analise-descritiva

April 1, 2019

1 Análise descritiva

In [19]:
```
%matplotlib inline
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import geopandas as gp
import pysal as ps
import palettable
import nupis
import mapclassify as mc
import matplotlib.lines as mlines
import pysal.contrib.viz.mapping as maps
import pickle
import statsmodels.formula.api as smf
plt.style.use('seaborn-whitegrid')

plt.rc('figure', max_open_warning = 50)
pd.options.display.max_rows = 1000

# dicionário com os dataframes dos anos
temp = open('../dados/separado_rgi.pickle', 'rb')
separado_rgi = pickle.load(temp)

# informações sobre MG - geo e dados
mg = pd.read_csv('../dados/mg-mapas.csv')
mg = gp.GeoDataFrame(mg)   # transforma em geopandas
```

In [18]:
```
mg.head()
```

Out[18]:

	NM_MUNICIP	mun	\
0	ABADIA DOS DOURADOS	3100104	
1	ABADIA DOS DOURADOS	3100104	
2	ABADIA DOS DOURADOS	3100104	
3	ABADIA DOS DOURADOS	3100104	
4	ABADIA DOS DOURADOS	3100104	

```
                                                geometry            Município  \
0  POLYGON ((-47.429672447 -18.16543081755956, -4...  Abadia dos Dourados
1  POLYGON ((-47.429672447 -18.16543081755956, -4...  Abadia dos Dourados
2  POLYGON ((-47.429672447 -18.16543081755956, -4...  Abadia dos Dourados
3  POLYGON ((-47.429672447 -18.16543081755956, -4...  Abadia dos Dourados
4  POLYGON ((-47.429672447 -18.16543081755956, -4...  Abadia dos Dourados

    ano  area_colhida  area_per  rend  meso  \
0  2002            76      35.2   789  3105
1  2003            96      77.4   781  3105
2  2004            96      78.0   895  3105
3  2005            74      72.5  1324  3105
4  2006            74      73.3  1689  3105

                          nome_meso  micro  nome_micro  cod_rgi  \
0  Triângulo Mineiro/Alto Paranaíba  31019  Patrocínio   310061
1  Triângulo Mineiro/Alto Paranaíba  31019  Patrocínio   310061
2  Triângulo Mineiro/Alto Paranaíba  31019  Patrocínio   310061
3  Triângulo Mineiro/Alto Paranaíba  31019  Patrocínio   310061
4  Triângulo Mineiro/Alto Paranaíba  31019  Patrocínio   310061

        nome_rgi  cod_rgint  nome_rgint  producao
0  Monte Carmelo       3111  Uberlândia     59964
1  Monte Carmelo       3111  Uberlândia     74976
2  Monte Carmelo       3111  Uberlândia     85920
3  Monte Carmelo       3111  Uberlândia     97976
4  Monte Carmelo       3111  Uberlândia    124986
```

Divisão por regiões geográficas imediatas
Estatísticas descritivas

1.1 Produtividade

```
In [21]: medi=[]
         desvio=[]
         mini=[]
         vinteecinco=[]
         cinquenta=[]
         setentaecinco=[]
         maximo=[]
         for i in range (2002,2018):
             medi.append(round(separado_rgi[i].rendimento.mean(),2))
             desvio.append(round(separado_rgi[i].rendimento.std(),2))
             mini.append(round(separado_rgi[i].rendimento.min(),2))
             vinteecinco.append(round(separado_rgi[i].rendimento.quantile(0.25),2))
             cinquenta.append(round(separado_rgi[i].rendimento.median(),2))
             setentaecinco.append(round(separado_rgi[i].rendimento.quantile(0.75),2))
             maximo.append(round(separado_rgi[i].rendimento.max(),2))
```

```
In [22]: descritivas=pd.DataFrame({'ano':list(range(2002,2018)),'média':medi,'desvio padrão':des
                                   'mín':mini,'$q_{0,25}$':vinteecinco,'$q_{0,50}$':cinquenta,'$q_
         descritivas
```

Out[22]:

	$q_{0,25}$	$q_{0,50}$	$q_{0,75}$	ano	desvio padrão	máx	média	\
0	710.69	999.98	1238.12	2002	542.33	2866.10	1025.35	
1	660.23	866.48	1084.82	2003	409.19	2200.00	916.67	
2	878.36	1034.40	1239.48	2004	499.10	3057.69	1106.10	
3	784.71	999.99	1185.97	2005	508.68	3425.00	1071.16	
4	840.24	1096.16	1447.11	2006	616.48	4000.00	1241.51	
5	799.60	969.79	1130.25	2007	487.15	2638.86	1061.82	
6	1000.51	1120.11	1433.52	2008	574.50	3734.17	1243.36	
7	986.13	1079.42	1294.82	2009	568.00	3491.24	1195.08	
8	1008.05	1122.58	1575.06	2010	698.93	3871.42	1307.11	
9	994.14	1151.67	1410.18	2011	623.75	3832.00	1284.83	
10	1143.64	1336.75	1751.76	2012	659.20	3993.97	1426.60	
11	1145.30	1432.11	1632.95	2013	634.88	3967.54	1420.03	
12	968.64	1183.35	1640.84	2014	650.23	3632.93	1312.20	
13	1100.99	1232.41	1422.57	2015	505.72	2710.58	1260.02	
14	1198.09	1550.85	1884.10	2016	632.61	2793.19	1501.88	
15	1082.86	1479.77	1782.05	2017	637.61	2547.43	1382.23	

