DIGWISE

Data Analysis

Hockchen 2022/08/18

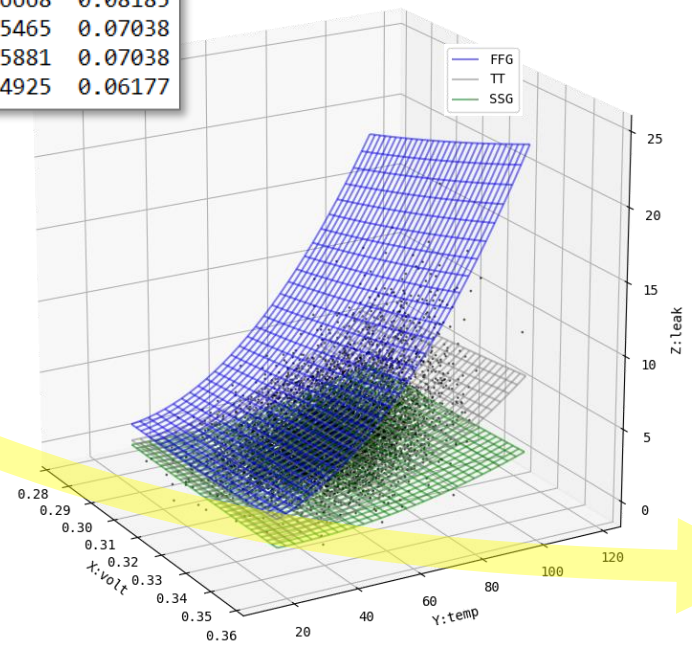
corner	volt	temp	cell	area	leak	tr	tf
FFG	0.29	50	NAND2V1_C10UL	0.052272	4.28330	0.0569379	0.0866719
FFG	0.29	85	NAND2V1_C10UL	0.052272	12.87500	0.0515327	0.0797780
FFG	0.31	25	NAND2V1_C10UL	0.052272	1.86788	0.0543973	0.0784176
FFG	0.31	50	NAND2V1_C10UL	0.052272	4.70400	0.0510046	0.0745741
FFG	0.31	85	NAND2V1_C10UL	0.052272	14.10400	0.0470171	0.0699880
...	...	...	...	...	...	...	...
TTS	0.31	85	CLKINV1_C12UL	0.034848	6.78940	0.0532715	0.0453096
TT	0.31	85	CLKINV1_C10UL_P2	0.034848	4.26030	0.0561441	0.0587706
TT	0.31	85	CLKINV1_C12UL_P2	0.034848	3.50480	0.0643876	0.0537757
TT	0.31	85	INV1_C10UL_P2	0.034848	4.26030	0.0561441	0.0587706
TT	0.31	85	INV1_C12UL_P2	0.034848	3.50480	0.0645061	0.0538953

Information

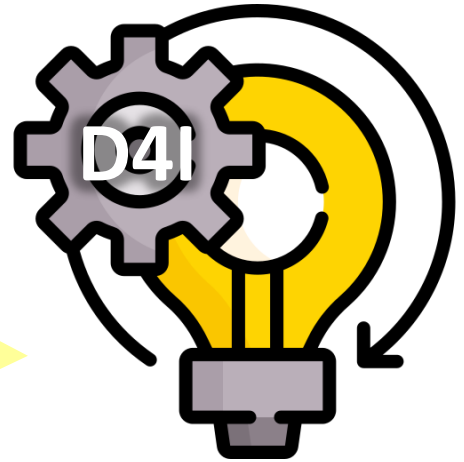
Intelligence

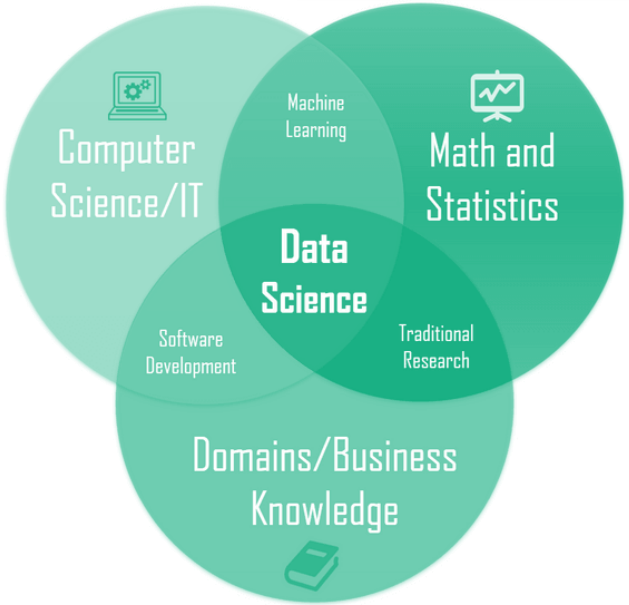
cell	corner	volt	temp	area	leak	tr	tf	dr	df
CLKINV1_C10UL	FFG	0.29	50	0.03485	3.88950	0.05267	0.05245	0.04331	0.04746
			85	0.03485	11.80200	0.04697	0.04870	0.03524	0.04213
		0.31	25	0.03485	1.68682	0.05049	0.04806	0.04442	0.04594
			50	0.03485	4.27640	0.04685	0.04600	0.03852	0.04190
			85	0.03485	12.93900	0.04268	0.04353	0.03202	0.03783
...				...	...	...	...	...	...
NAND2V1_C12UL	TT	0.75	85	0.05227	13.24740	0.02292	0.02800	0.01797	0.02034
	TTS	0.29	50	0.05227	2.20785	0.07321	0.09489	0.06668	0.08185
			85	0.05227	6.96693	0.06484	0.08511	0.05465	0.07038
		0.31	50	0.05227	2.40899	0.06347	0.08043	0.05881	0.07038
			85	0.05227	7.59226	0.05760	0.07388	0.04925	0.06177

Insight



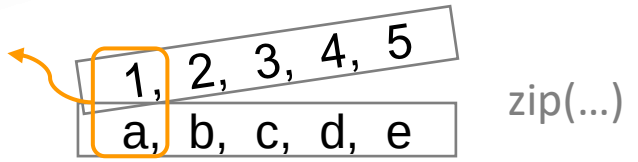
Innovation

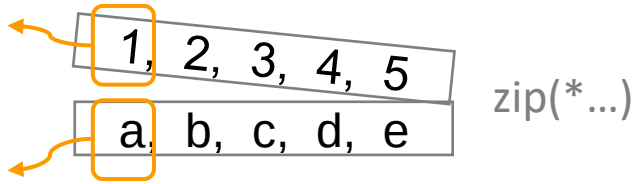




Python Basics (1/2)

