

Article

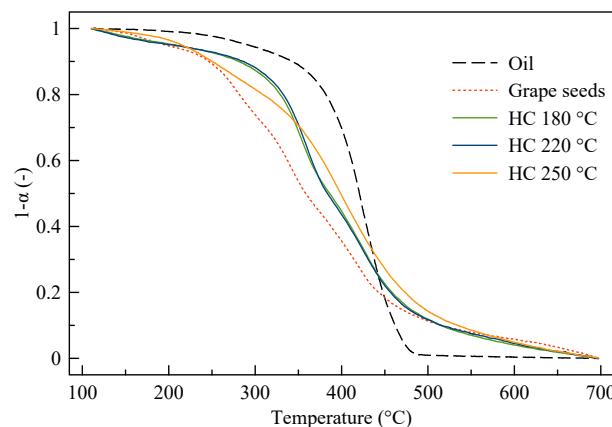
Supplementary material: Thermal analysis and kinetic modeling of pyrolysis and oxidation of hydrochars

Gabriella Gonnella ¹, Giulia Ischia ¹, Luca Fambri ² and Luca Fiori ^{1,*}

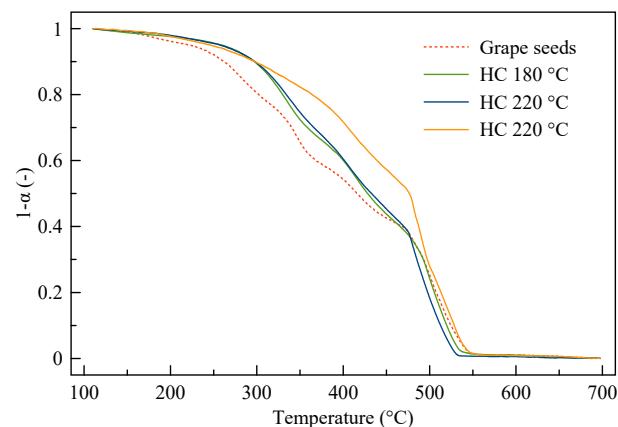
¹ Department of Civil, Environmental and Mechanical Engineering, University of Trento, 38123 Trento, Italy; ga.gonnella@gmail.com (G.G.); giulia.ischia-1@unitn.it (G.I.)

² Department of Industrial Engineering, University of Trento, Via Sommarive 9, 38123 Trento, Italy; luca.fambri@unitn.it

* Correspondence: luca.fiori@unitn.it

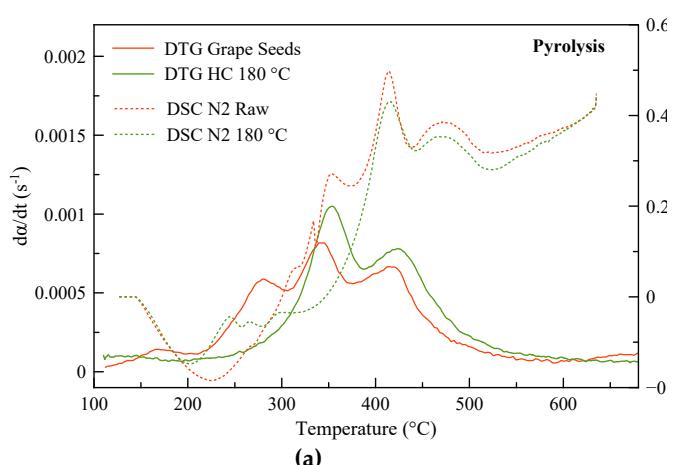


(a)

