

---

## Table of Contents

.....	1
Read the data and parameters .....	1
Compute first convolution between $\exp(-\alpha_1 t)$ and $C_p(t)$ .....	2
Compute second convolution between $\exp(-\alpha_2 t)$ and $C_p(t)$ .....	2
Plot $C_p$ and $C_t$ versus time. ....	2
Interpretation of the plot .....	3
get_parameters .....	3
convolution_by_integration .....	4
alpha_function .....	5
import_data .....	5

```
% Description
% Compute the total concentration
% of FDG in the brain over time,  $C_i(t)$ , by performing a
% numerical integration.
%  $C_i(t) = C_e(t) + C_m(t)$ 
% where
% -  $C_e(t)$ : "free" FDG in the brain tissue
% -  $C_m(t)$ : FDG concentration "trapped" in brain tissue
% However in the first part of the project, we showed that the
%  $C_i(t)$  can be expressed as a sum of two convolutions:
% =>  $A * \text{conv}(\exp(-\alpha_1 * t), C_p(t))$ 
% =>  $B * \text{conv}(\exp(-\alpha_2 * t), C_p(t))$ 
% where  $C_p(t)$  is the FDG concentration in the arterial system.

% The full expression of  $C_i(t)$  and values for
%  $\alpha_1$ ,  $\alpha_2$ ,  $C_i(t)$ ,  $A$ , and  $B$  are given in Brooks paper:
% equation (3), (4), (5) and (6).
% Next we perform a numerical integration to compute the two sums
% above
% to obtain  $C_i$ .
% Finally with  $C_i(t)$ , we plot  $C_p(t)$  and  $C_i(t)$  versus time, and make
% observations related to the plot.

% Clean environment
clear all;
close all;
clc;
format long g
```

## Read the data and parameters

```
[ts, cp, alpha_1, alpha_2, A, B] = get_parameters();
```

---

## Compute first convolution between $\exp(-\alpha_1 t)$ and $C_p(t)$

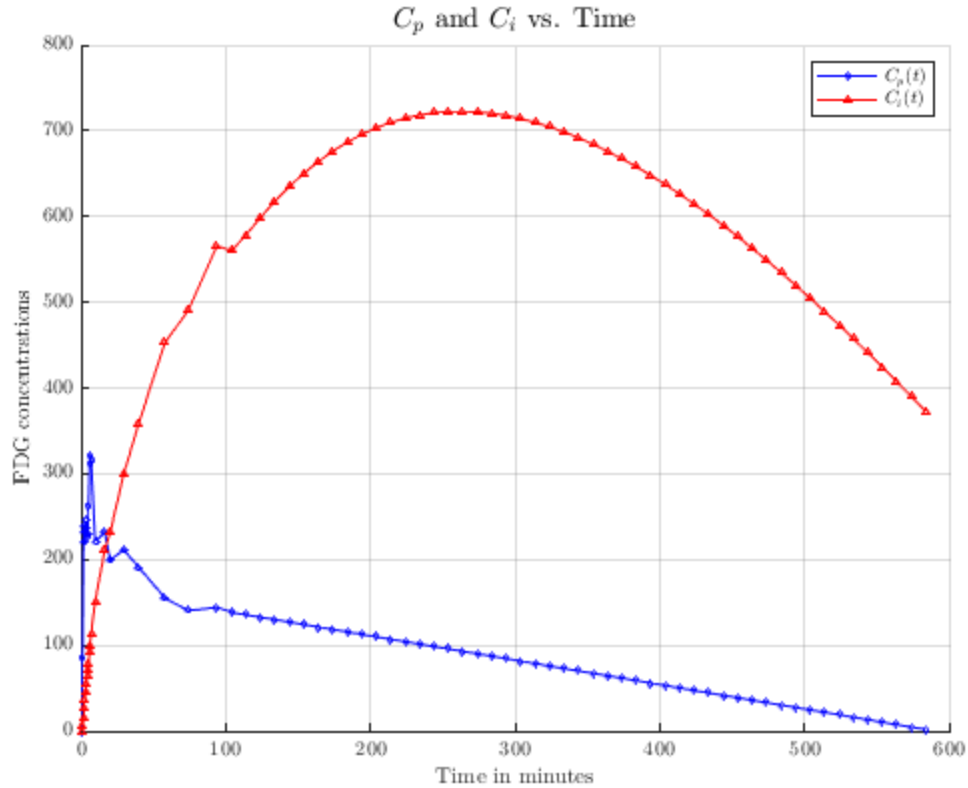
```
conv_alpha_1 = convolution_by_integration(cp, ts, alpha_1);
```

## Compute second convolution between $\exp(-\alpha_2 t)$ and $C_p(t)$

```
conv_alpha_2 = convolution_by_integration(cp, ts, alpha_2);  
ci = A .* conv_alpha_1 + B .* conv_alpha_2;
```

## Plot $C_p$ and $C_t$ versus time.

```
figure, hold on, grid on  
  
plot(ts, cp, "b-o", 'LineWidth',1,'MarkerSize',3);  
plot(ts, ci, "r-^", 'LineWidth',1,'MarkerSize',3);  
  
title('$C_p$ and $C_i$ vs. Time','FontSize', 14);  
xlabel('Time in minutes');  
ylabel('FDG concentrations');  
legend('$C_p(t)$','$C_i(t)$');  
saveas(gcf,'cp_ci_vs_time','pdf')
```



## Interpretation of the plot

As the FDG is injected into the blood, the concentration of FDG into the brain increases rapidly to reach a peak after 250 minutes or four hours of the initial injection of FDG into the blood and then slowly decreases as the FDG is eliminated from the blood by urination. In order to have the best PET tracing, in this experiment, scanning has to be performed around 4 hours after the injection of FDG into the blood.

## get\_parameters

```
function [ts, cp, alpha_1, alpha_2, A, B] = get_parameters(workbook,
    worksheet)
% Description
% Set the parameters needed to determine Ci(t).
% The parameters are given in PET Scan Brooks paper.
% Inputs:
% workbook: Excel workbook with the values points for Cp(t)
% extended pass 94 minutes.
% worksheet: Excel worksheet where the data is
% Outputs:
% ts: time in minutes when Cp data is sampled.
% Cp: FDG concentration in arterial system.
% alpha_1: computed value in equation (3) of Brooks paper.
% alpha_2: computed value in equation (3) of Brooks paper.
% A: computed value in equation (3) of Brooks paper.
% B: computed value in equation (3) of Brooks paper.
```

---

```

if ~exist('workbook','var')
    workbook = "./project_1_extended_data.xlsx";
    worksheet = "Sheet1";
end
data = import_data(workbook, worksheet);

% Cp(t) is given from an experiment running from 0 to 94 minutes
% It is then extended pass 94 minutes with a least-square regression
% using the the samples starting at 58 minutes up to 584 minutes
% when the "extended" concentration Cp(t) is almost zero.

cp = data.Concentration(2:end);
ts = data.Time(2:end);
% k1, k2 ,k3 , k4 rate constants
k1 = 0.102;
k2 = 0.130;
k3 = 0.062;
k4 = .0068;
d = sqrt((k2 + k3 + k4)^2 -4 * k2 *k4);
% Alpha_1, Alpha_2, A, B
alpha_1 = 0.5 * (k2 + k3 + k4 - d);
alpha_2 = 0.5 * (k2 + k3 + k4 + d);
A = (k1 * (k3 + k4 - alpha_1)) / (alpha_2 - alpha_1);
B = (k1 * (alpha_2 -k3 -k4)) / (alpha_2 - alpha_1);

end

```

## convolution\_by\_integration

```

function conv_res = convolution_by_integration(cp, ts, alpha)
% Description
% Performs a convolution between the exponential function
% parametrized by alpha with concentration Cp over time.
% Inputs:
% Cp: FDG concentration in arterial system.
% ts: time in minutes when Cp data is sampled.
% alpha: one of the computed values in equation (3) of Brooks paper.
% Ouput:
% conv_res: convolution between  $\exp(-\alpha t)$  and  $Cp(t)$ .

% To compute the integral, we use the trapezoidal method as described
% here:
% https://www.mathworks.com/help/matlab/ref/trapz.html
% We use two loops:
% - first loop is indexed by i, and use the time ts(i)
% for which we want to determine the value of the convolution
% conv_res(ts(i))
% - second loop concerns the "running" or integration variable,
% indexed by ts(j).
% For each ts(i)
%     total_area = 0
%     For each ts(j)

```

---

```

%           area = compute integral  between ts(j), ts(j-1)
%           total_area = total_area + area
%       End
% End

n = size(ts,1);
conv_res = zeros(n,1);
for i=1: n
    total_area = 0;
    for j=2: i
        dt_j = ts(j) - ts(j-1);
        f_value_j_1 = alpha_function(i, j-1, alpha, cp, ts);
        f_value_j = alpha_function(i, j, alpha, cp, ts);
        area = dt_j * ((f_value_j_1 + f_value_j)/2);
        total_area = total_area + area;
    end
    conv_res(i) = total_area;
end
end

```

## alpha\_function

```

function f_value = alpha_function(t1, t2, alpha, cp, ts)
% Description
% Compute the value of  $\exp(-\alpha * (t_2 - t_1)) * C_p(t_2)$ 
% Inputs:
%   t1: index of time of sampling.
%   t2: index of "running "or integration time variable .
%   alpha: one of the computed values in equation (3) of Brooks paper.
%   Cp: FDG concentration in arterial system.
%   ts: time in minutes when Cp data is sampled.
% Output:
%    $\exp(-\alpha * (ts(t_1) - ts(t_2))) * C_p(t_2)$ 
    diff_t = ts(t1) - ts(t2);
    exp_f = exp(-alpha * diff_t);
    f_value = exp_f * cp(t2);
end

```

## import\_data

```

function projectextendeddata = import_data(workbookFile, sheetName,
    dataLines)
% import_data Import data from a spreadsheet
% data = import_data(FILE) reads data from the first
% worksheet in the Microsoft Excel spreadsheet file named
% workbookFile.
% Returns the data as a table.
%
% data = import_data(FILE, SHEET) reads from the
% specified worksheet.
%
% data = import_data(FILE, SHEET, DATALINES) reads from
% the specified worksheet for the specified row interval(s). Specify

```

---

```

% DATALINES as a positive scalar integer or a N-by-2 array of
% positive
% scalar integers for dis-contiguous row intervals.
%
% Example:
% data = importfile("./project_1_extended_data.xlsx", "Sheet1", [1,
% 71]);
%

% Input handling

% If no sheet is specified, read first sheet
if nargin == 1 || isempty(sheetName)
    sheetName = 1;
end

% If row start and end points are not specified, define defaults
if nargin <= 2
    dataLines = [1, 71];
end

% Set up the Import Options and import the data
opts = spreadsheetImportOptions("NumVariables", 3);

% Specify sheet and range
opts.Sheet = sheetName;
opts.DataRange = "A" + dataLines(1, 1) + ":C" + dataLines(1, 2);

% Specify column names and types
opts.VariableNames = ["VarName1", "Time", "Concentration"];
opts.VariableTypes = ["double", "double", "double"];

% Import the data
projectextendeddata = readtable(workbookFile, opts, "UseExcel",
    false);

for idx = 2:size(dataLines, 1)
    opts.DataRange = "A" + dataLines(idx, 1) + ":C" + dataLines(idx,
    2);
    tb = readtable(workbookFile, opts, "UseExcel", false);
    projectextendeddata = [projectextendeddata; tb]; %#ok<AGROW>
end

end

```

*Published with MATLAB® R2021a*