本项目旨在通过Python对小米公司的财务数据进行全面分析，并以可视化的方式呈现各项关键财务指标的趋势和现状。通过对财务报表的深入挖掘，我们能够更好地理解小米公司在资本结构、融资策略、财务风险等方面的特征和变化规律。

环境检查

In [1]:
```
! python --version
! pip list | findstr "numpy pandas scikit-learn"
```

```
Python 3.12.7
numpy                 1.26.4
pandas                2.2.2
scikit-learn          1.4.2
WARNING: Ignoring invalid distribution ~ip (C:\Program Files\python312\Lib\site-packages)
```

In [2]:
```
import matplotlib_inline
import pandas as pd
from IPython.core.interactiveshell import InteractiveShell
```

In [3]:
```
matplotlib_inline.backend_inline.set_matplotlib_formats("svg")
InteractiveShell.ast_node_interactivity = "all"

# 显示所有的列
pd.set_option("display.max_columns", None)
# 显示所有的行
pd.set_option("display.max_rows", None)
```

数据清洗

财务报告

In [4]:
```
df_zc=pd.read_excel("rawdata\财务报告\资产负债表_01810.HK.xls")
df_lr=pd.read_excel("rawdata\财务报告\利润表_01810.HK.xls")
df_xj=pd.read_excel("rawdata\财务报告\现金流量表_01810.HK.xls")
```

In [5]:
```
df_zc.head()
df_lr.head()
df_xj.head()
```

Out[5]:

	Unnamed: 0	2023-12-31	2022-12-31	2021-12-31	2020-12-31	2019-12-31	2018-12-31	2017-12-31	2016-12-31	2015-12-31
0	报告期	年报	年报	年报	年报	年报	年报	年报	年报	年报
1	报表类型	合并报表	合并报表	合并报表	合并报表	合并报表	合并报表	合并报表	合并报表	合并报表
2	报表年结日	12-31	12-31	12-31	12-31	12-31	12-31	12-31	12-31	12-31
3	上市前/上市后	上市后	上市后	上市后	上市后	上市后	上市后	上市前	上市前	上市前
4	公司类型	通用	通用	通用	通用	通用	通用	通用	通用	通用

Out[5]:

	Unnamed: 0	2023-12-31	2022-12-31	2021-12-31	2020-12-31	2019-12-31	2018-12-31	2017-12-31	2016-12-31	2015-12-31
0	报告期	年报	年报	年报	年报	年报	年报	年报	年报	年报
1	报表类型	合并报表	合并报表	合并报表	合并报表	合并报表	合并报表	合并报表	合并报表	合并报表
2	报表年结日	12-31	12-31	12-31	12-31	12-31	12-31	12-31	12-31	12-31
3	上市前/上市后	上市后	上市后	上市后	上市后	上市后	上市后	上市前	上市前	上市前
4	公司类型	通用	通用	通用	通用	通用	通用	通用	通用	通用

Out[5]:

	Unnamed: 0	2023-12-31	2022-12-31	2021-12-31	2020-12-31	2019-12-31	2018-12-31	2017-12-31	2016-12-31	2015-12-31
0	报告期	年报	年报	年报	年报	年报	年报	年报	年报	年报
1	报表类型	合并报表	合并报表	合并报表	合并报表	合并报表	合并报表	合并报表	合并报表	合并报表
2	报表年结日	12-31	12-31	12-31	12-31	12-31	12-31	12-31	12-31	12-31
3	上市前/上市后	上市后	上市后	上市后	上市后	上市后	上市后	上市前	上市前	上市前
4	公司类型	通用	通用	通用	通用	通用	通用	通用	通用	通用

宽数据转换为长数据

In [6]:
```
df_zc_wide=df_zc.iloc[5:]
df_zc_wide=df_zc_wide.reset_index(drop=True)

df_lr_wide=df_lr.iloc[5:]
df_lr_wide=df_lr_wide.reset_index(drop=True)

df_xj_wide=df_xj.iloc[5:]
df_xj_wide=df_xj_wide.reset_index(drop=True)
```

In [7]:
```
df_zc_wide.head()
df_lr_wide.head()
df_xj_wide.head()
```

Out[7]:

	Unnamed: 0	2023-12-31	2022-12-31	2021-12-31	2020-12-31	2019-12-31	20
0	非流动资产：	NaN	NaN	NaN	NaN	NaN	
1	固定资产(元)	NaN	NaN	NaN	NaN	NaN	
2	物业厂房及设备(元)	13720825000	9138221000	6964621000	6305657000	6992331000	50
3	投资物业(元)	NaN	NaN	NaN	NaN	NaN	
4	无形资产(元)	8628739000	4629676000	5579159000	4265619000	1672002000	20

Out[7]:

	Unnamed: 0	2023-12-31	2022-12-31	2021-12-31	2020-12-31	2019-
0	营业额(元)	270970141000	280044016000	328309145000	245865633000	205838
1	其他营业收入(元)	NaN	NaN	NaN	NaN	
2	营运收入其他项目(元)	NaN	NaN	NaN	NaN	
3	营运收入平衡项目(元)	NaN	NaN	NaN	NaN	
4	营运收入(元)	270970141000	280044016000	328309145000	245865633000	205838

Out[7]:

	Unnamed: 0	2023-12-31	2022-12-31	2021-12-31	2020-12-31	2019-12-31
0	经营活动产生的现金流量：	NaN	NaN	NaN	NaN	NaN
1	除税前溢利(业务利润)(元)	22011047000	3933956000	24417033000	21633432000	12162646000
2	减:利息收入(元)	3558347000	1663941000	1229826000	963555000	930889000
3	加:利息支出(元)	1555970000	546483000	2841457000	3364852000	528460000
4	减:投资收益(元)	157569000	-147491000	2634379000	1742322000	48468000

资产负债表

In [8]:
```
dates = df_zc_wide.columns[1:].tolist()
accounting_items = df_zc_wide.iloc[1:, 0].tolist()
data = df_zc_wide.iloc[1:, 1:].values

df_zc_result = pd.DataFrame(data, columns=dates, index=accounting_items)

df_zc_result = df_zc_result.T
df_zc_result.reset_index(inplace=True)
df_zc_result.rename(columns={'index': '日期'}, inplace=True)
```

In [9]:
```
df_zc_result.columns = df_zc_result.columns.str.strip()

column_names_to_keep = ['日期', '总资产(元)', '总负债(元)', '股东权益合计(元)', '

df_ZC = df_zc_result.loc[:, column_names_to_keep]

df_ZC["年份"]=df_ZC["日期"].str.slice(0,4).astype("int64")

df_ZC.head()
```

Out[9]:

	日期	总资产(元)	总负债(元)	股东权益合计(元)	流动资产合计(元)	流动负债合计(元
0	2023-12-31	324247439000	159985671000	164261768000	199052700000	11558759600
1	2022-12-31	273507211000	129584151000	143923060000	160414795000	8962753300
2	2021-12-31	292891870000	155459374000	137432496000	185851401000	11572747100
3	2020-12-31	253679823000	129666308000	124013515000	176282835000	10792692800
4	2019-12-31	183629207000	101971531000	81657676000	137539086000	9218070500

In [10]:
```
df_ZC.dtypes
```

Out[10]:
```
日期            object
总资产(元)        object
总负债(元)        object
股东权益合计(元)     object
流动资产合计(元)     object
流动负债合计(元)     object
短期贷款(元)       object
长期贷款(元)       object
年份             int64
dtype: object
```

In [11]:
```
columns_to_convert = ['总资产(元)', '总负债(元)', '股东权益合计(元)', '流动资产合

for col in columns_to_convert:
    df_ZC[col] = pd.to_numeric(df_ZC[col], errors='coerce')

df_ZC.dtypes
```

Out[11]:
```
日期             object
总资产(元)         int64
总负债(元)         int64
股东权益合计(元)       int64
流动资产合计(元)       int64
流动负债合计(元)       int64
短期贷款(元)       float64
长期贷款(元)         int64
年份              int64
dtype: object
```

利润表

In [12]:
```
dates = df_lr_wide.columns[1:].tolist()


accounting_items = df_lr_wide.iloc[1:, 0].tolist()


data = df_lr_wide.iloc[1:, 1:].values


df_lr_result = pd.DataFrame(data, columns=dates, index=accounting_items)


df_lr_result = df_lr_result.T
df_lr_result.reset_index(inplace=True)
df_lr_result.rename(columns={'index': '日期'}, inplace=True)
```

In [13]:
```
df_lr_result.columns = df_lr_result.columns.str.strip()

column_names_to_keep = ['日期', '营运收入(元)', '毛利(元)', '利息收入(元)', '融资

df_LR = df_lr_result.loc[:, column_names_to_keep]

df_LR["年份"]=df_LR["日期"].str.slice(0,4).astype("int64")

df_LR.head()
```

Out[13]:

	日期	营运收入(元)	毛利(元)	利息收入(元)	融资成本(元)	每股基本盈利(元)	年份
0	2023-12-31	270970141000	57476239000	3558347000	1555970000	0.7	2023
1	2022-12-31	280044016000	47577190000	1663941000	546483000	0.1	2022
2	2021-12-31	328309145000	58260941000	1229826000	2841457000	0.78	2021
3	2020-12-31	245865633000	36751862000	963555000	3364852000	0.85	2020
4	2019-12-31	205838682000	28554033000	402429000	NaN	0.42	2019

In [14]:
```
df_LR.dtypes
```

Out[14]:
```
日期           object
营运收入(元)      object
毛利(元)         object
利息收入(元)      object
融资成本(元)      object
每股基本盈利(元)    object
年份            int64
dtype: object
```

In [15]:
```
columns_to_convert = ['营运收入(元)', '毛利(元)', '利息收入(元)', '融资成本(元)',

for col in columns_to_convert:
    df_LR[col] = pd.to_numeric(df_LR[col], errors='coerce')

df_LR.dtypes
```

Out[15]:
```
日期            object
营运收入(元)        int64
毛利(元)          int64
利息收入(元)       float64
融资成本(元)       float64
每股基本盈利(元)     float64
年份             int64
dtype: object
```

现金流量表

In [16]:
```
dates = df_xj_wide.columns[1:].tolist()


accounting_items = df_xj_wide.iloc[1:, 0].tolist()

data = df_xj_wide.iloc[1:, 1:].values


df_xj_result = pd.DataFrame(data, columns=dates, index=accounting_items)


df_xj_result = df_xj_result.T
df_xj_result.reset_index(inplace=True)
df_xj_result.rename(columns={'index': '日期'}, inplace=True)
```

In [17]:
```
df_xj_result.columns = df_xj_result.columns.str.strip()

column_names_to_keep = ['日期', '经营业务现金净额(元)', '投资业务现金净额(元)', '

df_XJ = df_xj_result.loc[:, column_names_to_keep]

df_XJ["年份"]=df_XJ["日期"].str.slice(0,4).astype("int64")

df_XJ.head()
```

Out[17]:

	日期	经营业务现金净额(元)	投资业务现金净额(元)	融资业务现金净额(元)	期末现金(元)	年份
0	2023-12-31	41300495000	-35169054000	-504972000	33631313000	2023
1	2022-12-31	-4389730000	15548773000	-7854799000	27607261000	2022
2	2021-12-31	9785288000	-45007945000	4498686000	23511579000	2021
3	2020-12-31	21878500000	-17678852000	26215568000	54752443000	2020
4	2019-12-31	23810354000	-31570136000	3121238000	25919861000	2019

In [18]:
```
df_XJ.dtypes
```

Out[18]:
```
日期              object
经营业务现金净额(元)    object
投资业务现金净额(元)    object
融资业务现金净额(元)    object
期末现金(元)         object
年份               int64
dtype: object
```

In [19]:
```
columns_to_convert = ['经营业务现金净额(元)', '投资业务现金净额(元)', '融资业务现

for col in columns_to_convert:
    df_XJ[col] = pd.to_numeric(df_XJ[col], errors='coerce')

df_XJ.dtypes
```

Out[19]:
```
日期              object
经营业务现金净额(元)     int64
投资业务现金净额(元)     int64
融资业务现金净额(元)     int64
期末现金(元)          int64
年份               int64
dtype: object
```

合并财务报告

In [20]:
```
df_fs=pd.merge(df_ZC,df_LR,how="inner",on=["年份"])
df_fs=pd.merge(df_fs,df_XJ,how="inner",on=["年份"])
```

In [21]:
```
df_fs.columns
```

Out[21]:
```
Index(['日期_x', '总资产(元)', '总负债(元)', '股东权益合计(元)', '流动资产合计
(元)', '流动负债合计(元)',
       '短期贷款(元)', '长期贷款(元)', '年份', '日期_y', '营运收入(元)', '毛利
(元)', '利息收入(元)',
       '融资成本(元)', '每股基本盈利(元)', '日期', '经营业务现金净额(元)', '投资
业务现金净额(元)',
       '融资业务现金净额(元)', '期末现金(元)'],
      dtype='object')
```

In [22]:
```
df_fs.drop(['日期_x','日期_y','日期'],axis=1,inplace=True)
df_fs.columns
```

Out[22]:
```
Index(['总资产(元)', '总负债(元)', '股东权益合计(元)', '流动资产合计(元)', '流动负债合计(元)', '短期贷款(元)',
       '长期贷款(元)', '年份', '营运收入(元)', '毛利(元)', '利息收入(元)', '融资成本(元)', '每股基本盈利(元)',
       '经营业务现金净额(元)', '投资业务现金净额(元)', '融资业务现金净额(元)', '期末现金(元)'],
      dtype='object')
```

财务指标

In [23]:
```
df_cz=pd.read_excel("rawdata\财务指标\成长能力_01810.HK.xls")
df_mg=pd.read_excel("rawdata\财务指标\每股指标_01810.HK.xls")
df_ys=pd.read_excel("rawdata\财务指标\盈利能力与收益质量_01810.HK.xls")
df_zb=pd.read_excel("rawdata\财务指标\资本结构与偿债能力_01810.HK.xls")
```

In [24]:
```
df_cz.head()
df_mg.head()
df_ys.head()
df_zb.head()
```

Out[24]:

	Unnamed: 0	2023年年报	2022年年报	2021年年报	2020年年报	2019年年报	2018年年报	2017年年报	2016年年报	201年年报
0	总资产同比增长率(%)	18.55	-6.62	15.46	38.15	26.44	61.60	77.03	29.71	NaN
1	每股净资产同比增长率(%)	13.60	4.87	11.80	45.56	11.76	121.78	-30.04	-13.58	NaN
2	总负债同比增长率(%)	23.46	-16.64	19.89	27.16	37.84	-65.92	51.99	13.55	NaN
3	税前利润同比增长率(%)	459.51	-83.89	12.87	77.87	-12.67	133.30	-3658.40	115.73	NaN
4	基本每股收益同比增长率(%)	600.00	-87.18	-8.24	100.71	-49.82	118.77	-78892.98	172.92	NaN

Out[24]:

	Unnamed: 0	2023年年报	2022年年报	2021年年报	2020年年报	2019年年报	2018年年报	2017年年报	2016年年报	201年
0	每股收益EPS(基本)(元)	0.7	0.1	0.78	0.849	0.423	0.843	-44.912	0.571	-7.8
1	每股收益EPS(稀释)(元)	0.69	0.1	0.76	0.825	0.41	0.044	-44.912	0.57	-7.8
2	每股收益EPS(TTM)(元)	0.697	0.0992	0.7738	0.8082	0.4166	0.5737	-4.7728	0.064	-0.82
3	每股收益EPS(调整股本数)(元)	0.697	0.0992	0.7738	0.8082	0.4166	0.5737	NaN	NaN	N
4	每股收益EPS(最新股本摊薄)(元)	0.7012	0.0993	0.776	0.8168	0.403	0.5439	-1.7586	0.0222	-0.30

Out[24]:

	Unnamed: 0	2023年年报	2022年年报	2021年年报	2020年年报	2019年年报	2018年年报	2017年年报	2016年年报	2015年年报
0	盈利能力	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	销售毛利率(%)	21.21	16.99	17.75	14.95	13.87	12.69	13.22	10.59	4.04
2	销售净利率(%)	6.45	0.89	5.87	8.26	4.91	7.71	-38.29	0.72	-11.42
3	净资产收益率(平均)(%)	11.36	1.76	14.82	19.86	13.16	-48.45	39.94	-0.62	NaN
4	净资产收益率(年化)(%)	11.36	1.76	14.82	19.86	13.16	-48.45	39.94	-0.62	NaN

Out[24]:

	Unnamed: 0	2023年年报	2022年年报	2021年年报	2020年年报	2019年年报	2018年年报	2017年年报	2016年年报	2015年年报
0	资本结构	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	资产负债率(%)	49.34	47.38	53.08	51.11	55.53	50.94	241.55	281.34	321.37
2	权益乘数	1.97	1.90	2.13	2.05	2.25	2.04	-0.71	-0.55	-0.45
3	产权比率	0.98	0.90	1.13	1.05	1.25	1.04	-1.71	-1.55	-1.45
4	流动资产/总资产(%)	61.39	58.65	63.45	69.49	74.90	73.00	68.03	60.35	63.76