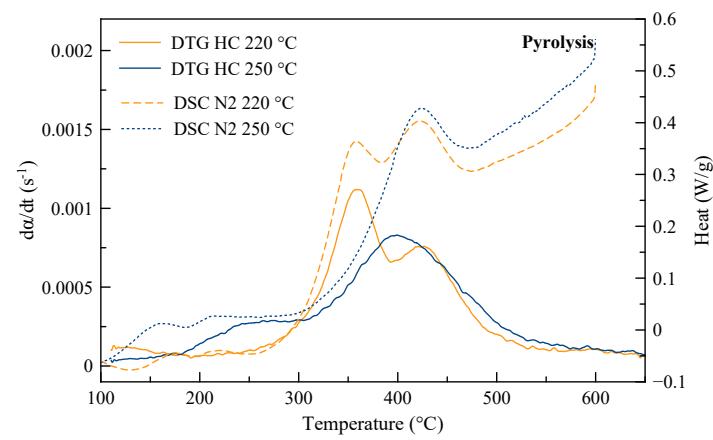


(b)

Figure S1. (1- α) at different temperatures of feedstock during: (a) pyrolysis; (b) oxidation. Heating rate: 10 °C/min.



(a)



(b)

Figure S2. Comparison between DTGA and DSC curves during pyrolysis at 10 °C/min of: (a) grape seeds and hydrochar 180 °C; (b) hydrochar 220 and 250 °C.

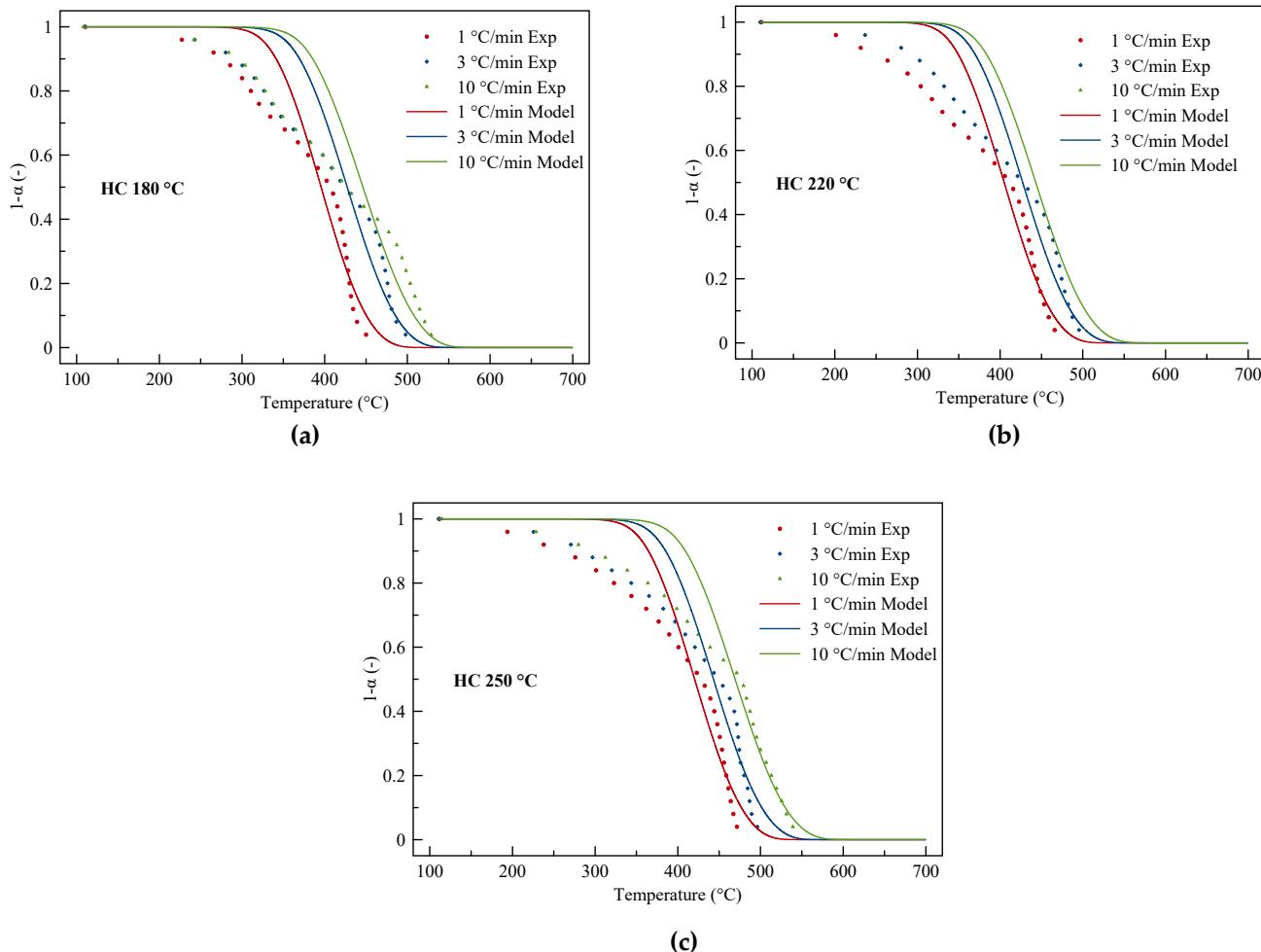


Figure S3. Comparison between experimental and predicted data ($1-\alpha$ vs temperature) computed through the Gaussian model, during oxidation of different samples: (a) hydrochar 180 °C; (b) hydrochar 220 °C; (c) hydrochar 250 °C.

Table S1. Details of Miura-Maki model, pyrolysis (on a dry basis).

α (-)	HC 180 °C			HC 220 °C			HC 250 °C		
	E (kJ/mol)	k_0 (s ⁻¹)	R ²	E (kJ/mol)	k_0 (s ⁻¹)	R ²	E (kJ/mol)	k_0 (s ⁻¹)	R ²
0.1	147	1.7E+12	0.999	123	1.4E+09	0.951	100	5.2E+07	0.995
0.2	185	7.5E+13	0.976	156	1.5E+11	0.965	110	2.3E+07	0.953
0.3	190	4.8E+13	0.969	219	1.3E+16	0.994	144	3.9E+09	0.957
0.4	214	1.9E+15	0.956	317	8.2E+23	0.997	224	5.8E+15	0.989
0.5	315	8.0E+22	0.969	502	1.1E+38	0.951	289	2.3E+20	0.999
0.6	386	3.6E+27	1.000	437	3.2E+31	0.963	349	1.8E+24	0.997
0.7	374	5.3E+25	0.989	502	2.0E+35	0.931	371	1.1E+25	0.991

Table S2. Details of Miura-Maki model, oxidation (on a dry basis).

α (-)	HC 180 °C			HC 220 °C			HC 250 °C		
	E	k_0	R^2	E	k_0	R^2	E	k_0	R^2
	(kJ/mol)	(s ⁻¹)	(-)	(kJ/mol)	(s ⁻¹)	(-)	(kJ/mol)	(s ⁻¹)	(-)
0.1	201	3.0E+16	0.975	74	1.0E+04	0.890	92	8.9E+05	0.925
0.2	275	5.9E+21	0.943	160	1.9E+11	0.885	149	8.0E+09	0.995
0.3	350	1.0E+27	0.966	196	3.9E+13	0.903	198	9.1E+12	0.987
0.4	325	1.2E+23	0.929	313	1.3E+22	0.961	221	8.7E+13	0.995
0.5	357	3.0E+24	0.936	379	8.5E+25	0.957	189	6.3E+10	1.000
0.6	186	4.6E+10	0.948	220	1.2E+13	0.987	224	1.0E+13	0.992
0.7	135	3.4E+06	0.974	194	7.4E+10	0.985	225	6.9E+12	1.000
0.8	123	3.5E+05	0.981	185	1.1E+10	0.993	188	1.0E+10	0.997
0.9	115	6.6E+04	0.990	180	3.0E+09	0.999	164	1.5E+08	0.993

Gaussian DAEM – MATLAB code

```
% The following code was developed in order to determine
% pyrolysis kinetic parameters through a Gaussian DAEM.
% The code was developed by Gabriella Gonnella.

alpha = ar_a; %calling experimental data from TGA
Temp = Tr_a; %calling experimental data from TGA
beta = 10/60; %calling experimental data from TGA
k0 = 1.67*10^13;
R = 8.314*10^-3;
%for loop to calculate conversion rate
alpha1_tga = zeros(1,length(alpha));

for j = 1:length(alpha)
    alpha1_tga(j) = (alpha(j)-min(alpha))./(max(alpha)-min(alpha));
    %alpha1_tga(j)=alpha1(j);
    %alpha1_tga(j) = 1 - alpha1_tga(j);
end

T1=Temp+273.15;
E1= 150;
E2 = 300;
Estep = 5;

sigma1 = 10;
sigma2 = 70;
sigmastep = 5;
Matrix = zeros(size(E1:Estep:E2,2)*size(sigma1:sigmastep:sigma2,2), 3); % initialize the
matrix
count = 1;
%for loop calling solvedaem function - large grid step
for E0 = E1:Estep:E2
    for sigma = sigma1:sigmastep:sigma2
        h2 = solvedaem(T1,beta,k0,R,E0,sigma,alpha1_tga);
        Matrix(count, :) = [E0,sigma,h2];
        count = count + 1;
    end
end

B = sortrows(Matrix,3);

minMat = B(1:3,:)
count=1;
%for loop calling solvedaem function - small grid step
for i = 1:size(minMat,1)
    E1 = minMat(i,1)-5;
    E2 = minMat(i,1)+5;
```

```
sigma1 = minMat(i,2)-5;
sigma2 = minMat(i,2)+5;

for E0 = E1:E2
    for sigma = sigma1:sigma2
        h2 = solvedaem(T1,beta,k0,R,E0,sigma,alpha1_tga);
        Matrix1(count, :) = [E0,sigma,h2];
        count = count + 1;
    end
end
B = sortrows(Matrix1,3);

%final values of E0 and sigma
E0 = B(1,1)
sigma = B(1,2)
h2 = B(1,3)

%
[alpha_daem,h2] = finaltga(T1,beta,k0,R,E0,sigma,alpha1_tga);

alpha_daem = (alpha_daem-min(alpha_daem))./(max(alpha_daem)-min(alpha_daem));
M=100;
N = M/4;
%for loop to calculate discretize values of experimental alpha
for n = 1:N
val = M/N*n/100;
[m,h1] = min(abs(alpha1_tga-val));
alphasel1(n) = alpha1_tga(h1);
T_fix1(n)=T1(h1);
end
plot(T1-273.15,alpha_daem)
hold on
grid on
%plot(T1-273.15,alpha1_tga)
plot(T_fix1 - 273.15,alphasel1, 'o');
xlabel 'Temperature [°C]'
ylabel '1- $\alpha$  [-]'
%}
%legend
```

```
% function solving the integral with Simpson's 1/3 rule
function [h2] = solvedaem(T1,beta,k0,R,E0, sigma,alpha1_tga)
h2 = 0;
f = @(x) (exp(-k0./beta.*R.*T1.^2./x.*exp(-x./R./T1)))./(sigma.*sqrt(2*pi)).*exp(-(x-E0).^2./(2.*sigma.^2));
```

```
a = E0-25; b = E0+25; %lower and upper limits
nSimp = 100; %number of intervals
hSimp = (b - a)/nSimp; %length of each interval
s = 0;

for i=0:nSimp
    if i==0 || i==nSimp
        p = 1;
    elseif mod(i,2) ~= 0
        p = 4;
    else
        p = 2;
    end
    x = a + i*hSimp;
    s = s + p*f(x);
end
alpha_daem = hSimp/3*s;
h2 = h2+sqrt((sum((alpha1_tga - alpha_daem).^2)/length(T1)));
disp(h2);
disp(sigma);
disp(E0);
end
```