

## Appendix 2

### Three-Body Potential Molecular Simulation Implementation

In this appendix we report an easy and correct way to implement the 3-body potential (namely, the Axilrod-Teller potential [Axi43]) in a molecular dynamics and Monte Carlo computer simulation program.

In the first section we show how to verify Newton's third law of dynamics with the 3-body potential, since in this way we are able to find useful expressions. In the second section we point out the problem arising using the minimum image convention with the 3-body potential, suggesting also the correct method to avoid mistakes. In the third section we write the expressions for the forces and the pressure, and in the last section we implement these expressions in an algorithm optimized for vector computers and designed to be fast enough to make the simulations feasible.

#### A2.1 Newton's third law of dynamics for 3-body potential

Considering three atoms  $i$ ,  $j$  and  $k$  the 3body Axilrod-Teller potential is (see Chapter 2):

$$u_{ijk} = \frac{v (1 + 3 \cos \mathbf{q}_i \cos \mathbf{q}_j \cos \mathbf{q}_k)}{(r_{ij} r_{ik} r_{jk})^3} \quad (\text{A2.1})$$

where  $v_{DDD}$  is a non-additive coefficient and where the angles and intermolecular separations

$$r_{ij}^2 = (x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2 \quad (\text{A2.2})$$

refer to a triangular configuration of atoms (see Figure 2.1). Using the cosine law:

$$r_{ij}^2 = r_{ik}^2 + r_{jk}^2 - 2r_{ik}r_{jk} \cos \mathbf{a} \quad \rightarrow \quad \cos \mathbf{a} = \frac{-r_{ij}^2 + r_{ik}^2 + r_{jk}^2}{2r_{ik}r_{jk}} \quad (\text{A2.3})$$

the potential in eq (A2.1) can be written as:

$$u_{ijk} = v \left[ \frac{1}{r_{ij}^3 r_{ik}^3 r_{jk}^3} + \frac{3(-r_{ij}^2 + r_{ik}^2 + r_{jk}^2)(r_{ij}^2 - r_{ik}^2 + r_{jk}^2)(r_{ij}^2 + r_{ik}^2 - r_{jk}^2)}{8r_{ij}^5 r_{ik}^5 r_{jk}^5} \right] \quad (\text{A2.4})$$

Expressing the potential as a function of the relative coordinates:

$$u(x_i; x_j; x_k; y_i; \dots) = u(x_i - x_j; x_i - x_k; x_j - x_k; y_i - y_j; \dots) = u(x_{ij}; x_{ik}; x_{jk}; y_{ij}; \dots) \quad (\text{A2.5})$$

the derivatives in the coordinates are easily obtained by:

$$F_{i(jk)}^{3b;x} \equiv -\frac{\partial u_{ijk}}{\partial x_i} = -\left[ \frac{\partial x_{ij}}{\partial x_i} \frac{\partial u_{ijk}}{\partial x_{ij}} + \frac{\partial x_{ik}}{\partial x_i} \frac{\partial u_{ijk}}{\partial x_{ik}} \right] = -\left[ \frac{\partial u_{ijk}}{\partial x_{ij}} + \frac{\partial u_{ijk}}{\partial x_{ik}} \right] = F_{i(j)k}^{3b;x} + F_{i(k)j}^{3b;x} \quad (\text{A2.6})$$

Here  $F_{i(jk)}^{3b;x}$  is the total 3-body force (due to atoms  $j$  and  $k$ ) on atom  $i$  in the  $x$  direction,

$F_{i(j)k}^{3b;x}$  is the contribution from atom  $j$  only. There are similar expressions for the other

coordinates:

$$\left. \begin{aligned} F_{j(ik)}^{3b;x} &\equiv -\frac{\partial u_{ijk}}{\partial x_j} = -\left[ \frac{\partial x_{ij}}{\partial x_j} \frac{\partial u_{ijk}}{\partial x_{ij}} + \frac{\partial x_{jk}}{\partial x_j} \frac{\partial u_{ijk}}{\partial x_{jk}} \right] = -\left[ -\frac{\partial u_{ijk}}{\partial x_{ij}} + \frac{\partial u_{ijk}}{\partial x_{jk}} \right] \\ F_{j(i)k}^{3b;x} + F_{j(k)i}^{3b;x} &= -F_{i(j)k}^{3b;x} + F_{j(k)i}^{3b;x} \end{aligned} \right\} \quad (\text{A2.7})$$

