Índex

1 Info

```
# FORMULACIÓ:

#   -> AMBIT D'ESTUDI
#       -> Acotar l'objecte d'estudi
#       -> Revisar el context (present, passat, futur)
#       -> permet fer primeres assumcions

#   -> OBJECTIU D'ESTUDI
#       -> Mesurable --> ha de poder-se mesurar
#       -> Raonable --> hem de poder tenir recursos per l'estudi (suficients dades)
<-- pot ser util com a analisi critic/resultat
#       -> Útil --> ha de tenir utilitat

#   -> PRECISIÓ DE L'ESTUDI
#       -> Criteri de qualitat de l'estudi <-- pot ser util per l'analisi critic/resultat




# ANALISI DEL SISTEMA:

#   -> CONTEXT
#       -> Definir ambit/situació --> definir objectius --> identificar sistema i context

#   -> FACTORS I ESDEVENIMENTS MESURABLES
#       -> Identificar els factors més rellevants --> Analitzar unitats i rellevancia -->
 considererar elements negligibles --> SEMPRE MANTENIR MODEL SENCILL
#       -> Triarem factors que a priori afectaran més al resultat -->
despres analitzar la seva sensibilidad
#       -> En el cicle de modelització podem afegir o eliminar factors -->
 podem tornar en rere en els passos per simplificar o arreglar model
```

```
#   -> OBSERVACIÓ "REALITAT"
#       -> Analitzar dades obtingudes --> graficar y explicar comportaments aparents -->
permet plantejar un model sencill inicial

#   -> RELACIONS I COMPORTAMENT D'ESTATS
#       -> Agrupar i relacionar factors --> subsistemes, categories, efectes...
#       -> Identificar relacions i comportament dels factors --> proporcionalitat i
comportament --> permet plantejar el model
#       -> Identificar estats i esdeveniments --> IMPORTANT SI SISTEMA CAMBIA AMB TEMPS -->
permet plantejar noves relacions

# ANALISI DIMENSIONAL: <-- util i és recomana fer
#   -> Treballar amb conceptes simplificats: L,T,M...
#   -> Recomana passar sempre unitats al sisteme internacional
#   -> Model ha de ser consistent --> ha de donar les unitats esperades
#   -> És pot construïr un model amb l'analisi dimensional --> recamanat si tenim poques
variables i amb unitats amb diferents màgnituds

# CONSTRUCCIÓ MODELS MATEMÀTICS: <-- sempre fer model sencill i simple --> (en cas de panik
utilitzar una recta :D)
#   -> Analisi Dimensional --> Analitzar les relacions i comportaments del paràmetres
#   -> Representació diferencial --> plantejar EDOs
#   -> Plantejament equacions per parts/diferencies --> dP = a dt ==> P_i - P_{i-1} = a (t_i

- t_{i-1}) <-- forma discreta i iterativa

# ANÀLISI DEL MODEL:
#   -> Pes relatiu dels coeficients
#   -> Sensibilidad del model --> coeficients de sensitivitat : Ci = xi/M * dM/dxi; M =
Model, xi = paràmetre i

# INTERPRETACIÓ DEL MODEL EN EL CONTEXT: <-- imaginar el perque el model és així
#   -> Tracta de separar els components del model --> perque creix, on decreix més, punts
```

```
d'inflexió, extrems relatius...
#   -> Analitza representacions gràfiques dels resultats --> té sentit, fa el que esperem...
#   -> Compara els resultats amb els valors esperats
#   -> Petit anàlisi crític del resultat


# EXPERIMENTACIÓ:
#   -> Necessàries per refutar el nostre model, calibrar-lo i verificar-lo -->
 analisi critica i validació del model
#         -> Resolució --> Diferencia entre dos punts distingibles d'un sistema de mesura.
#         -> Precisió --> Diferencia entre el resultat de la mesura amb el seu valor real.
#         -> Error --> Distribució aleatòria dels valors obtinguts al voltant dels valors
 mesurats.

# ERROR I INCERTESES:
#   -> Els valors de les mesures tindran errors
#   -> Els paràmetres del model també tindran errors
#   -> Els càlculs tindran certa resolució numèrica
#   ERRORS:
#         -> Error estadístic: Aquell que varia durant el procés de mesura -->
El valor esperat és ma mitjana de les observacions +- la variança de les observacions
#         -> Error sistemàtic: Aquell que no varia durant el procés de mesura
 <-- CONSIDERAR NEGLIGIBLES ==> MÉS SIMPLE :D
#   INCERTESES (PROPAGACIÓ D'ERRORS):
#         -> Monte Carlo --> NO USAR
#         -> Serie Taylor --> Uc = +-sqrt{Br^2 + Sr^2}; Uc = estimació error del resultat
 (interval d'incertesa) <-- EXPLIQUEM EN DETALL EN VALIDACIÓ DEL MODEL
#                 -> Br = estimació de l'error sistemàtic; Sr = estimció error estadístic
#   INCERTESA RELATIVA:
#         -> Ur = \sum Ui^2; Ui^2 = (Ci*Ii)^2 incertesa del relativa de la predicció
#                 -> Ci = xi/M * dM/dxi coeficient de sensitivitat del paràmetre i-essim
#                 -> Ii = incertesa de les mesures del paràmetre i-essim

# NIVELL DE CONFIANÇA:
```

```
#   -> Fent servir t-test: Ur = t*Uc = t*(Br^2 + Sr^2) ----> t = 2 si mostres grans (N > 9
 per exemple)
#   -> Fent servir covariances entre variables i mesures independents: Ur = \sum (Ui*I)^2
```

