Supporting Information

Detection and Characterisation of Sub-Critical Nuclei during Reactive Pd Metal Nucleation by X-ray Absorption Spectroscopy

S.-Y. Chang,[a] Y. Gründer,[a,b,e] S.G. Booth,[b] L.B. Molleta,[a] A. Uehara,[b,f] J.F.W. Mosselmans,[c] G. Cibin,[c] V.-T. Pham,[g, h] L. Nataf,[g] R.A.W. Dryfe,[b] S.L.M. Schroeder[a-d]

[a] School of Chemical Engineering and Analytical Science, School of Chemistry, University of Manchester, Manchester M13 9PL.

[b] Diamond Light Source Ltd., Diamond House, Harwell Science and Innovation Campus, Fermi Ave, Didcot, Oxfordshire OX11 0DE.

[c] School of Chemical and Process Engineering, Faculty of Engineering, University of Leeds, Leeds LS2 9JT.

[d] Oliver Lodge Laboratory, Department of Physics, University of Liverpool, Liverpool L69 7ZE.

[e] Present address: Oliver Lodge Laboratory, Department of Physics, University of Liverpool, Liverpool L69 7ZE.

[f] Present address: Division of Nuclear Engineering Science, Research Reactor Institute, Kyoto University, Asashironishi, Kumatori, Osaka, 590-0494, Japan.

[g] Synchrotron SOLEIL, L'Orme des Merisiers, Saint-Aubin, BP48, 91192, Gif-sur-Yvette, France.

[h] Center for Quantum Electronics, Institute of Physics, Vietnam Academy of Science and Technology, P.O. Box 429, Boho, 10000 Hanoi, Viet Nam.

Open circuit potential, dispersive XAFS results and the MATLAB script for edge-step height fitting.

A. Open Circuit Potential

The open circuit potential (OCP) of the liquid-liquid interface was measured in a four-electrode cell (fig. S1, inset). The concentrations of $[PdCl_4]^{2-}$ and $Fe(cp)_2$ used in the OCP measurement were the same as the XAFS experiment where density fluctuation was captured: 1 mM $[PdCl_4]^{2-}$ and 0.1 M LiCl in the aqueous phase; 4 mM $Fe(cp)_2$ and 15 mM [BTPPA][TFPB] in the TFT phase. Pt flag working and counter electrodes (WE and CE) were used along with Ag/AgCl reference electrodes (RE) made in the laboratory. The organic RE was submerged in an aqueous solution of 10 mM LiCl and 1 mM [BTPPA]Cl in contact with the organic solution. Luggin capillaries helped position the two reference solutions close to the interface. The interfacial potential fluctuated at a range of ~30 mV, suggesting that the density fluctuations captured in XAFS may be influenced by the fluctuating interfacial potential (fig. S1).

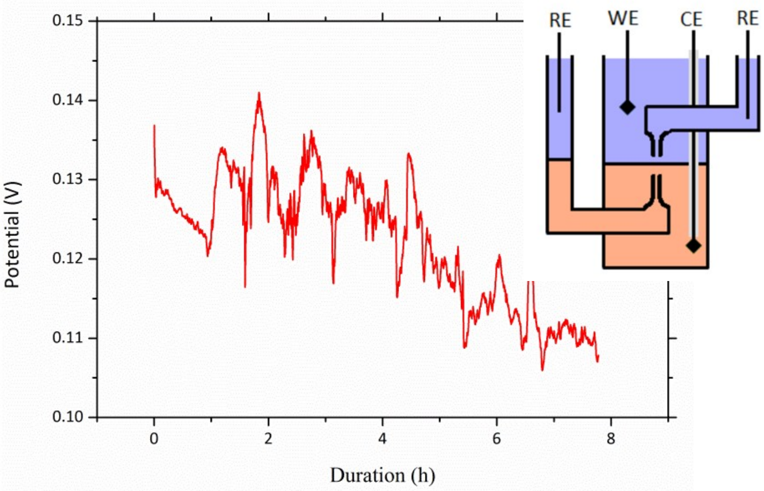

Figure S1. Time profile of the interfacial potential recorded under zero current conditions. Inset: The four-electrode liquid-liquid electrochemical cell used to measure the open circuit potential across a liquid-liquid interface.

