x(t)= e^−2t .μ(t) +e^−t.cos(3t)μ(t)

Para el algoritmo, la expresión se escribe como

ft = sym.exp(-2\*t)\*u + sym.exp(-t)\*sym.cos(3\*t)\*u

f(t):

-t -2\*t

e \*cos(3\*t)\*Heaviside(t) + e \*Heaviside(t)

F(s):

2

2\*s + 5\*s + 12

---------------------

3 2

s + 4\*s + 14\*s + 20

>>>

Para disponer de expresiones mas simples de F(s) en fracciones parciales, se añade la instrucción

Fs = sym.apart(Fs) # separa en fracciones parciales

con lo que se obtiene el siguiente resultado para F(s),

F(s):

s + 1 1

------------- + -----

2 s + 2

s + 2\*s + 10

>>>

El primer componente de la suma corresponde a la parte de sym.exp(-t)\*sym.cos(3\*t) y la segunda parte corresponde a sym.exp(-2\*t)

# Transformadas de Laplace Unilateral con Sympy

# supone f(t) causal, usa escalón u(t)

import sympy as sym

# INGRESO

t = sym.Symbol('t', real=True)

s = sym.Symbol('s')

u = sym.Heaviside(t)

d = sym.DiracDelta(t)

# coeficientes como racional en dominio 'ZZ' enteros

k1 = sym.Rational(1/3).limit\_denominator(100)

k2 = sym.Rational(4/3).limit\_denominator(100)

ft = sym.exp(-a\*t)\*u

ft = 3\*sym.exp(-2\*t)\*u + sym.exp(-t)\*sym.cos(3\*t)\*u

ft = u - u.subs(t,t-2)

# PROCEDIMIENTO

def laplace\_transform\_suma(ft):

'''transformada de Laplace de suma de terminos

separa constantes para conservar en resultado

'''

def separa\_constante(termino):

''' separa constante antes de usar

sym.laplace\_transform(term\_suma,t,s)

para incorporarla luego de la transformada

inconveniente revisado en version 1.11.1

'''

constante = 1

if termino.is\_Mul:

factor\_mul = sym.Mul.make\_args(termino)

for factor\_k in factor\_mul:

if not(factor\_k.has(t)):

constante = constante\*factor\_k

termino = termino/constante

return([termino,constante])

# transformadas de Laplace por términos suma

ft = sym.expand(ft) # expresion de sumas

ft = sym.powsimp(ft) # simplifica exponentes

term\_suma = sym.Add.make\_args(ft)

Fs = 0

for term\_k in term\_suma:

[term\_k,constante] = separa\_constante(term\_k)

Fsk = sym.laplace\_transform(term\_k,t,s)

Fs = Fs + Fsk[0]\*constante

# separa exponenciales constantes

Fs = sym.expand\_power\_exp(Fs)

return(Fs)

Fs = laplace\_transform\_suma(ft)

# SALIDA

print('\n f(t): ')

sym.pprint(ft)

print('\n F(s): ')

sym.pprint(Fs)