Problem Set 1: Linear Regression

To run and solve this assignment, one must have a working IPython Notebook installation. The easiest way to set it up for both Windows and Linux is to install [Anaconda (https://www.continuum.io/downloads)](https://www.continuum.io/downloads). Then save this file to your computer (use "Raw" link on gist\github), run Anaconda and choose this file in Anaconda's file explorer. Use the `Python 3` version. Everything that follows assumes that you have already followed these instructions. If you are new to Python or its scientific library, Numpy, there are some nice tutorials [here (https://www.learnpython.org/)](https://www.learnpython.org/) and [here (http://www.scipy-lectures.org/)](http://www.scipy-lectures.org/).

To run code in a cell or to render [Markdown (https://en.wikipedia.org/wiki/Markdown)](https://en.wikipedia.org/wiki/Markdown)+[LaTeX (https://en.wikipedia.org/wiki/LaTeX)](https://en.wikipedia.org/wiki/LaTeX) press `Ctr+Enter` or `[>|]` (like "play") button above. To edit any code or text cell [double]click on its content. To change cell type, choose "Markdown" or "Code" in the drop-down menu above.

If certain output is given for some cells, that means that you are expected to get similar results.

Total: 155 points.

1. Numpy Tutorial

1.1 [5pt] Modify the cell below to return a 5x5 matrix of ones. Put some code there and press `Ctrl+Enter` to execute contents of the cell. You should see something like the output below. [[1] (https://docs.scipy.org/doc/numpy-1.13.0/user/basics.creation.html#arrays-creation)](https://docs.scipy.org/doc/numpy-1.13.0/user/basics.creation.html#arrays-creation) [[2] (https://docs.scipy.org/doc/numpy-1.13.0/reference/routines.array-creation.html#routines-array-creation)](https://docs.scipy.org/doc/numpy-1.13.0/reference/routines.array-creation.html#routines-array-creation)

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
array = (5,5)
def return_ones_matrix(a):
    return(np.ones(a))
print(return_ones_matrix(array))
```

```
[[1. 1. 1. 1. 1.]
 [1. 1. 1. 1. 1.]
 [1. 1. 1. 1. 1.]
 [1. 1. 1. 1. 1.]
 [1. 1. 1. 1. 1.]]
```

1.2 [5pt] Vectorizing your code is very important to get results in a reasonable time. Let A be a 10x10 matrix and x be a 10-element column vector. Your friend writes the following code. How would you vectorize this code to run without any for loops? Compare execution speed for different values of `n` with [`%timeit` (http://ipython.readthedocs.io/en/stable/interactive/magics.html#magic-timeit)](http://ipython.readthedocs.io/en/stable/interactive/magics.html#magic-timeit).

In [2]:
```
import timeit
```

In [3]:
```
import_module = "import random"
import numpy as np
n = 10
def compute_something(A, x):
    v = np.zeros((n, 1))
    for i in range(n):
        for j in range(n):
            v[i] += A[i, j] * x[j]
    return v

A = np.random.rand(n, n)
x = np.random.rand(n, 1)
print(compute_something(A, x))
starttime = timeit.default_timer()
print("The start time is :",starttime)
compute_something(A, x)
print("The time difference is :", timeit.default_timer() - starttime)

n = 20
A = np.random.rand(n, n)
x = np.random.rand(n, 1)
#print(compute_something(A, x))
print("n = 20")
starttime = timeit.default_timer()
print("The start time is :",starttime)
compute_something(A, x)
print("The time difference is :", timeit.default_timer() - starttime)

n = 30
A = np.random.rand(n, n)
x = np.random.rand(n, 1)
#print(compute_something(A, x))
print("n = 30")
starttime = timeit.default_timer()
print("The start time is :",starttime)
compute_something(A, x)
print("The time difference is :", timeit.default_timer() - starttime)

n = 100
A = np.random.rand(n, n)
x = np.random.rand(n, 1)
#print(compute_something(A, x))
print("n = 100")
starttime = timeit.default_timer()
print("The start time is :",starttime)
compute_something(A, x)
print("The time difference is :", timeit.default_timer() - starttime)

n = 999
A = np.random.rand(n, n)
x = np.random.rand(n, 1)
#print(compute_something(A, x))
print("n = 999")
starttime = timeit.default_timer()
print("The start time is :",starttime)
```

```
compute_something(A, x)
print("The time difference is :", timeit.default_timer() - starttime)
```

```
[[1.29719432]
 [2.64223855]
 [2.2405223 ]
 [1.97919377]
 [2.03236526]
 [1.35164744]
 [1.87205519]
 [2.06296264]
 [2.6449505 ]
 [1.92280936]]
The start time is : 3.3506072
The time difference is : 0.000759500000000024
n = 20
The start time is : 3.3517421
The time difference is : 0.0023952000000000417
n = 30
The start time is : 3.3549077
The time difference is : 0.004984599999999784
n = 100
The start time is : 3.3605191
The time difference is : 0.05819799999999997
n = 999
The start time is : 3.4364394
The time difference is : 5.253641100000001
```

In [4]:
```
arr1 =np.average(A,axis=1)
print(arr1)
```

```
[0.48011906 0.500943   0.49994264 0.5065809  0.5186723  0.5051601
 0.4914489  0.50074796 0.4967449  0.47682008 0.50763748 0.50431852
 0.492269   0.49088086 0.50961558 0.51078472 0.50023503 0.47897583
 0.48795897 0.48934088 0.50942589 0.49873235 0.50093368 0.5048258
 0.51259009 0.52234481 0.52017165 0.51452392 0.5004423  0.50588486
 0.51215791 0.50282312 0.50737188 0.50523257 0.52006186 0.50245537
 0.4941509  0.49315175 0.50018734 0.49658375 0.49856474 0.50767751
 0.50116863 0.50844115 0.48521878 0.49382266 0.50011071 0.4899011
 0.4999998  0.50214445 0.49723477 0.51284968 0.51068744 0.50585478
 0.49784929 0.49276001 0.51849027 0.52485584 0.49260569 0.49356829
 0.48320568 0.51012451 0.50641767 0.50186615 0.51182585 0.50615919
 0.49355376 0.50410051 0.50159911 0.50697287 0.50206654 0.48925562
 0.50713699 0.48386161 0.50462799 0.48842598 0.50050831 0.50106503
 0.49742632 0.49519067 0.48875509 0.49382956 0.49884369 0.49511935
 0.52212055 0.49255135 0.51009788 0.50982309 0.49729201 0.50056276
 0.51026206 0.49240121 0.50310172 0.49961636 0.4867487  0.50700766
 0.4876037  0.48898674 0.50913581 0.4923542  0.50067206 0.5038228
 0.49687693 0.51590417 0.4815789  0.51350417 0.49388522 0.48602845
 0.5029147  0.49427525 0.48927626 0.50934543 0.4962771  0.50913486
 0.49947847 0.49625021 0.50273377 0.49919562 0.50324586 0.50259827
 0.49615607 0.50097857 0.49690129 0.4967599  0.48331149 0.49327386
 0.49415836 0.50295917 0.4937156  0.5025207  0.49128177 0.48649181
 0.5031322  0.50490358 0.486538   0.50308898 0.49328478 0.49096984
 0.50937375 0.49673708 0.50831289 0.49504389 0.49404776 0.51245464
 0.48320687 0.49110815 0.51079363 0.49653407 0.51382306 0.49649938
 0.50169835 0.4981662  0.48413608 0.48834747 0.4994247  0.49389076
 0.49906105 0.49675229 0.50812068 0.50610892 0.50261311 0.50393211
 0.50795978 0.48973902 0.50863447 0.50223661 0.49982348 0.49488539
 0.50323229 0.50377465 0.49982473 0.48821136 0.51080138 0.49087905
 0.49908458 0.48219425 0.51028154 0.52256944 0.50898323 0.5016933
 0.5066256  0.49991669 0.50556975 0.49512842 0.50331982 0.50428577
 0.49976835 0.49258434 0.49953836 0.49855126 0.50846001 0.48834149
 0.50725405 0.49520179 0.48696185 0.48662134 0.49036779 0.4994912
 0.48880272 0.50454441 0.49858062 0.49070433 0.50629497 0.49286268
 0.50791704 0.50293692 0.50423498 0.50711418 0.50972568 0.51191606
 0.50960702 0.50353636 0.49745126 0.51128765 0.50903703 0.49731389
 0.50666495 0.49142283 0.5116652  0.4804733  0.50587877 0.50282283
 0.49803623 0.50269985 0.49859401 0.48088249 0.4975658  0.48721151
 0.49912447 0.49952159 0.51132531 0.49914228 0.5096641  0.49922454
 0.50948265 0.49367361 0.50865965 0.48706356 0.48808492 0.49837492
 0.49494167 0.48615089 0.51299233 0.4862416  0.48962956 0.50034797
 0.49770949 0.51402359 0.49943918 0.49412902 0.48935278 0.50022086
 0.51439453 0.49279172 0.49970303 0.49833174 0.49709457 0.50331691
 0.50308278 0.49208686 0.49269993 0.51475271 0.51271254 0.49720926
 0.49207542 0.50028004 0.504218   0.50920159 0.50717497 0.50076899
 0.50037496 0.50501595 0.50062042 0.51538035 0.50107981 0.49713574
 0.50028921 0.50471518 0.50536509 0.4838829  0.48294593 0.49174478
 0.50500966 0.50438816 0.48993899 0.50405956 0.493061   0.48913398
 0.51508746 0.50154733 0.50096314 0.50143749 0.48482327 0.50436427
 0.49543729 0.49181649 0.50518965 0.5014248  0.49147954 0.49124465
 0.5023383  0.48779977 0.50745667 0.50453436 0.49606165 0.49115272
 0.50197003 0.5129781  0.51655765 0.50068997 0.49719655 0.49833546
 0.50389231 0.48920957 0.51044349 0.49242459 0.50577566 0.50110978
 0.48777508 0.49390486 0.50977794 0.4867208  0.51172175 0.4845532
 0.49354503 0.49385105 0.49890065 0.51290338 0.50922853 0.50970539
 0.48930732 0.49511722 0.49400487 0.49787339 0.49422705 0.5053827
 0.50921278 0.49946344 0.49836514 0.50135488 0.51478715 0.49299798
```

