In [2]:
```
import pandas as pd
import pandas_datareader.data as web
from pandas.plotting import autocorrelation_plot
from pandas.tseries.offsets import MonthEnd
from pandas.plotting import scatter_matrix
from numpy import corrcoef, sum, log, arange
from numpy.random import rand
from pylab import pcolor, show, colorbar, xticks, yticks
```

In [3]:
```
import datetime
```

In [4]:
```
start= datetime.datetime(2012, 7, 31)
end = datetime.datetime(2017,6,30)
fb = web.DataReader('FB.US', 'quandl', start, end) # Facebook
mmm = web.DataReader('MMM.US', 'quandl', start, end) # 3M
ibm = web.DataReader('IBM.US', 'quandl', start, end) # IBM
amzn = web.DataReader('AMZN.US', 'quandl', start, end) # Amazon
```

In [5]:
```
month_end = pd.date_range(start='31/07/2012', end='30/06/2017', freq='BM')
month_end.values[8] = month_end[8].replace(day = 28)
```

In [48]:
```
fb_close = fb.loc[month_end, ['AdjClose']]
fb_close
```

Out[48]:

	AdjClose
2012-07-31	21.7100
2012-08-31	18.0580
2012-09-28	21.6600
2012-10-31	21.1100
2012-11-30	28.0000
2012-12-31	26.6197
2013-01-31	30.9810
2013-02-28	27.2500
2013-03-28	25.5800
2013-04-30	27.7690
2013-05-31	24.3480
2013-06-28	24.8800
2013-07-31	36.8000
2013-08-30	41.2940
2013-09-30	50.2300
2013-10-31	50.2050
2013-11-29	47.0100
2013-12-31	54.6490
2014-01-31	62.5700
2014-02-28	68.4600
2014-03-31	60.2400
2014-04-30	59.7800
2014-05-30	63.3000
2014-06-30	67.2900
2014-07-31	72.6500
2014-08-29	74.8200
2014-09-30	79.0400
2014-10-31	74.9900
2014-11-28	77.7000
2014-12-31	78.0200
2015-01-30	75.9100
2015-02-27	78.9700
2015-03-31	82.2150
2015-04-30	78.7700

	AdjClose
2015-05-29	79.1900
2015-06-30	85.7650
2015-07-31	94.0100
2015-08-31	89.4300
2015-09-30	89.7300
2015-10-30	101.9700
2015-11-30	104.2400
2015-12-31	104.6600
2016-01-29	112.2100
2016-02-29	106.9200
2016-03-31	114.1000
2016-04-29	117.5800
2016-05-31	118.8100
2016-06-30	114.2800
2016-07-29	123.9400
2016-08-31	126.1200
2016-09-30	128.2700
2016-10-31	130.9900
2016-11-30	118.4200
2016-12-30	115.0500
2017-01-31	130.3200
2017-02-28	135.5400
2017-03-31	142.0500
2017-04-28	150.2500
2017-05-31	151.4600
2017-06-30	150.9800

In [49]:
```
mmm_close = mmm.loc[month_end, ['AdjClose']]
mmm_close
```

Out[49]:

	AdjClose
2012-07-31	80.117221
2012-08-31	81.838026
2012-09-28	81.678945
2012-10-31	77.419126
2012-11-30	80.912524
2012-12-31	82.602835
2013-01-31	89.453043
2013-02-28	93.093473
2013-03-28	95.161222
2013-04-30	93.729015
2013-05-31	99.271674
2013-06-28	98.443435
2013-07-31	105.717536
2013-08-30	102.824007
2013-09-30	108.101907
2013-10-31	113.932041
2013-11-29	121.460776
2013-12-31	127.592494
2014-01-31	116.620904
2014-02-28	123.374090
2014-03-31	124.225704
2014-04-30	127.366601
2014-05-30	131.328426
2014-06-30	131.964109
2014-07-31	129.799101
2014-08-29	133.448707
2014-09-30	131.298700
2014-10-31	142.502831
2014-11-28	149.158384
2014-12-31	153.099542
2015-01-30	151.217476
2015-02-27	158.113092
2015-03-31	154.644260
2015-04-30	146.619072

	AdjClose
2015-05-29	150.085228
2015-06-30	145.575501
2015-07-31	142.782866
2015-08-31	135.045373
2015-09-30	134.693841
2015-10-30	149.363185
2015-11-30	149.733457
2015-12-31	144.053187
2016-01-29	144.397446
2016-02-29	151.103017
2016-03-31	160.504212
2016-04-29	161.226639
2016-05-31	163.210890
2016-06-30	169.804485
2016-07-29	172.946140
2016-08-31	174.871967
2016-09-30	171.935319
2016-10-31	161.271681
2016-11-30	168.635989
2016-12-30	175.342545
2017-01-31	171.660322
2017-02-28	184.165220
2017-03-31	189.086834
2017-04-28	193.534075
2017-05-31	203.291397
2017-06-30	206.989955

In [50]:
```
ibm_close = ibm.loc[month_end, ['AdjClose']]
ibm_close
```

Out[50]:

	AdjClose
2012-07-31	167.112403
2012-08-31	166.858424
2012-09-28	177.648345
2012-10-31	166.584394
2012-11-30	163.488846
2012-12-31	164.761869
2013-01-31	174.670805
2013-02-28	173.474505
2013-03-28	184.245939
2013-04-30	174.951582
2013-05-31	180.518558
2013-06-28	165.844157
2013-07-31	169.254589
2013-08-30	158.969760
2013-09-30	161.507764
2013-10-31	156.300931
2013-11-29	157.541673
2013-12-31	164.459548
2014-01-31	154.911302
2014-02-28	163.238274
2014-03-31	169.691286
2014-04-30	173.199890
2014-05-30	163.468620
2014-06-30	160.728774
2014-07-31	169.950263
2014-08-29	171.517421
2014-09-30	169.314363
2014-10-31	146.632678
2014-11-28	145.629115
2014-12-31	144.075571
2015-01-30	137.672811
2015-02-27	146.443279
2015-03-31	145.141079
2015-04-30	154.898539

	AdjClose
2015-05-29	154.588309
2015-06-30	148.218888
2015-07-31	147.608372
2015-08-31	135.880886
2015-09-30	133.197999
2015-10-30	128.705081
2015-11-30	129.303220
2015-12-31	127.633834
2016-01-29	115.734821
2016-02-29	122.766148
2016-03-31	141.898292
2016-04-29	136.735799
2016-05-31	145.413010
2016-06-30	143.559170
2016-07-29	151.920370
2016-08-31	151.572963
2016-09-30	151.544342
2016-10-31	146.621655
2016-11-30	156.155648
2016-12-30	159.784712
2017-01-31	167.995831
2017-02-28	174.473283
2017-03-31	168.962171
2017-04-28	155.523983
2017-05-31	149.543338
2017-06-30	150.719070

In [51]:
```
amzn_close = amzn.loc[month_end, ['AdjClose']]
amzn_close
```

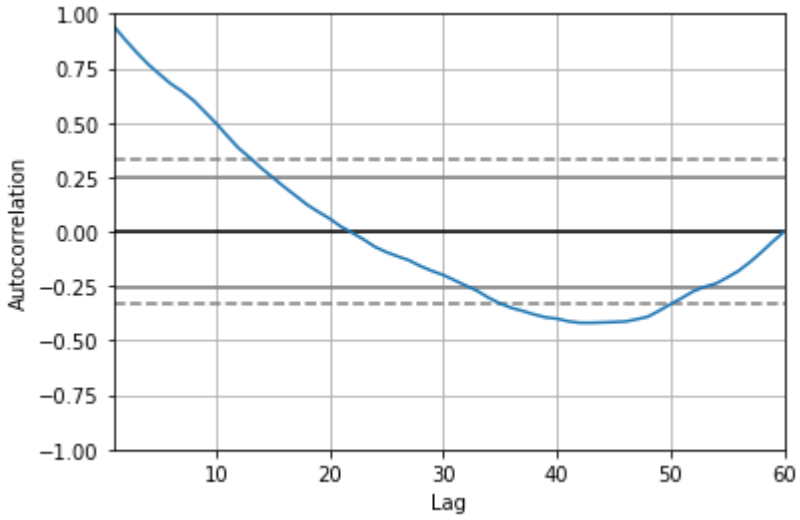
Out[51]:

	AdjClose
2012-07-31	233.3000
2012-08-31	248.2700
2012-09-28	254.3200
2012-10-31	232.8931
2012-11-30	252.0500
2012-12-31	250.8700
2013-01-31	265.5000
2013-02-28	264.2700
2013-03-28	266.4900
2013-04-30	253.8100
2013-05-31	269.2000
2013-06-28	277.6900
2013-07-31	301.2200
2013-08-30	280.9800
2013-09-30	312.6400
2013-10-31	364.0300
2013-11-29	393.6200
2013-12-31	398.7900
2014-01-31	358.6900
2014-02-28	362.1000
2014-03-31	336.3650
2014-04-30	304.1300
2014-05-30	312.5500
2014-06-30	324.7800
2014-07-31	312.9900
2014-08-29	339.0400
2014-09-30	322.4400
2014-10-31	305.4600
2014-11-28	338.6400
2014-12-31	310.3500
2015-01-30	354.5300
2015-02-27	380.1600
2015-03-31	372.1000
2015-04-30	421.7800

	AdjClose
2015-05-29	429.2300
2015-06-30	434.0900
2015-07-31	536.1500
2015-08-31	512.8900
2015-09-30	511.6700
2015-10-30	625.9000
2015-11-30	664.8000
2015-12-31	675.8900
2016-01-29	587.0000
2016-02-29	552.5200
2016-03-31	593.6400
2016-04-29	659.5900
2016-05-31	722.7900
2016-06-30	715.6200
2016-07-29	758.8100
2016-08-31	769.1600
2016-09-30	837.3100
2016-10-31	789.8200
2016-11-30	750.5700
2016-12-30	749.8700
2017-01-31	823.4800
2017-02-28	845.0400
2017-03-31	886.5400
2017-04-28	924.9900
2017-05-31	994.6200
2017-06-30	968.0000

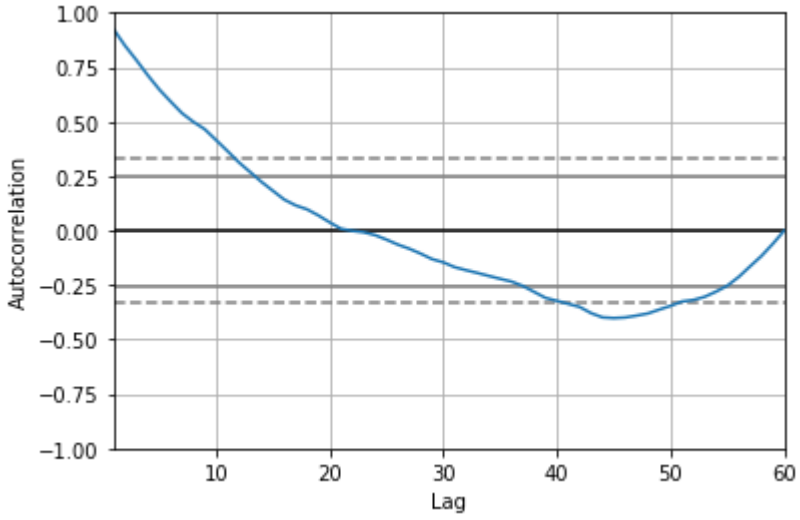
In [52]:
```
autocorrelation_plot(fb_close)
```