$$F_{k(ij)}^{3b;x} \equiv -\frac{\partial u_{ijk}}{\partial x_k} = -\left[ \frac{\partial x_{ik}}{\partial x_k} \frac{\partial u_{ijk}}{\partial x_{ik}} + \frac{\partial x_{jk}}{\partial x_k} \frac{\partial u_{ijk}}{\partial x_{jk}} \right] = -\left[ -\frac{\partial u_{ijk}}{\partial x_{ik}} - \frac{\partial u_{ijk}}{\partial x_{jk}} \right] = \left. \begin{aligned} F_{k(ij)}^{3b;x} + F_{k(ji)}^{3b;x} &= -F_{i(k)j}^{3b;x} - F_{j(k)i}^{3b;x} \end{aligned} \right\} . \quad (\text{A2.8})$$

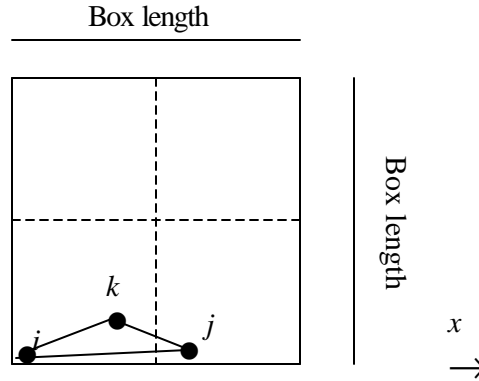
Using the previous relationships, it is possible to show that the total forces on the atoms  $i$  and  $j$  are equal and opposite to the total force on the atom  $k$ .

$$F_{i(jk)}^{3b;x} + F_{j(ik)}^{3b;x} = F_{i(j)k}^{3b;x} + F_{i(k)j}^{3b;x} + F_{j(i)k}^{3b;x} + F_{j(k)i}^{3b;x} = F_{i(k)j}^{3b;x} + F_{j(k)i}^{3b;x} = -F_{k(ij)}^{3b;x} . \quad (\text{A2.9})$$

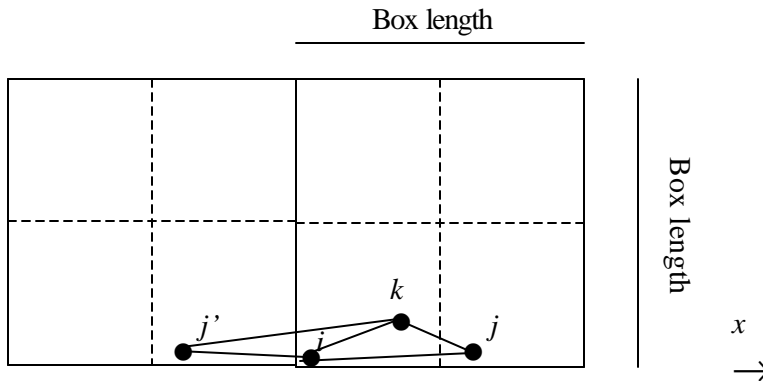
We point out that to obtain the previous result we have only used the fact that the 3-body potential is a function of the relative distances between the three atoms.

## A2.2 Three-body potential and minimum image convention

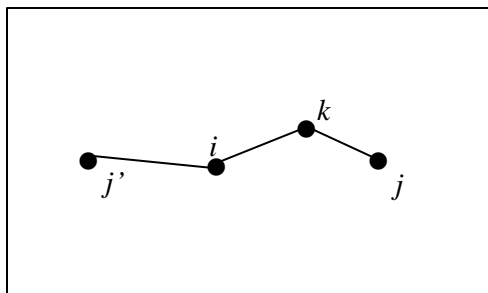
Given a triplet of atoms, applying the minimum image convention requires some care. In general the minimum image convention does not keep the ‘shape’ of the triangle. Let us analyze the  $ijk$  triplet in the following picture:



When we apply the minimum image convention on the length  $ij$ , we have to consider the imagine of  $j$  ( $j'$ ), because  $(x_i - x_j)$  is longer than half the box length; so we consider the side  $\overline{ij'}$ .



