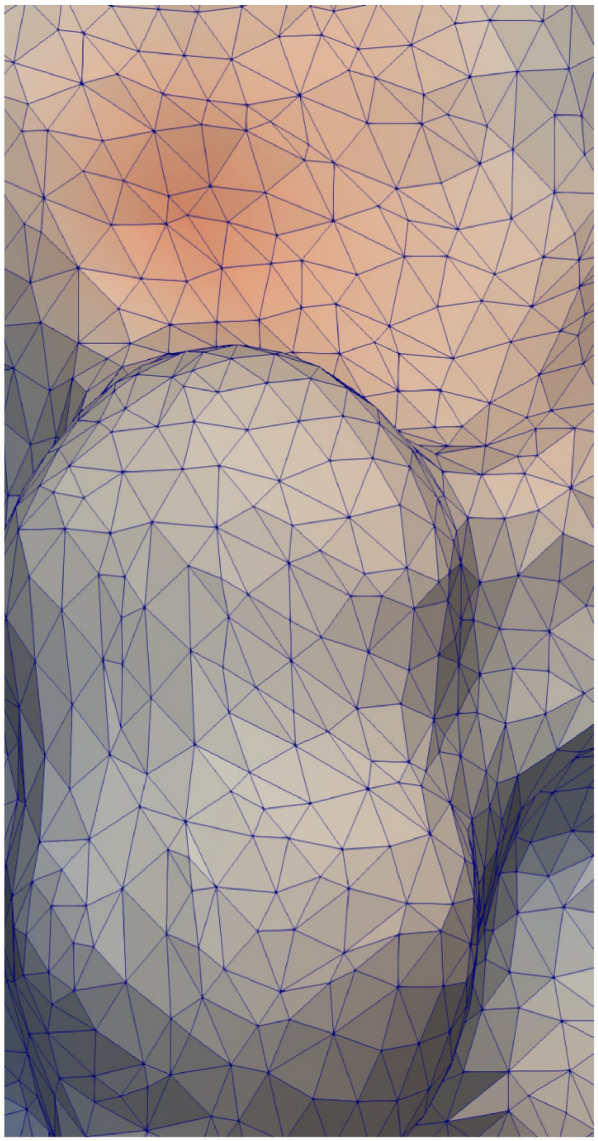
jcc_26692_converge_faces_area_1il5.eps



jcc_26692_converge_faces