成长能力

In [25]:
```
dates = df_cz.columns[1:].tolist()
accounting_items = df_cz.iloc[1:, 0].tolist()
```

```
data = df_cz.iloc[1:, 1:].values


df_cz_result = pd.DataFrame(data, columns=dates, index=accounting_items)


df_cz_result = df_cz_result.T
df_cz_result.reset_index(inplace=True)
df_cz_result.rename(columns={'index': '日期'}, inplace=True)
```

In [26]:
```
df_cz_result.columns = df_cz_result.columns.str.strip()

column_names_to_keep = ['日期', '每股净资产同比增长率(%)', '总负债同比增长率(%)',
                        '营业收入同比增长率(%)', '基本每股收益同比增长率(%)', '股

df_CZ = df_cz_result.loc[:, column_names_to_keep]

df_CZ["年份"]=df_CZ["日期"].str.slice(0,4).astype("int64")

df_CZ.head()
```

Out[26]:

	日期	每股净资产同比增长率(%)	总负债同比增长率(%)	营业收入同比增长率(%)	基本每股收益同比增长率(%)	股东权益合计同比增长率(%)	年份
0	2023年年报	13.60	23.46	-3.24	600.00	14.13	2023
1	2022年年报	4.87	-16.64	-14.70	-87.18	4.72	2022
2	2021年年报	11.80	19.89	33.53	-8.24	10.82	2021
3	2020年年报	45.56	27.16	19.45	100.71	51.87	2020
4	2019年年报	11.76	37.84	17.68	-49.82	14.61	2019

In [27]:
```
df_CZ.dtypes
```

Out[27]:
```
日期                  object
每股净资产同比增长率(%)     float64
总负债同比增长率(%)        float64
营业收入同比增长率(%)       float64
基本每股收益同比增长率(%)     float64
股东权益合计同比增长率(%)     float64
年份                   int64
dtype: object
```

每股指标

In [28]:
```
dates = df_mg.columns[1:].tolist()
accounting_items = df_mg.iloc[1:, 0].tolist()
data = df_mg.iloc[1:, 1:].values


df_mg_result = pd.DataFrame(data, columns=dates, index=accounting_items)
```

```
df_mg_result = df_mg_result.T
df_mg_result.reset_index(inplace=True)
df_mg_result.rename(columns={'index': '日期'}, inplace=True)
```

In [29]:
```
df_mg_result.columns = df_mg_result.columns.str.strip()

column_names_to_keep = ['日期', '每股收益EPS(TTM)(元)', '每股净资产BPS(元)',
                        '每股经营现金净流量(元)', '每股营业收入(元)']

df_MG = df_mg_result.loc[:, column_names_to_keep]

df_MG["年份"]=df_MG["日期"].str.slice(0,4).astype("int64")

df_MG.head()
```

Out[29]:

	日期	每股收益EPS(TTM)(元)	每股净资产BPS(元)	每股经营现金净流量(元)	每股营业收入(元)	年份
0	2023年年报	0.697	6.5406	1.6472	10.8071	2023
1	2022年年报	0.0992	5.7575	-0.1759	11.2236	2022
2	2021年年报	0.7738	5.4902	0.3915	13.1363	2021
3	2020年年报	0.8082	4.9109	0.8686	9.7615	2020
4	2019年年报	0.4166	3.3737	0.9877	8.5384	2019

In [30]:
```
df_MG.dtypes
```

Out[30]:
```
日期                  object
每股收益EPS(TTM)(元)    object
每股净资产BPS(元)        object
每股经营现金净流量(元)       object
每股营业收入(元)          object
年份                  int64
dtype: object
```

In [31]:
```
columns_to_convert = ['每股收益EPS(TTM)(元)', '每股净资产BPS(元)',
                      '每股经营现金净流量(元)', '每股营业收入(元)']

for col in columns_to_convert:
    df_MG[col] = pd.to_numeric(df_MG[col], errors='coerce')

df_MG.dtypes
```

Out[31]:
```
日期                  object
每股收益EPS(TTM)(元)    float64
每股净资产BPS(元)        float64
每股经营现金净流量(元)       float64
每股营业收入(元)          float64
年份                  int64
dtype: object
```

盈利能力与收益质量

In [32]:
```
dates = df_ys.columns[1:].tolist()
accounting_items = df_ys.iloc[1:, 0].tolist()
data = df_ys.iloc[1:, 1:].values

df_ys_result = pd.DataFrame(data, columns=dates, index=accounting_items)


df_ys_result = df_ys_result.T
df_ys_result.reset_index(inplace=True)
df_ys_result.rename(columns={'index': '日期'}, inplace=True)
```

In [33]:
```
df_ys_result.columns = df_ys_result.columns.str.strip()

column_names_to_keep = ['日期', '销售净利率(%)', '总资产净利率(%)',
                        '净资产收益率(TTM)(%)', '投入资本回报率(TTM)(%)']

df_YS = df_ys_result.loc[:, column_names_to_keep]

df_YS["年份"]=df_YS["日期"].str.slice(0,4).astype("int64")

df_YS.head()
```

Out[33]:

	日期	销售净利率(%)	总资产净利率(%)	净资产收益率(TTM)(%)	投入资本回报率(TTM)(%)	年份
0	2023年年报	6.45	5.85	10.66	7.01	2023
1	2022年年报	0.89	0.87	1.72	0.80	2022
2	2021年年报	5.87	7.08	14.09	12.25	2021
3	2020年年报	8.26	9.31	16.46	8.83	2020
4	2019年年报	4.91	6.11	12.35	8.82	2019

In [34]:
```
df_YS.dtypes
```

Out[34]:
```
日期                  object
销售净利率(%)          float64
总资产净利率(%)         float64
净资产收益率(TTM)(%)     float64
投入资本回报率(TTM)(%)    float64
年份                  int64
dtype: object
```

资本结构与偿债能力

In [35]:
```
dates = df_zb.columns[1:].tolist()
accounting_items = df_zb.iloc[1:, 0].tolist()
data = df_zb.iloc[1:, 1:].values
```

```
df_zb_result = pd.DataFrame(data, columns=dates, index=accounting_items)


df_zb_result = df_zb_result.T
df_zb_result.reset_index(inplace=True)
df_zb_result.rename(columns={'index': '日期'}, inplace=True)
```

In [36]:
```
df_zb_result.columns = df_zb_result.columns.str.strip()

column_names_to_keep = ['日期', '资产负债率(%)', '流动负债/负债合计(%)','流动比率
                        '速动比率','产权比率','非流动负债/负债合计(%)', '营业利润

df_ZB = df_zb_result.loc[:, column_names_to_keep]

df_ZB["年份"]=df_ZB["日期"].str.slice(0,4).astype("int64")

df_ZB.head()
```

Out[36]:

	日期	资产负债率(%)	流动负债/负债合计(%)	流动比率	速动比率	产权比率	非流动负债/负债合计(%)	营业利润/负债合计(%)	营业利润/流动负债(%)	年份
0	2023年年报	49.34	72.25	1.72	1.34	0.98	27.75	12.51	17.31	2023
1	2022年年报	47.38	69.17	1.79	1.23	0.90	30.83	2.17	3.14	2022
2	2021年年报	53.08	74.44	1.61	1.15	1.13	25.56	16.74	22.49	2021
3	2020年年报	51.11	83.23	1.63	1.25	1.05	16.77	18.54	22.27	2020
4	2019年年报	55.53	90.40	1.49	1.14	1.25	9.60	11.53	12.76	2019

In [37]:
```
df_ZB.dtypes
```

Out[37]:
```
日期                 object
资产负债率(%)          float64
流动负债/负债合计(%)      float64
流动比率               float64
速动比率               float64
产权比率               float64
非流动负债/负债合计(%)     float64
营业利润/负债合计(%)       float64
营业利润/流动负债(%)       float64
年份                   int64
dtype: object
```

财务分析

In [38]:
```
df_cwfx=pd.read_excel("rawdata\财务指标\财务分析摘要_01810.HK.xls")

df_cwfx.head()
```

Out[38]:

	Unnamed: 0	2023年年报	2022年年报	2021年年报	2020年年报	2019年年报	2018年年报	2017年年报	2016年年报
0	原始币种	人民币	人民币	人民币	人民币	人民币	人民币	人民币	人民币
1	每股指标	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	每股收益EPS(基本)(元)	0.7	0.1	0.78	0.849	0.423	0.843	-44.912	0.571
3	每股净资产(元)	6.5406	5.7575	5.4902	4.9109	3.3737	3.0188	-13.8604	-10.6589
4	每股经营现金净流量(元)	1.6472	-0.1759	0.3915	0.8686	0.9877	-0.0599	-0.1084	0.5239

In [39]:
```
dates = df_cwfx.columns[1:].tolist()
accounting_items = df_cwfx.iloc[1:, 0].tolist()
data = df_cwfx.iloc[1:, 1:].values

df_cwfx_result = pd.DataFrame(data, columns=dates, index=accounting_items)


df_cwfx_result = df_cwfx_result.T
df_cwfx_result.reset_index(inplace=True)
df_cwfx_result.rename(columns={'index': '日期'}, inplace=True)
```

In [40]:
```
df_cwfx_result.columns = df_cwfx_result.columns.str.strip()

column_names_to_keep = ['日期', '存货周转率(次)','总资产同比增长率(%)',
                        '总负债同比增长率(%)','基本每股收益同比增长率(%)','固定资

df_CWFX = df_cwfx_result.loc[:, column_names_to_keep]

df_CWFX["年份"]=df_CWFX["日期"].str.slice(0,4).astype("int64")

df_CWFX.head()
```

Out[40]:

	日期	存货周转率(次)	总资产同比增长率(%)	总负债同比增长率(%)	基本每股收益同比增长率(%)	固定资产周转率(次)	股东权益合计同比增长率(%)	年份
0	2023年年报	4.5	18.55	23.46	600	23.71	14.13	2023
1	2022年年报	4.52	-6.62	-16.64	-87.18	34.78	4.72	2022
2	2021年年报	5.74	15.46	19.89	-8.24	49.48	10.82	2021
3	2020年年报	5.63	38.15	27.16	100.71	36.98	51.87	2020
4	2019年年报	5.71	26.44	37.84	-49.82	34.13	14.61	2019

In [41]:
```
df_CWFX.dtypes
```

Out[41]:
```
日期                  object
存货周转率(次)           object
总资产同比增长率(%)        object
总负债同比增长率(%)        object
基本每股收益同比增长率(%)     object
固定资产周转率(次)         object
股东权益合计同比增长率(%)     object
年份                   int64
dtype: object
```

In [42]:
```
columns_to_convert = ['存货周转率(次)','总资产同比增长率(%)','固定资产周转率(次)'

for col in columns_to_convert:
    df_CWFX[col] = pd.to_numeric(df_CWFX[col], errors='coerce')

df_CWFX.dtypes
```

Out[42]:
```
日期                  object
存货周转率(次)           float64
总资产同比增长率(%)        float64
总负债同比增长率(%)         object
基本每股收益同比增长率(%)      object
固定资产周转率(次)         float64
股东权益合计同比增长率(%)      object
年份                   int64
dtype: object
```

合并财务指标

In [43]:
```
df_fi=pd.merge(df_CZ,df_MG,how="inner",on=["年份"])
df_fi=pd.merge(df_fi,df_YS,how="inner",on=["年份"])
```

In [44]:
```
df_fi.columns
```

Out[44]:
```
Index(['日期_x', '每股净资产同比增长率(%)', '总负债同比增长率(%)', '营业收入同比增长率(%)',
       '基本每股收益同比增长率(%)', '股东权益合计同比增长率(%)', '年份', '日期_y', '每股收益EPS(TTM)(元)',
       '每股净资产BPS(元)', '每股经营现金净流量(元)', '每股营业收入(元)', '日期', '销售净利率(%)',
       '总资产净利率(%)', '净资产收益率(TTM)(%)', '投入资本回报率(TTM)(%)'],
      dtype='object')
```

In [45]:
```
df_fi.drop(['日期_x','日期_y','日期'],axis=1,inplace=True)
df_fi.columns
```

Out[45]:
```
Index(['每股净资产同比增长率(%)', '总负债同比增长率(%)', '营业收入同比增长率(%)', '基本每股收益同比增长率(%)',
       '股东权益合计同比增长率(%)', '年份', '每股收益EPS(TTM)(元)', '每股净资产BPS(元)',
       '每股经营现金净流量(元)', '每股营业收入(元)', '销售净利率(%)', '总资产净利率(%)', '净资产收益率(TTM)(%)',
       '投入资本回报率(TTM)(%)'],
      dtype='object')
```

In [46]:
```
df_fi=pd.merge(df_fi,df_ZB,how="inner",on=["年份"])
df_fi=pd.merge(df_fi,df_CWFX,how="inner",on=["年份"])
df_fi.columns
```

Out[46]:
```
Index(['每股净资产同比增长率(%)', '总负债同比增长率(%)_x', '营业收入同比增长率(%)', '基本每股收益同比增长率(%)_x',
       '股东权益合计同比增长率(%)_x', '年份', '每股收益EPS(TTM)(元)', '每股净资产BPS(元)',
       '每股经营现金净流量(元)', '每股营业收入(元)', '销售净利率(%)', '总资产净利率(%)', '净资产收益率(TTM)(%)',
       '投入资本回报率(TTM)(%)', '日期_x', '资产负债率(%)', '流动负债/负债合计(%)', '流动比率', '速动比率',
       '产权比率', '非流动负债/负债合计(%)', '营业利润/负债合计(%)', '营业利润/流动负债(%)', '日期_y',
       '存货周转率(次)', '总资产同比增长率(%)', '总负债同比增长率(%)_y', '基本每股收益同比增长率(%)_y',
       '固定资产周转率(次)', '股东权益合计同比增长率(%)_y'],
      dtype='object')
```

In [47]:
```
df_fi.drop(['日期_x','日期_y','总负债同比增长率(%)_y','基本每股收益同比增长率(%)_
df_fi.columns
```

Out[47]:
```
Index(['每股净资产同比增长率(%)', '总负债同比增长率(%)_x', '营业收入同比增长率(%)', '基本每股收益同比增长率(%)_x',
       '股东权益合计同比增长率(%)_x', '年份', '每股收益EPS(TTM)(元)', '每股净资产BPS(元)',
       '每股经营现金净流量(元)', '每股营业收入(元)', '销售净利率(%)', '总资产净利率(%)', '净资产收益率(TTM)(%)',
       '投入资本回报率(TTM)(%)', '资产负债率(%)', '流动负债/负债合计(%)', '流动比率', '速动比率', '产权比率',
       '非流动负债/负债合计(%)', '营业利润/负债合计(%)', '营业利润/流动负债(%)', '存货周转率(次)',
       '总资产同比增长率(%)', '固定资产周转率(次)'],
      dtype='object')
```

数据合并

In [48]:
```
df=pd.merge(df_fs,df_fi,how="inner",on=["年份"])
```

In [49]:
```
df.columns
```

Out[49]:
```
Index(['总资产(元)', '总负债(元)', '股东权益合计(元)', '流动资产合计(元)', '流动负债合计(元)', '短期贷款(元)',
       '长期贷款(元)', '年份', '营运收入(元)', '毛利(元)', '利息收入(元)', '融资成本(元)', '每股基本盈利(元)',
       '经营业务现金净额(元)', '投资业务现金净额(元)', '融资业务现金净额(元)', '期末现金(元)', '每股净资产同比增长率(%)',
       '总负债同比增长率(%)_x', '营业收入同比增长率(%)', '基本每股收益同比增长率(%)_x', '股东权益合计同比增长率(%)_x',
       '每股收益EPS(TTM)(元)', '每股净资产BPS(元)', '每股经营现金净流量(元)', '每股营业收入(元)',
       '销售净利率(%)', '总资产净利率(%)', '净资产收益率(TTM)(%)', '投入资本回报率(TTM)(%)',
       '资产负债率(%)', '流动负债/负债合计(%)', '流动比率', '速动比率', '产权比率', '非流动负债/负债合计(%)',
       '营业利润/负债合计(%)', '营业利润/流动负债(%)', '存货周转率(次)', '总资产同比增长率(%)',
       '固定资产周转率(次)'],
      dtype='object')
```

In [50]:
```
new_column_names = {'总负债同比增长率(%)_x': '总负债同比增长率(%)',
                    '基本每股收益同比增长率(%)_x': '基本每股收益同比增长率(%)',
                    '股东权益合计同比增长率(%)_x': '股东权益合计同比增长率(%)'}
df = df.rename(columns=new_column_names)

df.columns
```