	mín
0	0.0
1	0.0
2	0.0
3	0.0
4	0.0
5	0.0
6	0.0
7	0.0
8	0.0
9	0.0
10	0.0
11	0.0
12	0.0
13	0.0
14	0.0
15	0.0

```
In [4]: descritivas.to_latex()
```

```
Out[4]: '\\begin{tabular}{lrrrrrrrr}\n\\toprule\n{} &   ano &    média &  desvio padrão &  mín &
```

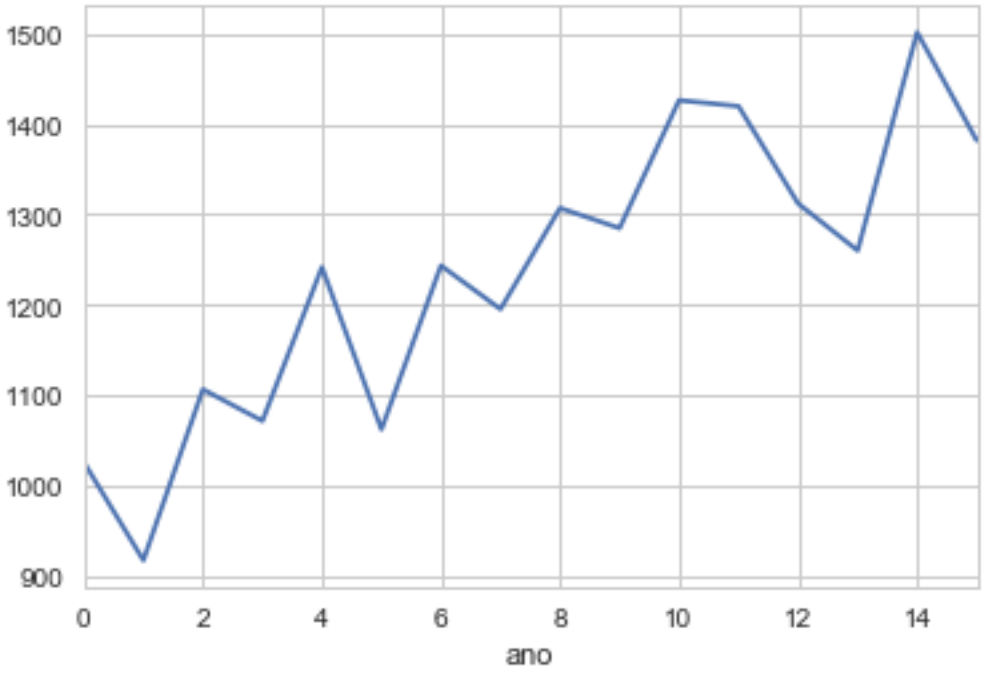
Considerando a divisão em regiões geográficas imediatas, aparentemente a média também está aumentando, indicando melhoria na produtividade em geral. O terceiro quartil também se aproxima valor máximo com o passar dos anos. Pode ser feita modelagem via MQO para verificar essa questão.

In [23]:
```
medias = descritivas['média']
a = np.stack([np.arange(0, 16), medias], axis=1)
df = pd.DataFrame(a, columns=['ano', 'dados'])
smf.ols('dados ~ ano', data=df).fit().summary()
```

```
C:\Users\Renan\Anaconda3\lib\site-packages\scipy\stats\stats.py:1334: UserWarning: kurtosistest
  "anyway, n=%i" % int(n))
```