```
0.50360973 0.52561079 0.49616294 0.50051284 0.49237044 0.49555143
0.50333956 0.51541676 0.49502411 0.48736037 0.4992346  0.49224749
0.49267191 0.49283017 0.49848019 0.50124903 0.4920424  0.49190731
0.50813163 0.49810919 0.50175714 0.4942661  0.49659527 0.50223591
0.48884856 0.49252112 0.49683512 0.50119503 0.51562678 0.4966654
0.49711849 0.51924175 0.48276656 0.50919862 0.49150149 0.48877996
0.50871606 0.48725136 0.50902116 0.49317814 0.49829636 0.51632529
0.48798755 0.50525397 0.49246081 0.49410043 0.49759973 0.50283882
0.50066222 0.50842597 0.49740081 0.50221506 0.48777996 0.5107111
0.51690052 0.48449152 0.49231022 0.48678291 0.48871254 0.50463213
0.49937101 0.49790195 0.51062144 0.49292871 0.49133761 0.50925481
0.4943698  0.51343194 0.50852725 0.50174822 0.4928362  0.50226559
0.51096779 0.49449711 0.49580452 0.49284879 0.51782254 0.49139114
0.5073191  0.4855636  0.504397   0.51791192 0.50085108 0.49780227
0.50087081 0.48242826 0.49483009 0.50562082 0.50814234 0.50017373
0.49415018 0.49016022 0.48836397 0.50937355 0.4933808  0.51406731
0.49178408 0.5018365  0.49230549 0.50056342 0.50608314 0.50297434
0.50323076 0.49682424 0.48791782 0.491256   0.50910339 0.50257353
0.4822575  0.48869851 0.48561102 0.49253662 0.51409652 0.49501523
0.49327483 0.48960692 0.48948454 0.50896346 0.50806958 0.49851543
0.51360435 0.48815476 0.49544708 0.51648796 0.51157809 0.50315829
0.49333503 0.50201308 0.50274838 0.49587301 0.50071496 0.49490803
0.50362599 0.50045739 0.49059955 0.48442263 0.50033847 0.51055442
0.50547267 0.500805   0.49795099 0.4975644  0.50018621 0.50484893
0.5107237  0.50661593 0.48621603 0.49967574 0.50116805 0.52087148
0.50185315 0.49357109 0.50700878 0.50242086 0.5022093  0.49550776
0.4870325  0.49325426 0.50280749 0.49366915 0.4996176  0.49018814
0.4978999  0.50619559 0.50795691 0.48882594 0.51952322 0.48343506
0.49161037 0.49973216 0.50279361 0.49869578 0.49685262 0.49133162
0.49552745 0.50685722 0.49503454 0.49905909 0.48205836 0.49168167
0.50414214 0.4998652  0.50075872 0.50225419 0.48985685 0.49547791
0.4860599  0.49793723 0.49921692 0.50247518 0.49999622 0.49352558
0.50533284 0.50114477 0.49121163 0.49683683 0.52222054 0.51517099
0.51163643 0.48449129 0.51095787 0.5076849  0.48868209 0.49820563
0.51603722 0.48803686 0.49307987 0.50904146 0.49467069 0.51408834
0.48530136 0.49571087 0.51758337 0.48021651 0.50938391 0.4873177
0.50140622 0.49461075 0.49486694 0.51162897 0.49408604 0.49290189
0.49926355 0.48762956 0.49356503 0.49712436 0.49891804 0.49547569
0.49175958 0.49701645 0.50294533 0.49787834 0.50103602 0.50440524
0.51007034 0.50010058 0.50678842 0.49480381 0.49467889 0.50299224
0.500255   0.50079257 0.50866781 0.4947774  0.5025384  0.47822879
0.50600712 0.5142694  0.49257294 0.50484858 0.49779219 0.50267789
0.51118998 0.49697653 0.49973593 0.50956594 0.5132434  0.5024335
0.49552947 0.50642163 0.49056123 0.49475623 0.49870498 0.50455252
0.49609095 0.50679845 0.501295   0.50524355 0.50584171 0.50288668
0.49276894 0.5109969  0.50516282 0.49859332 0.50665393 0.50454775
0.49624577 0.48905971 0.50552065 0.50657958 0.50277756 0.50914
0.49391736 0.49674701 0.50369475 0.49026643 0.50786722 0.51085425
0.49574917 0.49103387 0.50308286 0.48932659 0.49899853 0.49882744
0.50259246 0.50189327 0.4975403  0.50069706 0.49411616 0.4965793
0.5024628  0.50880024 0.50979997 0.50176611 0.50475754 0.48706962
0.5046634  0.49462172 0.50498279 0.48450561 0.50147568 0.50758669
0.51345726 0.50376898 0.49818375 0.49510115 0.48471148 0.49008954
0.50501469 0.51077267 0.50329452 0.5036019  0.50157828 0.49838549
0.50436231 0.50478342 0.4917168  0.5039792  0.4943945  0.49551368
0.522132   0.50571358 0.50594144 0.51597306 0.49381079 0.49622562
0.50517104 0.50397523 0.51126389 0.48394513 0.49558232 0.50292648
```

```
0.51330212 0.50223305 0.50311294 0.4993309  0.49752633 0.51539839
0.49906379 0.50451205 0.50014563 0.50513384 0.51610928 0.48961348
0.49961013 0.49795874 0.5052914  0.50173829 0.50390579 0.49214055
0.48728053 0.50924028 0.48208052 0.50320976 0.50490742 0.49014051
0.49274312 0.51016949 0.47746104 0.51862861 0.50267459 0.499577
0.49403115 0.49797572 0.48270183 0.48989672 0.48985922 0.50048368
0.48119082 0.51077982 0.48812845 0.4851953  0.50245699 0.50315592
0.5207903  0.48453239 0.50769521 0.50878398 0.49789184 0.50077437
0.50207992 0.50350957 0.50785879 0.50773397 0.50154371 0.51628612
0.49229642 0.49512777 0.49931567 0.50203871 0.51857216 0.49297367
0.48420348 0.48937155 0.49356326 0.50684548 0.48941593 0.49765643
0.48167893 0.51138272 0.49841516 0.49460475 0.49004999 0.5042683
0.49941958 0.50168102 0.50321286 0.49531005 0.49876257 0.50094371
0.51038949 0.501325   0.49477732 0.48811972 0.5012456  0.50490419
0.49733115 0.48621905 0.50236576 0.48412665 0.49836942 0.50335165
0.48994926 0.50167136 0.50702296 0.50905911 0.5214108  0.49755949
0.49597489 0.49333063 0.49962378 0.49288196 0.49211942 0.48934995
0.48026167 0.5101296  0.50477241 0.50268007 0.50532215 0.50811701
0.49141651 0.49840191 0.4812838  0.50634738 0.49643835 0.50286319
0.49896405 0.50363446 0.50158135 0.49785685 0.5134923  0.50322685
0.50188386 0.50143614 0.49063998 0.50004917 0.51108075 0.51904062
0.5042962  0.4954333  0.49650324 0.49764464 0.49402753 0.49657
0.48023182 0.50183025 0.50304761 0.49898253 0.49504102 0.50572028
0.5040802  0.49899868 0.51832553 0.50757366 0.5061623  0.49386828
0.49068026 0.48996997 0.49702828 0.48755724 0.50653401 0.49902654
0.49767147 0.51153215 0.50191508 0.49636803 0.48556724 0.49594682
0.49158376 0.50403004 0.50596697 0.49781997 0.4974105  0.51129609
0.49544326 0.49467926 0.48809076 0.4978063  0.5077752  0.49801566
0.50663435 0.51435656 0.49695834 0.50056621 0.49806862 0.48900775
0.51278283 0.49581925 0.49301058 0.50028673 0.5159046  0.47538284
0.50973104 0.49463601 0.49823397 0.49691506 0.5005525  0.48656573
0.48899244 0.48482223 0.4974723  0.49486045 0.48136143 0.50708764
0.498834   0.49162027 0.50104856 0.51396505 0.48966879 0.4834556
0.50031446 0.48659821 0.48152649 0.50882516 0.50599191 0.49796522
0.5003341  0.5020116  0.50733956 0.48690807 0.48569842 0.51740555
0.4920385  0.49881275 0.49986383 0.49080813 0.50910532 0.49295903
0.48888062 0.52274849 0.52322479 0.49987656 0.50411034 0.49725864
0.49462701 0.48312969 0.51868401 0.49069476 0.50914793 0.49271231
0.49985194 0.50171568 0.49020336 0.50069261 0.49658512 0.49448823
0.49107058 0.49911618 0.48490738 0.51884176 0.49240574 0.50376078
0.4848653  0.50734687 0.49752337 0.49925666 0.50480399 0.51017733
0.48554616 0.50927886 0.47871949 0.49509417 0.49516492 0.49063622
0.5004203  0.49889931 0.51042872 0.49604449 0.51497997 0.51002333
0.49031083 0.49991478 0.49706757 0.52020986 0.49630858 0.49100286
0.50182159 0.49982472 0.48921312 0.48930238 0.49518354 0.50968457
0.49258305 0.50867633 0.5047099  0.5086008  0.48884985 0.50517434
0.50141095 0.48203476 0.49942626 0.50280339 0.49105983 0.5060853
0.50665204 0.49408884 0.48167454 0.50220458 0.48182414 0.50846098
0.51077267 0.4871399  0.51090758 0.50007007 0.49657029 0.48764768
0.51609858 0.50574673 0.51202512 0.49655423 0.50004619 0.5058973
0.50191539 0.50437676 0.51307177 0.5065532  0.51406586 0.48767902
0.50473506 0.51217731 0.48816288 0.50029901 0.4862821  0.49365726
0.51156893 0.50915938 0.49200389]
```

In [5]:
```
n = 10
A = np.random.rand(n, n)
x = np.random.rand(n, 1)
def vectorized(A, x):
    return np.matmul(A,x)
print(vectorized(A, x))
assert np.max(abs(vectorized(A, x) - compute_something(A, x))) < 1e-3
print("To vectorize without loops, use numpy matrix multiplication. As you can multiply 1x10 matrix with 10x10 matrix.")
```

```
[[2.08901783]
 [1.449847  ]
 [1.70128058]
 [2.41020491]
 [1.64513535]
 [1.74812242]
 [3.30006839]
 [3.1264574 ]
 [2.50606332]
 [1.67370486]]
To vectorize without loops, use numpy matrix multiplication. As you can multi
ply 1x10 matrix with 10x10 matrix.
```

In [6]:
```
for n in [5, 10, 100, 500]:
    A = np.random.rand(n, n)
    x = np.random.rand(n, 1)
    %timeit -n 5 compute_something(A, x)
    %timeit -n 5 vectorized(A, x)
    print('---')
```

```
135 µs ± 21.9 µs per loop (mean ± std. dev. of 7 runs, 5 loops each)
3.21 µs ± 1.76 µs per loop (mean ± std. dev. of 7 runs, 5 loops each)
---
472 µs ± 25.7 µs per loop (mean ± std. dev. of 7 runs, 5 loops each)
5.55 µs ± 3.76 µs per loop (mean ± std. dev. of 7 runs, 5 loops each)
---
74.5 ms ± 4.01 ms per loop (mean ± std. dev. of 7 runs, 5 loops each)
6.88 µs ± 2.42 µs per loop (mean ± std. dev. of 7 runs, 5 loops each)
---
1.8 s ± 12.9 ms per loop (mean ± std. dev. of 7 runs, 5 loops each)
The slowest run took 21.37 times longer than the fastest. This could mean tha
t an intermediate result is being cached.
264 µs ± 478 µs per loop (mean ± std. dev. of 7 runs, 5 loops each)
---
```

2. Linear regression with one variable

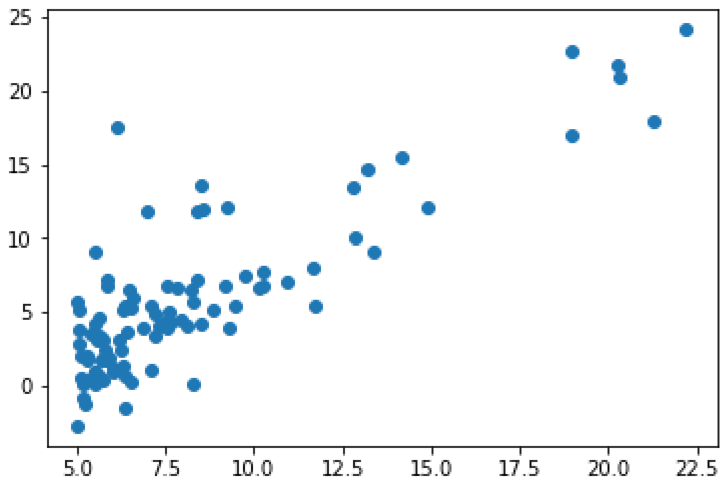
In this part of the exercise, you will implement linear regression with one variable to predict profits for a food truck. Suppose you are the CEO of a restaurant franchise and are considering different cities for opening a new outlet. The chain already has trucks in various cities and you have data for profits and populations from the cities. You would like to use this data to help you select which city to expand to next. The file ex1data.txt contains the dataset for our linear regression problem. The first column is the population of a city and the second column is the profit of a food truck in that city. A negative value for profit indicates a loss.

2.1 [10pt] Generate a plot similar to the one below : [1] (https://matplotlib.org/devdocs/api/_as_gen/matplotlib.pyplot.scatter.html) [2] (https://matplotlib.org/api/pyplot_api.html?highlight=xlim#matplotlib.pyplot.xlim) [3] (https://matplotlib.org/api/pyplot_api.html?highlight=matplotlib%20pyplot%20xlabel#matplotlib.pyplot.xlabel)

Before starting on any task, it is often useful to understand the data by visualizing it. For this dataset, you can use a scatter plot to visualize the data, since it has only two properties to plot (profit and population). Many other problems that you will encounter in real life are multi-dimensional and can’t be plotted on a 2-d plot.

In [7]:
```
data = np.loadtxt('ex1data1.txt', delimiter=',')
X, y = data[:, 0, np.newaxis], data[:, 1, np.newaxis]
n = data.shape[0]
print(X.shape, y.shape, n)
print(X[:10], '\n', y[:10])

import matplotlib.pyplot as plt
plt.scatter(X,y)
#raise NotImplementedError('Put the visualziation code here.')

plt.show()
```

```
(97, 1) (97, 1) 97
[[6.1101]
 [5.5277]
 [8.5186]
 [7.0032]
 [5.8598]
 [8.3829]
 [7.4764]
 [8.5781]
 [6.4862]
 [5.0546]]
 [[17.592 ]
 [ 9.1302]
 [13.662 ]
 [11.854 ]
 [ 6.8233]
 [11.886 ]
 [ 4.3483]
 [12.    ]
 [ 6.5987]
 [ 3.8166]]
```



2.2 Gradient Descent

In this part, you will fit the linear regression parameter θ to our dataset using gradient descent.

The objective of linear regression is to minimize the cost function

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} \left(h(x^{(i)}; \theta) - y^{(i)} \right)^2$$

where the hypothesis $h(x; \theta)$ is given by the linear model (x' has an additional fake feature always equal to '1')

$$h(x; \theta) = \theta^T x' = \theta_0 + \theta_1 x$$

Recall that the parameters of your model are the θ_j values. These are the values you will adjust to minimize the cost J(θ). One way to do this is to use the gradient descent. In batch gradient descent algorithm, value of θ is updated iteratively using the gradient of J(θ).

$$\theta_j^{(k+1)} = \theta_j^{(k)} - \eta \frac{1}{m} \sum_i \left(h(x^{(i)}; \theta) - y^{(i)} \right) x_j^{(i)}$$

With each step of gradient descent, your parameter θ_j comes closer to the optimal values that will achieve the lowest cost J(θ).

2.2.1 [5pt] Where does this update rule come from?

2.2.2 [30pt] Cost Implementation

As you perform gradient descent to learn to minimize the cost function, it is helpful to monitor the convergence by computing the cost. In this section, you will implement a function to calculate $J(\theta)$ so you can check the convergence of your gradient descent implementation.

In the following lines, we add another dimension to our data to accommodate the intercept term and compute the prediction and the loss. As you are doing this, remember that the variables X and y are not scalar values, but matrices whose rows represent the examples from the training set. In order to get x' add a column (https://docs.scipy.org/doc/numpy/reference/generated/numpy.insert.html) of ones to the data matrix `X` .

You should expect to see a cost of approximately 32.

The update rule comes from taking the derivative of the linear model. We want to slowly decrease our cost to test for the most optimal lowest cost function value. (slopes of slopes of slopes) These will be turned into weights to slowly test the values/outputs.

In [8]:
```
import math

def sigmoid(x):
  return 1 / (1 + math.exp(-x))
```

In [9]:
```
np.array([X[0]]).shape
```

Out[9]:
```
(1, 1)
```

In [12]:
```
predict(np.array([X[0]]), theta_init)
```

Out[12]:
```
array([[0.]])
```

In [11]:
```
# assertions below are true only for this
# specific case and are given to ease debugging!
import numpy as np
def add_column(X):
    assert len(X.shape) == 2 and X.shape[1] == 1
    return np.insert(X,0,1,axis=1)
    #raise NotImplementedError("Insert a column of ones to the _left_ side of
 the matrix")

def predict(X, theta):
    """ Computes h(x; theta) """
    assert len(X.shape) == 2 and X.shape[1] == 1
    assert theta.shape == (2, 1)
    #print(np.array(X).shape)
    X_prime = add_column(X)
    Theta_T = np.transpose(theta)
    X_prime = np.transpose(X_prime)
    pred = np.matmul(Theta_T,X_prime)
    return pred

def loss(X, y, theta):
    assert X.shape == (n, 1)
    assert y.shape == (n, 1)
    assert theta.shape == (2, 1)

    X_prime = add_column(X)
    assert X_prime.shape == (n, 2)
    Theta_T = np.transpose(theta)
    #X_prime = np.transpose(X_prime)
    #print(X_prime.shape)
    print(np.array(theta))
    print(theta.shape)
    #print(Theta_T.shape)
    #print(Theta_T)
    total = 0
    for i in range(len(X_prime)):
        pred = np.matmul(Theta_T,X_prime[i])
        total += (pred-y[i])**2
    #raise NotImplementedError("Compute the model loss; use the predict() func
tion")
    loss = total/194
    return loss

theta_init = np.zeros((2, 1))
print(loss(X, y, theta_init))
```

```
[[0.]
 [0.]]
(2, 1)
[32.07273388]
```

2.2.3 [40pt] GD Implementation

Next, you will implement gradient descent. The loop structure has been written for you, and you only need to supply the updates to θ within each iteration.

As you write your code, make sure you understand what you are trying to optimize and what is being updated. Keep in mind that the cost is parameterized by the vector θ not X and y. That is, we minimize the value of $J(\theta)$ by changing the values of the vector θ, not by changing X or y.

A good way to verify that gradient descent is working correctly is to look at the value of $J(\theta)$ and check that it is decreasing with each step. Your value of $J(\theta)$ should never increase, and should converge to a steady value by the end of the algorithm. Another way of making sure your gradient estimate is correct is to check it againts a [finite difference (https://en.wikipedia.org/wiki/Finite_difference)](https://en.wikipedia.org/wiki/Finite_difference) approximation.

We also initialize the initial parameters to 0 and the learning rate alpha to `0.01` .

In [17]:
```
np.array([X_prime[0]]).T.shape
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-17-0ac924ef0532> in <module>
----> 1 np.array([X_prime[0]]).T.shape

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

In [16]:
```
import scipy.optimize
from functools import partial

def loss_gradient(X, y, theta):
    X_prime = add_column(X)
    Theta_T = np.transpose(theta)
    total = 0
    for i in range(len(X_prime)):
        pred = np.matmul(Theta_T,X_prime[i])
        predactualminustest = (pred - y[i])
        total += (np.matmul(predactualminustest,np.array([X_prime[i]])))
    loss_grad = total/97
    return np.array([loss_grad]).T
assert loss_gradient(X, y, theta_init).shape == (2, 1)

def finite_diff_grad_check(f, grad, points, eps=1e-10):
    errs = []
    for point in points:
        point_errs = []
        grad_func_val = grad(point)
        for dim_i in range(point.shape[0]):
            diff_v = np.zeros_like(point)
            diff_v[dim_i] = eps
            dim_grad = (f(point+diff_v) - f(point-diff_v))/(2*eps)
            point_errs.append(abs(dim_grad - grad_func_val[dim_i]))
        errs.append(point_errs)
    return errs

test_points = [np.random.rand(2, 1) for _ in range(10)]
finite_diff_errs = finite_diff_grad_check(
    partial(loss, X, y), partial(loss_gradient, X, y), test_points
)

print('max grad comp error', np.max(finite_diff_errs))
assert np.max(finite_diff_errs) < 1e-3, "grad computation error is too large"

def run_gd(loss, loss_gradient, X, y, theta_init, lr=0.01, n_iter=1500):
    theta_current = theta_init.copy()
    loss_values = []
    theta_values = []

    for i in range(n_iter):
        loss_value = loss(X, y, theta_current)
        #lg = loss_gradient(X, y, theta_init)
        #multiply = lr/97

        theta_current = theta_current - lr*loss_gradient(X, y, theta_current)
        loss_values.append(loss_value)
        theta_values.append(theta_current)

    return theta_current, loss_values, theta_values

result = run_gd(loss, loss_gradient, X, y, theta_init)
theta_est, loss_values, theta_values = result

print('estimated theta value', theta_est.ravel())
```

```
print('resulting loss', loss(X, y, theta_est))
plt.ylabel('loss')
plt.xlabel('iter_i')
plt.plot(loss_values)
plt.show()

plt.ylabel('log(loss)')
plt.xlabel('iter_i')
plt.semilogy(loss_values)
plt.show()
```

```
[[0.73749812]
 [0.8689889 ]]
(2, 1)
[[0.73749812]
 [0.8689889 ]]
(2, 1)
[[0.73749812]
 [0.8689889 ]]
(2, 1)
[[0.73749812]
 [0.8689889 ]]
(2, 1)
[[0.07944256]
 [0.15296409]]
(2, 1)
[[0.07944256]
 [0.15296409]]
(2, 1)
[[0.07944256]
 [0.15296409]]
(2, 1)
[[0.07944256]
 [0.15296409]]
(2, 1)
[[0.64165013]
 [0.6608586 ]]
(2, 1)
[[0.64165013]
 [0.6608586 ]]
(2, 1)
[[0.64165013]
 [0.6608586 ]]
(2, 1)
[[0.64165013]
 [0.6608586 ]]
(2, 1)
[[0.37375157]
 [0.89639412]]
(2, 1)
[[0.37375157]
 [0.89639412]]
(2, 1)
[[0.37375157]
 [0.89639412]]
(2, 1)
[[0.37375157]
 [0.89639411]]
(2, 1)
[[0.2163608 ]
 [0.39851535]]
(2, 1)
[[0.2163608 ]
 [0.39851535]]
(2, 1)
[[0.2163608 ]
 [0.39851535]]
(2, 1)
```

```
[[0.2163608 ]
 [0.39851535]]
(2, 1)
[[0.46208463]
 [0.79571582]]
(2, 1)
[[0.46208463]
 [0.79571582]]
(2, 1)
[[0.46208463]
 [0.79571582]]
(2, 1)
[[0.46208463]
 [0.79571582]]
(2, 1)
[[0.36616392]
 [0.76480951]]
(2, 1)
[[0.36616392]
 [0.76480951]]
(2, 1)
[[0.36616392]
 [0.76480951]]
(2, 1)
[[0.36616392]
 [0.76480951]]
(2, 1)
[[0.17664974]
 [0.67444911]]
(2, 1)
[[0.17664974]
 [0.67444911]]
(2, 1)
[[0.17664974]
 [0.67444911]]
(2, 1)
[[0.17664974]
 [0.67444911]]
(2, 1)
