SUPPLEMENTARY MATERIAL

GAEAM: A New Package for Optimizing the Embedded Atom Method Potentials of Solids by using the Genetic Algorithm

Yang-Yang Zhang, [a,b]* Yu Cheng,[a,b] Shu-Wen Zhang [a,b]

[a] Fundamental Science Center of Rare Earths, Ganjiang Innovation Academy, Chinese Academy of Sciences, Ganzhou, 341000 Jiangxi, China

[b] School of Rare Earths, University of Science and technology of China, Hefei 230026, China

* Corresponding author: yyzhang@gia.cas.cn

Table of Contents

Part A. SI for the “Computational details”

```python
def makeFunc(a, b, r_e, c):
    # Creates functions of the form used for density function.
    # Functional form also forms components of pair potential.
    def func(r):
        return (a * math.exp(-b*(r/r_e - 1)))/(1+(r/r_e - c)**20.0)
    return func

def makePairPotAA(A, gamma, r_e, kappa,
                  B, omega, lamda):
    # Function factory that returns functions parameterised for homogeneous pair interactions
    f1 = makeFunc(A, gamma, r_e, kappa)
    f2 = makeFunc(B, omega, r_e, lamda)

    def func(r):
        return f1(r) - f2(r)
    return func

def makePairPotAB(dens_a, phi_aa, dens_b, phi_bb):
    # Function factory that returns functions parameterised for heterogeneous pair interactions
    def func(r):
        return 0.5 * ((dens_b(r)/dens_a(r) * phi_aa(r)) + (dens_a(r)/dens_b(r) * phi_bb(r)))
    return func

def makeEmbed(rho_e, rho_s, F_ni, F_i, F_e, eta):
    # Function factory returning parameterised embedding function.
    rho_n = 0.85*rho_e
    rho_0 = 1.15*rho_e

    def e1(rho):
        return sum([F_ni[i] * (rho/rho_n - 1)**float(i) for i in range(4)])
```

```python
class GeneticAlgorithm:
    """Genetic Algorithm for global optimization of multiple parameters"""

    def __init__(self, objective_func, param_ranges, pop_size=50, generations=100,
                 mutation_rate=0.01, crossover_rate=0.8, elitism_ratio=0.1):
        """
        Initialize genetic algorithm parameters

        Parameters:
            objective_func: Objective function, takes a list of parameters and returns a fitness value
            param_ranges: List of parameter ranges, each element is a tuple (min, max, is_integer)
                          is_integer=True means the parameter is an integer, False means floating-point number
            pop_size: Population size
            generations: Number of evolutionary generations
            mutation_rate: Mutation probability
            crossover_rate: Crossover probability
            elitism_ratio: Elitism retention ratio
        """
        self.objective_func = objective_func
        self.param_ranges = param_ranges
        self.num_params = len(param_ranges)
        self.pop_size = pop_size
        self.generations = generations
        self.mutation_rate = mutation_rate
        self.crossover_rate = crossover_rate
        self.elitism_size = int(elitism_ratio * pop_size)

        # Initialize population
        self.population = self.initialize_population()
```

Figure S1. Selected dominant codes for achieving the GAEAM method developed in this work written in the Python (v3.15) languages interfaced with the LAMMPS (19 Nov 2024) package.

```python
    def initialize_population(self):
        """Initialize population"""
        population = []
        for _ in range(self.pop_size):
            individual = []
            for (min_val, max_val, is_integer) in self.param_ranges:
                if is_integer:
                    gene = random.randint(min_val, max_val)
                else:
                    gene = random.uniform(min_val, max_val)
                individual.append(gene)
            population.append(individual)
        return population

    def calculate_fitness(self, individual):
        """Calculate fitness of an individual"""
        return self.objective_func(individual)

    def select_parents(self, fitness_scores):
        """Roulette wheel selection for parent individuals"""
        # Ensure fitness scores are positive (adjust if the goal is minimization)
        min_fitness = min(fitness_scores)
        if min_fitness < 0:
            adjusted_scores = [score - min_fitness + 1e-6 for score in fitness_scores]
        else:
            adjusted_scores = fitness_scores

        total_fitness = sum(adjusted_scores)
        probabilities = [score / total_fitness for score in adjusted_scores]

        # Select two different parents
        parent1_idx = np.random.choice(self.pop_size, p=probabilities)
        parent2_idx = np.random.choice(self.pop_size, p=probabilities)
```

```python
    def crossover(self, parent1, parent2):
        """Single-point crossover to generate offspring"""
        if random.random() < self.crossover_rate:
            # Randomly select crossover point
            crossover_point = random.randint(1, self.num_params - 1)
            child1 = parent1[:crossover_point] + parent2[crossover_point:]
            child2 = parent2[:crossover_point] + parent1[crossover_point:]
            return child1, child2
        else:
            return parent1.copy(), parent2.copy()

    def mutate(self, individual):
        """Perform mutation on an individual"""
        mutated_individual = individual.copy()

        for i in range(self.num_params):
            if random.random() < self.mutation_rate:
                min_val, max_val, is_integer = self.param_ranges[i]
                if is_integer:
                    mutated_individual[i] = random.randint(min_val, max_val)
                else:
                    # Gaussian mutation centered on current value, within parameter range
                    mean = mutated_individual[i]
                    std = (max_val - min_val) / 10  # Standard deviation is 1/10 of the range
                    new_val = random.gauss(mean, std)
                    # Ensure value is within range
                    new_val = max(min_val, min(new_val, max_val))
                    mutated_individual[i] = new_val

        return mutated_individual
```

Figure S2. Selected dominant codes for achieving the GAEAM method developed in this work written in the Python (v3.15) languages interfaced with the LAMMPS (19 Nov 2024) package.

Part B. Evolution of total energy

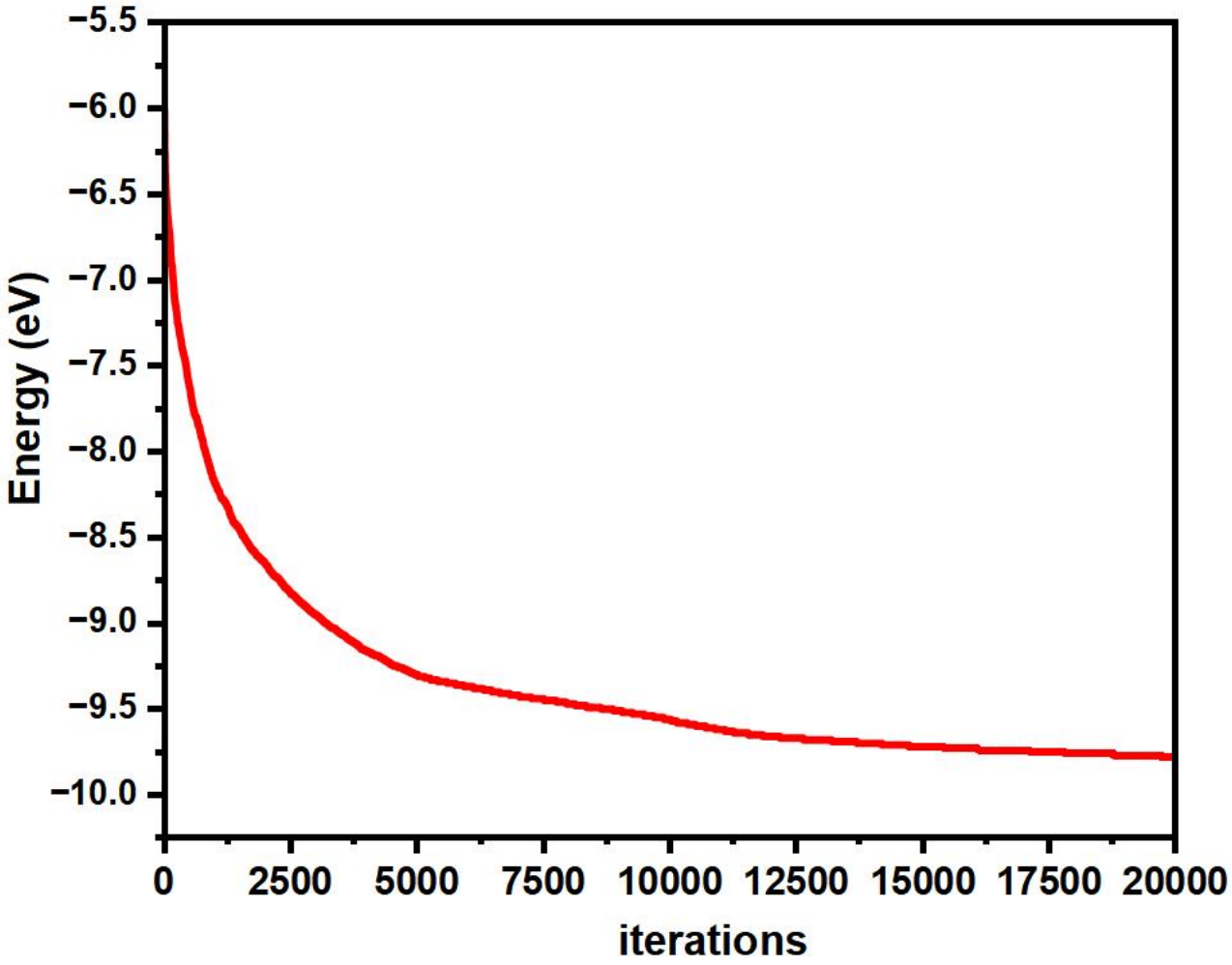

Figure S3. Evolution of total energy (in eV) of Al-Cu alloys optimized by using the GAEAM package and improved genetic algorithm developed in this work interfaced with the LAMMPS (19 Nov 2024) package.