In the case of the side  $\overline{ik}$ , we do not have to consider the images because  $(x_i - x_k)$  is not longer than half the box length. So the triangle now should be  $ij'k$ , but when we consider the atoms  $j$  and  $k$ , we have to calculate the side  $\overline{jk}$  and not  $\overline{kj'}$ . So we have a “triangle” with the shape:



Using a cut-off for the 3body potential for which a triangle is accepted if each side is less than a quarter of the box length, avoids these undesirable situations. Even if this condition is very strict, we have tested that for simulations at liquid densities with 500 argon atoms, the relative cut-off guarantees a good accuracy for the Axillord-Teller potential ([Mar99], see also Chapter 3). It is worthwhile to point out that the condition for the cut-off works also with Lees-Edwards sliding brick boundary conditions ([Eva90], see also Chapter 2).

In his work [Att92] Attard recognized this problem, but suggested a different solution. He implemented a different minimum image algorithm to be applied only for triplets of atoms and he adopted a cut-off for the three-body potential smaller than half the box length, as is the case for the two-body potential. Even if this mathematically solves the problem, we can not attribute to it a clear physical meaning. In the case depicted in the first figure, all the pair interactions between the three atoms are calculated, since all the three sides are less than half of the box length, according to the traditional minimum image convention. On the other hand, when Attard's algorithm is applied to calculate the three-body interactions, the same triplet of atoms is rejected, since  $\overline{jk}$  is greater than half the box length. Furthermore, it is not clear if Attard's algorithm can be generalized for Lees-Edwards sliding brick boundary conditions.

Sometimes, a misleading condition for the three-body potential cut-off is adopted [Cor00]. The three-body force on atom  $i$  is considered different from zero if and only if both atoms  $j$  and  $k$  lie within the cut-off distance to atom  $i$ . No additional requirement is made on the distance between atoms  $j$  and  $k$ . This leads to non-symmetric situations. Let us consider the case where the distances between  $i$  and  $j$  and  $i$  and  $k$  are less than the cut-off, and the distance between  $j$  and  $k$  is greater than the cut-off. The three-body force on atom  $i$  is not zero. For atoms  $j$  and  $k$  the three-body force is zero. This clearly violates Newton's third law of dynamics, as expressed in Eq. (A2.9), since the total three-body force on the triplet of atoms is not zero.

### A2.3 Forces and pressure for 3-body potential

In general, the  $\mathbf{ab}$  ( $\mathbf{a}, \mathbf{b} = x, y, z$ ) component of the configurational pressure tensor is usually defined by [Ala87]:

$$P_{ab}V = \sum_i \mathbf{a}_i F_i^b = - \sum_i \mathbf{a}_i \frac{\partial u_{tot}}{\partial \mathbf{b}_i} \quad (\text{A2.10})$$

Considering for simplicity the  $xx$  component, it is possible to write:

$$\sum_i x_i F_i^x = \sum_i \sum_{j>i} x_{ij} F_{ij}^x \quad (\text{A2.11})$$

where:

$$\left. \begin{aligned} x_{ij} &= x_i - x_j \\ F_{ij}^x &= - \frac{\partial u_{ij}}{\partial x_{ij}} \end{aligned} \right\} \quad (\text{A2.12})$$

The left-hand-side of Eq. (A2.11) is not suitable to be used with the minimum image convention, since in general it changes the value of  $x_{ij}$ ; we are forced to work with the second term of the relation (A2.11) [All87], i.e. with expressions which contain  $r_{ij}$  rather than  $r_i$ . In particular we also have to do this with the 3-body potential.

A similar relation holds for the 3-body case:

$$P_{xx}^{3b}V = \sum_i x_i F_i^{3b;x} = \sum_i \sum_{j \neq i} \sum_{k \neq j} x_i F_{i(jk)}^{3b;x} \quad (\text{A2.13})$$

We can also write:

$$\sum_i x_i F_i^{3b;x} = \frac{1}{3} \sum_i \sum_{j \neq i} \sum_{k \neq j} \left( x_i F_{i(jk)}^{3b;x} + x_j F_{j(ik)}^{3b;x} + x_k F_{k(ij)}^{3b;x} \right) \quad (\text{A2.14})$$

since the indices  $i, j, k$  are equivalent.

