Table of Contents

P11 : Cortés y García

```
clear all;
format long;
```

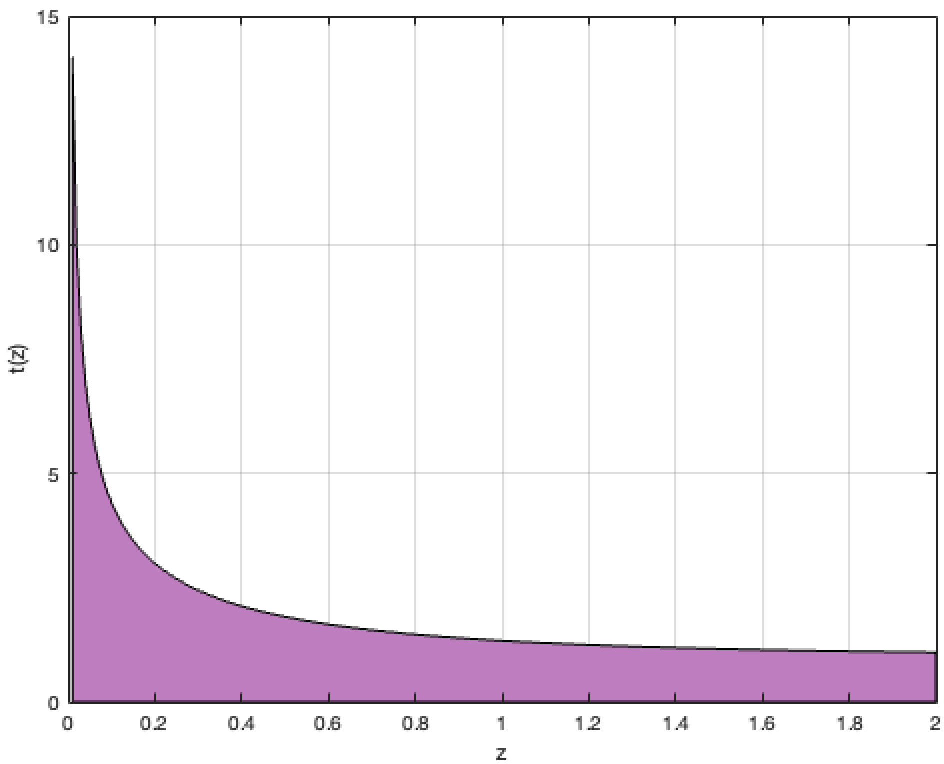
Cálculo de T(2 m)

