```
> restart; assume(a,'positive'); with(LinearAlgebra);
```

[&x, Add, Adjoint, BackwardSubstitute, BandMatrix, Basis, BezoutMatrix, BidiagonalForm, BilinearForm, CARE, CharacteristicMatrix, CharacteristicPolynomial, Column, ColumnDimension, ColumnOperation, ColumnSpace, CompanionMatrix, CompressedSparseForm, ConditionNumber, ConstantMatrix, ConstantVector, Copy, CreatePermutation, CrossProduct, DARE, DeleteColumn, DeleteRow, Determinant, Diagonal, DiagonalMatrix, Dimension, Dimensions, DotProduct, EigenConditionNumbers, Eigenvalues, Eigenvectors, Equal, ForwardSubstitute, FrobeniusForm, FromCompressedSparseForm, FromSplitForm, GaussianElimination, GenerateEquations, GenerateMatrix, Generic, GetResultDataType, GetResultShape, GivensRotationMatrix, GramSchmidt, HankelMatrix, HermiteForm, HermitianTranspose, HessenbergForm, HilbertMatrix, HouseholderMatrix, IdentityMatrix, IntersectionBasis, IsDefinite, IsOrthogonal, IsSimilar, IsUnitary, JordanBlockMatrix, JordanForm, KroneckerProduct, LA_Main, LUDecomposition, LeastSquares, LinearSolve, LyapunovSolve, Map, Map2, MatrixAdd, MatrixExponential, MatrixFunction, MatrixInverse, MatrixMatrixMultiply, MatrixNorm, MatrixPower, MatrixScalarMultiply, MatrixVectorMultiply, MinimalPolynomial, Minor, Modular, Multiply, NoUserValue, Norm, Normalize, NullSpace, OuterProductMatrix, Permanent, Pivot, PopovForm, ProjectionMatrix, QRDecomposition, RandomMatrix, RandomVector, Rank, RationalCanonicalForm, ReducedRowEchelonForm, Row, RowDimension, RowOperation, RowSpace, ScalarMatrix, ScalarMultiply, ScalarVector, SchurForm, SingularValues, SmithForm, SplitForm, StronglyConnectedBlocks, SubMatrix, SubVector, SumBasis, SylvesterMatrix, SylvesterSolve, ToeplitzMatrix, Trace, Transpose, TridiagonalForm, UnitVector, VandermondeMatrix, VectorAdd, VectorAngle, VectorMatrixMultiply, VectorNorm, VectorScalarMultiply, ZeroMatrix, ZeroVector, Zip] **(1)**

Define lattice translation

```
> T1 := Vector(3, [a, 0, 0]); T2 := Vector(3, [0, a, 0]); T3 :=
      Vector(3, [0, 0, a]);
```

$$T1 := \begin{bmatrix} a\sim \\ 0 \\ 0 \end{bmatrix}$$

$$T2 := \begin{bmatrix} 0 \\ a\sim \\ 0 \end{bmatrix}$$

$$T3 := \begin{bmatrix} 0 \\ 0 \\ a\sim \end{bmatrix} \quad \textbf{(2)}$$

Define reciprocal lattice translation

> $G1 := Vector\left(3, \left[\frac{2\cdot \mathrm{Pi}}{a}, 0, 0\right]\right)$; $G2 := Vector\left(3, \left[0, \frac{2\cdot \mathrm{Pi}}{a}, 0\right]\right)$;
$G3 := Vector\left(3, \left[0, 0, \frac{2\cdot \mathrm{Pi}}{a}\right]\right)$;

$$G1 := \begin{bmatrix} \frac{2\,\pi}{a\sim} \\ 0 \\ 0 \end{bmatrix}$$

$$G2 := \begin{bmatrix} 0 \\ \frac{2\,\pi}{a\sim} \\ 0 \end{bmatrix}$$

$$G3 := \begin{bmatrix} 0 \\ 0 \\ \frac{2\,\pi}{a\sim} \end{bmatrix} \quad (3)$$

> tau := $Vector(3, [0.5\cdot a, 0.5\cdot a, 0.5\cdot a])$;

$$\tau := \begin{bmatrix} 0.5\,a\sim \\ 0.5\,a\sim \\ 0.5\,a\sim \end{bmatrix} \quad (4)$$

>

> $\eta := \frac{4}{a^2}$; $\Omega := a^3$; $con1 := \frac{4\cdot \mathrm{Pi}}{\mathrm{Omega}}$; $con2 := \mathrm{sqrt}\left(\frac{\eta}{\mathrm{Pi}}\right)$;

$$\eta := \frac{4}{a\sim^2}$$

$$\Omega := a\sim^3$$

$$con1 := \frac{4\,\pi}{a\sim^3}$$

$$con2 := \frac{2}{a\sim\sqrt{\pi}} \quad (5)$$

Initial terms -- Cl-Cl **and** *Cs-Cs*

>

```
> tot := -evalf(con2·2);
```

$$tot := -\frac{2.256758334}{a\sim} \tag{6}$$

```
> for n from -8 by 1 while n < 8 do  for m from -8 by 1
  while m < 8 do  for l from -8 by 1 while l < 8
  do  if (n ≠ 0 or m ≠ 0 or l ≠ 0) then g :=
  (n·G1 + m·G2 + l·G3) ;  tot := tot + evalf( 2·con1·(1
  -exp(-I·DotProduct(g, τ))) · exp(-DotProduct(g, g)/eta) / DotProduct(g, g) )
  end if end do end do end do; evalf(tot);
```

$$-\frac{2.256758334}{a\sim} + \frac{0.0003951360355 + 1.907345200\,10^{-24}\,I}{a\sim} \tag{7}$$

```
> for n from -8 by 1 while n < 8 do  for m from -8 by 1
  while m < 8 do  for l from -8 by 1 while l < 8
  do  t := (n·T1 + m·T2 + l·T3) ;
  tot := tot
  -evalf( 2·(erfc(sqrt(eta)/2 ·VectorNorm(τ + t, 2))) / VectorNorm(tau + t, 2) ) ;
  if (n ≠ 0 or m ≠ 0 or l ≠ 0)
  then  tot := tot
  + evalf( 2·erfc(sqrt(eta)/2 ·VectorNorm(t, 2)) / VectorNorm(t, 2) ) end if end do
  end do end do; evalf(tot);
```

$$-\frac{4.071118106}{a\sim} + \frac{0.0003951360355 + 1.907345200\,10^{-24}\,I}{a\sim} \tag{8}$$

```
> Re(%);
```

$$-\frac{4.070722970}{a\sim} \tag{9}$$

```
> η := 8/a^2; Ω := a^3; con1 := 4·Pi/Omega; con2 := sqrt(η/Pi);
```

$$\eta := \frac{8}{a\sim^2}$$

$$\Omega := a\sim^3$$

$$con1 := \frac{4\,\pi}{a\sim^3}$$

$$con2 := \frac{2\sqrt{2}}{a\sim\sqrt{\pi}} \tag{10}$$

Initial terms -- Cl-Cl **and** *Cs-Cs*

>

> $tot := -evalf(con2 \cdot 2);$

$$tot := -\frac{3.191538242}{a\sim} \tag{11}$$

> **for** n **from** -8 **by** 1 **while** $n < 8$ **do** **for** m **from** -8 **by** 1 **while** $m < 8$ **do** **for** l **from** -8 **by** 1 ***while*** $l < 8$ ***do*** **if** $(n \neq 0$ **or** $m \neq 0$ **or** $l \neq 0)$ **then** $g :=$ $(n \cdot G1 + m \cdot G2 + l \cdot G3)$; $tot := tot + evalf\left(2 \cdot con1 \cdot (1 - \exp(-I \cdot DotProduct(g, \tau))) \cdot \frac{\exp\left(-\frac{DotProduct(g, g)}{eta}\right)}{DotProduct(g, g)}\right)$ **end if** **end do** **end do** **end do**; $evalf(tot);$

$$-\frac{3.191538242}{a\sim} + \frac{0.05494320470 - 3.590131459\,10^{-22}\,I}{a\sim} \tag{12}$$

> **for** n **from** -8 **by** 1 **while** $n < 8$ **do** **for** m **from** -8 **by** 1 **while** $m < 8$ **do** **for** l **from** -8 **by** 1 ***while*** $l < 8$ ***do*** $t := (n \cdot T1 + m \cdot T2 + l \cdot T3)$;

$$tot := tot - evalf\left(\frac{2 \cdot \left(erfc\left(\frac{\sqrt{eta}}{2} \cdot VectorNorm(\tau + t, 2)\right)\right)}{VectorNorm(\tau + t, 2)}\right);$$

if $(n \neq 0$ **or** $m \neq 0$ **or** $l \neq 0)$ **then** $tot := tot$

$$+ evalf\left(\frac{2\cdot erfc\left(\frac{sqrt(eta)}{2}\cdot VectorNorm(t,\,2)\right)}{VectorNorm(t,\,2)}\right)$$ **end if end do**

end do end do; $evalf\,(tot)$;

$$-\frac{4.125666244}{a\sim}+\frac{0.05494320470-3.590131459\,10^{-22}\,I}{a\sim} \tag{13}$$

> Re(%);

$$-\frac{4.070723039}{a\sim} \tag{14}$$

>