Out[23]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                  dados   R-squared:                       0.748
Model:                            OLS   Adj. R-squared:                  0.730
Method:                 Least Squares   F-statistic:                     41.52
Date:                Thu, 28 Feb 2019   Prob (F-statistic):           1.54e-05
Time:                        11:50:11   Log-Likelihood:                -92.666
No. Observations:                  16   AIC:                             189.3
Df Residuals:                      14   BIC:                             190.9
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept   1012.6874     40.452     25.034      0.000     925.926    1099.449
ano           29.6079      4.595      6.443      0.000      19.753      39.463
==============================================================================
Omnibus:                        1.720   Durbin-Watson:                   2.566
Prob(Omnibus):                  0.423   Jarque-Bera (JB):                0.994
Skew:                          -0.234   Prob(JB):                        0.608
Kurtosis:                       1.872   Cond. No.                         17.0
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specif
"""
```

In [25]:
```
df.index = df.ano
df = df.loc[:, 'dados']
df.plot();
```

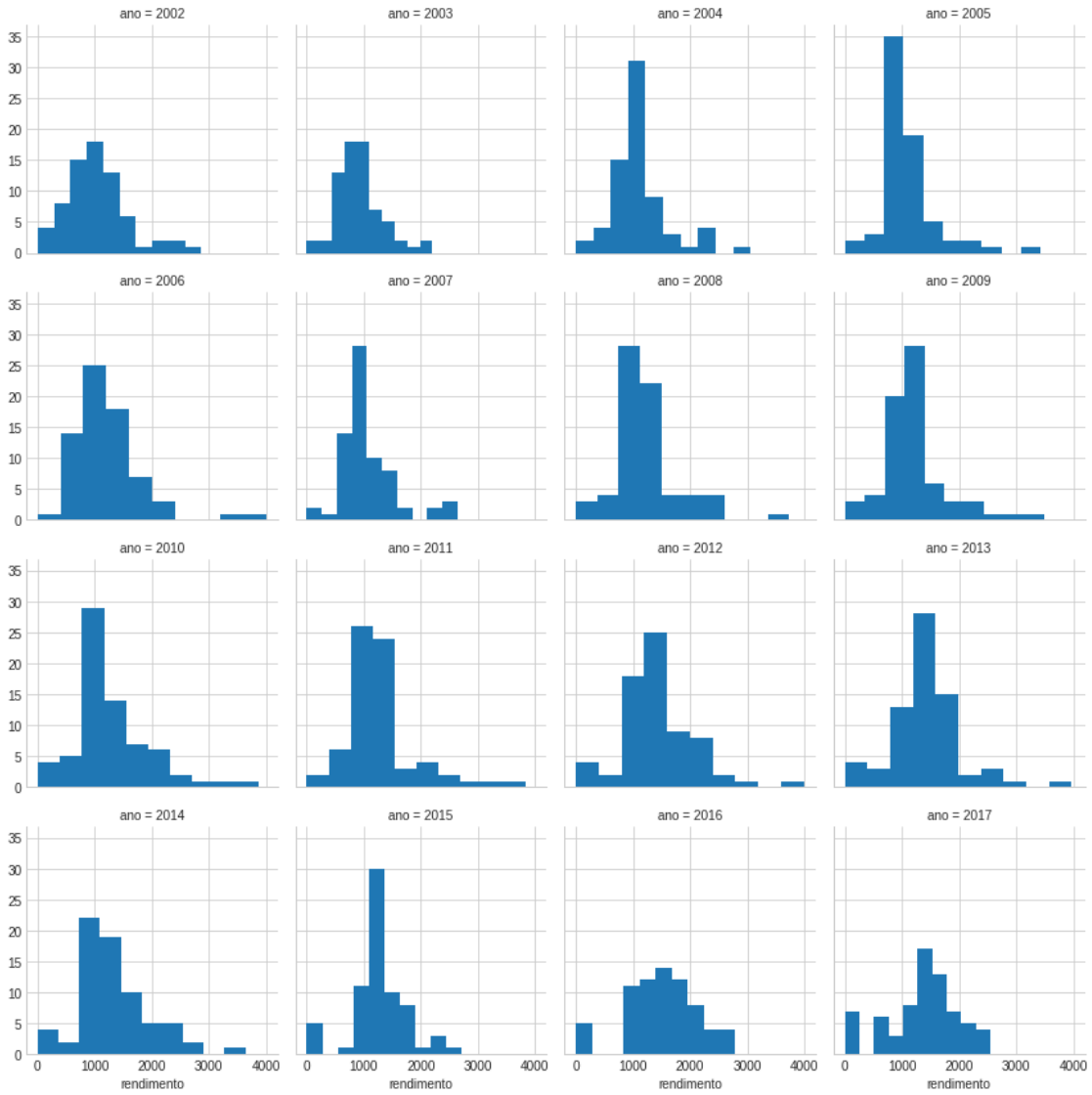


Assim, a equação para as médias seria $Y = 1013 + 29,6X$ com $r = 0,86$ e o teste F foi significativo $(1,54e - 05)$. Assim, percebe-se tendência de aumento dos valores das médias de produtividade ao longo dos anos.

1.1.1 Histogramas

```
In [26]: # ir concatenando as linhas dos dataframes em um só dataframe para gerar os gráficos
         n = len(mg.ano.unique())
         anos = mg.ano.unique()
         for j in range(n):
             separado_rgi[anos[j]] = separado_rgi[anos[j]].reset_index()
         df_rgi = pd.concat([separado_rgi[2002], separado_rgi[2003]], axis=0) # os dois prmeiros
         for i in range(2, n):   # terceiro ano em diante
             df_rgi = pd.concat([df_rgi, separado_rgi[anos[i]]], axis=0)
```

```
In [10]: # histogramas dos rendimentos por ano
         g = sns.FacetGrid(df_rgi, col='ano', col_wrap=4)
         g.map(plt.hist, 'rendimento', bins=10)
         plt.show()
         plt.savefig('hist_rend.png' , bbox_inches='tight');
```




```
<Figure size 432x288 with 0 Axes>
```

Pelos histogramas da produtividade das microrregiões, a impressão também é de que a média está aumentando.

1.1.2 Boxplots

```
In [7]: df_rgi2=df_rgi
        df_rgi2.rename(columns={'rendimento': 'produtividade'}, inplace=True)
        fig, ax = plt.subplots(figsize=(10,7))
        sns.boxplot(ax=ax, x='produtividade', y='ano', orient='h', color='gray', data=df_rgi2);
```

```
---------------------------------------------------------------------------

NameError                                 Traceback (most recent call last)

<ipython-input-7-e73a2e2062f0> in <module>()
----> 1 df_rgi2=df_rgi
      2 df_rgi2.rename(columns={'rendimento': 'produtividade'}, inplace=True)
      3 fig, ax = plt.subplots(figsize=(10,7))
      4 sns.boxplot(ax=ax, x='produtividade', y='ano', orient='h', color='gray', data=df_rgi


NameError: name 'df_rgi' is not defined
```

1.2 Área

In [27]:
```
mediarea=[]
desvioarea=[]
miniarea=[]
vinteecincoarea=[]
cinquentaarea=[]
setentaecincoarea=[]
maximoarea=[]
for i in range (2002,2018):
    mediarea.append(separado_rgi[i].area_colhida.mean())
    desvioarea.append(separado_rgi[i].area_colhida.std())
    miniarea.append(separado_rgi[i].area_colhida.min())
    vinteecincoarea.append(separado_rgi[i].area_colhida.quantile(0.25))
    cinquentaarea.append(separado_rgi[i].area_colhida.median())
    setentaecincoarea.append(separado_rgi[i].area_colhida.quantile(0.75))
    maximoarea.append(separado_rgi[i].area_colhida.max())
descritivasarea=pd.DataFrame({'ano':list(range(2002,2018)),'média':mediarea,'desvio pad
                        'mín':miniarea,'$q_{0,25}$':vinteecincoarea,'$q_{0,50}$':cinque
                              '$q_{0,75}$':setentaecincoarea,'máx':maximoarea})
descritivasarea
```

Out[27]:

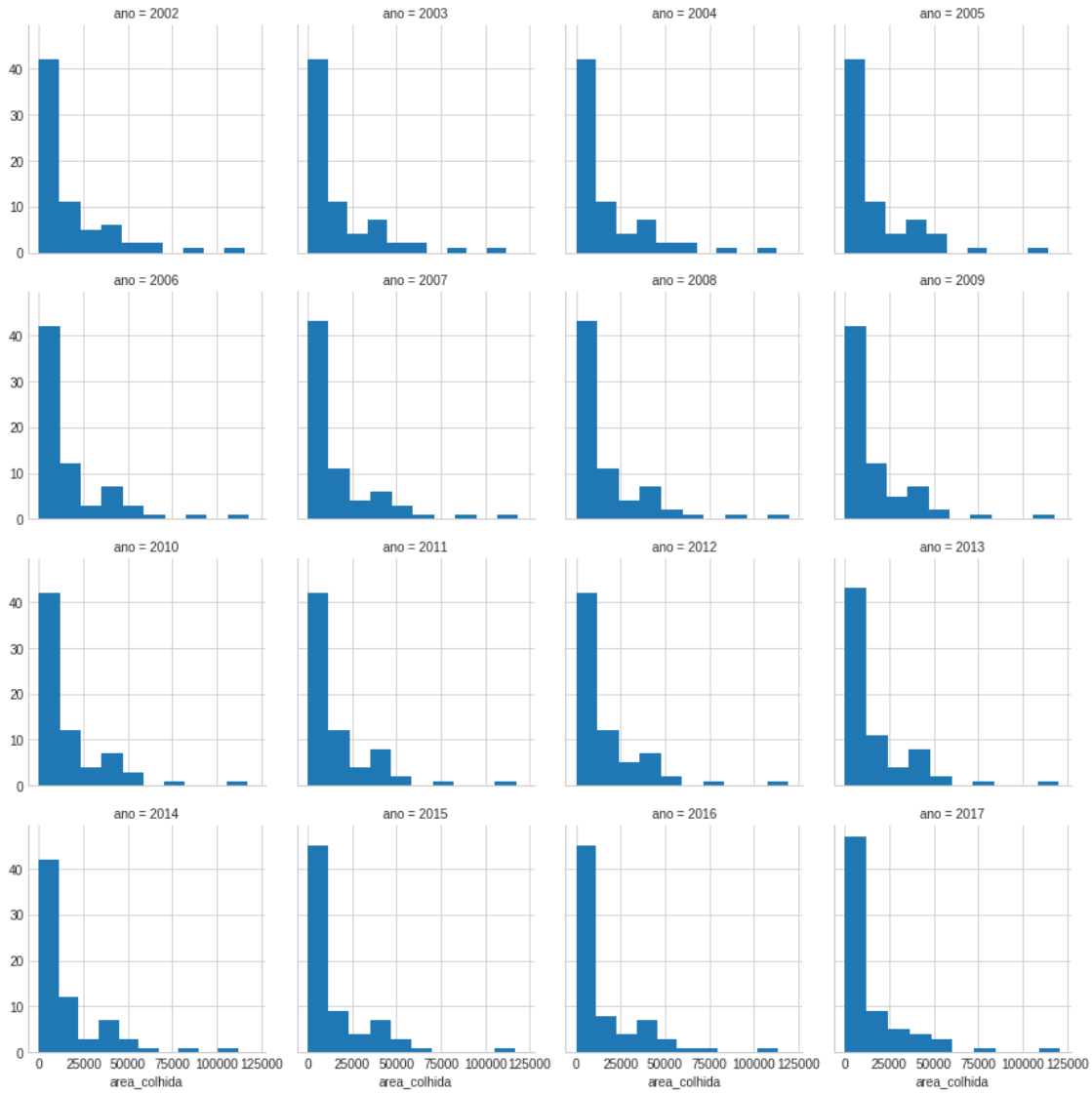
	$q_{0,25}$	$q_{0,50}$	$q_{0,75}$	ano	desvio padrão	máx	\
0	570.75	2561.0	21900.75	2002	22694.113198	115949	
1	421.50	2841.0	19971.25	2003	22204.089361	111905	
2	401.00	2928.5	20336.50	2004	22652.233093	112995	
3	403.25	2774.5	20428.00	2005	21750.923060	114571	
4	431.75	3381.5	20853.25	2006	22733.084717	117970	
5	352.75	3502.5	19988.00	2007	22392.000650	118481	
6	377.50	3337.5	21216.75	2008	22583.486559	119731	
7	364.75	3407.0	19923.50	2009	21255.163022	117707	
8	367.00	3621.0	20273.50	2010	21564.927135	117440	
9	391.25	3430.5	19490.00	2011	21373.929291	117310	

```
10      350.75      3111.0    20369.75  2012   21797.648913  119266
11      273.50      3210.0    20522.50  2013   21885.374415  120383
12      234.50      3255.0    19598.25  2014   21171.259013  112162
13      274.50      3409.0    19662.25  2015   20943.243333  116850
14      242.25      3249.0    21005.75  2016   21830.653437  113582
15      126.25      2660.5    19270.00  2017   21693.134275  121277
```

	média	mín
0	15522.414286	0
1	15165.200000	0
2	15410.114286	0
3	14904.400000	0
4	15349.571429	0
5	15146.671429	0
6	15193.571429	0
7	14447.942857	0
8	14665.900000	0
9	14648.085714	0
10	14745.814286	0
11	14838.128571	0
12	14415.571429	0
13	14195.257143	0
14	14879.142857	0
15	13215.828571	0

1.2.1 Histogramas

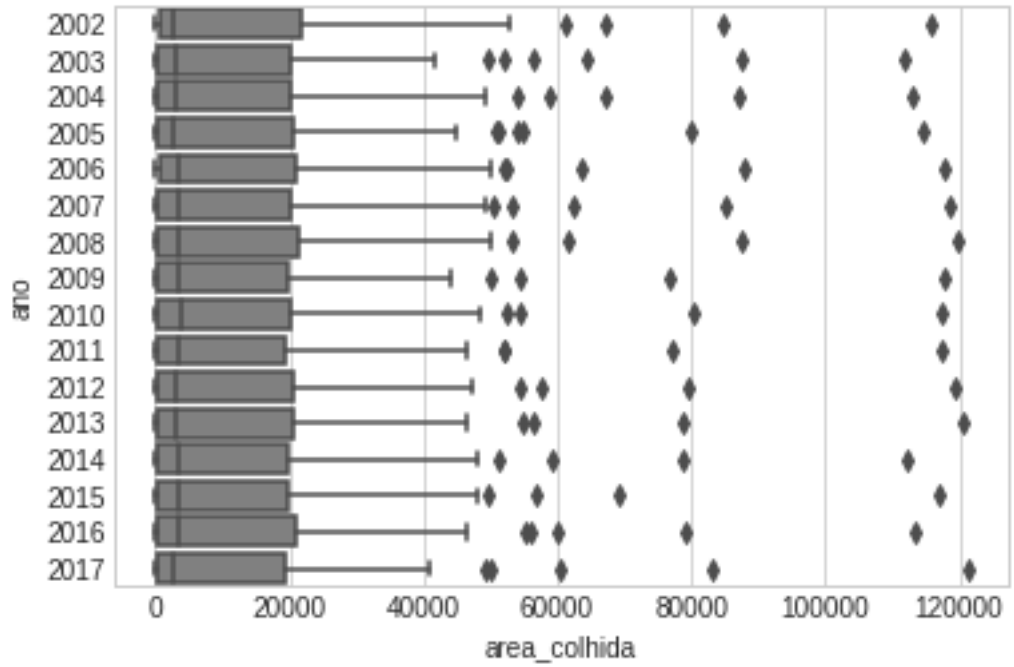
In [12]:
```
# histogramas da area colhida por ano
g = sns.FacetGrid(df_rgi, col='ano', col_wrap=4)
g.map(plt.hist, 'area_colhida', bins=10)
plt.show();
```



A área colhida média parece estar reduzindo ao longo do tempo.

1.2.2 Box plots