_area_3fr0.eps

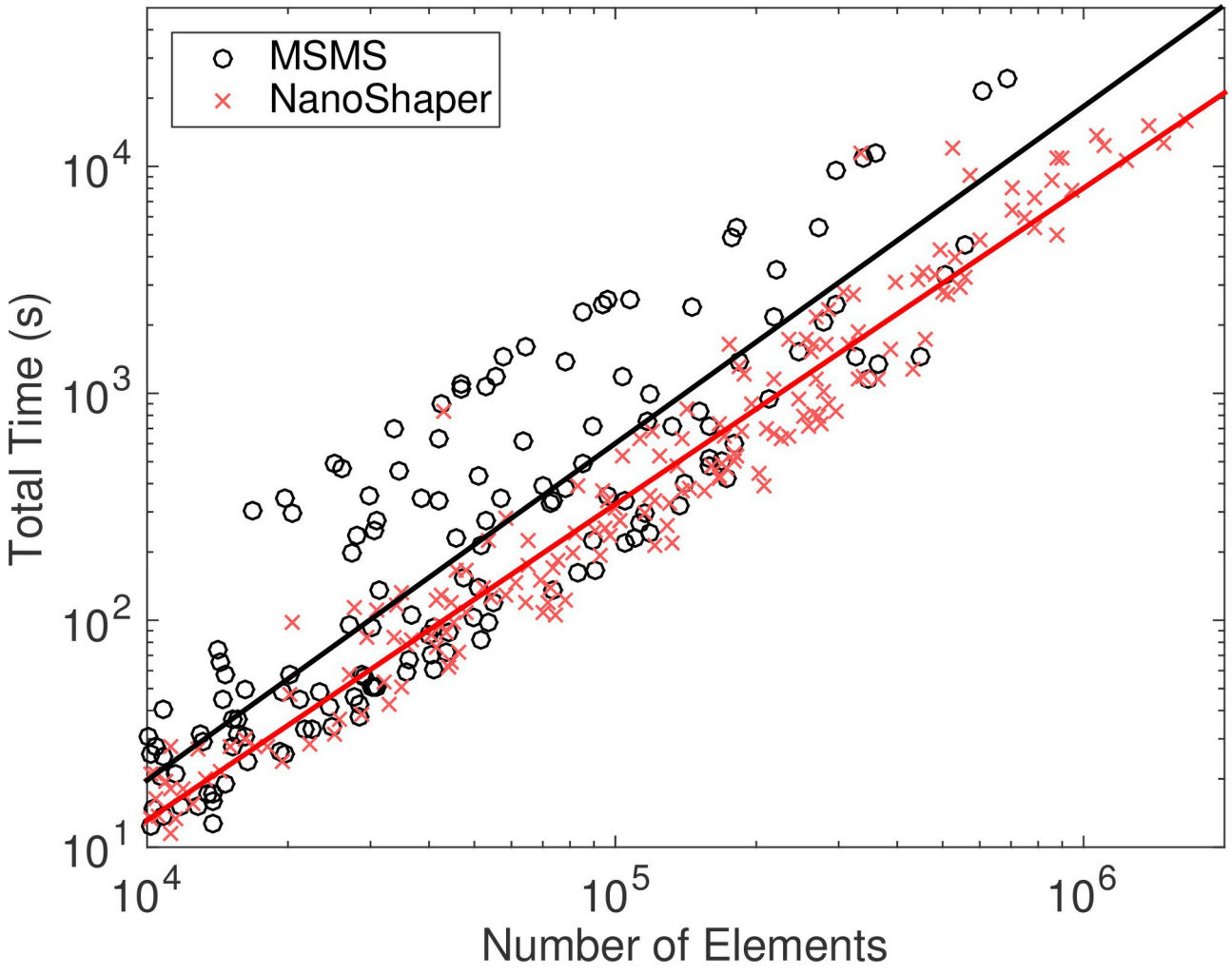


MSMS

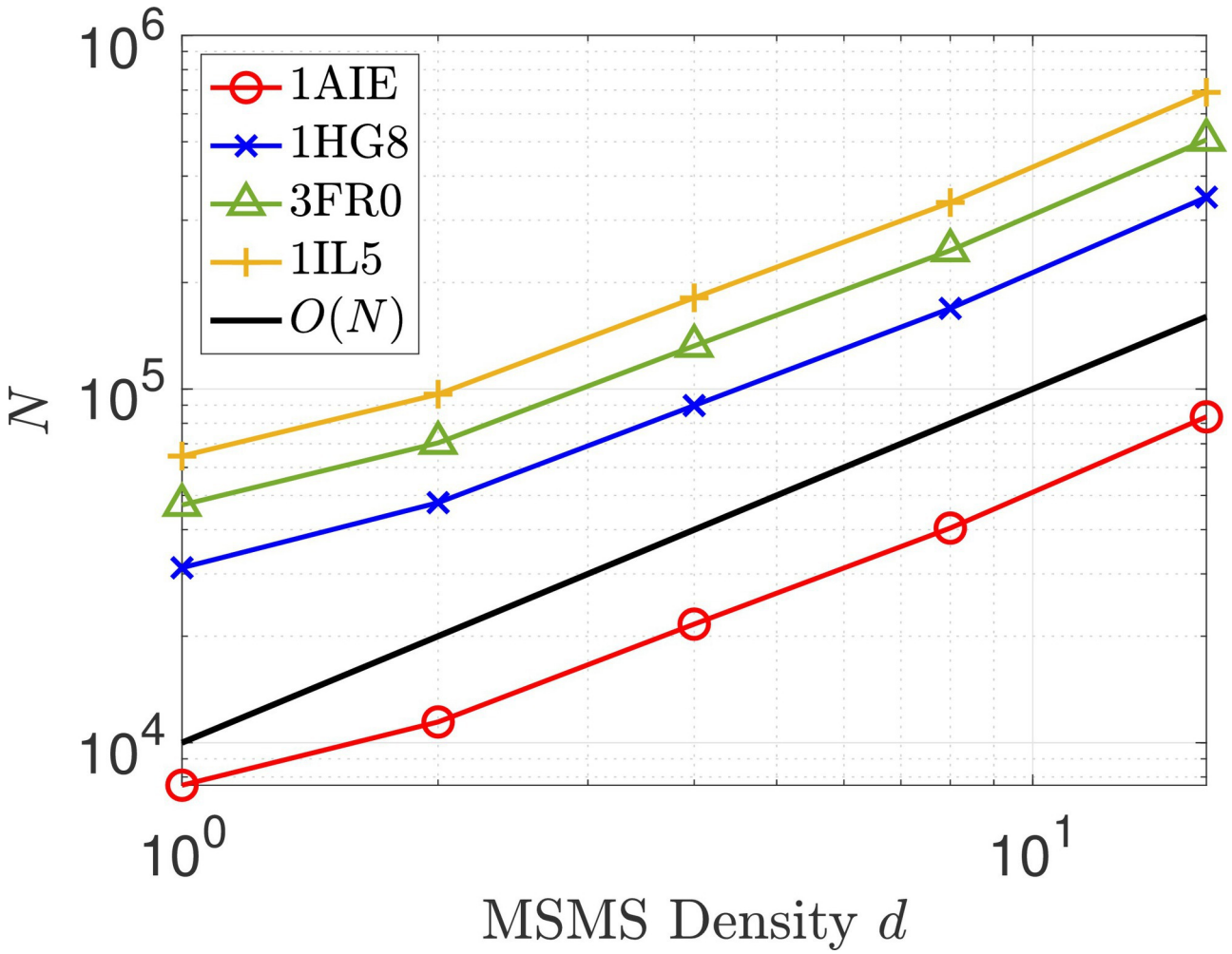