Out[52]: <matplotlib.axes._subplots.AxesSubplot at 0xc2f9128>



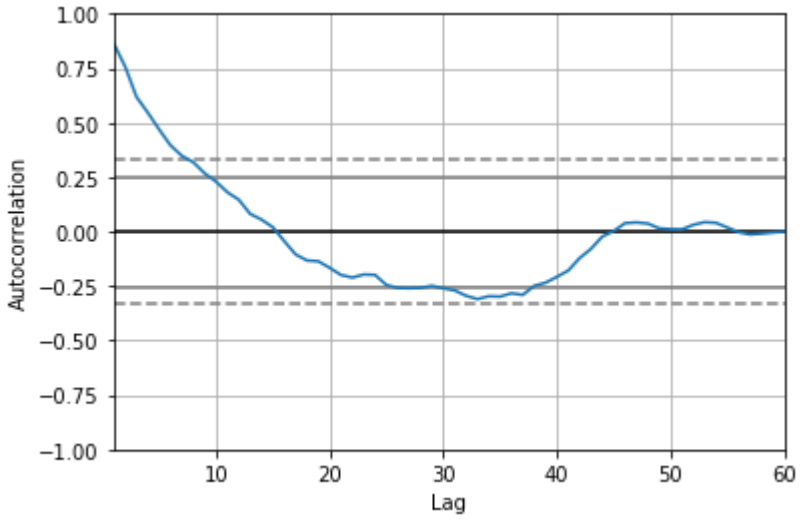
In [53]:
```
autocorrelation_plot(mmm_close)
```

Out[53]: <matplotlib.axes._subplots.AxesSubplot at 0xc440748>



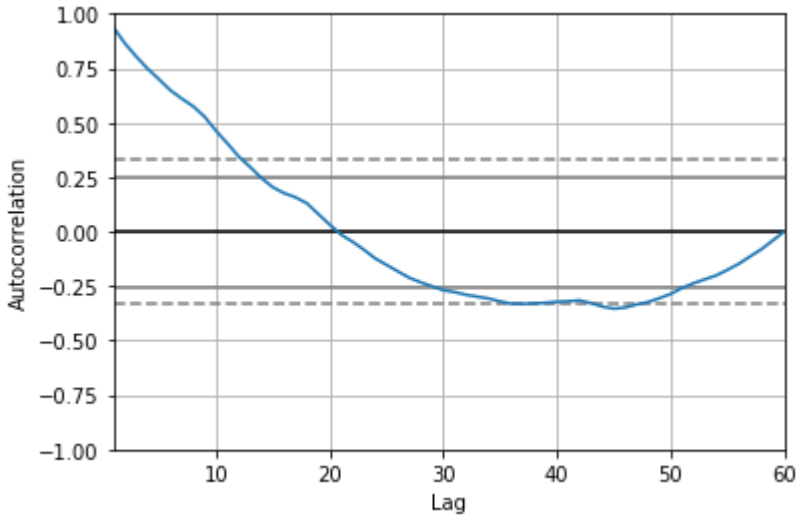
In [54]:
```
autocorrelation_plot(ibm_close)
```

Out[54]:
```
<matplotlib.axes._subplots.AxesSubplot at 0xc4b15c0>
```



In [55]:
```
autocorrelation_plot(amzn_close)
```

Out[55]:
```
<matplotlib.axes._subplots.AxesSubplot at 0xc4ab048>
```



According to the autocorrelation plots of the 4 stocks, we can tell that the autocorrelations are significantly different from zero when the time lags are not so large (With most stocks slightly larger than 10 and IBM slightly larger than 5). Thus, we can conclude that the time series for each of the stocks are autocorrelated and not random. When the price of the stock rises, it tends to continue rising. When the price of the stock falls, it tends to continue falling. However, when the time lag is large, the prices are not related. A price change 20 months ago probably would not affect the price trend today.

Monthly Returns of Stocks:

In [56]:
```
fb_return = fb_close / fb_close.shift(1) -1
fb_return.columns = ['fb_return']
fb_return
```

Out[56]:

	fb_return
2012-07-31	NaN
2012-08-31	-0.168217
2012-09-28	0.199468
2012-10-31	-0.025392
2012-11-30	0.326386
2012-12-31	-0.049296
2013-01-31	0.163837
2013-02-28	-0.120429
2013-03-28	-0.061284
2013-04-30	0.085575
2013-05-31	-0.123195
2013-06-28	0.021850
2013-07-31	0.479100
2013-08-30	0.122120
2013-09-30	0.216399
2013-10-31	-0.000498
2013-11-29	-0.063639
2013-12-31	0.162497
2014-01-31	0.144943
2014-02-28	0.094135
2014-03-31	-0.120070
2014-04-30	-0.007636
2014-05-30	0.058883
2014-06-30	0.063033
2014-07-31	0.079655
2014-08-29	0.029869
2014-09-30	0.056402
2014-10-31	-0.051240
2014-11-28	0.036138
2014-12-31	0.004118
2015-01-30	-0.027044
2015-02-27	0.040311
2015-03-31	0.041092

	fb_return
2015-04-30	-0.041902
2015-05-29	0.005332
2015-06-30	0.083028
2015-07-31	0.096135
2015-08-31	-0.048718
2015-09-30	0.003355
2015-10-30	0.136409
2015-11-30	0.022261
2015-12-31	0.004029
2016-01-29	0.072138
2016-02-29	-0.047144
2016-03-31	0.067153
2016-04-29	0.030500
2016-05-31	0.010461
2016-06-30	-0.038128
2016-07-29	0.084529
2016-08-31	0.017589
2016-09-30	0.017047
2016-10-31	0.021205
2016-11-30	-0.095962
2016-12-30	-0.028458
2017-01-31	0.132725
2017-02-28	0.040055
2017-03-31	0.048030
2017-04-28	0.057726
2017-05-31	0.008053
2017-06-30	-0.003169