2 Comun

```
import warnings


import numpy as np
import pandas as pd
import scipy
from scipy import stats
import matplotlib.pyplot as plt
from IPython.display import Math, display
import sympy as sp
from sympy.physics.units.systems import SI
from sympy.physics.units import meter, second, liter, hour, degree, convert_to
from sympy import sin, cos, pi, Abs, sqrt
import math

sp.init_printing()
n = 50

def show(*args):
    out = ""
    for arg in args:
        if isinstance(arg, (sp.Expr, sp.Eq)):
            arg = sp.latex(arg)
        else:
            arg = str(arg)
        out += arg
    display(Math(out))

qq = "\quad "

# Dades
data = pd.read_csv("C:/Users/onasa/Documents/3r2n_semestre/Mod_Sim/Entrega5/dp_data.csv")
print(data.columns)
```

```
l = pd.plotting.scatter_matrix(data[data.columns])
```

3 Modelos

```
# LINEAL --> f(x) = ax + b; f'(x) = a
X = np.linspace(-2, 2, n)
Y1 = 1 + X
Y2 = np.ones(n)

fig, ax = plt.subplots()

ax.plot(X, Y1, X, Y2)
ax.set_ylabel('$f(x) \quad f\'(x)$')
ax.set_xlabel('x')
plt.show()

# CREIXEMENT EXPONENCIAL --> f(x) = a*e^{bx}+c; f'(x) = f(x)
X = np.linspace(-1, 1, n)
Y1 = np.exp(2 * X)
Y2 = 2 * np.exp(2 * X)

fig, ax = plt.subplots()

ax.plot(X, Y1, X, Y2)
ax.set_ylabel('$f(x) \quad f\'(x)$')
ax.set_xlabel('x')
ax.set_ylim([-1, 10])
plt.show()

# DECREIXEMENT EXPONENCIAL --> f(x) = a*e^{-bx}+c; f'(x) = -f(x)
X = np.linspace(-1, 1, n)
Y1 = np.exp(-2 * X)
Y2 = -2 * Y1
```

```
fig, ax = plt.subplots()


ax.plot(X, Y1, X, Y2)
ax.set_ylabel('$f(x) \quad f\'(x)$')
ax.set_xlabel('x')
ax.set_ylim([-10, 10])
plt.show()

# MÀXIM SIMPLE / PARÀBOLA CÒNCAVA --> f(x) = a - (x-b)^2; f'(x) = -x
X = np.linspace(-1, 3, n)
Y1 = 1 - (X - 1)**2
Y2 = -2 * (X - 1)


fig, ax = plt.subplots()


ax.plot(X, Y1, X, Y2)
ax.set_ylabel('$f(x) \quad f\'(x)$')
ax.set_xlabel('x')
plt.show()

# OSCIL·LACIÓ --> f(x) = a*sin(bx+c); f'(x) = ab*cos(bx+c)
X = np.linspace(-2, 2, n)
Y1 = np.sin(np.pi * X)
Y2 = np.cos(np.pi * X)


fig, ax = plt.subplots()


ax.plot(X, Y1, X, Y2)
ax.set_ylabel('$f(x) \quad f\'(x)$')
ax.set_xlabel('x')
plt.show()

# DISTRIBUCIÓ NORMAL
X = np.linspace(-2, 2, n)
Y1 = np.sqrt(2 / np.pi) * np.exp(-2 * X * X )
```

```
Y2 = - 4 * X * Y1

fig, ax = plt.subplots()

ax.plot(X, Y1, X, Y2)
ax.set_ylabel('$f(x) \quad f\'(x)$')
ax.set_xlabel('x')
plt.show()

# CREIXEMENT LIMITAT --> f(x) = a-b*e^{-cx}; f'(x) = f(x)
X = np.linspace(0, 2, n)
Y1 = (1 - np.exp(-4 * X))
Y2 = 4 * (1 - Y1)

fig, ax = plt.subplots()

ax.plot(X, Y1, X, Y2)
ax.set_ylabel('$f(x) \quad f\'(x)$')
ax.set_xlabel('x')
ax.set_ylim([-1, 4])
plt.show()

# CREIXEMENT ASSIMPTÒTIC --> f(x) = a + bx + c*e^{-dx}
X = np.linspace(0, 2, n)
Y1 = 1 + X + 2 * np.exp(-10 * X)
Y2 = 1 - 20 * np.exp(-10 * X)

fig, ax = plt.subplots()

ax.plot(X, Y1, X, Y2)
ax.set_ylabel('$f(x) \quad f\'(x)$')
ax.set_xlabel('x')
ax.set_ylim([-1, 4])
plt.show()
```

```
# MÀXIM AMB CUA --> f(x) = ax*e^{-x/b}
X = np.linspace(0.001, 2, n)
Y1 = 10 * X * np.exp(-4 * X)
Y2 = Y1 * ( 1 / X - 4)

fig, ax = plt.subplots()

ax.plot(X, Y1, X, Y2)
ax.set_ylabel('$f(x) \quad f\'(x)$')
ax.set_xlabel('x')
ax.set_ylim([-1, 1])
plt.show()

# OSCIL·LACIÓ ESMORTEIDA --> a*e^{-x/b}*sin(cx+d)
X = np.linspace(-2, 2, n)
Y1 = np.exp(-X) * np.sin(np.pi * X)
Y2 = np.exp(-X) * (np.cos(np.pi * X) - np.sin(np.pi * X))

fig, ax = plt.subplots()

ax.plot(X, Y1, X, Y2)
ax.set_ylabel('$f(x) \quad f\'(x)$')
ax.set_xlabel('x')
plt.show()
```

4 Definir/representar model

```
data.plot(y="Concentració (mg/l)", x="t (min)") #nom de les columnes
```

4.1 Model 1 eq

```
# RESOLDRE EDOs
from sympy.abc import t, C, V, T
Q = sp.Symbol('Q', nonzero=True, positive=True)
```

```
x = sp.Function('x')(t)

xp = sp.diff(x, t)

fx = - x * Q/V

eq = sp.Eq(xp, fx)

ics = {x.subs(t, 0): C} #condició inicial

sol = sp.dsolve(eq, x, ics=ics)
xt = sol.rhs
show(xt)
```

4.2 Model 2 eq

```
from sympy.abc import l

Q = sp.symbols("Q")

m1 = sp.Function("m_1")(t)
m2 = sp.Function("m_2")(t)

dm1 = -(Q+l)*m1
dm2 = Q*(m1 - m2) -m2*l

# Resolució sistema EDOS
m1p = sp.diff(m1, t)
eq1 = sp.Eq(dm1,m1p)

m2p = sp.diff(m2, t)
eq2 = sp.Eq(dm2,m2p)

#si tinguéssim condició inicial ics = {m1.subs(t,0): C, m2.subs(t,0): 0}
```

```
M1, M2 = sp.dsolve([eq1, eq2], [m1,m2])

show(M1,qq,M2)
M1 = M1.rhs
M2 = M2.rhs
```

4.3 Numèricament

```
from sympy.abc import t, a, b, m, T, C, D
T0, Tn = sp.symbols('T_0 T_n')
P = sp.Function('P', real=True)(t)

dP = sp.diff(P, t)
R = a + b * sp.sin(2 * sp.pi * (t - T0) / Tn)
M = m

eq = sp.Eq(dP, R * P - m)

try:
  Pt = sp.dsolve(eq, P) #<-- non preocupare, tarda en ejecutar-se esto :D
except:
  print("Este mensaje sale porque el dsolve no puede resolver esto asi que enchufamos el mètoc

  # Aproximacions
  cons = {'C1': C}
  valors = {a: 0.00001, b: -0.0001, T0: 0, Tn: 4, m: 100, C: 5000}

  Pt5 = sp.dsolve(eq, P, hint='1st_power_series', n=3) # <-- tranqui tarda aqui bastante
  fPt5 = sp.lambdify('t', Pt5.rhs.subs(cons).subs(valors).removeO())

  Pt6 = sp.dsolve(eq, P, hint='1st_power_series', n=4) # <-- tranqui tarda aqui bastante más.
  fPt6 = sp.lambdify('t', Pt6.rhs.subs(cons).subs(valors).removeO())

  x = np.linspace(0, 10)
```

```
  p6 = fPt6(x)
  p5 = fPt5(x)
  l = plt.plot(x, p6, x, p5)
```

4.4 Representació model

```
# PLantejament del model
from sympy.abc import t, C, V, T
Q = sp.Symbol('Q', nonzero=True, positive=True)
x = sp.Function('x')(t)
xp = sp.diff(x, t)
fx = - x * Q/V
eq = sp.Eq(xp, fx)
ics = {x.subs(t, 0): C}
sol = sp.dsolve(eq, x, ics=ics)
xt = sol.rhs
show(xt)

# Representació
valors = {C: 1, V: 1000000, Q: 100000} # mg/l, l, l/min

fxt = sp.lambdify(t, xt.subs(valors))
temps = np.linspace(0, 60, 50)

con = fxt(temps)
l = plt.plot(temps, con)
plt.legend(["$x$"])
```

5 Coefs sensitivitat

```
# Concentració de partícules
from sympy.abc import T, E, L

C = - sp.log(T) / (E * L)
```

```
show("C =", C)
print("-"*40)

# Coeficients de sensitivitat
CT = T / C * sp.diff(C, T)
CE = E / C * sp.diff(C, E)
CL = L / C * sp.diff(C, L)
show(CT, qq, CE, qq, CL)
```

6 Simulacio i discrepància

6.1 Simulació 2 variable

```
# Exemple amb caiguda de pressió
from sympy.abc import rho, epsilon, mu, Q, L, d, f
P = sp.symbols('P')
tururu = sp.Eq(P, f * 8 * rho * Q**2 * L / (sp.pi**2 * d**5))
F = sp.solve(tururu, f)
show('f= ', F)

valors = {rho: 1000, epsilon: 0.01, mu: 1.2E-3, L: 1000, d: 0.15}

fF = sp.lambdify([Q, P], F.subs(valors))
q = data['Flow (l/s)'] / 1000
p = data['\Delta P (Pa)']
dp = fF(q,p)

l = plt.hist(dp)
print(np.mean(dp), np.std(dp))
```

6.2 Simulacio 1 var

```
# Exemple sense caiguda de pressió
valors = {rho: 1000, epsilon: 0.01, mu: 1.2E-3, L: 1000, d: 0.15}
```

```
ff = sp.lambdify(Q,f.subs(valors))
q = data['Flow (l/s)'] / 1000
dp2 = ff(q)

l = plt.hist(dp2)
print(np.mean(dp2), np.std(dp2))
```

6.3 Discrepancia

```
# S --> calculada per nosaltres
# D --> la del model (la bona)
S = np.mean(dp2)
D = np.mean(dp) # --> directament data['nom_columna'].mean()
E = S - D
print(E)
l = plt.hist(dp, histtype='step') # --> directament l = plt.hist(data['nom_columna'],
 histtype='step')
l = plt.hist(dp2, histtype='step')
l = plt.legend(['Amb dP', 'Sense dP'])

# Derivades parcials
mesures = {Q: q.mean()}
valors = {rho: 1000, epsilon: 0.01, mu: 1.2E-3, L: 1000, d: 0.15}

DE = sp.diff(f, epsilon).subs(valors).subs(mesures).n() # f és el nostre model
Dd = sp.diff(f, d).subs(valors).subs(mesures).n()
Dm = sp.diff(f, mu).subs(valors).subs(mesures).n()
DQ = sp.diff(f, Q).subs(valors).subs(mesures).n()
Dr = sp.diff(f, rho).subs(valors).subs(mesures).n()

show(DE, qq, Dd, qq, Dm, qq, DQ, qq, Dr, qq)

# temps se pone el temps máximo
```

```
# Incertesa dels paràmetres al quadrat
up2 = np.sum(np.power([
DE * valors[epsilon] * 0.001,
DQ * data['Flow (l/s)'].std(), # de les dades reals
Dm * valors[mu] * 0.001,
Dd * valors[d] * 0.001,
Dr * valors[rho] * 0.001,
], 2))
up2

# Incertesa de l'estimació de la discrepància
#uD = data['\Delta P (Pa)'].std()
uD = np.std(dp) # dp és el model amb caiguda --> amb el k comparem --> dades reals
uV = np.sqrt(float(uD**2 + up2))
uV
# Discrepància
show("E = %0.4f \pm %0.4f" % (E, uV))
```

7 Error i interval confiança

```
ur = E/S
ur

t = 0.05 / ur
l, h = stats.t.cdf([-t, t], 1000)
CL = h - l
CL

# Interval de t per a un nivell de confiança del 99%
from scipy import stats
ti = stats.t.interval(0.99,60)[1] # el 60 és el temps


uV*ti
```