```
In [13]: sns.boxplot(x='area_colhida', y='ano', orient='h', color='gray', data=df_rgi);
```



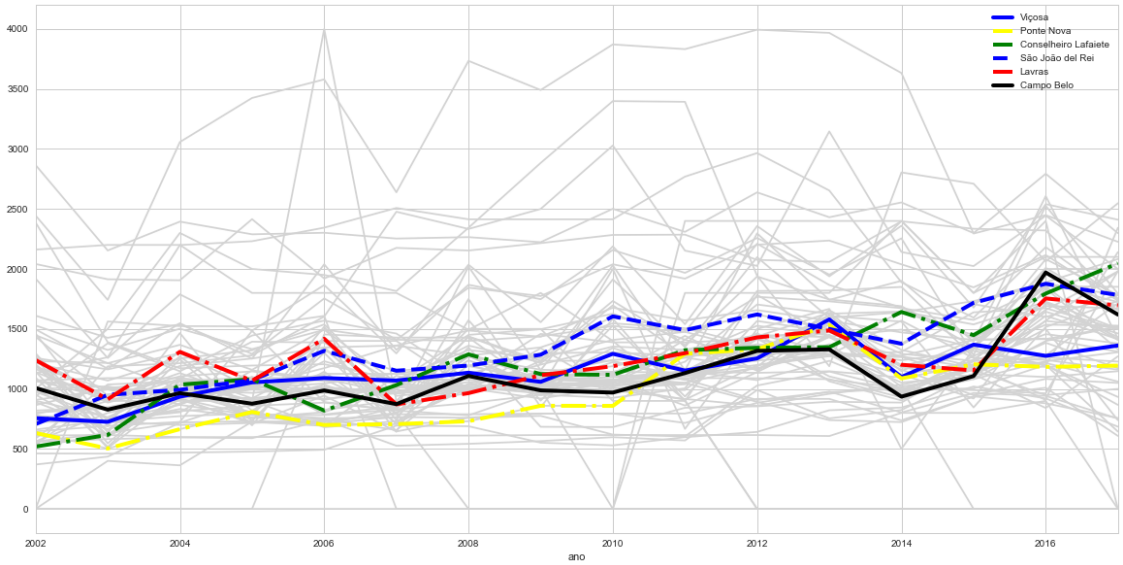
1.3 Séries históricas de produtividade

```
In [28]: # os 5 maiores valores por ano
         for i in range(len(anos)-1, 0, -1):
             (separado_rgi[anos[i]].sort_values(by=['ano', 'rendimento'], ascending=False)
             .loc[:,['nome_rgi', 'rendimento', 'ano']]
             .iloc[:5,:])
```

```
In [17]: # nomes das regiões
         serie_rend.columns


         ---------------------------------------------------------------------------

         NameError                                 Traceback (most recent call last)

         <ipython-input-17-17e32507e0f5> in <module>
           1 # nomes das regiões
         ----> 2 serie_rend.columns


         NameError: name 'serie_rend' is not defined
```

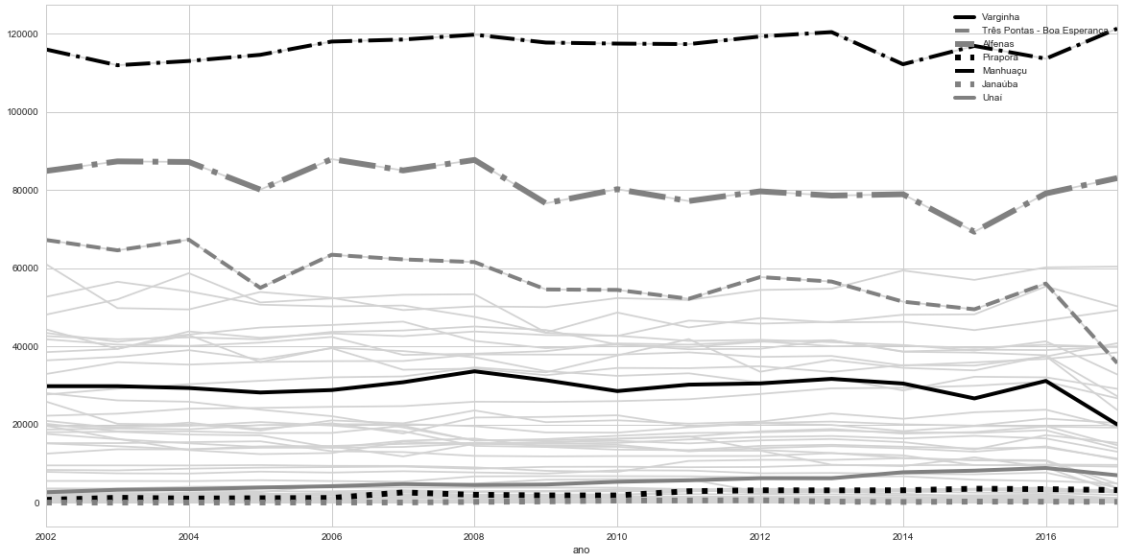
```
In [29]: serie_rend = df_rgi.pivot_table(values='rendimento', columns='nome_rgi', index='ano')
         serie_rend.plot(legend=False, c='lightgray', figsize=(20, 10))
```

```
#serie_rend['Varginha'].plot(legend=True, c='black', linewidth=4, label='Varginha')
#serie_rend['Patrocínio'].plot(legend=True, c='red', linewidth=4, ls='--', label='Patro
#serie_rend['Três Pontas - Boa Esperança'].plot(legend=True, c='red', linewidth=4, ls='
#serie_rend['Alfenas'].plot(legend=True, c='green', linewidth=4, ls='--', label='Alfena
#serie_rend['Pirapora'].plot(legend=True, c='red', linewidth=4, label='Pirapora')
#serie_rend['Manhuaçu'].plot(legend=True, c='blue', linewidth=4, ls='dashdot',  label='
#serie_rend['Janaúba'].plot(legend=True, c='blue', linewidth=4, ls=':', label='Janaúba'
#serie_rend['Unaí'].plot(legend=True, c='green', linewidth=4, label='Unaí')

serie_rend['Viçosa'].plot(legend=True, c='blue', linewidth=4,  label='Viçosa')
serie_rend['Ponte Nova'].plot(legend=True, c='yellow', linewidth=4, ls='dashdot', label
serie_rend['Conselheiro Lafaiete'].plot(legend=True, c='green', linewidth=4, ls='dashdo
serie_rend['São João del Rei'].plot(legend=True, c='blue', linewidth=4, ls='--', label=
serie_rend['Lavras'].plot(legend=True, c='red', linewidth=4, ls='dashdot', label='Lavra
#serie_rend['Juiz de Fora'].plot(legend=True, c='red', linewidth=4, label='Juiz de Fora
serie_rend['Campo Belo'].plot(legend=True, c='black', linewidth=4, label='Campo Belo');
```



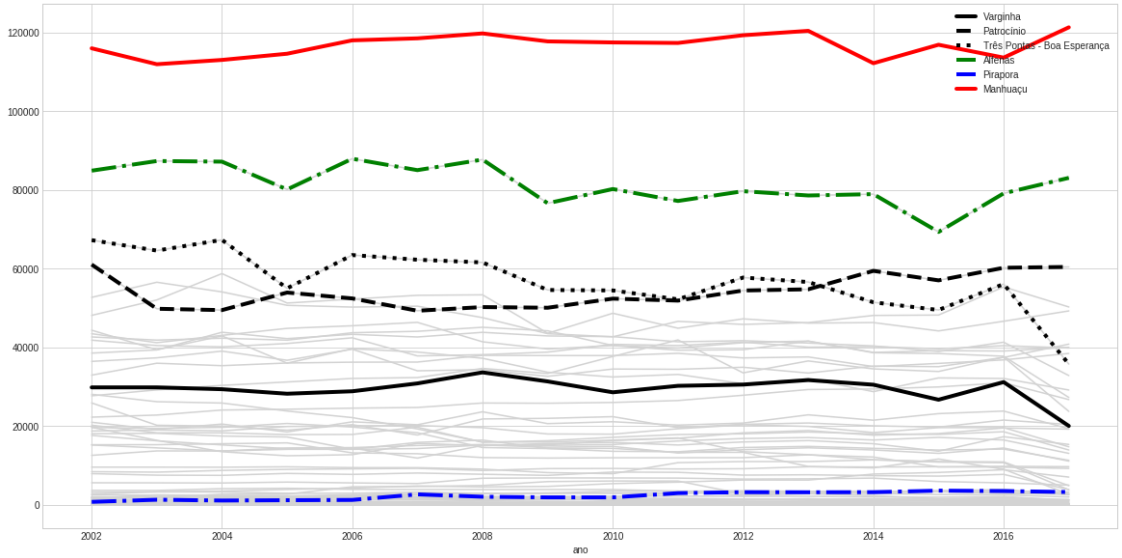
In [36]:
```
serie_rend = df_rgi.pivot_table(values='area_colhida', columns='nome_rgi', index='ano')
serie_rend.plot(legend=False, c='lightgray', figsize=(20, 10))
serie_rend['Varginha'].plot(legend=True, c='black', linewidth=4, label='Varginha')
#serie_rend['Patrocínio'].plot(legend=True, c='red', linewidth=4, ls='--', label='Patro
serie_rend['Três Pontas - Boa Esperança'].plot(legend=True, c='gray', linewidth=4, ls='
serie_rend['Alfenas'].plot(legend=True, c='gray', linewidth=6, ls='-.', label='Alfenas'
serie_rend['Pirapora'].plot(legend=True, c='black', linewidth=6,ls=':', label='Pirapora
serie_rend['Manhuaçu'].plot(legend=True, c='black', linewidth=4, ls='dashdot',  label='
serie_rend['Janaúba'].plot(legend=True, c='gray', linewidth=6, ls=':', label='Janaúba')
serie_rend['Unaí'].plot(legend=True, c='gray', linewidth=4, label='Unaí');
```