In [57]:
```
mmm_return = mmm_close / mmm_close.shift(1) -1
mmm_return.columns = ['mmm_return']
mmm_return
```

Out[57]:

	mmm_return
2012-07-31	NaN
2012-08-31	0.021479
2012-09-28	-0.001944
2012-10-31	-0.052153
2012-11-30	0.045123
2012-12-31	0.020891
2013-01-31	0.082929
2013-02-28	0.040697
2013-03-28	0.022212
2013-04-30	-0.015050
2013-05-31	0.059135
2013-06-28	-0.008343
2013-07-31	0.073891
2013-08-30	-0.027370
2013-09-30	0.051329
2013-10-31	0.053932
2013-11-29	0.066081
2013-12-31	0.050483
2014-01-31	-0.085989
2014-02-28	0.057907
2014-03-31	0.006903
2014-04-30	0.025284
2014-05-30	0.031106
2014-06-30	0.004840
2014-07-31	-0.016406
2014-08-29	0.028117
2014-09-30	-0.016111
2014-10-31	0.085333
2014-11-28	0.046705
2014-12-31	0.026423
2015-01-30	-0.012293
2015-02-27	0.045601
2015-03-31	-0.021939

	mmm_return
2015-04-30	-0.051895
2015-05-29	0.023641
2015-06-30	-0.030048
2015-07-31	-0.019183
2015-08-31	-0.054191
2015-09-30	-0.002603
2015-10-30	0.108909
2015-11-30	0.002479
2015-12-31	-0.037936
2016-01-29	0.002390
2016-02-29	0.046438
2016-03-31	0.062217
2016-04-29	0.004501
2016-05-31	0.012307
2016-06-30	0.040399
2016-07-29	0.018502
2016-08-31	0.011135
2016-09-30	-0.016793
2016-10-31	-0.062021
2016-11-30	0.045664
2016-12-30	0.039769
2017-01-31	-0.021000
2017-02-28	0.072847
2017-03-31	0.026724
2017-04-28	0.023520
2017-05-31	0.050417
2017-06-30	0.018193

In [58]:
```
ibm_return = ibm_close / ibm_close.shift(1) -1
ibm_return.columns = ['ibm_return']
ibm_return
```

Out[58]:

	ibm_return
2012-07-31	NaN
2012-08-31	-0.001520
2012-09-28	0.064665
2012-10-31	-0.062280
2012-11-30	-0.018582
2012-12-31	0.007787
2013-01-31	0.060141
2013-02-28	-0.006849
2013-03-28	0.062092
2013-04-30	-0.050445
2013-05-31	0.031820
2013-06-28	-0.081290
2013-07-31	0.020564
2013-08-30	-0.060765
2013-09-30	0.015965
2013-10-31	-0.032239
2013-11-29	0.007938
2013-12-31	0.043911
2014-01-31	-0.058058
2014-02-28	0.053753
2014-03-31	0.039531
2014-04-30	0.020676
2014-05-30	-0.056185
2014-06-30	-0.016761
2014-07-31	0.057373
2014-08-29	0.009221
2014-09-30	-0.012845
2014-10-31	-0.133962
2014-11-28	-0.006844
2014-12-31	-0.010668
2015-01-30	-0.044440
2015-02-27	0.063705
2015-03-31	-0.008892

	ibm_return
2015-04-30	0.067227
2015-05-29	-0.002003
2015-06-30	-0.041202
2015-07-31	-0.004119
2015-08-31	-0.079450
2015-09-30	-0.019744
2015-10-30	-0.033731
2015-11-30	0.004647
2015-12-31	-0.012911
2016-01-29	-0.093228
2016-02-29	0.060754
2016-03-31	0.155842
2016-04-29	-0.036382
2016-05-31	0.063460
2016-06-30	-0.012749
2016-07-29	0.058242
2016-08-31	-0.002287
2016-09-30	-0.000189
2016-10-31	-0.032483
2016-11-30	0.065024
2016-12-30	0.023240
2017-01-31	0.051389
2017-02-28	0.038557
2017-03-31	-0.031587
2017-04-28	-0.079534
2017-05-31	-0.038455
2017-06-30	0.007862

In [67]:
```
amzn_return = amzn_close / amzn_close.shift(1) -1
amzn_return.columns = ['amzn_return']
amzn_return
```

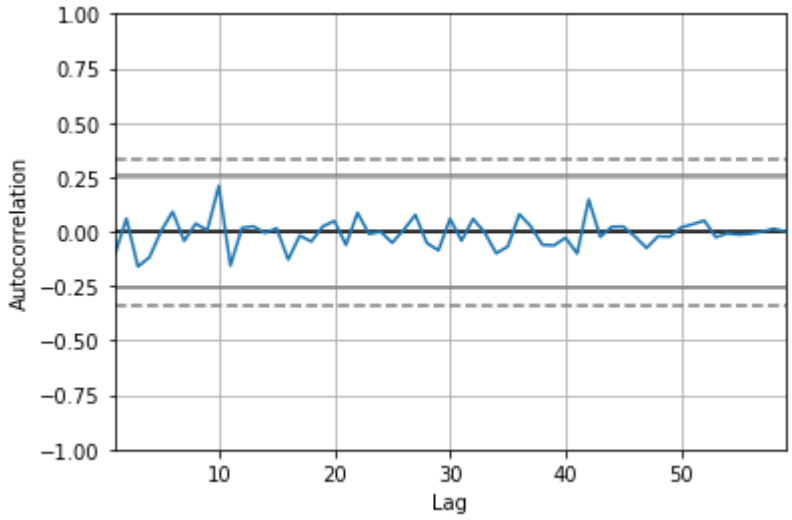
Out[67]:

