In [32]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
%matplotlib inline
```

读取数据

In [2]:
```
data = pd.read_excel('./consumption_data.xls', sheet_name='Sheet1')
data.head()
```

Out[2]:

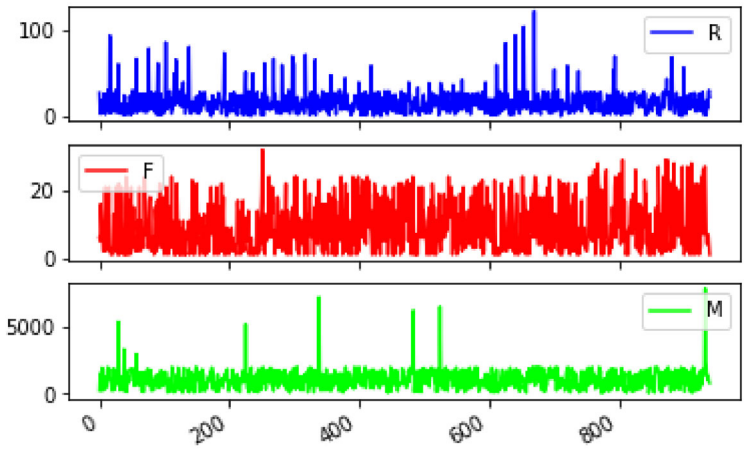
	Id	R	F	M
0	1	27	6	232.61
1	2	3	5	1507.11
2	3	4	16	817.62
3	4	3	11	232.81
4	5	14	7	1913.05

数据可视化

In [3]:
```
1 data[['R', 'F', 'M']].plot(cmap='brg', subplots=True)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\pandas\plotting\_tools.py:308: MatplotlibDeprecation
Warning:
The rowNum attribute was deprecated in Matplotlib 3.2 and will be removed two minor releases la
ter. Use ax.get_subplotspec().rowspan.start instead.
  layout[ax.rowNum, ax.colNum] = ax.get_visible()
C:\ProgramData\Anaconda3\lib\site-packages\pandas\plotting\_tools.py:308: MatplotlibDeprecation
Warning:
The colNum attribute was deprecated in Matplotlib 3.2 and will be removed two minor releases la
ter. Use ax.get_subplotspec().colspan.start instead.
  layout[ax.rowNum, ax.colNum] = ax.get_visible()
C:\ProgramData\Anaconda3\lib\site-packages\pandas\plotting\_tools.py:314: MatplotlibDeprecation
Warning:
The rowNum attribute was deprecated in Matplotlib 3.2 and will be removed two minor releases la
ter. Use ax.get_subplotspec().rowspan.start instead.
  if not layout[ax.rowNum + 1, ax.colNum]:
C:\ProgramData\Anaconda3\lib\site-packages\pandas\plotting\_tools.py:314: MatplotlibDeprecation
Warning:
The colNum attribute was deprecated in Matplotlib 3.2 and will be removed two minor releases la
ter. Use ax.get_subplotspec().colspan.start instead.
  if not layout[ax.rowNum + 1, ax.colNum]:
C:\ProgramData\Anaconda3\lib\site-packages\pandas\plotting\_tools.py:308: MatplotlibDeprecation
Warning:
The rowNum attribute was deprecated in Matplotlib 3.2 and will be removed two minor releases la
ter. Use ax.get_subplotspec().rowspan.start instead.
  layout[ax.rowNum, ax.colNum] = ax.get_visible()
C:\ProgramData\Anaconda3\lib\site-packages\pandas\plotting\_tools.py:308: MatplotlibDeprecation
Warning:
The colNum attribute was deprecated in Matplotlib 3.2 and will be removed two minor releases la
ter. Use ax.get_subplotspec().colspan.start instead.
  layout[ax.rowNum, ax.colNum] = ax.get_visible()
C:\ProgramData\Anaconda3\lib\site-packages\pandas\plotting\_tools.py:314: MatplotlibDeprecation
Warning:
The rowNum attribute was deprecated in Matplotlib 3.2 and will be removed two minor releases la
ter. Use ax.get_subplotspec().rowspan.start instead.
  if not layout[ax.rowNum + 1, ax.colNum]:
C:\ProgramData\Anaconda3\lib\site-packages\pandas\plotting\_tools.py:314: MatplotlibDeprecation
Warning:
The colNum attribute was deprecated in Matplotlib 3.2 and will be removed two minor releases la
ter. Use ax.get_subplotspec().colspan.start instead.
  if not layout[ax.rowNum + 1, ax.colNum]:
```

Out[3]: array([<AxesSubplot:>, <AxesSubplot:>, <AxesSubplot:>], dtype=object)



数据预处理

In [7]:

```
data = data.dropna()  # 去除空值
s_data = (data[['R', 'F','M']]-data[['R', 'F','M']].mean(axis=0))/(data[['R', 'F','M']].std(
data.head()
s_data['id'] = data['Id']
s_data
```

Out[7]:

	R	F	M	id
0	0.764186	-0.493579	-1.158711	1
1	-1.024757	-0.630079	0.622527	2
2	-0.950217	0.871423	-0.341103	3
3	-1.024757	0.188922	-1.158432	4
4	-0.204824	-0.357079	1.189868	5
5	0.167872	-0.493579	-1.176237	6
6	-0.875678	-1.039580	-0.623124	7
7	0.689647	-1.039580	-0.002828	8
8	0.316951	-0.084078	-1.057791	9
9	-1.099296	1.553924	0.232388	10
10	-0.130285	-1.039580	-0.755630	11
11	0.689647	-0.903079	-0.871351	12
12	0.018793	0.188922	0.954373	13
13	0.987804	0.871423	1.251907	14
14	-0.875678	-0.357079	0.911383	15
15	-0.950217	1.553924	0.987300	16
16	5.683780	-1.039580	-0.063372	17
17	-0.055746	-0.903079	-0.155585	18
18	-0.950217	-1.176080	-0.428718	19
19	0.764186	-1.176080	-1.072591	20
20	-0.875678	-1.176080	-1.210856	21
21	0.018793	-0.903079	1.095237	22
22	-0.353903	0.461922	0.520754	23
23	0.316951	-0.903079	-1.098279	24
24	0.093333	-0.630079	-0.855223	25
25	0.987804	1.553924	0.792433	26
26	-0.950217	-1.039580	1.025455	27
27	-0.726600	0.325422	1.012639	28
28	0.093333	-1.176080	-0.534223	29
29	3.223983	-0.357079	5.949751	30
...	...	...	...	...
910	-0.577521	0.598423	0.790351	913
911	-0.428442	2.509425	0.130000	914
912	0.018793	-1.176080	-1.436875	915
913	-0.130285	-0.084078	1.254982	916
914	-0.502982	1.690424	0.725991	917
915	0.764186	-0.630079	1.143426	918

	R	F	M	id
916	0.615108	-0.903079	0.112809	919
917	-0.130285	-1.176080	-1.239730	920
918	-0.875678	-1.039580	-0.591007	921
919	0.167872	-0.766579	0.008521	922
920	-0.204824	1.690424	0.397248	923
921	-0.801139	-0.903079	0.348528	924
922	0.689647	-0.220578	-0.138451	925
923	0.466029	-1.176080	-1.084135	926
924	0.018793	1.007923	0.824578	927
925	-0.055746	-0.903079	0.617999	928
926	-0.801139	0.461922	0.621647	929
927	0.391490	1.963424	-0.610140	930
928	0.838726	1.007923	0.952137	931
929	0.018793	2.236425	0.322183	932
930	-0.055746	-0.357079	1.033798	933
931	-1.173835	2.372925	0.731526	934
932	0.018793	2.372925	9.410511	935
933	-1.099296	1.007923	1.789270	936
934	-1.173835	-0.630079	1.123804	937
935	0.167872	-0.766579	0.141712	938
936	-0.577521	-0.357079	-0.076342	939
937	0.764186	-0.357079	0.365132	940
938	0.987804	-0.766579	-0.281299	941
939	0.391490	-1.176080	-0.398292	942

940 rows × 4 columns

In [13]:
```
kmodel = KMeans(n_clusters=10, n_jobs=8)  # n_jobs是并行数，一般等于CPU数较好
kmodel.fit(s_data[['R', 'F', 'M']])  ##训练模型
```

Out[13]:
```
KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
    n_clusters=10, n_init=10, n_jobs=8, precompute_distances='auto',
    random_state=None, tol=0.0001, verbose=0)
```

In [27]:
```
label = pd.Series(kmodel.labels_)  # 各样本的类别
num = pd.Series(kmodel.labels_).value_counts()  # 统计各样本对应的类别的数目
center = pd.DataFrame(kmodel.cluster_centers_)  # 找出聚类中心
center_max = center.values.max()
center_min = center.values.min()
```

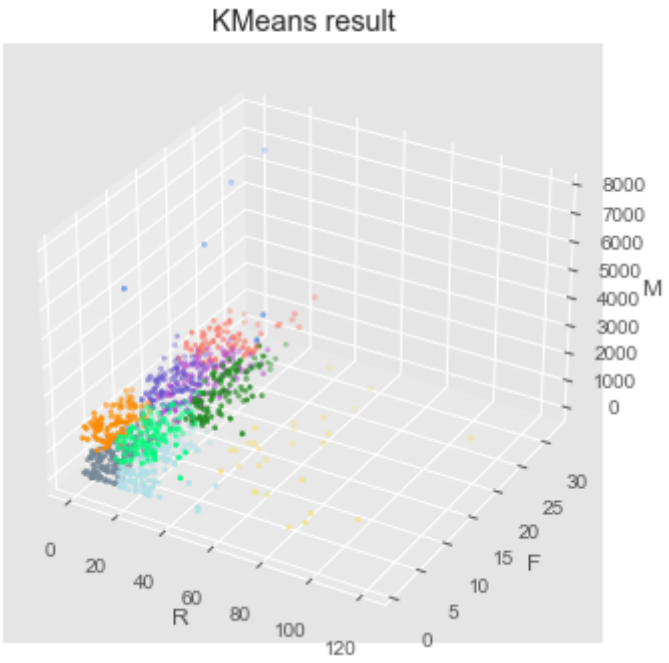
In [38]:
```
from mpl_toolkits.mplot3d import Axes3D  # 用于绘制3D撒黏土
```

In [59]:

```
result = data
result['c_r'] = kmodel.labels_  # 将聚类类别序号写入数据中

# plt.style.use('seaborn')
plt.style.use('ggplot')  # plt使用色彩风格
fig = plt.figure()  # 生成plt对象
c_list= ['#6A5ACD','#B0E0E6','#778899', '#00FF7F','#228B22', '#F0E68C'
        ,'#FF8C00','#FA8072','#6495ED','#BA55D3']  # 色值列表
ax = fig.add_subplot(111, projection='3d')  # 创建3D图像底版
for i in range(0, 10):  # 提取每一类别的数据并绘图
    c_data = result[result['c_r']==i]
    x0 = c_data['R'].values  # 转为array
    y0 = c_data['F'].values
    z0 = c_data['M'].values
    color = c_list[i]  # 选择颜色
    ax.scatter(x0, y0, z0, c=color, marker='.')  # 绘制3D散点图
ax.set_xlabel('R')
ax.set_ylabel('F')
ax.set_zlabel('M')
plt.title('KMeans result')
fig = plt.gcf()  # 捕捉当前图像信息，用于保存图片
plt.show()

fig.savefig('clustering.png', dpi=800)  # 保存图片
```



In [ ]:

```
1
```