NanoShaper

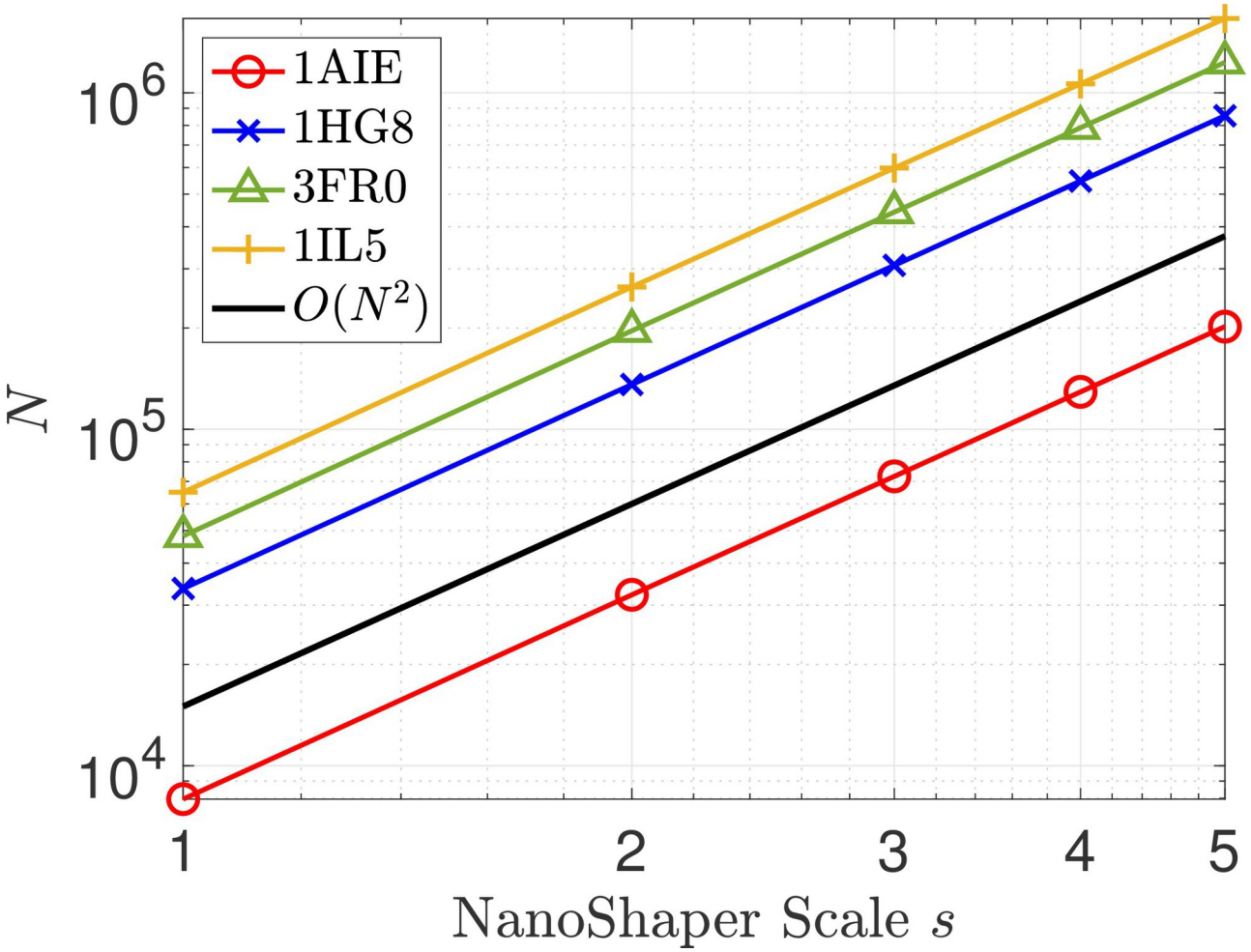
jcc_26692_graphical_abstract_2.eps



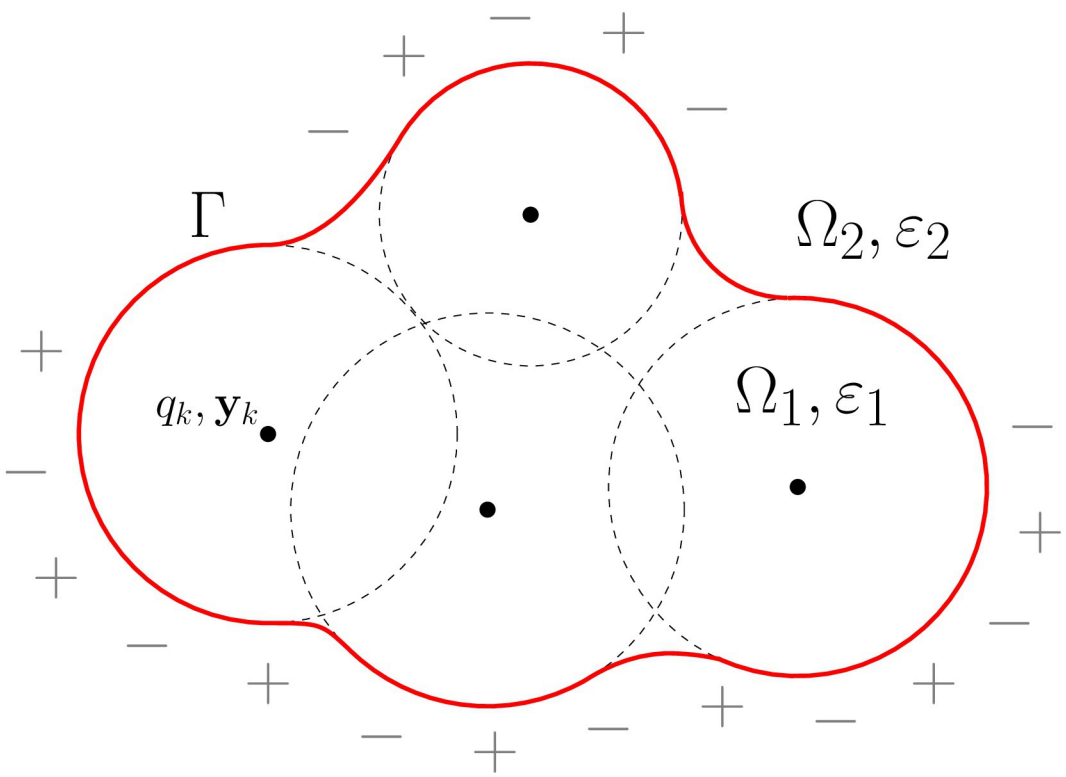
jcc_26692_meshtimetotal2loglog.eps



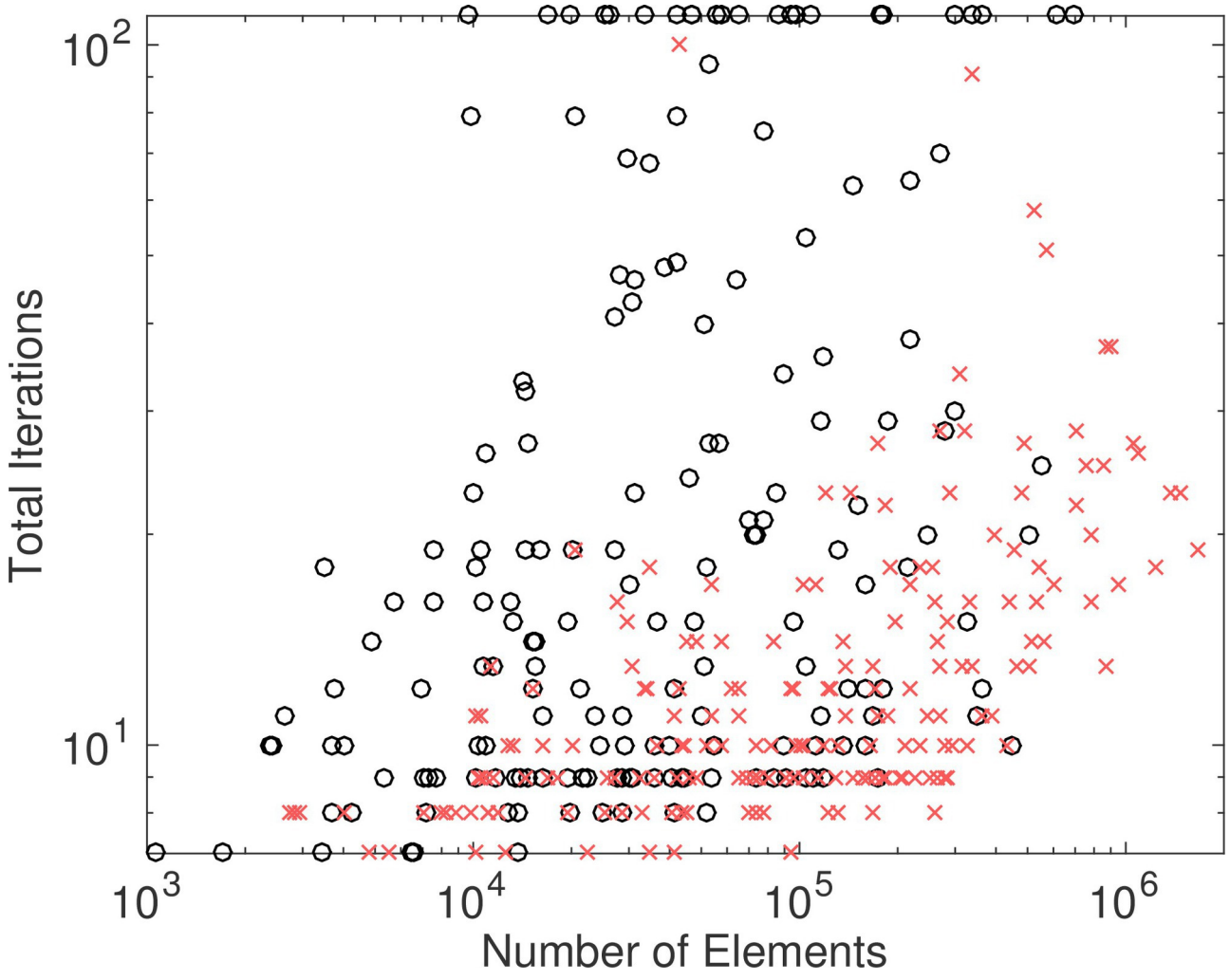
jcc_26692_msms_v_n_log.eps



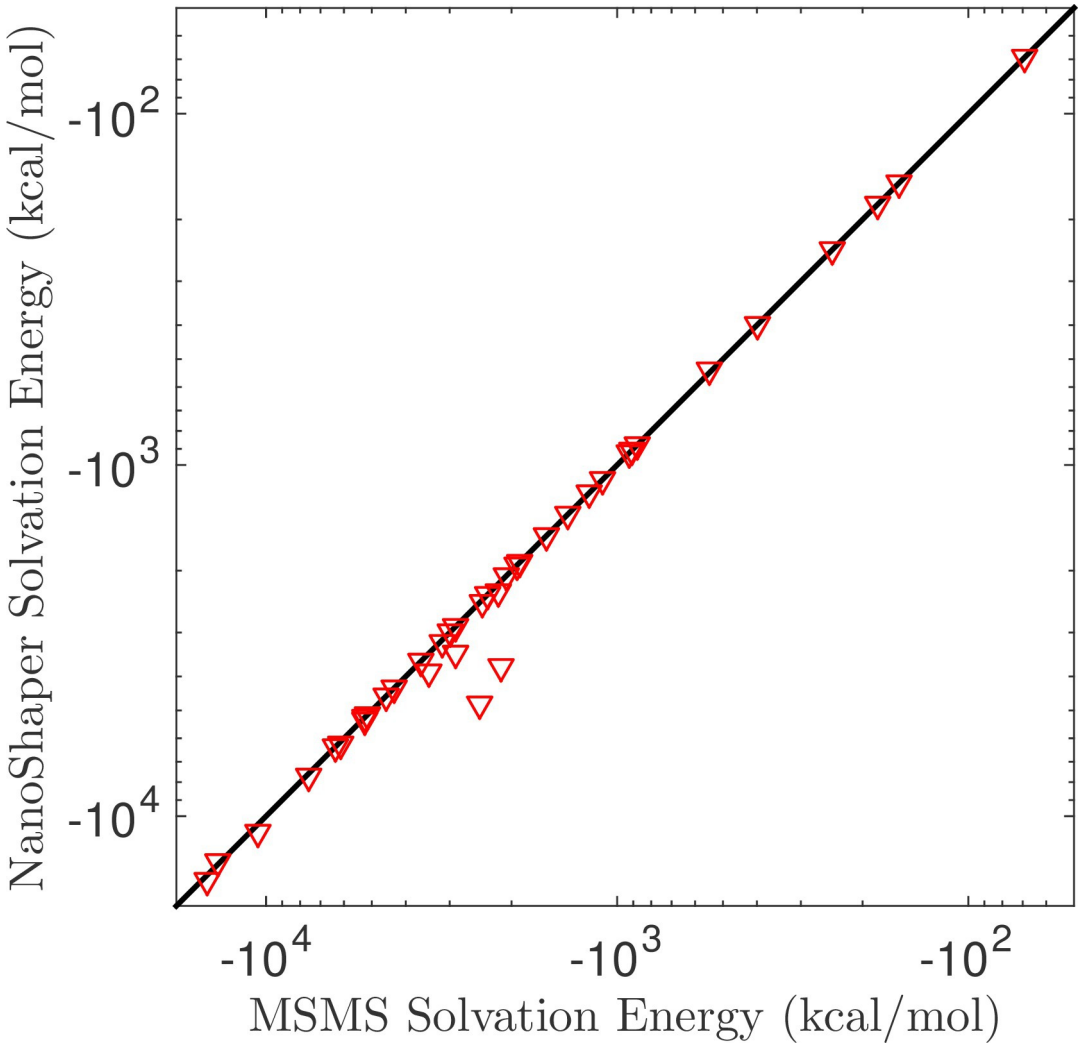
jcc_26692_nanoshaper_v_n_log.eps



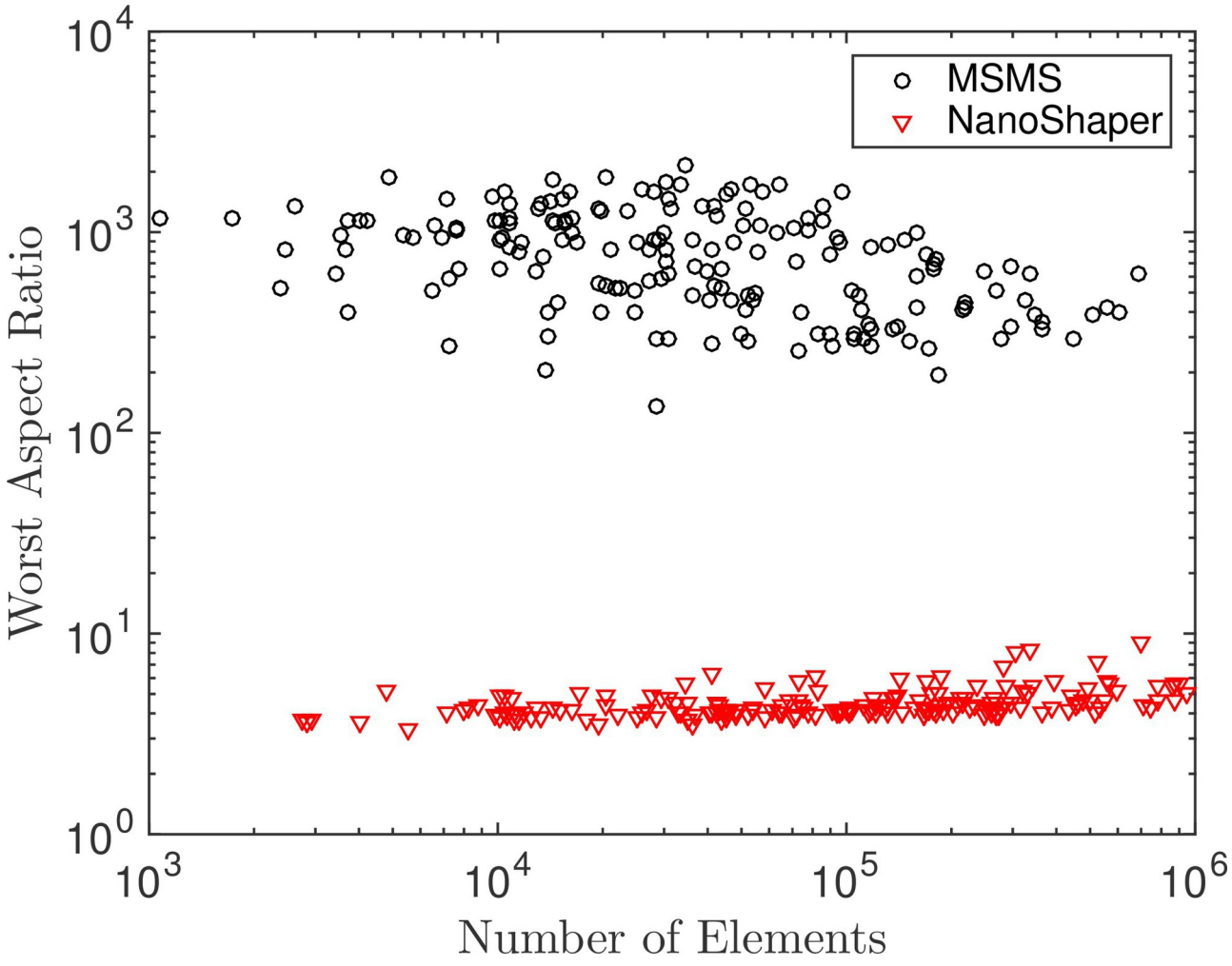
jcc_26692_pbmodel.eps



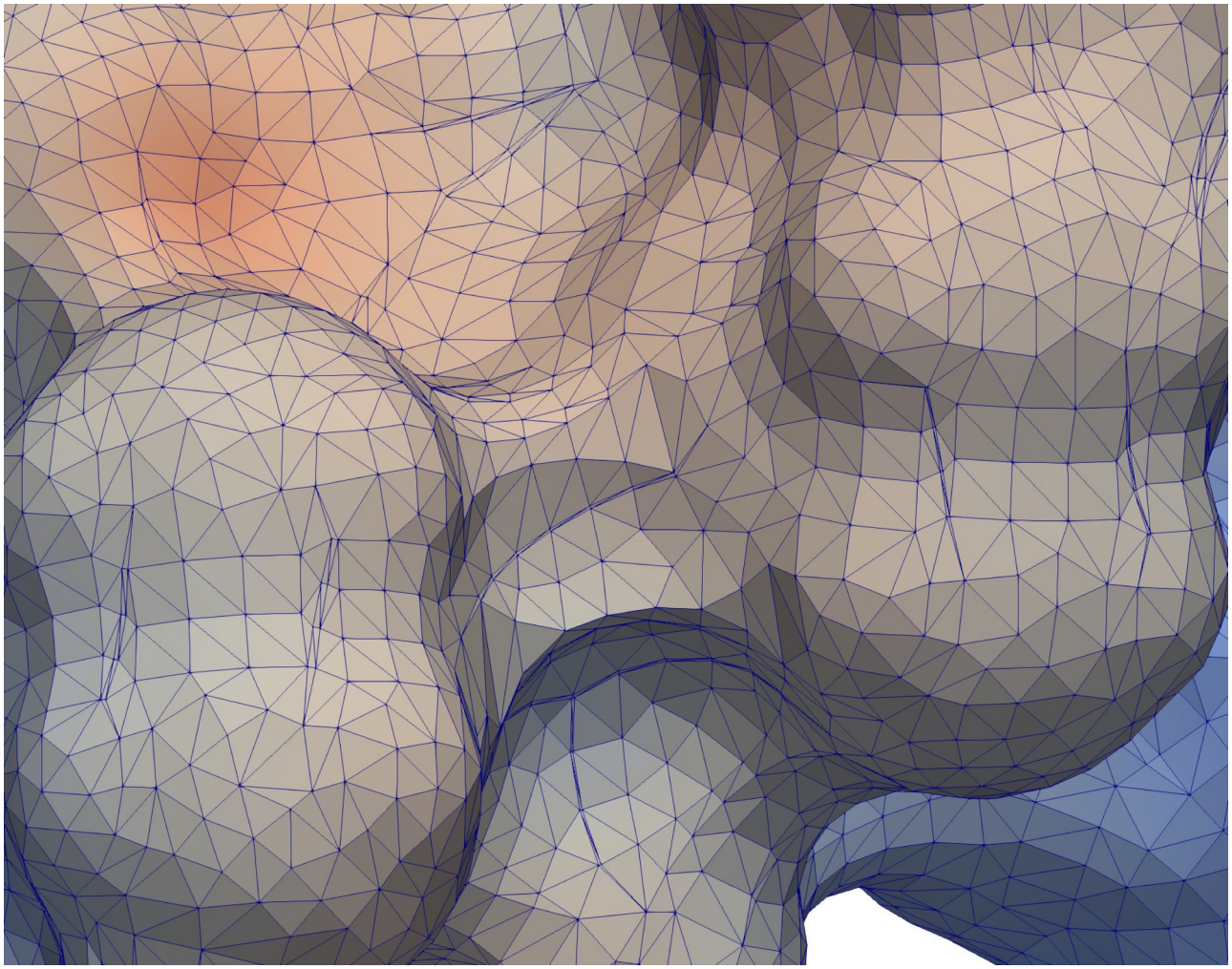
jcc_26692_plotsiter2.eps



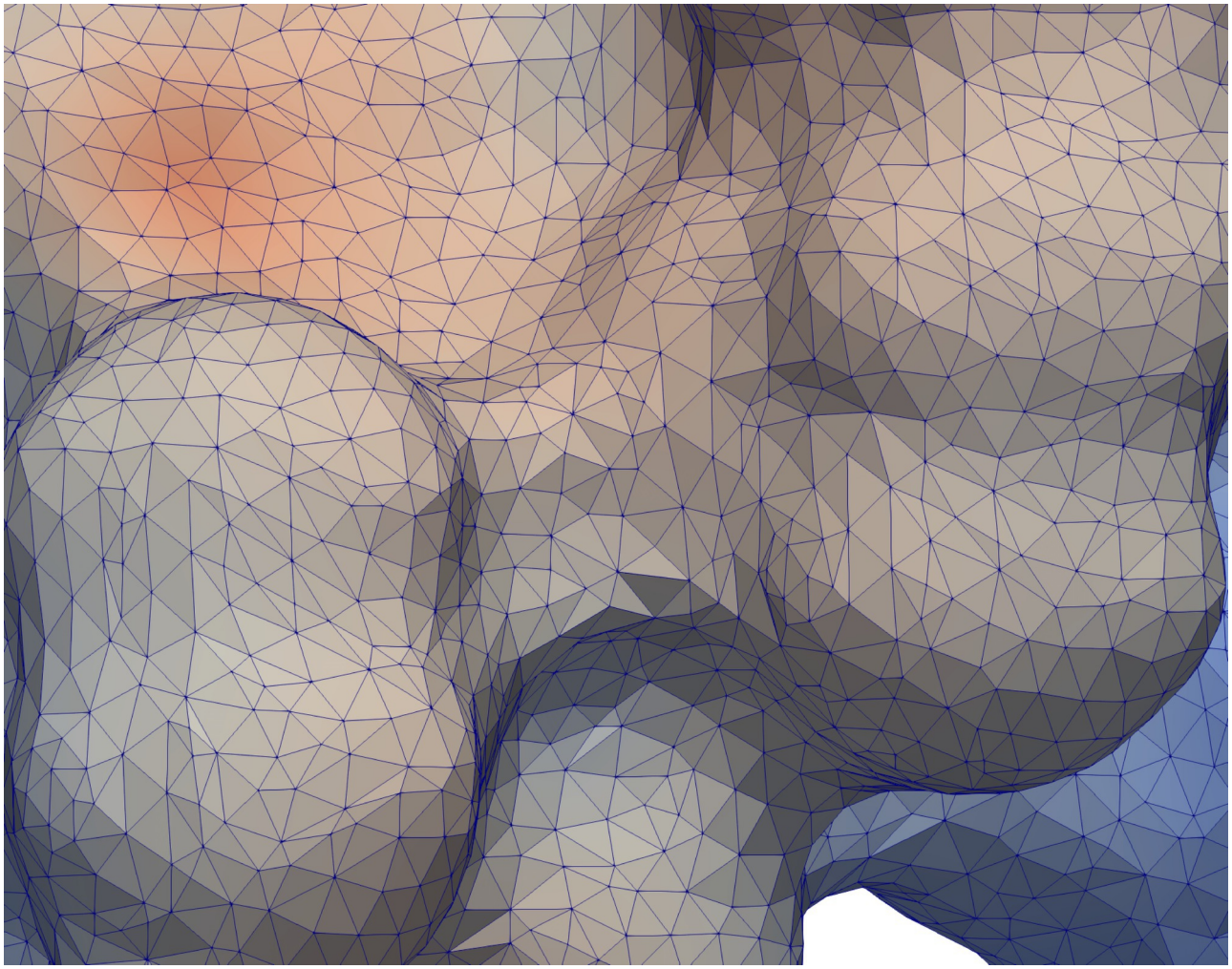
jcc_26692_solvengcomp2.eps



jcc_26692_worst_aspect_ratio.eps



jcc_26692_zoom2_1aie_msms_6.eps



jcc_26692_zoom2_1aie_nanoshaper_ses_2.eps