```
% Example_5_1.mlx
%
% This is a live script version of Example 5.1 from the paper
% 'Geometric modelling of polycrystalline materials: Laguerre tessellations and periodic
% semi-discrete optimal transport' by D.P. Bourne, M. Pearce & S.M. Roper.
```

Example 5.1 from the paper *"Geometric modelling of polycrystalline materials: Laguerre tessellations and periodic semi-discrete optimal transport" by D.P. Bourne, M. Pearce and S. M. Roper.*

Grain sizes for three different microstructures are specified. The microstructures considered are single phase, dual phase and log-normal. Semi-discrete optimal transport is used to find a **periodic Laguerre diagram** with prescribed random seeds whose cell volumes match the grain volumes of the target microstructure. The calculation involves solving a concave maximisation problem, for which we used the damped Newton method with backtracking. In this example we report average run-times for each microstructure.

Microstructures

In this example a single-phase microstructure is one in which all the grains have the same volume. A dual-phase microstructure has two populations of grains, within each population the grains have the same volume, but grains from different populations have distinct volumes, and both the sizes of the populations and the relative volumes between populations can be varied. A log-normal microstructure has grain volumes drawn from an approximate log-normal distribution.

The box is the unit cube $[0,1]\times[0,1]\times[0,1]$

```
%%% Specify box dimensions
bx=[1,1,1];
```

We compute periodic Laguerre diagrams

```
%%% Specify periodicity
periodic=true;
```

Specify the maximum percentage error in the cell volumes

```
%%% Specify maximum percentage error
percent_tol=1;
```

The vector `n` is a row vector of numbers of cells. In the example below we choose `n=1000`

```
%%% Specify number of grains (cells)
% Set n=[100,250,500,1000,2500,5000,10000,25000,50000,100000]
% to reproduce the results from Example 5.1 of the paper
n=[1000];
nn=length(n);
```

The number of random draws over which the average run-time will be calculated

```
%%% Specify number of experiments
nexp=100;
```

We control which microstructures are considered using the boolean variables below

```
%%% Specify whether the microstructure is simulated or not
singlephase=true;
dualphase=false; % Set to true to reproduce Example 5.1
lognormal=false; % Set to true to reproduce Example 5.1
```

Single phase

All grains have the same volume $v = 1/n$ where n is the number of grains.

```
%%% Single phase run times

if(singlephase)

    disp('Single phase')

    for idx_n=1:nn
        % For each number of cells

        disp(sprintf('Calculating run times for n = %d',n(idx_n)));

        % Initial guess is always w=0
        w_0=zeros(n(idx_n),1);

        for idx_exp=1:nexp
            % For each experiment

            fprintf('\t Experiment number %d',idx_exp);

            % Seeds locations and target volumes
            [X,target_vols]=nPhase(n(idx_n),1,bx);

            % Store X and target_vols in case later interrogation is required
            X_sp{idx_n,idx_exp}=X;
            tv_sp{idx_n,idx_exp}=target_vols;

            % Time the damped Newton method
            tic
            [w,max_percent_err,actual_vols,EXITFLAG] = ...
                SDOT_damped_Newton(w_0,X,target_vols,bx,periodic,percent_tol);
            t=toc;

            fprintf(' : completed in %f\n',t);

            % Store w in case later interrogation is required
            w_sp{idx_n,idx_exp}=w;

            % Store the run time
            runtime_sp(idx_n,idx_exp)=t;
        end

        disp(sprintf('Average run time %f',mean(runtime_sp(idx_n,:)')));
```

```
    end
end
```

```
Single phase
Calculating run times for n = 1000
     Experiment number 1
 : completed in 0.498660
     Experiment number 2
 : completed in 0.420881
     Experiment number 3
 : completed in 0.456592
     Experiment number 4
 : completed in 0.409669
     Experiment number 5
 : completed in 0.404189
     Experiment number 6
 : completed in 0.401299
     Experiment number 7
 : completed in 0.402175
     Experiment number 8
 : completed in 0.426038
     Experiment number 9
 : completed in 0.455368
     Experiment number 10
 : completed in 0.424841
     Experiment number 11
 : completed in 0.422520
     Experiment number 12
 : completed in 0.425252
     Experiment number 13
 : completed in 0.398209
     Experiment number 14
 : completed in 0.391381
     Experiment number 15
 : completed in 0.432474
     Experiment number 16
 : completed in 0.423172
     Experiment number 17
 : completed in 0.545976
     Experiment number 18
 : completed in 0.408215
     Experiment number 19
 : completed in 0.449926
     Experiment number 20
 : completed in 0.423689
     Experiment number 21
 : completed in 0.423108
     Experiment number 22
 : completed in 0.408668
     Experiment number 23
 : completed in 0.436771
     Experiment number 24
 : completed in 0.429886
     Experiment number 25
 : completed in 0.447819
     Experiment number 26
 : completed in 0.603518
     Experiment number 27
 : completed in 0.443976
     Experiment number 28
 : completed in 0.428622
     Experiment number 29
 : completed in 0.427447
     Experiment number 30
```

```
: completed in 0.450526
    Experiment number 31
: completed in 0.414113
    Experiment number 32
: completed in 0.473048
    Experiment number 33
: completed in 0.426658
    Experiment number 34
: completed in 0.425208
    Experiment number 35
: completed in 0.433322
    Experiment number 36
: completed in 0.463152
    Experiment number 37
: completed in 0.476249
    Experiment number 38
: completed in 0.509432
    Experiment number 39
: completed in 0.483030
    Experiment number 40
: completed in 0.494060
    Experiment number 41
: completed in 0.474286
    Experiment number 42
: completed in 0.692564
    Experiment number 43
: completed in 0.471578
    Experiment number 44
: completed in 0.471836
    Experiment number 45
: completed in 0.516039
    Experiment number 46
: completed in 0.427123
    Experiment number 47
: completed in 0.458736
    Experiment number 48
: completed in 0.541227
    Experiment number 49
: completed in 0.563370
    Experiment number 50
: completed in 0.571213
    Experiment number 51
: completed in 0.523992
    Experiment number 52
: completed in 0.506400
    Experiment number 53
: completed in 0.489119
    Experiment number 54
: completed in 0.492642
    Experiment number 55
: completed in 0.489361
    Experiment number 56
: completed in 0.500890
    Experiment number 57
: completed in 0.499622
    Experiment number 58
: completed in 0.519988
    Experiment number 59
: completed in 0.492507
    Experiment number 60
: completed in 0.507814
    Experiment number 61
: completed in 0.461960
    Experiment number 62
```

```
: completed in 0.472196
    Experiment number 63
: completed in 0.507476
    Experiment number 64
: completed in 0.483591
    Experiment number 65
: completed in 0.654111
    Experiment number 66
: completed in 0.500502
    Experiment number 67
: completed in 0.506579
    Experiment number 68
: completed in 0.491275
    Experiment number 69
: completed in 0.531919
    Experiment number 70
: completed in 0.589576
    Experiment number 71
: completed in 0.498188
    Experiment number 72
: completed in 0.580317
    Experiment number 73
: completed in 0.502092
    Experiment number 74
: completed in 0.520084
    Experiment number 75
: completed in 0.512867
    Experiment number 76
: completed in 0.462629
    Experiment number 77
: completed in 0.575688
    Experiment number 78
: completed in 0.488852
    Experiment number 79
: completed in 0.515006
    Experiment number 80
: completed in 0.487369
    Experiment number 81
: completed in 0.483427
    Experiment number 82
: completed in 0.483357
    Experiment number 83
: completed in 0.534666
    Experiment number 84
: completed in 0.531274
    Experiment number 85
: completed in 0.504303
    Experiment number 86
: completed in 0.536595
    Experiment number 87
: completed in 0.501528
    Experiment number 88
: completed in 0.472351
    Experiment number 89
: completed in 0.479532
    Experiment number 90
: completed in 0.481866
    Experiment number 91
: completed in 0.513942
    Experiment number 92
: completed in 0.533845
    Experiment number 93
: completed in 0.515031
    Experiment number 94
```

```
: completed in 0.483079
    Experiment number 95
: completed in 0.491296
    Experiment number 96
: completed in 0.476897
    Experiment number 97
: completed in 0.582293
    Experiment number 98
: completed in 0.527526
    Experiment number 99
: completed in 0.612207
    Experiment number 100
: completed in 0.570495
Average run time 0.485872
```

Dual phase

There are n_1 grains of volume v_1 and n_2 grains of volume v_2 where $n = n_1 + n_2$ and $n_1 v_1 + n_2 v_2 = 1$ (since the total volume must be the same as the volume of $[0,1] \times [0,1] \times [0,1]$, which is 1).

```
%%% Dual phase run times

if(dualphase)

    disp('Dual phase');

    % Set a size ratio
    r=5;

    % Set the fraction of seeds that are the smaller phase
    f=0.5;

    for idx_n=1:nn
        % For each number of cells

        disp(sprintf('Calculating run times for n = %d',n(idx_n)));

        % Initial guess is always w=0
        w_0=zeros(n(idx_n),1);

        for idx_exp=1:nexp
            % For each experiment

            fprintf('\t Experiment number %d',idx_exp);

            pn=round(n(idx_n)*[f 1-f]); % number of cells of each volume fraction
            vr=[1,r]; % volume fractions

            % Seeds locations and target volumes
            [X,target_vols]=nPhase(pn,vr,bx);

            % Store X and target_vols in case later interrogation is required
            X_dp{idx_n,idx_exp}=X;
            tv_dp{idx_n,idx_exp}=target_vols;
```

```
            % Time the damped Newton method
            tic
            [w,max_percent_err,actual_vols,EXITFLAG] = ...
                SDOT_damped_Newton(w_0,X,target_vols,bx,periodic,percent_tol);
            t=toc;

            fprintf(' : completed in %f\n',t);

            % Store w in case later interrogation is required
            w_dp{idx_n,idx_exp}=w;

            % Store the run time
            runtime_dp(idx_n,idx_exp)=t;
        end

        disp(sprintf('Average run time %f',mean(runtime_dp(idx_n,:)')));

    end
end
```

Log-normal

In the log-normal microstructure the grain volumes are approximately log-normally distributed.

```
%%% Log-normal distribution run times

if(lognormal)

    disp('Log-normal')

    % Parameters in log-normal distribution
    ln_mean=1; % mean
    std_dev=0.35; % standard deviation
    sigma=sqrt((log(1+(std_dev/ln_mean)^2))); % log-normal parameter sigma
    mu=-0.5*sigma^2+log(ln_mean); % log-normal parameter mu

    for idx_n=1:nn
        % For each number of cells

        disp(sprintf('Calculating run times for n = %d',n(idx_n)));

        % Initial guess is always w=0
        w_0=zeros(n(idx_n),1);

        for idx_exp=1:nexp
            % For each experiment

            fprintf('\t Experiment number %d',idx_exp);

            % Seed locations
            X=rand(n(idx_n),3);

            % Compute target volumes:
            % Draw radii from log-normal distribution
```

```
            rad=lognrnd(mu,sigma,n(idx_n),1);
            % Calculate the corresponding grain volumes
            % (we don't need the factor 4pi/3 as we'll be renormalising)
            target_vols=rad.^3;
            % Normalise the volumes so that they add to the volume of the box
            target_vols=target_vols*prod(bx)/sum(target_vols); % target volumes of the grains

            % Store X and target_vols in case later interrogation is required
            X_ln{idx_n,idx_exp}=X;
            tv_ln{idx_n,idx_exp}=target_vols;

            % Time the damped Newton method
            tic
            [w,max_percent_err,actual_vols,EXITFLAG] = ...
                SDOT_damped_Newton(w_0,X,target_vols,bx,periodic,percent_tol);
            t=toc;

            fprintf(' : completed in %f\n',t);

            % Store w in case later interrogation is required
            w_ln{idx_n,idx_exp}=w;

            % Store the run time
            runtime_ln(idx_n,idx_exp)=t;
        end

        disp(sprintf('Average run time %f',mean(runtime_ln(idx_n,:)')));

    end
end
```