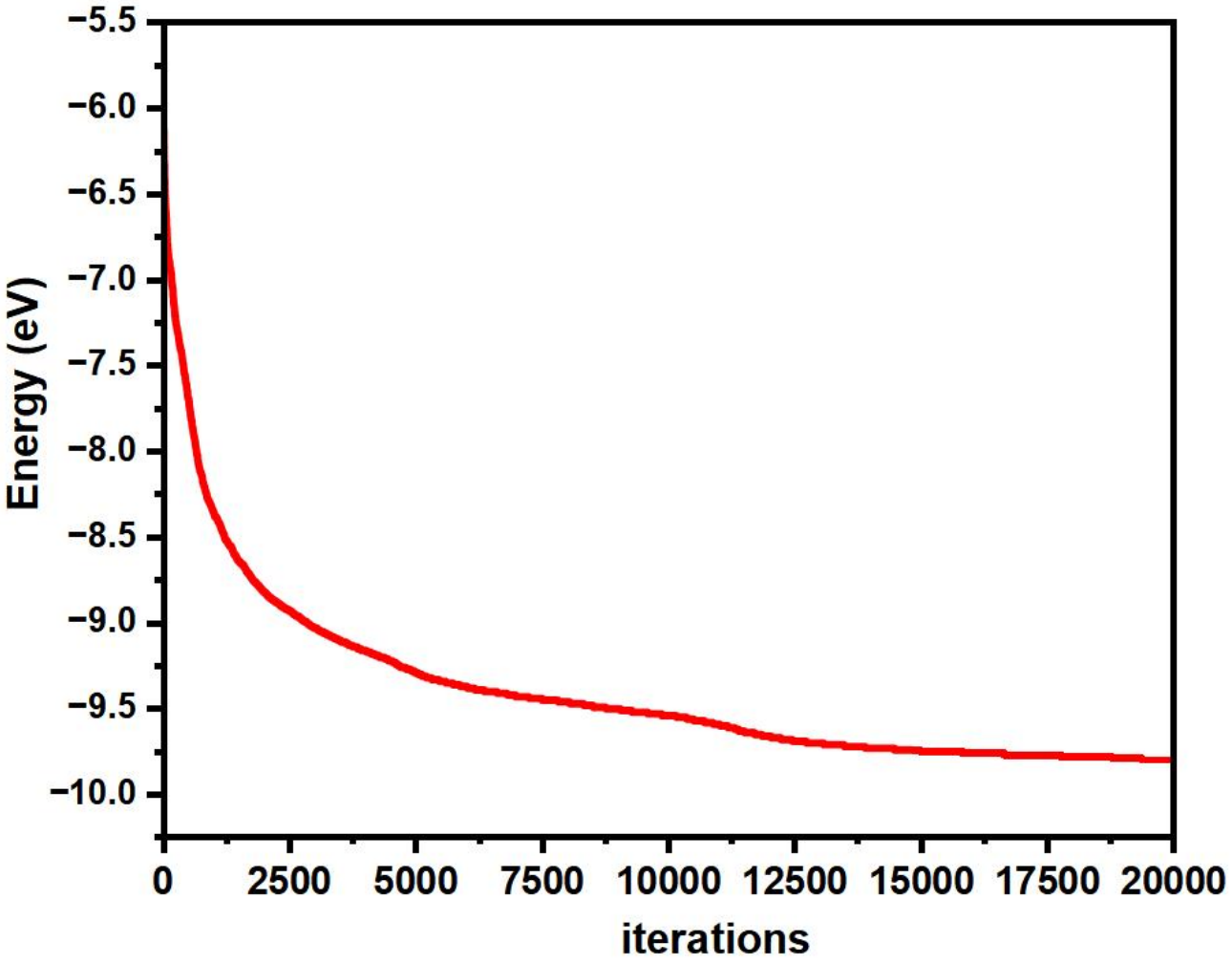

Figure S4. Evolution of total energy (in eV) of Al-Sm alloys optimized by using the GAEAM package and improved genetic algorithm developed in this work interfaced with the LAMMPS (19 Nov 2024) package.

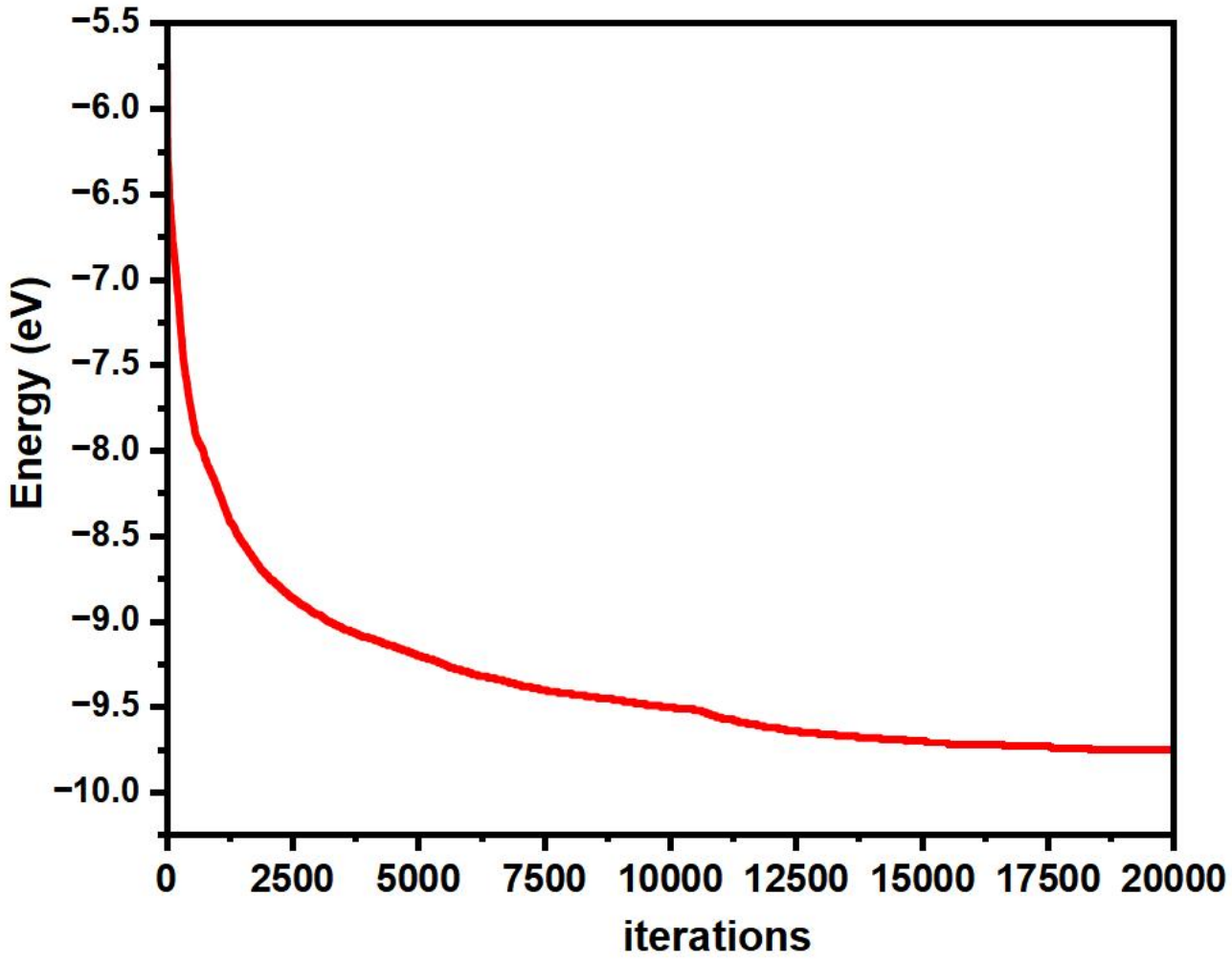

Figure S5. Evolution of total energy (in eV) of Mg-Nd alloys optimized by using the GAEAM package and improved genetic algorithm developed in this work interfaced with the LAMMPS (19 Nov 2024) package.

