T16_Sympy

January 31, 2021

1 Python de cero a experto

Autor: Luis Miguel de la Cruz Salas

1.1 SymPy

SymPy, es una biblioteca que permite realizar cálculo símbolico, también llamada sistemas de álgebra computacional (CAS por sus siglas en inglés). Es un proyecto de software libre, cuenta con una gran documentación y una excelente comunidad. Está escrita por completo en Python, por lo tanto no tiene dependencias externas. La documentación completa la puede consultar aquí .

Entre otras cosas, con SymPy es posible realizar lo siguiente:

- Simplificación de expresiones.
- Cálculo de integrales.
- Cálculo de derivadas.
- Cálculo de límites.
- Solución de ecuaciones.
- Álgebra Lineal.
- Estadística.
- Combinatoria.
- Física.
- Teoría de números.
- Geometría.
- Lógica.

Es posible evaluar expresiones de SymPy en internet en el sitio: http://gamma.sympy.org/ de manera similar al proyecto *Wolfram Alpha*.

1.1.1 Ejemplos

[1]:
```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

import sympy        # Importamos el modulo de Sympy
from sympy import pi, E, I, oo
```

```
sympy.init_printing()           # Para mostar graficamente las expresiones␣
↪presentadas
```

La línea 2 de la celda anterior tiene por objeto la impresión de las fórmulas matemáticas en formato LaTeX.

1.2 Números especiales

Python evalúa las operaciones aritméticas usando una representación numérica que introduce errores de redondeo. Se puede trabajar directamente con la representación matemática de algunos números:

[2]:
```
sympy.Rational(1,3) # Define un numero racional
```

[2]: $\frac{1}{3}$

[3]:
```
sympy.pi      # representación del número Pi
```

[3]: π

[4]:
```
pi.evalf()      # Valor de pi
```

[4]: 3.14159265358979

[5]:
```
3 + 4 * I  # Número complejo
```

[5]: $3 + 4i$

[6]:
```
E ** 2  # Número de Euler
```

[6]: e^2

[7]:
```
oo # Numero infinito
```

[7]: ∞

[8]:
```
oo+1 # Numero infinito
```

[8]: ∞

1.3 Símbolos

Son etiquetas que se usan en operaciones simbólicas. Se deben definir antes de usarse. Se definen usando la función **symbols**. Las operaciones entre símbolos devuelven símbolos.

[9]:
```
x = sympy.symbols('x') # símbolo x
x
```

[9]: x

[10]:
```
y = sympy.symbols('y') # símbolo y
y
```

[10]: y

[11]:
```
print(type(x), type(y), sep='\n')
```

```
<class 'sympy.core.symbol.Symbol'>
<class 'sympy.core.symbol.Symbol'>
```

[12]:
```
alpha, gamma = sympy.symbols('alpha gamma') # varios símbolos a la vez
alpha, gamma
```

[12]: $(\alpha,\ \gamma)$

[13]:
```
T_1 = sympy.symbols('T_1')   # símbolos con subíndices
T_1
```

[13]: T_1

RECOMENDACIÓN: las etiquetas y los símbolos deberían coincidir.

1.4 Expresiones

[14]:
```
z = x + x - 3*y + 2*x          # Podemos aplicar operaciones algebraicas
print(type(z))
z
```

```
<class 'sympy.core.add.Add'>
```

[14]: $4x - 3y$

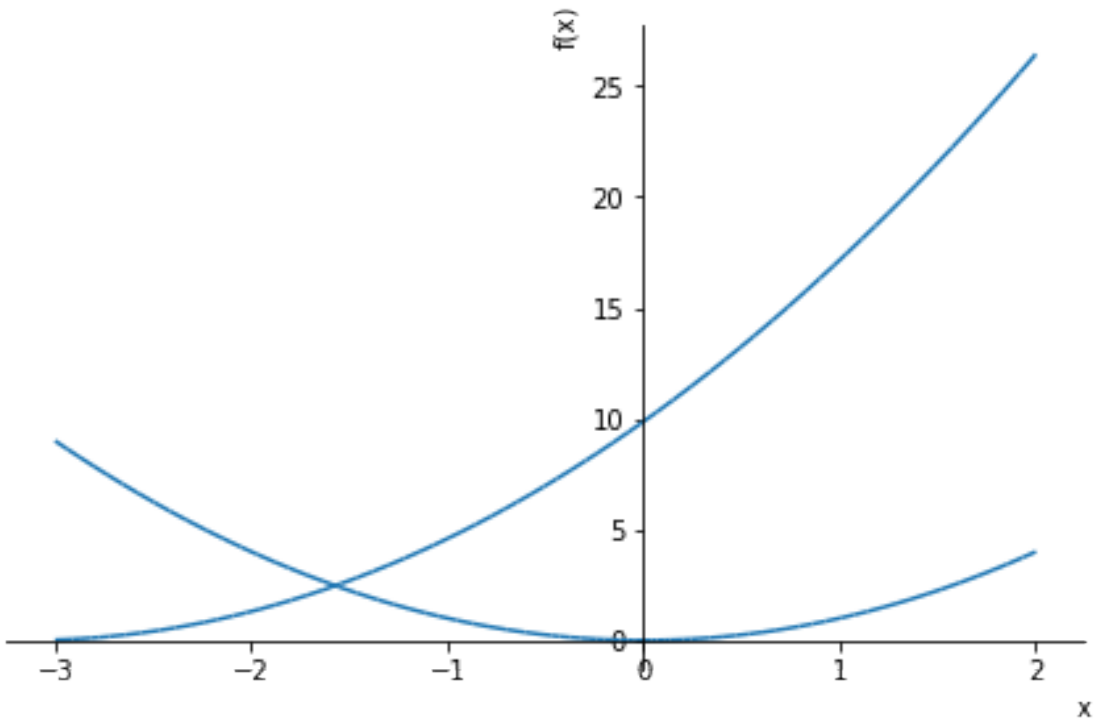
[15]:
```
t = (x + pi)**2
t
```

[15]: $(x + \pi)^2$

[16]:
```
lam = sympy.symbols('lambda')
lam
```

[16]: λ

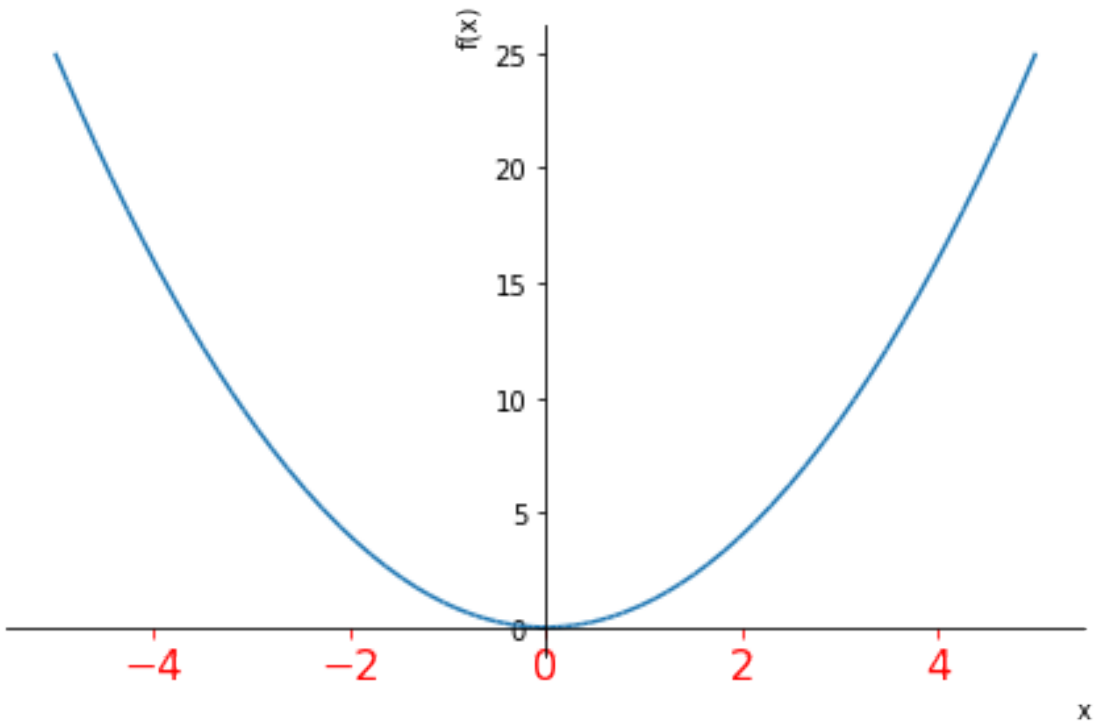
[17]:
```
sympy.plot(t, x**2, (x,-3,2))
```



[17]:
```
<sympy.plotting.plot.Plot at 0x7f39283e5910>
```

[18]:
```
plt.rcParams["xtick.labelsize"] = 16
plt.rcParams["xtick.color"] = "red"
```

[19]:
```
p = sympy.plot(x**2, (x, -5, 5))
```



1.5 Expansión de expresiones

[20]:
```
sympy.expand((x+y)**3)
```

[20]: $x^3 + 3x^2y + 3xy^2 + y^3$

1.6 Fracciones Parciales

[21]:
```
expr = (4*x**3 + 21*x**2 + 10*x + 12)/(x**4 + 5*x**3 + 5*x**2 + 4*x)
expr
```

[21]: $\frac{4x^3 + 21x^2 + 10x + 12}{x^4 + 5x^3 + 5x^2 + 4x}$

[22]:
```
sympy.apart(expr)
```

[22]: $\frac{2x - 1}{x^2 + x + 1} - \frac{1}{x + 4} + \frac{3}{x}$

1.7 Factorización

[23]:
```
sympy.factor(x**3 - x**2 + x - 1)
```

[23]: $(x - 1)\left(x^2 + 1\right)$

1.8 Simplificación

[24]:
```
sympy.simplify((x+x*y)/x)
```

[24]: $y + 1$

[25]:
```
sympy.simplify(sympy.cos(x)**2 + sympy.sin(x)**2)
```

[25]: 1

1.9 Substitución

[26]:
```
y = sympy.sin(x)**2 + sympy.cos(x)**2
y
```

[26]: $\sin^2(x) + \cos^2(x)$

[27]:
```
y.subs(x, x**2)  # Sustitucion de código.
```

[27]: $\sin^2(x^2) + \cos^2(x^2)$

[28]:
```
y.replace(sympy.sin, sympy.exp) # Sirve para reemplazar una función por otra.
```

[28]: $e^{2x} + \cos^2(x)$

1.10 Evaluación

[29]:
```
y = sympy.sqrt(2) * pi
y
```

[29]: $\sqrt{2}\pi$

[30]:
```
sympy.N(y)
```

[30]: 4.44288293815837

[31]:
```
y.evalf()
```

[31]: 4.44288293815837

[32]:
```
sympy.N(y)
```

[32]: 4.44288293815837

[33]:
```
x = sympy.symbols('x')
f = sympy.pi * x**2 + x / 3
```

[34]:
```
sympy.N(f,10)
```

[34]: $3.141592654x^2 + 0.3333333333x$

[35]:
```
f.evalf(n=4)
```

[35]: $3.142x^2 + 0.3333x$

1.11 Resolución de ecuaciones

[36]:
```
x = sympy.symbols('x')
sympy.solve(x**4 - 1, x)  # Una sola variable
```

[36]: $[-1,\ 1,\ -i,\ i]$

[37]:
```
y = sympy.symbols('y')
sympy.solve([x + 5*y - 2, -3*x + 6*y - 15], [x, y]) # Dos variables
```

[37]: $\{x : -3,\ y : 1\}$

[38]:
```
sympy.solve(sympy.exp(x) + 1, x)
```

[38]: $[i\pi]$

[39]:
```
ec = sympy.Eq(x**2 + 2*x, 1)
ec
```

[39]: $x^2 + 2x = 1$

[40]:
```
sympy.solve(ec,x)
```

[40]: $\left[-1 + \sqrt{2},\ -\sqrt{2} - 1\right]$

[41]:
```
c = sympy.symbols('c')
c
```

[41]: c

[42]:
```
ec1 = sympy.Eq(x**2 + 2*x, c)
ec1
```

[42]: $x^2 + 2x = c$

[43]:
```
sympy.solve(ec1,x)
```

[43]: $\left[-\sqrt{c+1} - 1,\ \sqrt{c+1} - 1\right]$

1.12 Cálculo

1.12.1 Limites

[44]:
```
sympy.limit(sympy.sin(x)/x, x, 0)  # Aplicamos un limite
```

[44]: 1

[45]:
```
sympy.limit(x/x,x, sympy.oo)       # Limites al infinito
```

[45]:
$$1$$

[46]:
```
sympy.limit(1 / x, x, sympy.oo)
```

[46]:
$$0$$

[47]:
```
sympy.limit((x + 1)*(x + 2)*(x + 3)/x**3, x, sympy.oo)
```

[47]:
$$1$$

[48]:
```
sympy.limit(sympy.tan(x), x, pi / 2, dir='+')
```

[48]:
$$-\infty$$

[49]:
```
sympy.limit(sympy.tan(x), x, pi / 2, dir='-')
```

[49]:
$$\infty$$

1.12.2 Derivadas

[50]:
```
sympy.diff(1/(x**2+x+1),x)     # Derivada con respecto a x
```

[50]:
$$\frac{-2x-1}{\left(x^2+x+1\right)^2}$$

[51]:
```
sympy.diff(1/(x**2+y+1),y)      # Derivada con respecto a y
```

[51]:
$$-\frac{1}{\left(x^2+y+1\right)^2}$$

[52]:
```
sympy.diff(x**2, x, 2)          # Segunda derivada con respecto a x
```

[52]:
$$2$$

[53]:
```
sympy.diff(1/(x**2+y+1),y,3)    # Tercera derivada con respecto a y
```

[53]:
$$-\frac{6}{\left(x^2+y+1\right)^4}$$

[54]:
```
sympy.diff(x * y * sympy.log(x * y), x, y)  # Derivada parcial con respecto a x␣
↪e y
```

[54]:
$$\log\left(xy\right)+2$$

1.13 Series de Taylor

[55]:
```
sympy.series(sympy.cos(x))    # Punto central en cero
```

[55]:
$$1-\frac{x^2}{2}+\frac{x^4}{24}+O\left(x^6\right)$$

[56]:
```
sympy.series(sympy.cos(x),n=10) # Calcula hasta orden 10 de la variable x
```

[56]:
$$1 - \frac{x^2}{2} + \frac{x^4}{24} - \frac{x^6}{720} + \frac{x^8}{40320} + O\left(x^{10}\right)$$

[57]:
```
sympy.series(sympy.cos(x),n=10).removeO() #
```

[57]:
$$\frac{x^8}{40320} - \frac{x^6}{720} + \frac{x^4}{24} - \frac{x^2}{2} + 1$$

[58]:
```
sympy.series(sympy.cos(x), n=5, x0=pi/2)
```

[58]:
$$\frac{\pi}{2} + \frac{\left(x - \frac{\pi}{2}\right)^3}{6} - x + O\left(\left(x - \frac{\pi}{2}\right)^5; x \to \frac{\pi}{2}\right)$$

1.14 Integración

[59]:
```
sympy.integrate(6*x**5, x)
```

[59]:
$$x^6$$

[60]:
```
sympy.integrate(sympy.cos(x), (x, -pi/2, pi/2)) # Integrales definidas
```

[60]:
$$2$$

[61]:
```
sympy.integrate(sympy.exp(-x), (x, 0, oo))  #Integrales impropias
```

[61]:
$$1$$

[62]:
```
sympy.integrate(sympy.exp(-x**2), (x, -oo, oo)) #Integrales impropias
```

[62]:
$$\sqrt{\pi}$$

1.15 Algebra lineal

http://docs.sympy.org/latest/modules/matrices/matrices.html#creating-matrices

[63]:
```
M = sympy.Matrix([[3,1], [0,1]])
M
```

[63]:
$$\begin{bmatrix} 3 & 1 \\ 0 & 1 \end{bmatrix}$$

[64]:
```
M.eigenvals()
```

[64]:
$$\{1 : 1,\ 3 : 1\}$$

[65]:
```
M.inv()
```

[65]:
$$\begin{bmatrix} \frac{1}{3} & -\frac{1}{3} \\ 0 & 1 \end{bmatrix}$$

[66]:
```
M*M.inv()
```

[66]:
$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

[67]:
```
A = sympy.Matrix([[1,x], [y,1]])
A
```

[67]:
$$\begin{bmatrix} 1 & x \\ y & 1 \end{bmatrix}$$

[68]:
```
phi = sympy.symbols('phi')
rotation = sympy.Matrix([[sympy.cos(phi), -sympy.sin(phi)],
                         [sympy.sin(phi), sympy.cos(phi)]])
rotation
```

[68]:
$$\begin{bmatrix} \cos(\phi) & -\sin(\phi) \\ \sin(\phi) & \cos(\phi) \end{bmatrix}$$

[69]:
```
rotation.subs('phi',2).evalf(14)
```

[69]:
$$\begin{bmatrix} -0.41614683654714 & -0.90929742682568 \\ 0.90929742682568 & -0.41614683654714 \end{bmatrix}$$

[70]:
```
rotation.T # Transpuestarotation.T*rotation
```

[70]:
$$\begin{bmatrix} \cos(\phi) & \sin(\phi) \\ -\sin(\phi) & \cos(\phi) \end{bmatrix}$$

[71]:
```
rotation.T*rotation
```

[71]:
$$\begin{bmatrix} \sin^2(\phi) + \cos^2(\phi) & 0 \\ 0 & \sin^2(\phi) + \cos^2(\phi) \end{bmatrix}$$

[72]:
```
sympy.simplify(rotation.T*rotation - sympy.eye(2))
```

[72]:
$$\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

[73]:
```
rotation.inv()
```

[73]:
$$\begin{bmatrix} -\frac{\sin^2(\phi)}{\cos(\phi)} + \frac{1}{\cos(\phi)} & \sin(\phi) \\ -\sin(\phi) & \cos(\phi) \end{bmatrix}$$

[74]:
```
sympy.simplify(rotation.T - rotation.inv())
```

[74]:
$$\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

[75]:
```
M = sympy.Matrix(( [1, 2, 3], [3, 6, 2], [2, 0, 1] ))
M
```

[75]:
$$\begin{bmatrix} 1 & 2 & 3 \\ 3 & 6 & 2 \\ 2 & 0 & 1 \end{bmatrix}$$

[76]:
```
M.det()
```

[76]:
$$-28$$

[77]:
```
M.inv(method="LU")
```

[77]:
$$\begin{bmatrix} -\frac{3}{14} & \frac{1}{14} & \frac{1}{2} \\ -\frac{1}{28} & \frac{5}{28} & -\frac{1}{4} \\ \frac{3}{7} & -\frac{1}{7} & 0 \end{bmatrix}$$

[78]:
```
Q, R = M.QRdecomposition()
```

[79]:
```
Q
```

[79]:
$$\begin{bmatrix} \frac{\sqrt{14}}{14} & \frac{\sqrt{35}}{35} & \frac{3\sqrt{10}}{10} \\ \frac{3\sqrt{14}}{14} & \frac{3\sqrt{35}}{35} & -\frac{\sqrt{10}}{10} \\ \frac{\sqrt{14}}{7} & -\frac{\sqrt{35}}{7} & 0 \end{bmatrix}$$

[80]:
```
R
```

[80]:
$$\begin{bmatrix} \sqrt{14} & \frac{10\sqrt{14}}{7} & \frac{11\sqrt{14}}{14} \\ 0 & \frac{4\sqrt{35}}{7} & \frac{4\sqrt{35}}{35} \\ 0 & 0 & \frac{7\sqrt{10}}{10} \end{bmatrix}$$

[81]:
```
Q * R
```

[81]:
$$\begin{bmatrix} 1 & 2 & 3 \\ 3 & 6 & 2 \\ 2 & 0 & 1 \end{bmatrix}$$

[82]:
```
A = sympy.Matrix([ [2, 3, 5], [3, 6, 2], [8, 3, 6] ])
A
```

[82]:
$$\begin{bmatrix} 2 & 3 & 5 \\ 3 & 6 & 2 \\ 8 & 3 & 6 \end{bmatrix}$$

[83]:
```
x = sympy.Matrix(3,1,[3,7,5])
x
```

[83]:

$$\begin{bmatrix} 3 \\ 7 \\ 5 \end{bmatrix}$$

[84]:
```
b = A*x
b
```

[84]:
$$\begin{bmatrix} 52 \\ 61 \\ 75 \end{bmatrix}$$

[85]:
```
soln = A.LUsolve(b)
```

[86]:
```
soln
```

[86]:
$$\begin{bmatrix} 3 \\ 7 \\ 5 \end{bmatrix}$$

2 Ecuaciones diferenciales ordinarias

[87]:
```
# incognitas
x = sympy.Symbol('x')
y = sympy.Function('y')
```

[88]:
```
# expreso la ecuacion
f = 6*x**2 - 3*x**2*(y(x))
sympy.Eq(y(x).diff(x), f)
```

[88]:
$$\frac{d}{dx}y(x) = -3x^2y(x) + 6x^2$$

[89]:
```
# Resolver la ecuación
sympy.dsolve(y(x).diff(x) - f)
```

[89]:
$$y(x) = C_1e^{-x^3} + 2$$

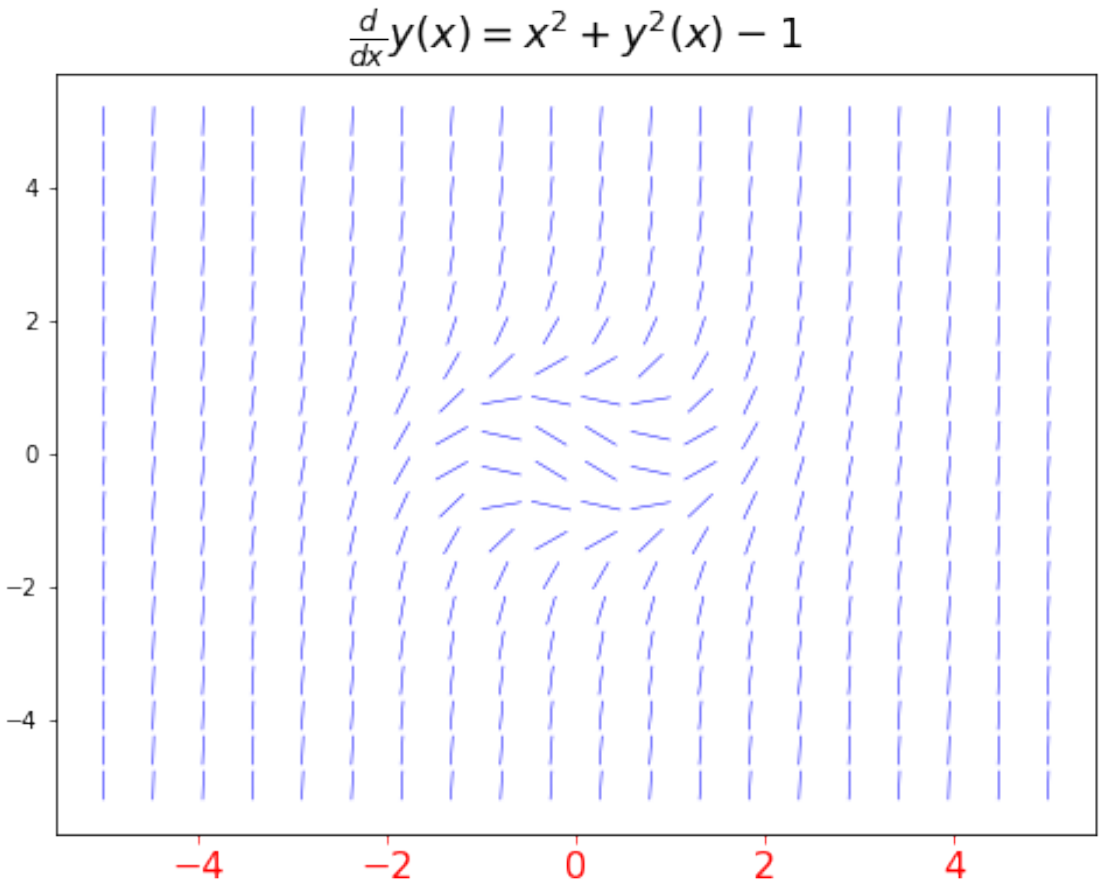
2.0.1 Solución aproximada con serie de potencias

[90]:
```
# Solución con serie de potencias
f = y(x)**2 + x**2 -1
sympy.dsolve(y(x).diff(x) - f)
```

[90]:
$$y(x) = x\left(C_1^2 - 1\right) + \frac{x^3\left(\left(C_1^2 - 1\right)\left(3C_1^2 - 1\right) + 1\right)}{3} + \frac{x^5\left(5C_1^2 + \left(C_1^2 - 1\right)\left(4C_1^2\left(3C_1^2 - 2\right) + 2\left(C_1^2 - 1\right)\left(9C_1^2 - 2\right) + 1\right)\right.}{30}$$
$$C_1 + C_1x^2\left(C_1^2 - 1\right) + \frac{C_1x^4\left(2\left(C_1^2 - 1\right)\left(3C_1^2 - 2\right) + 1\right)}{6} + O\left(x^6\right)$$

[91]:
```
def plot_direction_field(x, y_x, f_xy, x_lim=(-5, 5), y_lim=(-5, 5), ax=None):
    f_np = sympy.lambdify((x, y_x), f_xy, 'numpy')
    x_vec = np.linspace(x_lim[0], x_lim[1], 20)
    y_vec = np.linspace(y_lim[0], y_lim[1], 20)

    if ax is None:
        _, ax = plt.subplots(figsize=(4, 4))

    dx = x_vec[1] - x_vec[0]
    dy = y_vec[1] - y_vec[0]

    for m, xx in enumerate(x_vec):
        for n, yy in enumerate(y_vec):
            Dy = f_np(xx, yy) * dx
            Dx = 0.8 * dx**2 / np.sqrt(dx**2 + Dy**2)
            Dy = 0.8 * Dy*dy / np.sqrt(dx**2 + Dy**2)
            ax.plot([xx - Dx/2, xx + Dx/2],
                    [yy - Dy/2, yy + Dy/2], 'b', lw=0.5)

    ax.axis('tight')
    ax.set_title(r"$%s$" %
                 (sympy.latex(sympy.Eq(y(x).diff(x), f_xy))), fontsize=18)
    return ax
```

[92]:
```
import matplotlib.pyplot as plt
# Defino incognitas
x = sympy.symbols('x')
y = sympy.Function('y')

# Defino la función
f = y(x)**2 + x**2 -1

# grafico de campo de dirección
fig, axes = plt.subplots(1, 1, figsize=(8, 6))
campo_dir = plot_direction_field(x, y(x), f, ax=axes)
```



[93]:
```
# Condición inicial
ics = {y(0): 0}

# Resolviendo la ecuación diferencial
edo_sol = sympy.dsolve(y(x).diff(x) - f, ics=ics)
edo_sol
```

[93]:
$$y(x) = -x + \frac{2x^3}{3} - \frac{4x^5}{15} + O\left(x^6\right)$$

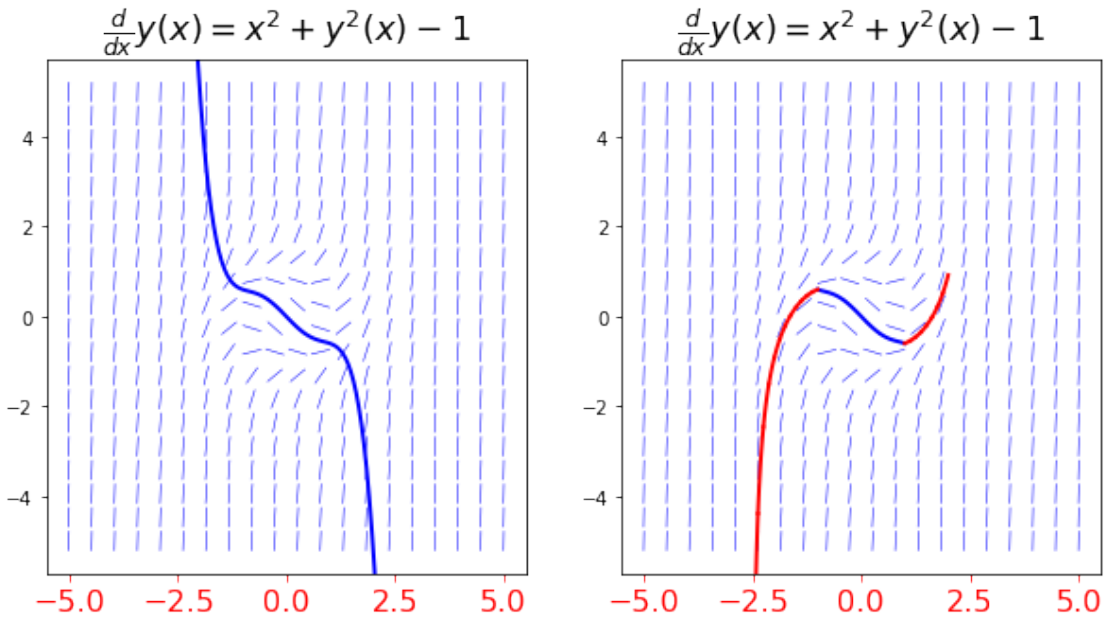
[94]:
```
fig, axes = plt.subplots(1, 2, figsize=(10, 5))

# panel izquierdo - solución aproximada por Serie de potencias
plot_direction_field(x, y(x), f, ax=axes[0])
x_vec = np.linspace(-3, 3, 100)
axes[0].plot(x_vec, sympy.lambdify(x, edo_sol.rhs.removeO())(x_vec),
             'b', lw=2)

# panel derecho - Solución por método iterativo
```

```
plot_direction_field(x, y(x), f, ax=axes[1])
x_vec = np.linspace(-1, 1, 100)
axes[1].plot(x_vec, sympy.lambdify(x, edo_sol.rhs.removeO())(x_vec),
             'b', lw=2)

# Resolviendo la EDO en forma iterativa
edo_sol_m = edo_sol_p = edo_sol
dx = 0.125

# x positivos
for x0 in np.arange(1, 2., dx):
    x_vec = np.linspace(x0, x0 + dx, 100)
    ics = {y(x0): edo_sol_p.rhs.removeO().subs(x, x0)}
    edo_sol_p = sympy.dsolve(y(x).diff(x) - f, ics=ics, n=6)
    axes[1].plot(x_vec, sympy.lambdify(x, edo_sol_p.rhs.removeO())(x_vec),
                 'r', lw=2)

# x negativos
for x0 in np.arange(1, 5, dx):
    x_vec = np.linspace(-x0-dx, -x0, 100)
    ics = {y(-x0): edo_sol_m.rhs.removeO().subs(x, -x0)}
    edo_sol_m = sympy.dsolve(y(x).diff(x) - f, ics=ics, n=6)
    axes[1].plot(x_vec, sympy.lambdify(x, edo_sol_m.rhs.removeO())(x_vec),
                 'r', lw=2)
```

```
<lambdifygenerated-30>:2: RuntimeWarning: invalid value encountered in add
  return (1.33875423934794e+585*x + 5.83054096312614e+1757*(0.333333333333333*x
+ 1)**5 - 5.31174437443959e+1464*(0.333333333333333*x + 1)**4 +
4.83910986610459e+1171*(0.333333333333333*x + 1)**3 -
4.40852997537204e+878*(0.333333333333333*x + 1)**2 + 4.01626271804381e+585)
<lambdifygenerated-31>:2: RuntimeWarning: invalid value encountered in multiply
  return (5.36173424903873e+3501*x + 4.59373525414738e+10507*(0.32*x + 1)**5 -
2.00753579830897e+8756*(0.32*x + 1)**4 + 8.77325261148528e+7004*(0.32*x + 1)**3
- 3.83405174890371e+5253*(0.32*x + 1)**2 + 1.6755419528246e+3502)
<lambdifygenerated-31>:2: RuntimeWarning: invalid value encountered in add
  return (5.36173424903873e+3501*x + 4.59373525414738e+10507*(0.32*x + 1)**5 -
2.00753579830897e+8756*(0.32*x + 1)**4 + 8.77325261148528e+7004*(0.32*x + 1)**3
- 3.83405174890371e+5253*(0.32*x + 1)**2 + 1.6755419528246e+3502)
<lambdifygenerated-32>:2: RuntimeWarning: invalid value encountered in add
  return (2.21274739417512e+21001*x +
3.92836939434924e+63006*(0.307692307692308*x + 1)**5 -
2.5695852985236e+52505*(0.307692307692308*x + 1)**4 +
1.68079117403937e+42004*(0.307692307692308*x + 1)**3 -
1.09942214113376e+31503*(0.307692307692308*x + 1)**2 + 7.19142903106913e+21001)
<lambdifygenerated-33>:2: RuntimeWarning: invalid value encountered in add
  return (1.09317869203404e+125999*x +
5.72061024659671e+377999*(0.296296296296296*x + 1)**5 -
```

```
5.12652410208009e+314999*(0.296296296296296*x + 1)**4 +
4.59413388367846e+251999*(0.296296296296296*x + 1)**3 -
4.11703246115603e+188999*(0.296296296296296*x + 1)**2 +
3.68947808561488e+125999)
<lambdifygenerated-34>:2: RuntimeWarning: invalid value encountered in add
  return (1.58945075781228e+755985*x +
2.10902367502941e+2267958*(0.285714285714286*x + 1)**5 -
1.51143638124966e+1889965*(0.285714285714286*x + 1)**4 +
1.08317415380992e+1511972*(0.285714285714286*x + 1)**3 -
7.76259101631373e+1133978*(0.285714285714286*x + 1)**2 +
5.56307765234296e+755985)
<lambdifygenerated-35>:2: RuntimeWarning: invalid value encountered in add
  return (1.50169797984602e+4535902*x +
2.11976561991136e+13607709*(0.275862068965517*x + 1)**5 -
4.77186924493042e+11339757*(0.275862068965517*x + 1)**4 +
1.0742100860974e+9071806*(0.275862068965517*x + 1)**3 -
2.41818719215598e+6803854*(0.275862068965517*x + 1)**2 +
5.44365517694181e+4535902)
<lambdifygenerated-36>:2: RuntimeWarning: invalid value encountered in add
  return (1.06806014436932e+27215404*x +
9.03531850376628e+81646214*(0.266666666666667*x + 1)**5 -
2.33138685840126e+68038512*(0.266666666666667*x + 1)**4 +
6.01568686401084e+54430809*(0.266666666666667*x + 1)**3 -
1.55223009495081e+40823107*(0.266666666666667*x + 1)**2 +
4.00522554138494e+27215404)
<lambdifygenerated-37>:2: RuntimeWarning: invalid value encountered in add
  return (1.38252943258139e+163292415*x +
2.308775426971e+489877248*(0.258064516129032*x + 1)**5 -
1.60240684596494e+408231040*(0.258064516129032*x + 1)**4 +
1.11215134655344e+326584832*(0.258064516129032*x + 1)**3 -
7.71889249447012e+244938623*(0.258064516129032*x + 1)**2 +
5.35730155125288e+163292415)
<lambdifygenerated-38>:2: RuntimeWarning: invalid value encountered in multiply
  return (6.50348951696191e+979754481*x + 2.81669154573948e+2939263448*(0.25*x +
1)**5 - 8.73184618146191e+2449386206*(0.25*x + 1)**4 +
2.70690405742294e+1959508965*(0.25*x + 1)**3 -
8.39150097679116e+1469631723*(0.25*x + 1)**2 + 2.60139580678476e+979754482)
<lambdifygenerated-38>:2: RuntimeWarning: invalid value encountered in add
  return (6.50348951696191e+979754481*x + 2.81669154573948e+2939263448*(0.25*x +
1)**5 - 8.73184618146191e+2449386206*(0.25*x + 1)**4 +
2.70690405742294e+1959508965*(0.25*x + 1)**3 -
8.39150097679116e+1469631723*(0.25*x + 1)**2 + 2.60139580678476e+979754482)
<lambdifygenerated-39>:2: RuntimeWarning: invalid value encountered in add
  return (7.04658665981152e+5878526881*x +
4.17884397598077e+17635580648*(0.242424242424242*x + 1)**5 -
1.20682085006197e+14696317207*(0.242424242424242*x + 1)**4 +
3.48521402693065e+11757053765*(0.242424242424242*x + 1)**3 -
1.00650538254211e+8817790324*(0.242424242424242*x + 1)**2 +
```

```
2.90671699717225e+5878526882)
<lambdifygenerated-40>:2: RuntimeWarning: invalid value encountered in add
  return (1.14017867466205e+35271161282*x +
2.05524401410144e+105813483849*(0.235294117647059*x + 1)**5 -
4.52885071255961e+88177903207*(0.235294117647059*x + 1)**4 +
9.9795881345111e+70542322565*(0.235294117647059*x + 1)**3 -
2.19906076961824e+52906741924*(0.235294117647059*x + 1)**2 +
4.8457593673137e+35271161282)
<lambdifygenerated-41>:2: RuntimeWarning: invalid value encountered in add
  return (2.04615068593563e+211626967683*x +
1.37310303633699e+634880903053*(0.228571428571429*x + 1)**5 -
6.93835095268286e+529067419210*(0.228571428571429*x + 1)**4 +
3.50597971664382e+423253935368*(0.228571428571429*x + 1)**3 -
1.77158720528037e+317440451526*(0.228571428571429*x + 1)**2 +
8.95190925096836e+211626967683)
<lambdifygenerated-42>:2: RuntimeWarning: invalid value encountered in add
  return (6.8348000116617e+1269761806090*x +
5.89169156897512e+3809285418275*(0.222222222222222*x + 1)**5 -
5.00800304024068e+3174404515229*(0.222222222222222*x + 1)**4 +
4.2568580105463e+2539523612183*(0.222222222222222*x + 1)**3 -
3.61837642196826e+1904642709137*(0.222222222222222*x + 1)**2 +
3.07566000524776e+1269761806091)
<lambdifygenerated-43>:2: RuntimeWarning: invalid value encountered in add
  return (9.49412617450283e+7618570836535*x +
1.81102181876892e+22855712509611*(0.216216216216216*x + 1)**5 -
4.01868944305068e+19046427091342*(0.216216216216216*x + 1)**4 +
8.91754294305808e+15237141673073*(0.216216216216216*x + 1)**3 -
1.97881854938553e+11427856254805*(0.216216216216216*x + 1)**2 +
4.39103335570756e+7618570836536)
<lambdifygenerated-44>:2: RuntimeWarning: invalid value encountered in add
  return (6.82071847770894e+45711425019206*x +
7.67288027834615e+137134275057623*(0.210526315789474*x + 1)**5 -
6.1851430713815e+114278562548019*(0.210526315789474*x + 1)**4 +
4.98587146230104e+91422850038415*(0.210526315789474*x + 1)**3 -
4.01913326041067e+68567137528811*(0.210526315789474*x + 1)**2 +
3.23984127691175e+45711425019207)
<lambdifygenerated-45>:2: RuntimeWarning: invalid value encountered in add
  return (9.37736622963676e+274268550115231*x +
2.27046981672122e+822805650345699*(0.205128205128205*x + 1)**5 -
4.80950824978599e+685671375288082*(0.205128205128205*x + 1)**4 +
1.01879220918971e+548537100230466*(0.205128205128205*x + 1)**3 -
2.1580949893408e+411402825172849*(0.205128205128205*x + 1)**2 +
4.57146603694792e+274268550115232)
```


$\frac{d}{dx}y(x) = x^2 + y^2(x) - 1$
4
2
0
−2
−4
−5.0
−2.5
0.0
2.5
5.0
$\frac{d}{dx}y(x) = x^2 + y^2(x) - 1$
4
2
0
−2
−4
−5.0
−2.5
0.0
2.5
5.0

[ ]: