```
 1  using System;
 2
 3  namespace AtomicGroundStateEnergiesGaussian
 4  {
 5      class BerylliumAtom
 6      {
 7          private Integration integ;
 8          private double N;
 9          private int Z;
10          public double integ1, integ2, integ3;
11          public double integ4, integ5, integ6;
12          public double integ7, integ8, integ9;
13          public double integA, integB, integC;
14          public double integD, integE;
15          public BerylliumAtom()
16          {
17              integ = new Integration();
18          }
19
20          public double Psi(double[] x, double[] alpha)
21          {
22              double r1 = Math.Sqrt(x[0] * x[0] + x[1] * x[1] + x[2] * x[2]);
23              double r2 = Math.Sqrt(x[3] * x[3] + x[4] * x[4] + x[5] * x[5]);
24              double r3 = Math.Sqrt(x[6] * x[6] + x[7] * x[7] + x[8] * x[8]);
25              double r4 = Math.Sqrt(x[9] * x[9] + x[10] * x[10] + x[11] * x[11]);
26              double r12 = Math.Sqrt(Math.Pow(x[0] - x[3], 2.0) +
27                  Math.Pow(x[1] - x[4], 2.0) + Math.Pow(x[2] - x[5], 2.0));
28              double r13 = Math.Sqrt(Math.Pow(x[0] - x[6], 2.0) +
29                  Math.Pow(x[1] - x[7], 2.0) + Math.Pow(x[2] - x[8], 2.0));
30              double r14 = Math.Sqrt(Math.Pow(x[0] - x[9], 2.0) +
31                  Math.Pow(x[1] - x[10], 2.0) + Math.Pow(x[2] - x[11], 2.0));
32              double r23 = Math.Sqrt(Math.Pow(x[3] - x[6], 2.0) +
33                  Math.Pow(x[4] - x[7], 2.0) + Math.Pow(x[5] - x[8], 2.0));
34              double r24 = Math.Sqrt(Math.Pow(x[3] - x[9], 2.0) +
35                  Math.Pow(x[4] - x[10], 2.0) + Math.Pow(x[5] - x[11], 2.0));
36              double r34 = Math.Sqrt(Math.Pow(x[6] - x[9], 2.0) +
37                  Math.Pow(x[7] - x[10], 2.0) + Math.Pow(x[8] - x[11], 2.0));
38              double exp1 = Math.Exp(-alpha[0] * r1);
39              double exp2 = Math.Exp(-alpha[1] * r2);
40              double exp3 = Math.Exp(-alpha[2] * r2);
41              double exp4 = Math.Exp(-alpha[3] * r12);
42              double exp5 = Math.Exp(-alpha[4] * r13);
43              double exp6 = Math.Exp(-alpha[4] * r14);
44              double exp7 = Math.Exp(-alpha[4] * r23);
45              double exp8 = Math.Exp(-alpha[4] * r24);
46              double exp9 = Math.Exp(-alpha[4] * r34);
47
48              return exp1 * exp2 * exp3 * exp4 * exp5 *
49                  exp6 * exp7 * exp8 * exp9;
50          }
51
52          public double Psi2(double[] x, double[] alpha)
```

```
53          {
54              double psi = Psi(x, alpha);
55
56              return psi * psi;
57          }
58
59          public double Normalize(double[] alpha, int nSteps)
60          {
61              double[] lower = new double[12];
62              double[] upper = new double[12];
63              int[] steps = new int[12];
64
65              lower[0] = 0.001;
66              lower[1] = 0.001;
67              lower[2] = 0.001;
68              lower[3] = 0.001;
69              lower[4] = 0.001;
70              lower[5] = 0.001;
71              lower[6] = 0.001;
72              lower[7] = 0.001;
73              lower[8] = 0.001;
74              lower[9] = 0.001;
75              lower[10] = 0.001;
76              lower[11] = 0.001;
77
78              upper[0] = 10.0;
79              upper[1] = 10.0;
80              upper[2] = 10.0;
81              upper[3] = 10.0;
82              upper[4] = 10.0;
83              upper[5] = 10.0;
84              upper[6] = 10.0;