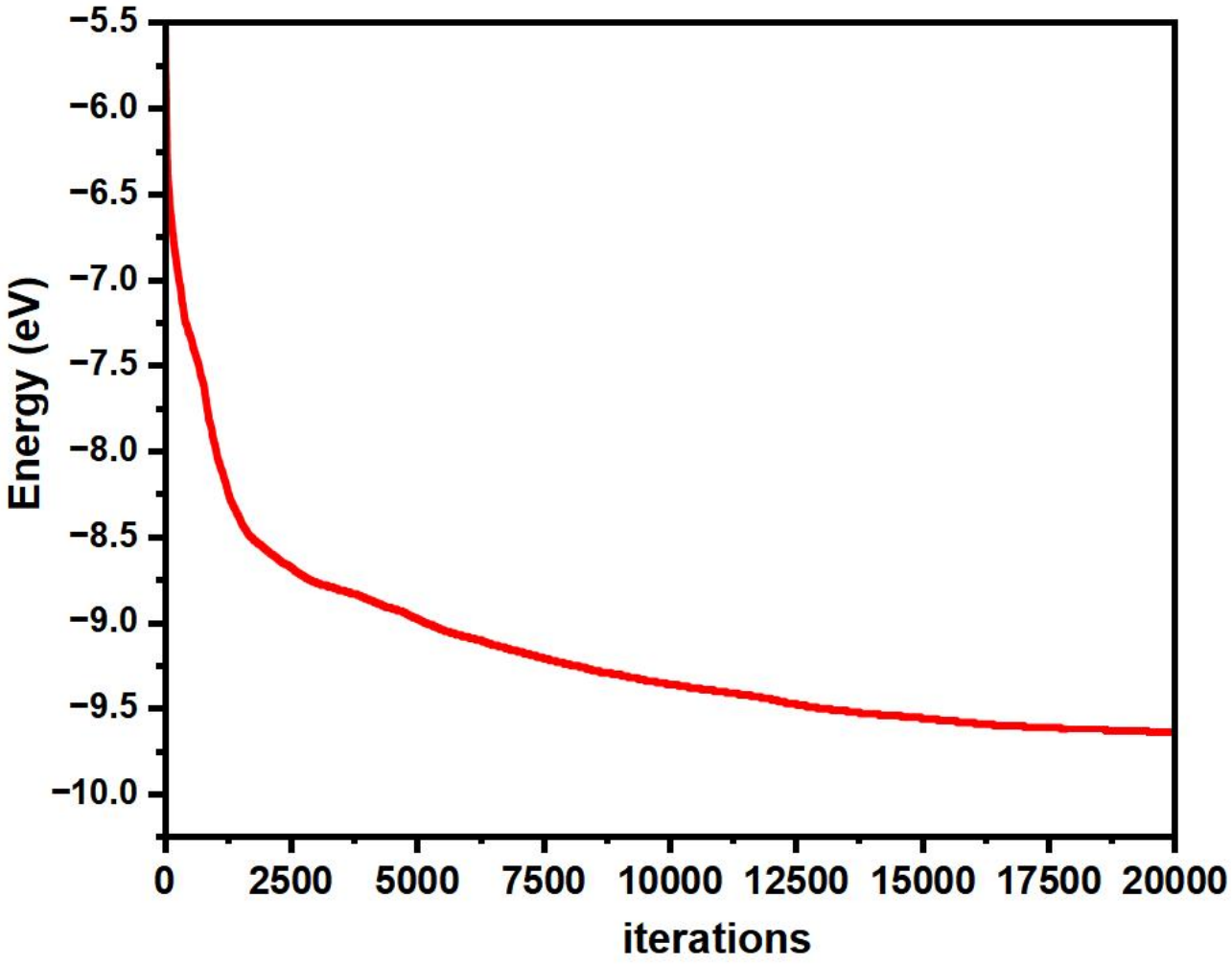

Figure S6. Evolution of total energy (in eV) of Sc-Al alloys optimized by using the GAEAM package and improved genetic algorithm developed in this work interfaced with the LAMMPS (19 Nov 2024) package.

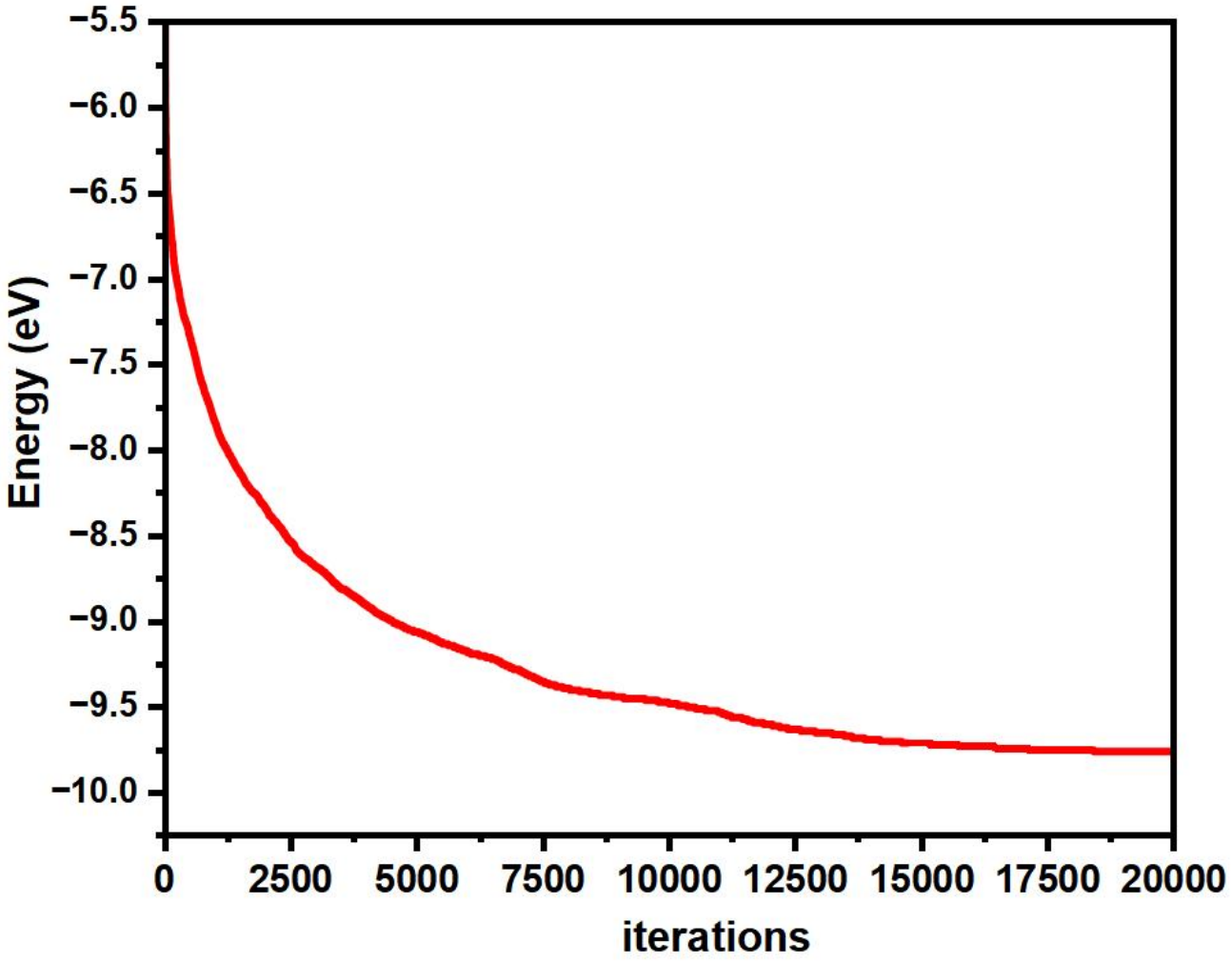

Figure S7. Evolution of total energy (in eV) of Sm-Co alloys optimized by using the GAEAM package and improved genetic algorithm developed in this work interfaced with the LAMMPS (19 Nov 2024) package.

Part C. Evolution of fitness

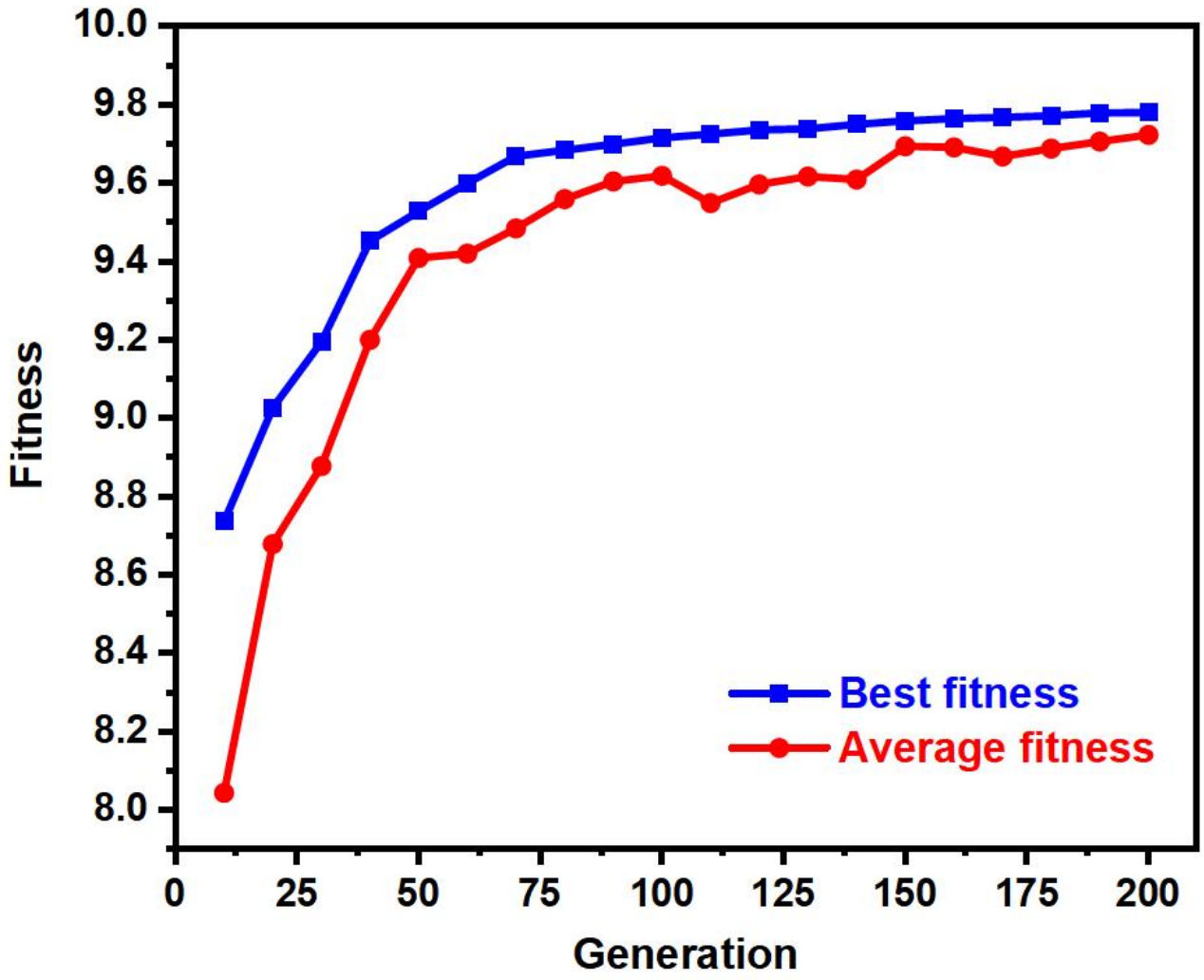

Figure S8. Evolution of best fitness (in blue) and average fitness (in red) in each generation of Al-Cu alloys optimized by using the GAEAM package and improved genetic algorithm developed in this work interfaced with the LAMMPS (19 Nov 2024) package.

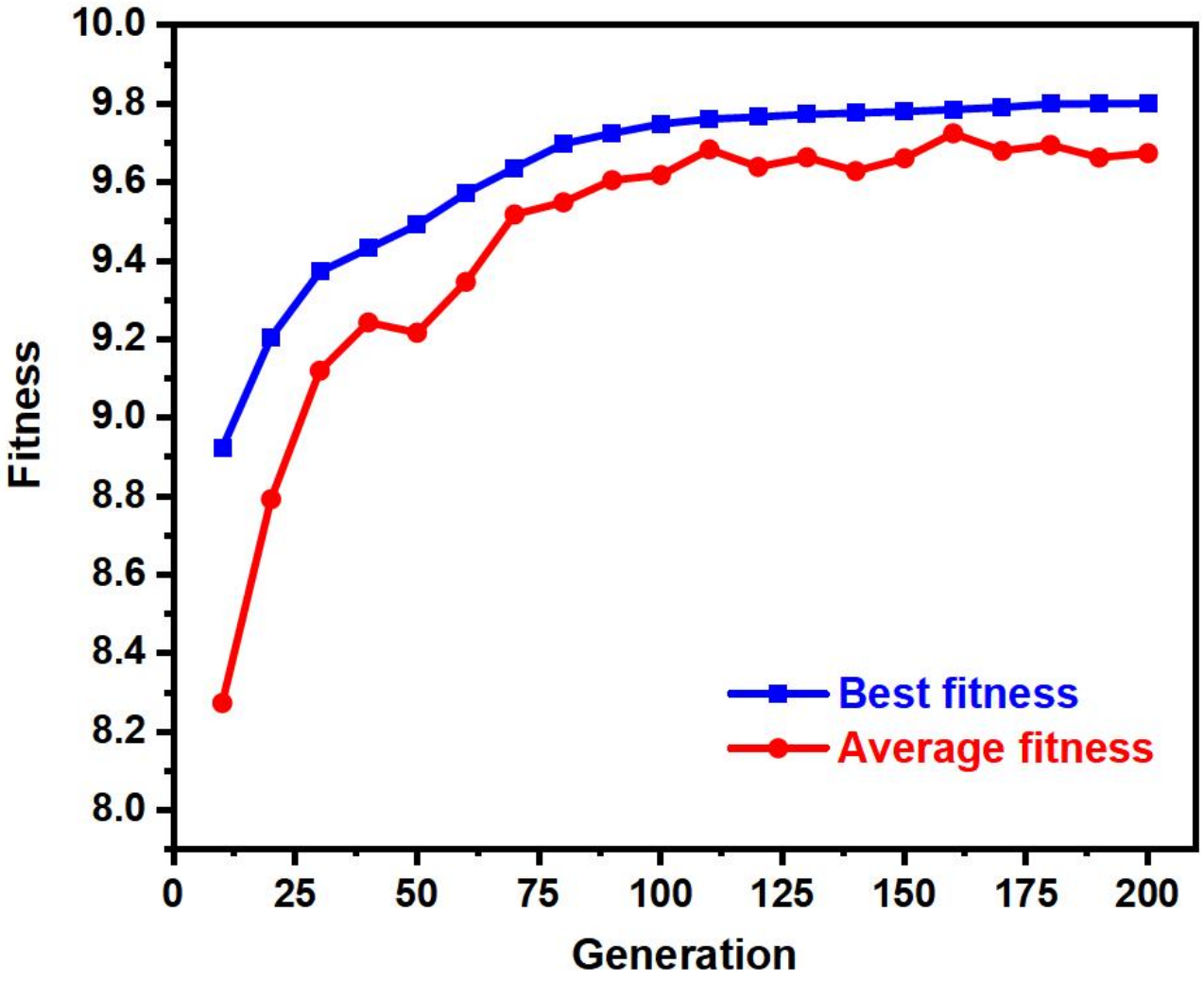

Figure S9. Evolution of best fitness (in blue) and average fitness (in red) in each generation of Al-Sm alloys optimized by using the GAEAM package and improved genetic algorithm developed in this work interfaced with the LAMMPS (19 Nov 2024) package.

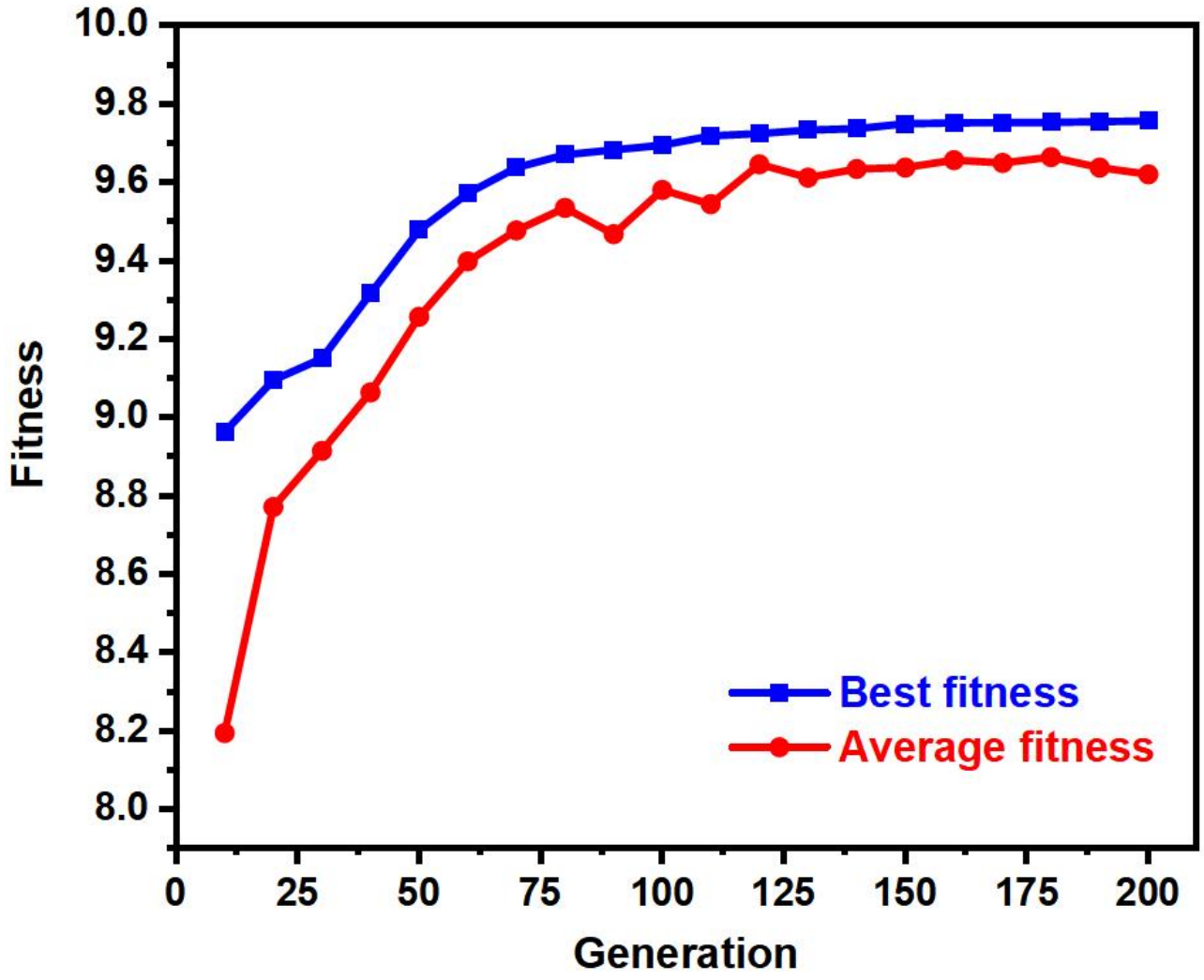

Figure S10. Evolution of best fitness (in blue) and average fitness (in red) in each generation of Mg-Nd alloys optimized by using the GAEAM package and improved genetic algorithm developed in this work interfaced with the LAMMPS (19 Nov 2024) package.

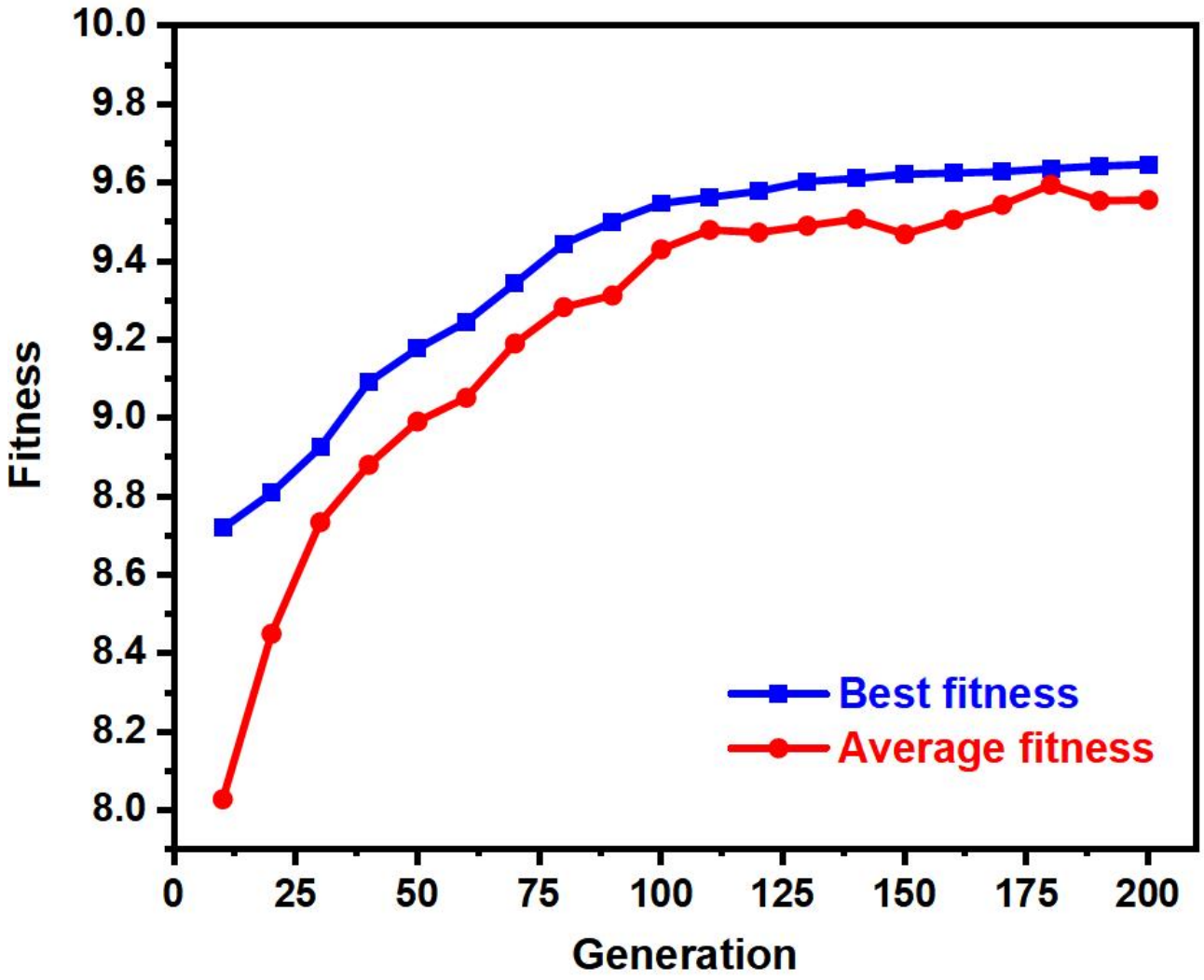

Figure S11. Evolution of best fitness (in blue) and average fitness (in red) in each generation of Sc-Al alloys optimized by using the GAEAM package and improved genetic algorithm developed in this work interfaced with the LAMMPS (19 Nov 2024) package.

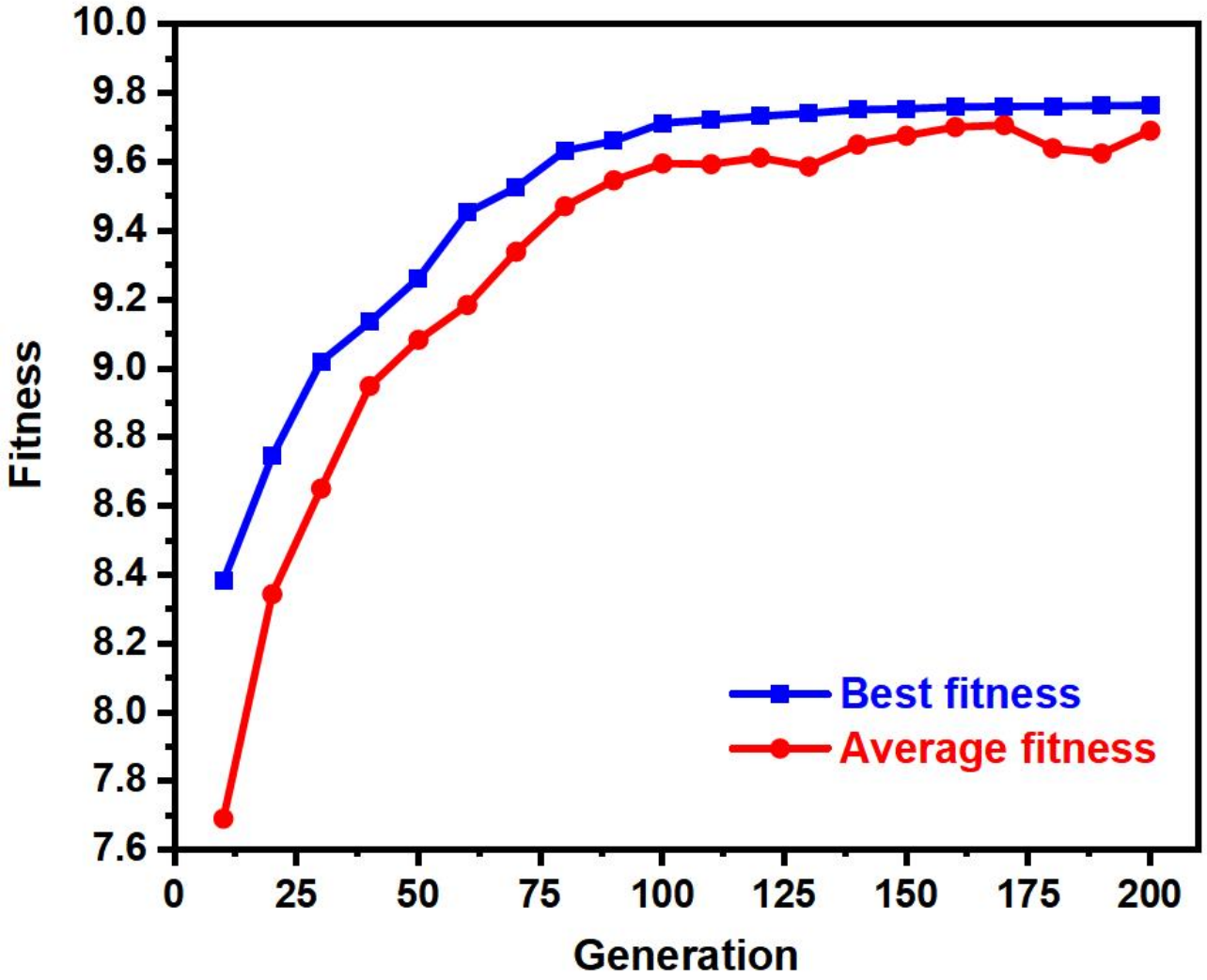

Figure S12. Evolution of best fitness (in blue) and average fitness (in red) in each generation of Sm-Co alloys optimized by using the GAEAM package and improved genetic algorithm developed in this work interfaced with the LAMMPS (19 Nov 2024) package.

Table S1. Evolution of best fitness and average fitness (in eV) in each generation of Al-Cu alloys optimized by using the GAEAM package and improved genetic algorithm developed in this work interfaced with the LAMMPS (19 Nov 2024) package.

Generation	Best fitness	Average fitness
10	8.7358	8.0435
20	9.0258	8.6782
30	9.1944	8.8778
40	9.4524	9.1993
50	9.5276	9.4087
60	9.5992	9.4201
70	9.6684	9.4840
80	9.6836	9.5590
90	9.6994	9.6042
100	9.7152	9.6185
110	9.7246	9.5489
120	9.7353	9.5963
130	9.7386	9.6170
140	9.7496	9.6090
150	9.7585	9.6940
160	9.7647	9.6908
170	9.7675	9.6681
180	9.7717	9.6883
190	9.7790	9.7057
200	9.7799	9.7233

Table S2. Evolution of best fitness and average fitness (in eV) in each generation of Al-Sm alloys optimized by using the GAEAM package and improved genetic algorithm developed in this work interfaced with the LAMMPS (19 Nov 2024) package.

Generation	Best fitness	Average fitness
10	8.9231	8.2734
20	9.2048	8.7925
30	9.3717	9.1200
40	9.4315	9.2432
50	9.4912	9.2166
60	9.5723	9.3461
70	9.6364	9.5183
80	9.6981	9.5489
90	9.7249	9.6053
100	9.7485	9.6186
110	9.7614	9.6841
120	9.7667	9.6391
130	9.7728	9.6637
140	9.7767	9.6289
150	9.7804	9.6610
160	9.7854	9.7246
170	9.7911	9.6801
180	9.7992	9.6954
190	9.8002	9.6630
200	9.8007	9.6746

Table S3. Evolution of best fitness and average fitness (in eV) in each generation of Mg-Nd alloys optimized by using the GAEAM package and improved genetic algorithm developed in this work interfaced with the LAMMPS (19 Nov 2024) package.

Generation	Best fitness	Average fitness
10	8.9631	8.1939
20	9.0947	8.7713
30	9.1504	8.9145
40	9.3156	9.0633
50	9.4774	9.2568
60	9.5718	9.3980
70	9.6375	9.4773
80	9.6705	9.5347
90	9.6827	9.4677
100	9.6946	9.5805
110	9.7183	9.5444
120	9.7245	9.6460
130	9.7329	9.6120
140	9.7368	9.6338
150	9.7493	9.6376
160	9.7516	9.6565
170	9.7520	9.6496
180	9.7530	9.6641
190	9.7547	9.6374
200	9.7565	9.6201

Table S4. Evolution of best fitness and average fitness (in eV) in each generation of Sc-Al alloys optimized by using the GAEAM package and improved genetic algorithm developed in this work interfaced with the LAMMPS (19 Nov 2024) package.

Generation	Best fitness	Average fitness
10	8.7188	8.0280
20	8.8084	8.4494
30	8.9250	8.7343
40	9.0914	8.8802
50	9.1767	8.9908
60	9.2451	9.0517
70	9.3432	9.1902
80	9.4438	9.2825
90	9.5003	9.3123
100	9.5473	9.4301
110	9.5624	9.4796
120	9.5783	9.4727
130	9.6030	9.4900
140	9.6107	9.5075
150	9.6212	9.4687
160	9.6245	9.5057
170	9.6278	9.5435
180	9.6359	9.5950
190	9.6424	9.5538
200	9.6465	9.5559

Table S5. Evolution of best fitness and average fitness (in eV) in each generation of Sm-Co alloys optimized by using the GAEAM package and improved genetic algorithm developed in this work interfaced with the LAMMPS (19 Nov 2024) package.

Generation	Best fitness	Average fitness
10	8.3826	7.6913
20	8.7447	8.3438
30	9.0198	8.6507
40	9.1348	8.9489
50	9.2625	9.0829
60	9.4528	9.1844
70	9.5261	9.3395
80	9.6330	9.4715
90	9.6610	9.5473
100	9.7137	9.5956
110	9.7230	9.5941
120	9.7333	9.6131
130	9.7416	9.5872
140	9.7520	9.6511
150	9.7551	9.6766
160	9.7603	9.7021
170	9.7614	9.7072
180	9.7620	9.6399
190	9.7634	9.6249
200	9.7642	9.6910

Part D. Cartesian coordinates of selected solids

Table S6. Geometrical structures of selected solids including the AlCu, MgNd, $ScAl_3$, Al_2Sm and $SmCo_5$ crystals obtained from the Database of Materials Project and Materials Springer.

AlCu
data_AlCu _symmetry_space_group_name_H-M 'P 1' _cell_length_a 2.99750900 _cell_length_b 2.99750900 _cell_length_c 2.99750900 _cell_angle_alpha 90.00000000 _cell_angle_beta 90.00000000 _cell_angle_gamma 90.00000000 _symmetry_Int_Tables_number 1 _chemical_formula_structural AlCu _chemical_formula_sum 'Al1 Cu1' _cell_volume 26.93279883 _cell_formula_units_Z 1 loop_ _symmetry_equiv_pos_site_id _symmetry_equiv_pos_as_xyz 1 'x, y, z' loop_ _atom_site_type_symbol _atom_site_label _atom_site_symmetry_multiplicity _atom_site_fract_x _atom_site_fract_y _atom_site_fract_z _atom_site_occupancy

Al Al0 1 0.00000000 0.00000000 0.00000000 1
Cu Cu1 1 0.50000000 0.50000000 0.50000000 1
MgNd
data_sm_isp_SD0250273-standardized_unitcell
#Cell Parameters
_cell_length_a 3.881
_cell_length_b 3.881
_cell_length_c 3.881
_cell_angle_alpha 90
_cell_angle_beta 90
_cell_angle_gamma 90
_sm_length_ratio_ab 1.000
_sm_length_ratio_bc 1.000
_sm_length_ratio_ca 1.000
_cell_volume 58.46
_symmetry_space_group_name_H-M 'Pm-3m'
_symmetry_Int_Tables_number 221
_cell_formula_units_Z 1
_sm_cell_transformation
;No transformation from published to standardized cell parameters necessary.
;
#Atom Coordinates
loop_
_atom_site_label
_atom_site_type_symbol
_atom_site_Wyckoff_symbol
_sm_site_symmetry
_atom_site_fract_x

_atom_site_fract_y

_atom_site_fract_z

_atom_site_occupancy

_sm_coordination_number

_sm_atomic_environment_type

Nd 'Nd' .1b .m-3m 0.5 0.5 0.5 1 14 'rhombic dodecahedron, Mg₈Nd₆'

Mg 'Mg' .1a .m-3m 0 0 0 1 14 'rhombic dodecahedron, Nd₈Mg₆'

$ScAl_3$

data_ScAl3

_symmetry_space_group_name_H-M 'P 1'

_cell_length_a 4.10727700

_cell_length_b 4.10727700

_cell_length_c 4.10727700

_cell_angle_alpha 90.00000000

_cell_angle_beta 90.00000000

_cell_angle_gamma 90.00000000

_symmetry_Int_Tables_number 1

_chemical_formula_structural ScAl3

_chemical_formula_sum 'Sc1 Al3'

_cell_volume 69.28863084

_cell_formula_units_Z 1

loop_

_symmetry_equiv_pos_site_id

_symmetry_equiv_pos_as_xyz

1 'x, y, z'

loop_

_atom_site_type_symbol

_atom_site_label

_atom_site_symmetry_multiplicity

```
 _atom_site_fract_x
 _atom_site_fract_y
 _atom_site_fract_z
 _atom_site_occupancy
  Sc   Sc0   1   0.00000000   0.00000000   0.00000000   1
  Al   Al1   1   0.50000000   0.00000000   0.50000000   1
  Al   Al2   1   0.00000000   0.50000000   0.50000000   1
  Al   Al3   1   0.50000000   0.50000000   0.00000000   1
```

Al_2Sm

```
#======================================================================
# CRYSTAL DATA
#----------------------------------------------------------------------
data_VESTA_phase_1


_chemical_name_common                  'sm_isp_SD0250345-standardized_unitcell'
_cell_length_a                         7.941800
_cell_length_b                         7.941800
_cell_length_c                         7.941800
_cell_angle_alpha                      90.000000
_cell_angle_beta                       90.000000
_cell_angle_gamma                      90.000000
_cell_volume                           500.906719
_space_group_name_H-M_alt              'F d -3 m'
_space_group_IT_number                 227


loop_
_space_group_symop_operation_xyz
   'x, y, z'
```

'-x, -y, -z'

'-x+3/4, -y+1/4, z+1/2'

'x+1/4, y+3/4, -z+1/2'

'-x+1/4, y+1/2, -z+3/4'

'x+3/4, -y+1/2, z+1/4'

'x+1/2, -y+3/4, -z+1/4'

'-x+1/2, y+1/4, z+3/4'

'z, x, y'

'-z, -x, -y'

'z+1/2, -x+3/4, -y+1/4'

'-z+1/2, x+1/4, y+3/4'

'-z+3/4, -x+1/4, y+1/2'

'z+1/4, x+3/4, -y+1/2'

'-z+1/4, x+1/2, -y+3/4'

'z+3/4, -x+1/2, y+1/4'

'y, z, x'

'-y, -z, -x'

'-y+1/4, z+1/2, -x+3/4'

'y+3/4, -z+1/2, x+1/4'

'y+1/2, -z+3/4, -x+1/4'

'-y+1/2, z+1/4, x+3/4'

'-y+3/4, -z+1/4, x+1/2'

'y+1/4, z+3/4, -x+1/2'

'y+3/4, x+1/4, -z+1/2'

'-y+1/4, -x+3/4, z+1/2'

'-y, -x, -z'

'y, x, z'

'y+1/4, -x+1/2, z+3/4'

'-y+3/4, x+1/2, -z+1/4'

'-y+1/2, x+3/4, z+1/4'

'y+1/2, -x+1/4, -z+3/4'

'x+3/4, z+1/4, -y+1/2'

'-x+1/4, -z+3/4, y+1/2'

'-x+1/2, z+3/4, y+1/4'

'x+1/2, -z+1/4, -y+3/4'

'-x, -z, -y'

'x, z, y'

'x+1/4, -z+1/2, y+3/4'

'-x+3/4, z+1/2, -y+1/4'

'z+3/4, y+1/4, -x+1/2'

'-z+1/4, -y+3/4, x+1/2'

'z+1/4, -y+1/2, x+3/4'

'-z+3/4, y+1/2, -x+1/4'

'-z+1/2, y+3/4, x+1/4'

'z+1/2, -y+1/4, -x+3/4'

'-z, -y, -x'

'z, y, x'

'x, y+1/2, z+1/2'

'-x, -y+1/2, -z+1/2'

'-x+3/4, -y+3/4, z'

'x+1/4, y+1/4, -z'

'-x+1/4, y, -z+1/4'

'x+3/4, -y, z+3/4'

'x+1/2, -y+1/4, -z+3/4'

'-x+1/2, y+3/4, z+1/4'

'z, x+1/2, y+1/2'

'-z, -x+1/2, -y+1/2'

'z+1/2, -x+1/4, -y+3/4'

'-z+1/2, x+3/4, y+1/4'

'-z+3/4, -x+3/4, y'

'z+1/4, x+1/4, -y'

'-z+1/4, x, -y+1/4'

'z+3/4, -x, y+3/4'

'y, z+1/2, x+1/2'

'-y, -z+1/2, -x+1/2'

'-y+1/4, z, -x+1/4'

'y+3/4, -z, x+3/4'

'y+1/2, -z+1/4, -x+3/4'

'-y+1/2, z+3/4, x+1/4'

'-y+3/4, -z+3/4, x'

'y+1/4, z+1/4, -x'

'y+3/4, x+3/4, -z'

'-y+1/4, -x+1/4, z'

'-y, -x+1/2, -z+1/2'

'y, x+1/2, z+1/2'

'y+1/4, -x, z+1/4'

'-y+3/4, x, -z+3/4'

'-y+1/2, x+1/4, z+3/4'

'y+1/2, -x+3/4, -z+1/4'

'x+3/4, z+3/4, -y'

'-x+1/4, -z+1/4, y'

'-x+1/2, z+1/4, y+3/4'

'x+1/2, -z+3/4, -y+1/4'

'-x, -z+1/2, -y+1/2'

'x, z+1/2, y+1/2'

'x+1/4, -z, y+1/4'

'-x+3/4, z, -y+3/4'

'z+3/4, y+3/4, -x'

'-z+1/4, -y+1/4, x'

'z+1/4, -y, x+1/4'

'-z+3/4, y, -x+3/4'

'-z+1/2, y+1/4, x+3/4'

'z+1/2, -y+3/4, -x+1/4'

'-z, -y+1/2, -x+1/2'

'z, y+1/2, x+1/2'

'x+1/2, y, z+1/2'

'-x+1/2, -y, -z+1/2'

'-x+1/4, -y+1/4, z'

'x+3/4, y+3/4, -z'

'-x+3/4, y+1/2, -z+1/4'

'x+1/4, -y+1/2, z+3/4'

'x, -y+3/4, -z+3/4'

'-x, y+1/4, z+1/4'

'z+1/2, x, y+1/2'

'-z+1/2, -x, -y+1/2'

'z, -x+3/4, -y+3/4'

'-z, x+1/4, y+1/4'

'-z+1/4, -x+1/4, y'

'z+3/4, x+3/4, -y'

'-z+3/4, x+1/2, -y+1/4'

'z+1/4, -x+1/2, y+3/4'

'y+1/2, z, x+1/2'

'-y+1/2, -z, -x+1/2'

'-y+3/4, z+1/2, -x+1/4'

'y+1/4, -z+1/2, x+3/4'

'y, -z+3/4, -x+3/4'

'-y, z+1/4, x+1/4'

'-y+1/4, -z+1/4, x'

'y+3/4, z+3/4, -x'

'y+1/4, x+1/4, -z'

'-y+3/4, -x+3/4, z'

'-y+1/2, -x, -z+1/2'

'y+1/2, x, z+1/2'

'y+3/4, -x+1/2, z+1/4'

'-y+1/4, x+1/2, -z+3/4'

'-y, x+3/4, z+3/4'

'y, -x+1/4, -z+1/4'

'x+1/4, z+1/4, -y'

'-x+3/4, -z+3/4, y'

'-x, z+3/4, y+3/4'

'x, -z+1/4, -y+1/4'

'-x+1/2, -z, -y+1/2'

'x+1/2, z, y+1/2'

'x+3/4, -z+1/2, y+1/4'

'-x+1/4, z+1/2, -y+3/4'

'z+1/4, y+1/4, -x'

'-z+3/4, -y+3/4, x'

'z+3/4, -y+1/2, x+1/4'

'-z+1/4, y+1/2, -x+3/4'

'-z, y+3/4, x+3/4'

'z, -y+1/4, -x+1/4'

'-z+1/2, -y, -x+1/2'

'z+1/2, y, x+1/2'

'x+1/2, y+1/2, z'

'-x+1/2, -y+1/2, -z'

'-x+1/4, -y+3/4, z+1/2'

'x+3/4, y+1/4, -z+1/2'

'-x+3/4, y, -z+3/4'

'x+1/4, -y, z+1/4'

'x, -y+1/4, -z+1/4'

'-x, y+3/4, z+3/4'

'z+1/2, x+1/2, y'

'-z+1/2, -x+1/2, -y'

'z, -x+1/4, -y+1/4'

'-z, x+3/4, y+3/4'

'-z+1/4, -x+3/4, y+1/2'

'z+3/4, x+1/4, -y+1/2'

'-z+3/4, x, -y+3/4'

'z+1/4, -x, y+1/4'

'y+1/2, z+1/2, x'

'-y+1/2, -z+1/2, -x'

'-y+3/4, z, -x+3/4'

'y+1/4, -z, x+1/4'

'y, -z+1/4, -x+1/4'

'-y, z+3/4, x+3/4'

'-y+1/4, -z+3/4, x+1/2'

'y+3/4, z+1/4, -x+1/2'

'y+1/4, x+3/4, -z+1/2'

'-y+3/4, -x+1/4, z+1/2'

'-y+1/2, -x+1/2, -z'

'y+1/2, x+1/2, z'

'y+3/4, -x, z+3/4'

'-y+1/4, x, -z+1/4'

'-y, x+1/4, z+1/4'

```
    'y, -x+3/4, -z+3/4'
    'x+1/4, z+3/4, -y+1/2'
    '-x+3/4, -z+1/4, y+1/2'
    '-x, z+1/4, y+1/4'
    'x, -z+3/4, -y+3/4'
    '-x+1/2, -z+1/2, -y'
    'x+1/2, z+1/2, y'
    'x+3/4, -z, y+3/4'
    '-x+1/4, z, -y+1/4'
    'z+1/4, y+3/4, -x+1/2'
    '-z+3/4, -y+1/4, x+1/2'
    'z+3/4, -y, x+3/4'
    '-z+1/4, y, -x+1/4'
    '-z, y+1/4, x+1/4'
    'z, -y+3/4, -x+3/4'
    '-z+1/2, -y+1/2, -x'
    'z+1/2, y+1/2, x'

loop_
    _atom_site_label
    _atom_site_occupancy
    _atom_site_fract_x
    _atom_site_fract_y
    _atom_site_fract_z
    _atom_site_adp_type
    _atom_site_U_iso_or_equiv
    _atom_site_type_symbol
    Al          1.0     0.000000     0.000000     0.000000     Uiso  ? Al
    Sm          1.0     0.375000     0.375000     0.375000     Uiso  ? Sm
```

$SmCo_5$

```
data_SmCo5
_symmetry_space_group_name_H-M      'P 1'
_cell_length_a      4.96798097
_cell_length_b      4.96798097
_cell_length_c      3.96287000
_cell_angle_alpha      90.00000000
_cell_angle_beta      90.00000000
_cell_angle_gamma      120.00001285
_symmetry_Int_Tables_number      1
_chemical_formula_structural      SmCo5
_chemical_formula_sum      'Sm1 Co5'
_cell_volume      84.70328381
_cell_formula_units_Z      1
loop_
 _symmetry_equiv_pos_site_id
 _symmetry_equiv_pos_as_xyz
   1    'x, y, z'
loop_
 _atom_site_type_symbol
 _atom_site_label
 _atom_site_symmetry_multiplicity
 _atom_site_fract_x
 _atom_site_fract_y
 _atom_site_fract_z
 _atom_site_occupancy
   Sm    Sm0    1    0.00000000    0.00000000    0.00000000    1
   Co    Co1    1    0.66666700    0.33333300    0.00000000    1
   Co    Co2    1    0.33333300    0.66666700    0.00000000    1
```

Co	Co3	1	0.50000000	0.50000000	0.50000000	1
Co	Co4	1	0.50000000	0.00000000	0.50000000	1
Co	Co5	1	0.00000000	0.50000000	0.50000000	1

Part E. Complete user guide for the usage of GAEAM package

Table S7. Usage procedures and a clear description of how to implement and use the package.

Input structure:

- Input file 1: Alloy crystal structure file (POSCAR/CIF format, from Materials Project).
- Input file 2: Parameter range configuration file (JSON format, defining upper/lower limits of EAM parameters).
- Input file 3: GA hyperparameter file (population size, crossover rate, mutation rate, generation number).

Output structure:

- Output file 1: Optimized EAM potential file (.eam.alloy format, compatible with LAMMPS).
- Output file 2: GA evolution report (fitness change curve, optimal parameter list).
- Output file 3: MD simulation result file (RDF, CN, RMSD, energy data).

Usage procedure:

Step 1: Configure input files and set alloy system parameters.

Step 2: Run the main Python script of GAEAM to start automatic optimization.

Step 3: Wait for GA convergence and obtain the final EAM potential.

Step 4: Call LAMMPS for MD simulation using the optimized potential.

Part F. Optimized parameters for EAM potentials

Table S8. Optimized parameters for EAM potentials for different selected alloys (AlCu, MgNd, $ScAl_3$, Al_2Sm, and $SmCo_5$) in this work by using the GAEAM package.

AlCu
Best parameters: [3.178956, 1.557458, 6.547033, 3.491751, 0.311724, 0.405762, 0.376048, 0.841896, 21.482392, 25.747271, -2.779526, -0.298421, 1.307284, -1.235128, -2.8017, 0.0, 0.634934, -2.585016, 0.786316, -3.118954, 2.5306, 1.53894, 9.021658, 4.291384, 0.392654, 0.608374, 0.305694, 0.839732, 23.147772, 23.504688, -2.216616, -0.26115, 1.098368, -0.809427, -2.1681, 0.0, 0.598997, -2.079589, 0.307385, -2.404475]
MgNd
Best parameters: [3.178956, 1.557458, 6.547033, 3.491751, 0.311724, 0.405762, 0.376048, 0.843258, 21.416061, 25.747271, -2.779526, -0.298421, 1.373873, -1.235128, -2.8017, 0.0, 0.647122, -2.463362, 0.846925, -3.103126, 2.5306, 1.53894, 9.021658, 4.291384, 0.392654, 0.608374, 0.312094, 0.839732, 21.365999, 23.504688, -2.148566, -0.26115, 1.159177, -0.809427, -2.1681, 0.0, 0.591995, -2.079589, 0.310066, -2.402993]
$ScAl_3$
Best parameters: [3.178956, 1.557458, 6.547033, 3.491751, 0.311724, 0.405762, 0.376048, 0.843258, 20.792432, 25.747271, -2.779526, -0.298421, 1.247973, -1.235128, -2.8017, 0.0,

0.658215, -2.463362, 0.817622, -2.986848, 2.5306, 1.53894, 9.021658, 4.291384, 0.392654, 0.608374, 0.336951, 0.839732, 22.445078, 23.156701, -2.148566, -0.26115, 1.146025, -0.809427, -2.1681, 0.0, 0.609773, -2.079589, 0.307385, -2.415509]
Al_2Sm
Best parameters: [3.178956, 1.557458, 6.547033, 3.491751, 0.311724, 0.405762, 0.376048, 0.843258, 20.680891, 25.747271, -2.779526, -0.298421, 1.329601, -1.235128, -2.8017, 0.0, 0.657049, -2.463362, 0.813513, -3.12561, 2.5306, 1.53894, 9.021658, 4.291384, 0.392654, 0.608374, 0.305694, 0.839732, 21.756382, 23.504688, -2.148566, -0.26115, 1.20503, -0.892376, -2.1681, 0.0, 0.556212, -2.079589, 0.307385, -2.420449]
$SmCo_5$
Best parameters: [3.178956, 1.557458, 6.547033, 3.491751, 0.311724, 0.405762, 0.376048, 0.843258, 21.440862, 25.747271, -2.779526, -0.324667, 1.397004, -1.304151, -2.8017, 0.0, 0.660743, -2.463362, 0.778043, -3.101124, 2.5306, 1.53894, 9.021658, 4.291384, 0.392654, 0.608374, 0.305694, 0.839732, 22.503494, 22.939174, -2.148566, -0.26115, 1.120768, -0.809427, -2.1681, 0.0, 0.574564, -2.079589, 0.307385, -2.407977]

Part G. Reported parameters for EAM potentials

Table S9. Reported optimized parameters for EAM potentials for AlCu alloys obtained from the Interatomic Potentials Repository-NIST at https://www.ctcms.nist.gov/potentials/.

Parameter	Cu	Al
r_e	2.556162	2.863924
f_e	1.554485	1.403115
ρ_e	21.175871	20.418205
ρ_s	21.175395	23.195740
γ	8.127620	6.613165
ω	4.334731	3.527021
A	0.396620	0.314873
B	0.548085	0.365551
κ	0.308782	0.379846
λ	0.756515	0.759692
F_{n0}	−2.170269	−2.807602
F_{n1}	−0.263788	−0.301435
F_{n2}	1.088878	1.258562
F_{n3}	−0.817603	−1.247604
F_0	−2.19	−2.83
F_1	0	0
F_2	0.561830	0.622245
F_3	−2.100595	−2.488244
η	0.310490	0.785902
F_e	−2.186568	−2.824528