python lifelines moduleを使って生存分析を行う

In [1]:

```
#  必要なmoduleのインストール
# 以下、インストール未であれば　ターミナルで実行
# pip install numpy
# pip install pandas
# pip install matplotlib
# pip install lifelines
```

In [2]:

```
# module 呼び出しとセッテイング
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
from lifelines import KaplanMeierFitter
kmf= KaplanMeierFitter()
```

In [3]:

```
# plotのスタイルを設定
plt.style.use("seaborn-whitegrid")
```

サンプルデータで行う

In [4]:

```
# sample dataの読み込み
from lifelines.datasets import load_dd
```

In [5]:

```
# 確認
df = load_dd()
df.shape
```

Out[5]:

```
(1808, 12)
```

In [6]:

```
df.head()
```

Out[6]:

	ctryname	cowcode2	politycode	un_region_name	un_continent_name	ehe
0	Afghanistan	700	700.0	Southern Asia	Asia	Mohamm Zahir Sha
1	Afghanistan	700	700.0	Southern Asia	Asia	Sardar Mohamm Daoud
2	Afghanistan	700	700.0	Southern Asia	Asia	Mohamm Zahir Sha
3	Afghanistan	700	700.0	Southern Asia	Asia	Sardar Mohamm Daoud
4	Afghanistan	700	700.0	Southern Asia	Asia	Nur Mohamm Taraki

In [7]:

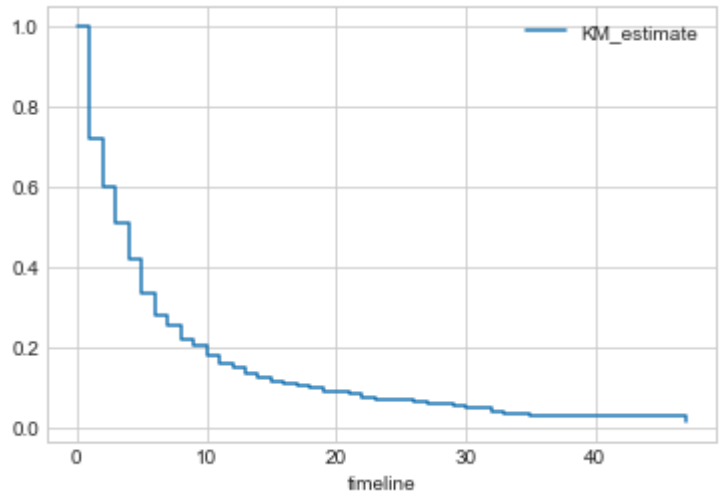
```
# durationを生存時間、observedをイベントとみなす
time = df["duration"]
event = df["observed"]
```

In [8]:

```
# Fit ファイルを作成
fit=kmf.fit(time, event_observed=event)
#もっともシンプルなプロット .survival_function_.plot
fig = plt.figure()
ax = fig.add_subplot(111)
fit.plot(ax=ax,ci_show=False)
```

Out[8]:

<matplotlib.axes._subplots.AxesSubplot at 0x1a1c4603c8>

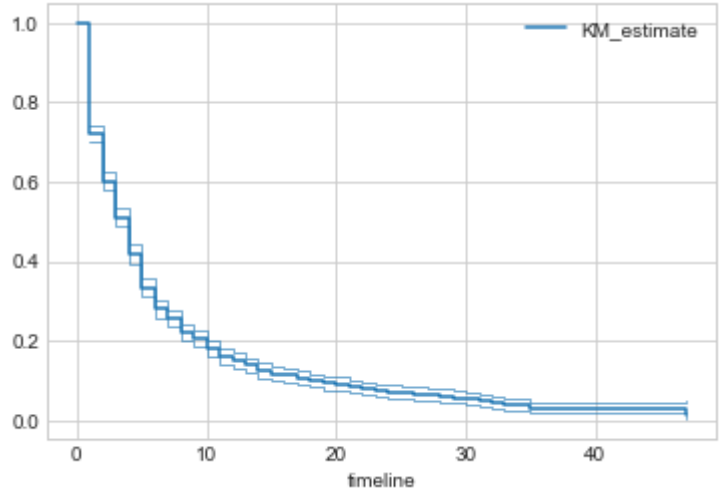


In [9]:

```
# 信頼区間を追加 .plot(ci_force_lines=True)
fit=kmf.fit(time, event_observed=event)
fig = plt.figure()
ax = fig.add_subplot(111)
fit.plot(ax=ax, ci_force_lines=True)
```

Out[9]:

<matplotlib.axes._subplots.AxesSubplot at 0x1a1c17deb8>

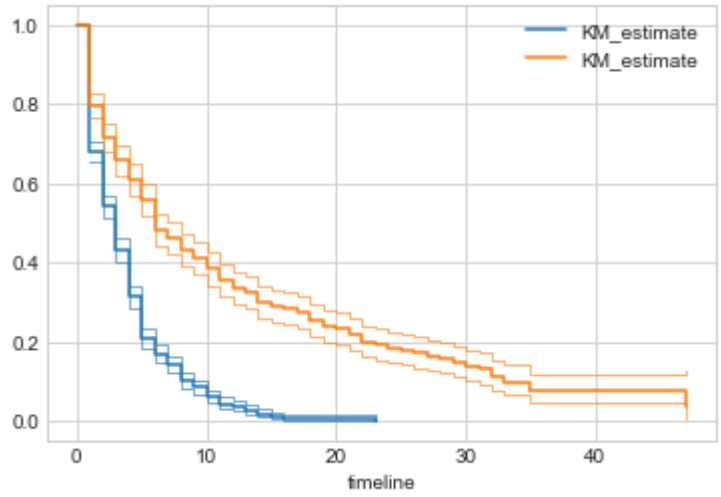


In [10]:

```
# democratic vs non-democratic　で2群に分けて重ねてプロット
fig = plt.figure()
ax = fig.add_subplot(111)

democratic = df["democracy"] == "Democracy"
fit1 = kmf.fit(time[democratic], event_observed=event[democratic])
fit1.plot(ax=ax, ci_force_lines=True)

fit2 = kmf.fit(time[~democratic], event_observed=event[~democratic])
fit2.plot(ax=ax, ci_force_lines=True)
```