B. Dispersive XAFS

The stochastic density fluctuations captured in the quick XAFS measurement at DIAMOND (figs. 2e, 2f and 3) were reproduced in several dispersive XAFS measurements carried out at SOLEIL. An example of the dispersive XAFS result is shown in fig. S2. Initially, the edge-step height was almost constant at ~0.01, which later increased dramatically to ~0.035 at around ~30 minutes. Following this, the measured edge-step height fluctuated significantly, which is indicative of Pd density fluctuations especially during the first four hours of monitoring.

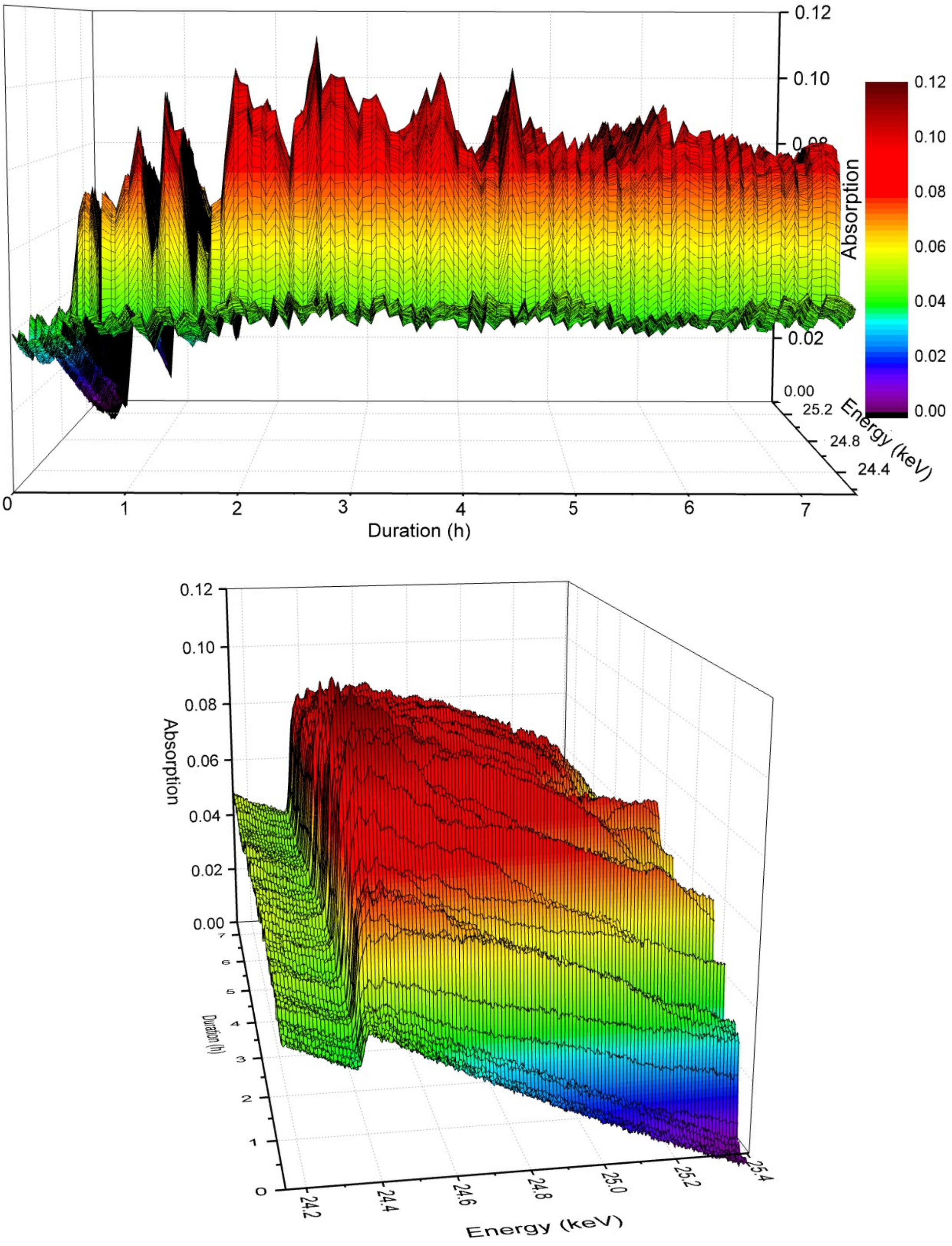

Figure S2. Fluctuating Pd density during the initial stages of the reaction as seen by dispersive XAFS. In an Eppendorf safe-lock tube, an aqueous solution containing 10 mM $[PdCl_4]^{2-}$ and 0.1 M LiCl was contacted with a TFT solution containing 20 mM $Fe(cp)_2$ and 10 mM [BTPPA][TFPB].