Out[50]:
```
Index(['总资产(元)', '总负债(元)', '股东权益合计(元)', '流动资产合计(元)', '流动负债合计(元)', '短期贷款(元)',
       '长期贷款(元)', '年份', '营运收入(元)', '毛利(元)', '利息收入(元)', '融资成本(元)', '每股基本盈利(元)',
       '经营业务现金净额(元)', '投资业务现金净额(元)', '融资业务现金净额(元)', '期末现金(元)', '每股净资产同比增长率(%)',
       '总负债同比增长率(%)', '营业收入同比增长率(%)', '基本每股收益同比增长率(%)', '股东权益合计同比增长率(%)',
       '每股收益EPS(TTM)(元)', '每股净资产BPS(元)', '每股经营现金净流量(元)', '每股营业收入(元)',
       '销售净利率(%)', '总资产净利率(%)', '净资产收益率(TTM)(%)', '投入资本回报率(TTM)(%)',
       '资产负债率(%)', '流动负债/负债合计(%)', '流动比率', '速动比率', '产权比率', '非流动负债/负债合计(%)',
       '营业利润/负债合计(%)', '营业利润/流动负债(%)', '存货周转率(次)', '总资产同比增长率(%)',
       '固定资产周转率(次)'],
      dtype='object')
```

数据分析

In [51]:
```
df.shape
```

Out[51]:
```
(9, 41)
```

In [52]:
```
df.isnull().sum()
```

Out[52]:
```
总资产(元)                0
总负债(元)                0
股东权益合计(元)            0
流动资产合计(元)            0
流动负债合计(元)            0
短期贷款(元)                1
长期贷款(元)                0
年份                     0
营运收入(元)                0
毛利(元)                  0
利息收入(元)                3
融资成本(元)                2
每股基本盈利(元)            0
经营业务现金净额(元)          0
投资业务现金净额(元)          0
融资业务现金净额(元)          0
期末现金(元)                0
每股净资产同比增长率(%)       1
总负债同比增长率(%)          1
营业收入同比增长率(%)         1
基本每股收益同比增长率(%)      1
股东权益合计同比增长率(%)      1
每股收益EPS(TTM)(元)     0
每股净资产BPS(元)          0
每股经营现金净流量(元)         0
每股营业收入(元)            0
销售净利率(%)              0
总资产净利率(%)             1
净资产收益率(TTM)(%)       0
投入资本回报率(TTM)(%)      1
资产负债率(%)              0
流动负债/负债合计(%)         0
流动比率                   0
速动比率                   0
产权比率                   0
非流动负债/负债合计(%)        0
营业利润/负债合计(%)         0
营业利润/流动负债(%)         0
存货周转率(次)              1
总资产同比增长率(%)          1
固定资产周转率(次)           1
dtype: int64
```

In [53]:
```
#新建一列'上市'，若年份是上市及其以后取值为1，否则为0
df['上市'] = (df['年份'] >= 2018).astype(int)
```

In [54]:
```
df.describe()
```

Out[54]:

	总资产(元)	总负债(元)	股东权益合计(元)	流动资产合计(元)	流动负债合计(元)	
count	9.000000e+00	9.000000e+00	9.000000e+00	9.000000e+00	9.000000e+00	8
mean	1.836617e+11	1.373693e+11	4.629242e+10	1.202090e+11	7.473896e+10	5
std	1.081049e+11	3.991902e+10	1.154252e+11	6.742987e+10	3.818896e+10	3
min	3.913654e+10	7.397782e+10	-1.272107e+11	2.495253e+10	1.646428e+10	2
25%	8.986976e+10	1.257748e+11	-8.663831e+10	6.113846e+10	4.713267e+10	3
50%	1.836292e+11	1.296663e+11	8.165768e+10	1.375391e+11	8.962753e+10	4
75%	2.735072e+11	1.554594e+11	1.374325e+11	1.762828e+11	1.079269e+11	6
max	3.242474e+11	2.170805e+11	1.642618e+11	1.990527e+11	1.157275e+11	1

In [55]:
```
df.columns
```

Out[55]:
```
Index(['总资产(元)', '总负债(元)', '股东权益合计(元)', '流动资产合计(元)', '流动负债合计(元)', '短期贷款(元)',
       '长期贷款(元)', '年份', '营运收入(元)', '毛利(元)', '利息收入(元)', '融资成本(元)', '每股基本盈利(元)',
       '经营业务现金净额(元)', '投资业务现金净额(元)', '融资业务现金净额(元)', '期末现金(元)', '每股净资产同比增长率(%)',
       '总负债同比增长率(%)', '营业收入同比增长率(%)', '基本每股收益同比增长率(%)', '股东权益合计同比增长率(%)',
       '每股收益EPS(TTM)(元)', '每股净资产BPS(元)', '每股经营现金净流量(元)', '每股营业收入(元)',
       '销售净利率(%)', '总资产净利率(%)', '净资产收益率(TTM)(%)', '投入资本回报率(TTM)(%)',
       '资产负债率(%)', '流动负债/负债合计(%)', '流动比率', '速动比率', '产权比率', '非流动负债/负债合计(%)',
       '营业利润/负债合计(%)', '营业利润/流动负债(%)', '存货周转率(次)', '总资产同比增长率(%)', '固定资产周转率(次)',
       '上市'],
      dtype='object')
```

资本结构现状分析

资产负债率分析

对于小米公司而言，通过分析资产负债率，可以了解其债务负担情况，评估其偿债能力和财务风险。一个健康的资产负债率表明公司能够有效地利用债务来扩大业务规模，同时保持较低的财务风险。

In [56]:
```
import matplotlib.pyplot as plt

period1 = (2015, 2018)
period2 = (2018, 2023)

mean_separation1 = df[(df['年份'] >= period1[0]) & (df['年份'] <= period1[1])].g
mean_separation2 = df[(df['年份'] >= period2[0]) & (df['年份'] <= period2[1])].g
```

```
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.rcParams['axes.unicode_minus'] = False

#2015-2018
plt.figure(figsize=(10, 6))
plt.subplot(2, 1, 1)
plt.plot(mean_separation1.index, mean_separation1.values, marker='o', label='200
plt.xlabel('年份')
plt.ylabel('资产负债率')
plt.title('小米公司非上市期资产负债率均值趋势')
plt.legend()

#2018-2023
plt.subplot(2, 1, 2)
plt.plot(mean_separation2.index, mean_separation2.values, marker='o', label='201
plt.xlabel('年份')
plt.ylabel('资产负债率')
plt.title('小米公司上市期资产负债率均值趋势')
plt.legend()


plt.tight_layout()
plt.show()
```

Out[56]: <Figure size 1000x600 with 0 Axes>

Out[56]: <Axes: >

Out[56]: [<matplotlib.lines.Line2D at 0x1fd5cfd4950>]

Out[56]: Text(0.5, 0, '年份')

Out[56]: Text(0, 0.5, '资产负债率')

Out[56]: Text(0.5, 1.0, '小米公司非上市期资产负债率均值趋势')

Out[56]: <matplotlib.legend.Legend at 0x1fd5cfd45c0>

Out[56]: <Axes: >

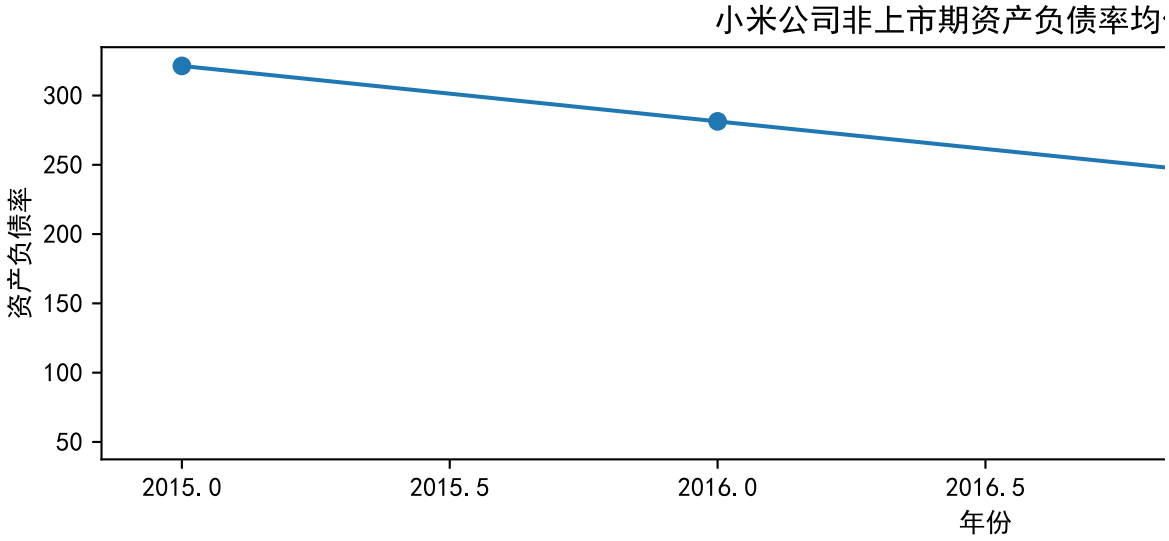
Out[56]: [<matplotlib.lines.Line2D at 0x1fd5cff0a40>]

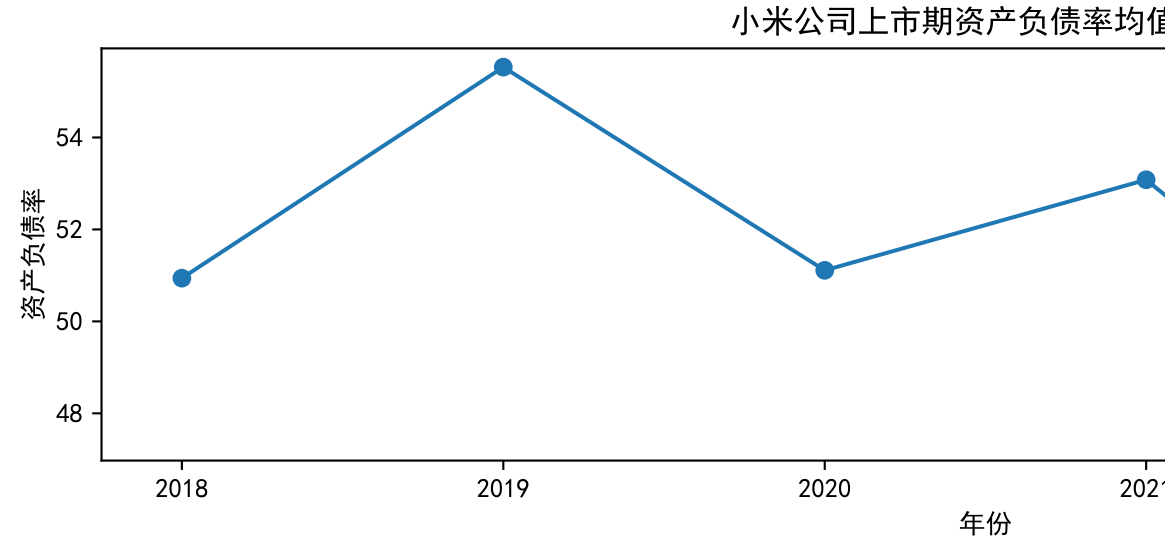
Out[56]: Text(0.5, 0, '年份')

Out[56]: Text(0, 0.5, '资产负债率')

Out[56]: Text(0.5, 1.0, '小米公司上市期资产负债率均值趋势')

Out[56]: <matplotlib.legend.Legend at 0x1fd5d09f8f0>





产权比率分析

分析小米公司的产权比率有助于理解其资本结构是否合理，以及股东权益在公司资产中的比重。这有助于投资者判断公司的财务稳定性及长期发展潜力。

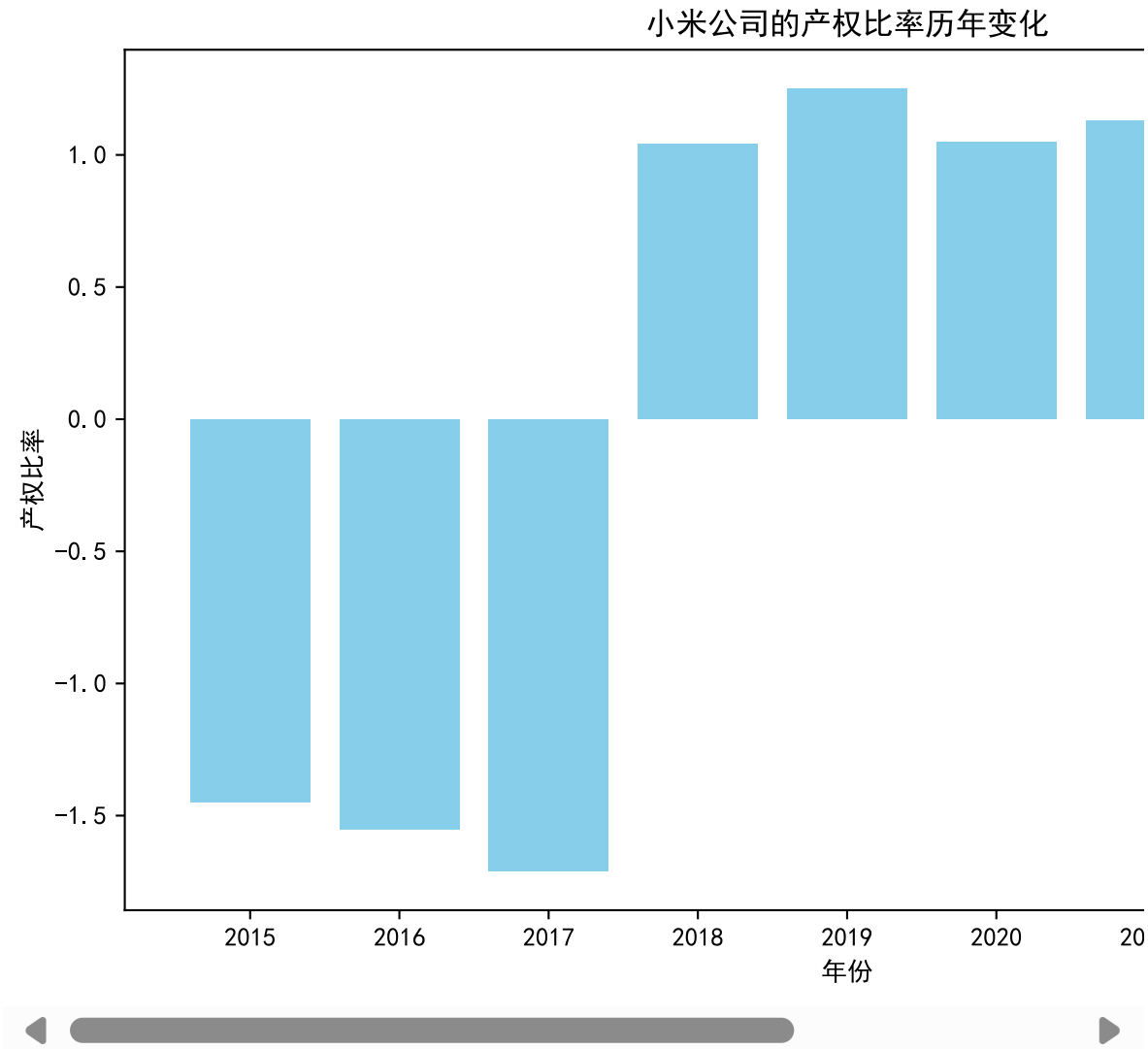
In [57]:
```
import matplotlib.pyplot as plt

plt.figure(figsize=(10, 6))
plt.bar(df['年份'], df['产权比率'], color='skyblue')

plt.title('小米公司的产权比率历年变化')
plt.xlabel('年份')
plt.ylabel('产权比率')

plt.xticks(df['年份'])

plt.show()
```

Out[57]: <Figure size 1000x600 with 0 Axes>

Out[57]: <BarContainer object of 9 artists>

Out[57]: Text(0.5, 1.0, '小米公司的产权比率历年变化')

```
Out[57]: Text(0.5, 0, '年份')

Out[57]: Text(0, 0.5, '产权比率')

Out[57]: ([<matplotlib.axis.XTick at 0x1fd5d0e7ef0>,
  <matplotlib.axis.XTick at 0x1fd5d1bc8f0>,
  <matplotlib.axis.XTick at 0x1fd5cff09e0>,
  <matplotlib.axis.XTick at 0x1fd5d0f98e0>,
  <matplotlib.axis.XTick at 0x1fd5d1be300>,
  <matplotlib.axis.XTick at 0x1fd5d1bebd0>,
  <matplotlib.axis.XTick at 0x1fd5d1bedb0>,
  <matplotlib.axis.XTick at 0x1fd5d1bf5c0>,
  <matplotlib.axis.XTick at 0x1fd5d1bfef0>],
 [Text(2023, 0, '2023'),
  Text(2022, 0, '2022'),
  Text(2021, 0, '2021'),
  Text(2020, 0, '2020'),
  Text(2019, 0, '2019'),
  Text(2018, 0, '2018'),
  Text(2017, 0, '2017'),
  Text(2016, 0, '2016'),
  Text(2015, 0, '2015')])
```



固定资产周转率

对小米公司来说，分析固定资产周转率可以评估其固定资产的利用效率，了解公司在生产、销售等环节的运营效率。高周转率意味着公司能更有效地利用固定资产创造收入。

In [80]:
```
import matplotlib.pyplot as plt
import pandas as pd


plt.figure(figsize=(10, 6))

# 绘制柱状图
bars = plt.bar(df['年份'], df['固定资产周转率(次)'], color='#3498db', edgecolor=

# 在每个柱子上方显示数值
for bar in bars:
    yval = bar.get_height()
    plt.text(bar.get_x() + bar.get_width()/2, yval + 0.5, f'{yval:.2f}', ha='cen

# 设置标题和标签
plt.title('小米公司固定资产周转率历年变化', fontsize=18, fontweight='bold')
plt.xlabel('年份', fontsize=14)
plt.ylabel('固定资产周转率(次)', fontsize=14)

# 设置X轴刻度
plt.xticks(df['年份'], fontsize=12)

# 添加网格
plt.grid(True, axis='y', linestyle='--', alpha=0.5, color='lightgray')

# 调整布局
plt.tight_layout()

# 显示图表
plt.show()
```

Out[80]:
```
<Figure size 1000x600 with 0 Axes>
```

Out[80]:
```
Text(2023.0, 24.21, '23.71')
```

Out[80]:
```
Text(2022.0, 35.28, '34.78')
```

Out[80]:
```
Text(2021.0, 49.98, '49.48')
```

Out[80]:
```
Text(2020.0, 37.48, '36.98')
```

Out[80]:
```
Text(2019.0, 34.63, '34.13')
```

Out[80]:
```
Text(2018.0, 51.95, '51.45')
```

Out[80]:
```
Text(2017.0, 89.38, '88.88')
```

Out[80]:
```
Text(2016.0, 120.71, '120.21')
```

Out[80]:
```
Text(2015.0, nan, 'nan')
```

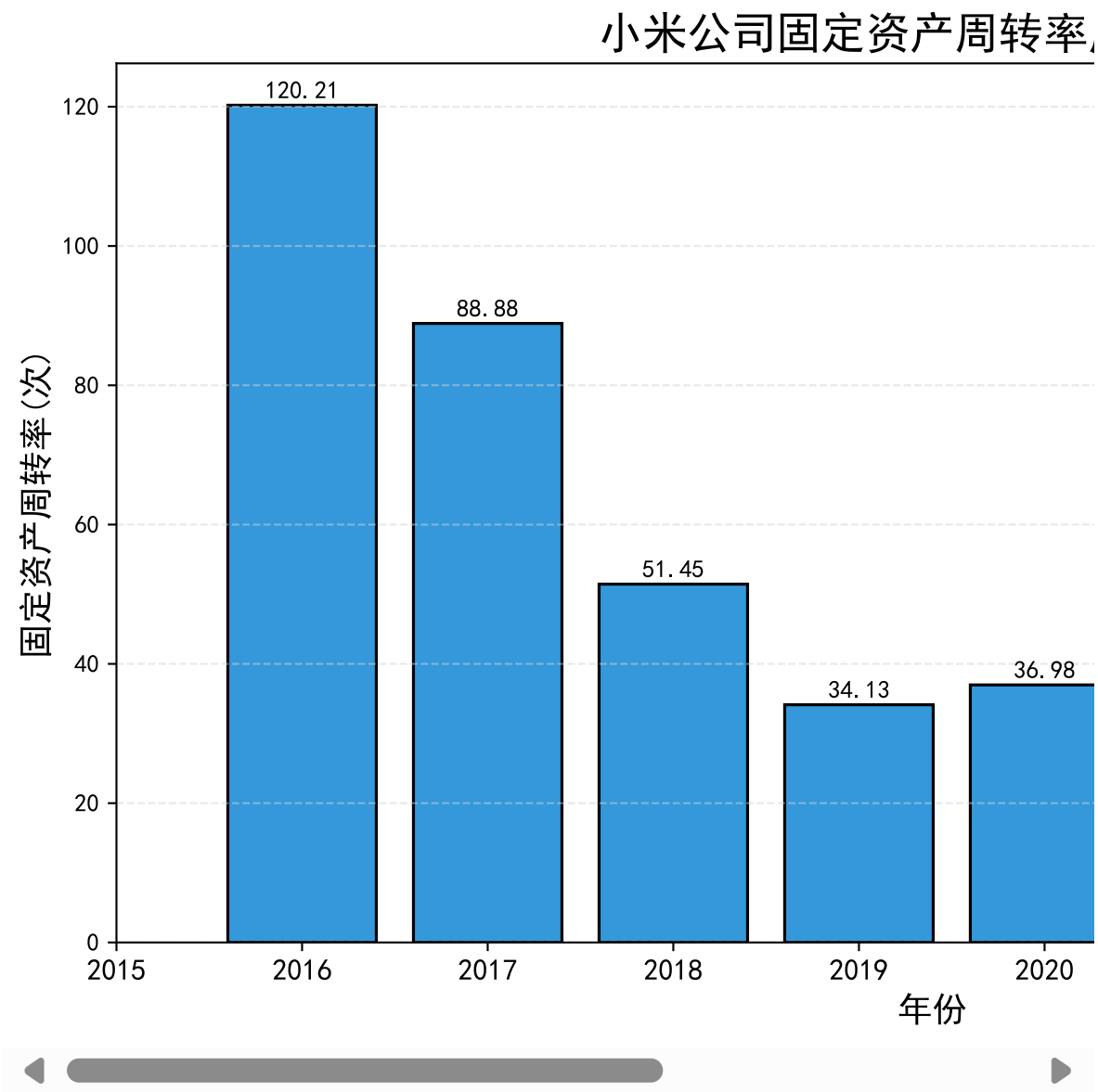
Out[80]:
```
Text(0.5, 1.0, '小米公司固定资产周转率历年变化')
```

Out[80]:
```
Text(0.5, 0, '年份')
```

Out[80]:
```
Text(0, 0.5, '固定资产周转率(次)')
```

```
Out[80]: ([<matplotlib.axis.XTick at 0x1fd64473dd0>,
           <matplotlib.axis.XTick at 0x1fd64507590>,
           <matplotlib.axis.XTick at 0x1fd6487b3e0>,
           <matplotlib.axis.XTick at 0x1fd64878050>,
           <matplotlib.axis.XTick at 0x1fd648be810>,
           <matplotlib.axis.XTick at 0x1fd648bf140>,
           <matplotlib.axis.XTick at 0x1fd648bfaa0>,
           <matplotlib.axis.XTick at 0x1fd648ec3b0>,
           <matplotlib.axis.XTick at 0x1fd648be9f0>],
          [Text(2023, 0, '2023'),
           Text(2022, 0, '2022'),
           Text(2021, 0, '2021'),
           Text(2020, 0, '2020'),
           Text(2019, 0, '2019'),
           Text(2018, 0, '2018'),
           Text(2017, 0, '2017'),
           Text(2016, 0, '2016'),
           Text(2015, 0, '2015')])
```

```
posx and posy should be finite values
posx and posy should be finite values
```



长期负债率分析

通过分析小米公司的长期负债率，可以了解其长期债务结构和偿债压力，帮助投资者和管理层评估公司的长期财务稳定性和融资能力。

In [58]:
```
df['长期负债率']=df['长期贷款(元)']/df['总资产(元)']

import matplotlib.pyplot as plt

plt.figure(figsize=(10, 6))
plt.plot(df['年份'], df['长期负债率'], marker='o', linestyle='-', color='b', lab


plt.title('小米公司的长期债务比率历年变化', fontsize=16)
plt.xlabel('年份', fontsize=14)
plt.ylabel('长期债务比率', fontsize=14)

plt.xticks(df['年份'])

plt.grid(True, linestyle='--', alpha=0.7)

plt.legend()

plt.show()
```

Out[58]:
```
<Figure size 1000x600 with 0 Axes>
```

Out[58]:
```
[<matplotlib.lines.Line2D at 0x1fd5f4a2690>]
```

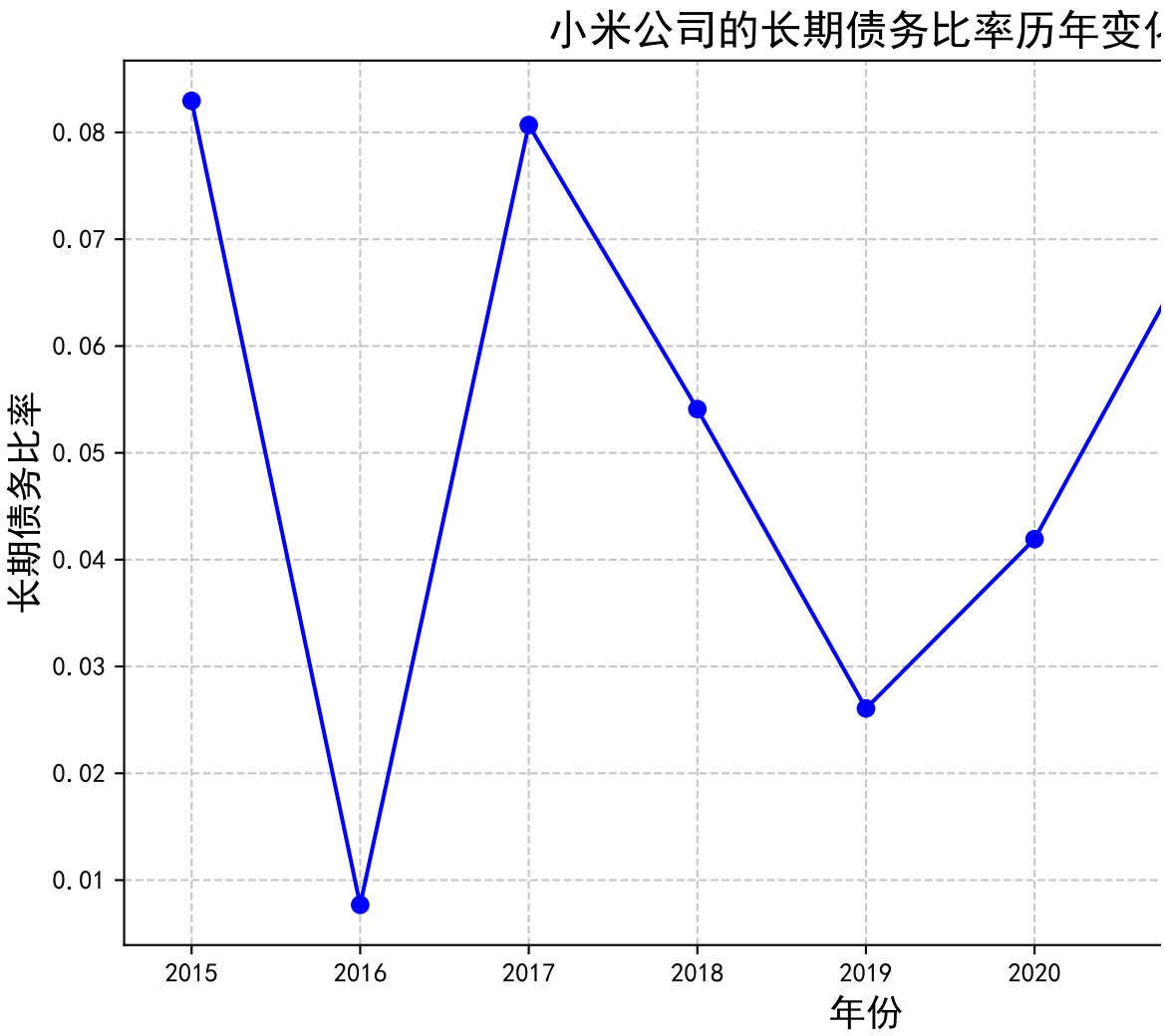
Out[58]:
```
Text(0.5, 1.0, '小米公司的长期债务比率历年变化')
```

Out[58]:
```
Text(0.5, 0, '年份')
```

Out[58]:
```
Text(0, 0.5, '长期债务比率')
```

Out[58]:
```
([<matplotlib.axis.XTick at 0x1fd5d0d45c0>,
  <matplotlib.axis.XTick at 0x1fd5d22d310>,
  <matplotlib.axis.XTick at 0x1fd5d1bf200>,
  <matplotlib.axis.XTick at 0x1fd5f4a32c0>,
  <matplotlib.axis.XTick at 0x1fd5f4a38c0>,
  <matplotlib.axis.XTick at 0x1fd5f4a3da0>,
  <matplotlib.axis.XTick at 0x1fd5f4d0a10>,
  <matplotlib.axis.XTick at 0x1fd5f4d1310>,
  <matplotlib.axis.XTick at 0x1fd5f4d1be0>],
 [Text(2023, 0, '2023'),
  Text(2022, 0, '2022'),
  Text(2021, 0, '2021'),
  Text(2020, 0, '2020'),
  Text(2019, 0, '2019'),
  Text(2018, 0, '2018'),
  Text(2017, 0, '2017'),
  Text(2016, 0, '2016'),
  Text(2015, 0, '2015')])
```

Out[58]:
```
<matplotlib.legend.Legend at 0x1fd5d0f9820>
```



存货周转率

对小米公司而言，分析存货周转率可以评估其库存管理水平和产品销售速度，及时发现库存积压或缺货问题，优化供应链管理，提高资金使用效率。

In [72]:
```
df.columns
```

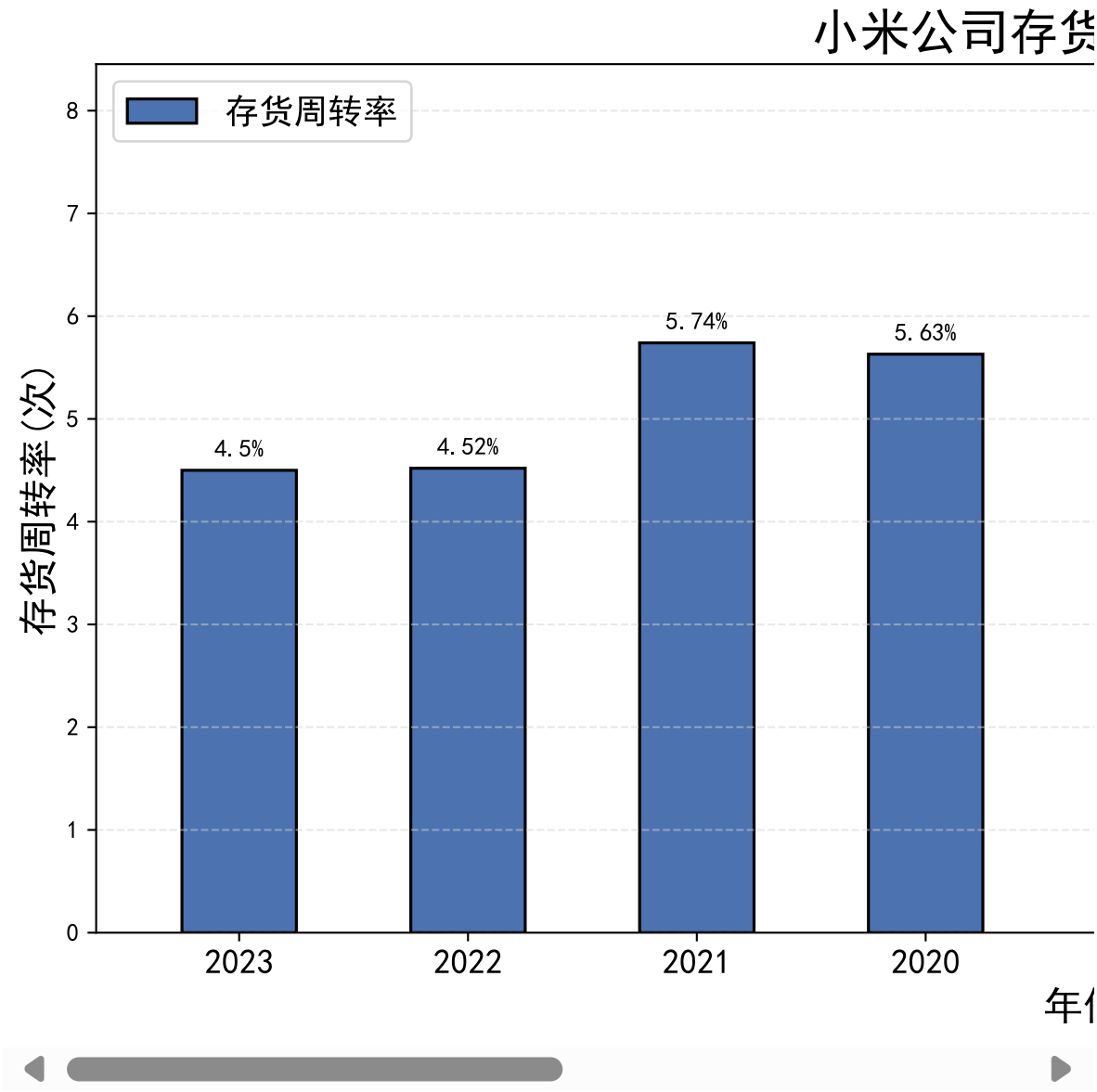
Out[72]:
```
Index(['总资产(元)', '总负债(元)', '股东权益合计(元)', '流动资产合计(元)', '流动负债合计(元)', '短期贷款(元)',
       '长期贷款(元)', '年份', '营运收入(元)', '毛利(元)', '利息收入(元)', '融资成本(元)', '每股基本盈利(元)',
       '经营业务现金净额(元)', '投资业务现金净额(元)', '融资业务现金净额(元)', '期末现金(元)', '每股净资产同比增长率(%)',
       '总负债同比增长率(%)', '营业收入同比增长率(%)', '基本每股收益同比增长率(%)', '股东权益合计同比增长率(%)',
       '每股收益EPS(TTM)(元)', '每股净资产BPS(元)', '每股经营现金净流量(元)', '每股营业收入(元)',
       '销售净利率(%)', '总资产净利率(%)', '净资产收益率(TTM)(%)', '投入资本回报率(TTM)(%)',
       '资产负债率(%)', '流动负债/负债合计(%)', '流动比率', '速动比率', '产权比率', '非流动负债/负债合计(%)',
       '营业利润/负债合计(%)', '营业利润/流动负债(%)', '存货周转率(次)', '总资产同比增长率(%)', '固定资产周转率(次)',
       '上市', '长期负债率', '资本结构比率'],
      dtype='object')
```

In [78]:
```python
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd


fig, ax = plt.subplots(figsize=(12, 6))

bar_width = 0.5
index = np.arange(len(df['年份']))

# 绘制柱状图
ax.bar(index, df['存货周转率(次)'], bar_width, color='#4C72B0', edgecolor='black

# 在每个柱子上方显示数值
for i, txt in enumerate(df['存货周转率(次)']):
    ax.text(i, txt + 0.1, f'{txt}%', ha='center', va='bottom', fontsize=10, colo

# 设置标题和标签
ax.set_title('小米公司存货周转率分析', fontsize=20, fontweight='bold')
ax.set_xlabel('年份', fontsize=16)
ax.set_ylabel('存货周转率(次)', fontsize=16)

# 设置X轴刻度
ax.set_xticks(index)
ax.set_xticklabels(df['年份'], fontsize=14)

# 添加网格
ax.grid(True, axis='y', linestyle='--', alpha=0.5, color='lightgray')

# 显示图例
ax.legend(fontsize=14, loc='upper left', framealpha=0.8)

# 调整布局
plt.tight_layout()

# 显示图表
plt.show()
```

Out[78]:
```
<BarContainer object of 9 artists>
```

Out[78]:
```
Text(0, 4.6, '4.5%')
```

```
Out[78]: Text(1, 4.619999999999999, '4.52%')

Out[78]: Text(2, 5.84, '5.74%')

Out[78]: Text(3, 5.7299999999999995, '5.63%')

Out[78]: Text(4, 5.81, '5.71%')

Out[78]: Text(5, 6.77, '6.67%')

Out[78]: Text(6, 8.15, '8.05%')

Out[78]: Text(7, 7.29, '7.19%')

Out[78]: Text(8, nan, 'nan%')

Out[78]: Text(0.5, 1.0, '小米公司存货周转率分析')

Out[78]: Text(0.5, 0, '年份')

Out[78]: Text(0, 0.5, '存货周转率(次)')

Out[78]: [<matplotlib.axis.XTick at 0x1fd64488260>,
          <matplotlib.axis.XTick at 0x1fd64470770>,
          <matplotlib.axis.XTick at 0x1fd63e62780>,
          <matplotlib.axis.XTick at 0x1fd63e1cc50>,
          <matplotlib.axis.XTick at 0x1fd644c22d0>,
          <matplotlib.axis.XTick at 0x1fd644c2c00>,
          <matplotlib.axis.XTick at 0x1fd644c3500>,
          <matplotlib.axis.XTick at 0x1fd61e4f170>,
          <matplotlib.axis.XTick at 0x1fd644c3bf0>]

Out[78]: [Text(0, 0, '2023'),
          Text(1, 0, '2022'),
          Text(2, 0, '2021'),
          Text(3, 0, '2020'),
          Text(4, 0, '2019'),
          Text(5, 0, '2018'),
          Text(6, 0, '2017'),
          Text(7, 0, '2016'),
          Text(8, 0, '2015')]

Out[78]: <matplotlib.legend.Legend at 0x1fd64471820>

posx and posy should be finite values
posx and posy should be finite values
```



流动负债率和非流动负债率

分析小米公司的流动负债率和非流动负债率，可以帮助了解其短期和长期偿债能力，评估财务灵活性和风险分布，为制定合理的财务策略提供依据。

In [60]:
```
import matplotlib.pyplot as plt

plt.figure(figsize=(12, 6))

plt.plot(df['年份'], df['流动负债/负债合计(%)'], label='流动负债率', marker='o',
plt.plot(df['年份'], df['非流动负债/负债合计(%)'], label='非流动负债率', marker='

plt.title('小米公司流动负债率与非流动负债率变化图', fontsize=18)
plt.xlabel('年份', fontsize=14)
plt.ylabel('比率', fontsize=14)

plt.xticks(df['年份'])

plt.grid(True, linestyle='--', alpha=0.7)

plt.legend(loc='upper right')

plt.tight_layout()
```

```
plt.show()
```

Out[60]:
```
<Figure size 1200x600 with 0 Axes>
```

Out[60]:
```
[<matplotlib.lines.Line2D at 0x1fd5f895610>]
```

Out[60]:
```
[<matplotlib.lines.Line2D at 0x1fd5f533d70>]
```

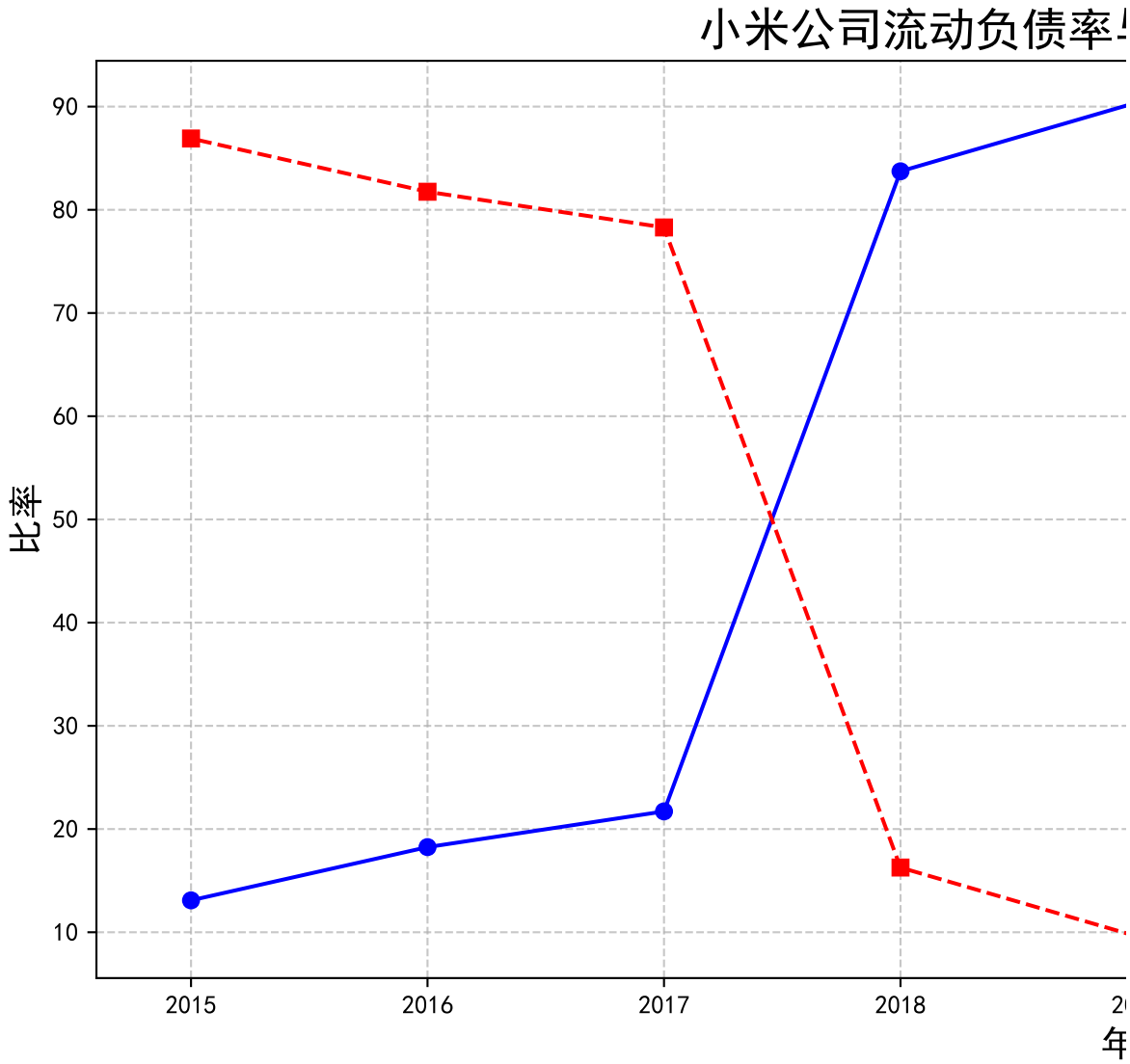
Out[60]:
```
Text(0.5, 1.0, '小米公司流动负债率与非流动负债率变化图')
```

Out[60]:
```
Text(0.5, 0, '年份')
```

Out[60]:
```
Text(0, 0.5, '比率')
```

Out[60]:
```
([<matplotlib.axis.XTick at 0x1fd5f595fa0>,
  <matplotlib.axis.XTick at 0x1fd5f5794c0>,
  <matplotlib.axis.XTick at 0x1fd5f507200>,
  <matplotlib.axis.XTick at 0x1fd5f5798e0>,
  <matplotlib.axis.XTick at 0x1fd5f896810>,
  <matplotlib.axis.XTick at 0x1fd5f897080>,
  <matplotlib.axis.XTick at 0x1fd5f895a60>,
  <matplotlib.axis.XTick at 0x1fd5f897a40>,
  <matplotlib.axis.XTick at 0x1fd5f8c0380>],
 [Text(2023, 0, '2023'),
  Text(2022, 0, '2022'),
  Text(2021, 0, '2021'),
  Text(2020, 0, '2020'),
  Text(2019, 0, '2019'),
  Text(2018, 0, '2018'),
  Text(2017, 0, '2017'),
  Text(2016, 0, '2016'),
  Text(2015, 0, '2015')])
```

Out[60]:
```
<matplotlib.legend.Legend at 0x1fd5f533410>
```



总负债同比增长率

对小米公司进行总负债同比增长率分析，可以了解其负债扩张的速度和原因，评估财务杠杆的使用效果和潜在风险，为未来财务规划提供参考。

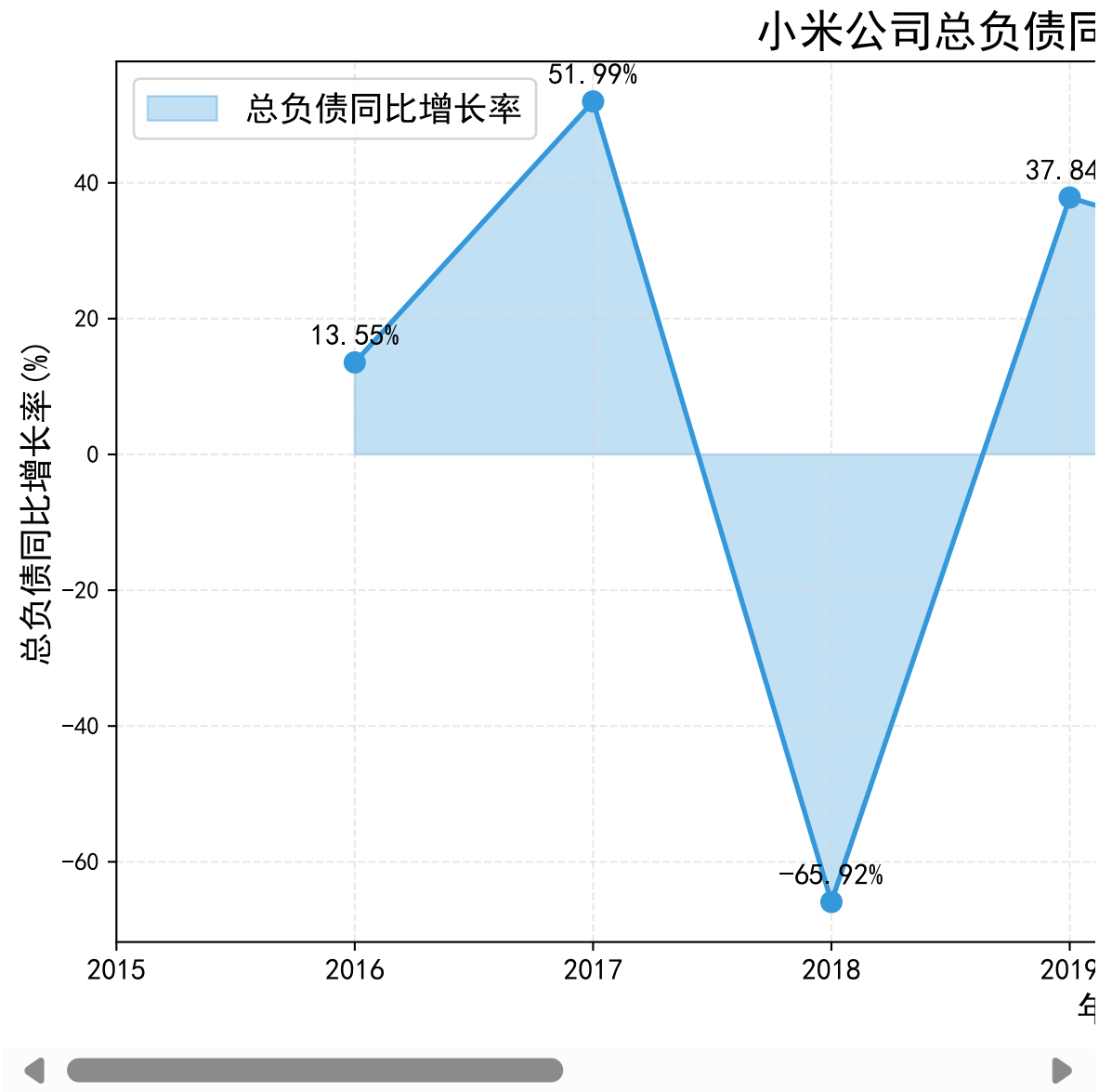
In [81]:
```
import matplotlib.pyplot as plt
import pandas as pd


plt.figure(figsize=(12, 6))

# 绘制带填充区域的折线图
plt.fill_between(df['年份'], df['总负债同比增长率(%)'], color='#3498db', alpha=0
plt.plot(df['年份'], df['总负债同比增长率(%)'], marker='o', linestyle='-', color

# 在每个数据点旁边显示数值
for i, txt in enumerate(df['总负债同比增长率(%)']):
    plt.text(df['年份'][i], txt + 2, f'{txt:.2f}%', ha='center', va='bottom', fo

# 设置标题和标签
plt.title('小米公司总负债同比增长率历年变化', fontsize=18, fontweight='bold')
plt.xlabel('年份', fontsize=14)
plt.ylabel('总负债同比增长率(%)', fontsize=14)
```

```python
# 设置X轴刻度
plt.xticks(df['年份'], fontsize=12)

# 添加网格
plt.grid(True, linestyle='--', alpha=0.5, color='lightgray')

# 显示图例
plt.legend(fontsize=14, loc='upper left', framealpha=0.8)

# 调整布局
plt.tight_layout()

# 显示图表
plt.show()
```

```
Out[81]: <Figure size 1200x600 with 0 Axes>

Out[81]: <matplotlib.collections.PolyCollection at 0x1fd649019a0>

Out[81]: [<matplotlib.lines.Line2D at 0x1fd649006e0>]

Out[81]: Text(2023, 25.46, '23.46%')

Out[81]: Text(2022, -14.64, '-16.64%')

Out[81]: Text(2021, 21.89, '19.89%')

Out[81]: Text(2020, 29.16, '27.16%')

Out[81]: Text(2019, 39.84, '37.84%')

Out[81]: Text(2018, -63.92, '-65.92%')

Out[81]: Text(2017, 53.99, '51.99%')

Out[81]: Text(2016, 15.55, '13.55%')

Out[81]: Text(2015, nan, 'nan%')

Out[81]: Text(0.5, 1.0, '小米公司总负债同比增长率历年变化')

Out[81]: Text(0.5, 0, '年份')

Out[81]: Text(0, 0.5, '总负债同比增长率(%)')

Out[81]: ([<matplotlib.axis.XTick at 0x1fd64900770>,
  <matplotlib.axis.XTick at 0x1fd64902c30>,
  <matplotlib.axis.XTick at 0x1fd648ed040>,
  <matplotlib.axis.XTick at 0x1fd6491b920>,
  <matplotlib.axis.XTick at 0x1fd64baa660>,
  <matplotlib.axis.XTick at 0x1fd64baae70>,
  <matplotlib.axis.XTick at 0x1fd64bab770>,
  <matplotlib.axis.XTick at 0x1fd64baab70>,
  <matplotlib.axis.XTick at 0x1fd64babf80>],
 [Text(2023, 0, '2023'),
  Text(2022, 0, '2022'),
  Text(2021, 0, '2021'),
  Text(2020, 0, '2020'),
  Text(2019, 0, '2019'),
  Text(2018, 0, '2018'),
  Text(2017, 0, '2017'),
  Text(2016, 0, '2016'),
  Text(2015, 0, '2015')])
```

Out[81]: <matplotlib.legend.Legend at 0x1fd64528500>

```
posx and posy should be finite values
posx and posy should be finite values
```



总资产净利率

通过分析小米公司的总资产净利率，可以评估其整体资产的盈利能力和经营效率，了解公司创造利润的能力和资产使用的有效性，为投资者和管理层提供决策支持。

In [86]:
```python
import matplotlib.pyplot as plt

plt.figure(figsize=(10, 6))

# 绘制折线图，使用明亮的颜色
plt.plot(df['年份'], df['总资产净利率(%)'], marker='o', linestyle='-', color='#F

# 在每个数据点旁边显示数值
for x, y in zip(df['年份'], df['总资产净利率(%)']):
    plt.text(x, y + 0.01, f'{y:.2f}', ha='center', va='bottom', fontsize=12, col

# 设置标题和标签
plt.title('小米公司的总资产净利率历年变化', fontsize=16)
plt.xlabel('年份', fontsize=14)
```

```
plt.ylabel('总资产净利率', fontsize=14)

# 设置X轴刻度
plt.xticks(df['年份'], fontsize=12)

# 添加网格
plt.grid(True, linestyle='--', alpha=0.7)

# 显示图例
plt.legend(fontsize=14, loc='upper left', framealpha=0.8)

# 调整布局
plt.tight_layout()

# 显示图表
plt.show()
```

Out[86]:
```
<Figure size 1000x600 with 0 Axes>
```

Out[86]:
```
[<matplotlib.lines.Line2D at 0x1fd64473d10>]
```

Out[86]:
```
Text(2023, 5.859999999999999, '5.85')
```

Out[86]:
```
Text(2022, 0.88, '0.87')
```

Out[86]:
```
Text(2021, 7.09, '7.08')
```

Out[86]:
```
Text(2020, 9.32, '9.31')
```

Out[86]:
```
Text(2019, 6.12, '6.11')
```

Out[86]:
```
Text(2018, 11.54, '11.53')
```

Out[86]:
```
Text(2017, -62.32, '-62.33')
```

Out[86]:
```
Text(2016, 1.24, '1.23')
```

Out[86]:
```
Text(2015, nan, 'nan')
```

Out[86]:
```
Text(0.5, 1.0, '小米公司的总资产净利率历年变化')
```

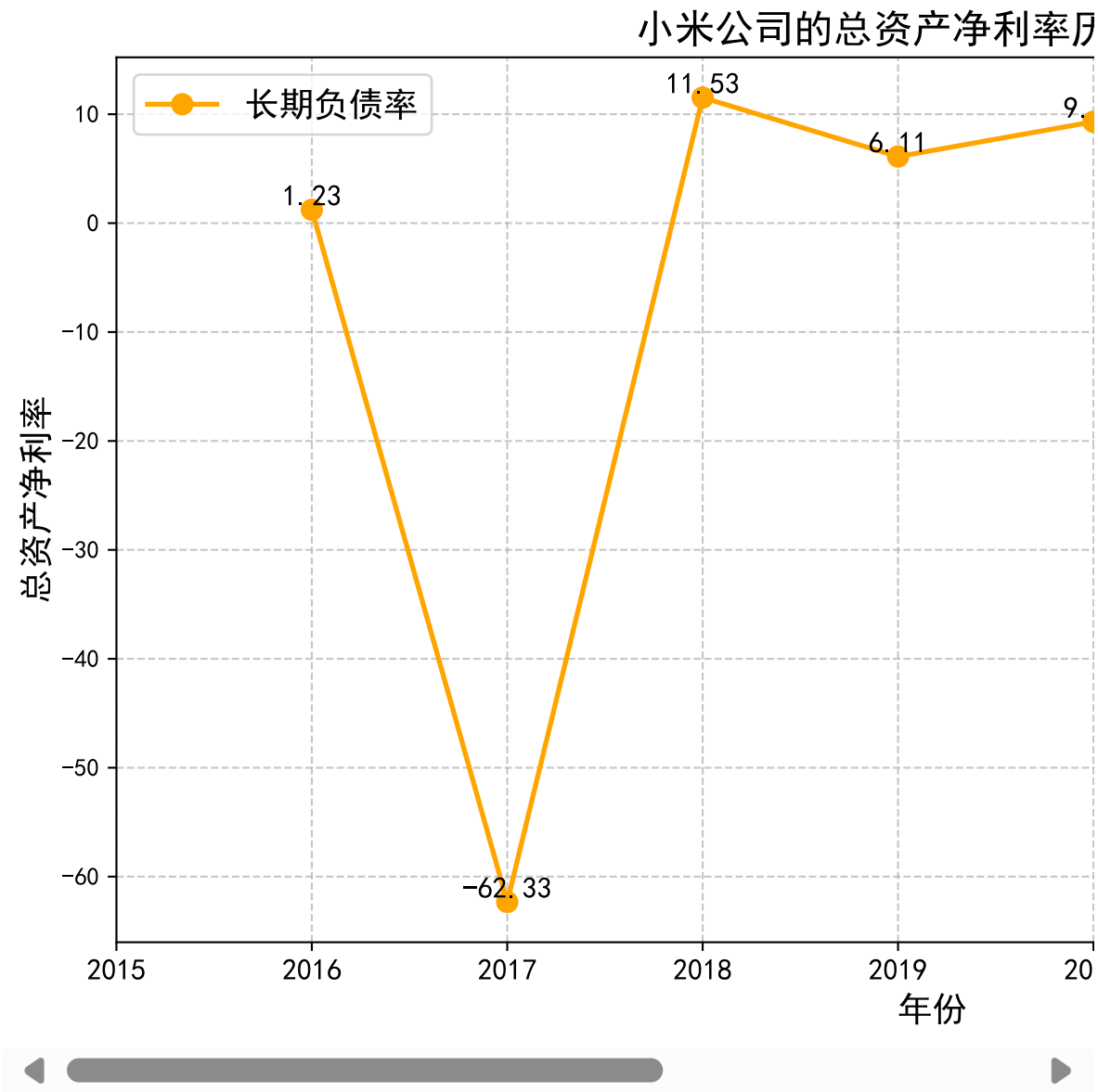
Out[86]:
```
Text(0.5, 0, '年份')
```

Out[86]:
```
Text(0, 0.5, '总资产净利率')
```

Out[86]:
```
([<matplotlib.axis.XTick at 0x1fd61584c20>,
  <matplotlib.axis.XTick at 0x1fd644f68d0>,
  <matplotlib.axis.XTick at 0x1fd632ba780>,
  <matplotlib.axis.XTick at 0x1fd64903ce0>,
  <matplotlib.axis.XTick at 0x1fd648bf020>,
  <matplotlib.axis.XTick at 0x1fd648ee540>,
  <matplotlib.axis.XTick at 0x1fd648ee9c0>,
  <matplotlib.axis.XTick at 0x1fd645078c0>,
  <matplotlib.axis.XTick at 0x1fd6484b350>],
 [Text(2023, 0, '2023'),
  Text(2022, 0, '2022'),
  Text(2021, 0, '2021'),
  Text(2020, 0, '2020'),
  Text(2019, 0, '2019'),
  Text(2018, 0, '2018'),
  Text(2017, 0, '2017'),
  Text(2016, 0, '2016'),
  Text(2015, 0, '2015')])
```

Out[86]: <matplotlib.legend.Legend at 0x1fd645282f0>

```
posx and posy should be finite values
posx and posy should be finite values
```



融资结构分析

资本结构比率的趋势分析可以看出小米是在稳健发展，还是在激进扩张？以及在面对市场波动，它是否有足够的财务弹性？

In [61]:
```
df['资本结构比率']=df['总负债(元)']/df['股东权益合计(元)']

import matplotlib.pyplot as plt

plt.figure(figsize=(12, 6))

plt.fill_between(df['年份'], df['资本结构比率'], color='#FFB6C1', alpha=0.7, lab

plt.plot(df['年份'], df['资本结构比率'], marker='o', linestyle='-', color='#FF69

plt.title('小米公司资本结构比率变化趋势', fontsize=16)
plt.xlabel('年份', fontsize=14)
plt.ylabel('资本结构比率', fontsize=14)
```

```
plt.xticks(df['年份'])

plt.grid(True, linestyle='--', alpha=0.7)

for i, txt in enumerate(df['资本结构比率']):
    plt.annotate(round(txt, 2), (df['年份'][i], txt), textcoords="offset points"

plt.legend()

plt.tight_layout()

plt.show()
```

Out[61]:
```
<Figure size 1200x600 with 0 Axes>
```

Out[61]:
```
<matplotlib.collections.PolyCollection at 0x1fd5f5799a0>
```

Out[61]:
```
[<matplotlib.lines.Line2D at 0x1fd5f906f30>]
```

Out[61]:
```
Text(0.5, 1.0, '小米公司资本结构比率变化趋势')
```

Out[61]:
```
Text(0.5, 0, '年份')
```

Out[61]:
```
Text(0, 0.5, '资本结构比率')
```

Out[61]:
```
([<matplotlib.axis.XTick at 0x1fd4bf9e240>,
  <matplotlib.axis.XTick at 0x1fd5f8f5610>,
  <matplotlib.axis.XTick at 0x1fd5f8e20f0>,
  <matplotlib.axis.XTick at 0x1fd5f907830>,
  <matplotlib.axis.XTick at 0x1fd5f907da0>,
  <matplotlib.axis.XTick at 0x1fd5f94ca10>,
  <matplotlib.axis.XTick at 0x1fd5f94d310>,
  <matplotlib.axis.XTick at 0x1fd5f906cf0>,
  <matplotlib.axis.XTick at 0x1fd5f94c350>],
 [Text(2023, 0, '2023'),
  Text(2022, 0, '2022'),
  Text(2021, 0, '2021'),
  Text(2020, 0, '2020'),
  Text(2019, 0, '2019'),
  Text(2018, 0, '2018'),
  Text(2017, 0, '2017'),
  Text(2016, 0, '2016'),
  Text(2015, 0, '2015')])
```

Out[61]:
```
Text(0, 5, '0.97')
```

Out[61]:
```
Text(0, 5, '0.9')
```

Out[61]:
```
Text(0, 5, '1.13')
```

Out[61]:
```
Text(0, 5, '1.05')
```

Out[61]:
```
Text(0, 5, '1.25')
```

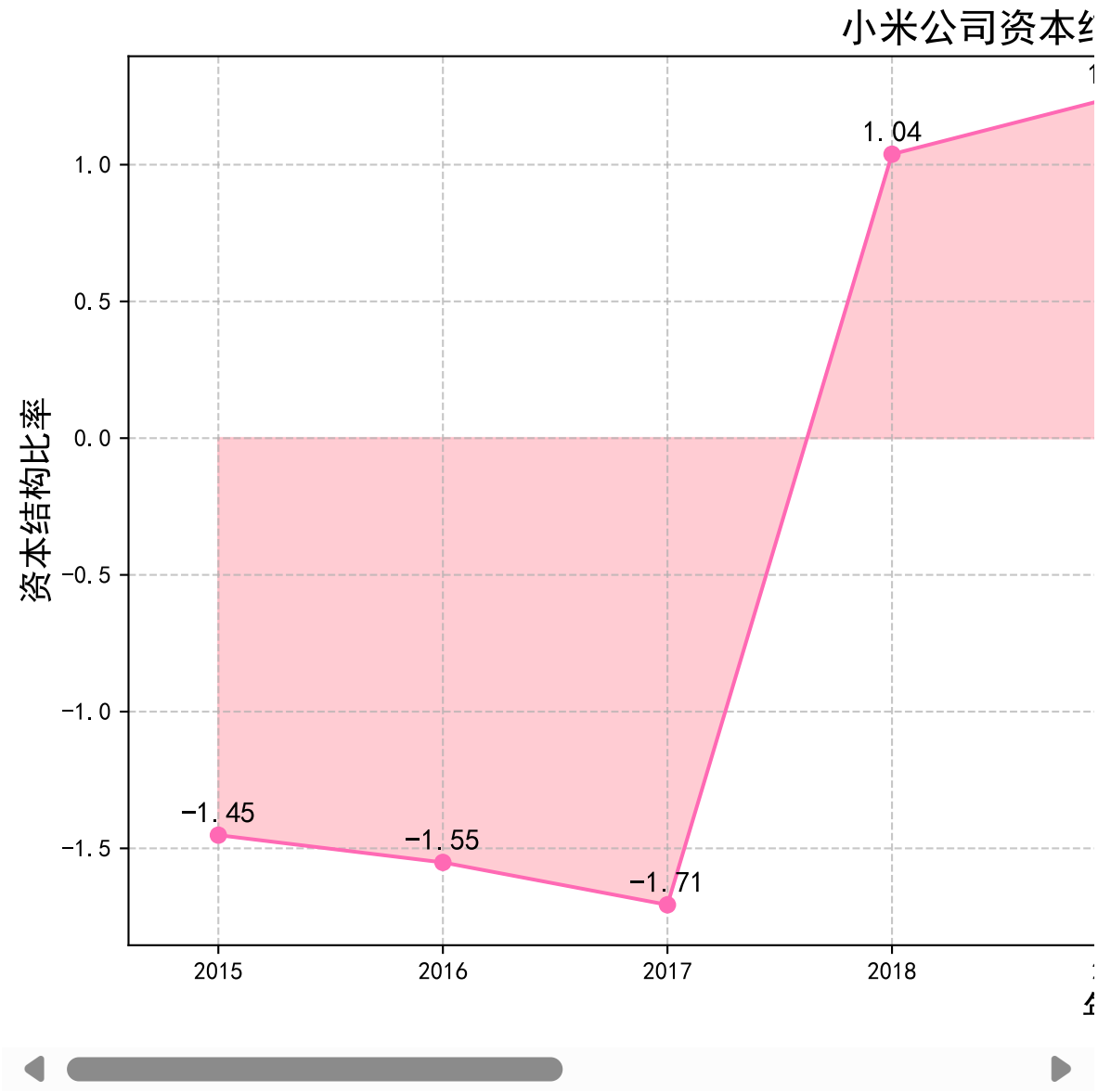
Out[61]:
```
Text(0, 5, '1.04')
```

Out[61]:
```
Text(0, 5, '-1.71')
```

Out[61]:
```
Text(0, 5, '-1.55')
```

Out[61]:
```
Text(0, 5, '-1.45')
```

Out[61]:
```
<matplotlib.legend.Legend at 0x1fd5f9071a0>
```



每股收益EPS(TTM)与每股净资产BPS

In [62]:
```
fig, ax = plt.subplots(figsize=(12, 6))


ax.plot(df['年份'], df['每股收益EPS(TTM)(元)'], marker='o', linestyle='-', color
ax.plot(df['年份'], df['每股净资产BPS(元)'], marker='v', linestyle='--', color='


for i, txt in enumerate(df['每股收益EPS(TTM)(元)']):
    ax.annotate(f'{txt:.2f}', (df['年份'][i], df['每股收益EPS(TTM)(元)'][i]), te

for i, txt in enumerate(df['每股净资产BPS(元)']):
    ax.annotate(f'{txt:.2f}', (df['年份'][i], df['每股净资产BPS(元)'][i]), textc


ax.set_title('小米公司每股收益与净资产分析', fontsize=18, fontweight='bold')
ax.set_xlabel('年份', fontsize=14)
ax.set_ylabel('数额 (元)', fontsize=14)


ax.set_xticks(df['年份'])
ax.set_xticklabels(df['年份'], fontsize=12)
```

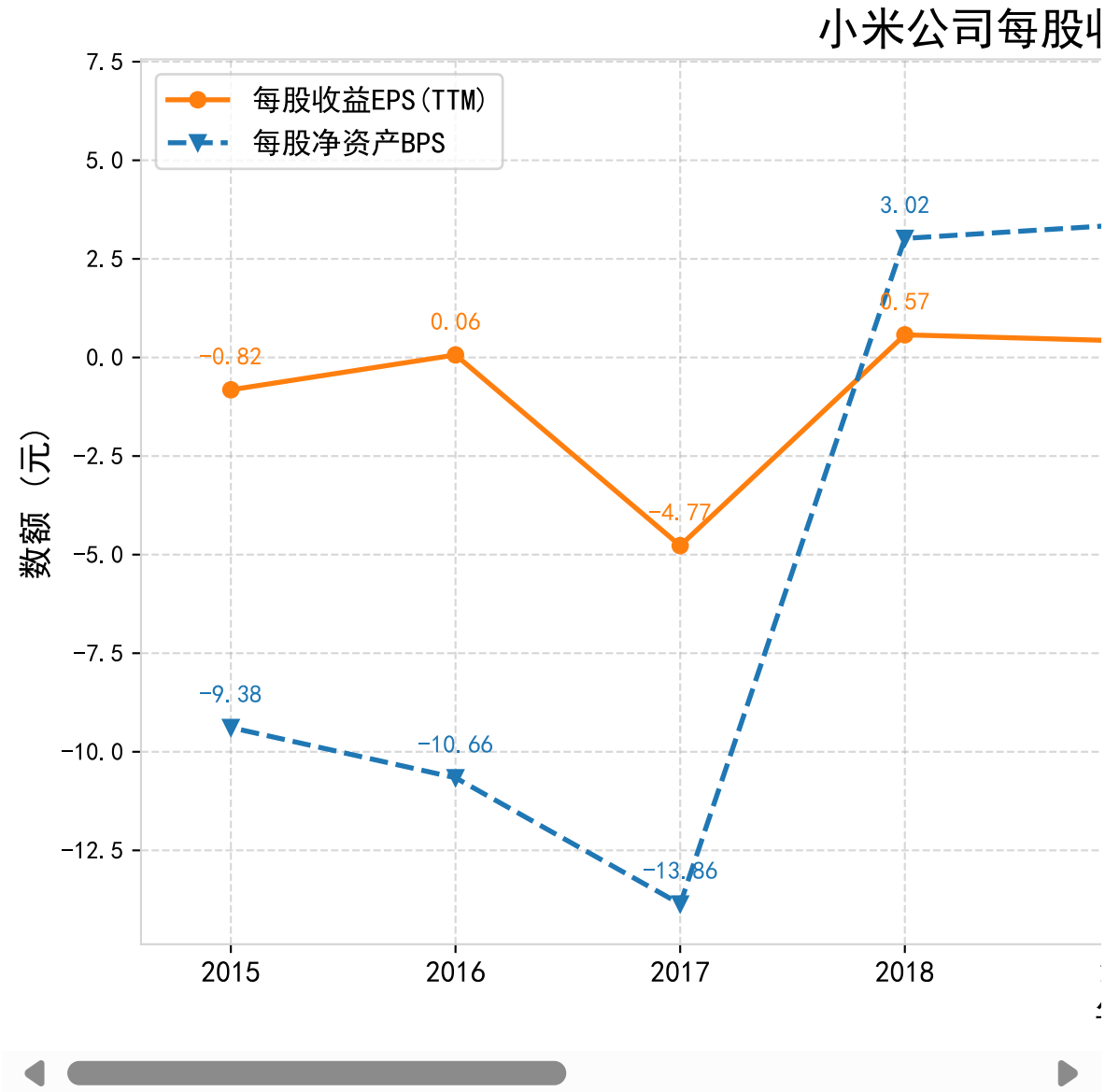
```
ax.grid(True, linestyle='--', alpha=0.5)


for spine in ax.spines.values():
    spine.set_edgecolor('#D5D5D5')


ax.legend(fontsize=12, loc='upper left')


plt.tight_layout()


plt.show()
```

```
Out[62]: [<matplotlib.lines.Line2D at 0x1fd5f55d400>]

Out[62]: [<matplotlib.lines.Line2D at 0x1fd5fca8290>]

Out[62]: Text(0, 10, '0.70')

Out[62]: Text(0, 10, '0.10')

Out[62]: Text(0, 10, '0.77')

Out[62]: Text(0, 10, '0.81')

Out[62]: Text(0, 10, '0.42')

Out[62]: Text(0, 10, '0.57')

Out[62]: Text(0, 10, '-4.77')

Out[62]: Text(0, 10, '0.06')

Out[62]: Text(0, 10, '-0.82')

Out[62]: Text(0, 10, '6.54')

Out[62]: Text(0, 10, '5.76')

Out[62]: Text(0, 10, '5.49')

Out[62]: Text(0, 10, '4.91')

Out[62]: Text(0, 10, '3.37')

Out[62]: Text(0, 10, '3.02')

Out[62]: Text(0, 10, '-13.86')

Out[62]: Text(0, 10, '-10.66')

Out[62]: Text(0, 10, '-9.38')

Out[62]: Text(0.5, 1.0, '小米公司每股收益与净资产分析')

Out[62]: Text(0.5, 0, '年份')

Out[62]: Text(0, 0.5, '数额 (元)')
```

```
Out[62]: [<matplotlib.axis.XTick at 0x1fd5fc6d1f0>,
          <matplotlib.axis.XTick at 0x1fd5fc6c710>,
          <matplotlib.axis.XTick at 0x1fd5fc409e0>,
          <matplotlib.axis.XTick at 0x1fd5fc418e0>,
          <matplotlib.axis.XTick at 0x1fd5fcaaff0>,
          <matplotlib.axis.XTick at 0x1fd5fcab8c0>,
          <matplotlib.axis.XTick at 0x1fd5fcc0260>,
          <matplotlib.axis.XTick at 0x1fd5fcab2c0>,
          <matplotlib.axis.XTick at 0x1fd5fcc0b00>]
```

```
Out[62]: [Text(2023, 0, '2023'),
          Text(2022, 0, '2022'),
          Text(2021, 0, '2021'),
          Text(2020, 0, '2020'),
          Text(2019, 0, '2019'),
          Text(2018, 0, '2018'),
          Text(2017, 0, '2017'),
          Text(2016, 0, '2016'),
          Text(2015, 0, '2015')]
```

```
Out[62]: <matplotlib.legend.Legend at 0x1fd5d22ea50>
```



财务风险分析

偿债能力分析

短期偿债能力

流动比率评估小米短期内能否用流动资产覆盖流动负债的偿债能力。

速动比率剔除存货后更真实地反映小米在紧急情况下的短期流动性安全。

In [63]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

fig, ax = plt.subplots(figsize=(12, 6))

ax.plot(df['年份'], df['流动比率'], marker='o', linestyle='-', color='#4C72B0',
ax.plot(df['年份'], df['速动比率'], marker='s', linestyle='--', color='#55A868',


ax.set_title('小米公司短期偿债能力分析', fontsize=18, fontweight='bold')
ax.set_xlabel('年份', fontsize=14)
ax.set_ylabel('比率', fontsize=14)

for i, txt in enumerate(df['流动比率']):
    ax.annotate(txt, (df['年份'][i], df['流动比率'][i]), textcoords="offset poin

for i, txt in enumerate(df['速动比率']):
    ax.annotate(txt, (df['年份'][i], df['速动比率'][i]), textcoords="offset poin

ax.set_xticks(df['年份'])
ax.set_xticklabels(df['年份'], fontsize=12)

ax.grid(True, linestyle='--', alpha=0.5)


for spine in ax.spines.values():
    spine.set_edgecolor('#D5D5D5')


ax.legend(fontsize=12, loc='upper left')


plt.tight_layout()


plt.show()
```

Out[63]:
```
[<matplotlib.lines.Line2D at 0x1fd602fc8f0>]
```

Out[63]:
```
[<matplotlib.lines.Line2D at 0x1fd5fd29460>]
```

Out[63]:
```
Text(0.5, 1.0, '小米公司短期偿债能力分析')
```

Out[63]:
```
Text(0.5, 0, '年份')
```

Out[63]:
```
Text(0, 0.5, '比率')
```

Out[63]:
```
Text(0, 10, '1.72')
```

Out[63]:
```
Text(0, 10, '1.79')
```

Out[63]: Text(0, 10, '1.61')

Out[63]: Text(0, 10, '1.63')

Out[63]: Text(0, 10, '1.49')

Out[63]: Text(0, 10, '1.71')

Out[63]: Text(0, 10, '1.3')

Out[63]: Text(0, 10, '1.18')

Out[63]: Text(0, 10, '1.52')

Out[63]: Text(0, 10, '1.34')

Out[63]: Text(0, 10, '1.23')

Out[63]: Text(0, 10, '1.15')

Out[63]: Text(0, 10, '1.25')

Out[63]: Text(0, 10, '1.14')

Out[63]: Text(0, 10, '1.24')

Out[63]: Text(0, 10, '0.95')

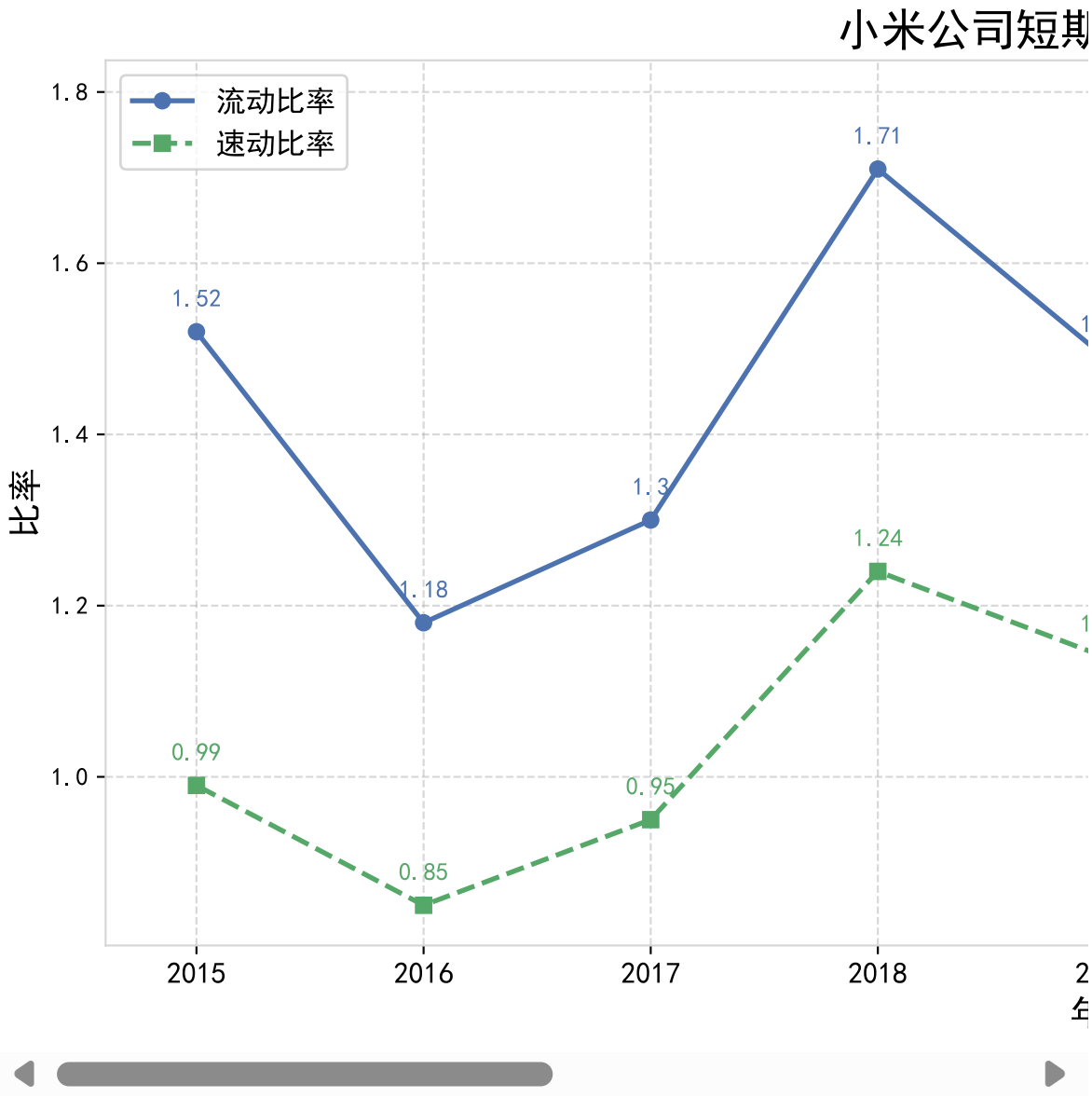
Out[63]: Text(0, 10, '0.85')

Out[63]: Text(0, 10, '0.99')

Out[63]:
```
[<matplotlib.axis.XTick at 0x1fd5fce7470>,
 <matplotlib.axis.XTick at 0x1fd5fca8770>,
 <matplotlib.axis.XTick at 0x1fd5fcaa0f0>,
 <matplotlib.axis.XTick at 0x1fd5fd28050>,
 <matplotlib.axis.XTick at 0x1fd602ff680>,
 <matplotlib.axis.XTick at 0x1fd602fffb0>,
 <matplotlib.axis.XTick at 0x1fd60318800>,
 <matplotlib.axis.XTick at 0x1fd602ff170>,
 <matplotlib.axis.XTick at 0x1fd603181d0>]
```

Out[63]:
```
[Text(2023, 0, '2023'),
 Text(2022, 0, '2022'),
 Text(2021, 0, '2021'),
 Text(2020, 0, '2020'),
 Text(2019, 0, '2019'),
 Text(2018, 0, '2018'),
 Text(2017, 0, '2017'),
 Text(2016, 0, '2016'),
 Text(2015, 0, '2015')]
```

Out[63]: <matplotlib.legend.Legend at 0x1fd602ff440>



现金流量分析

经营现金流揭示小米主营业务是否真正产生"真金白银"的现金回报。

投资现金流反映小米在造车、研发和生态链布局上的资金投入或回收情况。

融资现金流显示小米是否依赖借款或股权融资来支持扩张或回报股东。

期末现金余额衡量小米应对长期投入（如造车）和市场波动的财务底气。

In [64]:
```
fig, ax = plt.subplots(figsize=(12, 6))

ax.plot(df['年份'], df['经营业务现金净额(元)'], marker='o', linestyle='-', color
ax.plot(df['年份'], df['投资业务现金净额(元)'], marker='s', linestyle='--', colo
ax.plot(df['年份'], df['融资业务现金净额(元)'], marker='^', linestyle=':', color
ax.plot(df['年份'], df['期末现金(元)'], marker='*', linestyle='-.', color='#8172

#for col, color in zip(['经营业务现金净额(元)', '投资业务现金净额(元)', '融资业务
    #for i, txt in enumerate(df[col]):
        #ax.annotate(f'{txt/100000000:.2f}亿', (df['年份'][i], df[col][i]), text

ax.set_title('小米公司现金流量及期末现金趋势', fontsize=20, fontweight='bold')
```

```
ax.set_xlabel('年份', fontsize=16)
ax.set_ylabel('现金流量 (万元)', fontsize=16)

ax.set_xticks(df['年份'])
ax.set_xticklabels(df['年份'], fontsize=14)

ax.grid(True, linestyle='--', alpha=0.5, color='lightgray')

ax.legend(fontsize=14, loc='upper left', framealpha=0.8)

plt.tight_layout()

plt.show()
```

Out[64]:
```
[<matplotlib.lines.Line2D at 0x1fd60318b30>]
```

Out[64]:
```
[<matplotlib.lines.Line2D at 0x1fd60361b50>]
```

Out[64]:
```
[<matplotlib.lines.Line2D at 0x1fd60362960>]
```

Out[64]:
```
[<matplotlib.lines.Line2D at 0x1fd60362c00>]
```

Out[64]:
```
Text(0.5, 1.0, '小米公司现金流量及期末现金趋势')
```

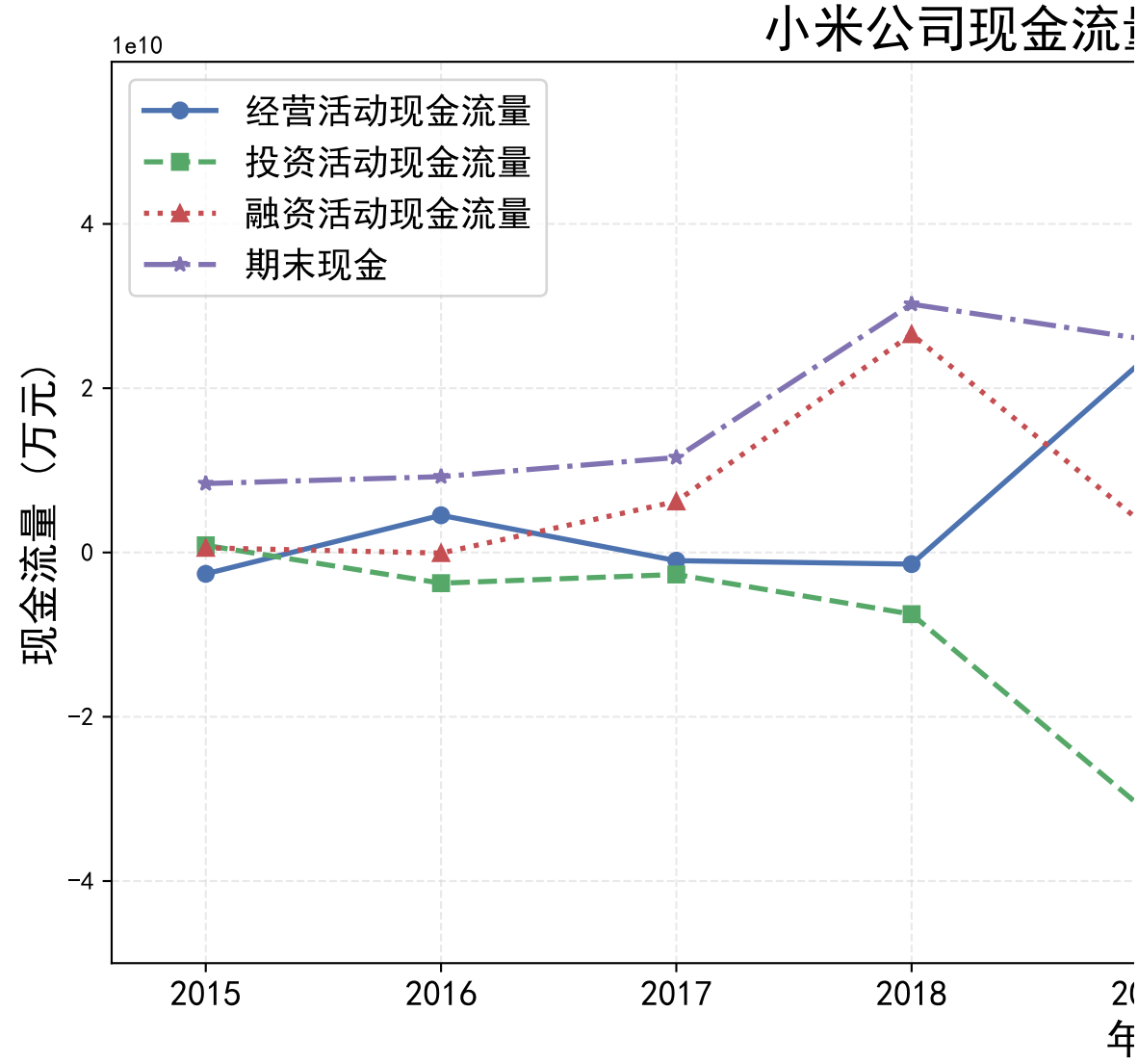
Out[64]:
```
Text(0.5, 0, '年份')
```

Out[64]:
```
Text(0, 0.5, '现金流量 (万元)')
```

Out[64]:
```
[<matplotlib.axis.XTick at 0x1fd60356930>,
 <matplotlib.axis.XTick at 0x1fd60354e60>,
 <matplotlib.axis.XTick at 0x1fd603320f0>,
 <matplotlib.axis.XTick at 0x1fd603318e0>,
 <matplotlib.axis.XTick at 0x1fd60363ce0>,
 <matplotlib.axis.XTick at 0x1fd603630b0>,
 <matplotlib.axis.XTick at 0x1fd603a0830>,
 <matplotlib.axis.XTick at 0x1fd603a1160>,
 <matplotlib.axis.XTick at 0x1fd603a19a0>]
```

Out[64]:
```
[Text(2023, 0, '2023'),
 Text(2022, 0, '2022'),
 Text(2021, 0, '2021'),
 Text(2020, 0, '2020'),
 Text(2019, 0, '2019'),
 Text(2018, 0, '2018'),
 Text(2017, 0, '2017'),
 Text(2016, 0, '2016'),
 Text(2015, 0, '2015')]
```

Out[64]:
```
<matplotlib.legend.Legend at 0x1fd602fd3d0>
```



盈利能力分析

利润收入分析

分析毛利可判断小米核心硬件业务的盈利能力和成本控制水平。

营运收入反映小米主营业务在扣除运营费用后的实际盈利能力。

利息收入体现小米对账上巨额现金的利用效率和财务收益能力。

In [65]:
```
import pandas as pd
import matplotlib.pyplot as plt

fig, ax = plt.subplots(figsize=(12, 6))

ax.plot(df['年份'], df['毛利(元)'], marker='o', linestyle='-', color='#4C72B0',
ax.plot(df['年份'], df['营运收入(元)'], marker='s', linestyle='--', color='#55A8
ax.plot(df['年份'], df['利息收入(元)'], marker='^', linestyle=':', color='#C44E5

ax.set_title('小米公司利润收入分析 (2018-2022)', fontsize=20, fontweight='bold')
ax.set_xlabel('年份', fontsize=16)
ax.set_ylabel('数额 ', fontsize=16)
```

```
ax.set_xticks(df['年份'])
ax.set_xticklabels(df['年份'], fontsize=14)


ax.grid(True, linestyle='--', alpha=0.5, color='lightgray')


ax.legend(fontsize=14, loc='upper left', framealpha=0.8)


plt.tight_layout()


plt.show()
```

Out[65]:
```
[<matplotlib.lines.Line2D at 0x1fd60997320>]
```

Out[65]:
```
[<matplotlib.lines.Line2D at 0x1fd60997380>]
```

Out[65]:
```
[<matplotlib.lines.Line2D at 0x1fd60997500>]
```

Out[65]:
```
Text(0.5, 1.0, '小米公司利润收入分析 (2018-2022)')
```

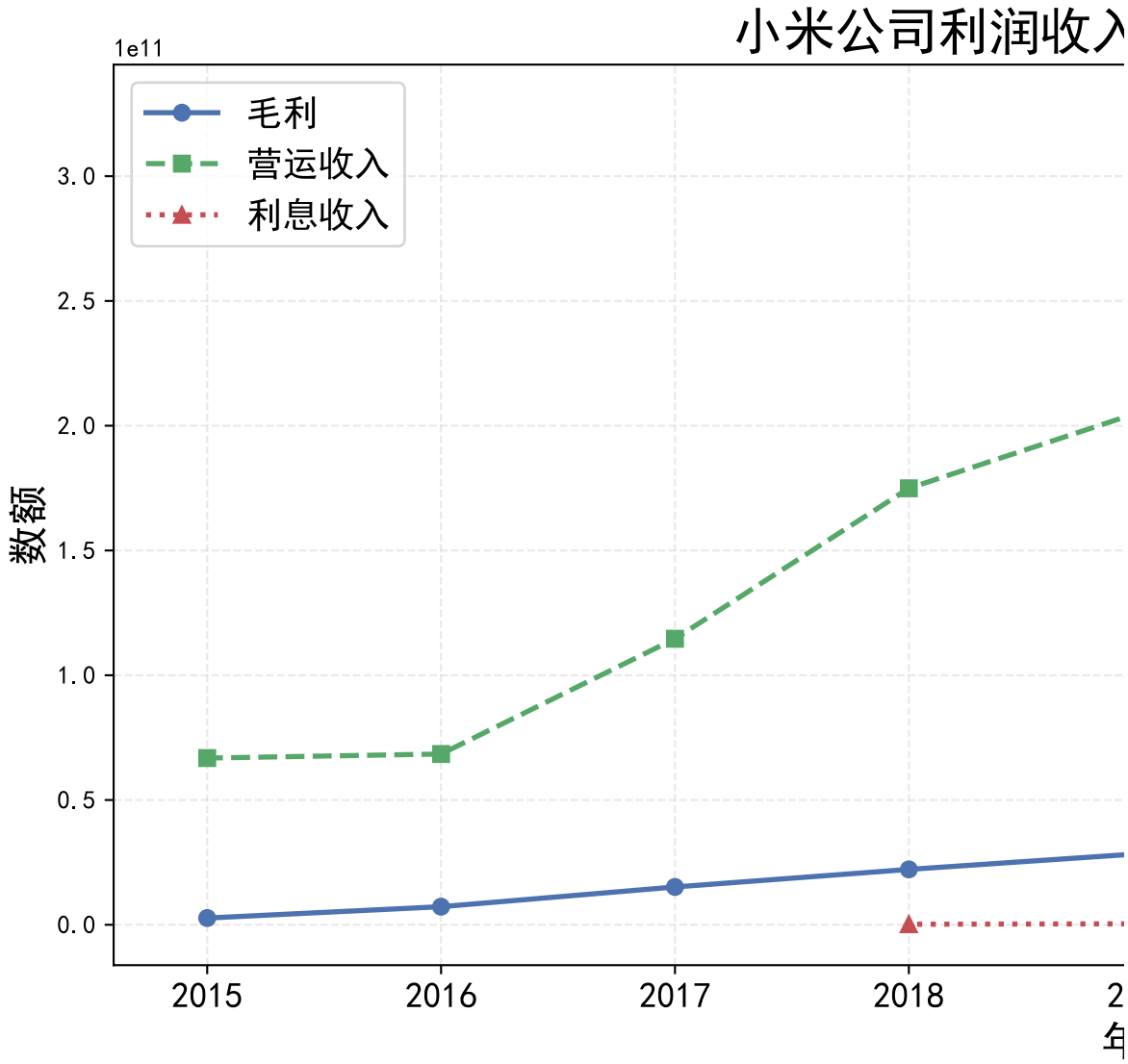
Out[65]:
```
Text(0.5, 0, '年份')
```

Out[65]:
```
Text(0, 0.5, '数额 ')
```

Out[65]:
```
[<matplotlib.axis.XTick at 0x1fd602fe9c0>,
 <matplotlib.axis.XTick at 0x1fd60994050>,
 <matplotlib.axis.XTick at 0x1fd602fd040>,
 <matplotlib.axis.XTick at 0x1fd60997c50>,
 <matplotlib.axis.XTick at 0x1fd609d47d0>,
 <matplotlib.axis.XTick at 0x1fd609d4b00>,
 <matplotlib.axis.XTick at 0x1fd609d5400>,
 <matplotlib.axis.XTick at 0x1fd609d5d30>,
 <matplotlib.axis.XTick at 0x1fd609d6660>]
```

Out[65]:
```
[Text(2023, 0, '2023'),
 Text(2022, 0, '2022'),
 Text(2021, 0, '2021'),
 Text(2020, 0, '2020'),
 Text(2019, 0, '2019'),
 Text(2018, 0, '2018'),
 Text(2017, 0, '2017'),
 Text(2016, 0, '2016'),
 Text(2015, 0, '2015')]
```

Out[65]:
```
<matplotlib.legend.Legend at 0x1fd60997ad0>
```

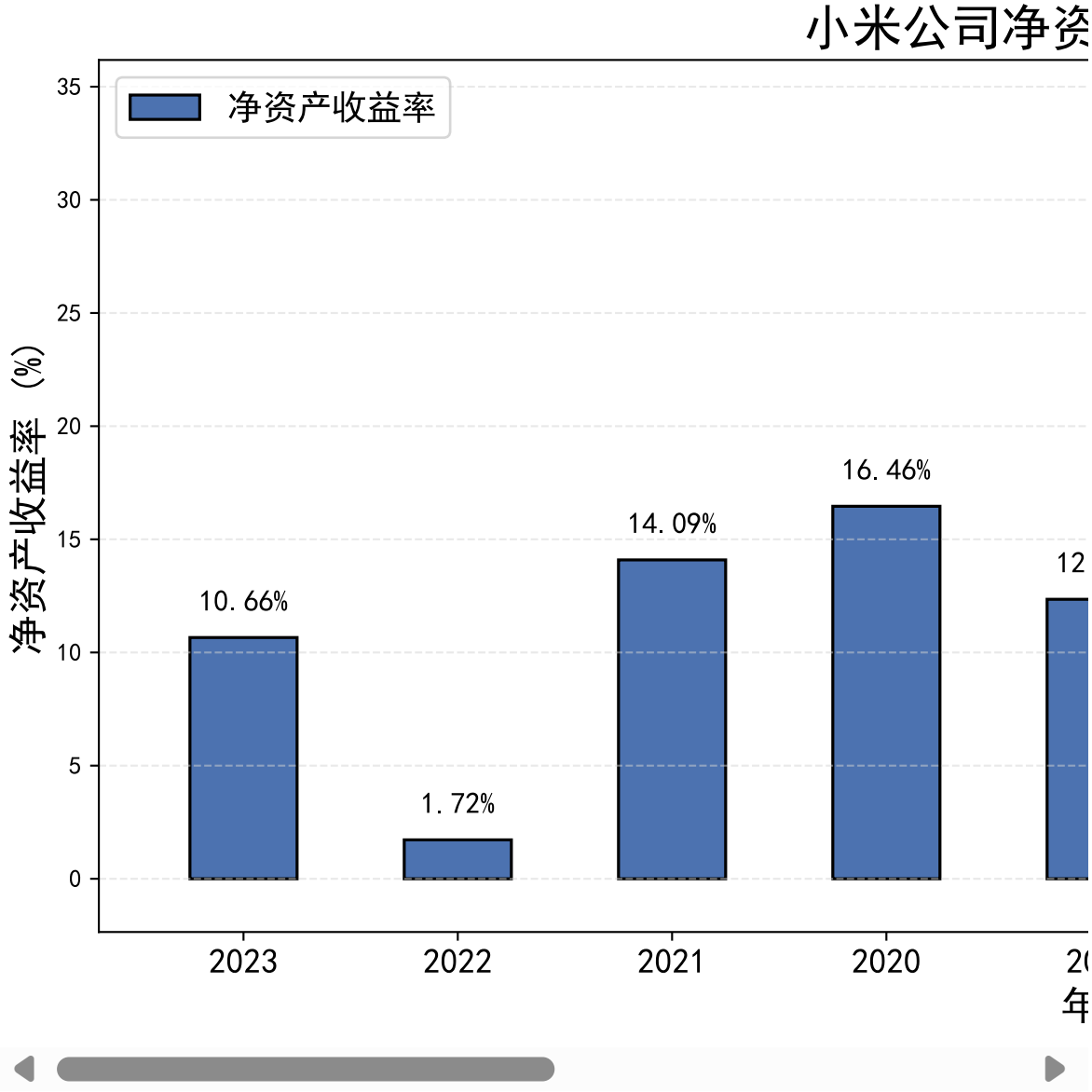


净资产收益率

In [66]:
```
fig, ax = plt.subplots(figsize=(12, 6))

bar_width = 0.5
index = np.arange(len(df['年份']))

ax.bar(index, df['净资产收益率(TTM)(%)'], bar_width, color='#4C72B0', edgecolor=


for i, txt in enumerate(df['净资产收益率(TTM)(%)']):
    ax.text(i, txt + 1, f'{txt}%', ha='center', va='bottom', fontsize=12, color=

ax.set_title('小米公司净资产收益率分析', fontsize=20, fontweight='bold')
ax.set_xlabel('年份', fontsize=16)
ax.set_ylabel('净资产收益率 (%)', fontsize=16)

ax.set_xticks(index)
ax.set_xticklabels(df['年份'], fontsize=14)

ax.grid(True, axis='y', linestyle='--', alpha=0.5, color='lightgray')

ax.legend(fontsize=14, loc='upper left', framealpha=0.8)
```

```
plt.tight_layout()

plt.show()
```

```
Out[66]: <BarContainer object of 9 artists>

Out[66]: Text(0, 11.66, '10.66%')

Out[66]: Text(1, 2.7199999999999998, '1.72%')

Out[66]: Text(2, 15.09, '14.09%')

Out[66]: Text(3, 17.46, '16.46%')

Out[66]: Text(4, 13.35, '12.35%')

Out[66]: Text(5, 20.0, '19.0%')

Out[66]: Text(6, 35.43, '34.43%')

Out[66]: Text(7, 0.4, '-0.6%')

Out[66]: Text(8, 9.74, '8.74%')

Out[66]: Text(0.5, 1.0, '小米公司净资产收益率分析')

Out[66]: Text(0.5, 0, '年份')

Out[66]: Text(0, 0.5, '净资产收益率 (%)')

Out[66]: [<matplotlib.axis.XTick at 0x1fd60a1ca10>,
          <matplotlib.axis.XTick at 0x1fd60a1c1d0>,
          <matplotlib.axis.XTick at 0x1fd609d76e0>,
          <matplotlib.axis.XTick at 0x1fd60356f00>,
          <matplotlib.axis.XTick at 0x1fd60a5ede0>,
          <matplotlib.axis.XTick at 0x1fd60a5f710>,
          <matplotlib.axis.XTick at 0x1fd60a5ffe0>,
          <matplotlib.axis.XTick at 0x1fd60a5f410>,
          <matplotlib.axis.XTick at 0x1fd60d5c7d0>]

Out[66]: [Text(0, 0, '2023'),
          Text(1, 0, '2022'),
          Text(2, 0, '2021'),
          Text(3, 0, '2020'),
          Text(4, 0, '2019'),
          Text(5, 0, '2018'),
          Text(6, 0, '2017'),
          Text(7, 0, '2016'),
          Text(8, 0, '2015')]

Out[66]: <matplotlib.legend.Legend at 0x1fd603a0170>
```



导出数据

In [67]:
```
df.to_excel("data/df.xlsx",index=False)
```