Out[10]:

<matplotlib.axes._subplots.AxesSubplot at 0x1a1c4bf4e0>

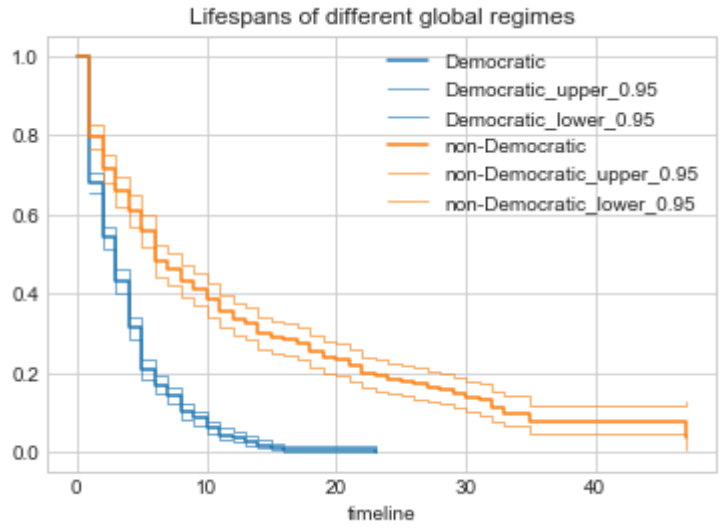


In [11]:

```
# タイトルと　　legendを追加する
fig = plt.figure()
ax = fig.add_subplot(111)
ax.set_title("Lifespans of different global regimes")

democratic = df["democracy"] == "Democracy"
fit1 = kmf.fit(time[democratic], event_observed=event[democratic],label="Democratic")
fit1.plot(ax=ax, ci_force_lines=True)

fit2 = kmf.fit(time[~democratic], event_observed=event[~democratic],label="non-Democratic")
fit2.plot(ax=ax, ci_force_lines=True)

plt.legend()
```

Out[11]:

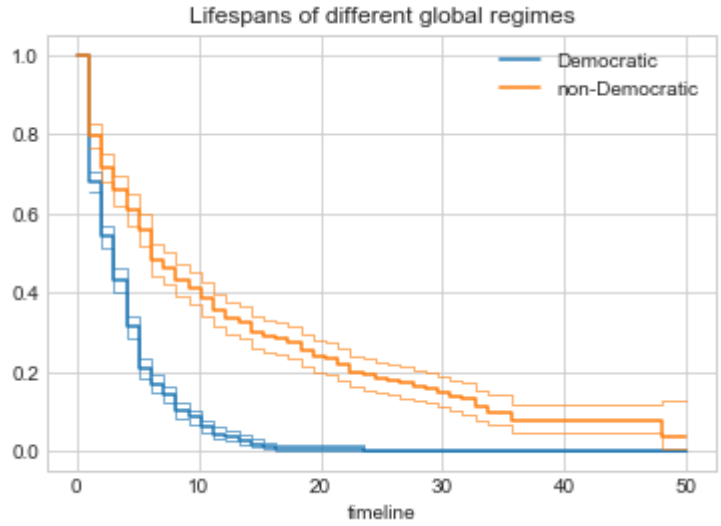
<matplotlib.legend.Legend at 0x1a1c5586a0>



In [12]:

```
# timeline を設定して各地点での生存率を求める
fig = plt.figure()
ax = fig.add_subplot(111)
ax.set_title("Lifespans of different global regimes")

t=np.linspace(0,50) # timepoint を指定する 0から50まで
democratic = df["democracy"] == "Democracy"
fit1 = kmf.fit(time[democratic], event_observed=event[democratic],timeline=t,label="Democratic")
fit1.plot(ax=ax, ci_force_lines=True)
print( "Median survival time of democratic:", fit1.median_ ) # 中央値

fit2 = kmf.fit(time[~democratic], event_observed=event[~democratic],timeline=t,label="non-Democr
atic")
fit2.plot(ax=ax, ci_force_lines=True)
print("Median survival time of non-democratic:", fit2.median_) # 中央値
```

```
Median survival time of democratic: 3.0612244897959187
Median survival time of non-democratic: 6.122448979591837
```



2群の生存率を比較する

In [13]:

```
from lifelines.statistics import logrank_test

results = logrank_test(time[democratic], time[~democratic], event[democratic], event[~democratic
], alpha=.99)
print( results.summary)
```

```
   test_statistic             p
0      260.469539  1.355714e-58
```

In [14]:

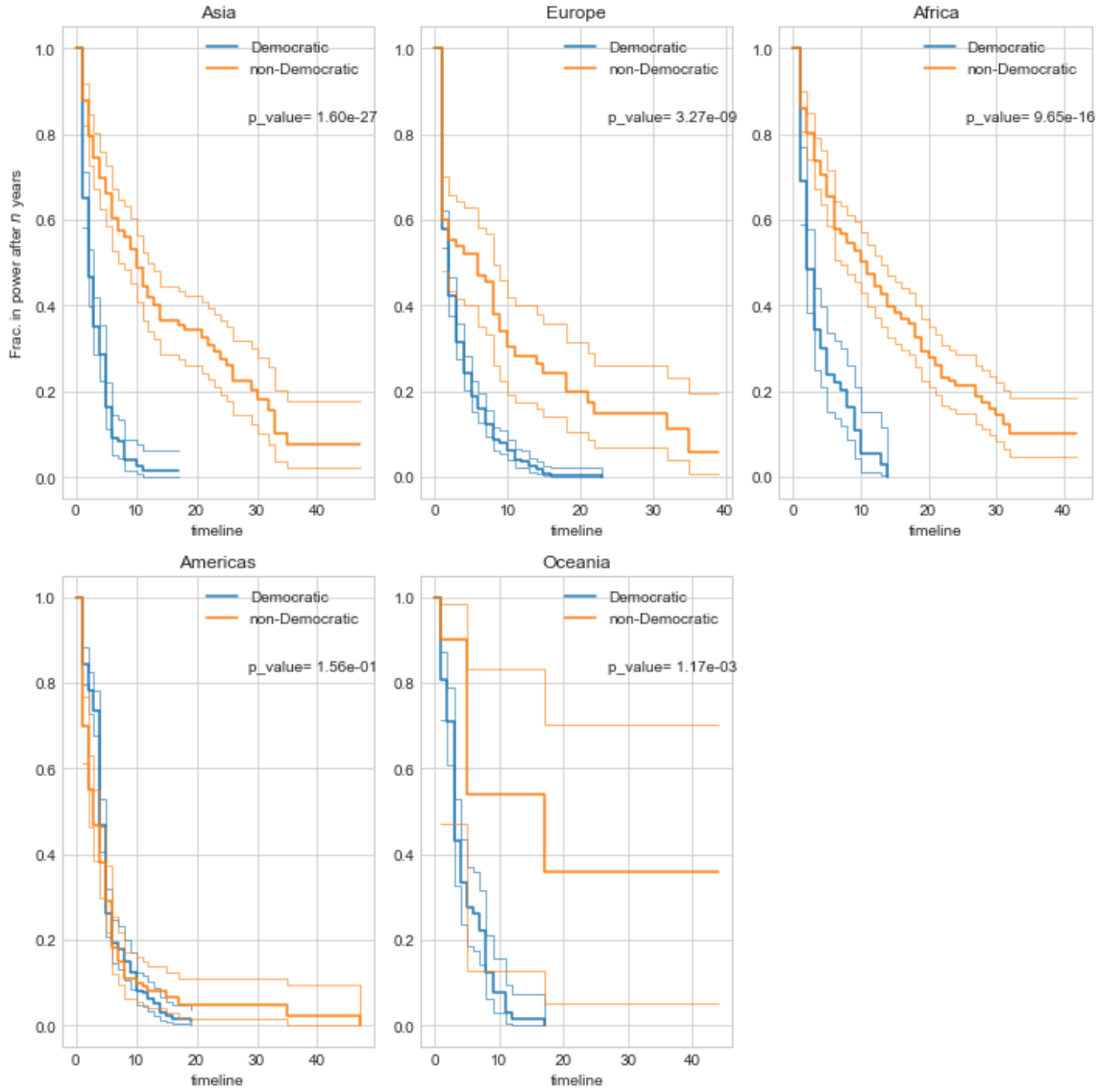
```
# un_continent_name "Asia"のみ抽出
df_tmp = df[df["un_continent_name"] == "Asia" ]
df_tmp.head()
```

Out[14]:

	ctryname	cowcode2	politycode	un_region_name	un_continent_name	ehe
0	Afghanistan	700	700.0	Southern Asia	Asia	Mohamm Zahir Sha
1	Afghanistan	700	700.0	Southern Asia	Asia	Sardar Mohamm Daoud
2	Afghanistan	700	700.0	Southern Asia	Asia	Mohamm Zahir Sha
3	Afghanistan	700	700.0	Southern Asia	Asia	Sardar Mohamm Daoud
4	Afghanistan	700	700.0	Southern Asia	Asia	Nur Mohamm Taraki

In [15]:

```
# un_continent_name ごとに  生存曲線を書く
# それぞれ democratic / non-democratic で分け、logrank_testのp.valueをプリントする
continent_types=df["un_continent_name"].unique()

fig = plt.figure(figsize=(10,10))
for i ,name in enumerate(continent_types):
    ax = fig.add_subplot(2,3,i+1)
    df_tmp = df[df["un_continent_name"] == name]
    time= df_tmp["duration"]
    event= df_tmp["observed"]
    index = df_tmp["democracy"] == "Democracy"

    fit1 = kmf.fit(time[index], event_observed=event[index], label="Democratic")
    fit1.plot(ax=ax, ci_force_lines=True)
    fit2 = kmf.fit(time[~index], event_observed=event[~index], label="non-Democratic")
    fit2.plot(ax=ax, ci_force_lines=True)

    p = logrank_test(time[index], time[~index], event[index], event[~index], alpha=.99).p_value
    ax.text(0.6,0.8, "p_value= {:.2e}".format(p) , transform = ax.transAxes) # p_valueをax上に表示

    ax.set_title(name)# title つける

    if i==0:
        plt.ylabel('Frac. in power after $n$ years') # 1回目だけyラベルいれる

plt.tight_layout() # 重ならないように余白を減らす
```



多変量解析を行う

In [16]:

```
from lifelines import CoxPHFitter
```

In [17]:

```
# データを整形、連続変数あるいは0,1,2..実数に変更
df_curated = df[["un_continent_name","democracy","regime","observed","duration"]]

columns=["un_continent_name","democracy","regime"]
for i ,name in enumerate(columns):
    categories=df_curated[name].unique()
    for j,category in enumerate(categories):
        df_curated[name] = df_curated[name].replace({category:j+1})

df_curated.head()
```

```
/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:8: SettingWithCopyWar
ning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
```

Out[17]:

	un_continent_name	democracy	regime	observed	duration
0	1	1	1	1	7
1	1	1	2	1	10
2	1	1	1	1	10
3	1	1	2	0	5
4	1	1	2	0	1

In [18]:

```
# 多変量解析
cph = CoxPHFitter()
cph.fit(df_curated, duration_col="duration", event_col="observed")
cph.print_summary()
```

```
<lifelines.CoxPHFitter: fitted with 1808 observations, 340 censored>
      duration col = 'duration'
         event col = 'observed'
number of subjects = 1808
  number of events = 1468
    log-likelihood = -9587.09
  time fit was run = 2019-01-30 00:30:07 UTC

---
                    coef  exp(coef)  se(coef)     z      p  log(p)  lower 0.95  upp
er 0.95
un_continent_name -0.06       0.95      0.02 -2.58   0.01   -4.63       -0.10
  -0.01    *
democracy          0.42       1.52      0.10  4.03 <0.005   -9.77        0.22
   0.63  ***
regime             0.24       1.27      0.03  6.94 <0.005  -26.29        0.17
   0.31  ***
---
Signif. codes: 0 '***' 0.0001 '**' 0.001 '*' 0.01 '.' 0.05 ' ' 1

Concordance = 0.63
Likelihood ratio test = 318.39 on 3 df, log(p)=-156.54
```

実際の患者データで行う

In [19]:

```
# 以下のurlからTCGAの大腸癌患者の臨床データをダウンロードし、読み込み
# http://www.cbioportal.org/study?id=coadread_tcga&tab=clinicalData
clinical_data = pd.read_csv("coadread_tcga_clinical_data.tsv",sep="¥t")
```

In [20]:

```
# データ確認
clinical_data.head()
```

Out[20]:

	Study ID	Patient ID	Sample ID	Diagnosis Age	American Joint Committee on Cancer Metastasis Stage Code	Neoplasm Disease Lymph Node Stage American Joint Committee on Cancer Code	Neoplasm Disease Stage American Joint Committee on Cancer Code	A Co or Pul
0	coadread_tcga	TCGA-3L-AA1B	TCGA-3L-AA1B-01	61.0	M0	N0	Stage I	7th
1	coadread_tcga	TCGA-4N-A93T	TCGA-4N-A93T-01	67.0	M0	N1b	Stage IIIB	7th
2	coadread_tcga	TCGA-4T-AA8H	TCGA-4T-AA8H-01	42.0	MX	N0	Stage IIA	7th
3	coadread_tcga	TCGA-5M-AAT4	TCGA-5M-AAT4-01	74.0	M1b	N0	Stage IV	6th
4	coadread_tcga	TCGA-5M-AAT5	TCGA-5M-AAT5-01	NaN	NaN	NaN	NaN	Nal

5 rows × 92 columns

In [21]:

```
# データ抽出、整形
clinical_data.set_index("Patient ID",inplace=True)
sig_columns = ["Diagnosis Age","Neoplasm Disease Stage American Joint Committee on Cancer Code",
               "Neoplasm Disease Lymph Node Stage American Joint Committee on Cancer Code","Sex"
,¥
               "Disease Free (Months)","Disease Free Status","Overall Survival (Months)","Overal
l Survival Status",¥
               "Patient Height","Initial Patient Weight","Primary Tumor Site","Surgical Margin R
esection Status"]
clinical_data = clinical_data[sig_columns]
```

In [22]:

```
clinical_data.rename(columns={"Diagnosis Age":"age",¥
                              "Neoplasm Disease Stage American Joint Committee on Cancer Code":
"stage",¥
                              "Neoplasm Disease Lymph Node Stage American Joint Committee on Can
cer Code":"LN",¥
                              "Disease Free (Months)":"DFS","Disease Free Status":"rec","Overall
 Survival (Months)":"OS",¥
                              "Overall Survival Status":"death","Patient Height":"height","Initi
al Patient Weight":"weight",¥
                              "Primary Tumor Site":"primary","Surgical Margin Resection Status":
"residual"},inplace=True)
```

In [23]:

```
clinical_data["Death"]=clinical_data["death"].replace({"LIVING":0,"DECEASED":1})
```

In [24]:

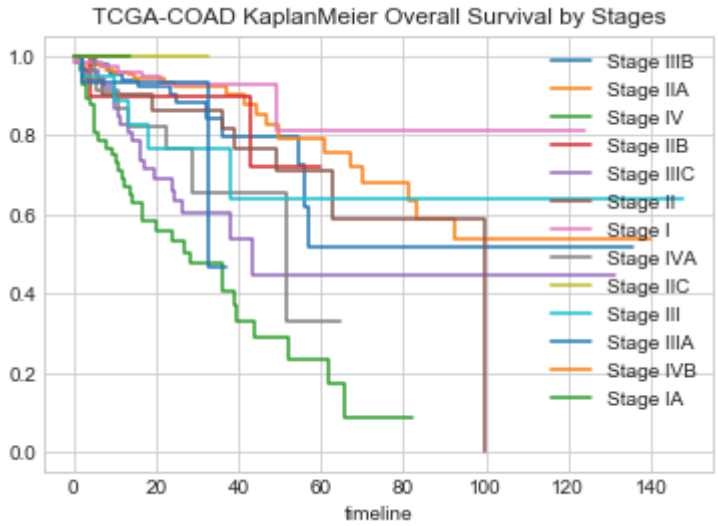
```
clinical_data["Rec"]=clinical_data["rec"].replace({"DiseaseFree":0,"Recurred/Progressed":1})
```

In [25]:

```
# stage で分類し  kaplan meier curve overall survival を書く
# OS, death  がnanだと計算できないのでdropする
df = clinical_data.dropna(subset=["stage","Death","OS"],axis=0,how="any")

fig = plt.figure()
ax = fig.add_subplot(111)

stages = df["stage"].unique()
for i, stage in enumerate(stages):
    index = df["stage"] == stage
    fit= kmf.fit(durations=df[index]["OS"], event_observed=df[index]["Death"],label=stage)
    fit.plot(ax=ax, ci_show=False)

plt.title("TCGA-COAD KaplanMeier Overall Survival by Stages")
```

Out[25]:

```
Text(0.5,1,'TCGA-COAD KaplanMeier Overall Survival by Stages')
```

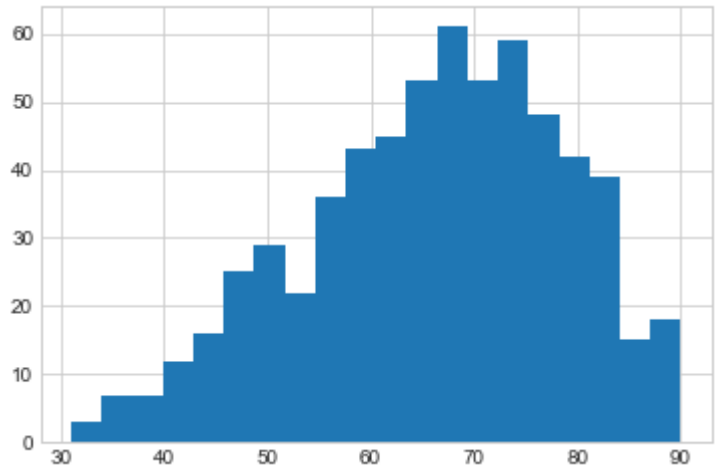


In [26]:

```
# Age はAge_rankにカテゴリー化したい　まずは分布の確認
plt.hist(clinical_data["age"].dropna(),bins=20)
```

Out[26]:

```
(array([ 3.,  7.,  7., 12., 16., 25., 29., 22., 36., 43., 45., 53., 61.,
        53., 59., 48., 42., 39., 15., 18.]),
 array([31.  , 33.95, 36.9 , 39.85, 42.8 , 45.75, 48.7 , 51.65, 54.6 ,
        57.55, 60.5 , 63.45, 66.4 , 69.35, 72.3 , 75.25, 78.2 , 81.15,
        84.1 , 87.05, 90.  ]),
 <a list of 20 Patch objects>)
```



In [27]:

```
# np.digitize を使って値で分類
a=50
b=60
c=70
d=80
bins = np.array([a,b,c,d])
clinical_data["Age_rank"]=np.digitize(clinical_data["age"], bins)
```

In [28]:

```
clinical_data["Age_rank"].astype("category",inplace=True).head()
```

Out[28]:

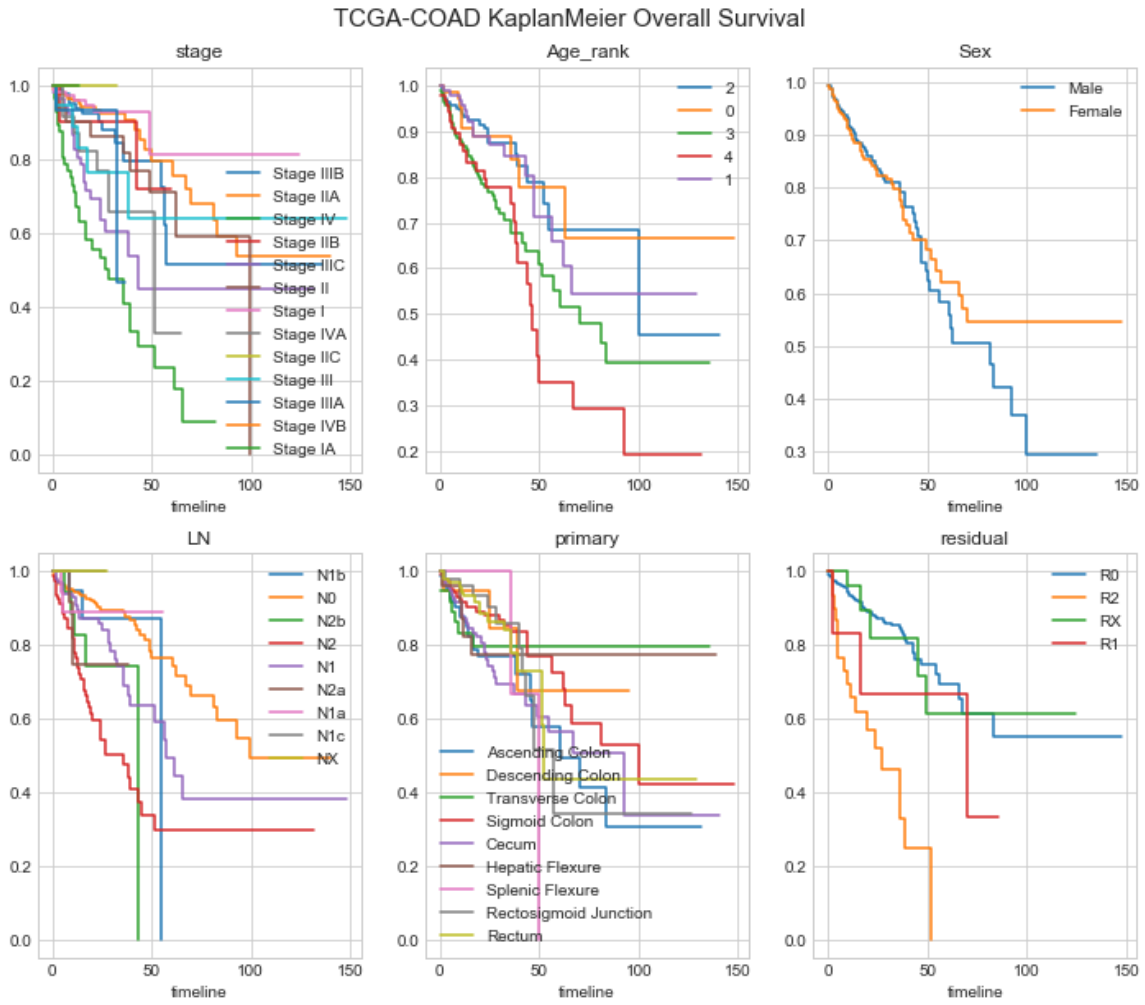
```
Patient ID
TCGA-3L-AA1B    2
TCGA-4N-A93T    2
TCGA-4T-AA8H    0
TCGA-5M-AAT4    3
TCGA-5M-AAT5    4
Name: Age_rank, dtype: category
Categories (5, int64): [0, 1, 2, 3, 4]
```

In [29]:

```
# stageに加えて、 age, Sex, LN, primary, residual でも書く
columns = ["stage","Age_rank","Sex","LN","primary","residual"]

fig = plt.figure(figsize=(12,10))
for i, name in enumerate(columns):
    # drop nan
    df = clinical_data.dropna(subset=[name,"Death","OS"], axis=0, how="any",inplace=False)
    ax = fig.add_subplot(2,3,i+1)
    categories = df[name].unique()
    for j, category in enumerate(categories):
        index = df[name] == category
        fit= kmf.fit(durations=df[index]["OS"], event_observed=df[index]["Death"],label=category
)
        fit.plot(ax=ax, ci_show=False)
        ax.set_title(name)

plt.suptitle("TCGA-COAD KaplanMeier Overall Survival",fontsize=15,y=0.93)
```

Out[29]:

Text(0.5,0.93,'TCGA-COAD KaplanMeier Overall Survival')



In [30]:

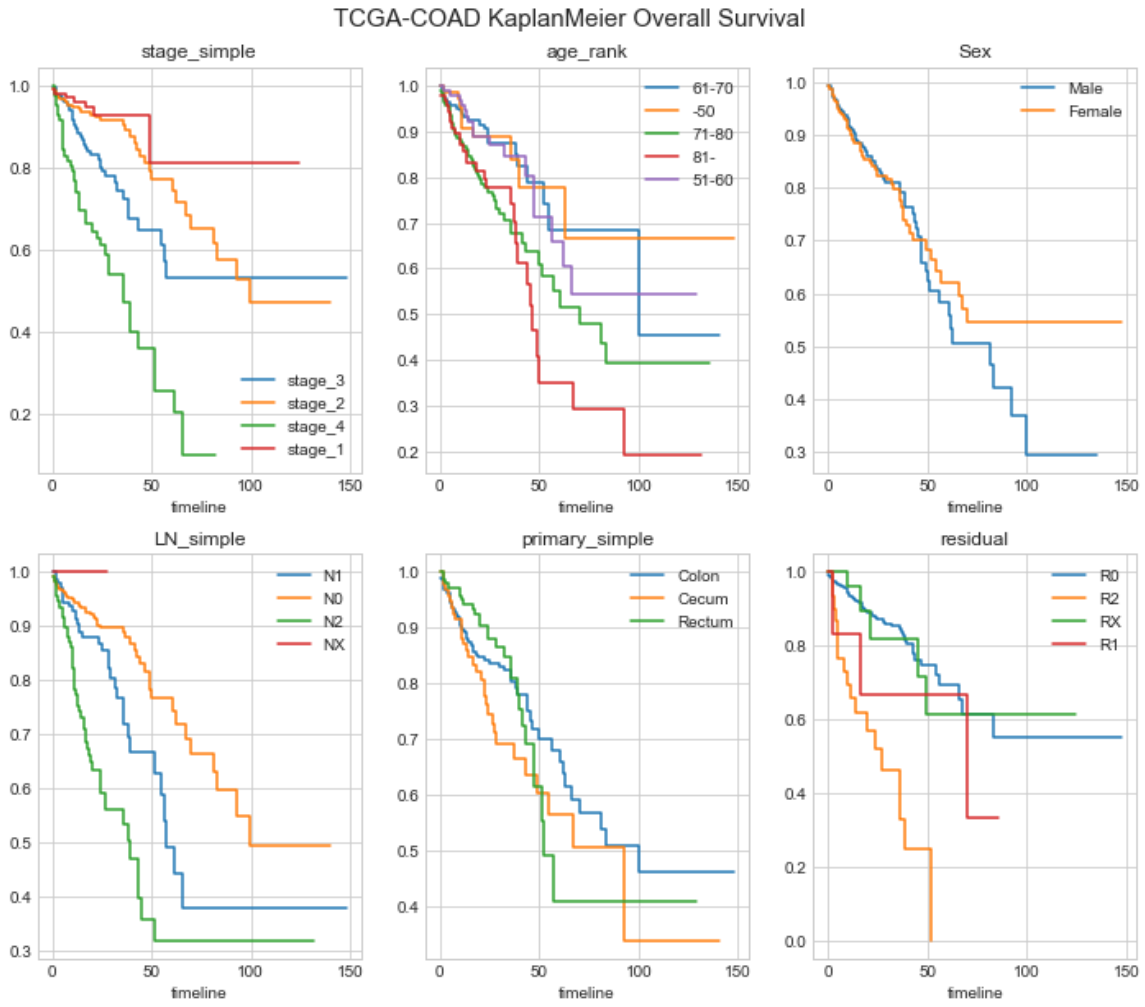
```
# 見にくいのでカテゴリーをもっとシンプルに要約する
clinical_data["stage_simple"] = clinical_data["stage"].replace({'Stage I':"stage_1", 'Stage III
B':"stage_3",¥
                                                               'Stage IIA':"stage_2",'Stage IV'
:"stage_4", 'Stage IIB':"stage_2",¥
                                                               'Stage IIIC':"stage_3", 'Stage I
I':"stage_2",¥
                                                               'Stage IVA':"stage_4", 'Stage II
C':"stage_2", 'Stage III':"stage_3",¥
                                                               'Stage IIIA':"stage_3", 'Stage I
VB':"stage_4", 'Stage IA':"stage_1"})

clinical_data["LN_simple"] = clinical_data["LN"].replace({'N1b':"N1", 'N2b':"N2", 'N2a':"N2", 'N
1a':"N1",'N1c':"N1"})

clinical_data["primary_simple"] = clinical_data["primary"].replace({'Ascending Colon':"Colon",¥
                                                                    'Descending Colon':"Colon",
'Transverse Colon':"Colon",¥
                                                                    'Sigmoid Colon':"Colon", 'He
patic Flexure':"Colon",¥
                                                                    'Splenic Flexure':"Colon",
'Rectosigmoid Junction':"Rectum"})

clinical_data["age_rank"] = clinical_data["Age_rank"].replace({0:"-50",1:"51-60",2:"61-70",3:"71
-80",4:"81-"})
```

In [31]:

```
# 再度書く
columns = ["stage_simple","age_rank","Sex","LN_simple","primary_simple","residual"]

fig = plt.figure(figsize=(12,10))
for i, name in enumerate(columns):
    df = clinical_data.dropna(subset=[name,"Death","OS"], axis=0, how="any",inplace=False)
    ax = fig.add_subplot(2,3,i+1)
    categories = df[name].unique()
    for j, category in enumerate(categories):
        index = df[name] == category
        fit= kmf.fit(durations=df[index]["OS"], event_observed=df[index]["Death"],label=category
)
        fit.plot(ax=ax, ci_show=False)
        ax.set_title(name)

plt.suptitle("TCGA-COAD KaplanMeier Overall Survival",fontsize=15,y=0.93)
```

Out[31]:

Text(0.5,0.93,'TCGA-COAD KaplanMeier Overall Survival')

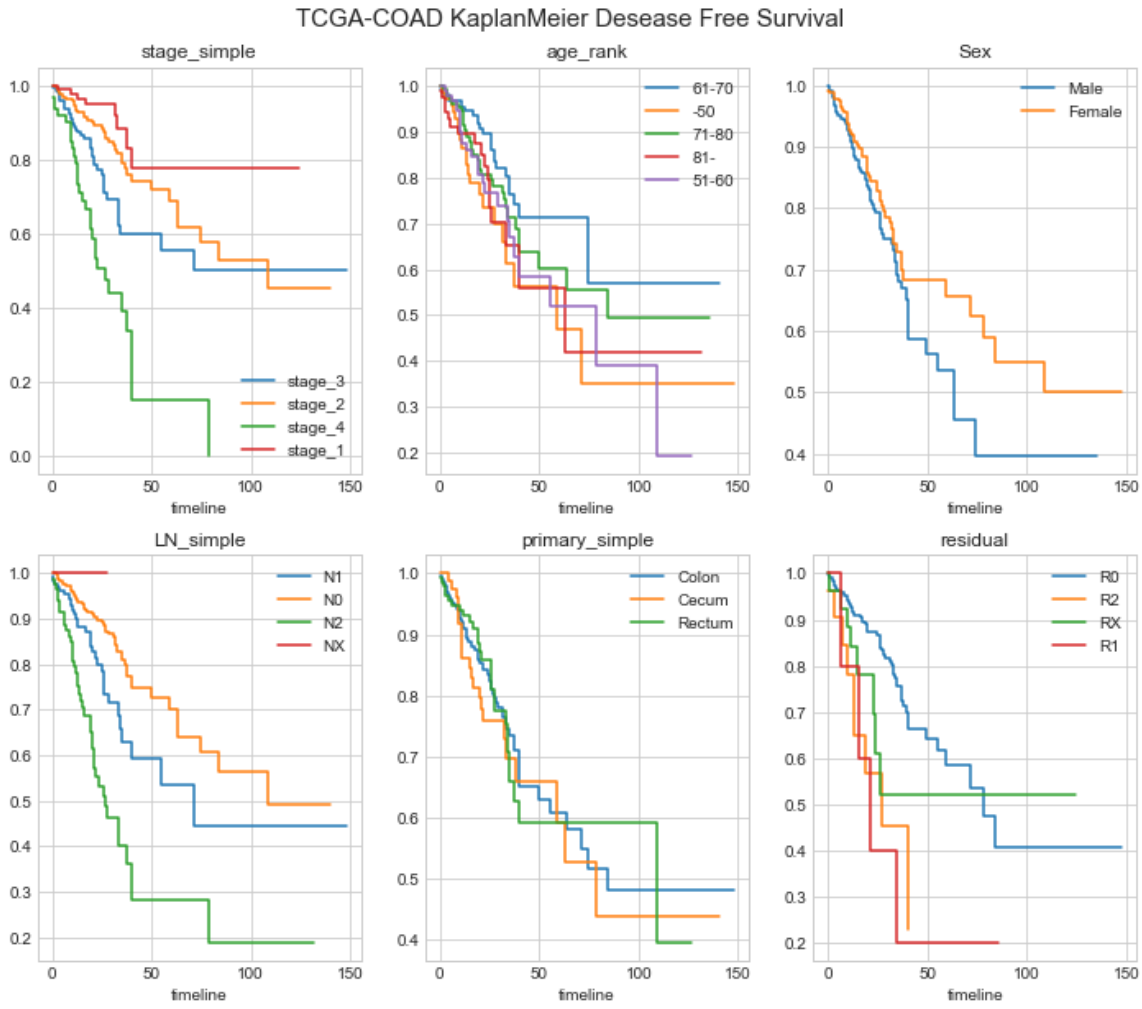


In [32]:

```
# DFSでも書いてみる
columns = ["stage_simple","age_rank","Sex","LN_simple","primary_simple","residual"]

fig = plt.figure(figsize=(12,10))
for i, name in enumerate(columns):
    df = clinical_data.dropna(subset=[name,"Rec","DFS"], axis=0, how="any")
    ax = fig.add_subplot(2,3,i+1)
    categories = df[name].unique()
    for j, category in enumerate(categories):
        index = df[name] == category
        fit= kmf.fit(durations=df[index]["DFS"], event_observed=df[index]["Rec"],label=category)
        fit.plot(ax=ax, ci_show=False)
        ax.set_title(name)

plt.suptitle("TCGA-COAD KaplanMeier Desease Free Survival",fontsize=15,y=0.93)
```

Out[32]:

Text(0.5,0.93,'TCGA-COAD KaplanMeier Desease Free Survival')



In [33]:

```
# 多変量解析のために 数字データに整形
columns = ["stage_simple","age_rank","Sex","LN_simple","primary_simple","residual"]
new_columns = columns +["OS","Death"]
df_selected = clinical_data[new_columns]

for i ,name in enumerate(columns):
    categories=df_selected[name].unique()
    for j,category in enumerate(categories):
        df_selected[name] = df_selected[name].replace({category:j+1})

df_selected.head()
```

```
/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:9: SettingWithCopyWar
ning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  if __name__ == '__main__':
```

Out[33]:

	stage_simple	age_rank	Sex	LN_simple	primary_simple	residual	OS	De
Patient ID								
TCGA-3L-AA1B	1.0	1	1.0	1	1	1	NaN	Na
TCGA-4N-A93T	2.0	1	2.0	2	2	1	4.80	0.0
TCGA-4T-AA8H	3.0	2	1.0	1	2	1	12.65	0.0
TCGA-5M-AAT4	4.0	3	2.0	1	2	1	1.61	1.0
TCGA-5M-AAT5	5.0	4	3.0	3	3	2	NaN	Na

In [34]:

```
# 多変量解析
df_selected.dropna(subset=["OS","Death"],inplace=True)

cph = CoxPHFitter()
cph.fit(df_selected, duration_col="OS", event_col="Death")
cph.print_summary()
```

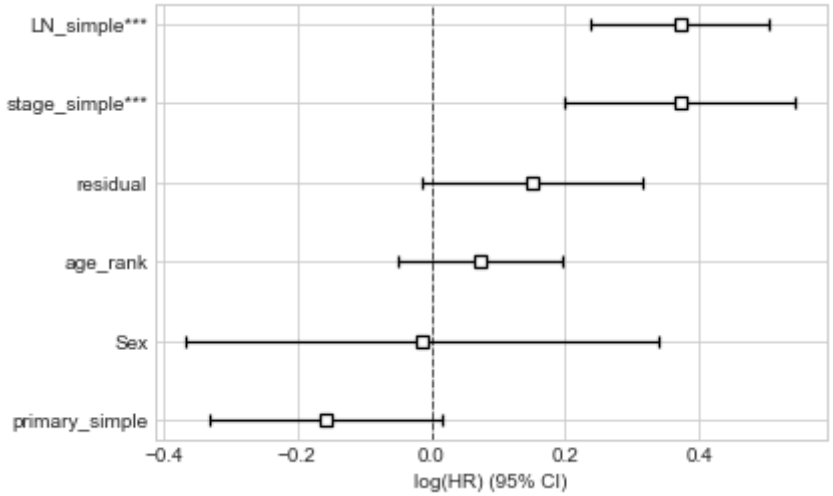
```
<lifelines.CoxPHFitter: fitted with 629 observations, 499 censored>
      duration col = 'OS'
         event col = 'Death'
number of subjects = 629
  number of events = 130
    log-likelihood = -692.65
  time fit was run = 2019-01-30 00:30:20 UTC

---
                coef  exp(coef)  se(coef)     z      p  log(p)  lower 0.95  upper
0.95
stage_simple    0.37       1.45      0.09  4.25 <0.005  -10.73        0.20
0.54  ***
age_rank        0.07       1.08      0.06  1.18   0.24   -1.44       -0.05
0.20
Sex            -0.01       0.99      0.18 -0.08   0.94   -0.06       -0.37
0.34
LN_simple       0.37       1.45      0.07  5.48 <0.005  -16.97        0.24
0.51  ***
primary_simple -0.16       0.85      0.09 -1.78   0.07   -2.59       -0.33
0.02
residual        0.15       1.16      0.08  1.80   0.07   -2.63       -0.01
0.31
---
Signif. codes: 0 '***' 0.0001 '**' 0.001 '*' 0.01 '.' 0.05 ' ' 1

Concordance = 0.72
Likelihood ratio test = 67.99 on 6 df, log(p)=-27.58

/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:2: SettingWithCopyWar
ning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
```

In [35]:

```
#  Plot
cph.plot()
```

Out[35]:

<matplotlib.axes._subplots.AxesSubplot at 0x1a2ca80e80>



In [81]:

```
# check assumptions
cph.check_assumptions(df_selected)
```

Proportional hazard assumption looks okay.