Furthermore,

$$\left. \begin{aligned} \sum_i x_i F_i^{3b;x} &= \frac{1}{3} \sum_i \sum_{j \neq i} \sum_{k \neq j} \left\{ x_i \left[ F_{i(j)k}^{3b;x} + F_{i(k)j}^{3b;x} \right] + \right. \\ &\left. x_j \left[ F_{j(i)k}^{3b;x} + F_{j(k)i}^{3b;x} \right] + x_k \left[ F_{k(i)j}^{3b;x} + F_{k(j)i}^{3b;x} \right] \right\} \end{aligned} \right\} \quad (\text{A2.15})$$

Here we write  $F_{i(jk)}^{3b;x} = F_{i(j)k}^{3b;x} + F_{i(k)j}^{3b;x}$ , and similarly for the other triplet terms.

Substituting Eqs. (A2.7) and (A2.8) in Eq. (A2.15) gives:

$$\left. \begin{aligned} \sum_i x_i F_i^{3b;x} &= \frac{1}{3} \sum_i \sum_{j \neq i} \sum_{k \neq j} \left\{ x_i \left[ F_{i(j)k}^{3b;x} + F_{i(k)j}^{3b;x} \right] + \right. \\ &\left. x_j \left[ -F_{i(j)k}^{3b;x} + F_{j(k)i}^{3b;x} \right] + x_k \left[ -F_{i(k)j}^{3b;x} - F_{j(k)i}^{3b;x} \right] \right\} \end{aligned} \right\} \quad (\text{A2.16})$$

So:

$$\sum_i x_i F_i^{3b;x} = \frac{1}{3} \sum_i \sum_{j \neq i} \sum_{k \neq j} \left[ (x_i - x_j) F_{i(j)k}^{3b;x} + (x_i - x_k) F_{i(k)j}^{3b;x} + (x_j - x_k) F_{j(k)i}^{3b;x} \right] \quad (\text{A2.17})$$

or:

$$\left. \begin{aligned} \sum_i x_i F_i^{3b;x} &= \frac{1}{3} \sum_i \sum_{j \neq i} \sum_{k \neq j} \left( x_{ij} F_{i(j)k}^{3b;x} + x_{ik} F_{i(k)j}^{3b;x} + x_{jk} F_{j(k)i}^{3b;x} \right) = \\ &\sum_i \sum_{j > i} \sum_{k > j} \left( x_{ij} F_{i(j)k}^{3b;x} + x_{ik} F_{i(k)j}^{3b;x} + x_{jk} F_{j(k)i}^{3b;x} \right) \end{aligned} \right\} \quad (\text{A2.18})$$

Since  $F_{i(j)k}^{3b;x} = -\frac{\partial u_{ijk}}{\partial x_{ij}}$  (and similarly for all the other terms), we achieve the goal of

expressing the pressure as a function of pair distances. Now we need to find an

expression for  $\frac{\partial u_{ijk}}{\partial x_{ij}}$ . The potential is:

$$\left. \begin{aligned}
 u_{ijk} &= v \left[ \frac{1}{r_{ij}^3 r_{ik}^3 r_{jk}^3} + \frac{3(-r_{ij}^2 + r_{ik}^2 + r_{jk}^2)(r_{ij}^2 - r_{ik}^2 + r_{jk}^2)(r_{ij}^2 + r_{ik}^2 - r_{jk}^2)}{8r_{ij}^5 r_{ik}^5 r_{jk}^5} \right] = \\
 n \left[ \frac{1}{r_{ij}^3 r_{ik}^3 r_{jk}^3} + \right. & \\
 \left. \frac{-r_{ij}^6 - r_{ik}^6 - r_{jk}^6 + r_{ij}^4 r_{ik}^2 + r_{ij}^4 r_{jk}^2 + r_{ij}^2 r_{ik}^4 + r_{ij}^2 r_{jk}^4 + r_{ik}^4 r_{jk}^2 + r_{ik}^2 r_{jk}^4 - 2r_{ij}^2 r_{ik}^2 r_{jk}^2}{8r_{ij}^5 r_{ik}^5 r_{jk}^5} \right] &
 \end{aligned} \right\} \quad . \quad (\text{A2.19})$$

So:

$$\left. \begin{aligned}
 \frac{\partial u_{ijk}}{\partial x_{ij}} &= \frac{\partial r_{ij}}{\partial x_{ij}} \frac{\partial u_{ijk}}{\partial r_{ij}} = \frac{x_{ij}}{r_{ij}} \frac{\partial u_{ijk}}{\partial r_{ij}} = \frac{x_{ij}}{r_{ij}} v \left[ \frac{-3}{r_{ij}^4 r_{ik}^3 r_{jk}^3} - \frac{3}{8r_{ik}^5 r_{jk}^5} + 3 \frac{5r_{ik}}{8r_{ij}^6 r_{jk}^5} + \right. \\
 3 \frac{5r_{jk}}{8r_{ij}^6 r_{ik}^5} &- \frac{3}{8r_{ij}^2 r_{ik}^3 r_{jk}^5} - \frac{3}{8r_{ij}^2 r_{ik}^5 r_{jk}^3} \\
 \left. - 3 \frac{3}{8r_{ij}^4 r_{ik}^5 r_{jk}^5} - 3 \frac{3}{8r_{ij}^4 r_{ik}^5 r_{jk}^5} - 3 \frac{5}{8r_{ij}^6 r_{ik}^3 r_{jk}^3} - 3 \frac{5}{8r_{ij}^6 r_{ik}^3 r_{jk}^3} + 3 \frac{6}{8r_{ij}^4 r_{ik}^3 r_{jk}^3} \right] &
 \end{aligned} \right\} \quad . \quad (\text{A2.20})$$

Similar expressions exist for all other derivatives of the triplet potential. This expression is ‘easy’ to implement in a program, as we report in the following section.

## A2.4 Algorithm

In what follows we report an algorithm to implement the 3-body potential for a system of  $n$  atoms. We optimize the algorithm in order to take advantage of the vectorisation, hence this algorithm is not suitable for a parallel computer.

As a first step we have to calculate the distances between all the pair of atoms, and apply the minimum image convention (note that this loop is not time consuming in comparison with the 3-body algorithm):

```

loop  $i \leftarrow 1, N-1$ 
  loop  $j \leftarrow i+1, N$ 

    // coordAtomX(i), coordAtomY(i) and coordAtomZ(i) are the coordinates of atom i.
    // distanceAtomsX, distanceAtomsY and distanceAtomsZ are the distances between two
    // atoms in the x, y, z directions respectively.

    distanceAtomsX  $\leftarrow$  coordAtomX(i) - coordAtomX(j)
    distanceAtomsY  $\leftarrow$  coordAtomY(i) - coordAtomY(j)
    distanceAtomsZ  $\leftarrow$  coordAtomZ(i) - coordAtomZ(j)

    // Minimum image convention [All87]. boxLength is the box length. The
    // efficiency of different algorithms for the implementation of the minimum image
    // convention is studied in the work of Hloucha and Deiters [Hlo97].

    distanceAtomsX  $\leftarrow$  distanceAtomsX - boxLength * NINT(distanceAtomsX / boxLength)
    distanceAtomsY  $\leftarrow$  distanceAtomsY - boxLength * NINT(distanceAtomsY / boxLength)
    distanceAtomsZ  $\leftarrow$  distanceAtomsZ - boxLength * NINT(distanceAtomsZ / boxLength)

    // distanceAtoms1(i,j), distanceAtoms2(i,j), ... distanceAtoms6(i,j) are arrays where the
    // distances between atoms (and their respective powers) are stored. These arrays
    // should be symmetrised, ( $d(i,j)=d(j,i)$ ), but in what follows it is not necessary.

    distanceAtoms2(i,j)  $\leftarrow$  distanceAtomsX**2 + distanceAtomsY**2 +
      distanceAtomsZ**2
    distanceAtoms1(i,j)  $\leftarrow$  SQRT(distanceAtoms2(i,j))
    distanceAtoms3(i,j)  $\leftarrow$  distanceAtoms2(i,j)*distanceAtoms1(i,j)
    distanceAtoms4(i,j)  $\leftarrow$  distanceAtoms2(i,j)*distanceAtoms2(i,j)
    distanceAtoms5(i,j)  $\leftarrow$  distanceAtoms2(i,j)*distanceAtoms3(i,j)
    distanceAtoms6(i,j)  $\leftarrow$  distanceAtoms2(i,j)*distanceAtoms4(i,j)

    // x(i,j), y(i,j) and z(i,j) are arrays to store the relative distances in the x, y, z directions.

    x(i,j)  $\leftarrow$  distanceAtomsX
    y(i,j)  $\leftarrow$  distanceAtomsY
    z(i,j)  $\leftarrow$  distanceAtomsZ

  end j loop
end i loop