```
#serie_rend['Viçosa'].plot(legend=True, c='yellow', linewidth=4,  label='Viçosa')
#serie_rend['Ponte Nova'].plot(legend=True, c='yellow', linewidth=4, ls='dashdot', labe
#serie_rend['Conselheiro Lafaiete'].plot(legend=True, c='green', linewidth=4, ls='dashd
#serie_rend['São João del Rei'].plot(legend=True, c='blue', linewidth=4, ls='dashdot',
#serie_rend['Lavras'].plot(legend=True, c='red', linewidth=4, ls='dashdot', label='Lavr
#serie_rend['Juiz de Fora'].plot(legend=True, c='red', linewidth=4, label='Juiz de Fora
#serie_rend['Campo Belo'].plot(legend=True, c='black', linewidth=4, label='Campo Belo')
```



1.4 Séries históricas de área colhida

In [15]:
```
serie_rend = df_rgi.pivot_table(values='area_colhida', columns='nome_rgi', index='ano')
serie_rend.plot(legend=False, c='lightgray', figsize=(20, 10))
serie_rend['Varginha'].plot(legend=True, c='black', linewidth=4, label='Varginha')
serie_rend['Patrocínio'].plot(legend=True, c='black', linewidth=4, ls='--', label='Patr
serie_rend['Três Pontas - Boa Esperança'].plot(legend=True, c='black', linewidth=4, ls=
serie_rend['Alfenas'].plot(legend=True, c='green', linewidth=4, ls='-.', label='Alfenas
serie_rend['Pirapora'].plot(legend=True, c='blue', linewidth=4, ls='dashdot', label='Pi
serie_rend['Manhuaçu'].plot(legend=True, c='red', linewidth=4,  label='Manhuaçu');
```



2 Coeficiente de correlação de Spearman

```
In [46]: import scipy
         for i in range (2002, 2018):
             print(scipy.stats.spearmanr(separado_rgi[i].rendimento,separado_rgi[i].area_colhida
```

```
SpearmanrResult(correlation=0.5149926939797621, pvalue=5.080435813659008e-06)
SpearmanrResult(correlation=-0.09995888266193007, pvalue=0.41032463803901453)
SpearmanrResult(correlation=0.22763459489425664, pvalue=0.05806614533365894)
SpearmanrResult(correlation=-0.09255695139447809, pvalue=0.4460112763850296)
SpearmanrResult(correlation=0.07532215311418583, pvalue=0.535439078122395)
SpearmanrResult(correlation=-0.1708031879127886, pvalue=0.1574431078192574)
SpearmanrResult(correlation=0.24687232069430104, pvalue=0.03936996820696436)
SpearmanrResult(correlation=0.18243059991424418, pvalue=0.1306465099539139)
SpearmanrResult(correlation=0.41909178405809777, pvalue=0.000305033840776283)
SpearmanrResult(correlation=0.31642856519184404, pvalue=0.00761427725537501)
SpearmanrResult(correlation=0.2835268486139558, pvalue=0.017384437682155095)
SpearmanrResult(correlation=0.3627438971038585, pvalue=0.002027776289058153)
SpearmanrResult(correlation=0.2597648047039059, pvalue=0.029882554593586533)
SpearmanrResult(correlation=0.25723512064624326, pvalue=0.03157501380287619)
SpearmanrResult(correlation=0.4262197524770091, pvalue=0.00023421297664804353)
SpearmanrResult(correlation=0.33378868281405977, pvalue=0.004744802338101944)
```

Os valores de correlação são baixos, sugerindo que o fato de que uma região geográfica ter uma grande área colhida não necessariamente faz com que a produtividade seja alta.

```
In [48]: i=2002
         rho=scipy.stats.spearmanr(separado_rgi[i].rendimento,separado_rgi[i].area_colhida)
```

In [55]:

Out[55]: SpearmanrResult(correlation=0.5149926939797621, pvalue=5.080435813659008e-06)

In [ ]: