rec7

October 29, 2016

1 Probability and distributions: uniform distribution

- more interesting than coin flips

```
In [6]: %matplotlib inline
        import numpy as np
        import matplotlib.pyplot as plt
        import matplotlib as mpl
```

```
In [3]: # uniform random number (from [0,1)])
        np.random.random()
```

```
Out[3]: 0.9840531854509492
```

```
In [5]: # generate a large array of them
        randArr = np.random.random((1000,))
        print(randArr)
```

```
[ 0.43932662  0.92303056  0.95362779  0.02507379  0.83851937  0.35469688
  0.0320268   0.30406117  0.45883142  0.40152035  0.76160598  0.79198227
  0.27431931  0.760312    0.70432578  0.85127965  0.02784033  0.42482476
  0.2569968   0.29446361  0.22716809  0.52394722  0.14830686  0.66193101
  0.24111876  0.35763802  0.54819984  0.18028755  0.20511505  0.26065344
  0.0129208   0.86549528  0.38462112  0.30245392  0.47046773  0.52114798
  0.64455454  0.23260381  0.53381666  0.06041079  0.68502009  0.57370341
  0.01354409  0.89999626  0.64444706  0.24819795  0.64366703  0.96643648
  0.7465468   0.90491128  0.83019648  0.48611479  0.50787884  0.1105018
  0.48140581  0.35895768  0.8425198   0.51873404  0.78455673  0.58223607
  0.42087894  0.23228898  0.38342084  0.00972764  0.66142659  0.77784459
  0.00397615  0.49816531  0.46996638  0.20360328  0.9385943   0.14654952
  0.14598159  0.89273158  0.15812949  0.5350867   0.97646967  0.82916241
  0.41132515  0.05288579  0.20432874  0.78283273  0.56570872  0.56443003
  0.51594208  0.68050231  0.19661863  0.08043029  0.15016969  0.77658018
  0.97104089  0.58462433  0.74754483  0.6495987   0.10052636  0.02861127
  0.74157117  0.38696799  0.50988199  0.71111949  0.51753063  0.64482443
  0.36589536  0.38969907  0.58907385  0.92638112  0.71451602  0.59843759
  0.06377072  0.35553937  0.47296108  0.24804759  0.66652508  0.3117917
  0.51148843  0.5319085   0.05760238  0.16958877  0.81987293  0.91378895
```

0.55937975 0.33384374 0.1398821 0.73122633 0.71360137 0.45312736
0.57651532 0.47617282 0.9461809 0.96356311 0.93511668 0.42332374
0.85842456 0.21360256 0.91805317 0.78873065 0.55770878 0.02240738
0.74744714 0.03033391 0.84507015 0.58910723 0.15073373 0.59461225
0.21457496 0.31698386 0.29511025 0.17333893 0.69991064 0.57095528
0.20065638 0.34185228 0.28005629 0.1832709 0.91377323 0.94377108
0.30782698 0.5587544 0.48450338 0.42084325 0.17049464 0.99695954
0.50210858 0.92857921 0.23761042 0.95685219 0.27714764 0.28223261
0.62799588 0.4077952 0.82303217 0.20348169 0.62772484 0.56742936
0.83884385 0.810281 0.15378064 0.50628967 0.15874285 0.99659345
0.50792328 0.67691276 0.18041525 0.69692144 0.50786358 0.88270624
0.59244193 0.52679537 0.15241995 0.16598696 0.35953992 0.16509911
0.9480007 0.08999627 0.92885129 0.15579174 0.37943517 0.07123411
0.38103516 0.22309144 0.26202543 0.59393957 0.23158819 0.44682806
0.69581977 0.75635696 0.06247903 0.18886842 0.92246008 0.76282843
0.5964749 0.5171499 0.06345998 0.73404381 0.16394927 0.36496767
0.1972421 0.53348667 0.67544531 0.41813454 0.96402583 0.58755629
0.34918835 0.70288453 0.09973856 0.86779494 0.85184236 0.10344631
0.95536792 0.90388592 0.11862975 0.51931685 0.96763422 0.87255248
0.13127469 0.77333015 0.81166002 0.66628673 0.80408877 0.21894738
0.0700805 0.60897729 0.64080776 0.58013586 0.85320161 0.56693035
0.8131021 0.63783704 0.25441161 0.59179871 0.54208472 0.77246441
0.75751766 0.71611498 0.93355952 0.69632649 0.03401593 0.84548702
0.46184747 0.05523436 0.1468018 0.31496159 0.47444963 0.25776627
0.55003102 0.404343 0.38498301 0.80882038 0.61959353 0.42884848
0.53195294 0.21601176 0.01105279 0.40119131 0.65309494 0.15388033
0.57859937 0.7321163 0.07577127 0.46993322 0.01524215 0.09393068
0.6961182 0.86144962 0.61262692 0.21246357 0.24680295 0.37929902
0.26945699 0.1713005 0.19412954 0.52561487 0.52870517 0.75511301
0.34056378 0.62820614 0.2273529 0.45282102 0.00702039 0.25784794
0.11116578 0.66115492 0.68624714 0.9439024 0.87152639 0.41086487
0.29715396 0.17541926 0.89771647 0.30535964 0.41981006 0.16236545
0.31062156 0.03667209 0.63214395 0.52224493 0.94954489 0.33884172
0.03649283 0.60126137 0.47659288 0.24433272 0.4372291 0.47979704
0.23857706 0.22216452 0.47182518 0.65444261 0.69217353 0.67754173
0.44212907 0.59419192 0.63549585 0.36383533 0.26357035 0.78064438
0.39462019 0.35581749 0.11699506 0.38069082 0.05322759 0.1466758
0.36499985 0.85950079 0.06211293 0.1845904 0.33616359 0.75150621
0.55483276 0.47152992 0.17451094 0.23515341 0.90755404 0.64995347
0.22558083 0.1929324 0.53486806 0.59087052 0.91899324 0.16706007
0.0456931 0.3863219 0.27136523 0.44569082 0.42172977 0.29638102
0.2127021 0.58231833 0.60884745 0.78896824 0.25065402 0.61507348
0.96483537 0.01722829 0.92264473 0.32004704 0.18459803 0.72114782
0.35852134 0.28698121 0.26868841 0.64706131 0.21180719 0.68089352
0.61090976 0.72536738 0.95704607 0.42853144 0.13518385 0.37613695
0.43585021 0.99413596 0.81712674 0.63559922 0.72458942 0.66116028
0.37253127 0.74794058 0.78464325 0.57293709 0.4576471 0.58393181
0.54138305 0.21701105 0.33881734 0.48459282 0.37691413 0.88644046

0.70501777 0.51802461 0.12541935 0.37590121 0.38558576 0.07252171
0.64777001 0.69315754 0.46176754 0.8868352 0.10376306 0.59170357
0.96743571 0.94043899 0.17716187 0.84060132 0.78314741 0.73924145
0.59247856 0.12220017 0.13563473 0.20155236 0.40447085 0.44215249
0.53658476 0.37558228 0.25910679 0.6471773 0.02491509 0.63812034
0.73069278 0.51927649 0.79606376 0.41565004 0.38767869 0.81450939
0.02341175 0.20117781 0.82433294 0.19301367 0.88930985 0.71171978
0.96106304 0.64866565 0.67235651 0.61693143 0.26005517 0.09376203
0.84433942 0.93984483 0.83656237 0.78064304 0.63337271 0.45928152
0.38751981 0.03315154 0.05642606 0.2801715 0.9399134 0.08353187
0.75063835 0.36924765 0.41062081 0.83278318 0.08746208 0.74671898
0.21883988 0.06179449 0.53010285 0.17432853 0.01595978 0.02526149
0.9774807 0.76380354 0.23489742 0.3417291 0.49217474 0.62758612
0.71593555 0.6909017 0.40052395 0.89378405 0.30500384 0.00107731
0.62565726 0.56515028 0.31893145 0.86382201 0.54015112 0.37985083
0.64833083 0.55625214 0.77958188 0.26198799 0.94677457 0.0416255
0.20750805 0.01597452 0.27092037 0.94590368 0.84403839 0.51537274
0.2279445 0.04948043 0.27373463 0.74225624 0.11514565 0.44441123
0.58837288 0.57407521 0.92857728 0.32328976 0.24894367 0.71712255
0.94981168 0.89542237 0.57874362 0.78587004 0.79987355 0.15739754
0.49298998 0.24889945 0.05244232 0.22076941 0.45194953 0.9804843
0.14452743 0.89307949 0.98562131 0.63688136 0.47573909 0.27929003
0.69467361 0.51792978 0.55315097 0.74848226 0.49749962 0.55826048
0.93208831 0.57708435 0.79198876 0.33811463 0.10287798 0.07309108
0.78789854 0.35694121 0.53000223 0.95328659 0.26303009 0.06476976
0.71839579 0.36298311 0.22415733 0.696899 0.00560628 0.42091965
0.4822852 0.23493967 0.46197674 0.07879657 0.37527484 0.4229862
0.74152402 0.41080962 0.31327701 0.99964755 0.55470863 0.76897624
0.31697359 0.56788991 0.06707363 0.36922984 0.15903509 0.25426496
0.28542875 0.13175239 0.13635778 0.08645855 0.38159811 0.35913535
0.86978696 0.60406654 0.30695474 0.13328814 0.61185743 0.68056821
0.60225342 0.65218193 0.52785508 0.76761758 0.60226084 0.40665953
0.87791654 0.32112559 0.88030398 0.78836286 0.31238076 0.57444553
0.11706464 0.44433153 0.67158736 0.11295846 0.66481026 0.23518386
0.62025794 0.62703058 0.40861567 0.03405474 0.73392949 0.83897385
0.40101056 0.15268419 0.05176716 0.07146226 0.08567706 0.62121937
0.1371628 0.37589467 0.6339966 0.12333397 0.16079589 0.00688329
0.98972781 0.86112564 0.89026177 0.91667138 0.2114442 0.06871565
0.81960936 0.15094872 0.23652675 0.75322694 0.79731321 0.31677258
0.44228134 0.23089302 0.02787079 0.23746463 0.68696909 0.71704725
0.743084 0.08254355 0.79241376 0.13948386 0.61654134 0.82328126
0.52182727 0.56237359 0.76896202 0.43471569 0.13894323 0.80243298
0.71944349 0.10209351 0.51426997 0.75030891 0.71225452 0.16234182
0.53910865 0.59798141 0.078647 0.71054931 0.23211762 0.45758345
0.59589025 0.43303924 0.26491673 0.47411696 0.45742957 0.58619684
0.36556212 0.15285757 0.82944139 0.59493611 0.45095902 0.87745856
0.38340057 0.69823855 0.78067903 0.942645 0.1202483 0.52081195
0.48249701 0.15176944 0.40545672 0.35689481 0.68621365 0.60559628

0.76421267 0.60606102 0.03110383 0.66821167 0.18707543 0.92368288
0.76145741 0.07585238 0.24282336 0.04216631 0.00382804 0.12557041
0.45153597 0.57625067 0.68324615 0.02624207 0.68814164 0.78145755
0.46624608 0.18040665 0.62592323 0.88037118 0.66442765 0.96668996
0.63013417 0.25797788 0.40656162 0.29106235 0.66247345 0.48487383
0.34088786 0.72081451 0.40848089 0.2826707 0.77383892 0.33494538
0.71879195 0.29774766 0.93991696 0.37779703 0.30689075 0.76684364
0.71762957 0.32355876 0.24894709 0.97990419 0.14187615 0.92538658
0.3940143 0.68573711 0.21937102 0.06815661 0.00903256 0.39850999
0.48630651 0.28022485 0.6103149 0.15803198 0.88045344 0.59132543
0.70533901 0.41980659 0.63125881 0.61234076 0.0386069 0.84003354
0.27399834 0.59956 0.01123798 0.0180974 0.30245131 0.01319276
0.96865767 0.4870637 0.53520469 0.4870407 0.3477795 0.54467148
0.59668636 0.53209259 0.54156836 0.56765144 0.84950404 0.5546395
0.89534344 0.79172727 0.3048083 0.33351259 0.15387158 0.40451773
0.88139497 0.40954657 0.84437798 0.79089875 0.50192308 0.72365858
0.43961813 0.83401361 0.69087981 0.64755321 0.42454302 0.49014904
0.80570654 0.52215229 0.74138136 0.51970188 0.94015521 0.45323925
0.27184695 0.45838716 0.32975391 0.67466557 0.44784556 0.87668905
0.75633446 0.15328136 0.70983551 0.10374459 0.20928868 0.3356614
0.02173325 0.78965579 0.36034044 0.94537085 0.87711555 0.2168508
0.16440045 0.7011963 0.98962202 0.7677211 0.85081785 0.46308655
0.90030043 0.31634549 0.38279634 0.86585344 0.96900432 0.74160708
0.24669413 0.51802949 0.89812708 0.06815765 0.46831067 0.73316151
0.93552231 0.86284065 0.27129189 0.97798991 0.58013436 0.38920452
0.12002624 0.05909501 0.4710373 0.73818626 0.06822255 0.81306965
0.38923105 0.92115704 0.3267842 0.69148187 0.0044167 0.25032246
0.37856981 0.18203591 0.44455971 0.08204781 0.57472614 0.77782608
0.2618245 0.04247161 0.65407639 0.48763423 0.27769929 0.3143389
0.32886905 0.25169513 0.13663463 0.15443939 0.83095545 0.45081846
0.82557636 0.73358935 0.79824905 0.60785886 0.85496023 0.48079173
0.79725715 0.72354572 0.11043345 0.37616595 0.21133766 0.53683559
0.80452218 0.66316972 0.23164575 0.29154686 0.77443356 0.72173494
0.69754316 0.29872704 0.19668142 0.50399012 0.57454168 0.97218849
0.4767616 0.10520071 0.19278107 0.59386833 0.93614735 0.99428087
0.64618155 0.46946538 0.81803433 0.37512709 0.94965987 0.22297383
0.85204064 0.15867521 0.34016733 0.62627007 0.61050765 0.16073221
0.93774393 0.73099545 0.08146973 0.97811464 0.18508904 0.50458728
0.81031978 0.58225243 0.21309503 0.41624688 0.27090479 0.38395516
0.88318522 0.41388706 0.07608012 0.78417215 0.82654159 0.31484083
0.2705083 0.397687 0.71524916 0.10315321 0.62014949 0.04299589
0.95053869 0.87185128 0.39792691 0.10222794 0.49194632 0.54197526
0.63745922 0.07116787 0.16742693 0.33192765 0.68103499 0.61395043
0.17392526 0.89001684 0.95276996 0.01353423 0.96762221 0.63234818
0.44633108 0.07266725 0.0280256 0.98085451 0.3969086 0.36787442
0.70924552 0.21341708 0.78733988 0.55147155 0.00175816 0.6206519
0.32630949 0.22801766 0.54388213 0.65927125 0.34628331 0.8289732
0.94455641 0.66675733 0.72155317 0.75097113 0.14694075 0.82850618

```
  0.93272662  0.89303781  0.79328387  0.29530145  0.52887442  0.86150836
  0.27020302  0.6402873   0.05707078  0.50577085  0.53507581  0.65536856
  0.17764123  0.9701673   0.52093017  0.37186804]
```

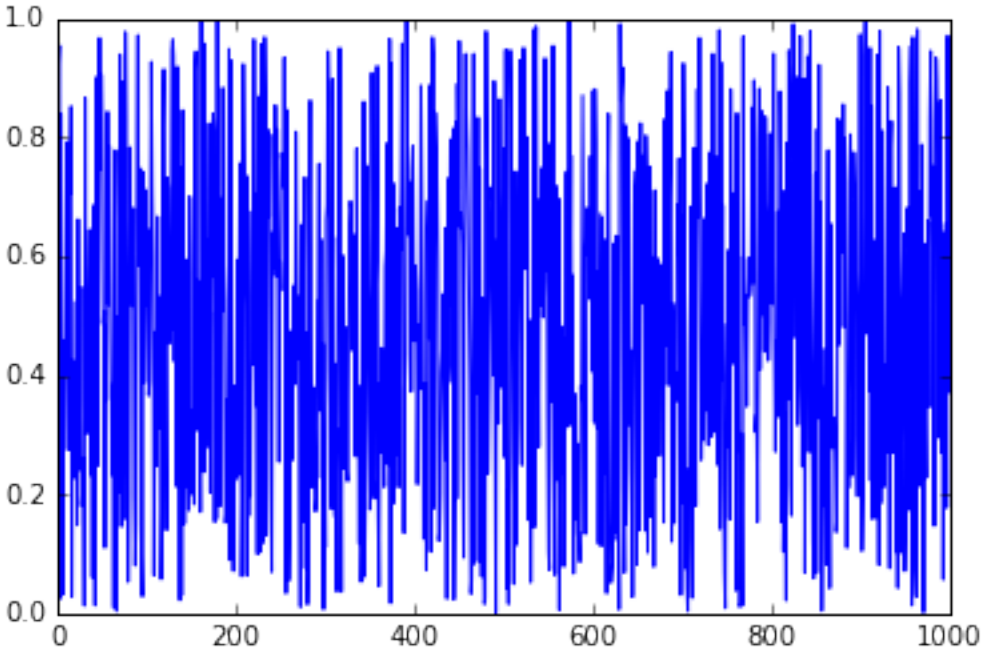
```
In [11]: np.set_printoptions(threshold=20)
         print(randArr)
```

```
[ 0.43932662  0.92303056  0.95362779 ...,  0.9701673   0.52093017
  0.37186804]
```

1.1 Use plots to look at it

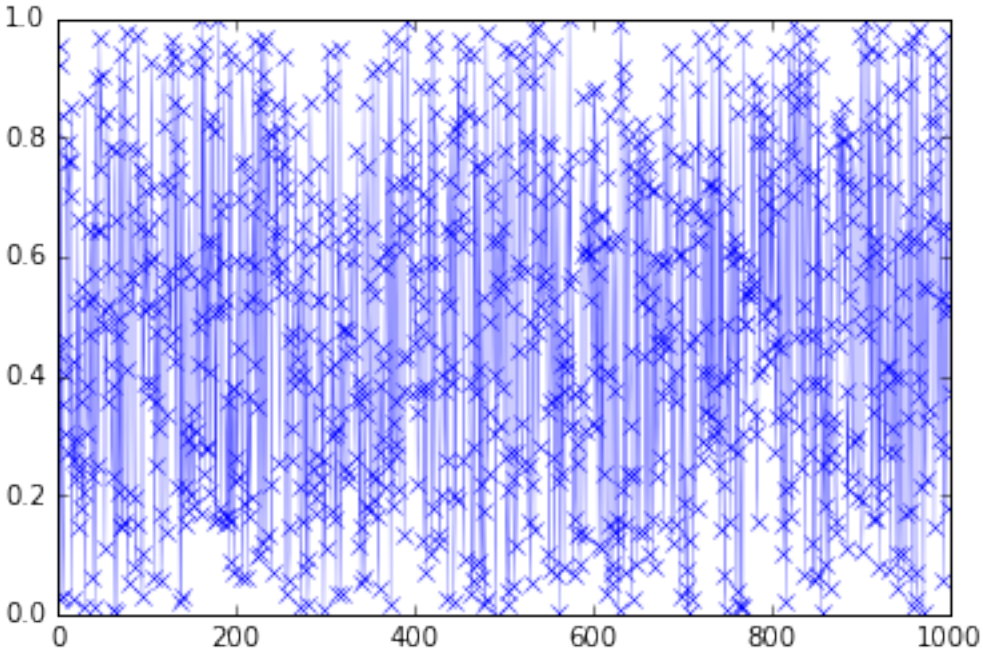
```
In [12]: plt.plot(randArr)
```

```
Out[12]: [<matplotlib.lines.Line2D at 0x10cbaddd8>]
```



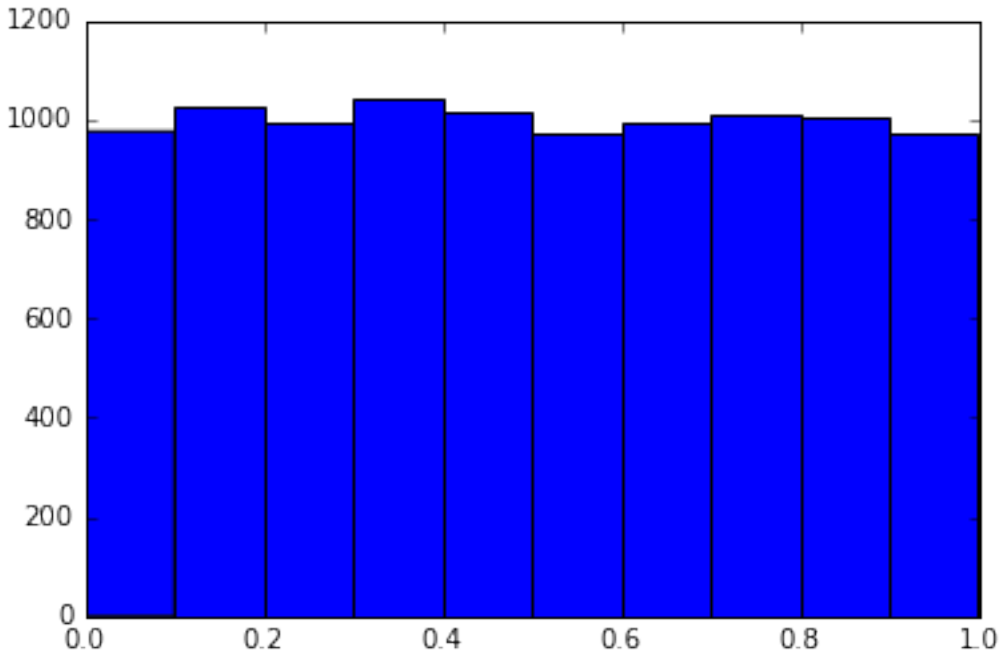

```
In [17]: plt.plot(randArr, marker='x', linewidth=0.2)
```

```
Out[17]: [<matplotlib.lines.Line2D at 0x10dd91320>]
```



1.2 Histogram

```
In [19]: bigArr = np.random.rand(10000,)
         plt.hist(bigArr)
         plt.xlabel('value')
         plt.ylabel('number in bin')
         plt.title('histogram')

Out[19]: (array([  976.,  1024.,   996.,  1042.,  1013.,   971.,   991.,  1009.,
                  1004.,   974.]),
          array([  1.97123596e-04,   1.00176522e-01,   2.00155920e-01,
                   3.00135319e-01,   4.00114717e-01,   5.00094115e-01,
                   6.00073514e-01,   7.00052912e-01,   8.00032310e-01,
                   9.00011709e-01,   9.99991107e-01]),
          <a list of 10 Patch objects>)
```




```
In [35]: # mean
         np.mean(bigArr)
         # measures of spread
         #np.var(bigArr)
         #np.std(bigArr)
```

```
Out[35]: 0.49880681536795812
```

1.3 SEM and confidence intervals

```
In [34]: nPts = len(bigArr)
         sem = np.std(bigArr)/np.sqrt(nPts)
         print(nPts)
         print(sem)
```

```
10000
0.00287522890105
```

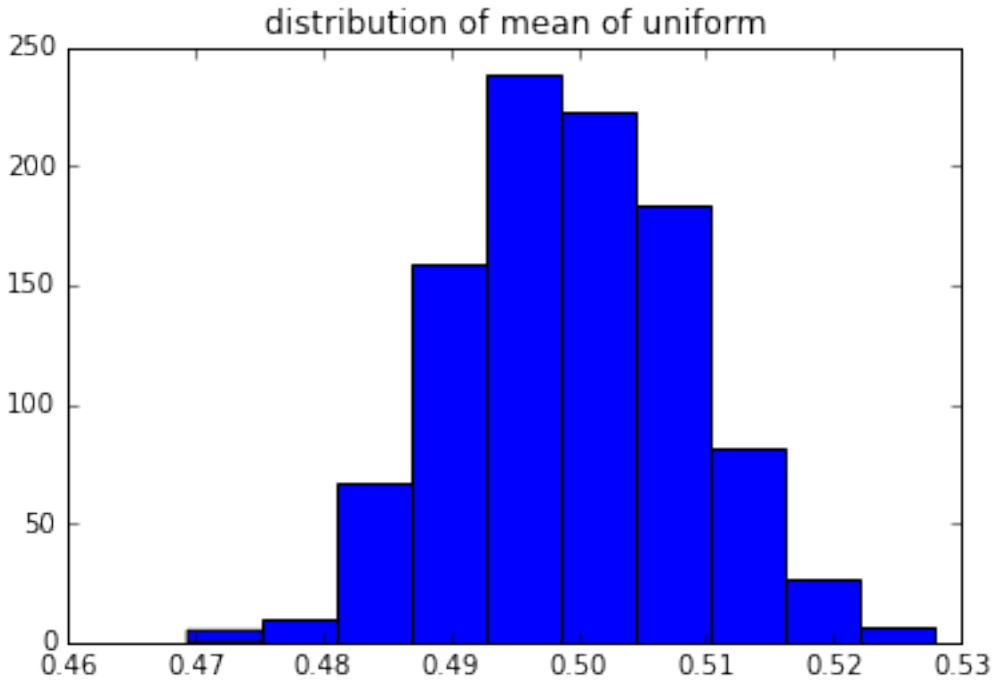
```
In [29]: # no equation, so we simulate the 95% CI for the mean
         nReps = 1000
         nPts = 1000
         meanV = np.zeros((nReps,))*np.nan
         # [show meanV here]
         for iR in range(nReps):
             tArr = np.random.random((nPts,))
```

```
        meanV[iR] = np.mean(tArr)


    print(np.percentile(meanV,[2.5, 97.5]))
```

```
[ 0.48270442  0.5174354 ]
```

```
In [39]: plt.hist(meanV)
         plt.title('distribution of mean of uniform')
```

```
Out[39]: <matplotlib.text.Text at 0x10ee46c50>
```



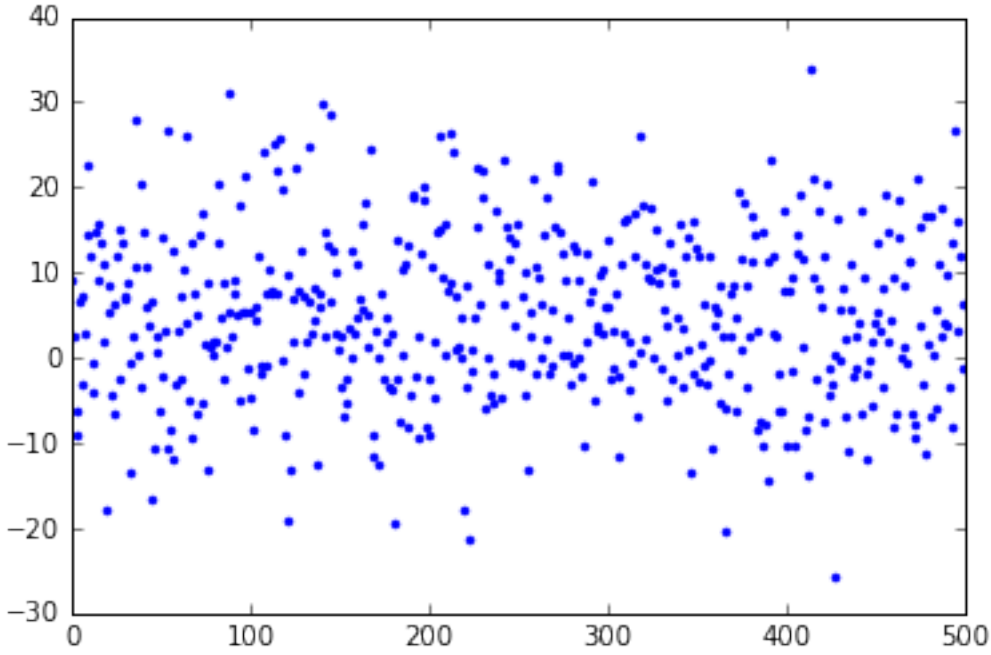
---

2 Normal dist

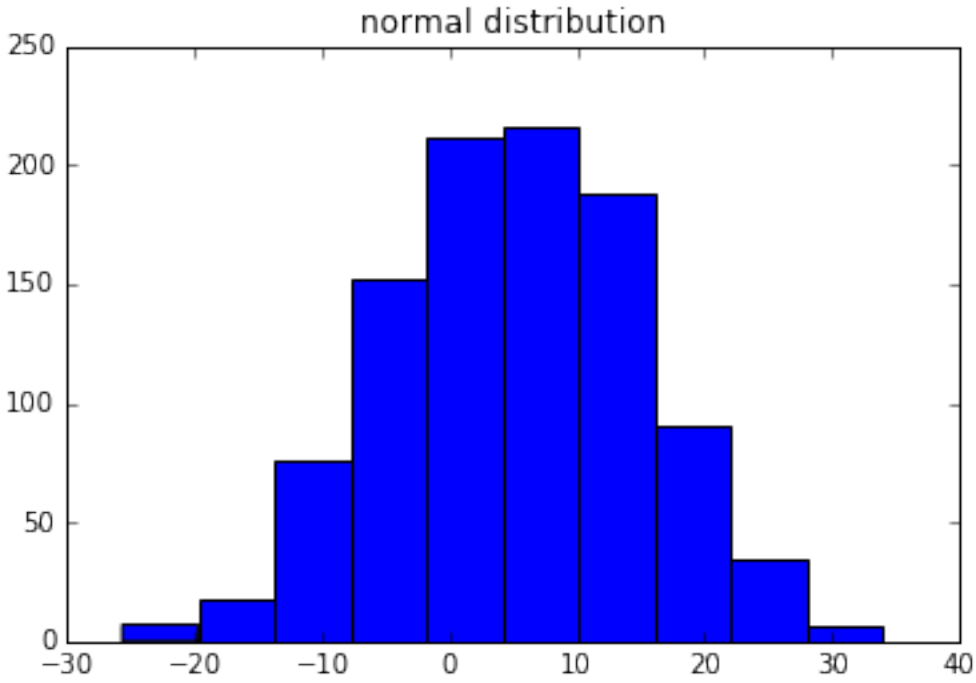
```
In [53]: #samples
         normArr = np.random.normal(loc=5.0, scale=10.0, size=(1000,))
```

```
In [50]: plt.plot(normArr[0:500], '.')
```

```
Out[50]: [<matplotlib.lines.Line2D at 0x10efd4400>]
```




```
In [51]: plt.hist(normArr)
         plt.title('normal distribution')

Out[51]: <matplotlib.text.Text at 0x10f07a390>
```



```
In [56]: mu = np.mean(normArr)
         print(mu)
```

```
4.84441322106
```

```
In [57]: np.std(normArr)
```

```
Out[57]: 9.8207158908412904
```

```
In [55]: sem = np.std(normArr)/np.sqrt(len(normArr))
         print(sem)
```

```
0.310558304685
```

Equation for the 95% CI from a SEM, assuming data are normally distributed

$$CI_{95} = mean \pm 1.96 * SEM$$

```
In [56]: CI_95 = (mu - 1.96*sem, mu + 1.96*sem)
         print(CI_95)
```

```
(4.3985522253596905, 5.6159407797236369)
```

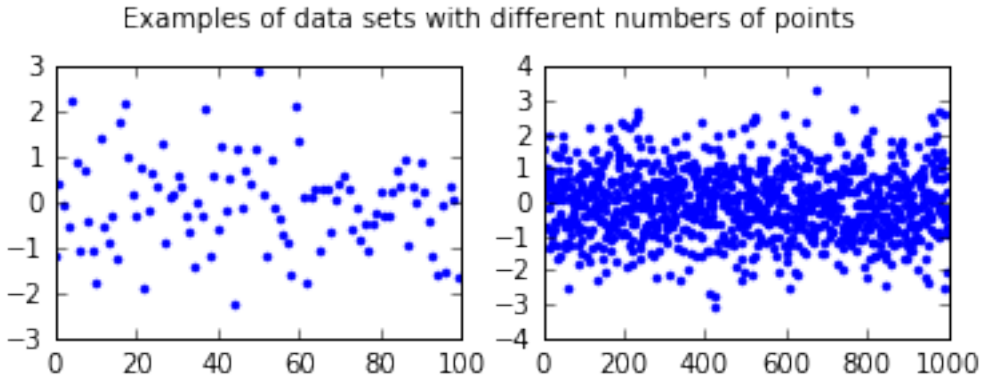
```
In [ ]: # simulate and see if we get it right
        nReps = 1000
        nPts = 1000
        meanV = np.zeros((nReps,))*np.nan
        # meanV here?
        for iR in range(nReps):
            tArr = np.random.normal(5, 10, (nPts,))
            meanV[iR] = np.mean(tArr)



        print(np.percentile(meanV,[2.5, 97.5]))
```

2.0.1 SEM as a function of number of trials

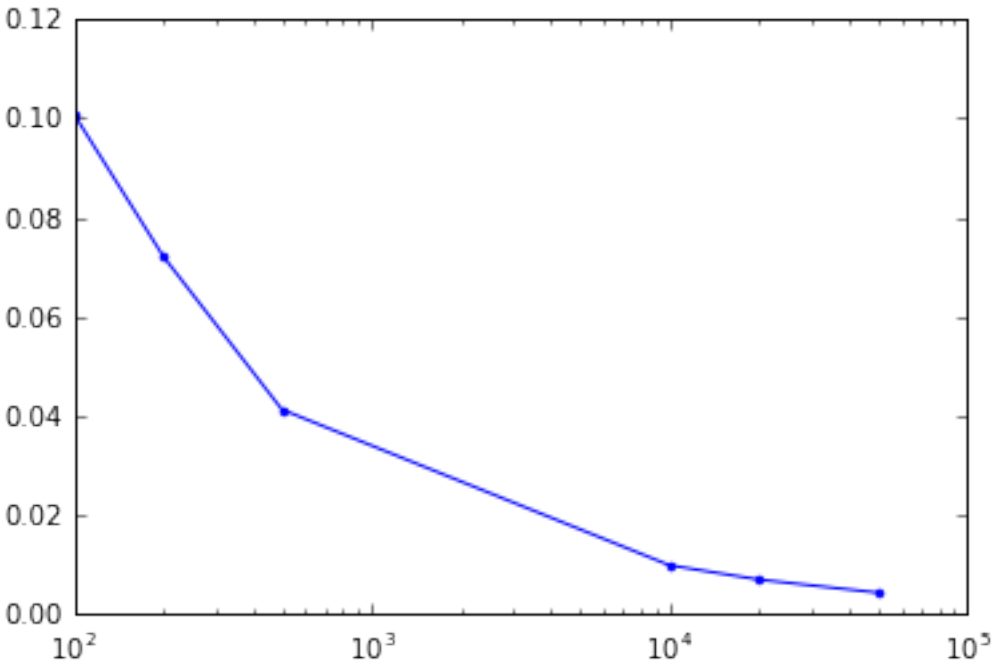
```
In [73]: data1 = np.random.normal(0,1,100)
         data2 = np.random.normal(0,1,1000)

         gs = mpl.gridspec.GridSpec(2,2)
         figH = plt.figure()
         axH = plt.subplot(gs[0,0])
```

```
plt.plot(data1,'.')
axH = plt.subplot(gs[0,1])
plt.plot(data2,'.')
plt.suptitle('Examples of data sets with different numbers of points')
```

Out[73]: <matplotlib.text.Text at 0x11aacfd68>

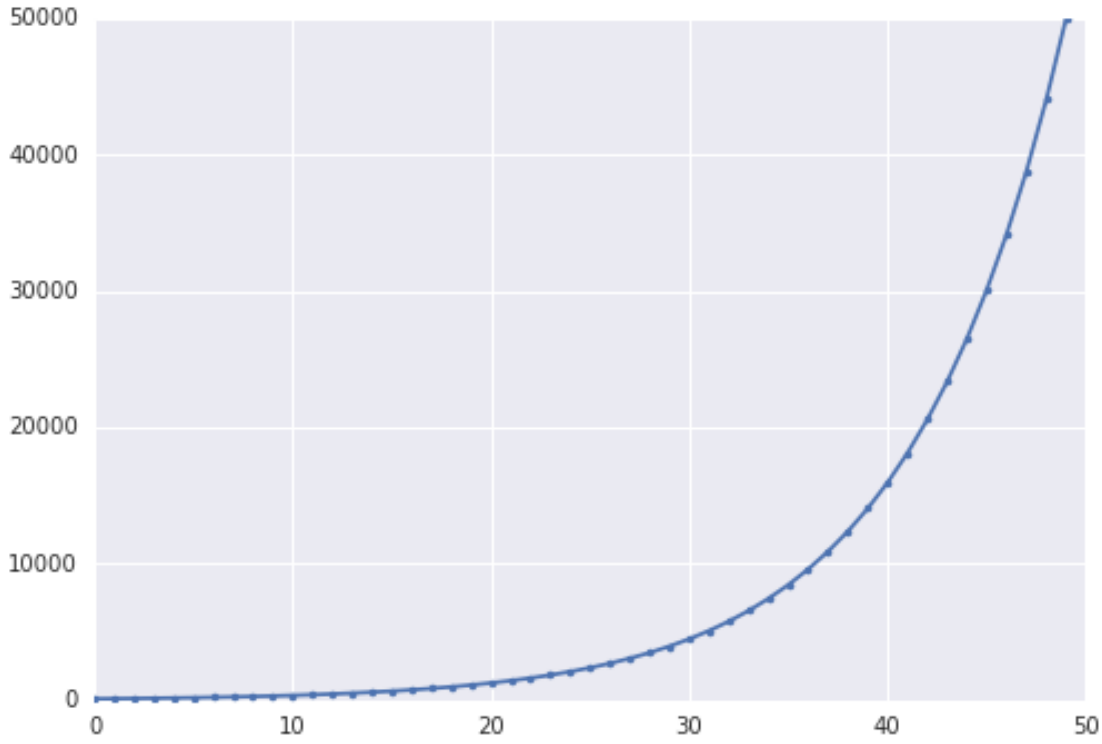


In [81]:
```
#xvals = [100,200,500,10000,20000,50000]
semV = np.zeros(len(xvals))*np.nan
for (iX,tX) in enumerate(xvals):
    semV[iX] = np.std(np.random.normal(0,1,[tX,]))/np.sqrt(tX)
#plt.plot(semV,'.-')
plt.plot(xvals, semV,'.-')
axH = plt.gca()
axH.set_xscale('log')
```



```
In [86]: xvals = [100,200,500,10000,20000,50000]
         xvals = np.logspace(np.log10(100),np.log10(50000))
         plt.plot(xvals, '.-')
```

```
Out[86]: [<matplotlib.lines.Line2D at 0x115d150f0>]
```

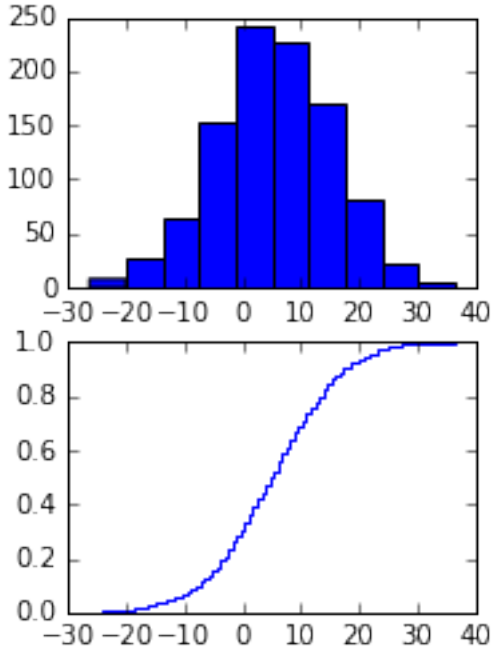



```
In [83]: import seaborn as sns
```

2.1 Hist vs CDF

```
In [68]: import statsmodels.api as sm # recommended according to the docs

         gs = mpl.gridspec.GridSpec(2,2)
         figH = plt.figure()
         axH = plt.subplot(gs[0,0])
         plt.hist(tArr)
         xmin,xmax = axH.get_xlim()

         axH = plt.subplot(gs[1,0])
         xvals = np.linspace(xmin, xmax, 100)
         p = sm.distributions.ECDF(tArr)(xvals)
         plt.step(xvals,p)
```

```
Out[68]: [<matplotlib.lines.Line2D at 0x119eb2278>]
```



2.2 Probability (combinatorics)

- Fair coin:
- prob of heads, tails?
- Biased coin: heads 0.2, tails?
- probability of 2 tails in a row?
- probability of 10 tails in a row?
- 4 sided die:
- probability of rolling a 1
- probability of rolling sum 3