```

As usual, a double loop can be used at this stage to calculate the 2body potential and forces:

```
loop  $i \leftarrow 1, N-1$   
  loop  $j \leftarrow i+1, N$   
  
  // Calculation of two-body potential and forces  
  
  end j loop  
end i loop
```

This loop is not time consuming in comparison with the 3-body algorithm; note that we do not use a neighbor list, since it could be complicated to implement with a 3body potential and because it would probably compromise the vectorisation.

Before the algorithm for the 3-body terms is implemented, some variables have to be initialized:

```
// total3BodyEnergy is the total 3-body energy. total3BodyForceX(i),  
// total3BodyForceY(i), total3BodyForceZ(i) are the total forces on the atom  $i$  in the  $x$ ,  
//  $y$ ,  $z$  directions. pressureTensor3body(1), ..., pressureTensor3body(6) are the  $xx$ ,  $xy$ ,  $xz$ ,  
//  $yy$ ,  $yz$ ,  $zz$  elements of the 3-body pressure tensor. The hydrostatic pressure is 1/3 of the  
// pressure tensor's trace.
```

```
total3BodyEnergy  $\leftarrow 0.0$ 
```

```
loop  $i \leftarrow 1, N$ 
```

```
  total3BodyForceX(i)  $\leftarrow 0.0$ 
```

```
  total3BodyForceY(i)  $\leftarrow 0.0$ 
```

```
  total3BodyForceZ(i)  $\leftarrow 0.0$ 
```

```
end i loop
```

```
loop  $i \leftarrow 1, 6$ 
```

```
  pressureTensor3body(i)  $\leftarrow 0.0$ 
```

```
end i loop
```

Now we have to apply the cut-off condition to know which triplets of atoms (triangles) can be counted. To do that we use the usual triple-loop:

```
// d1a(lc), d2a(lc),...d6a(lc) are arrays to store the first side (and powers) of the
// lc-th accepted triangle. d1b(lc), d2b(lc),...d6b(lc) are arrays to store the second
// side (and powers) of the lc-th accepted triangle. d1c(lc), d2c(lc),...d6c(lc) are
// arrays to store the third side (and powers) of the lc-th accepted triangle. dXa(lc),
// dYa(lc), dZa(lc) are arrays to store the relatives coordinates of the first side of
// the lc-th accepted triangle.
```

```
lc←0.0
```

```
loop i←1,N-2
```

```
loop j←i+1,N-1
```

```
loop k←j+1,N
```

```
if (distanceAtoms1(i,j) < boxLength/4 .and.
distanceAtoms1(i,k) < boxLength/4 .and.
distanceAtoms1(j,k) < boxLength/4)
```

```
// This is the cut-off condition: a triangle is accepted if each of its sides is less than
// a quarter of the box length.
```

```
lc ← lc+1
```

```
d1a(lc) ←distanceAtoms1(i,j)
```

```
d2a(lc) ←distanceAtoms2(i,j)
```

```
d3a(lc) ←distanceAtoms3(i,j)
```

```
d4a(lc) ←distanceAtoms4(i,j)
```

```
d5a(lc) ←distanceAtoms5(i,j)
```

```
d6a(lc) ←distanceAtoms6(i,j)
```

```
dXa(lc) ←x(i,j)
```

```
dYa(lc) ←y(i,j)
```

```
dZa(lc) ←z(i,j)
```

```
d1b(lc) ←distanceAtoms1(i,k)
```

```
d2b(lc) ←distanceAtoms2(i,k)
```

```
d3b(lc) ←distanceAtoms3(i,k)
```

```
d4b(lc) ←distanceAtoms4(i,k)
```

```
d5b(lc) ←distanceAtoms5(i,k)
```

```
d6b(lc) ←distanceAtoms6(i,k)
```

```
dXb(lc) ←x(i,k)
```

```
dYb(lc) ←y(i,k)
```

```
dZb(lc) ←z(i,k)
```

```

d1c(lc) ← distanceAtoms1(j,k)
d2c(lc) ← distanceAtoms2(j,k)
d3c(lc) ← distanceAtoms3(j,k)
d4c(lc) ← distanceAtoms4(j,k)
d5c(lc) ← distanceAtoms5(j,k)
d6c(lc) ← distanceAtoms6(j,k)

```

```

dXc(lc) ← x(j,k)
dYc(lc) ← y(j,k)
dZc(lc) ← z(j,k)

```

```

// l1(lc), l2(lc), l3(lc) are integer arrays to store the indices of the 3 atoms in the
// lc-th triangle; these arrays will be used to calculate the forces.

```

```

l1(lc) ← i
l2(lc) ← j
l3(lc) ← k

```

**end if**

**end k loop**

**end j loop**

**end i loop**

The next loop will be a ‘long’ loop over the number of all accepted triangles to calculate energy, forces and pressure. This loop speeds up the program, since it can be vectorised more intensely than a normal triple loop.

**loop**  $l \rightarrow 1,lc$

//  $dVdRa$ ,  $dVdRb$ ,  $dVdRc$  are  $\frac{\mathbf{n}}{r_{ij}} \frac{\partial u}{\partial r_{ij}}$ ,  $\frac{\mathbf{n}}{r_{ik}} \frac{\partial u}{\partial r_{ik}}$ ,  $\frac{\mathbf{n}}{r_{jk}} \frac{\partial u}{\partial r_{jk}}$  respectively (see section A2.3).

// *nonAdditiveCoef* is the non-additive coefficient  $\mathbf{n}$ .

```

dVdRa ← (3.*nonAdditiveCoef/(8.*d1a(l)))*
-8./(d4a(l)*d3b(l)*d3c(l))-1./(d5b(l)*d5c(l))
+5.*d1b(l)/(d6a(l)*d5c(l))+5.*d1c(l)/(d6a(l)*d5b(l))
-1./(d2a(l)*d3b(l)*d5c(l))-1./(d2a(l)*d5b(l)*d3c(l))
-3./(d4a(l)*d1b(l)*d5c(l))-3./(d4a(l)*d5b(l)*d1c(l))
-5./(d6a(l)*d1b(l)*d3c(l))-5./(d6a(l)*d3b(l)*d1c(l))
+6./(d4a(l)*d3b(l)*d3c(l))

```

$$dVdRb \leftarrow (3. *nonAdditiveCoef/(8. *d1b(l))) * ($$

$$-8./(d4b(l)*d3a(l)*d3c(l))-1./(d5a(l)*d5c(l))$$

$$+5.*d1a(l)/(d6b(l)*d5c(l))+5.*d1c(l)/(d6b(l)*d5a(l))$$

$$-1./(d2b(l)*d3a(l)*d5c(l))-1./(d2b(l)*d5a(l)*d3c(l))$$

$$-3./(d4b(l)*d1a(l)*d5c(l))-3./(d4b(l)*d5a(l)*d1c(l))$$

$$-5./(d6b(l)*d1a(l)*d3c(l))-5./(d6b(l)*d3a(l)*d1c(l))$$

$$+6./(d4b(l)*d3a(l)*d3c(l)) )$$

$$dVdRc \leftarrow (3. *nonAdditiveCoef/(8. *d1c(l))) * ($$

$$-8./(d4c(l)*d3b(l)*d3a(l))-1./(d5b(l)*d5a(l))$$

$$+5.*d1b(l)/(d6c(l)*d5a(l))+5.*d1a(l)/(d6c(l)*d5b(l))$$

$$-1./(d2c(l)*d3b(l)*d5a(l))-1./(d2c(l)*d5b(l)*d3a(l))$$

$$-3./(d4c(l)*d1b(l)*d5a(l))-3./(d4c(l)*d5b(l)*d1a(l))$$

$$-5./(d6c(l)*d1b(l)*d3a(l))-5./(d6c(l)*d3b(l)*d1a(l))$$

$$+6./(d4c(l)*d3b(l)*d3a(l)) )$$

// This is the calculation of the forces:

// total3BodyForceX(l1(l)) is the force on the atom  $i$  in the  $X$  direction.

// total3BodyForceX(l2(l)) is the force on the atom  $j$  in the  $X$  direction.

// total3BodyForceX(l3(l)) is the force on the atom  $k$  in the  $X$  direction.

$$total3BodyForceX(l1(l)) \leftarrow total3BodyForceX(l1(l))-dXa(l)*dVdRa-dXb(l)*dVdRb$$

$$total3BodyForceY(l1(l)) \leftarrow total3BodyForceY(l1(l))-dYa(l)*dVdRa-dYb(l)*dVdRb$$

$$total3BodyForceZ(l1(l)) \leftarrow total3BodyForceZ(l1(l))-dZa(l)*dVdRa-dZb(l)*dVdRb$$

$$total3BodyForceX(l2(l)) \leftarrow total3BodyForceX(l2(l))-dXa(l)*(-dVdRa)-dXc(l)*dVdRc$$

$$total3BodyForceY(l2(l)) \leftarrow total3BodyForceY(l2(l))-dYa(l)*(-dVdRa)-dYc(l)*dVdRc$$

$$total3BodyForceZ(l2(l)) \leftarrow total3BodyForceZ(l2(l))-dZa(l)*(-dVdRa)-dZc(l)*dVdRc$$

$$total3BodyForceX(l3(l)) \leftarrow total3BodyForceX(l3(l))-dXb(l)*(-dVdRb)-dXc(l)*(-dVdRc)$$

$$total3BodyForceY(l3(l)) \leftarrow total3BodyForceY(l3(l))-dYb(l)*(-dVdRb)-dYc(l)*(-dVdRc)$$

$$total3BodyForceZ(l3(l)) \leftarrow total3BodyForceZ(l3(l))-dZb(l)*(-dVdRb)-dZc(l)*(-dVdRc)$$

// Note, it is accumulating all the contributions of the same atom from all triangles

// where the atom is present).

// Calculation of the elements of the 3-body pressure tensor.

$$pressureTensor3body(1) \leftarrow pressureTensor3body(1)$$

$$-dXa(l)*dXa(l)*dVdRa -dXb(l)*dXb(l)*dVdRb -dXc(l)*dXc(l)*dVdRc$$

$$pressureTensor3body(2) \leftarrow pressureTensor3body(2)$$

$$-dXa(l)*dYa(l)*dVdRa -dXb(l)*dYb(l)*dVdRb -dXc(l)*dYc(l)*dVdRc$$

$$pressureTensor3body(3) \leftarrow pressureTensor3body(3)$$

$$-dXa(l)*dZa(l)*dVdRa -dXb(l)*dZb(l)*dVdRb -dXc(l)*dZc(l)*dVdRc$$

$$pressureTensor3body(4) \leftarrow pressureTensor3body(4)$$

$$-dYa(l)*dYa(l)*dVdRa -dYb(l)*dYb(l)*dVdRb -dYc(l)*dYc(l)*dVdRc$$

$$pressureTensor3body(5) \leftarrow pressureTensor3body(5)$$

$$-dYa(l)*dZa(l)*dVdRa -dYb(l)*dZb(l)*dVdRb -dYc(l)*dZc(l)*dVdRc$$

```

pressureTensor3body(6) ← pressureTensor3body(6)
      -dZa(l)*dZa(l)*dVdRa -dZb(l)*dZb(l)*dVdRb -dZc(l)*dZc(l)*dVdRc

// Calculation of the 3-body energy.

total3BodyEnergy ← total3BodyEnergy + nonAdditiveCoef*(1.0/(d3c(l)*d3b(l)*d3a(l))
      +(3.0*(d2c(l)+d2b(l)-d2a(l))*(d2c(l)-d2b(l)+d2a(l))
      *(-d2c(l)+d2b(l)+d2a(l))/(8.*d5c(l)*d5b(l)*d5a(l)))

end l loop

```

To check the validity of the previous calculations it is worthwhile to verify the relationship:

$$\frac{3u_{tot}^{3b}}{V} = \frac{1}{3} \left( P_{xx}^{3b} + P_{yy}^{3b} + P_{zz}^{3b} \right) \quad (\text{A2.21})$$

which holds since the Axilrod-Teller potential is a homogenous function of degree -9 in the variables  $r_{ij}$ ,  $r_{ik}$  and  $r_{jk}$  [Bar71]. We did this test to check our program, and Eq. (A2.21) turned out to be verified.

Using a NEC SX-4 supercomputer we found that this algorithm used for molecular dynamics simulations with 500 particles makes the program 10 times faster than a program using normal triple-loops.