- Getting Started
  - `print("Hello, World!")`
  - `# single comment`
  - `'''multiple comments'''`
- Variables
  - `a=1; a='efg'; a=[1,2,3,4,5]; a=(1,2,3,4,5); a={'a':1,'b':2,'c':3}`
  - `x,y,z="Orange","Banana","Cherry" # multiple assignment`
  - `a,b,_ = 1,2,3 # drop the last`
  - `a,b,*c = 1,2,3,4,5 # c=[3,4,5]`
  - `x=(1,2,3); y=('a','b','c'); vL=zip(x,y) # pair list elements`
  - `x,y=zip(*vL) # unpack paired elements`
  - `1+2; 3*4; 2**3; 16**0.5 # numeric`





Python Basics (2/2)

- String
  - `x='abc'; y='def'; x+y # concatenate`
  - `x=1.2345; y=123456; z='abcd'; f'{x:10.3g},{y:10,},{z:10s}' # formatter`
- List
  - `[1,2,3]+[2,3,4]+['a','b','c'] # concatenate`
  - `[v for v in range(0,10)] # iterator`
- Dictionary (JSON)
  - `a={'a':1,'b':2,'c':3}; b={'d':4,'e':5}; {**a,**b} # concatenate`
  - `{f'{v:c}:{ii}' for ii,v in enumerate(range(97,105))} # iterator`
- Logic
  - `x=5; 1<x<7; None or 123;`
- Control, Function, Class, Module ➔ **home work!**

Key Tips

- **Understanding data:**
  - Shape (size and dimension): shape
  - Ranges (distributions, means, extremes, outliers): describe
  - Indexing, Search, Filtering: filter
  - Visualization: matplotlib, plotly
  - Correlation: feature correlation matrix
  - Analysis: trend, comparison
  - Modeling: LS regression
  - Prediction: inner-product

Data Shape

Data: simple.csv
https://digwise-tech.atlassian.net/wiki/spaces/ZHUBEI/pages/21430301/Getting+Started+with+Data+Analysis

```
import pandas as pd
import numpy as np

# load data
pd.set_option('expand_frame_repr', False)
df = pd.read_csv(csv)

# cursory look at the data
df.shape # (328, 12)
df.index
df.columns
df.iloc[:,:9]
df[['corner','volt','temp','cell','area','leak','tr','tf']]
```

corner	volt	temp	cell	area	leak	tr	tf
FFG	0.29	50	NAND2V1_C10UL	0.052272	4.28330	0.0569379	0.0866719
FFG	0.29	85	NAND2V1_C10UL	0.052272	12.87500	0.0515327	0.0797780
FFG	0.31	25	NAND2V1_C10UL	0.052272	1.86788	0.0543973	0.0784176
FFG	0.31	50	NAND2V1_C10UL	0.052272	4.70400	0.0510046	0.0745741
FFG	0.31	85	NAND2V1_C10UL	0.052272	14.10400	0.0470171	0.0699880
...	...	...	...	...	...	...	...
TTS	0.31	85	CLKINV1_C12UL	0.034848	6.78940	0.0532715	0.0453096
TT	0.31	85	CLKINV1_C10UL_P2	0.034848	4.26030	0.0561441	0.0587706
TT	0.31	85	CLKINV1_C12UL_P2	0.034848	3.50480	0.0643876	0.0537757
TT	0.31	85	INV1_C10UL_P2	0.034848	4.26030	0.0561441	0.0587706
TT	0.31	85	INV1_C12UL_P2	0.034848	3.50480	0.0645061	0.0538953

Data Range

```
# know data range (outlier)
df[['area','leak','tr','tf']]
df[['area','leak','tr','tf']].max()
df[['area','leak','tr','tf']].min()
df[['area','leak','tr','tf']].std()
df[['area','leak','tr','tf']].mean()

# filter data
de = df[['area','leak','tr','tf']].describe()
de['leak']['25%']
de['leak']['75%']
u = de['leak']['mean']
s = de['leak']['std']
df[df['leak']>(u+3*s)]

df.describe(percentiles=[0.003,0.997]).round(3)
```

	area	leak	tr	tf
count	328.00000	328.00000	328.00000	328.00000
mean	0.04059	4.23852	0.06509	0.07146
std	0.00820	4.79975	0.02003	0.03067
min	0.03485	0.11667	0.01963	0.02009
25%	0.03485	1.06555	0.05223	0.04926
50%	0.03485	2.62495	0.06239	0.06475
75%	0.05227	5.71201	0.07589	0.08447
max	0.05227	27.04570	0.13969	0.20725

cell	corner	volt	temp	area	leak	tr	tf	dr	df	pr	pf
NAND2V1_C10UL	FFGS	0.29	85	0.05227	24.6993	0.04544	0.06782	0.03175	0.05316	0.00002	8.31000e-06
NAND2V1_C10UL	FFGS	0.31	85	0.05227	27.0457	0.04182	0.06047	0.02948	0.04783	0.00002	1.06000e-05
NAND2V1_C12UL	FFGS	0.31	85	0.05227	20.1112	0.04630	0.05538	0.03629	0.04364	0.00002	1.03000e-05
INV1_C10UL	FFGS	0.29	85	0.03485	22.7431	0.04091	0.04200	0.02814	0.03494	0.00005	5.00000e-06
INV1_C10UL	FFGS	0.31	85	0.03485	24.9195	0.03757	0.03830	0.02638	0.03221	0.00005	7.66000e-06
CLKINV1_C10UL	FFGS	0.29	85	0.03485	22.7431	0.04091	0.04200	0.02814	0.03494	0.00005	5.00000e-06
CLKINV1_C10UL	FFGS	0.31	85	0.03485	24.9195	0.03757	0.03830	0.02638	0.03221	0.00005	7.66000e-06

Re-indexing & Querying

```
#%% re-indexing
dt = df.set_index(['cell','corner','volt','temp'])
dt = dt.sort_values(['cell','corner','volt','temp'])
dt.index

# query by index
[v for v in dt.index.levels[0].unique() if 'NAND2V1_' in v]
d = dt.loc['NAND2V1_C10UL']
d[['area','leak','tr','tf','dr','df','pr','pf']].round(4)

#%% query & filter data by index
dc = dt.filter(regex='.*NAND.*UL.*TT.*0\.31.*85',axis=0)
dc[['area','leak','tr','tf','dr','df','pr','pf']].round(4)
```

index

columns

cell	corner	volt	temp	area	leak	tr	tf	dr	df
CLKINV1_C10UL	FFG	0.29	50	0.03485	3.88950	0.05267	0.05245	0.04331	0.04746
			85	0.03485	11.80200	0.04697	0.04870	0.03524	0.04213
		0.31	25	0.03485	1.68682	0.05049	0.04806	0.04442	0.04594
			50	0.03485	4.27640	0.04685	0.04600	0.03852	0.04190
			85	0.03485	12.93900	0.04268	0.04353	0.03202	0.03783
...				...	...	...	...	...	...
NAND2V1_C12UL	TT	0.75	85	0.05227	13.24740	0.02292	0.02800	0.01797	0.02034
	TTS	0.29	50	0.05227	2.20785	0.07321	0.09489	0.06668	0.08185
			85	0.05227	6.96693	0.06484	0.08511	0.05465	0.07038
		0.31	50	0.05227	2.40899	0.06347	0.08043	0.05881	0.07038
			85	0.05227	7.59226	0.05760	0.07388	0.04925	0.06177

Data Processing

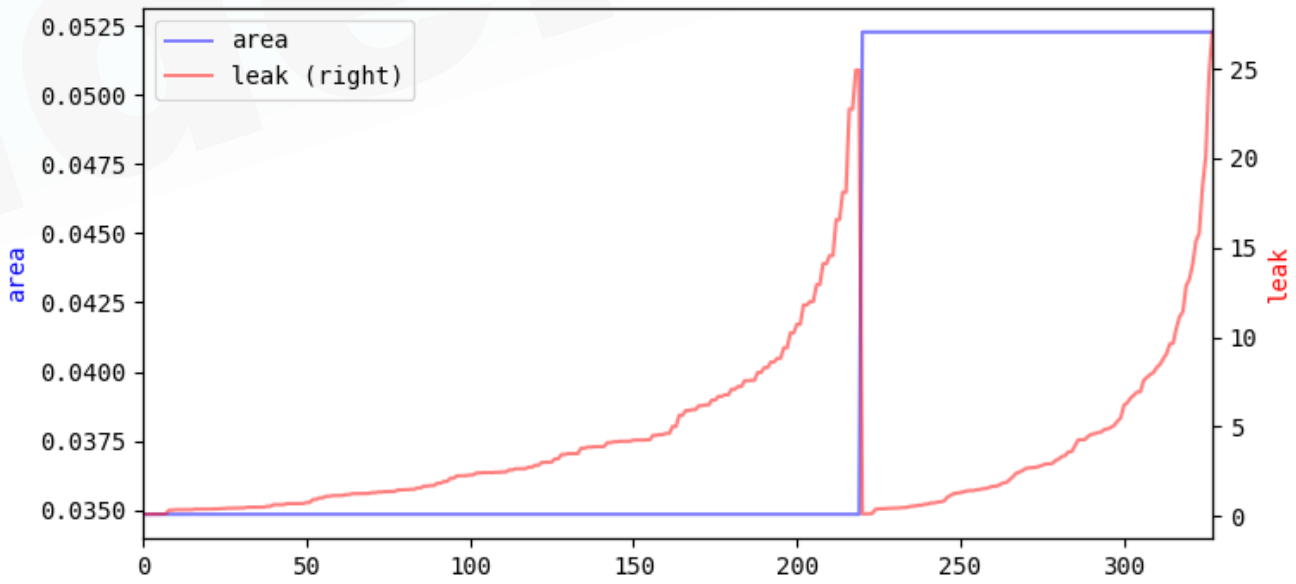
```
#%% (3) data processing
ta = (df['tr']+df['tf'])/2 # average 2 vectors
ta = df[['tr','tf']].mean(axis=1) # native pandas operation
imb = df['tr']/df['tf'] # rise/fall imbalance

# data augmentation: store the action back
df['ta'] = df[['tr','tf']].mean(axis=1)
df['da'] = df[['dr','df']].mean(axis=1)
df['pa'] = df[['pr','pf']].mean(axis=1)
df['imb'] = df['tr']/df['tf']

# sort data by area & leakage
df = df.sort_values(['area','leak'],ascending=False)
df[['corner','volt','temp','cell','area','leak','tr','tf','ta','imb']].round(3)
```

corner	volt	temp	cell	area	leak	tr	tf	ta	imb
FFGS	0.31	85	NAND2V1_C10UL	0.052	27.046	0.042	0.060	0.051	0.692
FFGS	0.29	85	NAND2V1_C10UL	0.052	24.699	0.045	0.068	0.057	0.670
FFGS	0.31	85	NAND2V1_C12UL	0.052	20.111	0.046	0.055	0.051	0.836
FFGS	0.29	85	NAND2V1_C12UL	0.052	18.427	0.051	0.062	0.056	0.823
TT	0.75	85	NAND2V1_C10UL	0.052	15.812	0.023	0.028	0.026	0.808
...	...	...	...	...	...	...	...	...	...
SSG	0.33	25	CLKINV1_C12UL	0.035	0.127	0.105	0.080	0.092	1.307
SSG	0.31	25	INV1_C10UL	0.035	0.125	0.117	0.112	0.114	1.045
SSG	0.31	25	CLKINV1_C10UL	0.035	0.125	0.117	0.112	0.114	1.045
SSG	0.31	25	INV1_C12UL	0.035	0.117	0.133	0.100	0.117	1.324
SSG	0.31	25	CLKINV1_C12UL	0.035	0.117	0.133	0.100	0.117	1.323

augment data

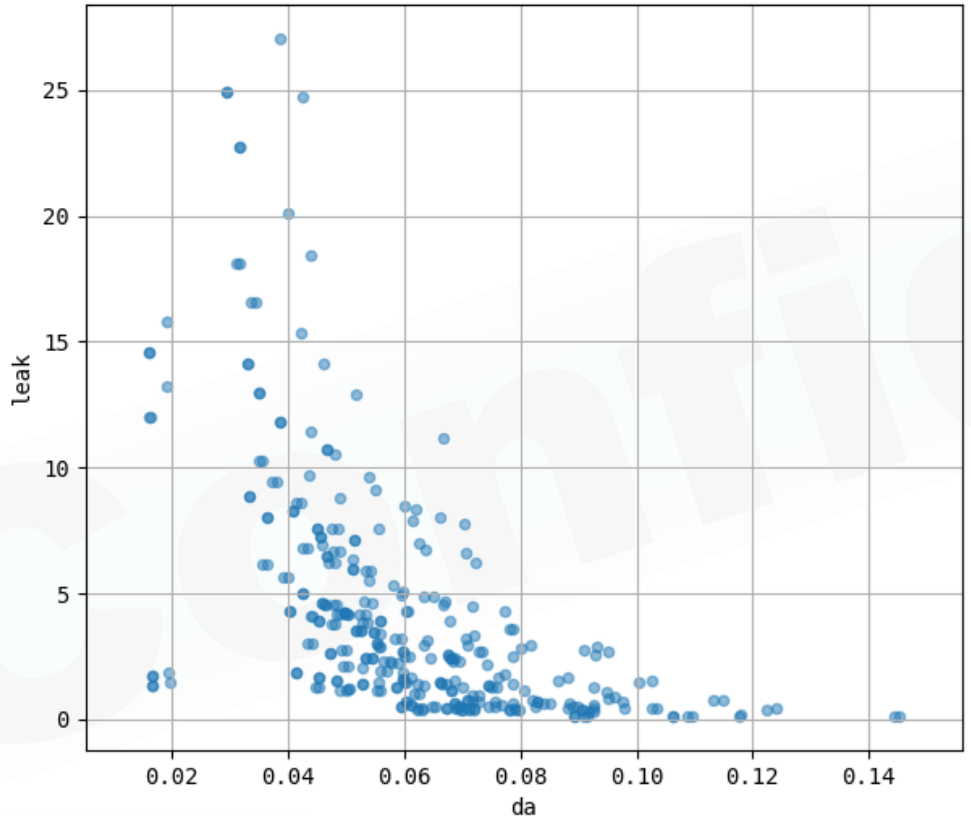
Visualization

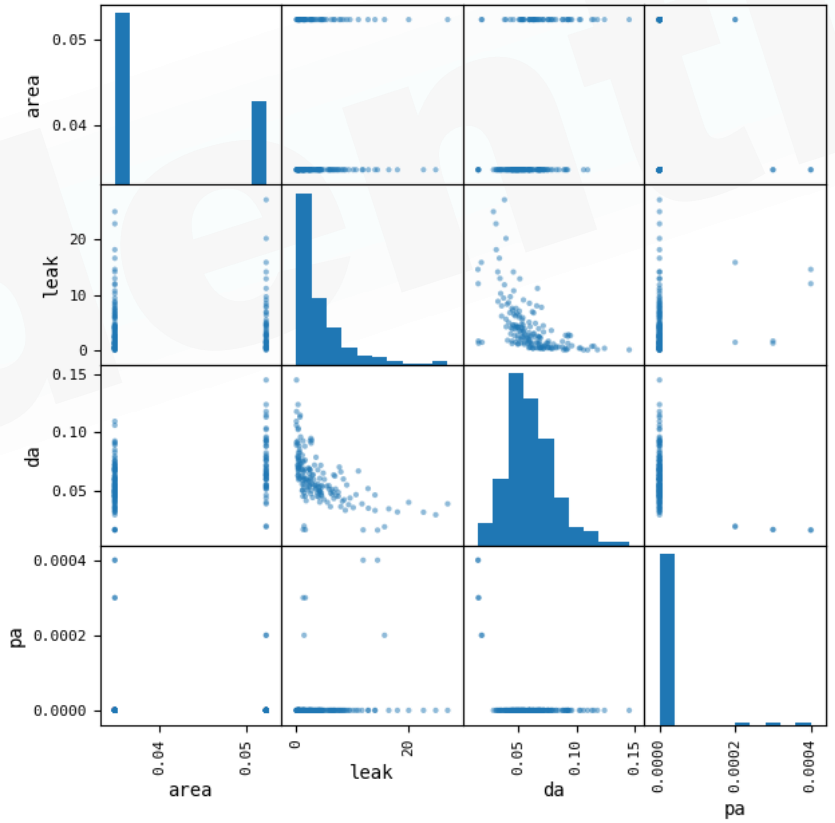
```
#%% (4) data visualization
dt = df.set_index(['cell']).sort_values(['area','leak'],ascending=True)
dt[['corner','volt','temp','area','leak']]
dt[['area','leak']].describe()

# pandas' native view
dt['area'].plot(use_index=False).set_ylabel('area')

# dual axis
g = dt[['area','leak']].plot(use_index=False,secondary_y='leak',alpha=0.5,color=['b','r'])
g.set_ylabel('area',color='b')
g.right_ax.set_ylabel('leak',color='r')
```



Feature Correlation

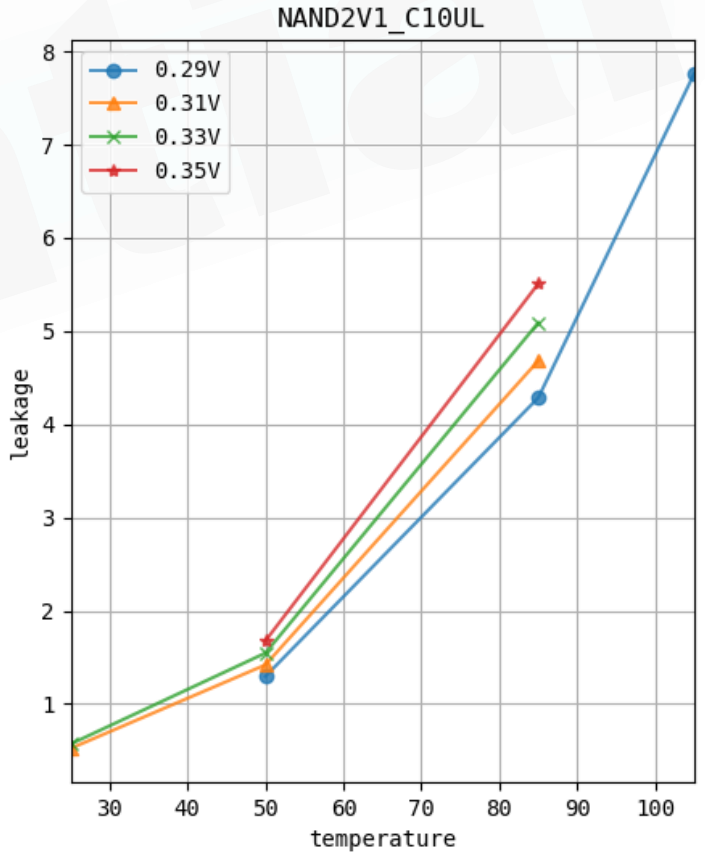
```
#%% feature correlation
df.plot(x='da',y='leak',figsize=(7,6),kind='scatter',alpha=0.5,grid=True)

# correlation matrix
dt = df[['area','leak','da','pa']].iloc[::2].round(4) # 1/2 sub sampling
pd.plotting.scatter_matrix(dt,figsize=(7,7),alpha=0.5,range_padding=0.2)
```





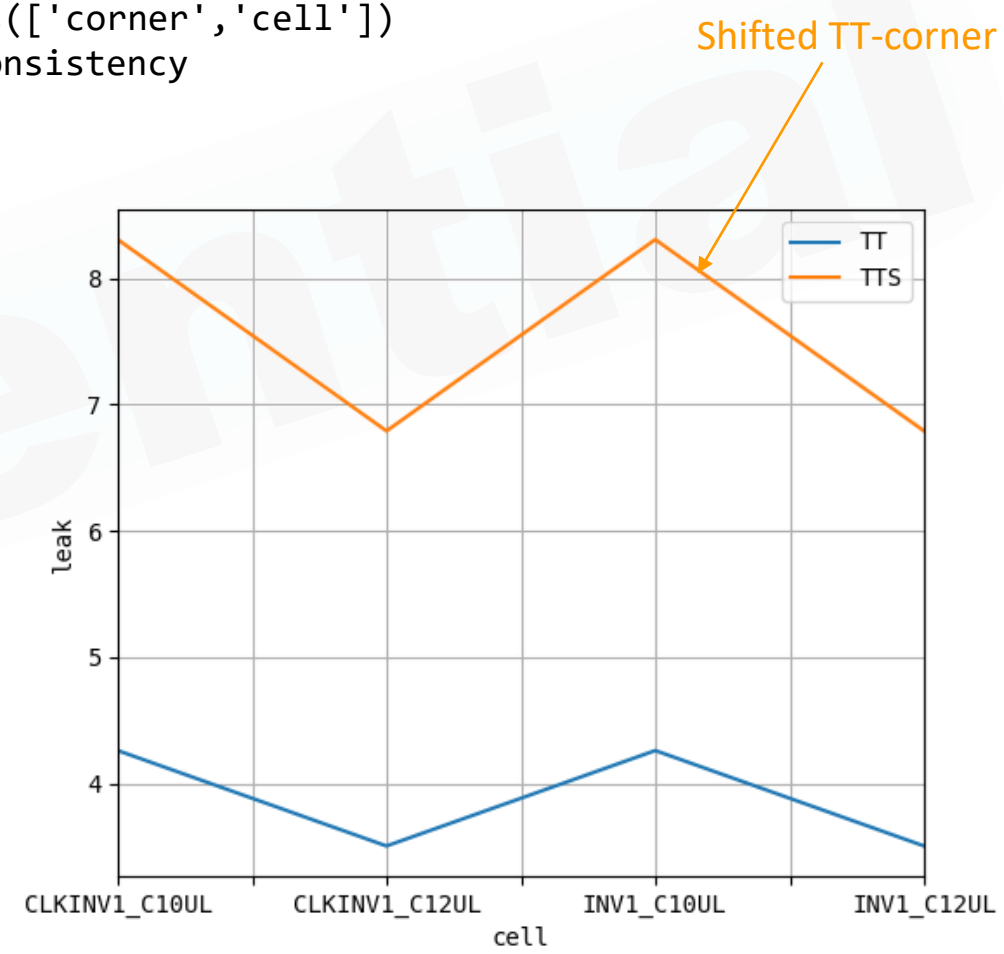
Analysis: Trend

```
#%% basic data analysis (visualization & trend)
dt = df.set_index(['cell','corner','volt'])
d = dt.loc['NAND2V1_C10UL','TT'].sort_values(['volt','temp'])
d[['temp','area','leak','da','pa']]

fig,ax = plt.subplots(figsize=(6,6),sharex=True)
d.loc[0.29].plot(x='temp',y='leak',marker='o',alpha=0.8,ax=ax,label='0.29V')
d.loc[0.31].plot(x='temp',y='leak',marker='^',alpha=0.8,ax=ax,label='0.31V')
d.loc[0.33].plot(x='temp',y='leak',marker='x',alpha=0.8,ax=ax,label='0.33V')
d.loc[0.35].plot(x='temp',y='leak',marker='*',alpha=0.8,ax=ax,label='0.35V')
ax.set_title('NAND2V1_C10UL')
ax.set_xlabel('temperature')
ax.set_ylabel('leakage')
ax.grid()
ax.legend()
```



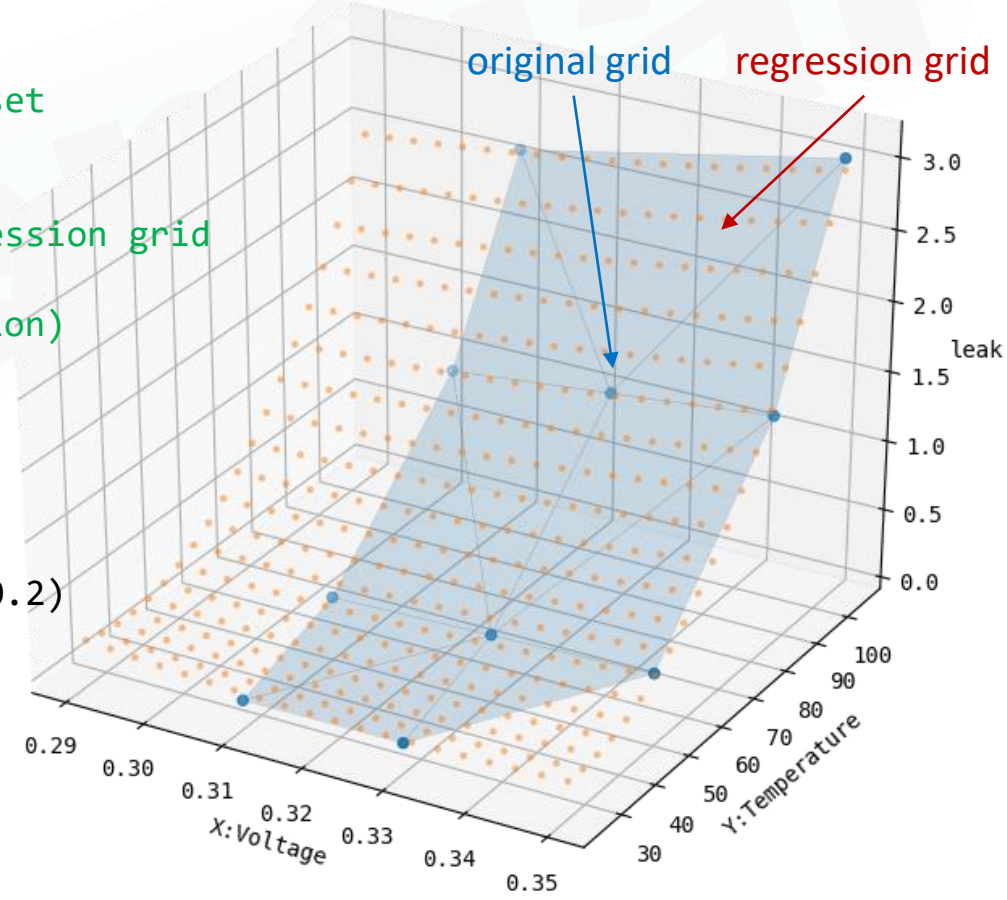
Analysis: Comparison

```
#%% basic data comparison
dt = df.set_index(['cell','corner','volt','temp'])
dc = dt.filter(regex='.*INV.*UL.*TT.*0\.31.*85',axis=0)
dc = dc.reset_index().set_index(['corner','cell']).sort_values(['corner','cell'])
all(dc.loc['TT'].index == dc.loc['TTS'].index) # check cell consistency

feature = 'leak'
fig,ax = plt.subplots(sharex=True)
fig.set_size_inches((12,6))
dc.loc['TT'].plot(y=feature,ax=ax,use_index=True,label='TT')
dc.loc['TTS'].plot(y=feature,ax=ax,use_index=True,label='TTS')
ax.set_xlabel('cell')
ax.set_ylabel(feature)
ax.grid()
ax.legend()
```



Regression Model

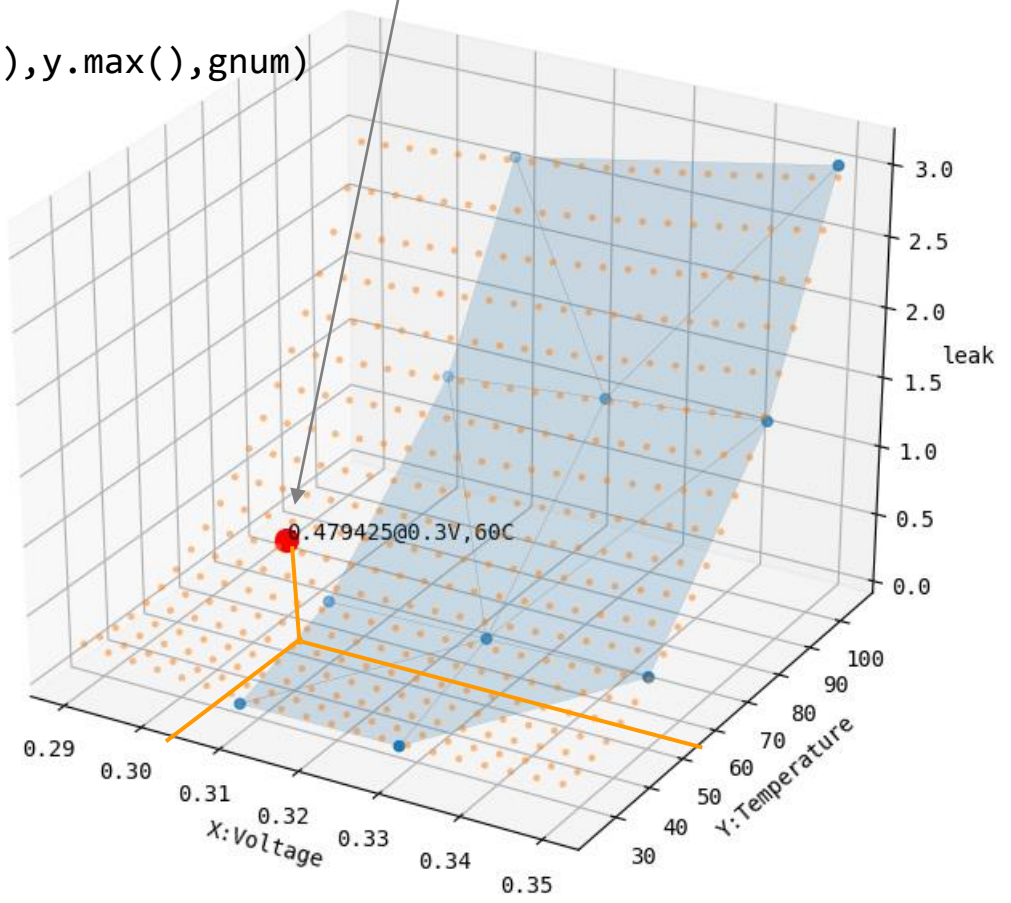
```
from mpl_toolkits.mplot3d import Axes3D
import scipy.linalg

dt = df.reset_index().set_index(['cell','corner'])
d = dt.loc['CLKINV1_C10UL','TT'].sort_values(['volt','temp'])
d = d[d['volt']!=0.75] # drop 0.75V (insufficient data points)

x,y,z = d['volt'].values,d['temp'].values,d['leak'].values
X = np.array([np.ones_like(x),x,x**2,y,y**2,x*y]).T # training set

C,_,_,_ = scipy.linalg.lstsq(X,z) # LS regression coefficient
rx,ry = np.linspace(0.29,0.35,20),np.linspace(25,105,20) # regression grid
gx,gy = np.meshgrid(rx,ry)    # mesh grid
tx,ty = gx.ravel(),gy.ravel() # mesh grid in vector (one dimension)
T = np.array([np.ones_like(tx),tx,tx**2,ty,ty**2,tx*ty]).T
gz = np.dot(T,C).reshape(gx.shape) # prediction (inner product)

f = plt.figure(figsize=(8,7))
ax = Axes3D(f)
ax.plot_trisurf(d['volt'],d['temp'],d[feature],ec='gray',alpha=0.2)
ax.scatter(d['volt'],d['temp'],d[feature],label='original')
ax.scatter(tx,ty,gz.ravel(),alpha=0.4,s=5,label='regression')
ax.set_xlabel('X:Voltage')
ax.set_ylabel('Y:Temperature')
plt.show()
```



Prediction

```
#%% encapsulate method as a function
def modelFitPredict(x,y,z,gnum=20):
    x,y,z = map(np.array,[x,y,z]) # convert to numpy array
    X = np.array([np.ones_like(x),x,x**2,y,y**2,x*y]).T
    C,_,_,_ = scipy.linalg.lstsq(X,z) # LS regression

    # regression grid
    px,py = np.linspace(x.min(),x.max(),gnum),np.linspace(y.min(),y.max(),gnum)
    gx,gy = np.meshgrid(px,py)
    tx,ty = gx.ravel(),gy.ravel()
    T = np.array([np.ones_like(tx),tx,tx**2,ty,ty**2,tx*ty]).T
    gz = np.dot(T,C).reshape(gx.shape) # prediction
    return C,(gx,gy,gz)

# model fitting
x,y,z = d['volt'].values,d['temp'].values,d['leak'].values
C,_ = modelFitPredict(x,y,z)

# specified condition for prediction
volt,temp = 0.30,60
T = np.array([1,volt,volt**2,temp,temp**2,volt*temp]).T
p = np.dot(T,C) # inner-product
```

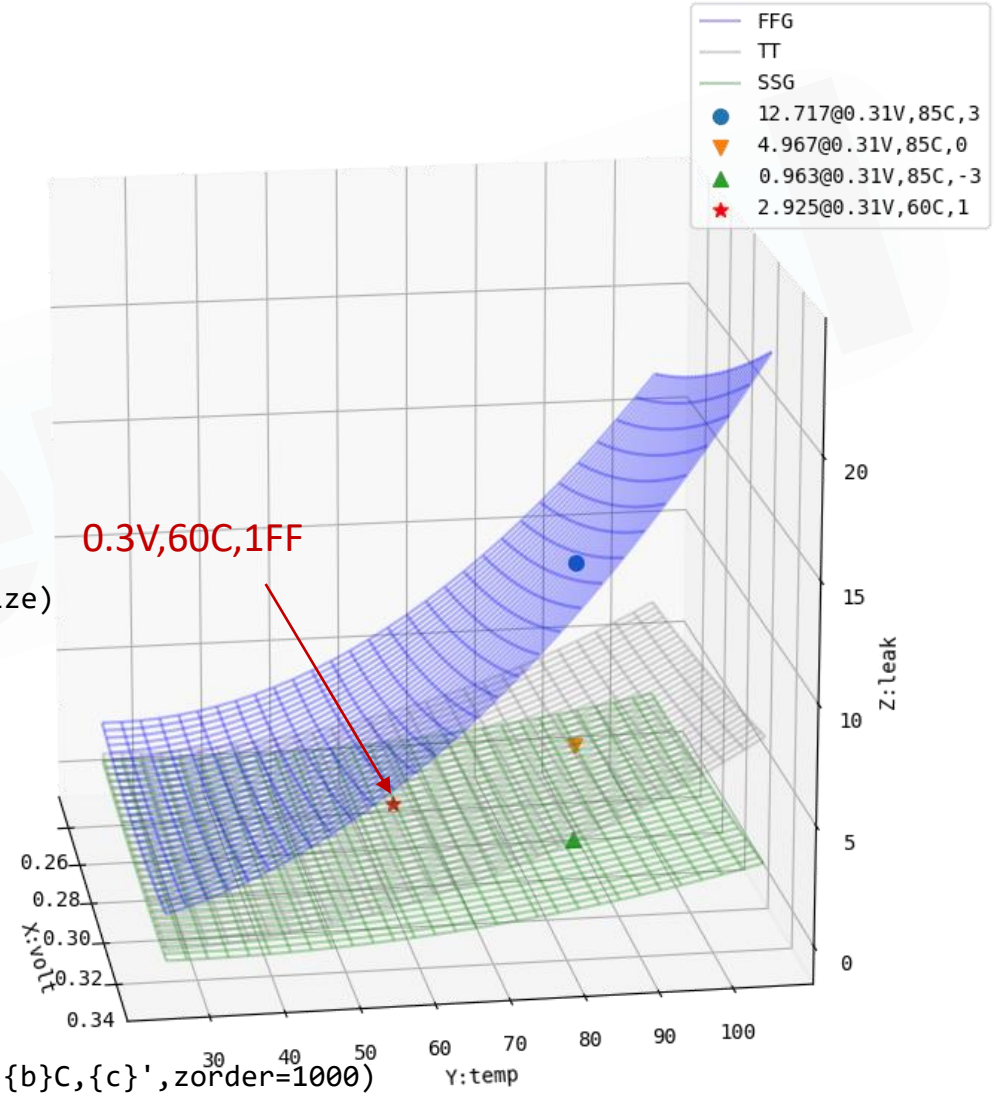
```
# prediction@ 0.31V,60C
volt,temp = 0.30,60
T = np.array([1,volt,volt**2,temp,temp**2,volt*temp]).T
p = np.dot(T,C) # inner-product
```



DIGWISE TECHNOLOGY
慥智科技有限公司

Multi-dimension Modeling

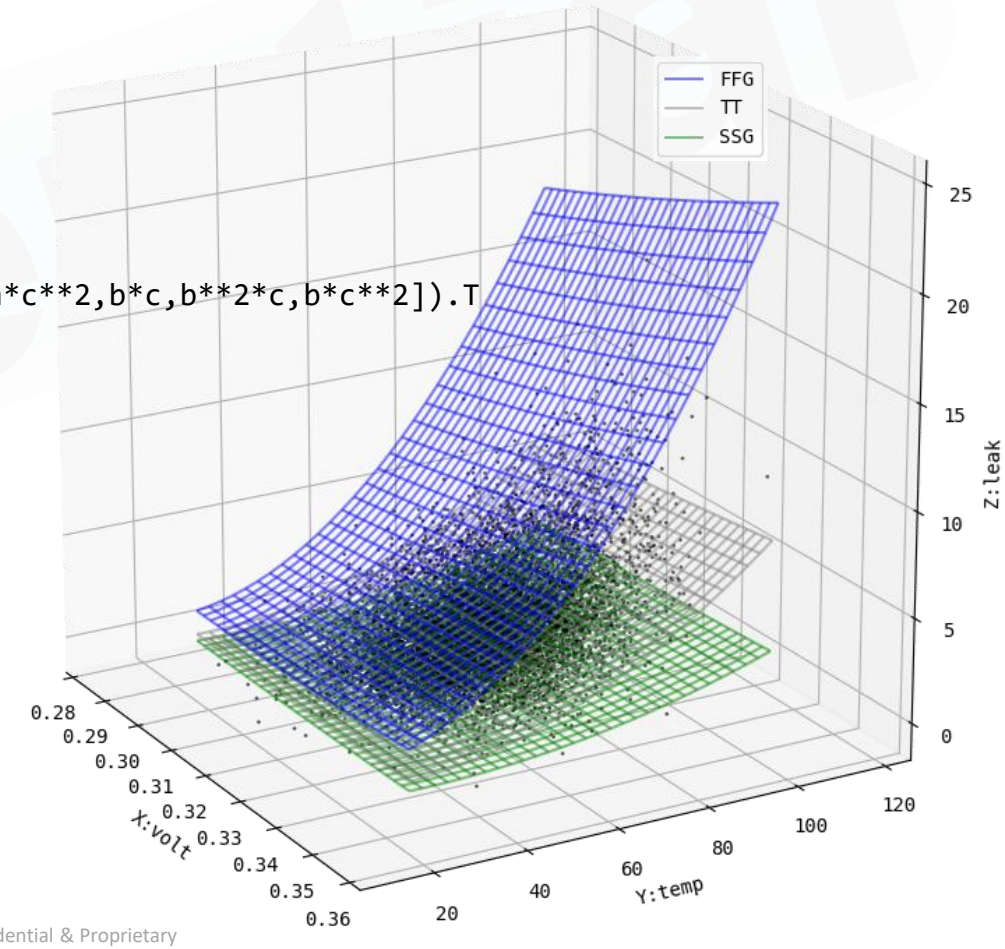
```
# model fitting f(volt,temp) for each corner
coeffL = {}
for corner in ['FFG','TT','SSG']:
    d = dt.loc[corner]
    x,y,z = map(np.array,zip(*d[['volt','temp','leak']].values))
    C = modelFit(x,y,z) # 3 sigma
    coeffL[corner] = C

# regression grid, x:volt,y:temp
px,py = np.linspace(0.25,0.35,gnum),np.linspace(25,105,gnum)
gx,gy = np.meshgrid(px,py)
tx,ty = gx.ravel(),gy.ravel()
T = np.array([np.ones_like(tx),tx,tx**2,ty,ty**2,tx*ty]).T
ff,tt,ss = np.dot(T,coeffL['FFG']),np.dot(T,coeffL['TT']),np.dot(T,coeffL['SSG'])

size = gnum**2
a,b,c = np.array(list(tx)*3),np.array(list(ty)*3),np.array([3]*size+[0]*size+[-3]*size)
z = np.array(list(ff)+list(tt)+list(ss))
X = np.array([np.ones_like(z),a,a**2,b,b**2,c,c**2,a*b,a*c,b*c]).T
C,_,_,_ = scipy.linalg.lstsq(X,z) # LS regression, as f(V,T,P)

f = plt.figure(figsize=(8,8)); ax = Axes3D(f)
ax.plot_wireframe(gx,gy,ff.reshape(gx.shape),color='blue',alpha=0.3,label='FFG')
ax.plot_wireframe(gx,gy,tt.reshape(gx.shape),color='gray',alpha=0.3,label='TT')
ax.plot_wireframe(gx,gy,ss.reshape(gx.shape),color='green',alpha=0.3,label='SSG')
examL = [('FFG',0.31,85,3),('TT',0.31,85,0),('SSG',0.31,85,-3),('special',0.3,60,1)]
for ii,(tag,a,b,c) in enumerate(examL):
    T = np.array([np.ones_like(a),a,a**2,b,b**2,c,c**2,a*b,a*c,b*c]).T
    p = np.dot(T,C)
    ax.scatter(a,b,p,color=colL[ii],marker=mrkL[ii],s=50,label=f'{tag}:{p:.3f}@{a}V,{b}C,{c}',zorder=1000)
```



DIGWISE TECHNOLOGY
鋭智科技有限公司

Probability Density

```
# model fitting f(V,T,P)
size = gnum**2
a,b,c = np.array(list(tx)*3),np.array(list(ty)*3),np.array([3]*size+[0]*size+[-3]*size)
z = np.array(list(ff)+list(tt)+list(ss)) # training set derived from each corner model respectively
X = np.array([np.ones_like(a),a,a**2,b,b**2,c,c**2,a*b,a**2*b,a*b**2,a*c,a**2*c,a*c**2,b*c,b**2*c,b*c**2]).T
C,_,_,_ = scipy.linalg.lstsq(X,z) # LS regression, as f(V,T,P)

# random samples
size = 10000
V = np.random.normal((0.35+0.29)/2,(0.35-0.29)/6,size)
T = np.random.normal((105+25)/2,(105-25)/6,size)
P = np.random.normal(0,1,size)

a,b,c = V,T,P
X = np.array([np.ones_like(a),a,a**2,b,b**2,c,c**2,a*b,a**2*b,a*b**2,a*c,a**2*c,a*c**2,b*c,b**2*c,b*c**2]).T
p = np.dot(X,C) # batch prediction

f = plt.figure(figsize=(8,8)); ax = Axes3D(f)
ax.plot_wireframe(gx,gy,ff.reshape(gx.shape),color='blue',alpha=0.5,label='FFG')
ax.plot_wireframe(gx,gy,tt.reshape(gx.shape),color='black',alpha=0.5,label='TT')
ax.plot_wireframe(gx,gy,ss.reshape(gx.shape),color='green',alpha=0.5,label='SSG’)

ax.scatter(a,b,p,s=2,alpha=0.3,color='gray',edgecolor=None)
ax.view_init(20,-30)
ax.set_xlabel('X:volt')
ax.set_ylabel('Y:temp')
ax.set_zlabel('Z:leak')
plt.show()
```



DIGWISE TECHNOLOGY
鋭智科技有限公司