	amzn_return
2012-07-31	NaN
2012-08-31	0.064166
2012-09-28	0.024369
2012-10-31	-0.084252
2012-11-30	0.082256
2012-12-31	-0.004682
2013-01-31	0.058317
2013-02-28	-0.004633
2013-03-28	0.008400
2013-04-30	-0.047582
2013-05-31	0.060636
2013-06-28	0.031538
2013-07-31	0.084735
2013-08-30	-0.067193
2013-09-30	0.112677
2013-10-31	0.164374
2013-11-29	0.081285
2013-12-31	0.013134
2014-01-31	-0.100554
2014-02-28	0.009507
2014-03-31	-0.071072
2014-04-30	-0.095833
2014-05-30	0.027686
2014-06-30	0.039130
2014-07-31	-0.036301
2014-08-29	0.083229
2014-09-30	-0.048962
2014-10-31	-0.052661
2014-11-28	0.108623
2014-12-31	-0.083540
2015-01-30	0.142355
2015-02-27	0.072293
2015-03-31	-0.021202

	amzn_return
2015-04-30	0.133512
2015-05-29	0.017663
2015-06-30	0.011323
2015-07-31	0.235113
2015-08-31	-0.043383
2015-09-30	-0.002379
2015-10-30	0.223249
2015-11-30	0.062151
2015-12-31	0.016682
2016-01-29	-0.131515
2016-02-29	-0.058739
2016-03-31	0.074423
2016-04-29	0.111094
2016-05-31	0.095817
2016-06-30	-0.009920
2016-07-29	0.060353
2016-08-31	0.013640
2016-09-30	0.088603
2016-10-31	-0.056717
2016-11-30	-0.049695
2016-12-30	-0.000933
2017-01-31	0.098164
2017-02-28	0.026182
2017-03-31	0.049110
2017-04-28	0.043371
2017-05-31	0.075276
2017-06-30	-0.026764

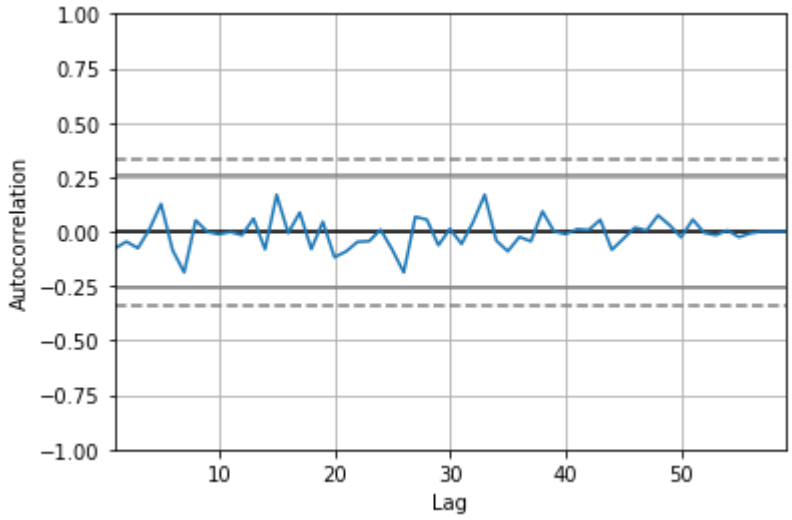
In [60]:
```
autocorrelation_plot(fb_return[1:])
```

Out[60]: <matplotlib.axes._subplots.AxesSubplot at 0xc533828>



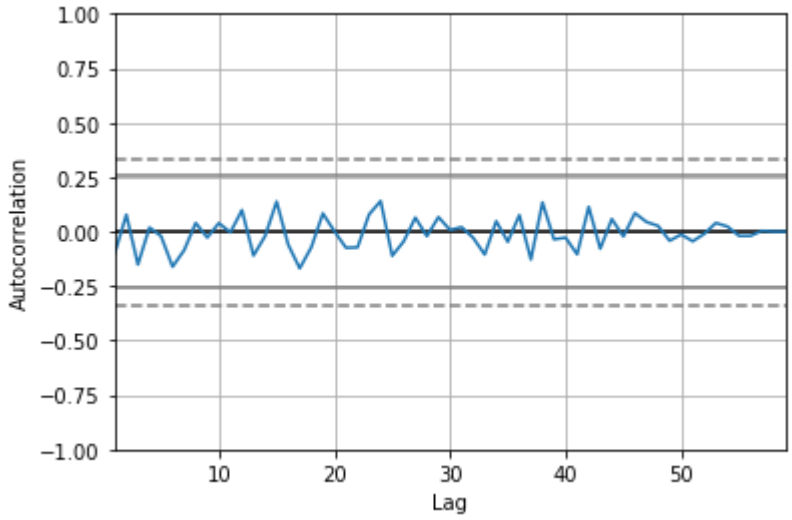
In [61]:
```
autocorrelation_plot(mmm_return[1:])
```

Out[61]: <matplotlib.axes._subplots.AxesSubplot at 0xc594828>



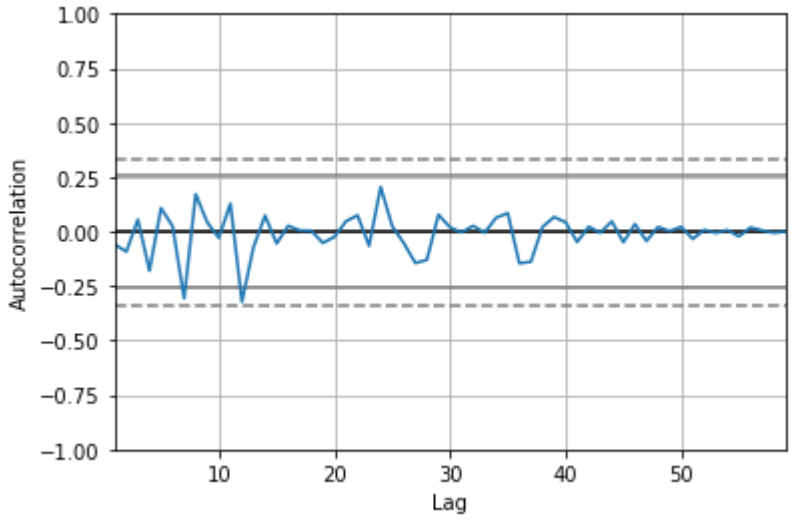
In [62]:
```
autocorrelation_plot(ibm_return[1:])
```

Out[62]:
```
<matplotlib.axes._subplots.AxesSubplot at 0xc645da0>
```



In [63]:
```
autocorrelation_plot(amzn_return[1:])
```

Out[63]:
```
<matplotlib.axes._subplots.AxesSubplot at 0xc5eb198>
```



Unfortunately, all 4 stocks' autocorrelation plots show insignificant level of autocorrelation of the monthly returns. The indicator is not significantly different from zero. We can conclude that monthly returns are not autocorrelated, and an increase in the monthly return does not lead to an increase in next months' return.

Scatterplot Matrix:

In [68]:
```
matrix = fb_return.join(mmm_return).join(ibm_return).join(amzn_return)
matrix
```

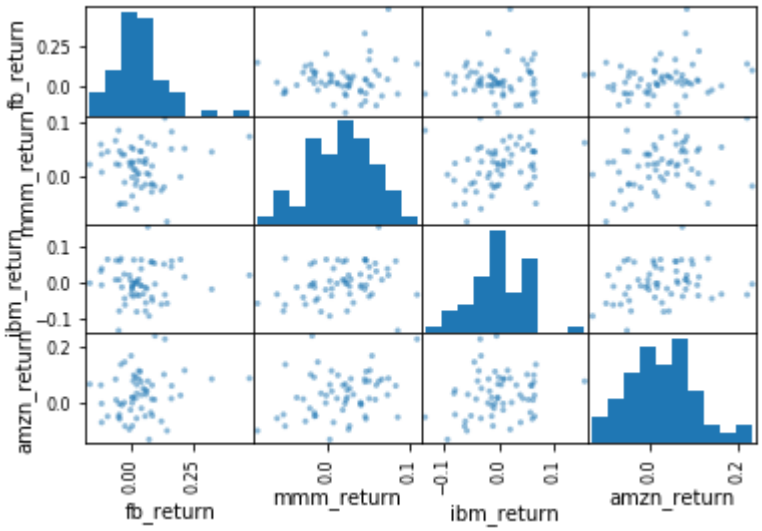
Out[68]:

	fb_return	mmm_return	ibm_return	amzn_return
2012-07-31	NaN	NaN	NaN	NaN
2012-08-31	-0.168217	0.021479	-0.001520	0.064166
2012-09-28	0.199468	-0.001944	0.064665	0.024369
2012-10-31	-0.025392	-0.052153	-0.062280	-0.084252
2012-11-30	0.326386	0.045123	-0.018582	0.082256
2012-12-31	-0.049296	0.020891	0.007787	-0.004682
2013-01-31	0.163837	0.082929	0.060141	0.058317
2013-02-28	-0.120429	0.040697	-0.006849	-0.004633
2013-03-28	-0.061284	0.022212	0.062092	0.008400
2013-04-30	0.085575	-0.015050	-0.050445	-0.047582
2013-05-31	-0.123195	0.059135	0.031820	0.060636
2013-06-28	0.021850	-0.008343	-0.081290	0.031538
2013-07-31	0.479100	0.073891	0.020564	0.084735
2013-08-30	0.122120	-0.027370	-0.060765	-0.067193
2013-09-30	0.216399	0.051329	0.015965	0.112677
2013-10-31	-0.000498	0.053932	-0.032239	0.164374
2013-11-29	-0.063639	0.066081	0.007938	0.081285
2013-12-31	0.162497	0.050483	0.043911	0.013134
2014-01-31	0.144943	-0.085989	-0.058058	-0.100554
2014-02-28	0.094135	0.057907	0.053753	0.009507
2014-03-31	-0.120070	0.006903	0.039531	-0.071072
2014-04-30	-0.007636	0.025284	0.020676	-0.095833
2014-05-30	0.058883	0.031106	-0.056185	0.027686
2014-06-30	0.063033	0.004840	-0.016761	0.039130
2014-07-31	0.079655	-0.016406	0.057373	-0.036301
2014-08-29	0.029869	0.028117	0.009221	0.083229
2014-09-30	0.056402	-0.016111	-0.012845	-0.048962
2014-10-31	-0.051240	0.085333	-0.133962	-0.052661
2014-11-28	0.036138	0.046705	-0.006844	0.108623
2014-12-31	0.004118	0.026423	-0.010668	-0.083540
2015-01-30	-0.027044	-0.012293	-0.044440	0.142355
2015-02-27	0.040311	0.045601	0.063705	0.072293
2015-03-31	0.041092	-0.021939	-0.008892	-0.021202

	fb_return	mmm_return	ibm_return	amzn_return
2015-04-30	-0.041902	-0.051895	0.067227	0.133512
2015-05-29	0.005332	0.023641	-0.002003	0.017663
2015-06-30	0.083028	-0.030048	-0.041202	0.011323
2015-07-31	0.096135	-0.019183	-0.004119	0.235113
2015-08-31	-0.048718	-0.054191	-0.079450	-0.043383
2015-09-30	0.003355	-0.002603	-0.019744	-0.002379
2015-10-30	0.136409	0.108909	-0.033731	0.223249
2015-11-30	0.022261	0.002479	0.004647	0.062151
2015-12-31	0.004029	-0.037936	-0.012911	0.016682
2016-01-29	0.072138	0.002390	-0.093228	-0.131515
2016-02-29	-0.047144	0.046438	0.060754	-0.058739
2016-03-31	0.067153	0.062217	0.155842	0.074423
2016-04-29	0.030500	0.004501	-0.036382	0.111094
2016-05-31	0.010461	0.012307	0.063460	0.095817
2016-06-30	-0.038128	0.040399	-0.012749	-0.009920
2016-07-29	0.084529	0.018502	0.058242	0.060353
2016-08-31	0.017589	0.011135	-0.002287	0.013640
2016-09-30	0.017047	-0.016793	-0.000189	0.088603
2016-10-31	0.021205	-0.062021	-0.032483	-0.056717
2016-11-30	-0.095962	0.045664	0.065024	-0.049695
2016-12-30	-0.028458	0.039769	0.023240	-0.000933
2017-01-31	0.132725	-0.021000	0.051389	0.098164
2017-02-28	0.040055	0.072847	0.038557	0.026182
2017-03-31	0.048030	0.026724	-0.031587	0.049110
2017-04-28	0.057726	0.023520	-0.079534	0.043371
2017-05-31	0.008053	0.050417	-0.038455	0.075276
2017-06-30	-0.003169	0.018193	0.007862	-0.026764

In [74]:
```
scatter_matrix(matrix)
```

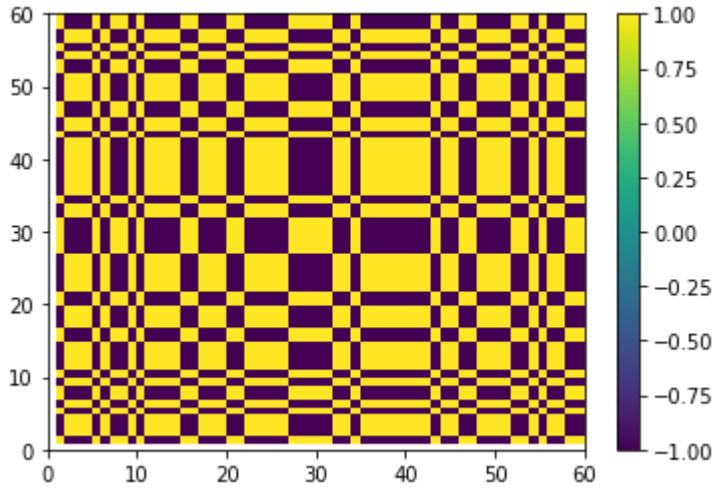
Out[74]:
```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x000000000E025A20>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000000000E0A5B70>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000000000E0DAC18>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000000000E0F7EB8>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x000000000E146128>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000000000E146160>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000000000E1ADFD0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000000000E1F14E0>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x000000000E216A20>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000000000E253A20>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000000000E288F60>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000000000E2C1FD0>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x000000000E2F9EF0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000000000E330470>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000000000E36A470>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000000000E3A5470>]],
      dtype=object)
```



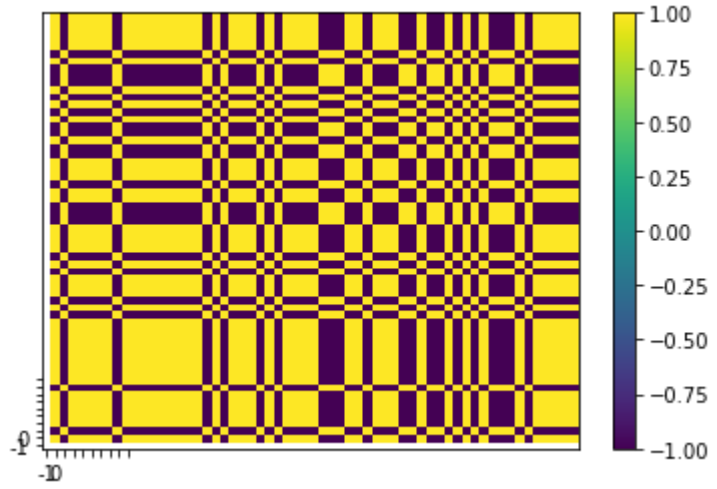
There might be some weak correlations between mmm and ibm, mmm and amzn, etc. However, the correlation is not very significant.

Visualize the correlation of the returns of all pair of stocks:

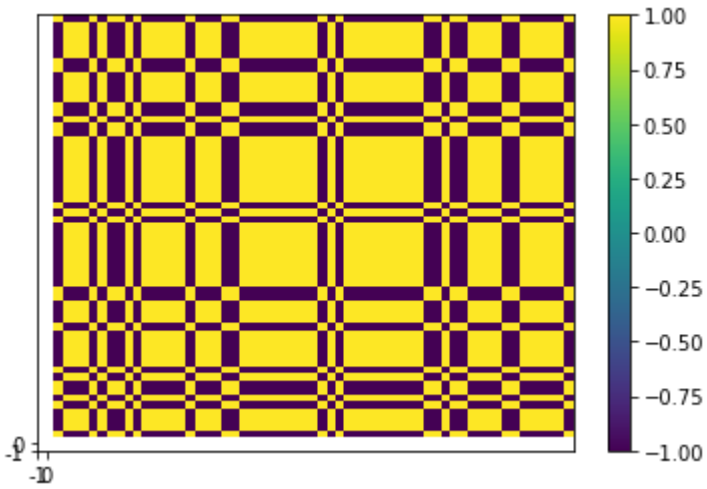
In [86]:
```
fb_mmm = matrix[['fb_return', 'mmm_return']]
R = corrcoef(fb_mmm)
pcolor(R)
colorbar()
yticks(arange(-1,1),range(-1,1))
xticks(arange(-1,1),range(-1,1))
show()
```



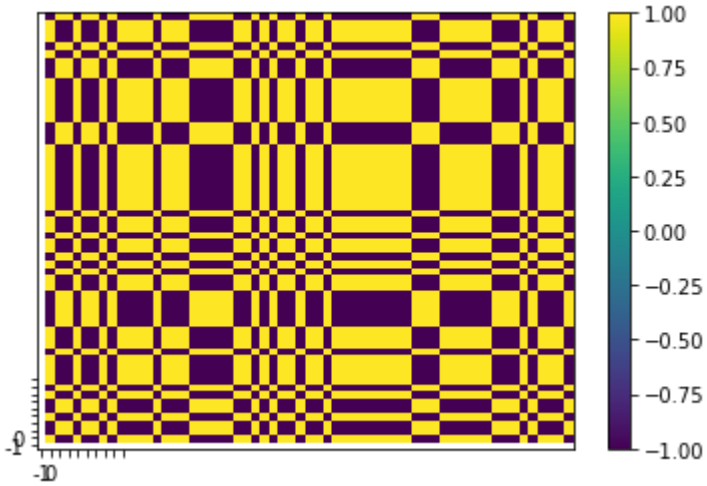
In [90]:
```
mmm_ibm = matrix[['mmm_return', 'ibm_return']]
R = corrcoef(mmm_ibm)
pcolor(R)
colorbar()
yticks(arange(-1,1),range(-1,1))
xticks(arange(-1,1),range(-1,1))
show()
```



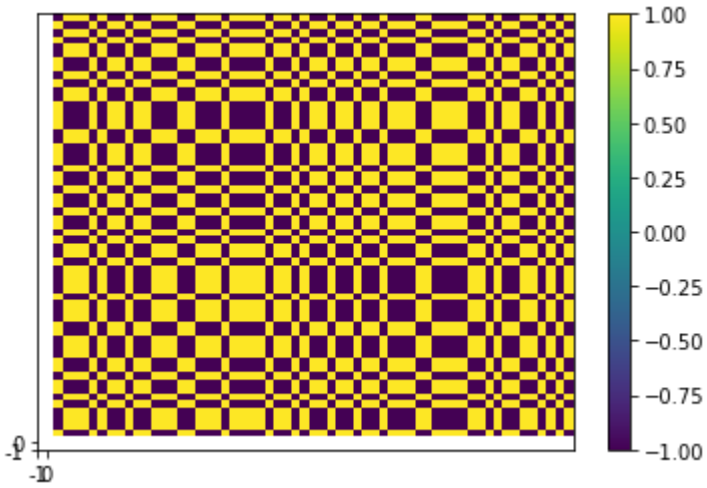
In [94]:
```
fb_ibm = matrix[['fb_return', 'ibm_return']]
R = corrcoef(fb_ibm)
pcolor(R)
colorbar()
yticks(arange(-1,1),range(-1,1))
xticks(arange(-1,1),range(-1,1))
show()
```



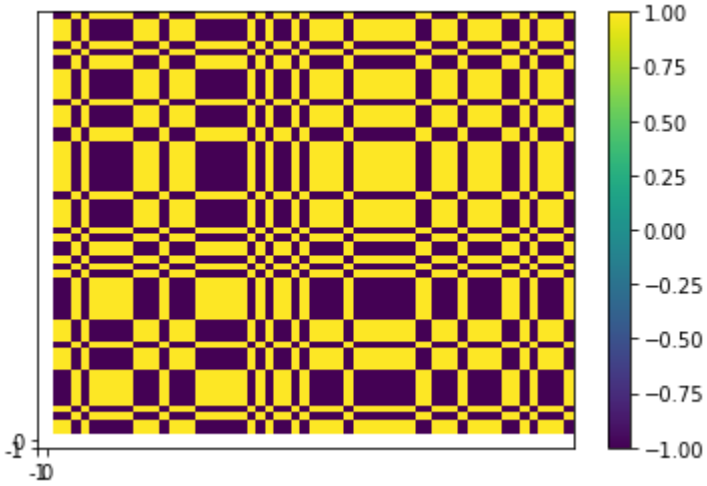
In [92]:
```
amzn_ibm = matrix[['amzn_return', 'ibm_return']]
R = corrcoef(amzn_ibm)
pcolor(R)
colorbar()
yticks(arange(-1,1),range(-1,1))
xticks(arange(-1,1),range(-1,1))
show()
```



In [95]:
```
amzn_fb = matrix[['amzn_return', 'fb_return']]
R = corrcoef(amzn_fb)
pcolor(R)
colorbar()
yticks(arange(-1,1),range(-1,1))
xticks(arange(-1,1),range(-1,1))
show()
```



In [96]:
```
amzn_mmm = matrix[['amzn_return', 'mmm_return']]
R = corrcoef(amzn_mmm)
pcolor(R)
colorbar()
yticks(arange(-1,1),range(-1,1))
xticks(arange(-1,1),range(-1,1))
show()
```



In [ ]: