Predicting Osmotic Coefficients from Radial Distribution Functions

Source: https://doi.org/10.1021/acs.jctc.4c00363

Step 1: Import data

In[1]:=
```
ReplaceNaN[x_?ArrayQ, rep_ : Indeterminate] := x /. Indeterminate → rep
```

In[2]:=
```
e0 = 1.60217663 * 10 ^ -19;
epsilon0 = 8.85418782 * 10 ^ -12;
epsilonr = 102;
zNa = 1;
zCl = -1;
ρNa = 6.02214076 * 10 ^ 23 * 0.15 * 1000;
ρCl = 6.02214076 * 10 ^ 23 * 0.15 * 1000;
β = 1 / (1.380649 × 10 ^ -23 × 303.15);
rMin = 0.2 * 10 ^ -9;
ord = 2;
```

In[12]:=
```
(*1. Import 0.15M NaCl solution box Na-Na RDF data generated by GROMACS*)
dataNaNa =
  Import["/Users/heiley/Desktop/mathematica/charge0/rdf_na-na.xvg", "Table"];
rNaNa = dataNaNa[[All, 1]] * 10 ^ -9;
gNaNa = dataNaNa[[All, 2]];
(*2. Interpolate to create a curve*)
(*Spline method ensures a continuous derivative*)
grNaNa = Interpolation[Transpose[{rNaNa, gNaNa}], Method → "Spline"];
(*3. Take the moving average to smooth the curve*)
grNaNaavg =
  MovingAverage[Table[grNaNa[x], {x, 0., 10.002 * 10 ^ -9, 0.002 * 10 ^ -9}], 3];
(*4. Interpolate to create a curve*)
grNaNaavgint = Interpolation[Transpose[{rNaNa, grNaNaavg}], Method → "Spline"];
```

In[18]:=
```
(*Repeat with Cl-Cl RDF*)
dataClCl = Import[
   "/Users/heiley/Desktop/mathematica/charge0/rdf_nacl-cl.xvg", "Table"];
rClCl = dataClCl[[All, 1]] * 10 ^ -9;
gClCl = dataClCl[[All, 2]];
grClCl = Interpolation[Transpose[{rClCl, gClCl}], Method → "Spline"];
grClClavg =
  MovingAverage[Table[grClCl[x], {x, 0., 10.002 * 10 ^ -9, 0.002 * 10 ^ -9}], 3];
grClClavgint = Interpolation[Transpose[{rClCl, grClClavg}], Method → "Spline"];
```

In[24]:=
```
(*Repeat with Na-Cl RDF*)
dataNaCl =
  Import["/Users/heiley/Desktop/mathematica/charge0/rdf_na-cl.xvg", "Table"];
rNaCl = dataNaCl[[All, 1]] * 10^-9;
gNaCl = dataNaCl[[All, 2]];
grNaCl = Interpolation[Transpose[{rNaCl, gNaCl}], Method → "Spline"];
grNaClavg =
  MovingAverage[Table[grNaCl[x], {x, 0., 10.006 * 10^-9, 0.002 * 10^-9}], 5];
grNaClavgint = Interpolation[Transpose[{rNaCl, grNaClavg}], Method → "Spline"];
```

Step 2: Determine rMax to achieve charge neutrality

$$\int_{r_{min}}^{r_{max}} 4\pi r^2 (z_{Na^+} e_0 \rho_{Na^+} g_{++} + z_{Cl^-} e_0 \rho_{Cl^-} g_{-+})\, dr = -e_0$$

- Typically, a value approximately half the size of the simulation box brings the charge close to neutrality

Fig.S1 shows the total charge change around the reference ionic species Na^+ by the equation $Charge = \int_{r_{min}}^{r} 4\pi r^2 (z_{Na^+} e_0 \rho_{Na^+} g_{++} + z_{Cl^-} e_0 \rho_{Cl^-} g_{-+}) dr$ in NaCl at 0.1 M with box size 50 Å. It clearly shows an increase in charge beyond half of the box length, which can attribute to the periodic images.

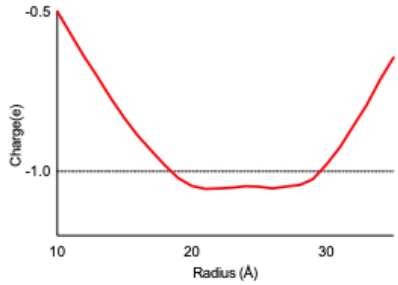


Figure S1 Total charge change in spheres about an Na^+ with radius 10 Å to 30 Å in NaCl at 0.1 M system.

In[162]:=

```
chargen[r_] := NIntegrate[-4 π x^2
    (zCl e0 ρCl grClClavgint[x] + zNa e0 ρNa grNaClavgint[x]), {x, 0.2*10^-9, r}];
chargesn2 = Table[{r, chargen[r] / e0}, {r, 10^-9, 10*10^-9, 0.1*10^-9}];
```

NIntegrate : Numerical integration converging too slowly; suspect one of the following: singularity, value of the integration is 0, highly oscillatory integrand , or WorkingPrecision too small.

NIntegrate : NIntegrate failed to converge to prescribed accuracy after 9 recursive bisections in x near {x} = $\{1.89801 \times 10^{-9}\}$. NIntegrate obtained -1.50246×10^{-19} and 2.1623699579175755` *^-23 for the integral and error estimates.

NIntegrate : NIntegrate failed to converge to prescribed accuracy after 9 recursive bisections in x near {x} = $\{1.67809 \times 10^{-9}\}$. NIntegrate obtained -1.55455×10^{-19} and 2.913989011885777` *^-23 for the integral and error estimates.

NIntegrate : Numerical integration converging too slowly; suspect one of the following: singularity, value of the integration is 0, highly oscillatory integrand , or WorkingPrecision too small.

NIntegrate : NIntegrate failed to converge to prescribed accuracy after 9 recursive bisections in x near {x} = $\{1.63297 \times 10^{-9}\}$. NIntegrate obtained -1.59891×10^{-19} and 3.9073083525040544` *^-23 for the integral and error estimates.

General : Further output of NIntegrate::ncvb will be suppressed during this calculation.

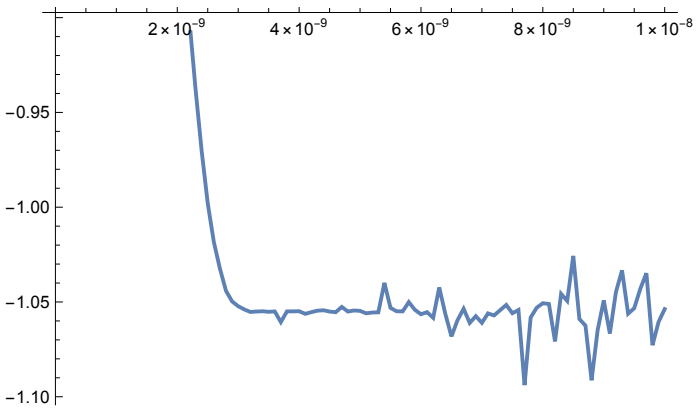
NIntegrate : Numerical integration converging too slowly; suspect one of the following: singularity, value of the integration is 0, highly oscillatory integrand , or WorkingPrecision too small.

General : Further output of NIntegrate::slwcon will be suppressed during this calculation.

In[164]:=

```
ListPlot[chargesn2, Joined → True]
```

Out[164]=



In[33]:=

```
(*Guess with half of the box size*)
rMax = 3.5*10^-9;
```

Step 3: Obtain the Mean Electrostatic Potential at 0.15M

$$\frac{1}{r^2}\frac{\mathrm{d}}{\mathrm{d}\,r}\left(r^2\frac{\mathrm{d}\,\varphi_j(r)}{\mathrm{d}\,r}\right) = -\frac{1}{\varepsilon_0\,\varepsilon_r}\sum_i z_i\,e_0\,\rho_i\,g_{ij}(r)$$

For NaCl :

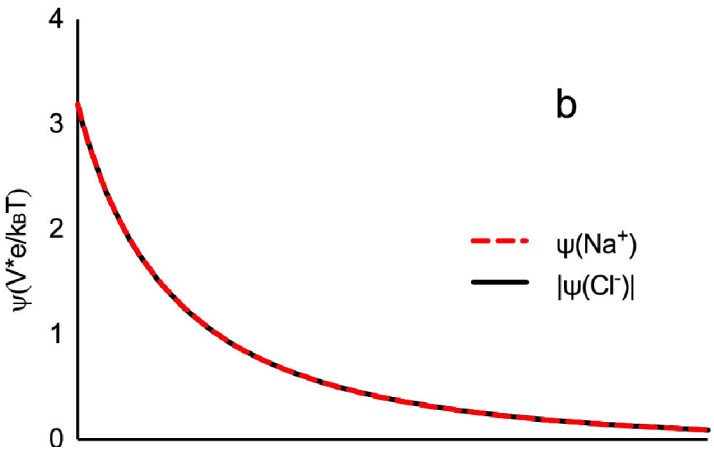
$$\frac{1}{r^2}\frac{d}{dr}\left(r^2\frac{d\,\varphi_{Na^+}(r)}{dr}\right) = -\frac{e_0}{\varepsilon_0\,\varepsilon_r}\left(z_{Na^+}\,\rho_{Na^+}\,g_{++}(r) + z_{Cl^-}\,\rho_{Cl^-}\,g_{-+}(r)\right)$$

Boundary conditions:

$$\varphi_j(r_{max}) = 0$$

$$\varphi_j'(r_{min}) = -\frac{z_j\,e_0}{4\,\pi\,\varepsilon_0\,\varepsilon_r}\,\frac{1}{r_{min}^2}$$

Reference:



In[34]:=
```
rhsna := -e0 / (epsilon0 * epsilonr) *
    (zNa ρNa * grNaNaavgint[#] + zCl ρCl * grNaClavgint[#]) &
rhscl := -e0 / (epsilon0 * epsilonr) *
    (zCl ρCl * grClClavgint[#] + zNa ρNa * grNaClavgint[#]) &
```

In[36]:=
```
(*Mean electrostatic potential of Na*)
ψNa = NDSolve[
  {
   1 / x^2 * D[x^2 D[y[x], x], x] == rhsna[x],
   y[rMax] == 0,
   y'[rMin] == -zNa e0 / (4 π epsilonr epsilon0 rMin^2)
  },
  y,
  {x, rMin}, StartingStepSize → 0.001 * 10^-9,
  Method → {"FixedStep", Method → "ExplicitEuler"}]
```

Out[36]=

{{y → InterpolatingFunction[Domain: {{2. ×10^-10, 3.5 ×10^-9}} Output : scalar]}}

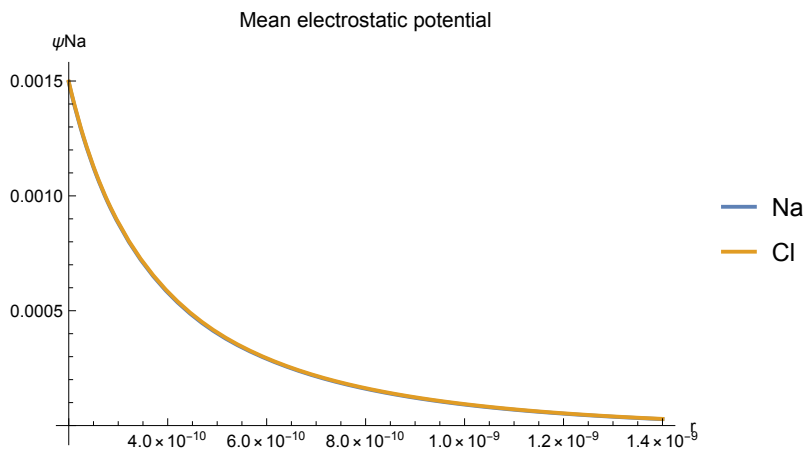
In[37]:=
```
(*Mean electrostatic potential of Cl*)
ψCl = NDSolve[
  {
   1 / x^2 * D[x^2 D[y[x], x], x] == rhscl[x],
   y[rMax] == 0,
   y'[rMin] == -zCl e0 / (4 π epsilonr epsilon0 rMin^2)
  },
  y,
  {x, rMin}, StartingStepSize → 0.001 * 10^-9,
  Method → {"FixedStep", Method → "ExplicitEuler"}]
```

Out[37]=

{{y → InterpolatingFunction[Domain: {{2. ×10^-10, 3.5 ×10^-9}} Output : scalar]}}

In[41]:=
```
(*Plot the mean potentials*)
ψNaSol := y[#] /. Flatten[ψNa] &
ψClSol := y[#] /. Flatten[ψCl] &
Plot[
 {ψNaSol[r] / (β e0), -ψClSol[r] / (β e0)},
 {r, 0.2 * 10^-9, 1.4 * 10^-9},
 PlotRange → All, PlotLabel → "Mean electrostatic potential",
 AxesLabel → {"r", "ψNa"}, PlotLegends → {"Na", "Cl"}]
```

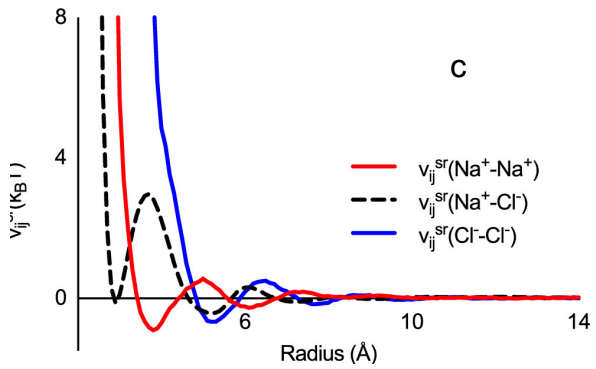
Out[43]=



Step 4: Obtain the Short Range PMF

$g_{ij}(r) = e^{-\beta(q_i \varphi_j(r) + v_{ij}^{sr}(r))}$

Reference:



In[44]:=
```
(*Solve ion-ion short range PMF*)
vSrNaCl := -Log[grNaClavgint[#]] / β - zCl e0 ψNaSol[#] &
vSrNaNa := -Log[grNaNaavgint[#]] / β - zNa e0 ψNaSol[#] &
vSrClCl := -Log[grClClavgint[#]] / β - zCl e0 ψClSol[#] &
```

In[74]:=
```
(*Obtain average value within 11 and 13 Å*)
vSrNaNamean = Mean[
   Table[vSrNaNa[x], {x, 1.1*10^-9, 1.3*10^-9, 0.002*10^-9}]
  ];
(*Normalization ensures the short-
 range PMFs asymptotically approach zero beyond 8 Å*)
vSrNaNacorr := (
   vSrNaNa[#] - vSrNaNamean
  ) &
(*Replace intermediate values*)
vSrNaNacorr2 = ReplaceNaN[
   Table[vSrNaNacorr[x], {x, 0.*10^-9, 9.998*10^-9, 0.002*10^-9}],
   Re[0]];
(*Interpolate to obtain Na-Na short range RDF*)
vSrNaNanan2 =
 Interpolation[Transpose[{rNaNa, vSrNaNacorr2}], InterpolationOrder → 5]
```

InterpolatingFunction : Input value {0.} lies outside the range of data in the interpolating function. Extrapolation will be used.

InterpolatingFunction : Input value $\{2.\times10^{-12}\}$ lies outside the range of data in the interpolating function. Extrapolation will be used.

InterpolatingFunction : Input value $\{4.\times10^{-12}\}$ lies outside the range of data in the interpolating function. Extrapolation will be used.

General : Further output of InterpolatingFunction::dmval will be suppressed during this calculation.

Out[77]=

InterpolatingFunction[Domain: $\{\{0., 1.\times10^{-8}\}\}$ Output: scalar]

In[69]:=
```
(*Obtain Na-Cl short range PMF*)
vSrNaClnan = Interpolation[
   Transpose[
    {rNaCl,
     Table[vSrNaCl[x], {x, 0., 9.998*10^-9, 0.002*10^-9}]}
   ], InterpolationOrder → ord
  ];
vSrNaClmean = Mean[
   Table[vSrNaClnan[x], {x, 1.1*10^-9, 1.3*10^-9, 0.002*10^-9}]
  ];
vSrNaClcorr := (
   vSrNaClnan[#] - vSrNaClmean
  ) &
vSrNaClcorr2 = ReplaceNaN[
   Table[vSrNaClcorr[x], {x, 0., 9.998*10^-9, 0.002*10^-9}],
   0];
vSrNaClnan2 =
 Interpolation[Transpose[{rNaCl, vSrNaClcorr2}], InterpolationOrder → 5]
```

InterpolatingFunction : Input value {0.} lies outside the range of data in the interpolating function. Extrapolation will be used.

InterpolatingFunction : Input value $\{2.\times10^{-12}\}$ lies outside the range of data in the interpolating function. Extrapolation will be used.

InterpolatingFunction : Input value $\{4.\times10^{-12}\}$ lies outside the range of data in the interpolating function. Extrapolation will be used.

General : Further output of InterpolatingFunction::dmval will be suppressed during this calculation.

Out[73]=

InterpolatingFunction[Domain: $\{\{0., 1.\times10^{-8}\}\}$ Output: scalar]

In[78]:=
```
(*Obtain Cl-Cl short range PMF*)
vSrClClnan = Interpolation[
   Transpose[
    {rClCl,
     Table[vSrClCl[x], {x, 0., 9.998*10^-9, 0.002*10^-9}]}
   ], InterpolationOrder → ord
  ];
vSrClClmean = Mean[
   Table[vSrClClnan[x], {x, 1.1*10^-9, 1.3*10^-9, 0.002*10^-9}]
  ];
vSrClClcorr := (
   vSrClClnan[#] - vSrClClmean
  ) &
vSrClClcorr2 = ReplaceNaN[
   Table[vSrClClcorr[x], {x, 0., 9.998*10^-9, 0.002*10^-9}],
   0];
vSrClClnan2 =
 Interpolation[Transpose[{rClCl, vSrClClcorr2}], InterpolationOrder → 5]
```

InterpolatingFunction : Input value {0.} lies outside the range of data in the interpolating function. Extrapolation will be used.

InterpolatingFunction : Input value $\{2.\times10^{-12}\}$ lies outside the range of data in the interpolating function. Extrapolation will be used.

InterpolatingFunction : Input value $\{4.\times10^{-12}\}$ lies outside the range of data in the interpolating function. Extrapolation will be used.

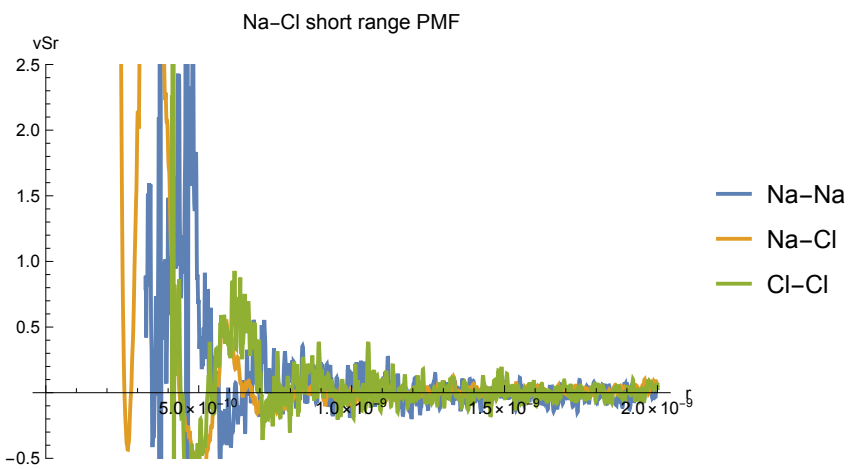
General : Further output of InterpolatingFunction::dmval will be suppressed during this calculation.

Out[82]=

InterpolatingFunction[Domain: $\{\{0., 1.\times10^{-8}\}\}$ Output: scalar]

In[83]:=
```
Plot[
 {vSrNaNanan2[x]*β, vSrNaClnan2[x]*β, vSrClClnan2[x]*β}, {x, 0., 2×10^-9},
 PlotRange → {-0.5, 2.5}, PlotLabel → "Na-Cl short range PMF",
 AxesLabel → {"r", "vSr"}, PlotLegends → {"Na-Na", "Na-Cl", "Cl-Cl"}]
```

Out[83]=



Step 5: Obtain the Mean Electrostatic Potential at Different Concentrations

- Assume short-range PMF is independent of concentration

$$\frac{1}{r^2}\frac{d}{dr}\left(r^2\frac{d\varphi_j(r)}{dr}\right) = -\frac{1}{\varepsilon_0\,\varepsilon_r}\sum_i z_i\, e_0\, \rho_i\, e^{-\beta\,(q_i\,\varphi_j(r)+v_{ij}^{sr}(r))}$$

For NaCl :

$$\frac{1}{r^2}\frac{d}{dr}\left(r^2\frac{d\varphi_{Na^+}(r)}{dr}\right) = -\frac{e_0}{\varepsilon_0\,\varepsilon_r}\left(z_{Na^+}\,\rho_{Na^+}\,e^{-\beta\,(q_{Na^+}\,\varphi_{Na^+}(r)+v_{++}^{sr}(r))} + z_{Cl^-}\,\rho_{Cl^-}\,e^{-\beta\,(q_{Cl^-}\,\varphi_{Na^+}(r)+v_{+-}^{sr}(r))}\right)$$

Boundary conditions :

$$\varphi_j(r_{max}) = 0$$

$$\varphi_j'(r_{min}) = -\frac{z_j\,e_0}{4\,\pi\,\varepsilon_0\,\varepsilon_r}\,\frac{1}{r_{min}^2}$$

In[129]:=

```
(*Function to solve the mean potential at different concentrations*)
ψNaSolver := NDSolve[
   {
    1 / x^2 * D[x^2 D[y[x], x], x] ==
     -e0 / (epsilon0 * epsilonr) *
      (
       zNa * #1 * Exp[
           -β (zNa * e0 * y[x] + vSrNaNanan2[x])]
        + zCl * #2 * Exp[
           -β (zCl * e0 * y[x] + vSrNaClnan2[x])]
      ),
    y[2 * 10^-9] == 0,
    y'[rMin] == -zNa e0 / (4 π epsilonr epsilon0 rMin^2)
   },
   y,
   {x, rMin},
   StartingStepSize → 0.01 * 10^-9,
   Method → {"FixedStep", Method → "ExplicitEuler"},
   AccuracyGoal → 50, InterpolationOrder → 1] &

(*Mean potential pred at 0.15M*)
ψN1 = ψNaSolver[ρNa, ρCl];
(*Mean potential prediction at 1M*)
ψN2 = ψNaSolver[1/0.15 ρNa, 1/0.15 ρCl];
```

Step 6: RDF Prediction

$$g_{ij}(r) = e^{-\beta(q_i \varphi_j(r) + v_{ij}^{sr}(r))}$$

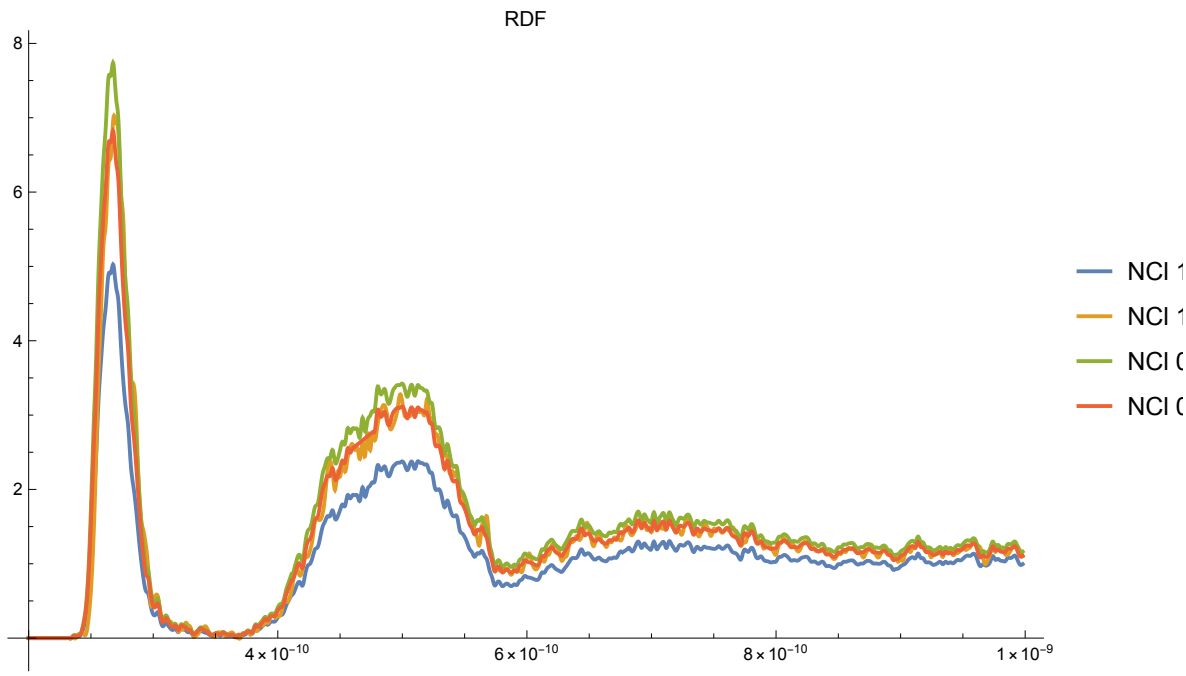
In[102]:=

```
(*Test with RDF data from 1M solution*)
dataNN1M = Import[
   "/Users/heiley/Desktop/mathematica/charge0/rdf_1M-na-na.xvg", "Table"];
rNN1M = dataNN1M[[All, 1]] * 10^-9;
gNN1M = dataNN1M[[All, 2]];
grNN1M = Interpolation[Transpose[{rNN1M, gNN1M}], Method → "Spline"];
grNN1Mavg =
  MovingAverage[Table[grNN1M[x], {x, 0., 10.002 * 10^-9, 0.002 * 10^-9}], 3];
grNN1Mavgint = Interpolation[Transpose[{rNN1M, grNN1Mavg}], Method → "Spline"];

dataNCl1M = Import[
   "/Users/heiley/Desktop/mathematica/charge0/rdf_1M-na-cl.xvg", "Table"];
rNCl1M = dataNCl1M[[All, 1]] * 10^-9;
gNCl1M = dataNCl1M[[All, 2]];
grNCl1M = Interpolation[Transpose[{rNCl1M, gNCl1M}], Method → "Spline"];
grNCl1Mavg =
  MovingAverage[Table[grNCl1M[x], {x, 0., 10.002 * 10^-9, 0.002 * 10^-9}], 3];
grNCl1Mavgint =
  Interpolation[Transpose[{rNCl1M, grNCl1Mavg}], Method → "Spline"];

dataClCl1M = Import[
   "/Users/heiley/Desktop/mathematica/charge0/rdf_nacl-cl-1M.xvg", "Table"];
rClCl1M = dataClCl1M[[All, 1]] * 10^-9;
gClCl1M = dataNN1M[[All, 2]];
grClCl1M = Interpolation[Transpose[{rClCl1M, gClCl1M}], Method → "Spline"];
grClCl1Mavg =
  MovingAverage[Table[grClCl1M[x], {x, 0., 10.002 * 10^-9, 0.002 * 10^-9}], 3];
grClCl1Mavgint =
  Interpolation[Transpose[{rClCl1M, grClCl1Mavg}], Method → "Spline"];
```

In[132]:=

```
(*Na-Cl RDF prediction of 0.15M*)
gNaClSolver015 := Exp[
    -β (
        zCl e0 y[#] + vSrNaCl[#]
      ) /. ψN1[[1]]
  ] &
(*Na-Cl RDF prediction of 1M*)
gNaClSolver1 := Exp[
    -β (
        zCl e0 y[#] + vSrNaCl[#]
      ) /. ψN2[[1]]
  ] &
Plot[{gNaClSolver1[x], grNCl1Mavgint[x], gNaClSolver015[x], grNaClavgint[x]},
 {x, 0.2 * 10^-9, 0.998 * 10^-9}, PlotRange → Full, PlotLabel → "RDF", PlotLegends →
  {"NCl 1M MPBE", "NCl 1M CMD", "NCl 0.15M MPBE", "NCl 0.15M CMD"}]
```

Out[134]=



Step 7: Osmotic Coefficient Prediction

$\Phi_v(\rho) = 1 - \frac{2\pi\beta}{3\rho} \sum_{i,j} \rho_i \rho_j \int_0^\infty g_{ij}(r,\rho) \frac{d v_{ij}^e(r)}{dr} r^3 \, dr$

$v_{ij}(r) = q_i \varphi_j(r) + v_{ij}^{sr}(r)$

- Assume effective pair potential at infinite dilution can be approximated by PMF at 0.1 M

Vapor Pressure Measurements; Source: https://doi.org/10.1063/1.3253108

1M: 0.936

0.1M: 0.933

In[143]:=

```
(*Osmotic coefficient prediction using 1M simulation RDF*)
phi := 1 - 2 π β/(3 #) (
    ρNa ρNa NIntegrate[
       grNN1Mavgint[r] r^3
        D[zNa * e0 * ψNaSol[r] + vSrNaNanan2[r], r], {r, 0, 2 * 10^-9}] +
     ρNa ρCl NIntegrate[
       grNCl1Mavgint[r] r^3
        D[zNa * e0 * ψNaSol[r] + vSrNaClnan2[r], r], {r, 0, 2 * 10^-9}] +
     ρNa ρCl NIntegrate[
       grNCl1Mavgint[r] r^3
        D[zCl * e0 * ψClSol[r] + vSrNaClnan2[r], r], {r, 0, 2 * 10^-9}] +
     ρCl ρCl NIntegrate[
       grClCl1Mavgint[r] r^3
        D[zCl * e0 * ψClSol[r] + vSrClClnan2[r], r], {r, 0, 2 * 10^-9}]
   ) &
phi[(ρNa + ρCl) * 1/0.15]
```

NIntegrate : Numerical integration converging too slowly; suspect one of the following: singularity, value of the integration is 0, highly oscillatory integrand, or WorkingPrecision too small.

NIntegrate : NIntegrate failed to converge to prescribed accuracy after 9 recursive bisections in r near {r} = $\{1.49606 \times 10^{-9}\}$. NIntegrate obtained $1.78341 \times 10^{-49} - 3.77468 \times 10^{-52}\, i$ and 7.573045560989907` *^-48 for the integral and error estimates.

NIntegrate : Numerical integration converging too slowly; suspect one of the following: singularity, value of the integration is 0, highly oscillatory integrand, or WorkingPrecision too small.

NIntegrate : NIntegrate failed to converge to prescribed accuracy after 9 recursive bisections in r near {r} = $\{1.82809 \times 10^{-9}\}$. NIntegrate obtained $-3.07466 \times 10^{-48} - 7.90865 \times 10^{-56}\, i$ and 4.091420175718245` *^-48 for the integral and error estimates.

NIntegrate : Numerical integration converging too slowly; suspect one of the following: singularity, value of the integration is 0, highly oscillatory integrand , or WorkingPrecision too small.

General : Further output of NIntegrate::slwcon will be suppressed during this calculation.

NIntegrate : NIntegrate failed to converge to prescribed accuracy after 9 recursive bisections in r near {r} = $\{1.82809 \times 10^{-9}\}$. NIntegrate obtained $-3.11387 \times 10^{-48} - 7.90865 \times 10^{-56}\, i$ and 4.0913651041902856` *^-48 for the integral and error estimates.

General : Further output of NIntegrate::ncvb will be suppressed during this calculation.

Out[144]=

$1.03213 - 0.000632656\, i$

In[155]:=

```
(*Osmotic coefficient prediction using 0.15M simulation RDF*)
phi := 1 - 2 π β/(3 #) (
    ρNa ρNa NIntegrate[
       grNaNaavgint[r] r^3
        D[zNa * e0 * ψNaSol[r] + vSrNaNanan2[r], r], {r, 0, 2 * 10^-9}] +
     ρNa ρCl NIntegrate[
       grNaClavgint[r] r^3
        D[zNa * e0 * ψNaSol[r] + vSrNaClnan2[r], r], {r, 0, 2 * 10^-9}] +
     ρNa ρCl NIntegrate[
       grNaClavgint[r] r^3
        D[zCl * e0 * ψClSol[r] + vSrNaClnan2[r], r], {r, 0, 2 * 10^-9}] +
     ρCl ρCl NIntegrate[
       grClClavgint[r] r^3
        D[zCl * e0 * ψClSol[r] + vSrClClnan2[r], r], {r, 0, 2 * 10^-9}]
   ) &
phi[ρNa + ρCl]
```

NIntegrate : Numerical integration converging too slowly; suspect one of the following: singularity, value of the integration is 0, highly oscillatory integrand, or WorkingPrecision too small.

NIntegrate : NIntegrate failed to converge to prescribed accuracy after 9 recursive bisections in r near {r} = $\{1.49606 \times 10^{-9}\}$. NIntegrate obtained $1.78341 \times 10^{-49} - 3.77468 \times 10^{-52} i$ and 7.573045560989907` *^-48 for the integral and error estimates.

NIntegrate : Numerical integration converging too slowly; suspect one of the following: singularity, value of the integration is 0, highly oscillatory integrand , or WorkingPrecision too small.

NIntegrate : NIntegrate failed to converge to prescribed accuracy after 9 recursive bisections in r near {r} = $\{1.82809 \times 10^{-9}\}$. NIntegrate obtained $-1.53521 \times 10^{-48} + 4.65944 \times 10^{-52} i$ and 4.235623790825623` *^-48 for the integral and error estimates.

NIntegrate : Numerical integration converging too slowly; suspect one of the following: singularity, value of the integration is 0, highly oscillatory integrand , or WorkingPrecision too small.

General : Further output of NIntegrate::slwcon will be suppressed during this calculation.

NIntegrate : NIntegrate failed to converge to prescribed accuracy after 9 recursive bisections in r near {r} = $\{1.82809 \times 10^{-9}\}$. NIntegrate obtained $-1.57434 \times 10^{-48} + 4.65944 \times 10^{-52} i$ and 4.235602024392781` *^-48 for the integral and error estimates.

General : Further output of NIntegrate::ncvb will be suppressed during this calculation.

Out[156]=

0.934111 + 0.0000865636 i