C. Particle Size and Coordination Number

Assuming cuboctahedral structures of the nanoparticles we estimate that the particles must grow to diameters beyond 3-5 nm to approach the EXAFS amplitude of bulk metal (fig. S3).

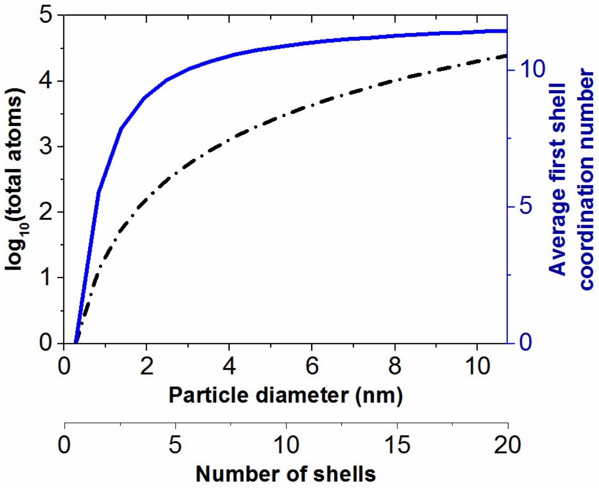

Figure S3. Estimations of average first shell coordination number and total number of atoms as functions of particle diameter and number of shells. The size was estimated from a cuboctahedron[1] which approximates the lattice packing in metal particles well.[2]

D. Edge Height Fitting

Edge-step height quantification of XAFS spectra was automated in MATLAB. Fig. S4 shows an example of fitted pre-edge and post-edge polynomials. The edge-step height can then be determined at the inflection point, E_0.

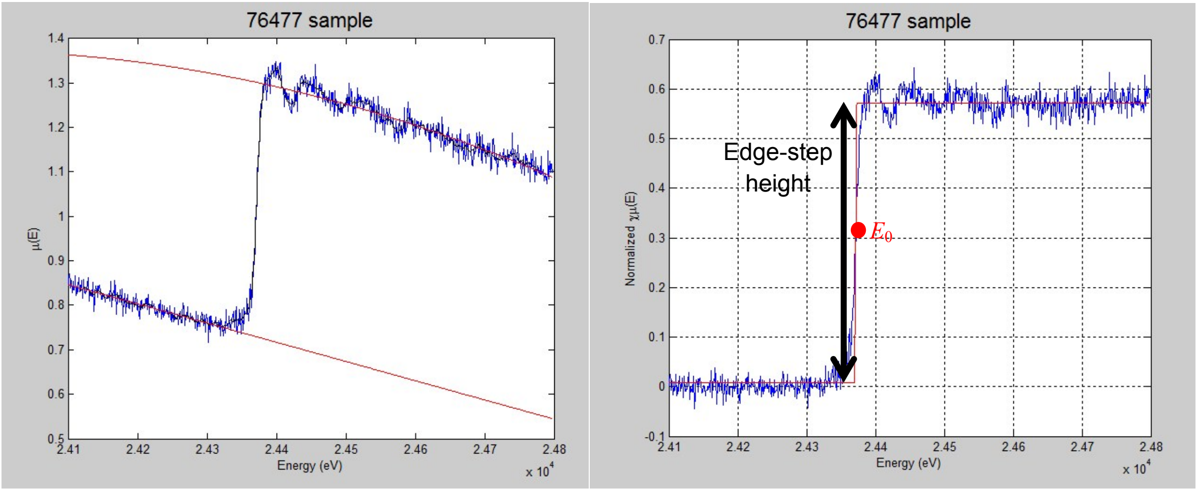

Figure S4. Left: The pre-edge and post-edge polynomials fitted to an experimental XAFS spectrum. Right: An experimental spectrum with the background subtracted.

Following is the MATLAB script for edge-step height calculation.

```
clc % clears command window
```

[1] R. E. Benfield, *J. Chem. Soc. Faraday Trans.*, 1992, **88**, 1107-1110.

[2] S. Díaz-Moreno, D. C. Koningsberger and A. Muñoz-Páez, *Nucl. Instrum. Meth. B*, 1997, **133**, 15-23.

```
clear all % clears all variables in workspace
tic % start measuring time
outputfile = fopen('test.txt', 'w'); % select output file
fprintf(outputfile,'sample\tdate\ttime\theight\tstatus\r\n');
index1i = 76418; % select first sample to analyse
index1f = 76477; % select final sample
descr = '_Pdnocommonion3_'%'_Pd_3_';
a = index1i;
b = 1; % final index of the first sample to be analysed
plt = 1; % use 1 to plot figures, 0 to plot no fgures at all.
xdatat = zeros(2000,index1f-index1i);
ydatat = zeros(2000,index1f-index1i);
norm_ydatat = zeros(2000,index1f-index1i);
index1 = zeros(index1f-index1i,1);
datet = zeros(index1f-index1i,1);
timet = zeros(index1f-index1i,1);
heightt = zeros(index1f-index1i,1);
data_points = zeros(index1f-index1i,1);
n = 1;
%% data importing, e0, normalization/flattening
while a <= index1f
    acount = num2str(a);
    bcount = num2str(b);
    filename = ['Experiment_1\' acount descr bcount '.dat'];
    chck = exist(filename, 'file'); % results in 2 if file exists in
directory
    if chck ~= 2
        b = 1;
        acount1 = num2str(a);
        bcount1 = num2str(b);
        filename1 = ['Experiment_1\' acount1 descr bcount1 '.dat'];
        chck1 = exist(filename1, 'file');
        if chck1 ~= 2
            a = a + 1;
        end
    else
        %% Data importing
        A1 = importdata(filename,'\t', 24);
        p = char(A1.textdata(2,1));
        date1 = p(39:49);
        time1 = p(51:58);
        xdata = A1.data(:,1);
        ydata = A1.data(:,18);
        data_size = size(xdata,1);
        time_format = 'HH:MM:SS';
        date_format = 'dd mmm yyyy';


        % this if condition catches beam dump (which means, blank data
        % points or few data, as in no ydata points)
        if size(xdata,1) < 400 || nnz(ydata == 0) ~= 0
            %fprintf(outputfile,'%d\t%s\t%s\tfew data\r\n',a,date1,time1);
            a = a + 1;
            b = b + 1;
        else

            index1(n) = a;
            datet(n) = datenum(date1,date_format);
            timet(n) = datenum(time1,time_format);
            data_points(n) = data_size;
            xdatat(1:data_size, n) = xdata;
            ydatat(1:data_size, n) = ydata;

            % determining e0
            yy = sgolayfilt(ydata(20:end-20),2,21);
            dif1 = diff(yy)./diff(xdata(20:end-20));
            [~,e0_pos] = max(dif1);
            e0 = xdata(e0_pos);

            % determining ranges of pre-edge line and post-edge poly.
            % if there is a need to change, do so only on the numbers.
            [~,pre_pos_i] = min(abs(xdata - e0 + 150)); %Sample 9 = 150
            [~,pre_pos_f] = min(abs(xdata - e0 + 60)); %Sample 9 = 60
            [~,post_pos_i] = min(abs(xdata - e0 - 20)); %Sample 9 = 10
            post_pos_f = size(xdata,1) - 10;

            % create the pre-edge line
            pre = @(x,xdata) x(1) + x(2)*xdata;
            pre_xdata = xdata(pre_pos_i:pre_pos_f);
            pre_ydata = ydata(pre_pos_i:pre_pos_f);
            pre_edge = lsqcurvefit(pre,[1,1],pre_xdata,pre_ydata);

            % create the post-edge line
            post_deg = 2; % degree of the poly (1, 2 or 3)
            if post_deg == 1
                post = @(x,xdata) x(1)*xdata + x(2);
            elseif post_deg == 2
                post = @(x,xdata) x(1).*xdata.^2 + x(2).*xdata + x(3);
            else
                post = @(x,xdata) x(1)*xdata.^3 + x(2)*xdata.^2 ...
                + x(3)*xdata + x(4);
            end
            post_xdata = xdata(post_pos_i:post_pos_f);
            post_ydata = ydata(post_pos_i:post_pos_f);
            post_edge = polyfit(post_xdata,post_ydata,post_deg);

            if plt == 1
                figure(1)
                plot(xdata,ydata,'-')
                hold on
                plot(xdata(20:end-20),yy,'k')
                plot(xdata,pre(pre_edge,xdata),'r-');
                plot(xdata,post(post_edge,xdata),'r-');
                title(['\fontsize{16}',num2str(a),' sample'])
                xlabel('Energy (eV)')
                ylabel('\mu(E)')
                hold off
            end

            % flattening
            bg_xdata = xdata(e0_pos:end);
            bg_ydata = ydata(e0_pos:end);
            norm_const = post(post_edge,e0) - pre(pre_edge,e0);
            post_flat_ydata = (bg_ydata - post(post_edge,bg_xdata))...
                + post(post_edge,xdata(post_pos_i))-
pre(pre_edge,xdata(post_pos_i));
            norm_ydata = (ydata - pre(pre_edge,xdata));
            flat_ydata = [norm_ydata(1:e0_pos-1);post_flat_ydata];
            norm_ydatat(1:data_size, n) = flat_ydata./(post(post_edge,e0)-
pre(pre_edge,e0));

            % error function fitting
            x0 = [1,e0,1];
            F=@(x,xdata) x(1)*erf(xdata-x(2))+x(3);
            q = lsqcurvefit(F,x0,xdata,flat_ydata);
            heightt(n) = F(q,e0+50)-F(q,e0-50);
            if plt == 1
                figure(2)
                plot(xdata,flat_ydata)
                grid on
                hold on
                plot(xdata,F(q,xdata),'r')
                title(['\fontsize{16}',num2str(a),' sample'])
                xlabel('Energy (eV)')
                ylabel('Normalized \chi\mu(E)')
                hold off
            end

            a = a + 1;
            b = b + 1;
            n = n + 1;
        end
    end
end
%% data sorting
samples_n = nnz(index1);
index1 = index1(1:samples_n);
timet = timet(1:samples_n);
datet = datet(1:samples_n);
heightt = heightt(1:samples_n);
sqr_dif = zeros(samples_n,1);
C = cell(samples_n,1);
[n_points, tmp_pos] = max(data_points);
xdatat = xdatat(1:n_points,1:samples_n);
ydatat = ydatat(1:n_points,1:samples_n);
i_ydatat = zeros(n_points,samples_n);
norm_ydatat = norm_ydatat(1:n_points,1:samples_n);

% interpolate values to all the same positions of x
for g = 1:samples_n
    tmp_size = nnz(norm_ydatat(:,g));
    xtemp = xdatat(1:tmp_size,g);
    ytemp = norm_ydatat(1:tmp_size,g);
    i_xdata = xdatat(1:tmp_size,tmp_pos);
    i_ydata = interp1(xtemp,ytemp,i_xdata);
    i_ydatat(1:tmp_size, g) = i_ydata;
end
av_norm_ydata = sum(i_ydatat,2);
av_norm_ydata = av_norm_ydata(1:n_points);
for  n = 1:n_points
    av_norm_ydata(n) = av_norm_ydata(n)/nnz(ydatat(n,:));
end
for g = 1:samples_n
    tmp = nnz(norm_ydatat(:,g));
    sqr_dif(g) = sum((norm_ydatat(1:tmp,g) - av_norm_ydata(1:tmp)));
    if sqr_dif(g) > 5 %4
        C{g,1} = 'bad result';
    else
        C{g,1} = 'ok';
    end
end
%% plot
if plt == 1
```

```
    g = 1;%17,27,42,68,83
    figure(3)
    plot(xdatat(:,tmp_pos),av_norm_ydata)
    hold on
    tmp = nnz(norm_ydatat(:,g));
    plot(xdatat(1:tmp,g),norm_ydatat(1:tmp,g),'r')
    xlim([xdatat(1,tmp_pos) xdatat(end,tmp_pos)])
    ylim([-.2 1.2])
    title(['\fontsize{16}',num2str(index1(g)),' sample'])
    xlabel('Energy (eV)')
    ylabel('Normalized \chi\mu(E)')
    leg1 = legend('normalized data','sample data');
    set(leg1,'Location','SouthEast')
    hold off
    C{g,1}
end
%% export data
for g = 1:samples_n
    fprintf(outputfile,'%d\t',index1(g));
    fprintf(outputfile,'%s\t',datestr(datet(g),date_format));
    fprintf(outputfile,'%s\t',datestr(timet(g),time_format));
    fprintf(outputfile,'%.5f\t',heightt(g));
    fprintf(outputfile,'%s\r\n',C{g});
end
%%
fclose(outputfile);
toc
```