Commun. Comput. Phys.
doi: 10.4208/cicp.220711.041011s

Vol. **13**, No. 1, pp. 195-206
January 2013

Numerical Optimization of a Walk-on-Spheres Solver for the Linear Poisson-Boltzmann Equation

Travis Mackoy[1], Robert C. Harris[1,2], Jesse Johnson[1,2], Michael Mascagni[3] and Marcia O. Fenley[1,*]

[1] *Institute of Molecular Biophysics, Florida State University, Tallahassee, FL 32306, USA.*
[2] *Department of Physics, Florida State University, Tallahassee, FL 32306, USA.*
[3] *Departments of Computer Science, Mathematics and Scientific Computing, Florida State University, Tallahassee, FL 32306, USA.*

Received 22 July 2011; Accepted (in revised version) 4 October 2011

Available online 12 June 2012

Abstract. Stochastic walk-on-spheres (WOS) algorithms for solving the linearized Poisson-Boltzmann equation (LPBE) provide several attractive features not available in traditional deterministic solvers: Gaussian error bars can be computed easily, the algorithm is readily parallelized and requires minimal memory and multiple solvent environments can be accounted for by reweighting trajectories. However, previously-reported computational times of these Monte Carlo methods were not competitive with existing deterministic numerical methods. The present paper demonstrates a series of numerical optimizations that collectively make the computational time of these Monte Carlo LPBE solvers competitive with deterministic methods. The optimization techniques used are to ensure that each atom's contribution to the variance of the electrostatic solvation free energy is the same, to optimize the bias-generating parameters in the algorithm and to use an epsilon-approximate rather than exact nearest-neighbor search when determining the size of the next step in the Brownian motion when outside the molecule.

PACS: 02.70.Uu, 41.20Cv, 05.40.Fb

Key words: Poisson-Boltzmann equation, walk-on-spheres, solvation.

1 Introduction

Implicit-solvent models, like the Poisson-Boltzmann equation (PBE) are commonly used to account for the aqueous environments and ionic atmospheres of biomolecules in elec-

*Corresponding author. *Email addresses:* `mackoy@cs.fsu.edu` (T. Mackoy), `rch02c@fsu.edu` (R. C. Harris), `johnson@biomaps.rutgers.edu` (J. Johnson), `mascagni@fsu.edu` (M. Mascagni), `mfenley@sb.fsu.edu` (M. O. Fenley)

trostatic calculations without requiring the explicit inclusion of water molecules and ions [1–3]. Instead, the water is represented as a high-dielectric continuum, the ions are represented as a continuous charge distribution that obeys the Boltzmann distribution and the biomolecule is represented as a low-dielectric cavity containing point charges at the atomic centers. Unfortunately, the PBE is a nonlinear partial differential equation (PDE), which presents challenges to numerical solvers. Instead, the PBE is often linearized in the limit of small potentials, producing the linearized Poisson-Boltzmann equation, LPBE [4]. The LPBE has been applied to many biophysical problems and has been solved with several different numerical methods, including finite-difference [5–10], finite element [11–13], boundary element [14–19] and stochastic methods [20–24].

In particular, walk-on-spheres (WOS) [20–24], methods can compute the electrostatic solvation free energy, ΔG_{el}, accurately with several features unavailable in deterministic methods, including natural parallelizability, low memory overhead, easily computed Gaussian error bars and the ability to compute ΔG_{el} across multiple solvent conditions simultaneously, accounting for both changes in the dielectric environment and salt concentration. However, previously-reported timings for WOS methods were not competitive with deterministic alternatives. The present paper illustrates that numerically optimizing WOS methods by dividing the variance of ΔG_{el} evenly over all atoms, optimizing the bias generating parameters in the algorithm and including an epsilon-approximate rather than exact nearest-neighbor search when computing the size of the next Markov step during the walk outside the molecule produces computational times competitive with deterministic methods while retaining all of the previously-mentioned advantages.

2 Computational methods

2.1 Structure preparation and Poisson-Boltzmann calculations

The 55 proteins in this study were a data set used by Tjong and Zhou [25], which in turn were taken from the RCSB Protein Databank, PDB [26], with charges taken from the AMBER force field [27] and the radii taken from the set used by Bondi [28]. Unless otherwise stated, all calculations in this paper used a temperature of 298.15K, 0.5M 1:1 salt (NaCl), an interior dielectric constant of 1 and an exterior dielectric constant of 80. The selection of these parameters does not significantly affect the results presented here. All WOS calculations were performed on a single core of an Intel Core 2 Duo T6500 processor operating at 2.10GHz with 4GB of random access memory. The deterministic calculations used to compare to the WOS solver were performed with either the ACG [29] or APBS [3] programs. The calculations in ACG were performed on a grid that was 3 times larger than the largest dimension of the molecule with a minimum grid spacing of 0.3Å. To verify that these electrostatic solvation free energies are converged, the same calculations were performed at a minimum grid spacing of 0.2Å and the two sets of calculations fit to a best-fit line with a slope of 1.0 and $R^2 = 0.999$ (data not shown). All calculations were performed with double precision.

2.2 The walk-on-spheres Monte Carlo linearized Poisson-Boltzmann solver

As discussed in the introduction, the PBE replaces the electrostatic interactions between a biomolecule, its aqueous environment and its ionic atmosphere with the electrostatic interactions between the biomolecule and an exterior high-dielectric continuum containing a continuous charge distribution. The resulting electrostatic potential, ϕ, can then be found by solving the PBE,

$$\nabla^2\cdot\varphi(\mathbf{r})=\kappa^2\sinh(\varphi(\mathbf{r})), \tag{2.1}$$

outside the molecule. Here φ is the normalized potential, $\varphi=e\phi/kT$, where e is the fundamental charge, k is Boltzmann's constant and T is the temperature in Kelvin. For a 1:1 salt (e.g., NaCl), κ is the inverse Debye length,

$$\kappa^2=\frac{8\pi c_b e^2}{\varepsilon_{out}kT}, \tag{2.2}$$

where c_b is the bulk concentration of 1:1 salt and ε_{out} is the dielectric constant of the high-dielectric medium outside the molecule. Eq. (2.1) can be simplified in the limit of small φ to obtain the LPBE,

$$\nabla^2\cdot\varphi(\mathbf{r})=\kappa^2\varphi(\mathbf{r}). \tag{2.3}$$

Currently, the WOS method is restricted to solving the LPBE, but this restriction could be lifted by, for example, implementing branching WOS techniques [30]. Inside the molecule, the potential obeys the Poisson equation,

$$\nabla^2\cdot\varphi(\mathbf{r})=\sum q_i\delta(\mathbf{r}-\mathbf{r_i}), \tag{2.4}$$

where q_i is the (partial) charge on the ith atom and $\mathbf{r_i}$ is the location of the ith atom's center. In practice; however, because the primary goal of the WOS solver is to compute the electrostatic solvation free energy, ΔG_{el}, the quantity desired is not φ, but rather the reaction-field potential, $\varphi^{rf}=\varphi-\varphi^{coul}$, where φ^{coul} is the Coulombic vacuum potential, $\varphi^{coul}=\sum\frac{q_i}{\varepsilon_{in}|\mathbf{r}-\mathbf{r_i}|}$. Substituting into the above equations produces the following equations for φ^{rf},

$$\nabla^2\cdot\varphi^{rf}(\mathbf{r})=\kappa^2\varphi^{rf}(\mathbf{r}), \tag{2.5}$$

outside the molecule and the Laplace equation,

$$\nabla^2\cdot\varphi^{rf}(\mathbf{r})=0 \tag{2.6}$$

inside the molecule. These equations are then solved subject to the standard electrostatic boundary conditions [31]: at the molecular boundary,

$$\varphi^{rf}_{in}=\varphi^{rf}_{out} \tag{2.7}$$

and

$$\varepsilon_{in}\frac{\partial\varphi^{rf}_{in}}{\partial\hat{n}}=\varepsilon_{out}\frac{\partial\varphi^{rf}_{out}}{\partial\hat{n}}, \tag{2.8}$$

where φ_{in}^{rf} is the reaction-field potential inside the molecule, φ_{out}^{rf} is the reaction-field potential outside the molecule, ε_{in} is the dielectric constant inside the molecule, $\hat{n}$ is the normal to the surface of the biomolecule and

$$\varphi_{out}^{rf} \to 0 \tag{2.9}$$

at infinity. The WOS solver presented here solves for φ^{rf} at the i'th atomic center, ϕ_i^{rf}, by running a series of Brownian motion trajectories from the i'th atomic center.

First, ϕ_i^{rf} is estimated by ϕ^{rf} at a point on the boundary chosen by sampling from the distribution of first exit points of a Brownian walk using the walk-on-subdomains technique, as outlined in a previous study [21]. Then, ϕ^{rf} at this first exit point is estimated by ϕ^{rf} on a sphere of auxiliary radius α chosen according to a probability density that enforces the boundary conditions at the surface [20]. If the next point in this walk is inside, ϕ^{rf} is estimated by a point on the surface sampled from the distribution of first exit points on the surface of the molecule, as was done for the first step of the walk. If the new point is instead outside, a WOS process is started, which either terminates on the surface when it enters the absorbing layer of thickness τ, in which case another sampling to an auxiliary sphere is performed, or the walk dies because each step outside is accompanied by a killing probability $p = \sinh(\kappa d)/\kappa d$, where d is the size of the step outside. Because the killing probability is finite, the walk eventually terminates. Each step outside is taken on a sphere centered at the walker's current location with a radius equal to the distance from the walker to the nearest point on the surface. Once φ^{rf} has been obtained at the atomic centers, ΔG_{el} can be computed by

$$\Delta G_{el} = \frac{1}{2}\sum q_i \phi_i^{rf}. \tag{2.10}$$

One difficulty with PBE methods is that how to define the boundary between the interior and exterior regions is not settled [32]. The WOS method presented here contains a van der Waal's surface definition, where the low-dielectric interior of the biomolecule is considered to be the union of spheres centered at the atomic centers with van der Waal's radii (see Fig. 1). However, the WOS method could be readily modified to use other molecular surface definitions, including the solvent-excluded, SE, molecular surface [33]. Solving the PBE for the SE surface, for example, could be accomplished by increasing the radii of atoms that are not solvent-exposed by the radius of the water probe and treating the nonspherical reentrant region as a third region in which a WOS trajectory could be used, as in the exterior region. A Stern layer could also be incorporated by treating the ion-exclusion region as an additional domain through which a WOS trajectory could pass, as in the exterior region.

2.3 Variance balancing

Previously, WOS methods computed ΔG_{el} by running an equal number of trajectories started from each atomic center until ΔG_{el} converged to an acceptably small standard de-

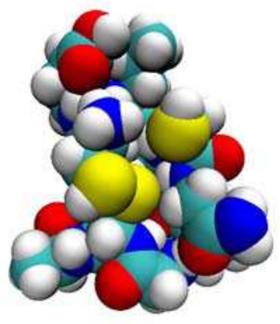

Figure 1: An image of an enterotoxin (PDB id: 1ETL) from the data set used in this paper showing the van der Waal's surface definition. The interior of the molecule is the union of spheres centered at the atomic centers with radii equal to the van der Waal's radii.

viation, σ [20–24]. This method is inefficient because not all of the atoms' contributions, $\mathrm{var}(\Delta G_{el}^{i})$, to the variance of ΔG_{el}, $\mathrm{var}(\Delta G_{el})$ are equal. When an equal number of trajectories are run from each atom, too much time is spent converging atoms with small $\mathrm{var}(\Delta G_{el}^{i})$. Instead, the computational time, t, can be reduced significantly by ensuring that each atom's $\mathrm{var}(\Delta G_{el}^{i})$, is the same. This condition can be obtained by running a small number of trajectories, N_{small}, over all atoms to estimate ΔG_{el}, ΔG_{el}^{i} and their variances over the small run, $[\mathrm{var}(\Delta G_{el})]_{\mathrm{small}}$ and $[\mathrm{var}(\Delta G_{el}^{i})]_{\mathrm{small}}$. The desired variance of each atom, δ_i, can then be estimated by computing

$$\delta_i = [\Delta G_{el}]^2_{small} \delta / N_{atoms}, \tag{2.11}$$

where $\delta = \sigma^2_{desired}$, where $\sigma_{desired}$ is the desired percent standard deviation of ΔG_{el} and N_{atoms} is the number of atoms in the molecule. As in most Monte Carlo methods, $[\mathrm{var}(\Delta G_{el}^{i})] \propto 1/N_{traj}^{i}$, where N_{traj}^{i} is the number of trajectories run from the i'th atom and therefore N_{traj}^{i} can be estimated by

$$N_{traj}^{i} = \frac{[\mathrm{var}(\Delta G_{el}^{i})]_{\mathrm{small}}}{\delta_i} N_{small}. \tag{2.12}$$

This method also allows ΔG_{el} to be converged to an arbitrary accuracy without reference to any prior knowledge of ΔG_{el}.

In the resulting calculations, the $\mathrm{var}(\Delta G_{el}^{i})$ are more uniform than when variance balancing is not performed, as illustrated in Fig. 2. To create this figure, the atoms were sorted in ascending order by $\mathrm{var}(\Delta G_{el}^{i})$ and the cumulative variance, or the sum of the $\mathrm{var}(\Delta G_{el}^{i})$ of the first n atoms as a fraction of $\mathrm{var}(\Delta G_{el})$, for runs with and without variance balancing was plotted as a function of n for an enterotoxin (PDB id: 1ETL). Many atoms in the run without variance balancing had negligible $\mathrm{var}(\Delta G_{el}^{i})$ and therefore, too much work was spent on converging them. This problem was less pronounced in the run with variance balancing, as can be seen from Table 1, where an energy computation was

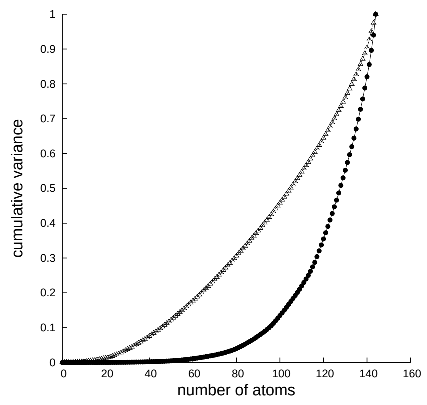

Figure 2: The atoms were sorted by their contribution to the total variance from smallest to largest and the cumulative variance, or the sum of the variances of all atoms less than atom n as a fraction of the total variance, for runs with (circles) and without (triangles) variance balancing is plotted as a function of n for an enterotoxin (PDB id: 1ETL).

performed with variance balancing until an overall 1% σ was reached and then a calculation was run where the same total number of trajectories was divided evenly across all atoms. On average, σ, was 1.49 times larger for the runs without variance balancing and $t \propto 1/\sigma^2$, variance balancing on average decreases t by a factor of 2.22.

2.4 Bias optimization

As discussed in previous studies [20, 23], WOS solvers contain two bias-generating parameters: the thickness of the absorbing layer, τ and the size of the auxiliary sphere, α. Optimizing the bias produced by α and τ is essential when computing ΔG_{el} because both parameters significantly alter t. As shown in a previous study [23],

$$t \sim A - B\log(\tau), \tag{2.13a}$$

$$t \propto \frac{1}{\alpha}, \tag{2.13b}$$

where A and B are positive functions independent of τ. As was shown before [20], the theoretical bias in the estimate of the potential at a point on the surface of a sphere of radius R from a single auxiliary jump is

$$\text{Bias}(\phi(x)) = \phi(x)\left(\frac{\alpha}{2R}\right)^3. \tag{2.14}$$

The order of the bias in the energy due to α is then

$$\text{Bias}(E) = En_{hits}\left(\frac{\alpha}{2R}\right)^3, \tag{2.15}$$

where n_{hits} is the number of boundary hits per trajectory.

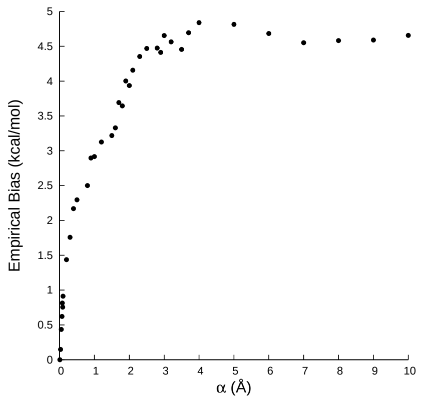

Figure 3: The empirical bias due to the auxiliary sphere radius,α, taken here to be the difference between ΔG_{el} at a given value of α and that at $\alpha = 0.01$Å converged to 0.1% at 0.01M NaCl for the first ASP residue in the pheromone ER-1 (PDB id: 2erl) plotted against α. The bias was not observable for any protein in the current data set.

Similarly, the order of the bias in the energy due to τ is [20]

$$\text{Bias}(E) = E n_{hits}\left(\frac{\tau}{2R}\right). \tag{2.16}$$

These two sources of bias are independent and because $\log(\tau)$ increases more slowly than τ, t can be minimized by using as small a τ as possible and placing all of the bias into the term depending on α. In addition, because the survival probability is proportional to $1/\alpha$, $n_{hits} \propto 1/\alpha$. The bias can therefore be set to a value smaller than $\sigma_{desired}\Delta G_{el}$ by running the simulation with a fixed value of α, α_{small}, for a small number of trajectories and then resetting α afterwards to

$$\alpha = \left((8R^3\sigma)/(n_{hits}^{small}\alpha_{small})\right)^{\frac{1}{2}}, \tag{2.17}$$

where n_{hits}^{small} is the average number of hits per trajectory during the initial small run.

Bias optimization decreased t by an average factor of 3.85 (Table 1). Fortunately, the bias due to α is typically smaller than that predicted by Eq. (2.17) because the maximum size of this bias is the difference between ΔG_{el} computed at infinite salt and ΔG_{el} computed at the salt concentration of interest and this difference is typically quite small. To demonstrate this overestimation of the bias, the empirical bias for the first ASP residue of the pheromone ER-1 (PDB id: 2erl) as a function of α is plotted in Fig. 3. This quantity was computed by taking the difference between the energy computed at different values of α and $\alpha = 0.01$Å at an accuracy of 0.1% at a salt concentration of 0.01M NaCl.

2.5 Approximate nearest-neighbor searches

The slowest component of the WOS algorithm is the nearest-neighbor search that must be performed each step during the walk outside the molecule. The algorithm's speed can therefore be improved with an epsilon-approximate rather than exact nearest-neighbor search. In the results presented here, nearest-neighbor searches were performed with the ANN library [34], which allows the accuracy of the search to be adjusted with an epsilon parameter, ε and returns a sphere that is within $1+\varepsilon$ times the distance to the nearest sphere. Increasing epsilon typically increases the number of trajectories required to reach a converged solution while decreasing the execution time of each trajectory. For the molecules in the present study, $\varepsilon=5$ provided a good compromise and the resulting calculations were an average of 1.6 times faster than those run with $\varepsilon=0$.

3 Results

Collectively, the numerical optimizations described in Methods allow the LPBE to be solved in times competitive with traditional finite-difference solvers, as none of these finite-difference calculations took more than a few minutes. Unfortunately, quantifying the difference in execution time between the two methods is not possible because the error in the electrostatic solvation energy cannot be extracted readily from the deterministic solvers. However the apparent error in the deterministic calculations appears to be approximately 1% and therefore ΔG_{el} was converged to 1% by the WOS solver in this study. The resulting predictions of ΔG_{el} are plotted against those computed with the ACG finite difference solver in Fig. 4. The slope of the best-fit line is 0.99 and $R^2=0.999$. The same calculations were also performed with the APBS solver and the resulting best-fit line had a slope of 0.99 and $R^2=0.999$ (data not shown). The complexes used in this study contained between 145 and 3564 atoms and the execution time was competitive with traditional deterministic solvers, as the 55 protein calculations took between 47 seconds and 630 seconds (Table 1). Collectively, the three optimizations accelerated the code by an average factor of 20.

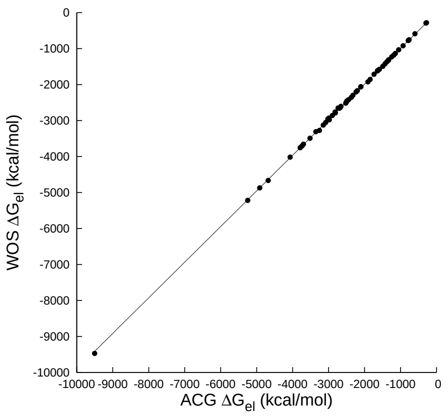

Figure 4: The electrostatic solvation free energy, ΔG_{el} converged to 1% desired standard deviation by the stochastic walk-on-spheres, WOS, solver for all 55 proteins in this data set plotted against the same quantity computed with the deterministic solver ACG. The slope of the plot is 0.99 and $R^2=0.999$.

Table 1: For each protein in the data set, the electrostatic solvation free energy, ΔG_{el}, was computed with variance balancing and converged to a 1% desired standard deviation, $\sigma_{desired}$. That number of trajectories was then split evenly across all atoms and another calculation without variance balancing was performed. Variance balancing reduced the standard deviation of ΔG_{el}, σ, by on average a factor of 1.49 and because the computational time is inversely proportional to σ^2, variance balancing on average reduced the computation time by a factor of 2.22. Similar runs were then performed with an ANN parameter of 5.0 and with bias optimization and the acceleration in the execution time is shown in the table. Also shown are the final timings with all optimizations.

PDB id	Trajectories /Atom	Without Optimization		Variance Balancing		Variance Balancing Improvement in σ	Variance Balancing Acceleration	ANN Acceleration	Bias Optimization Acceleration	Final Timings (s)
		ΔG_{el} (kcal/mol)	σ (kcal/mol)	ΔG_{el} (kcal/mol)	σ (kcal/mol)					
1A6M	74	-2709	31	-2725	20	1.56	2.44	2.38	3.82	314
1AH0	150	-1268	15	-1294	11	1.39	1.92	2.40	4.63	129
1BYI	64	-3553	42	-3593	26	1.58	2.51	2.25	3.48	463
1C75	116	-1378	15	-1403	11	1.43	2.05	2.45	3.86	111
1C7K	96	-2364	29	-2459	20	1.47	2.17	2.44	4.46	270
1CEX	78	-2760	34	-2887	22	1.55	2.40	2.24	4.07	433
1EB6	36	-4961	57	-5137	36	1.61	2.58	2.36	2.16	217
1EJG	391	-572	8	-572	5	1.39	1.94	2.43	5.30	350
1ETL	874	-285	3	-288	2	1.17	1.37	1.69	2.33	58
1EXR	24	-9233	82	-9432	53	1.53	2.34	2.15	1.55	108
1F94	155	-1173	14	-1218	10	1.38	1.91	2.10	5.25	133
1F9Y	78	-2956	33	-2949	21	1.55	2.39	2.36	4.06	293
1G4I	100	-2311	28	-2385	18	1.50	2.24	2.21	4.60	243
1G66	109	-2906	34	-2951	23	1.53	2.34	2.25	5.14	483
1GQV	76	-2530	31	-2570	21	1.51	2.28	2.15	4.00	263
1HJE	525	-274	3	-274	2	1.18	1.38	1.77	2.37	65
1IQZ	23	-4670	41	-4668	27	1.51	2.29	2.45	1.38	56
1IUA	163	-1269	15	-1274	10	1.48	2.19	2.35	5.24	251
1J0P	56	-2851	29	-2799	20	1.46	2.14	2.12	2.86	131
1K4I	67	-3813	44	-3899	28	1.55	2.39	2.18	3.67	412
1KTH	139	-1438	15	-1454	11	1.37	1.88	2.35	4.41	90
1L9L	45	-3032	32	-3110	22	1.46	2.14	2.36	2.40	68
1M1Q	67	-2332	26	-2377	17	1.51	2.27	2.12	3.30	89
1NLS	59	-4670	54	-4699	34	1.58	2.49	2.26	3.36	370
1NWZ	68	-2728	32	-2739	20	1.56	2.42	2.41	3.50	202
1OD3	106	-2005	26	-2050	17	1.51	2.29	2.51	4.78	285
1OK0	108	-1516	18	-1536	13	1.43	2.04	2.44	4.11	107
1P9G	194	-762	10	-747	7	1.36	1.85	2.26	3.74	188
1PQ7	101	-2502	31	-2607	20	1.53	2.35	2.21	4.93	539
1R6J	145	-1330	15	-1329	10	1.46	2.13	2.43	4.95	173
1SSX	79	-2561	33	-2631	21	1.56	2.43	2.24	4.13	609
1TG0	38	-3177	34	-3266	22	1.57	2.48	2.65	2.12	47
1TQG	62	-2874	32	-2911	20	1.57	2.45	2.32	3.25	160
1TT8	97	-2526	30	-2580	19	1.55	2.41	2.30	4.72	370
1U2H	96	-2024	23	-2035	15	1.48	2.19	2.40	4.28	162
1UCS	215	-1000	11	-1020	8	1.42	2.02	2.44	4.93	146
1UFY	93	-2281	26	-2298	17	1.52	2.31	2.44	4.38	242
1UNQ	55	-3391	39	-3431	25	1.52	2.32	2.33	3.00	163
1VB0	158	-1128	14	-1111	9	1.47	2.15	2.30	4.83	165
1VBW	78	-1784	20	-1808	14	1.50	2.25	2.75	3.39	112
1W0N	93	-2384	29	-2441	19	1.48	2.19	2.57	4.15	187
1WY3	223	-757	9	-773	6	1.39	1.94	2.51	4.02	107
1X6Z	89	-2089	26	-2183	17	1.55	2.40	2.42	4.14	174
1X8Q	71	-3514	40	-3584	26	1.54	2.39	2.29	3.78	333
1XMK	120	-1565	19	-1591	13	1.49	2.23	2.43	4.59	181
1YK4	55	-1864	22	-1888	14	1.57	2.45	2.47	2.49	48
1ZZK	106	-1549	20	-1596	13	1.52	2.30	2.49	4.17	149
2A6Z	66	-3633	42	-3674	27	1.57	2.48	2.26	3.64	467
2BF9	199	-915	9	-901	7	1.39	1.92	2.43	4.20	61
2CHH	117	-2116	25	-2126	17	1.51	2.29	2.42	5.01	201
2CWS	78	-3129	39	-3206	25	1.57	2.45	2.17	4.25	630
2ERL	105	-1147	13	-1160	9	1.41	2.00	2.31	3.37	57
2FDN	72	-1690	19	-1695	13	1.51	2.29	2.32	3.00	47
2FWH	85	-2216	26	-2234	16	1.55	2.39	2.47	4.06	211
3LZT	82	-2505	31	-2575	20	1.53	2.35	2.33	4.00	279

4 Conclusions

Although previous results with WOS solvers indicated that they were unacceptably slow compared to deterministic methods, the results presented here indicate that when appropriate numerical optimizations were performed, this WOS solver computed ΔG_{el} in times that were competitive with traditional deterministic methods. The resulting predictions of ΔG_{el} could be converged to arbitrary precision without reference to any prior knowledge and in addition, WOS algorithms have several attractive features not available in deterministic solvers, including well-behaved Gaussian error predictions, trivial parallelizability, minimal memory requirements and the ability to run at multiple solvent conditions simultaneously, making them attractive for biophysical applications.

Acknowledgments

One of the authors (Marcia O. Fenley) acknowledges the support of NIH grant 5R44GM073391-03 (PI: Dr. Alexander H. Boschtisch).

References

[1] B. Z. Lu, Y. C. Zhou, M. J. Holst and J. A. McCammon, Recent progress in numerical methods for the Poisson-Boltzmann equation in biophysical applications, Commun. Comput. Phys., 3 (2008), 973–1009.

[2] G. Lamm, The Poisson-Boltzmann equation, Rev. Comput. Chem., 19 (2003), 147–365.

[3] N. A. Baker, D. Sept, S. Joseph, M. J. Holst and J. A. McCammon, Electrostatics of nanosystems: application to microtubules and the ribosome, Proc. Natl. Acad. Sci. USA, 98 (2001), 10037–10041.

[4] F. Fogolari, P. Zuccato, G. Esposito and P. Viglino, Biomolecular electrostatics with the linearized Poisson-Boltzmann equation, Biophys. J., 76 (1999), 1–16.

[5] M. E. Davis and J. A. McCammon, Solving the finite difference linearized Poisson-Boltzmann equation: a comparison of relaxation and conjugate gradient methods, J. Comput. Chem., 10 (1989), 386–391.

[6] A. Nicholls and B. Honig, A rapid finite difference algorithm, utilizing successive over-relaxation to solve the Poisson-Boltzmann equation, J. Comput. Chem., 12 (1991), 435–445.

[7] W. Geng, S. Yu and G. Wei, Treatment of charge singularities in implicit solvent models, J. Chem. Phys., 127 (2007), 114106.

[8] W. Rocchia, S. Sridharan, A. Nicholls, E. Alexov, A. Chiabrera and B. Honig, Rapid grid-based construction of the molecular surface for both Molecules and geometric objects: applications to the finite difference Poisson-Boltzmann method, J. Comput. Chem., 23 (2002), 128–137.

[9] M. E. Davis, J. D. Madura, B. A. Luty and J. A. McCammon, Electrostatics and diffusion of molecules in solution: simulations with the University of Houston Brownian dynamics program, Comput. Phys. Commun., 62 (1991), 187–197.

[10] D. Bashford, Scientific Computing in Object-Oriented Parallel Environments, Y. Ishikawa, R. R. Oldehoeft, J. Reynders and V. W. M. Tholburn, Springer Berlin Heidelberg, 1343 (1997), 233–240.

[11] M. Holst and F. Saied, Multigrid solution of the Poisson-Boltzmann equation, J. Comput. Chem., 14 (1993), 105–113.

[12] M. Holst, N. Baker and F. Wang, Adaptive multilevel finite element solution of the Poisson-Boltzmann equations I: algorithms and examples, J. Comput. Chem., 21 (2000), 1319–1342.

[13] C. M. Cortis and R. A. Friesner, Numerical solution of the Poisson-Boltzmann equation using tetrahedral finite-element meshes, J. Comput. Chem., 18 (1997), 1591–1608.

[14] B. J. Yoon and A. M. Lenhoff, A boundary element method for molecular electrostatics with electrolyte effects, J. Comput. Chem., 11 (1990), 1080–1086.

[15] A. H. Boschitsch, M. O. Fenley and H.-X. Zhou, Fast boundary element method for the linear Poisson-Boltzmann equation, J. Phys. Chem. B, 106 (2002), 2741–2754.

[16] D. M. Chipman, Solution of the linearized Poisson-Boltzmann equation, J. Chem. Phys., 120 (2004), 5566–5575.

[17] M. D. Altman, J. P. Bardhan, J. K. White and B. Tidor, Accurate solution of multi-region continuum electrostatic problems using the linearized Poisson-Boltzmann equation and curved boundary elements, J. Comput. Chem., 30 (2009), 132–153.

[18] E.-H. Yap and T. Head-Gordon, A new and efficient Poisson-Boltzmann solver for interaction of multiple proteins, J. Chem. Theory Comput., 6 (2010), 2214–2224.

[19] A. Juffer, E. F. F. Botta, B. A. M. van Keulen, A. van der Ploeg and H. J. C. Berendsen, The electric potential of a macromolecule in a solvent: a fundamental approach, J. Comput. Phys., 97 (1991), 144–171.

[20] N. A. Simonov, Doklady Mathematics, Springer, 656–659.

[21] N. A. Simonov, M. Mascagni and M. O. Fenley, Monte Carlo-based linear Poisson-Boltzmann approach makes accurate salt-dependent solvation free energy predictions possible, J. Chem. Phys., 127 (2007), 185105.

[22] M. Bossy, N. Champagnat, S. Maire and D. Talay, Probabilistic interpretation and random walk on spheres algorithms for the Poisson-Boltzmann equation in molecular dynamics, ESAIM. Math. Model. Numer. Anal., 44 (2010), 997–1048.

[23] M. O. Fenley, M. Mascagni, J. McClain, A. R. J. Silalahi and N. A. Simonov, Using correlated Monte Carlo sampling for efficiently solving the linearized Poisson-Boltzmann equation over a broad range of salt concentration, J. Chem. Theory Comput., 6 (2009), 300–314.

[24] M. Mascagni and N. A. Simonov, Monte Carlo methods for calculating some physical properties of large molecules, SIAM J. Sci. Comput., 26 (2005), 339.

[25] H. Tjong and H.-X. Zhou, GBr^6: a parameterization-free, accurate, analytical generalized Born method, J. Phys. Chem. B, 111 (2007), 3055–3061.

[26] H. M. Berman, T. N. Bhat, P. E. Bourne, Z. Feng, G. Gilliland, H. Weissig and J. Westbrook, The protein data bank and the challenge of structural genomics, Nat. Struct. Mol. Biol., 7 (2000), 957–959.

[27] W. D. Cornell, P. Cieplak, C. I. Bayly, I. R. Gould, K. M. Merz, Jr. D. M. Ferguson, D. C. Spellmeyer, T. Fox, J. W. Caldwell and P. A. Kollman, A second generation force field for the simulation of proteins, nucleic acids, and organic molecules, J. Am. Chem. Soc., 117 (1995), 5179–5197.

[28] A. Bondi, van der Waals Volumes and Radii, J. Phys. Chem., 68 (1964), 441–451.

[29] A. H. Boschitsch and M. O. Fenley, A fast and robust Poisson-Boltzmann solver based on adaptive cartesian grids, J. Chem. Theory Comput., 7 (2011), 1524–1540.

[30] A. Rasulov, A. Karaivanova and M. Mascagni, Quasirandom sequences in branching random walks, Monte Carlo Methods Appl., 10 (2004), 551–558.

[31] J. D. Jackson, Classical Electrodynamics Third Edition, Wiley, 1998.

[32] S. Qin and H.-X. Zhou, Do electrostatic interactions destabilize protein-nucleic acid binding?, Biopolymers, 86 (2007), 112–118.

[33] M. L. Connolly, Solvent-accessible surfaces of proteins and nucleic acids, Science, 221 (1983), 709–713.

[34] D. M. Mount and S. Arya, ANN: A library for approximate nearest neighbor searching, Proc. Center for Geometric Computing Second Ann. Fall Workshop Computational Geometry, 1997.