Simulation of Ideal Gas

Simulation Assumptions

The chosen gas for simulation is Argon

Small particles in constant random linear motion
No interactive forces
Average kinetic energy depends on temperature
Maxwell-Boltzmann distribution for total velocities, but each vector component follows a normal curve

For the main simulation and verification of theoretical laws/behaviors, particle-particle collisions are ignored
In the last section, elastic collisions of 2 particles at a time are accounted for and animated

Project Objectives

Animate motion of ideal gas particles in a 3d box, under the previously mentioned assumptions

Verify Ideal Gas Law
Verify Boyle's Law
Verify Gay-Lussac's Law

Simulate isothermal compression and verify behavior by comparing to theoretical integral

After accomplishing all of the above under the assumption of no particle collisions, add code that animates motion of the gas that now also includes elastic collisions of 2 particles at a time

Initial Simulation

```
In[1]:= Sim[nParticles_, xParticles_, m_, length_] := {
    Clear[xPos, trajData];

    trajData = {};
    vTotalArray = {};
    F = 0;
    dpTotal = 0;
    nParticlesUsed = nParticles;
    xParticlesUsed = xParticles[[1 ;; nParticlesUsed]];

    For[i = 1, i ≤ nSteps, i++,
     v = {vTherm, vTherm, vTherm};
     vTotal = Sqrt[v[[1]]^2 + v[[2]]^2 + v[[3]]^2];
     AppendTo[vTotalArray, vTotal];

     For[n = 1, n ≤ nParticlesUsed, n++,
      xPos = xParticlesUsed[[n]][[i]];
      xPos += v dt;

      If[xPos[[1]] < 0, xPos[[1]] = -xPos[[1]];
       dp = Abs[2 m v[[1]]];
       dpTotal += dp;
       v[[1]] = -v[[1]];];
      If[xPos[[1]] > length, xPos[[1]] = 2 * length - xPos[[1]];
       dp = Abs[2 m v[[1]]];
       dpTotal += dp;
       v[[1]] = -v[[1]];];
      If[xPos[[2]] < 0, xPos[[2]] = -xPos[[2]];
       dp = Abs[2 m v[[2]]];
       dpTotal += dp;
       v[[2]] = -v[[2]];];
      If[xPos[[2]] > length, xPos[[2]] = 2 * length - xPos[[2]];
       dp = Abs[2 m v[[2]]];
       dpTotal += dp;
       v[[2]] = -v[[2]];];
      If[xPos[[3]] < 0, xPos[[3]] = -xPos[[3]];
       dp = Abs[2 m v[[3]]];
       dpTotal += dp;
       v[[3]] = -v[[3]];];
      If[xPos[[3]] > length, xPos[[3]] = 2 * length - xPos[[3]];
       dp = Abs[2 m v[[3]]];
       dpTotal += dp;
       v[[3]] = -v[[3]];];
```

```
      AppendTo[xParticlesUsed[[n]], xPos];
     ];

    ];

   F += dpTotal / time;
  }
```

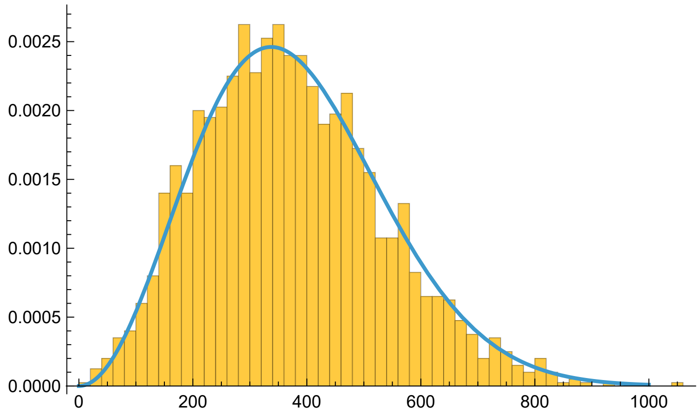
In[2]:=
```
RT1 = 8.314 * 273; (*units J/mol = (kg m2/s2)/mol*)
M = 39.9 / 1000; (*molar mass: units kg*)
vTherm := Random[NormalDistribution[0, Sqrt[(RT1) / (M)]]] (*units m/s*)
m = 6.6335209 * (10^(-26)); (*atomic mass of Argon in kg*)

length = 100;
nParticles = 400;
time = 100; (*units sec*)
dt = .05;
nSteps = Round[time / dt];

xParticles = {};
For[i = 1, i ≤ nParticles, i++,
 AppendTo[xParticles,
   {{RandomReal[{0, length}], RandomReal[{0, length}], RandomReal[{0, length}]}}];
]

Sim[nParticles, xParticles, m, length];

area = 6 * (length^2);
V1 = length^3;
P1 = F / area;
n = nParticlesUsed / (6.02 * (10^23));
Z = (P1 V1) / (n RT1);

Print["total force: ", F, " N"]
Print["total pressure: ", P1, " Pa"]
Print["compressibility: ", Z]

σ = ((RT1) / M)^0.5;
MaxwellCurve = Plot[PDF[MaxwellDistribution[σ], x],
   {x, 0, 1000}, PlotRange → {{0, 1000}, {0, .0035}}, PlotStyle → Thick];
vTotalDist = Histogram[vTotalArray, bins = 50, "PDF"];
Show[vTotalDist, MaxwellCurve]
```

total force: 8.66111×10^{-20} N

total pressure: 1.44352×10^{-24} Pa

compressibility: 0.957163

Out[25]=



In[26]:=
```
RT1 = 8.314 * 273; (*units J/mol = (kg m2/s2)/mol*)
M = 39.9 / 1000; (*molar mass: units kg*)
vTherm := Random[NormalDistribution[0, Sqrt[(RT1) / (M)]]] (*units m/s*)
m = 6.6335209 * (10^(-26)); (*atomic mass of Argon in kg*)

nParticles = 400;
length = 1000;
time = 100; (*units sec*)
dt = .15;
nSteps = Round[time / dt];

Sim[nParticles, xParticles, m, length];

animationArray = {};
For[i = 1, i ≤ nSteps + 1, i++,
 positionAllParticles = {};
 For[n = 1, n ≤ nParticlesUsed, n++,
  position = xParticlesUsed[[n]][[i]];
  AppendTo[positionAllParticles, position];
 ];
 AppendTo[animationArray, positionAllParticles]
]
```

In[38]:=
```
v = Map[Point, animationArray];
animation =
  Animate[Graphics3D[{White, v[[i]]}, PlotRange → {{0, length}, {0, length}, {0, length}}],
   {i, 1, Length[v], 1}, AnimationRate → 100, RefreshRate → 100, DisplayAllSteps → True];
```

In[40]:=
```
Export["C:\\Users\\prana\\OneDrive - The University of Texas at
   Austin\\Desktop\\Spring 2022\\comp chem\\gasanimation.avi", animation]
```

Out[40]=
```
C:\Users\prana\OneDrive - The University of
  Texas at Austin\Desktop\Spring 2022\comp chem\gasanimation.avi
```

Average Z Factor

In[41]:=
```
ZSim[N1_, N2_, Nstep_, xParticles_, m_, length_, nRuns_] := {
  ZarrayDiffNs = {};
  For[o = N1, o ≤ N2, o += Nstep,
   Zarray = {};
   For[k = 1, k ≤ nRuns, k++,
    Clear[xPos, trajData];

    trajData = {};
    vTotalArray = {};
    F = 0;
    dpTotal = 0;
    nParticlesUsed = o;
    xParticlesUsed = xParticles[[1 ;; nParticlesUsed]];

    For[i = 1, i ≤ nSteps, i++,
     v = {vTherm, vTherm, vTherm};
     vTotal = Sqrt[v[[1]]^2 + v[[2]]^2 + v[[3]]^2];
     AppendTo[vTotalArray, vTotal];

     For[n = 1, n ≤ nParticlesUsed, n++,
      xPos = xParticlesUsed[[n]][[i]];
      xPos += v dt;

      If[xPos[[1]] < 0, xPos[[1]] = -xPos[[1]];
       dp = Abs[2 m v[[1]]];
       dpTotal += dp;
       v[[1]] = -v[[1]];];
      If[xPos[[1]] > length, xPos[[1]] = 2 * length - xPos[[1]];
       dp = Abs[2 m v[[1]]];
       dpTotal += dp;
       v[[1]] = -v[[1]];];
      If[xPos[[2]] < 0, xPos[[2]] = -xPos[[2]];
       dp = Abs[2 m v[[2]]];
       dpTotal += dp;
       v[[2]] = -v[[2]];];
      If[xPos[[2]] > length, xPos[[2]] = 2 * length - xPos[[2]];
       dp = Abs[2 m v[[2]]];
       dpTotal += dp;
       v[[2]] = -v[[2]];];
      If[xPos[[3]] < 0, xPos[[3]] = -xPos[[3]];
       dp = Abs[2 m v[[3]]];
       dpTotal += dp;
       v[[3]] = -v[[3]];];
      If[xPos[[3]] > length, xPos[[3]] = 2 * length - xPos[[3]];
```

```
      dp = Abs[2 m v[[3]]];
      dpTotal += dp;
      v[[3]] = -v[[3]];];

     AppendTo[xParticlesUsed[[n]], xPos];
    ];

   ];

   F += dpTotal / time;

   area = 6 * (length^2);
   V1 = length^3;
   P1 = F / area;
   n = nParticlesUsed / (6.02 * (10^23));
   Z = (P1 V1) / (n RT1);

   AppendTo[Zarray, Z];
  ];
  AppendTo[ZarrayDiffNs, {o, Zarray}]
 ];
 ZmeanArrayDiffNs = {};
 For[l = 1, l ≤ Length[ZarrayDiffNs], l++,
  AppendTo[ZmeanArrayDiffNs, {ZarrayDiffNs[[l]][[1]], Mean[ZarrayDiffNs[[l]][[2]]]}]
 ]
}
```

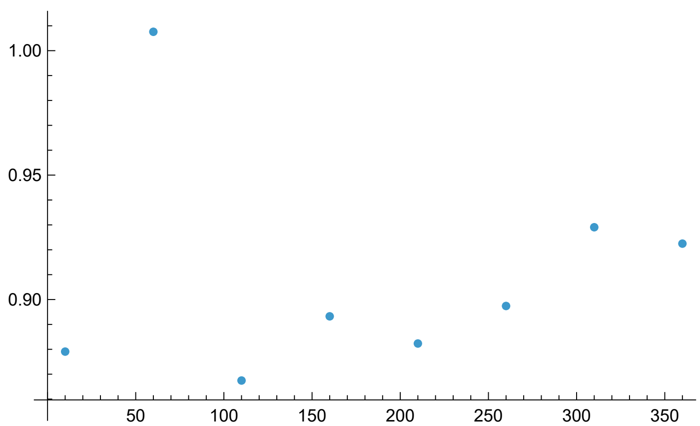
```
In[42]:= RT1 = 8.314 * 273; (*units J/mol = (kg m2/s2)/mol*)
M = 39.9 / 1000; (*molar mass: units kg*)
vTherm := Random[NormalDistribution[0, Sqrt[(RT1) / (M)]]]; (*units m/s*)
m = 6.6335209 * (10^(-26)); (*atomic mass of Argon in kg*)

length = 1000;
time = 100; (*units sec*)
dt = .05;
nSteps = Round[time / dt];

xParticles = {};
For[i = 1, i ≤ nParticles, i++,
  AppendTo[xParticles,
    {{RandomReal[{0, length}], RandomReal[{0, length}], RandomReal[{0, length}]}}];
 ];

N1 = 10;
N2 = 400;
Nstep = 50;
nRuns = 10;
ZSim[N1, N2, Nstep, xParticles, m, length, nRuns];
```

