KAN_coxCAN_intro

June 20, 2025

[1]:
```
# Install coxkan
! pip install coxkan
```

```
Requirement already satisfied: coxkan in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (0.0.2)
Requirement already satisfied: pykan==0.0.2 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from coxkan) (0.0.2)
Requirement already satisfied: lifelines>=0.28.0 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from coxkan) (0.30.0)
Requirement already satisfied: torch>=2.3.1 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from coxkan) (2.7.0+cu128)
Requirement already satisfied: tqdm in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from coxkan) (4.66.2)
Requirement already satisfied: optuna>=3.6.1 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from coxkan) (4.4.0)
Requirement already satisfied: torchtuples==0.2.2 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from coxkan) (0.2.2)
Requirement already satisfied: scikit-learn>=1.5.0 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from coxkan) (1.7.0)
Requirement already satisfied: feather-format>=0.4.0 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from coxkan) (0.4.1)
Requirement already satisfied: h5py>=2.9.0 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from coxkan) (3.14.0)
Requirement already satisfied: numba>=0.44 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from coxkan) (0.61.2)
Requirement already satisfied: requests>=2.22.0 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from coxkan) (2.32.3)
Requirement already satisfied: py7zr>=0.11.3 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from coxkan) (1.0.0)
Requirement already satisfied: scipy>=1.13.1 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from coxkan) (1.15.2)
Requirement already satisfied: numpy>=1.15.4 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from
torchtuples==0.2.2->coxkan) (1.26.4)
Requirement already satisfied: pandas>=0.24.2 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from
torchtuples==0.2.2->coxkan) (2.1.4)
Requirement already satisfied: matplotlib>=3.0.3 in
```

```
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from
torchtuples==0.2.2->coxkan) (3.6.2)
Requirement already satisfied: pyarrow>=0.4.0 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from feather-
format>=0.4.0->coxkan) (20.0.0)
Requirement already satisfied: autograd>=1.5 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from
lifelines>=0.28.0->coxkan) (1.8.0)
Requirement already satisfied: autograd-gamma>=0.3 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from
lifelines>=0.28.0->coxkan) (0.5.0)
Requirement already satisfied: formulaic>=0.2.2 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from
lifelines>=0.28.0->coxkan) (1.1.1)
Requirement already satisfied: llvmlite<0.45,>=0.44.0dev0 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from numba>=0.44->coxkan)
(0.44.0)
Requirement already satisfied: alembic>=1.5.0 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from optuna>=3.6.1->coxkan)
(1.16.2)
Requirement already satisfied: colorlog in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from optuna>=3.6.1->coxkan)
(6.9.0)
Requirement already satisfied: packaging>=20.0 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from optuna>=3.6.1->coxkan)
(24.2)
Requirement already satisfied: sqlalchemy>=1.4.2 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from optuna>=3.6.1->coxkan)
(2.0.41)
Requirement already satisfied: PyYAML in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from optuna>=3.6.1->coxkan)
(6.0.2)
Requirement already satisfied: texttable in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from py7zr>=0.11.3->coxkan)
(1.7.0)
Requirement already satisfied: pycryptodomex>=3.20.0 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from py7zr>=0.11.3->coxkan)
(3.23.0)
Requirement already satisfied: brotli>=1.1.0 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from py7zr>=0.11.3->coxkan)
(1.1.0)
Requirement already satisfied: psutil in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from py7zr>=0.11.3->coxkan)
(5.9.0)
Requirement already satisfied: pyzstd>=0.16.1 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from py7zr>=0.11.3->coxkan)
(0.17.0)
Requirement already satisfied: pyppmd<1.3.0,>=1.1.0 in
```

```
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from py7zr>=0.11.3->coxkan)
(1.2.0)
Requirement already satisfied: pybcj<1.1.0,>=1.0.0 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from py7zr>=0.11.3->coxkan)
(1.0.6)
Requirement already satisfied: multivolumefile>=0.2.3 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from py7zr>=0.11.3->coxkan)
(0.2.3)
Requirement already satisfied: inflate64<1.1.0,>=1.0.0 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from py7zr>=0.11.3->coxkan)
(1.0.3)
Requirement already satisfied: charset-normalizer<4,>=2 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from
requests>=2.22.0->coxkan) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from
requests>=2.22.0->coxkan) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from
requests>=2.22.0->coxkan) (2.3.0)
Requirement already satisfied: certifi>=2017.4.17 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from
requests>=2.22.0->coxkan) (2025.4.26)
Requirement already satisfied: joblib>=1.2.0 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from scikit-
learn>=1.5.0->coxkan) (1.4.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from scikit-
learn>=1.5.0->coxkan) (3.6.0)
Requirement already satisfied: filelock in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from torch>=2.3.1->coxkan)
(3.13.1)
Requirement already satisfied: typing-extensions>=4.10.0 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from torch>=2.3.1->coxkan)
(4.14.0)
Requirement already satisfied: sympy>=1.13.3 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from torch>=2.3.1->coxkan)
(1.13.3)
Requirement already satisfied: networkx in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from torch>=2.3.1->coxkan)
(3.3)
Requirement already satisfied: jinja2 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from torch>=2.3.1->coxkan)
(3.1.4)
Requirement already satisfied: fsspec in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from torch>=2.3.1->coxkan)
(2024.6.1)
Requirement already satisfied: colorama in
```

```
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from tqdm->coxkan) (0.4.6)
Requirement already satisfied: Mako in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from
alembic>=1.5.0->optuna>=3.6.1->coxkan) (1.3.10)
Requirement already satisfied: tomli in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from
alembic>=1.5.0->optuna>=3.6.1->coxkan) (2.0.1)
Requirement already satisfied: interface-meta>=1.2.0 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from
formulaic>=0.2.2->lifelines>=0.28.0->coxkan) (1.3.0)
Requirement already satisfied: wrapt>=1.0 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from
formulaic>=0.2.2->lifelines>=0.28.0->coxkan) (1.17.2)
Requirement already satisfied: contourpy>=1.0.1 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from
matplotlib>=3.0.3->torchtuples==0.2.2->coxkan) (1.3.2)
Requirement already satisfied: cycler>=0.10 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from
matplotlib>=3.0.3->torchtuples==0.2.2->coxkan) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from
matplotlib>=3.0.3->torchtuples==0.2.2->coxkan) (4.57.0)
Requirement already satisfied: kiwisolver>=1.0.1 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from
matplotlib>=3.0.3->torchtuples==0.2.2->coxkan) (1.4.8)
Requirement already satisfied: pillow>=6.2.0 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from
matplotlib>=3.0.3->torchtuples==0.2.2->coxkan) (11.0.0)
Requirement already satisfied: pyparsing>=2.2.1 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from
matplotlib>=3.0.3->torchtuples==0.2.2->coxkan) (3.2.3)
Requirement already satisfied: python-dateutil>=2.7 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from
matplotlib>=3.0.3->torchtuples==0.2.2->coxkan) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from
pandas>=0.24.2->torchtuples==0.2.2->coxkan) (2024.1)
Requirement already satisfied: tzdata>=2022.1 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from
pandas>=0.24.2->torchtuples==0.2.2->coxkan) (2023.3)
Requirement already satisfied: greenlet>=1 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from
sqlalchemy>=1.4.2->optuna>=3.6.1->coxkan) (3.2.3)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from
sympy>=1.13.3->torch>=2.3.1->coxkan) (1.3.0)
Requirement already satisfied: MarkupSafe>=2.0 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from
```

```
jinja2->torch>=2.3.1->coxkan) (2.1.5)
Requirement already satisfied: six>=1.5 in
c:\users\jorge\anaconda3\envs\kan\lib\site-packages (from python-
dateutil>=2.7->matplotlib>=3.0.3->torchtuples==0.2.2->coxkan) (1.17.0)
```

1 CoxKAN Introductory Demo

[2]:
```python
from coxkan import CoxKAN
from sklearn.model_selection import train_test_split
import numpy as np
```

1.0.1 Synthetic Dataset Example

The code below generates a synthetic survival dataset under the hazard function

$$\text{Hazard, } h(t, \mathbf{x}) = 0.01e^{\theta(\mathbf{x})},$$

where

$$\text{Log-Partial Hazard, } \theta(\mathbf{x}) = \tanh(5x_1) + \sin(2\pi x_2)$$

and a **uniform censoring distribution**.

[3]:
```python
from coxkan.datasets import create_dataset

log_partial_hazard = lambda x1, x2: np.tanh(5*x1) + np.sin(2*np.pi*x2)

df = create_dataset(log_partial_hazard, baseline_hazard=0.01, n_samples=10000,␣
↪seed=42)
df_train, df_test = train_test_split(df, test_size=0.2, random_state=42)

df_train.head()
```

```
Concordance index of true expression: 0.7524
```

[3]:

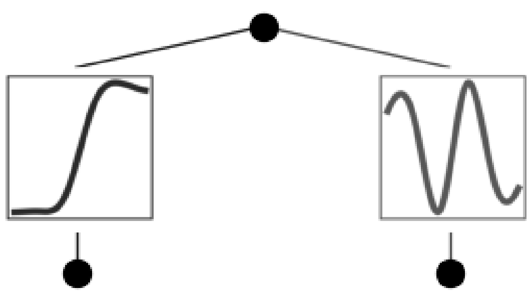
	x1	x2	duration	event
9254	0.541629	-0.706251	42.270669	1
1561	-0.526259	-0.492606	54.283488	1
1670	-0.238753	-0.326589	361.569903	1
6087	-0.588024	0.742029	57.335278	0
6669	-0.739364	-0.302907	95.975668	1

1.0.2 Train CoxKAN:

[4]:
```python
ckan = CoxKAN(width=[2,1], grid=5, seed=42)

_ = ckan.train(
```

```
    df_train,
    df_test,
    duration_col='duration',
    event_col='event',
    opt='Adam',
    lr=0.01,
    steps=100)

# evaluate CoxKAN
cindex = ckan.cindex(df_test)
print("\nCoxKAN C-Index: ", cindex)

# plot CoxKAN
fig = ckan.plot()
```

```
train loss: 2.77e+00 | val loss: 2.50e+00: 100%|          | 100/100
[00:51<00:00,  1.92it/s]


CoxKAN C-Index:  0.7553786667818724
```



1.0.3 Symbolic Fitting:

```
[5]: # auto-symbolic fitting
_ = ckan.auto_symbolic(lib=['x^2', 'sin', 'exp', 'log', 'sqrt', 'tanh'],␣
 ↪verbose=False)

# train affine parameters
_ = ckan.train(df_train, df_test, duration_col='duration', event_col='event',␣
 ↪opt='LBFGS', steps=10)

display(ckan.symbolic_formula(floating_digit=1)[0][0])
```

```
train loss: 2.77e+00 | val loss: 2.50e+00: 100%|          | 10/10
[00:12<00:00,  1.26s/it]
```

$-1.0 \sin (6.3 x_2 + 9.4) + 1.0 \tanh (4.4 x_1)$

1.0.4 Symbolic Expression Evaluation CoxKAN:

We see CoxKAN approximately recovers the true log-partial hazard:

$\hat{\theta}_{KAN} = \tanh(4.4x_1) - \sin(6.3x_2 + 9.4) \approx \tanh(5x_1) - \sin(2\pi x_2 + 3\pi) = \tanh(5x_1) + \sin(2\pi x_2)$

[6]:
```
log_partial_hazard = lambda x1, x2: np.tanh(5*x1) + np.sin(2*np.pi*x2)
Obtained_symbolic_expression =  lambda x1, x2: (-1.0*np.sin((6.3*x2)+9.4))+(1.
↪0*np.tanh((4.4*x1)))
```

[7]:
```
check1 = []
check2 = []
for i in np.arange(0,2*np.pi,0.1):
    for j in np.arange(0,2*np.pi,0.1):
        check1.append(log_partial_hazard(i,j))
        check2.append(Obtained_symbolic_expression(i,j))
```

[8]:
```
results = []
for i in range(len(check1)):
    results.append((check2[i] - check1[i])**2)

rmse = np.sqrt((sum(results)/len(results)))
rmse
```

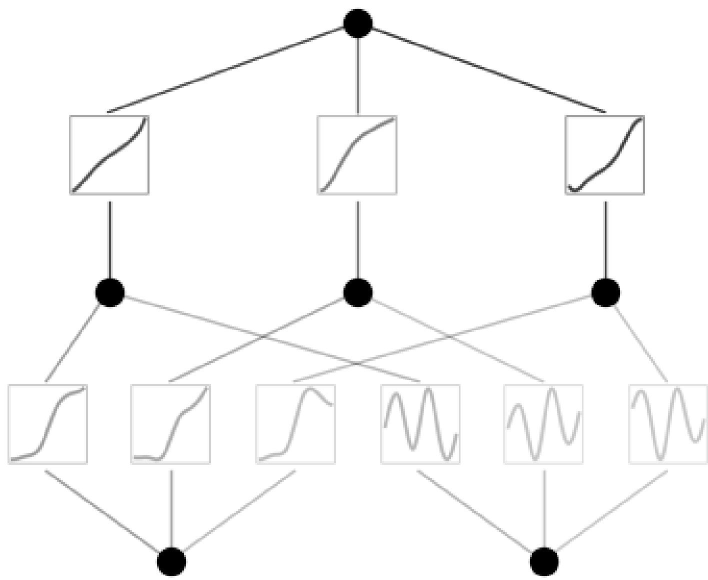
[8]: 0.03078551065178464

1.0.5 Train CoxKAN:

[9]:
```
ckan = CoxKAN(width=[2,3,1], grid=5, seed=42)

_ = ckan.train(
    df_train,
    df_test,
    duration_col='duration',
    event_col='event',
    opt='Adam',
    lr=0.01,
    steps=100)

# evaluate CoxKAN
cindex = ckan.cindex(df_test)
print("\nCoxKAN C-Index: ", cindex)

# plot CoxKAN
fig = ckan.plot()
```

```
train loss: 2.77e+00 | val loss: 2.50e+00: 100%|          | 100/100
[01:21<00:00,  1.23it/s]
```

```
CoxKAN C-Index:  0.7576267211425115
```



1.0.6 Symbolic Fitting:

[12]:
```
# auto-symbolic fitting
_ = ckan.auto_symbolic(lib=['x','x^2','x^3','x^4','sin', 'exp', 'log', 'sqrt',␣
↪'tanh'], verbose=False)

# train affine parameters
_ = ckan.train(df_train, df_test, duration_col='duration', event_col='event',␣
↪opt='LBFGS', steps=10)

display(ckan.symbolic_formula(floating_digit=1)[0][0])
```

```
train loss: 2.77e+00 | val loss: 2.50e+00: 100%|          | 10/10
[00:46<00:00,  4.66s/it]
```

$3.3\sqrt{-0.2\sin\left(5.9x_2-3.3\right)+0.1\tanh\left(3.8x_1-0.4\right)+1}+0.5\tanh\left(-0.6\sin\left(6.5x_2-9.4\right)+1.5\tanh\left(3.2x_1-0.2\right)+1\right.$
$2.7\tanh\left(0.3\sin\left(6.5x_2+9.5\right)-0.3\tanh\left(5.4x_1\right)+0.6\right)$

1.0.7 Symbolic Expression Evaluation CoxKAN:

We see CoxKAN approximately recovers the true log-partial hazard:

$\hat{\theta}_{KAN}=\tanh(4.4x_1)-\sin(6.3x_2+9.4)\approx\tanh(5x_1)-\sin(2\pi x_2+3\pi)=\tanh(5x_1)+\sin(2\pi x_2)$

[21]:
```
log_partial_hazard = lambda x1, x2: np.tanh(5*x1) + np.sin(2*np.pi*x2)
long_symbolic_expression =  lambda x1, x2: (3.3*np.sqrt((-0.2*np.sin((5.9*x2)-3.
↪3))+(0.1*np.tanh((3.8*x1)-0.4))+1))+(0.5*np.tanh((-0.6*np.sin((6.5*x2)-9.
↪4))+(1.5*np.tanh((3.2*x1)-0.2))+1.6))-(2.7*np.tanh(0.3*np.sin((6.4*x2)+9.
↪5)-(0.3*np.tanh((5.4*x1)))+0.6))
```

[22]:
```
l1 = []
l2 = []
for i in np.arange(0,2*np.pi,0.1):
    for j in np.arange(0,2*np.pi,0.1):
        l1.append(log_partial_hazard(i,j))
        l2.append(long_symbolic_expression(i,j))
```

[23]:
```
len(l1)
```

[23]: 3969

[24]:
```
len(l2)
```

[24]: 3969

[25]:
```
# Create new list of equal length for your predictions
l3 = []
```

[26]:
```
for i in range(len(l1)):
    l3.append((l2[i] - l1[i])**2)

#print(l3)
```

[27]:
```
rmse = np.sqrt((sum(l3)/len(l3)))
rmse
```

[27]: 2.2231100852976673

1.0.8 Real dataset example

[28]:
```
from coxkan.datasets import gbsg

# load dataset
df_train, df_test = gbsg.load(split=True)
name, duration_col, event_col, covariates = gbsg.metadata()

# init CoxKAN
ckan = CoxKAN(width=[len(covariates), 1], seed=42)

# pre-process and register data
df_train, df_test = ckan.process_data(df_train, df_test, duration_col,␣
↪event_col, normalization='standard')
```

```
# train CoxKAN
_ = ckan.train(
    df_train,
    df_test,
    duration_col=duration_col,
    event_col=event_col,
    opt='Adam',
    lr=0.01,
    steps=100)

print("\nCoxKAN C-Index: ", ckan.cindex(df_test))

# Auto symbolic fitting
fit_success = ckan.auto_symbolic(verbose=False)
display(ckan.symbolic_formula(floating_digit=2)[0][0])

# Plot coxkan
fig = ckan.plot(beta=20)
```

```
Using default train-test split (used in DeepSurv paper).

train loss: 2.60e+00 | val loss: 2.41e+00: 100%|          | 100/100
[00:17<00:00,  5.69it/s]


CoxKAN C-Index:  0.679797402909703
```

$$\begin{cases} 0.06 & \text{for } meno = 0 \\ 0.22 & \text{for } meno = 1.0 \\ \text{NaN} & \text{otherwise} \end{cases} + \begin{cases} 0.36 & \text{for } hormon = 0 \\ 0.02 & \text{for } hormon = 1.0 \\ \text{NaN} & \text{otherwise} \end{cases} + \begin{cases} -0.16 & \text{for } size = 0 \\ 0.09 & \text{for } size = 1.0 \\ 0.36 & \text{for } size = 2.0 \\ \text{NaN} & \text{otherwise} \end{cases} + 0.759 - 1.16e^{-0.03(-nodes-0.59)^2} - 0.31e^{-5.69(1-0.02age)^2}$$