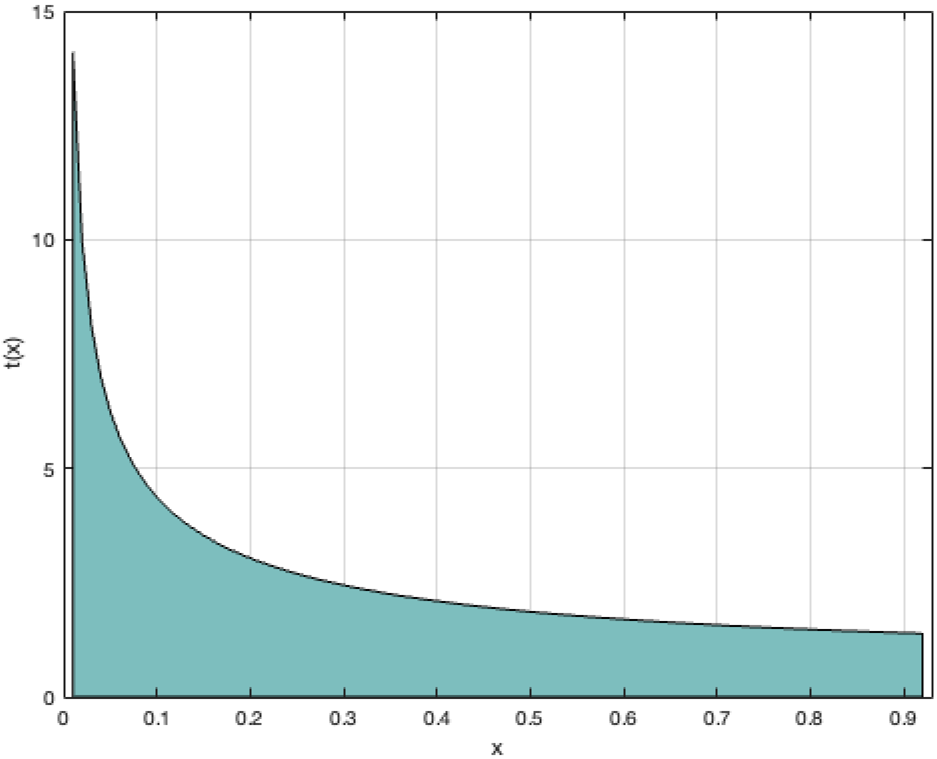
```
% Del enunciado tenemos que f(z) = exp(-z) [f(0) = 1]
% Por lo tanto, f'(z) = -exp(-z)
% Esto implica que la función t(z) que queremos integrar es
tz = @(z) sqrt((1+exp(-2.*z))./(1-exp(-z)));

figure(1)
plot([0:0.01:2],tz(0:0.01:2),'LineWidth',2, 'Color', [0.25 0.5 0.75])
area([0:0.01:2], tz(0:0.01:2),'FaceColor',[0.75 0.5 0.75]);
xlim([0 2]);
xlabel('z')
ylabel('t(z)')
grid on

% cambio de variable
% El cambio de variable que haremos es x^2 = 1 - exp(-z)
% Así, la función que queremos integrar queda como:
tx = @ (x) 2*sqrt(1 + (1 - x.^2).^2)./(1 - x.^2);

figure(2)
area([0:0.01:sqrt(1-exp(-2))], tz(0:0.01:sqrt(1-exp(-2))),'FaceColor',
[0.50 0.75 0.75]);
xlim([0 sqrt(1-exp(-2))]);
xlabel('x')
ylabel('t(x)')
grid on

% Podemos apreciar que la forma de las gráfica es similar. Sin
 embargo, si
% han cambiado los límites de integración: esto tiene sentido puesto
 que
% al hacer el cambio de variable, el valor de la integral (luego el
 área
% de la función) debe mantenerse y, sin embargo, el cambio de variable
 debe
% aplicarse también a los límites de integración, por lo que estos son
% diferentes.
```

```
% resolución numérica de la integral
% Resolvemos la integral con la Cuadratura de Clenshaw-Curtis
a = 0; b = sqrt(1-exp(-2));         % limites de integracion
n = 50;                             % numero de nodos
g = 9.81;
I = (1/sqrt(2*g)) * qclencurt(a, b, n, tx);
disp(I);

% resolución mediante tanh
a = 0; b = 2;                       % limites de integracion
c = 5;                              % scaling factor
n = 50;                             % numero de nodos
I_tan = (1/sqrt(2*g)) * qtanh(n, a, b, c, tz);
disp(I_tan)

% Obvservamos que el valor de la integral calculada por los dos
 metodos
% (Clensaw-Curtis y tanh) difiere en un orden de 10^(-8), por lo que
% podemos dar por valido el valor de estas.
```

0.878920993467084

0.878920977941525





FUNCIONES IMPLEMENTADAS

```
% qclencurt.m
% Code 8: Clenshaw-Curtis Quadrature Function
% Input:  a-b (low-up lim)
%         n (# intervals)
%         fun (function name)
% Output: I_n(fun)
% function Icc = qclencurt(a, b, n, fun)
%     l = [0:n]';
%     k = [2:n]';
%     x = cos(l*pi/n);
%
%     w = cos(l*l'*pi/n) \ [2; 0; (1 + (-1).^k)./(1-k.^2)];
%
%     z = a + .5*(b-a)*(x+1);
%     f = feval(fun, z);
%
%     Icc = .5*(b-a)*w'*f;
% end

% qtanh.m
% Code 10: tanh-rule for 2nd kind improper integrals (-1, +1)
% Input:    n (3 abcisses)
%           a-b (integration domain)
```

```
%           c (tanh scaling factor)
% Output:   I_n(f)
% function In = qtanh(n, a, b, c, fun)
%     h = c/sqrt(n);
%     u = [-n:n]*h/2;
%
%     x = (b-a)*.5*(tanh(u) + 1) + a;
%     f = feval(fun, x);
%
%     In = (b-a)*(.25*h)*sum(f./(cosh(u).^2));
% end
```

Published with MATLAB® R2020b