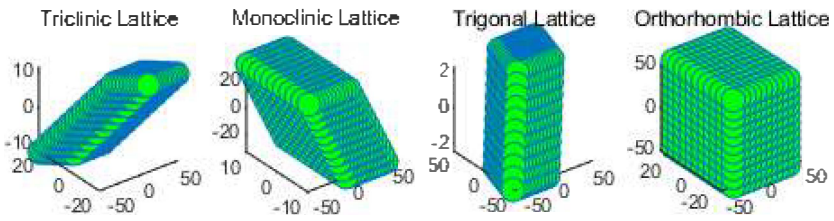
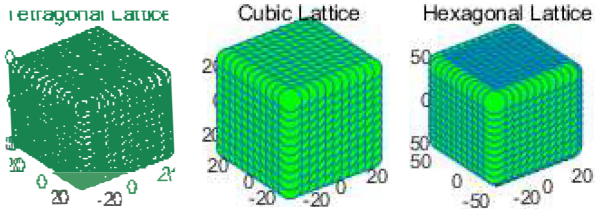
```
clc;
clear,

% Triclinic Lattice
lattice = build lattice('Triclinic', 5, 'a', 4.0, 'b', 3.0, 'c', 5.0, 'alpha', 1.2 * pi, 'beta', 0.7 * pi, 'gamma', 0.9 * pi),
subplot(2, 4, 1);
plot3(lattice(:, 1), lattice(:, 2), lattice(:, 3), 'o', 'MarkerFaceColor','g', 'MarkerSize', 10);
title('Triclinic Lattice'),
axis square,

% Monoclinic Lattice
lattice = build_lattice('Monoclinic', 5, 'a', 5.0, 'b', 2.0, 'c', 8.0, 'beta', 0.7 * pi),
subplot(2, 4, 2);
plot3(lattice(:, 1), lattice(:, 2), lattice(:, 3), 'o', 'MarkerFaceColor', 'g', 'MarkerSize', 10);
title('Monoclinic Lattice');
axis square,

% Trigonal Lattice
lattice = build_lattice('Trigonal', 5, 'a', 5.0, 'alpha', 0.4 * pi);
subplot(2, 4, 3);
plot3(lattice(:, 1), lattice(:, 2), lattice(:, 3), 'o', 'MarkerFaceColor', 'g', 'MarkerSize', 10);
title('Trigonal Lattice');
axis square;

% Orthorhombic Lattice
lattice = build_lattice('Orthorhombic', 5, 'a', 7.0, 'b', 5.0, 'c', 10.0),
subplot(2, 4, 4);
plot3(lattice(:, 1), lattice(:, 2), lattice(:, 3), 'o', 'MarkerFaceColor', 'g', 'MarkerSize', 10);
title('Orthorhombic Lattice');
axis square;

% Tetragonal Lattice
lattice = build_lattice('Tetragonal', 5, 'a', 5.0, 'c', 8.0);
subplot(2, 4, 5);
plot3(lattice(:, 1), lattice(:, 2), lattice(:, 3), 'o', 'MarkerFaceColor', 'g', 'MarkerSize', 10);
title('Tetragonal Lattice');
axis square,

% Cubic Lattice
lattice = build_lattice('Cubic', 5, 'a', 5.0);
subplot(2, 4, 6);
plot3(lattice(:, 1), lattice(:, 2), lattice(:, 3), 'o', 'MarkerFaceColor', 'g', 'MarkerSize', 10);
title('Cubic Lattice');
axis square,

% Hexagonal Lattice
lattice = build_lattice('Hexagonal', 5, 'a', 5.0);
subplot(2, 4, 7);
plot3(lattice(:, 1), lattice(:, 2), lattice(:, 3), 'o', 'MarkerFaceColor', 'g', 'MarkerSize', 10);
title('Hexagonal Lattice');
axis square;

% Function Definition
function [lattice] = build_lattice(lattice_name, enlarge_limit, varargin)
    if enlarge_limit < 0
        error('Invalid enlarge limitation');
    end
    switch lattice_name
        case 'Triclinic'
            try
                a = get_parameter('a'),
                b = get_parameter('b')
                c = get_parameter('c')
                alpha = get_parameter('alpha');
                beta = get_parameter('beta');
                gamma = get_parameter('gamma'),
            catch
```

```
        error('Parameters invalid or not enough'),
    end
    atom1 = [0.0, 0.0, 0.0];
    ux = [a, 0.0, 0.0];
    uy = [b * cos(gamma), b * sin(gamma), 0.0];
    uz = [c * sin(beta), c * sin(alpha), c * cos(beta) * cos(alpha)];
    id = 1;
    for i = -enlarge_limit:enlarge_limit
        for j = -enlarge_limit:enlarge_limit
            for k = -enlarge_limit:enlarge_limit
                vector = ux * i + uy * j + uz * k;
                lattice(id, 1:3) = atom1 + vector;
                id = id + 1;
            end
        end
    end

case 'Monoclinic'
    try
        a = get_parameter('a'),
        b = get_parameter('b');
        c = get_parameter('c');
        beta = 0.7*pi;
    catch
        error('Parameters invalid or not enough');
    end
    atom1 = [0.0, 0.0, 0.0];
    ux = [a, 0.0, 0.0];
    uy = [0.0, b, 0.0];
    uz = [c * cos(beta), 0.0, c * sin(beta)],
    id = 1;
    for i = -enlarge_limit:enlarge_limit
        for j = -enlarge_limit:enlarge_limit
            for k = -enlarge_limit:enlarge_limit
                vector = ux * i + uy * j + uz * k;
                lattice(id, 1:3) = atom1 + vector;
                id = id + 1;
            end
        end
    end

case 'Trigonal'
    try
        a = get_parameter('a');
        alpha = get_parameter('alpha'),
    catch
        error('Parameters invalid or not enough'),
    end
    atom1 = [0.0, 0.0, 0.0];
    ux = [a, 0.0, 0.0];
    uy = [a * cos(alpha), a * sin(alpha), 0.0];
    uz = [a * sin(alpha), a * sin(alpha), a * cos(alpha) * cos(alpha)],
    id = 1;
    for i = -enlarge_limit:enlarge_limit
        for j = -enlarge_limit:enlarge_limit
            for k = -enlarge_limit:enlarge_limit
                vector = ux * i + uy * j + uz * k,
                lattice(id, 1:3) = atom1 + vector;
                id = id + 1;
            end
        end
    end

case 'Orthorhombic'
    try
        a = get_parameter('a'),
        b = get_parameter('b');
        c = get_parameter('c');
    catch
```

```
        error('Parameters invalid or not enough')
    end
    atom1 = [0.0, 0.0, 0.0];
    ux = [a, 0.0, 0.0];
    uy = [0.0, b, 0.0];
    uz = [0.0, 0.0, c];
    id = 1;
    for i = -enlarge_limit:enlarge_limit
        for j = -enlarge_limit:enlarge_limit
            for k = -enlarge_limit:enlarge_limit
                vector = ux * i + uy * j + uz * k;
                lattice(id, 1:3) = atom1 + vector;
                id = id + 1;
            end
        end
    end

case 'Tetragonal'
    try
        a = get_parameter('a'),
        c = get_parameter('c');
    catch
        error('Parameters invalid or not enough'),
    end
    atom1 = [0.0, 0.0, 0.0];
    ux = [a, 0.0, 0.0];
    uy = [0.0, a, 0.0];
    uz = [0.0, 0.0, c];
    id = 1;
    for i = -enlarge_limit:enlarge_limit
        for j = -enlarge_limit:enlarge_limit
            for k = -enlarge_limit:enlarge_limit
                vector = ux * i + uy * j + uz * k,
                lattice(id, 1:3) = atom1 + vector;
                id = id + 1;
            end
        end
    end

case 'Cubic'
    try
        a = get_parameter('a'),
    catch
        error('Parameters invalid or not enough');
    end
    atom1 = [0.0, 0.0, 0.0];
    ux = [a, 0.0, 0.0];
    uy = [0.0, a, 0.0];
    uz = [0.0, 0.0, a];
    id = 1;
    for i = -enlarge_limit:enlarge_limit
        for j = -enlarge_limit:enlarge_limit
            for k = -enlarge_limit:enlarge_limit
                vector = ux * i + uy * j + uz * k;
                lattice(id, 1:3) = atom1 + vector;
                id = id + 1;
            end
        end
    end

case 'Hexagonal'
    try
        a = get_parameter('a'),
        try
            c = get_parameter('c');
        catch
            c = a * 1.633;
        end
    catch
```

```
            error('Parameters invalid or not enough'),
        end
        atom1 = [0.0, 0.0, 0.0];
        atom2 = [0.5 * a, sqrt(3) / 2 * a, 0.0];
        atom3 = [0.0, 2. / sqrt(3) * a, 0.5 * c],
        atom4 = [-0.5 * a, -a / sqrt(3) / 2, 0.5 * c],
        ux = [a, 0.0, 0.0];
        uy = [0.0, sqrt(3) * a, 0.0];
        uz = [0.0, 0.0, c];
        id = 1;
        for i = -enlarge_limit:enlarge_limit
            for j = -enlarge_limit:enlarge_limit
                for k = -enlarge_limit:enlarge_limit
                    vector = ux * i + uy * j + uz * k;
                    lattice(id, 1:3) = atom1 + vector;
                    id = id + 1;
                    lattice(id, 1:3) = atom2 + vector,
                    id = id + 1;
                    lattice(id, 1:3) = atom3 + vector;
                    id = id + 1;
                    lattice(id, 1:3) = atom4 + vector,
                    id = id + 1;
                end
            end
        end

    otherwise
        error('Invalid lattice name');
end

function result = get_parameter(parameter_name)
    result = varargin{find(strcmp(parameter_name, varargin)) + 1};
end
end
```





Published with MATLAB® R2023b