In[57]:=
```
ZarrayDiffNs
ZmeanArrayDiffNs
ListPlot[ZmeanArrayDiffNs]
```

Out[57]=
```
{{10, {0.826919, 0.68056, 0.914408, 0.856891,
   1.10258, 0.836947, 0.784385, 1.03016, 1.03799, 0.719967}},
 {60, {0.852499, 0.877119, 0.859559, 1.02112, 0.962566, 1.18605, 1.01924,
   1.00498, 1.26938, 1.02339}}, {110, {0.846388, 0.84335, 0.784601,
   0.890101, 1.01987, 0.940103, 0.665862, 0.93622, 0.914167, 0.833916}},
 {160, {0.974041, 0.861698, 0.901127, 0.916381, 0.926653,
   0.825778, 0.894629, 0.937129, 0.927642, 0.767439}},
 {210, {0.92614, 0.985782, 0.865546, 0.81724, 0.836411, 0.822901, 0.912883,
   0.922932, 0.890011, 0.843473}}, {260, {0.862718, 0.919657, 0.878689,
   0.916693, 0.913131, 0.962413, 0.897136, 0.863647, 0.852736, 0.907292}},
 {310, {0.883661, 0.914544, 0.848343, 1.03425, 0.922021, 0.908858, 0.88446,
   1.08371, 0.87783, 0.932823}}, {360, {0.970734, 0.904481, 0.937337,
   0.913116, 0.943568, 0.842158, 1.03091, 0.916303, 0.843177, 0.922905}}}
```

Out[58]=
```
{{10, 0.87908}, {60, 1.00759}, {110, 0.867458}, {160, 0.893252},
 {210, 0.882332}, {260, 0.897411}, {310, 0.92905}, {360, 0.922469}}
```

Out[59]=



Simulation with Different Volume

In[60]:=
```
DiffVolume[originalLength_, multiplier_, nRuns_] := {
  pressuresRatioVolume = {};
  lengthMultiplier = multiplier^(1/3);
  length = lengthMultiplier * originalLength;

  For[k = 1, k ≤ nRuns, k++,
   RT1 = 8.314 * 273; (*units J/mol = (kg m2/s2)/mol*)
   M = 39.9/1000; (*molar mass: units kg*)
   vTherm := Random[NormalDistribution[0, Sqrt[(RT1) / (M)]]]; (*units m/s*)
   m = 6.6335209 * (10^(-26)); (*atomic mass of Argon in kg*)

   nParticles = 400;
   time = 100; (*units sec*)
   dt = .05;
   nSteps = Round[time/dt];

   xParticles = {};
   For[i = 1, i ≤ nParticles, i++,
    AppendTo[xParticles,
      {{RandomReal[{0, length}], RandomReal[{0, length}], RandomReal[{0, length}]}}];
   ];

   Sim[nParticles, xParticles, m, length];

   area = 6 * (length^2);
   V2 = length^3;
   P2 = F/area;
   n = nParticlesUsed/(6.02 * (10^23));
   Z = (P2 V2) / (n RT1);

   pressuresRatioVolume = AppendTo[pressuresRatioVolume, P1/P2];

   σ = ((RT1) / M)^0.5;
   MaxwellCurve = Plot[PDF[MaxwellDistribution[σ], x],
     {x, 0, 1000}, PlotRange → {{0, 1000}, {0, .0035}}];
   vTotalDist = Histogram[vTotalArray, bins = 50, "PDF"];
  ];

 }
```

In[61]:=
```
DiffVolume[100, 8, 1];

Print["total force: ", F]
Print["total pressure: ", P2]
Print["compressibility: ", Z]
Print["P1/P2 = ", P1 / P2]
Print["V2/V1 = ", V2 / V1]

Show[MaxwellCurve, vTotalDist]
```

total force: 4.40305×10^{-20}

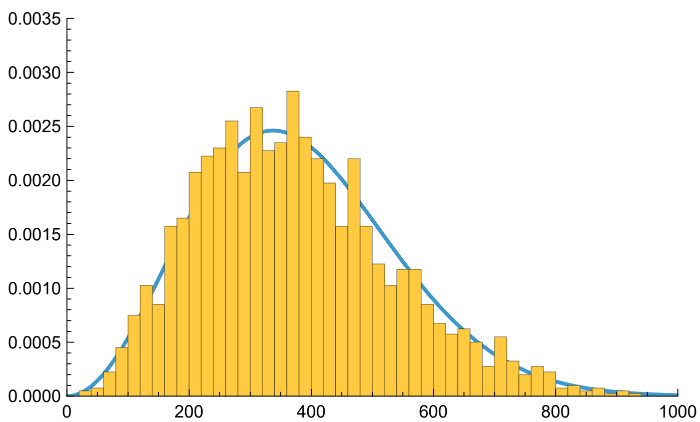
total pressure: 1.8346×10^{-25}

compressibility: 0.973186

P1/P2 = 0.006828

V2/V1 = $\frac{1}{125}$

Out[67]=



Below is the result if the simulation is run 10 times and the average pressure ratio is taken.

In[68]:=
```
multiplierVolume = 8;
nRuns = 10;
```

In[70]:=
```
DiffVolume[100, multiplierVolume, nRuns];
```

In[71]:=
```
Print[pressuresRatioVolume]
```

{0.00655236, 0.00661638, 0.00671062, 0.00655254,
 0.00651783, 0.00674524, 0.00677331, 0.00668591, 0.0063483, 0.00664297}

In[72]:=
```
averagePressureRatioVolume = Mean[pressuresRatioVolume]
Print["variation from ideal: ",
 Abs[((averagePressureRatioVolume - multiplierVolume) / multiplierVolume ) * 100], "%"]
```

Out[72]=
```
0.00661455
```

```
variation from ideal: 99.9173%
```

Simulation with Different Temperature

In[74]:=
```
DiffTemp[originalTemp_, multiplier_, nRuns_] := {
  pressuresRatioTemp = {};
  RT2 = multiplier * 8.314 * originalTemp; (*units J/mol = (kg m2/s2)/mol*)

  For[k = 1, k ≤ nRuns, k++,
   M = 39.9 / 1000; (*molar mass: units kg*)
   vTherm := Random[NormalDistribution[0, Sqrt[(RT2) / (M)]]]; (*units m/s*)
   m = 6.6335209 * (10^(-26)); (*atomic mass of Argon in kg*)

   length = 100;
   nParticles = 400;
   time = 100; (*units sec*)
   dt = .05;
   nSteps = Round[time / dt];

   xParticles = {};
   For[i = 1, i ≤ nParticles, i++,
    AppendTo[xParticles,
      {{RandomReal[{0, length}], RandomReal[{0, length}], RandomReal[{0, length}]}}];
   ];

   Sim[nParticles, xParticles, m, length];

   area = 6 * (length^2);
   V1 = length^3;
   P2 = F / area;
   n = nParticlesUsed / (6.02 * (10^23));
   Z = (P2 V1) / (n RT2);

   pressuresRatioTemp = AppendTo[pressuresRatioTemp, P1 / P2];

   σ = ((RT2) / M)^0.5;
   MaxwellCurve = Plot[PDF[MaxwellDistribution[σ], x],
     {x, 0, 1800}, PlotRange → {{0, 1800}, {0, .0035}}];
   vTotalDist = Histogram[vTotalArray, bins = 30, "PDF"];
  ];
 }
```

In[75]:=
```
DiffTemp[273, 3, 1];

Print["total force: ", F]
Print["total pressure: ", P2]
Print["compressibility: ", Z]
Print["P1/P2 = ", P1 / P2]
Print["T1/T2 = ", RT1 / RT2]

Show[MaxwellCurve, vTotalDist]
```

total force: 2.74593×10^{-19}

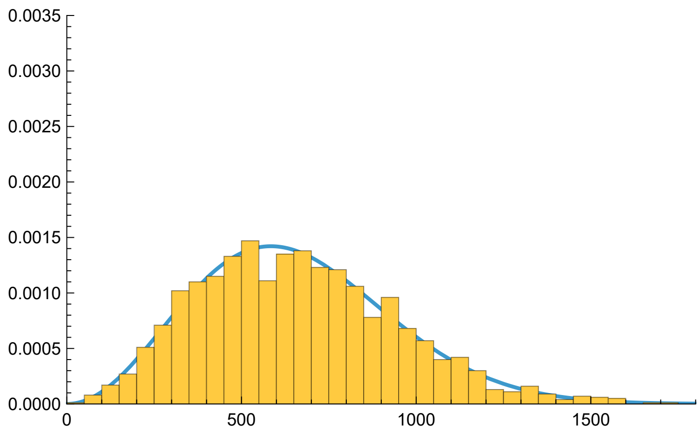
total pressure: 4.57654×10^{-24}

compressibility: 1.01153

P1/P2 = 0.000273715

T1/T2 = 0.333333

Out[81]=



Below is the result if the simulation is run 10 times and the average pressure ratio is taken.

In[82]:=
```
multiplierTemp = 3;
nRuns = 10;
```

In[84]:=
```
DiffTemp[273, multiplierTemp, nRuns];
```

In[85]:=
```
Print[pressuresRatioTemp]
```

{0.000280232, 0.000261232, 0.000281117, 0.000279225, 0.000273218,
0.000277227, 0.000283645, 0.000284484, 0.000278845, 0.000286093}

In[86]:=
```
averagePressureRatioTemp = Mean[pressuresRatioTemp]
Print["variation from ideal: ",
 Abs[((averagePressureRatioTemp - (1 / multiplierTemp)) / (1 / multiplierTemp)) * 100], "%"]
```

Out[86]=

0.000278532

variation from ideal: 99.9164%

Isothermal Compression

In[88]:=
```
IsothermalCE[nParticles_, h1_, h2_, hstep_, m_, length_] := {
  PVarray = {};
  Zarray = {};

  For[h = h1, h ≥ h2, h += hstep,
   Clear[xPos, trajData];
   xParticles = {};

   trajData = {};
   vTotalArray = {};
   F = 0;
   dpTotal = 0;

   For[o = 1, o ≤ nParticles, o++,
    AppendTo[xParticles,
      {{RandomReal[{0, length}], RandomReal[{0, length}], RandomReal[{0, h}]}}];
   ];

   For[i = 1, i ≤ nSteps, i++,
    v = {vTherm, vTherm, vTherm};
    vTotal = Sqrt[v⟦1⟧^2 + v⟦2⟧^2 + v⟦3⟧^2];
    AppendTo[vTotalArray, vTotal];

    For[n = 1, n ≤ nParticles, n++,
     xPos = xParticles⟦n⟧⟦i⟧;
     xPos += v dt;

     If[xPos⟦1⟧ < 0, xPos⟦1⟧ = -xPos⟦1⟧;
      dp = Abs[2 m v⟦1⟧];
      dpTotal += dp;
      v⟦1⟧ = -v⟦1⟧;];
     If[xPos⟦1⟧ > length, xPos⟦1⟧ = 2 * length - xPos⟦1⟧;
      dp = Abs[2 m v⟦1⟧];
      dpTotal += dp;
      v⟦1⟧ = -v⟦1⟧;];
     If[xPos⟦2⟧ < 0, xPos⟦2⟧ = -xPos⟦2⟧;
      dp = Abs[2 m v⟦2⟧];
      dpTotal += dp;
      v⟦2⟧ = -v⟦2⟧;];
     If[xPos⟦2⟧ > length, xPos⟦2⟧ = 2 * length - xPos⟦2⟧;
      dp = Abs[2 m v⟦2⟧];
      dpTotal += dp;
      v⟦2⟧ = -v⟦2⟧;];
     If[xPos⟦3⟧ < 0, xPos⟦3⟧ = -xPos⟦3⟧;
```

```
        dp = Abs[2 m v[[3]]];
        dpTotal += dp;
        v[[3]] = -v[[3]];];
       If[xPos[[3]] > h, xPos[[3]] = 2 * h - xPos[[3]];
        dp = Abs[2 m v[[3]]];
        dpTotal += dp;
        v[[3]] = -v[[3]];];

       AppendTo[xParticles[[n]], xPos];
      ];

     ];

     F += dpTotal / time;

     area = 4 * length * h + 2 * length^2;
     V = h * length^2;
     P = F / area;
     n = nParticles / (6.02 * (10^23));
     Z = (P V) / (n RT);

     AppendTo[PVarray, {V, P}];
     AppendTo[Zarray, Z];
    ]
   }
```

In[89]:=
```
RT = 8.314 * 273; (*units J/mol =  (kg m2/s2)/mol*)
M = 39.9 / 1000; (*molar mass: units kg*)
vTherm := Random[NormalDistribution[0, Sqrt[(RT) / (M)]]]; (*units m/s*)
m = 6.6335209 * (10^(-26)); (*atomic mass of Argon in kg*)

nParticles = 400;
time = 100; (*units sec*)
dt = .05;
nSteps = Round[time / dt];

length = 1000;
h1 = 1000;
h2 = 100;
hstep = -200;
nRuns = 10;
IsothermalCE[nParticles, h1, h2, hstep, m, length];
```

In[103]:=
```
Print[PVarray, Zarray]
```

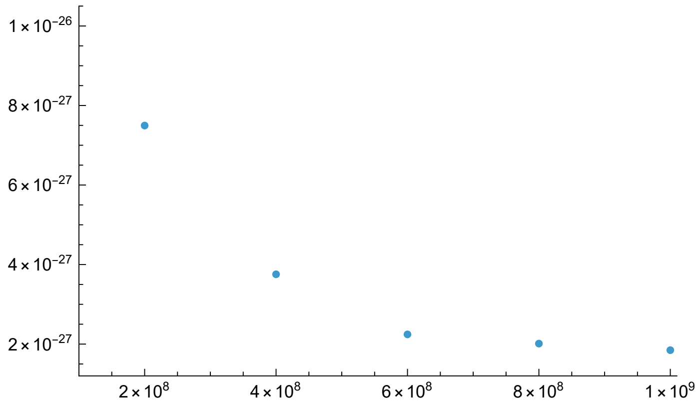
$\{\{1\,000\,000\,000,\ 1.84719\times10^{-27}\},\ \{800\,000\,000,\ 2.01263\times10^{-27}\},\ \{600\,000\,000,\ 2.24408\times10^{-27}\},\ \{400\,000\,000,\ 3.75551\times10^{-27}\},\ \{200\,000\,000,\ 7.49572\times10^{-27}\}\}$

{1.22483, 1.06762, 0.8928, 0.996076, 0.994047}

In[104]:=

```
PVplot =
 ListPlot[PVarray, PlotRange → {{h2 * length^2, 1.01 * h1 * length^2}, {1.2*^-27, 1.05*^-26}}]
```

Out[104]=



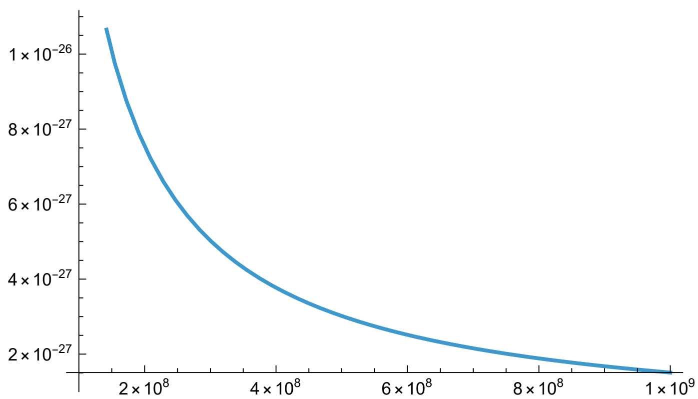
In[105]:=

```
Clear[V, W];
W[V_] = (n RT) / V
w = Integrate[W[V], {V, h2 * length^2, h1 * length^2}];
Plot[W[V], {V, h2 * length^2, h1 * length^2}]
Print["Theoretical work is ", w, " J."]
```

Out[106]=

$$\frac{1.50812 \times 10^{-18}}{V}$$

Out[108]=



Theoretical work is 3.47258×10^{-18} J.

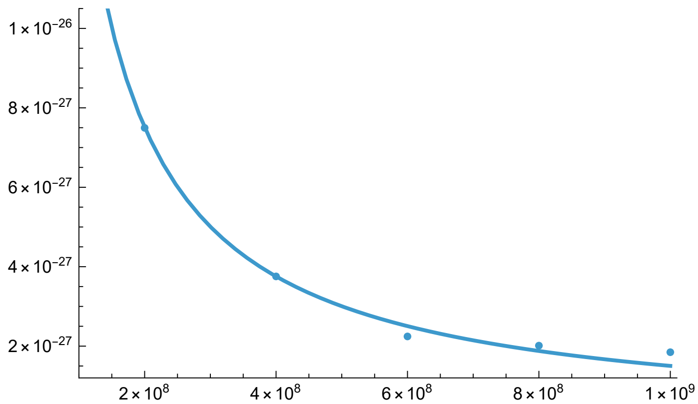
In[110]:=

```
regressionCoefficient = FindFit[PVarray, a / Vfit, a, Vfit];
Wfit[Vfit_] = a / Vfit /. {regressionCoefficient[[1]]}
wEstimate = Integrate[Wfit[x], {x, h2 * length^2, h1 * length^2}];
Show[PVplot, Plot[Wfit[Vfit], {Vfit, h2 * length^2, h1 * length^2}]]
Print["Estimated work calculated using the simulation is ", wEstimate, " J."]
Print["Percent error compared to the theoretical work is ",
 100 * Abs[(wEstimate - w) / w], "%"]
```

Out[111]=

$$\frac{1.50232\times 10^{-18}}{\text{Vfit}}$$

Out[113]=



Estimated work calculated using the simulation is 3.45923×10^{-18} J.

Percent error compared to the theoretical work is 0.384344%

In[116]:=

Collisions of Two Particles at a Time

```
Sim[nParticles_, m_, length_] := {
  Clear[xPos, v, xParticles, vParticles];

  F = 0;
  dpTotal = 0;
  collisiondata = {};
  isrunning = {};

  (*--------------------------------------------------------------------------*)

  xParticlesLoop = {};
  For[n = 1, n ≤ nParticles, n++,
   AppendTo[xParticlesLoop,
     {RandomReal[{0, length}], RandomReal[{0, length}], RandomReal[{0, length}]}];
  ];
  xParticles = {xParticlesLoop};

  vTotalArray = {};
  vParticlesLoop = {};
  For[n = 1, n ≤ nParticles, n++,
   vTotal = Sqrt[vAdd〚1〛^2 + vAdd〚2〛^2 + vAdd〚3〛^2];
   AppendTo[vParticlesLoop, {vTherm, vTherm, vTherm}];
   AppendTo[vTotalArray, vTotal]
  ];
  vParticles = {vParticlesLoop};


  (*--------------------------------------------------------------------------*)


  For[i = 1, i ≤ nSteps, i++,
   allParticlesAtOneTime = {};
   allVelocitiesAtOneTime = {};


   (*--------------------------------------------------------------------------*)


   positionsAtTime = xParticles〚i〛;
   distanceArray = {};

   For[k = 1, k ≤ Length[positionsAtTime], k++,
    firstParticlePos = positionsAtTime〚k〛;
    distanceWithEachParticle = {};
```

```
 For[l = 1, l ≤ Length[positionsAtTime], l++,
  If[l ≠ k,
    secondParticlePos = positionsAtTime[[l]];
    xdist = secondParticlePos[[1]] - firstParticlePos[[1]];
    ydist = secondParticlePos[[2]] - firstParticlePos[[2]];
    zdist = secondParticlePos[[3]] - firstParticlePos[[3]];
    distance = Sqrt[xdist^2 + ydist^2 + zdist^2];
    AppendTo[distanceWithEachParticle, {{k, l}, distance}];
   ];
 ];

 AppendTo[distanceArray, distanceWithEachParticle];
];

collidingParticles = {};
For[k = 1, k ≤ Length[distanceArray], k++,
 distancesWithParticleCenter = distanceArray[[k]];
 For[l = 1, l ≤ Length[distancesWithParticleCenter], l++,
  If[distancesWithParticleCenter[[l]][[2]] ≤ 2 r,
    AppendTo[collidingParticles, distancesWithParticleCenter[[l]][[1]]];
   ];
 ];
];

collidingParticles1Darray = {};
For[k = 1, k ≤ Length[collidingParticles], k++,
 For[l = 1, l ≤ Length[collidingParticles[[k]]], l++,
   AppendTo[collidingParticles1Darray, collidingParticles[[k]][[l]]];
   AppendTo[collidingParticles1Darray, collidingParticles[[k]][[l]]];
  ];
];
collidingParticles = DeleteDuplicates[collidingParticles1Darray];

AppendTo[collisiondata, collidingParticles];

If[Length[collidingParticles] == 2,
 AppendTo[isrunning, 1];
 particle1 = collidingParticles[[1]];
 particle2 = collidingParticles[[2]];
 v1 = vParticles[[i]][[particle1]];
 v2 = vParticles[[i]][[particle2]];

 vParticles[[i]][[particle1]] = v2;
 vParticles[[i]][[particle2]] = v1;
];
```

```
(*-----------------------------------------------------------------------*)


For[n = 1, n ≤ nParticles, n++,
 xPos = xParticles[[i]][[n]];
 v = vParticles[[i]][[n]];
 xPos += v dt;

 If[xPos[[1]] < 0, xPos[[1]] = -xPos[[1]];
  dp = Abs[2 m v[[1]]];
  dpTotal += dp;
  v[[1]] = -v[[1]];];
 If[xPos[[1]] > length, xPos[[1]] = 2 * length - xPos[[1]];
  dp = Abs[2 m v[[1]]];
  dpTotal += dp;
  v[[1]] = -v[[1]];];
 If[xPos[[2]] < 0, xPos[[2]] = -xPos[[2]];
  dp = Abs[2 m v[[2]]];
  dpTotal += dp;
  v[[2]] = -v[[2]];];
 If[xPos[[2]] > length, xPos[[2]] = 2 * length - xPos[[2]];
  dp = Abs[2 m v[[2]]];
  dpTotal += dp;
  v[[2]] = -v[[2]];];
 If[xPos[[3]] < 0, xPos[[3]] = -xPos[[3]];
  dp = Abs[2 m v[[3]]];
  dpTotal += dp;
  v[[3]] = -v[[3]];];
 If[xPos[[3]] > length, xPos[[3]] = 2 * length - xPos[[3]];
  dp = Abs[2 m v[[3]]];
  dpTotal += dp;
  v[[3]] = -v[[3]];];

 AppendTo[allParticlesAtOneTime, xPos];
 AppendTo[allVelocitiesAtOneTime, v];
 vTotal = Sqrt[v[[1]]^2 + v[[2]]^2 + v[[3]]^2];
 AppendTo[vTotalArray, vTotal];

];


(*-----------------------------------------------------------------------*)


AppendTo[xParticles, allParticlesAtOneTime];
AppendTo[vParticles, allVelocitiesAtOneTime];
```

```
  ];

  F += dpTotal / time;

 }
```

In[118]:=

```
RT1 = 8.314 * 273; (*units J/mol = (kg m2/s2)/mol*)
M = 39.9 / 1000; (*molar mass: units kg*)
vTherm := Random[NormalDistribution[0, Sqrt[(RT1) / (M)]]] (*units m/s*)
m = 6.6335209 * (10^(-26)); (*atomic mass of Argon in kg*)
r = 9.8*^-11; (*atomic radius of Argon in m*)

length = 1*^-8;
nParticles = 100;
time = 2*^-11; (*units sec*)
dt = 2*^-13;
nSteps = Round[time / dt];

Sim[nParticles, m, length];

area = 6 * (length^2);
V1 = length^3;
P1 = F / area;
n = nParticles / (6.02 * (10^23));
Z = (P1 V1) / (n RT1);

Print["total force: ", F, " N"]
Print["total pressure: ", P1, " Pa"]
Print["compressibility: ", Z]

σ = ((RT1) / M)^0.5;
MaxwellCurve = Plot[PDF[MaxwellDistribution[σ], x],
   {x, 0, 1000}, PlotRange → {{0, 1000}, {0, .0035}}, PlotStyle → Thick];
vTotalDist = Histogram[vTotalArray, bins = 50, "PDF"];
Show[vTotalDist, MaxwellCurve];
```

Part: Part specification vAdd⟦1⟧ is longer than depth of object.

Part: Part specification vAdd⟦2⟧ is longer than depth of object.

Part: Part specification vAdd⟦3⟧ is longer than depth of object.

General: Further output of Part::partd will be suppressed during this calculation.

total force: 2.49266×10^{-10} N

total pressure: 415443. Pa

compressibility: 1.10188

In[141]:=
```
pointsize = 2 r / length
V1
VolParticles = nParticles (4 / 3) (Pi) (r) ^3
availableV = V1 - VolParticles
volumeRatio = availableV / V1
(P1 availableV) / (n RT1)
collisiondata
isrunning
```

Out[141]=
0.0196

Out[142]=
$$\frac{1}{1\,000\,000\,000\,000\,000\,000\,000\,000}$$

Out[143]=
3.94246×10^{-28}

Out[144]=
9.99606×10^{-25}

Out[145]=
0.999606

Out[146]=
1.10145

Out[147]=
```
{{}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {},
 {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {},
 {8, 22}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {},
 {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {},
 {}, {}, {}, {}, {}, {}, {}, {43, 91}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}, {}}
```

Out[148]=
{1, 1}

In[149]:=
```
v1
```

Out[149]=
{22.7896, -234.759, 244.133}

In[150]:=
```
v2
```

Out[150]=
{254.279, 245.031, -11.357}

In[151]:=
```
vParticles[[17]][[9]]
vParticles[[18]][[66]]
```

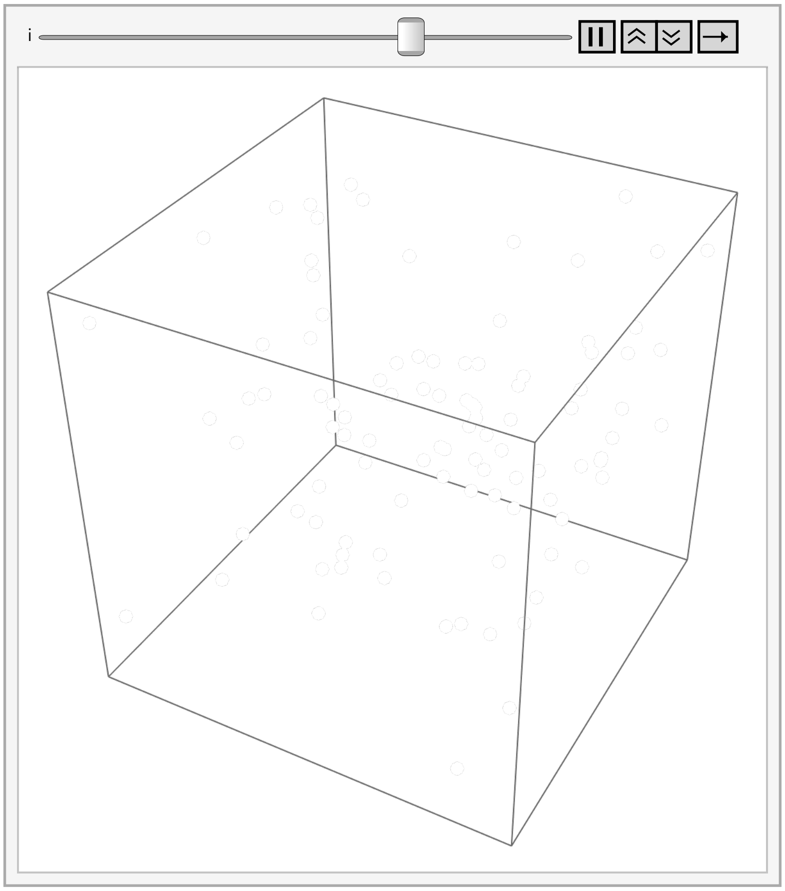
Out[151]=
{157.879, -179.128, 206.592}

Out[152]=
{55.8634, -317.54, 290.723}

In[153]:=

```
v = Map[Point, xParticles];
animation = Animate[Graphics3D[{White, PointSize[pointsize], v⟦i⟧},
    PlotRange → {{0, length}, {0, length}, {0, length}}], {i, 1, Length[v], 1},
   AnimationRate → 10, RefreshRate → 100, DisplayAllSteps → True]
```

Out[154]=



In[155]:=

Proving that Velocities Switch as Assumed for the Collisions Simulation

In[156]:=

```
Clear[v1x, v2x, v1y, v2y, v1z, v2z];
sol = Solve[
  m v1[[1]] + m v2[[1]] == m v1x + m v2x && 0.5 m v1[[1]]^2 + 0.5 m v2[[1]]^2 == 0.5 m v1x^2 + 0.5 m v2x^2 &&
   m v1[[2]] + m v2[[2]] == m v1y + m v2y && 0.5 m v1[[2]]^2 + 0.5 m v2[[2]]^2 == 0.5 m v1y^2 + 0.5 m v2y^2 &&
   m v1[[3]] + m v2[[3]] == m v1z + m v2z && 0.5 m v1[[3]]^2 + 0.5 m v2[[3]]^2 == 0.5 m v1z^2 + 0.5 m v2z^2 &&
   v1x ≠ v1[[1]], {v1x, v2x, v1y, v2y, v1z, v2z}]

finalSol = {1, 1, 1, 1, 1, 1};
For[i = 1, i ≤ Length[sol], i++,
 If[sol[[i]][[1]][[2]] ≠ v1[[1]], finalSol[[1]] = sol[[i]][[1]];];
 If[sol[[i]][[2]][[2]] ≠ v2[[1]], finalSol[[2]] = sol[[i]][[2]];];
 If[sol[[i]][[3]][[2]] ≠ v1[[2]], finalSol[[3]] = sol[[i]][[3]];];
 If[sol[[i]][[4]][[2]] ≠ v2[[2]], finalSol[[4]] = sol[[i]][[4]];];
 If[sol[[i]][[5]][[2]] ≠ v1[[3]], finalSol[[5]] = sol[[i]][[5]];];
 If[sol[[i]][[6]][[2]] ≠ v2[[3]], finalSol[[6]] = sol[[i]][[6]];];
]
finalSolArray = finalSol /. Rule → List
v1x = finalSolArray[[1]][[2]]
v2x = finalSolArray[[2]][[2]]
v1y = finalSolArray[[3]][[2]]
v2y = finalSolArray[[4]][[2]]
v1z = finalSolArray[[5]][[2]]
v2z = finalSolArray[[6]][[2]]
```

Solve: Solve was unable to solve the system with inexact coefficients. The answer was obtained by solving a corresponding exact system and numericizing the result.

Solve: Solve was unable to solve the system with inexact coefficients. The answer was obtained by solving a corresponding exact system and numericizing the result.

Solve: Solve was unable to solve the system with inexact coefficients. The answer was obtained by solving a corresponding exact system and numericizing the result.

General: Further output of Solve::ratnz will be suppressed during this calculation.

Out[157]=

```
{{v1x → 22.7896, v2x → 254.279, v1y → -234.759, v2y → 245.031, v1z → -11.357, v2z → 244.133},
 {v1x → 22.7896, v2x → 254.279, v1y → 245.031, v2y → -234.759, v1z → -11.357, v2z → 244.133},
 {v1x → 22.7896, v2x → 254.279, v1y → -234.759, v2y → 245.031, v1z → 244.133, v2z → -11.357},
 {v1x → 22.7896, v2x → 254.279, v1y → 245.031, v2y → -234.759, v1z → 244.133, v2z → -11.357},
 {v1x → 254.279, v2x → 22.7896, v1y → -234.759, v2y → 245.031, v1z → -11.357, v2z → 244.133},
 {v1x → 254.279, v2x → 22.7896, v1y → 245.031, v2y → -234.759, v1z → -11.357, v2z → 244.133},
 {v1x → 254.279, v2x → 22.7896, v1y → -234.759, v2y → 245.031, v1z → 244.133, v2z → -11.357},
 {v1x → 254.279, v2x → 22.7896, v1y → 245.031, v2y → -234.759, v1z → 244.133, v2z → -11.357}}
```

Out[160]=

```
{{v1x, 254.279}, {v2x, 22.7896}, {v1y, 245.031},
 {v2y, -234.759}, {v1z, -11.357}, {v2z, 244.133}}
```

Out[161]=

254.279

Out[162]=

22.7896

Out[163]=

245.031

Out[164]=

-234.759

Out[165]=

-11.357

Out[166]=

244.133

In[167]:=

```
v1[[1]]
v2[[1]]
v1[[2]]
v2[[2]]
v1[[3]]
v2[[3]]
```

Out[167]=

22.7896

Out[168]=

254.279

Out[169]=

-234.759

Out[170]=

245.031

Out[171]=

244.133

Out[172]=

-11.357