85              upper[7] = 10.0;
86              upper[8] = 10.0;
87              upper[9] = 10.0;
88              upper[10] = 10.0;
89              upper[11] = 10.0;
90
91              for (int i = 0; i < 12; i++)
92                  steps[i] = nSteps;
93
94              double norm = Math.Sqrt(integ.Integrate(
95                  lower, upper, alpha, Psi2, 12, steps));
96
97              return norm;
98          }
99
100         public double Integrand1(double[] x, double[] alpha)
101         {
102             double r1 = Math.Sqrt(x[0] * x[0] + x[1] * x[1] + x[2] * x[2]);
103             double r2 = Math.Sqrt(x[3] * x[3] + x[4] * x[4] + x[5] * x[5]);
104             double r3 = Math.Sqrt(x[6] * x[6] + x[7] * x[7] + x[8] * x[8]);
```

```
105             double r4 = Math.Sqrt(x[9] * x[9] + x[10] * x[10] + x[11] * x[11]);
106             double r12 = Math.Sqrt(Math.Pow(x[0] - x[3], 2.0) +
107                 Math.Pow(x[1] - x[4], 2.0) + Math.Pow(x[2] - x[5], 2.0));
108             double r13 = Math.Sqrt(Math.Pow(x[0] - x[6], 2.0) +
109                 Math.Pow(x[1] - x[7], 2.0) + Math.Pow(x[2] - x[8], 2.0));
110             double r14 = Math.Sqrt(Math.Pow(x[0] - x[9], 2.0) +
111                 Math.Pow(x[1] - x[10], 2.0) + Math.Pow(x[2] - x[11], 2.0));
112             double r23 = Math.Sqrt(Math.Pow(x[3] - x[6], 2.0) +
113                 Math.Pow(x[4] - x[7], 2.0) + Math.Pow(x[5] - x[8], 2.0));
114             double r24 = Math.Sqrt(Math.Pow(x[3] - x[9], 2.0) +
115                 Math.Pow(x[4] - x[10], 2.0) + Math.Pow(x[5] - x[11], 2.0));
116             double r34 = Math.Sqrt(Math.Pow(x[6] - x[9], 2.0) +
117                 Math.Pow(x[7] - x[10], 2.0) + Math.Pow(x[8] - x[11], 2.0));
118             double term = -2.0 * alpha[0] / r1 + alpha[0] * alpha[0];
119             double exp1 = Math.Exp(-2.0 * alpha[0] * r1);
120             double exp2 = Math.Exp(-2.0 * alpha[1] * r2);
121             double exp3 = Math.Exp(-2.0 * alpha[2] * r3);
122             double exp4 = Math.Exp(-2.0 * alpha[3] * r4);
123             double exp5 = Math.Exp(-2.0 * alpha[4] * r12);
124             double exp6 = Math.Exp(-2.0 * alpha[5] * r13);
125             double exp7 = Math.Exp(-2.0 * alpha[6] * r14);
126             double exp8 = Math.Exp(-2.0 * alpha[7] * r23);
127             double exp9 = Math.Exp(-2.0 * alpha[8] * r24);
128             double expA = Math.Exp(-2.0 * alpha[9] * r34);
129 
130             return N * N * term * exp1 * exp2 * exp3 * exp4 * exp5 *
131                 exp6 * exp7 * exp8 * exp9 * expA;
132         }
133 
134         public double Integrand2(double[] x, double[] alpha)
135         {
136             double r1 = Math.Sqrt(x[0] * x[0] + x[1] * x[1] + x[2] * x[2]);
137             double r2 = Math.Sqrt(x[3] * x[3] + x[4] * x[4] + x[5] * x[5]);
138             double r3 = Math.Sqrt(x[6] * x[6] + x[7] * x[7] + x[8] * x[8]);
139             double r4 = Math.Sqrt(x[9] * x[9] + x[10] * x[10] + x[11] * x[11]);
140             double r12 = Math.Sqrt(Math.Pow(x[0] - x[3], 2.0) +
141                 Math.Pow(x[1] - x[4], 2.0) + Math.Pow(x[2] - x[5], 2.0));
142             double r13 = Math.Sqrt(Math.Pow(x[0] - x[6], 2.0) +
143                 Math.Pow(x[1] - x[7], 2.0) + Math.Pow(x[2] - x[8], 2.0));
144             double r14 = Math.Sqrt(Math.Pow(x[0] - x[9], 2.0) +
145                 Math.Pow(x[1] - x[10], 2.0) + Math.Pow(x[2] - x[11], 2.0));
146             double r23 = Math.Sqrt(Math.Pow(x[3] - x[6], 2.0) +
147                 Math.Pow(x[4] - x[7], 2.0) + Math.Pow(x[5] - x[8], 2.0));
148             double r24 = Math.Sqrt(Math.Pow(x[3] - x[9], 2.0) +
149                 Math.Pow(x[4] - x[10], 2.0) + Math.Pow(x[5] - x[11], 2.0));
150             double r34 = Math.Sqrt(Math.Pow(x[6] - x[9], 2.0) +
151                 Math.Pow(x[7] - x[10], 2.0) + Math.Pow(x[8] - x[11], 2.0));
152             double term = -2.0 * alpha[1] / r1 + alpha[1] * alpha[1];
153             double exp1 = Math.Exp(-2.0 * alpha[0] * r1);
154             double exp2 = Math.Exp(-2.0 * alpha[1] * r2);
155             double exp3 = Math.Exp(-2.0 * alpha[2] * r3);
156             double exp4 = Math.Exp(-2.0 * alpha[3] * r4);
```

```
157             double exp5 = Math.Exp(-2.0 * alpha[4] * r12);
158             double exp6 = Math.Exp(-2.0 * alpha[5] * r13);
159             double exp7 = Math.Exp(-2.0 * alpha[6] * r14);
160             double exp8 = Math.Exp(-2.0 * alpha[7] * r23);
161             double exp9 = Math.Exp(-2.0 * alpha[8] * r24);
162             double expA = Math.Exp(-2.0 * alpha[9] * r34);
163 
164             return N * N * term * exp1 * exp2 * exp3 * exp4 * exp5 *
165                 exp6 * exp7 * exp8 * exp9 * expA;
166         }
167 
168         public double Integrand3(double[] x, double[] alpha)
169         {
170             double r1 = Math.Sqrt(x[0] * x[0] + x[1] * x[1] + x[2] * x[2]);
171             double r2 = Math.Sqrt(x[3] * x[3] + x[4] * x[4] + x[5] * x[5]);
172             double r3 = Math.Sqrt(x[6] * x[6] + x[7] * x[7] + x[8] * x[8]);
173             double r4 = Math.Sqrt(x[9] * x[9] + x[10] * x[10] + x[11] * x[11]);
174             double r12 = Math.Sqrt(Math.Pow(x[0] - x[3], 2.0) +
175                 Math.Pow(x[1] - x[4], 2.0) + Math.Pow(x[2] - x[5], 2.0));
176             double r13 = Math.Sqrt(Math.Pow(x[0] - x[6], 2.0) +
177                 Math.Pow(x[1] - x[7], 2.0) + Math.Pow(x[2] - x[8], 2.0));
178             double r14 = Math.Sqrt(Math.Pow(x[0] - x[9], 2.0) +
179                 Math.Pow(x[1] - x[10], 2.0) + Math.Pow(x[2] - x[11], 2.0));
180             double r23 = Math.Sqrt(Math.Pow(x[3] - x[6], 2.0) +
181                 Math.Pow(x[4] - x[7], 2.0) + Math.Pow(x[5] - x[8], 2.0));
182             double r24 = Math.Sqrt(Math.Pow(x[3] - x[9], 2.0) +
183                 Math.Pow(x[4] - x[10], 2.0) + Math.Pow(x[5] - x[11], 2.0));
184             double r34 = Math.Sqrt(Math.Pow(x[6] - x[9], 2.0) +
185                 Math.Pow(x[7] - x[10], 2.0) + Math.Pow(x[8] - x[11], 2.0));
186             double term = -2.0 * alpha[2] / r1 + alpha[2] * alpha[2];
187             double exp1 = Math.Exp(-2.0 * alpha[0] * r1);
188             double exp2 = Math.Exp(-2.0 * alpha[1] * r2);
189             double exp3 = Math.Exp(-2.0 * alpha[2] * r3);
190             double exp4 = Math.Exp(-2.0 * alpha[3] * r4);
191             double exp5 = Math.Exp(-2.0 * alpha[4] * r12);
192             double exp6 = Math.Exp(-2.0 * alpha[5] * r13);
193             double exp7 = Math.Exp(-2.0 * alpha[6] * r14);
194             double exp8 = Math.Exp(-2.0 * alpha[7] * r23);
195             double exp9 = Math.Exp(-2.0 * alpha[8] * r24);
196             double expA = Math.Exp(-2.0 * alpha[9] * r34);
197 
198             return N * N * term * exp1 * exp2 * exp3 * exp4 * exp5 *
199                 exp6 * exp7 * exp8 * exp9 * expA;
200         }
201 
202         public double Integrand4(double[] x, double[] alpha)
203         {
204             double r1 = Math.Sqrt(x[0] * x[0] + x[1] * x[1] + x[2] * x[2]);
205             double r2 = Math.Sqrt(x[3] * x[3] + x[4] * x[4] + x[5] * x[5]);
206             double r3 = Math.Sqrt(x[6] * x[6] + x[7] * x[7] + x[8] * x[8]);
207             double r4 = Math.Sqrt(x[9] * x[9] + x[10] * x[10] + x[11] * x[11]);
208             double r12 = Math.Sqrt(Math.Pow(x[0] - x[3], 2.0) +
```

```
209                 Math.Pow(x[1] - x[4], 2.0) + Math.Pow(x[2] - x[5], 2.0));
210             double r13 = Math.Sqrt(Math.Pow(x[0] - x[6], 2.0) +
211                 Math.Pow(x[1] - x[7], 2.0) + Math.Pow(x[2] - x[8], 2.0));
212             double r14 = Math.Sqrt(Math.Pow(x[0] - x[9], 2.0) +
213                 Math.Pow(x[1] - x[10], 2.0) + Math.Pow(x[2] - x[11], 2.0));
214             double r23 = Math.Sqrt(Math.Pow(x[3] - x[6], 2.0) +
215                 Math.Pow(x[4] - x[7], 2.0) + Math.Pow(x[5] - x[8], 2.0));
216             double r24 = Math.Sqrt(Math.Pow(x[3] - x[9], 2.0) +
217                 Math.Pow(x[4] - x[10], 2.0) + Math.Pow(x[5] - x[11], 2.0));
218             double r34 = Math.Sqrt(Math.Pow(x[6] - x[9], 2.0) +
219                 Math.Pow(x[7] - x[10], 2.0) + Math.Pow(x[8] - x[11], 2.0));
220             double term = -2.0 * alpha[3] / r1 + alpha[3] * alpha[3];
221             double exp1 = Math.Exp(-2.0 * alpha[0] * r1);
222             double exp2 = Math.Exp(-2.0 * alpha[1] * r2);
223             double exp3 = Math.Exp(-2.0 * alpha[2] * r3);
224             double exp4 = Math.Exp(-2.0 * alpha[3] * r4);
225             double exp5 = Math.Exp(-2.0 * alpha[4] * r12);
226             double exp6 = Math.Exp(-2.0 * alpha[5] * r13);
227             double exp7 = Math.Exp(-2.0 * alpha[6] * r14);
228             double exp8 = Math.Exp(-2.0 * alpha[7] * r23);
229             double exp9 = Math.Exp(-2.0 * alpha[8] * r24);
230             double expA = Math.Exp(-2.0 * alpha[9] * r34);
231 
232             return N * N * term * exp1 * exp2 * exp3 * exp4 * exp5 *
233                 exp6 * exp7 * exp8 * exp9 * expA;
234         }
235 
236         public double Integrand5(double[] x, double[] alpha)
237         {
238             double r1 = Math.Sqrt(x[0] * x[0] + x[1] * x[1] + x[2] * x[2]);
239             double r2 = Math.Sqrt(x[3] * x[3] + x[4] * x[4] + x[5] * x[5]);
240             double r3 = Math.Sqrt(x[6] * x[6] + x[7] * x[7] + x[8] * x[8]);
241             double r4 = Math.Sqrt(x[9] * x[9] + x[10] * x[10] + x[11] * x[11]);
242             double r12 = Math.Sqrt(Math.Pow(x[0] - x[3], 2.0) +
243                 Math.Pow(x[1] - x[4], 2.0) + Math.Pow(x[2] - x[5], 2.0));
244             double r13 = Math.Sqrt(Math.Pow(x[0] - x[6], 2.0) +
245                 Math.Pow(x[1] - x[7], 2.0) + Math.Pow(x[2] - x[8], 2.0));
246             double r14 = Math.Sqrt(Math.Pow(x[0] - x[9], 2.0) +
247                 Math.Pow(x[1] - x[10], 2.0) + Math.Pow(x[2] - x[11], 2.0));
248             double r23 = Math.Sqrt(Math.Pow(x[3] - x[6], 2.0) +
249                 Math.Pow(x[4] - x[7], 2.0) + Math.Pow(x[5] - x[8], 2.0));
250             double r24 = Math.Sqrt(Math.Pow(x[3] - x[9], 2.0) +
251                 Math.Pow(x[4] - x[10], 2.0) + Math.Pow(x[5] - x[11], 2.0));
252             double r34 = Math.Sqrt(Math.Pow(x[6] - x[9], 2.0) +
253                 Math.Pow(x[7] - x[10], 2.0) + Math.Pow(x[8] - x[11], 2.0));
254             double term = 1.0 / r1;
255             double exp1 = Math.Exp(-2.0 * alpha[0] * r1);
256             double exp2 = Math.Exp(-2.0 * alpha[1] * r2);
257             double exp3 = Math.Exp(-2.0 * alpha[2] * r3);
258             double exp4 = Math.Exp(-2.0 * alpha[3] * r4);
259             double exp5 = Math.Exp(-2.0 * alpha[4] * r12);
260             double exp6 = Math.Exp(-2.0 * alpha[5] * r13);
```

```
261             double exp7 = Math.Exp(-2.0 * alpha[6] * r14);
262             double exp8 = Math.Exp(-2.0 * alpha[7] * r23);
263             double exp9 = Math.Exp(-2.0 * alpha[8] * r24);
264             double expA = Math.Exp(-2.0 * alpha[9] * r34);
265 
266             return N * N * term * exp1 * exp2 * exp3 * exp4 * exp5 *
267                 exp6 * exp7 * exp8 * exp9 * expA;
268         }
269 
270         public double Integrand6(double[] x, double[] alpha)
271         {
272             double r1 = Math.Sqrt(x[0] * x[0] + x[1] * x[1] + x[2] * x[2]);
273             double r2 = Math.Sqrt(x[3] * x[3] + x[4] * x[4] + x[5] * x[5]);
274             double r3 = Math.Sqrt(x[6] * x[6] + x[7] * x[7] + x[8] * x[8]);
275             double r4 = Math.Sqrt(x[9] * x[9] + x[10] * x[10] + x[11] * x[11]);
276             double r12 = Math.Sqrt(Math.Pow(x[0] - x[3], 2.0) +
277                 Math.Pow(x[1] - x[4], 2.0) + Math.Pow(x[2] - x[5], 2.0));
278             double r13 = Math.Sqrt(Math.Pow(x[0] - x[6], 2.0) +
279                 Math.Pow(x[1] - x[7], 2.0) + Math.Pow(x[2] - x[8], 2.0));
280             double r14 = Math.Sqrt(Math.Pow(x[0] - x[9], 2.0) +
281                 Math.Pow(x[1] - x[10], 2.0) + Math.Pow(x[2] - x[11], 2.0));
282             double r23 = Math.Sqrt(Math.Pow(x[3] - x[6], 2.0) +
283                 Math.Pow(x[4] - x[7], 2.0) + Math.Pow(x[5] - x[8], 2.0));
284             double r24 = Math.Sqrt(Math.Pow(x[3] - x[9], 2.0) +
285                 Math.Pow(x[4] - x[10], 2.0) + Math.Pow(x[5] - x[11], 2.0));
286             double r34 = Math.Sqrt(Math.Pow(x[6] - x[9], 2.0) +
287                 Math.Pow(x[7] - x[10], 2.0) + Math.Pow(x[8] - x[11], 2.0));
288             double term = 1.0 / r2;
289             double exp1 = Math.Exp(-2.0 * alpha[0] * r1);
290             double exp2 = Math.Exp(-2.0 * alpha[1] * r2);
291             double exp3 = Math.Exp(-2.0 * alpha[2] * r3);
292             double exp4 = Math.Exp(-2.0 * alpha[3] * r4);
293             double exp5 = Math.Exp(-2.0 * alpha[4] * r12);
294             double exp6 = Math.Exp(-2.0 * alpha[5] * r13);
295             double exp7 = Math.Exp(-2.0 * alpha[6] * r14);
296             double exp8 = Math.Exp(-2.0 * alpha[7] * r23);
297             double exp9 = Math.Exp(-2.0 * alpha[8] * r24);
298             double expA = Math.Exp(-2.0 * alpha[9] * r34);
299 
300             return N * N * term * exp1 * exp2 * exp3 * exp4 * exp5 *
301                 exp6 * exp7 * exp8 * exp9 * expA;
302         }
303 
304         public double Integrand7(double[] x, double[] alpha)
305         {
306             double r1 = Math.Sqrt(x[0] * x[0] + x[1] * x[1] + x[2] * x[2]);
307             double r2 = Math.Sqrt(x[3] * x[3] + x[4] * x[4] + x[5] * x[5]);
308             double r3 = Math.Sqrt(x[6] * x[6] + x[7] * x[7] + x[8] * x[8]);
309             double r4 = Math.Sqrt(x[9] * x[9] + x[10] * x[10] + x[11] * x[11]);
310             double r12 = Math.Sqrt(Math.Pow(x[0] - x[3], 2.0) +
311                 Math.Pow(x[1] - x[4], 2.0) + Math.Pow(x[2] - x[5], 2.0));
312             double r13 = Math.Sqrt(Math.Pow(x[0] - x[6], 2.0) +
```

```
313                 Math.Pow(x[1] - x[7], 2.0) + Math.Pow(x[2] - x[8], 2.0));
314             double r14 = Math.Sqrt(Math.Pow(x[0] - x[9], 2.0) +
315                 Math.Pow(x[1] - x[10], 2.0) + Math.Pow(x[2] - x[11], 2.0));
316             double r23 = Math.Sqrt(Math.Pow(x[3] - x[6], 2.0) +
317                 Math.Pow(x[4] - x[7], 2.0) + Math.Pow(x[5] - x[8], 2.0));
318             double r24 = Math.Sqrt(Math.Pow(x[3] - x[9], 2.0) +
319                 Math.Pow(x[4] - x[10], 2.0) + Math.Pow(x[5] - x[11], 2.0));
320             double r34 = Math.Sqrt(Math.Pow(x[6] - x[9], 2.0) +
321                 Math.Pow(x[7] - x[10], 2.0) + Math.Pow(x[8] - x[11], 2.0));
322             double term = 1.0 / r3;
323             double exp1 = Math.Exp(-2.0 * alpha[0] * r1);
324             double exp2 = Math.Exp(-2.0 * alpha[1] * r2);
325             double exp3 = Math.Exp(-2.0 * alpha[2] * r3);
326             double exp4 = Math.Exp(-2.0 * alpha[3] * r4);
327             double exp5 = Math.Exp(-2.0 * alpha[4] * r12);
328             double exp6 = Math.Exp(-2.0 * alpha[5] * r13);
329             double exp7 = Math.Exp(-2.0 * alpha[6] * r14);
330             double exp8 = Math.Exp(-2.0 * alpha[7] * r23);
331             double exp9 = Math.Exp(-2.0 * alpha[8] * r24);
332             double expA = Math.Exp(-2.0 * alpha[9] * r34);
333 
334             return N * N * term * exp1 * exp2 * exp3 * exp4 * exp5 *
335                 exp6 * exp7 * exp8 * exp9 * expA;
336         }
337 
338         public double Integrand8(double[] x, double[] alpha)
339         {
340             double r1 = Math.Sqrt(x[0] * x[0] + x[1] * x[1] + x[2] * x[2]);
341             double r2 = Math.Sqrt(x[3] * x[3] + x[4] * x[4] + x[5] * x[5]);
342             double r3 = Math.Sqrt(x[6] * x[6] + x[7] * x[7] + x[8] * x[8]);
343             double r4 = Math.Sqrt(x[9] * x[9] + x[10] * x[10] + x[11] * x[11]);
344             double r12 = Math.Sqrt(Math.Pow(x[0] - x[3], 2.0) +
345                 Math.Pow(x[1] - x[4], 2.0) + Math.Pow(x[2] - x[5], 2.0));
346             double r13 = Math.Sqrt(Math.Pow(x[0] - x[6], 2.0) +
347                 Math.Pow(x[1] - x[7], 2.0) + Math.Pow(x[2] - x[8], 2.0));
348             double r14 = Math.Sqrt(Math.Pow(x[0] - x[9], 2.0) +
349                 Math.Pow(x[1] - x[10], 2.0) + Math.Pow(x[2] - x[11], 2.0));
350             double r23 = Math.Sqrt(Math.Pow(x[3] - x[6], 2.0) +
351                 Math.Pow(x[4] - x[7], 2.0) + Math.Pow(x[5] - x[8], 2.0));
352             double r24 = Math.Sqrt(Math.Pow(x[3] - x[9], 2.0) +
353                 Math.Pow(x[4] - x[10], 2.0) + Math.Pow(x[5] - x[11], 2.0));
354             double r34 = Math.Sqrt(Math.Pow(x[6] - x[9], 2.0) +
355                 Math.Pow(x[7] - x[10], 2.0) + Math.Pow(x[8] - x[11], 2.0));
356             double term = 1.0 / r4;
357             double exp1 = Math.Exp(-2.0 * alpha[0] * r1);
358             double exp2 = Math.Exp(-2.0 * alpha[1] * r2);
359             double exp3 = Math.Exp(-2.0 * alpha[2] * r3);
360             double exp4 = Math.Exp(-2.0 * alpha[3] * r4);
361             double exp5 = Math.Exp(-2.0 * alpha[4] * r12);
362             double exp6 = Math.Exp(-2.0 * alpha[5] * r13);
363             double exp7 = Math.Exp(-2.0 * alpha[6] * r14);
364             double exp8 = Math.Exp(-2.0 * alpha[7] * r23);
```

```
365             double exp9 = Math.Exp(-2.0 * alpha[8] * r24);
366             double expA = Math.Exp(-2.0 * alpha[9] * r34);
367
368             return N * N * term * exp1 * exp2 * exp3 * exp4 * exp5 *
369                 exp6 * exp7 * exp8 * exp9 * expA;
370         }
371
372         public double Integrand9(double[] x, double[] alpha)
373         {
374             double r1 = Math.Sqrt(x[0] * x[0] + x[1] * x[1] + x[2] * x[2]);
375             double r2 = Math.Sqrt(x[3] * x[3] + x[4] * x[4] + x[5] * x[5]);
376             double r12 = Math.Sqrt(Math.Pow(x[0] - x[3], 2.0) +
377                 Math.Pow(x[1] - x[4], 2.0) + Math.Pow(x[2] - x[5], 2.0));
378
379             if (r12 == 0)
380                 r12 = 0.01;
381
382             double term = 1.0 / r12;
383             double mul1 = Math.Exp(-2.0 * alpha[0] * r1);
384             double mul2 = Math.Exp(-2.0 * alpha[1] * r2);
385             double mul3 = Math.Exp(-2.0 * alpha[3] * r12);
386
387             return N * N * term * mul1 * mul2 * mul3;
388         }
389
390         public double IntegrandA(double[] x, double[] alpha)
391         {
392             double r1 = Math.Sqrt(x[0] * x[0] + x[1] * x[1] + x[2] * x[2]);
393             double r3 = Math.Sqrt(x[6] * x[6] + x[7] * x[7] + x[8] * x[8]);
394             double r13 = Math.Sqrt(Math.Pow(x[0] - x[6], 2.0) +
395                 Math.Pow(x[1] - x[7], 2.0) + Math.Pow(x[2] - x[8], 2.0));
396
397             if (r13 == 0)
398                 r13 = 0.01;
399
400             double term = 1.0 / r13;
401             double mul1 = Math.Exp(-2.0 * alpha[0] * r1);
402             double mul2 = Math.Exp(-2.0 * alpha[1] * r3);
403             double mul3 = Math.Exp(-2.0 * alpha[3] * r13);
404
405             return N * N * term * mul1 * mul2 * mul3;
406         }
407
408         public double IntegrandB(double[] x, double[] alpha)
409         {
410             double r1 = Math.Sqrt(x[0] * x[0] + x[1] * x[1] + x[2] * x[2]);
411             double r4 = Math.Sqrt(x[9] * x[9] + x[10] * x[10] + x[11] * x[11]);
412             double r14 = Math.Sqrt(Math.Pow(x[0] - x[9], 2.0) +
413                 Math.Pow(x[1] - x[10], 2.0) + Math.Pow(x[2] - x[11], 2.0));
414
415             if (r14 == 0)
416                 r14 = 0.01;
```

```
417
418             double term = 1.0 / r14;
419             double mul1 = Math.Exp(-2.0 * alpha[0] * r1);
420             double mul2 = Math.Exp(-2.0 * alpha[1] * r4);
421             double mul3 = Math.Exp(-2.0 * alpha[3] * r14);
422
423             return N * N * term * mul1 * mul2 * mul3;
424         }
425
426         public double IntegrandC(double[] x, double[] alpha)
427         {
428             double r2 = Math.Sqrt(x[3] * x[3] + x[4] * x[4] + x[5] * x[5]);
429             double r3 = Math.Sqrt(x[6] * x[6] + x[7] * x[7] + x[8] * x[8]);
430             double r23 = Math.Sqrt(Math.Pow(x[3] - x[6], 2.0) +
431                 Math.Pow(x[4] - x[7], 2.0) + Math.Pow(x[5] - x[8], 2.0));
432
433             if (r23 == 0)
434                 r23 = 0.01;
435
436             double term = 1.0 / r23;
437             double mul1 = Math.Exp(-2.0 * alpha[0] * r2);
438             double mul2 = Math.Exp(-2.0 * alpha[1] * r3);
439             double mul3 = Math.Exp(-2.0 * alpha[3] * r23);
440
441             return N * N * term * mul1 * mul2 * mul3;
442         }
443
444         public double IntegrandD(double[] x, double[] alpha)
445         {
446             double r2 = Math.Sqrt(x[3] * x[3] + x[4] * x[4] + x[5] * x[5]);
447             double r4 = Math.Sqrt(x[9] * x[9] + x[10] * x[10] + x[11] * x[11]);
448             double r24 = Math.Sqrt(Math.Pow(x[3] - x[9], 2.0) +
449                 Math.Pow(x[4] - x[10], 2.0) + Math.Pow(x[5] - x[11], 2.0));
450
451             if (r24 == 0)
452                 r24 = 0.01;
453
454             double term = 1.0 / r24;
455             double mul1 = Math.Exp(-2.0 * alpha[0] * r2);
456             double mul2 = Math.Exp(-2.0 * alpha[1] * r4);
457             double mul3 = Math.Exp(-2.0 * alpha[3] * r24);
458
459             return N * N * term * mul1 * mul2 * mul3;
460         }
461
462         public double IntegrandE(double[] x, double[] alpha)
463         {
464             double r3 = Math.Sqrt(x[6] * x[6] + x[7] * x[7] + x[8] * x[8]);
465             double r4 = Math.Sqrt(x[9] * x[9] + x[10] * x[10] + x[11] * x[11]);
466             double r34 = Math.Sqrt(Math.Pow(x[6] - x[9], 2.0) +
467                 Math.Pow(x[7] - x[10], 2.0) + Math.Pow(x[8] - x[11], 2.0));
468
```

```
469             if (r34 == 0)
470                 r34 = 0.01;
471
472             double term = 1.0 / r34;
473             double mul1 = Math.Exp(-2.0 * alpha[0] * r3);
474             double mul2 = Math.Exp(-2.0 * alpha[1] * r4);
475             double mul3 = Math.Exp(-2.0 * alpha[3] * r34);
476
477             return N * N * term * mul1 * mul2 * mul3;
478         }
479
480         public double Energy(double[] alpha, double beta, int nSteps, int Z)
481         {
482             double[] lower = new double[12];
483             double[] upper = new double[12];
484             int[] steps = new int[12];
485
486             lower[0] = lower[1] = lower[2] = lower[3] = lower[4] = lower[5] =
                  0.001;
487             lower[6] = lower[7] = lower[8] = lower[9] = lower[10] = lower[11] =
                  0.001;
488             upper[0] = upper[1] = upper[2] = upper[3] = upper[4] = upper[5] =
                  10.0;
489             upper[6] = upper[7] = upper[8] = upper[9] = upper[10] = upper[11] =
                  10.0;
490             steps[0] = steps[1] = steps[2] = steps[3] = steps[4] = steps[5] =
                  nSteps;
491             steps[6] = steps[7] = steps[8] = steps[9] = steps[10] = steps[11] =
                  nSteps;
492
493             N = Normalize(alpha, nSteps);
494
495             this.Z = Z;
496
497             integ1 = +integ.Integrate(lower, upper, alpha, Integrand1, 12,
                  steps);
498             integ2 = +integ.Integrate(lower, upper, alpha, Integrand2, 12,
                  steps);
499             integ3 = +integ.Integrate(lower, upper, alpha, Integrand3, 12,
                  steps);
500             integ4 = +integ.Integrate(lower, upper, alpha, Integrand4, 12,
                  steps);
501             integ5 = -integ.Integrate(lower, upper, alpha, Integrand5, 12,
                  steps);
502             integ6 = -integ.Integrate(lower, upper, alpha, Integrand6, 12,
                  steps);
503             integ7 = -integ.Integrate(lower, upper, alpha, Integrand7, 12,
                  steps);
504             integ8 = -integ.Integrate(lower, upper, alpha, Integrand8, 12,
                  steps);
505             integ9 = +integ.Integrate(lower, upper, alpha, Integrand9, 12,
                  steps);
```

```
506             integA = +integ.Integrate(lower, upper, alpha, IntegrandA, 12,
                  steps);
507             integB = +integ.Integrate(lower, upper, alpha, IntegrandB, 12,
                  steps);
508             integC = +integ.Integrate(lower, upper, alpha, IntegrandC, 12,
                  steps);
509             integD = +integ.Integrate(lower, upper, alpha, IntegrandD, 12,
                  steps);
510             integE = +integ.Integrate(lower, upper, alpha, IntegrandE, 12,
                  steps);
511
512             return (integ1 + integ2 + integ3 + integ4 + integ5 + integ6 + integ7
513                 + integ8 + beta * (integ9 + integA + integB + integC)) / (N * N);
514         }
515     }
516 }
```