[[0.3011567 ]
 [0.13285569]]
(2, 1)
[[0.3011567 ]
 [0.13285569]]
(2, 1)
[[0.3011567 ]
 [0.13285569]]
(2, 1)
[[0.3011567 ]
 [0.13285569]]
(2, 1)
[[0.02831584]
 [0.39051142]]
(2, 1)
[[0.02831584]
 [0.39051142]]
(2, 1)
```

```
[[0.02831584]
 [0.39051142]]
(2, 1)
[[0.02831584]
 [0.39051142]]
(2, 1)
max grad comp error 0.00012648280149818447
[[0.]
 [0.]]
(2, 1)
[[0.05839135]
 [0.6532885 ]]
(2, 1)
[[0.06289175]
 [0.77000978]]
(2, 1)
[[0.05782293]
 [0.79134812]]
(2, 1)
[[0.05106363]
 [0.79572981]]
(2, 1)
[[0.04401438]
 [0.79709618]]
(2, 1)
[[0.03692413]
 [0.79792547]]
(2, 1)
[[0.02983712]
 [0.79865824]]
(2, 1)
[[0.02276118]
 [0.79937279]]
(2, 1)
[[0.0156977 ]
 [0.80008305]]
(2, 1)
[[0.0086469]
 [0.8007915]]
(2, 1)
[[0.00160879]
 [0.80149857]]
(2, 1)
[[-0.00541662]
 [ 0.80220436]]
(2, 1)
[[-0.01242938]
 [ 0.80290886]]
(2, 1)
[[-0.01942949]
 [ 0.8036121 ]]
(2, 1)
[[-0.02641699]
 [ 0.80431407]]
(2, 1)
[[-0.03339189]
 [ 0.80501478]]
```

```
(2, 1)
[[-0.04035421]
 [ 0.80571422]]
(2, 1)
[[-0.04730399]
 [ 0.8064124 ]]
(2, 1)
[[-0.05424124]
 [ 0.80710932]]
(2, 1)
[[-0.06116598]
 [ 0.80780498]]
(2, 1)
[[-0.06807824]
 [ 0.8084994 ]]
(2, 1)
[[-0.07497804]
 [ 0.80919256]]
(2, 1)
[[-0.08186541]
 [ 0.80988447]]
(2, 1)
[[-0.08874035]
 [ 0.81057513]]
(2, 1)
[[-0.09560291]
 [ 0.81126455]]
(2, 1)
[[-0.10245309]
 [ 0.81195272]]
(2, 1)
[[-0.10929093]
 [ 0.81263966]]
(2, 1)
[[-0.11611644]
 [ 0.81332535]]
(2, 1)
[[-0.12292965]
 [ 0.81400981]]
(2, 1)
[[-0.12973057]
 [ 0.81469304]]
(2, 1)
[[-0.13651924]
 [ 0.81537504]]
(2, 1)
[[-0.14329567]
 [ 0.8160558 ]]
(2, 1)
[[-0.15005988]
 [ 0.81673534]]
(2, 1)
[[-0.15681191]
 [ 0.81741365]]
(2, 1)
[[-0.16355176]
 [ 0.81809075]]
```

```
(2, 1)
[[-0.17027946]
 [ 0.81876662]]
(2, 1)
[[-0.17699503]
 [ 0.81944127]]
(2, 1)
[[-0.1836985]
 [ 0.8201147]]
(2, 1)
[[-0.19038988]
 [ 0.82078693]]
(2, 1)
[[-0.1970692 ]
 [ 0.82145794]]
(2, 1)
[[-0.20373649]
 [ 0.82212774]]
(2, 1)
[[-0.21039175]
 [ 0.82279633]]
(2, 1)
[[-0.21703502]
 [ 0.82346372]]
(2, 1)
[[-0.22366631]
 [ 0.8241299 ]]
(2, 1)
[[-0.23028565]
 [ 0.82479489]]
(2, 1)
[[-0.23689305]
 [ 0.82545867]]
(2, 1)
[[-0.24348855]
 [ 0.82612126]]
(2, 1)
[[-0.25007216]
 [ 0.82678266]]
(2, 1)
[[-0.2566439 ]
 [ 0.82744286]]
(2, 1)
[[-0.26320379]
 [ 0.82810187]]
(2, 1)
[[-0.26975186]
 [ 0.8287597 ]]
(2, 1)
[[-0.27628812]
 [ 0.82941634]]
(2, 1)
[[-0.2828126 ]
 [ 0.83007179]]
(2, 1)
[[-0.28932533]
 [ 0.83072607]]
```

```
(2, 1)
[[-0.29582631]
 [ 0.83137916]]
(2, 1)
[[-0.30231557]
 [ 0.83203108]]
(2, 1)
[[-0.30879314]
 [ 0.83268182]]
(2, 1)
[[-0.31525902]
 [ 0.83333139]]
(2, 1)
[[-0.32171326]
 [ 0.83397978]]
(2, 1)
[[-0.32815586]
 [ 0.83462701]]
(2, 1)
[[-0.33458684]
 [ 0.83527308]]
(2, 1)
[[-0.34100624]
 [ 0.83591797]]
(2, 1)
[[-0.34741406]
 [ 0.83656171]]
(2, 1)
[[-0.35381033]
 [ 0.83720428]]
(2, 1)
[[-0.36019507]
 [ 0.8378457 ]]
(2, 1)
[[-0.3665683 ]
 [ 0.83848596]]
(2, 1)
[[-0.37293005]
 [ 0.83912507]]
(2, 1)
[[-0.37928032]
 [ 0.83976302]]
(2, 1)
[[-0.38561915]
 [ 0.84039982]]
(2, 1)
[[-0.39194656]
 [ 0.84103548]]
(2, 1)
[[-0.39826255]
 [ 0.84166999]]
(2, 1)
[[-0.40456717]
 [ 0.84230336]]
(2, 1)
[[-0.41086041]
 [ 0.84293558]]
```

```
(2, 1)
[[-0.41714232]
 [ 0.84356667]]
(2, 1)
[[-0.42341289]
 [ 0.84419662]]
(2, 1)
[[-0.42967217]
 [ 0.84482543]]
(2, 1)
[[-0.43592016]
 [ 0.84545311]]
(2, 1)
[[-0.44215689]
 [ 0.84607965]]
(2, 1)
[[-0.44838238]
 [ 0.84670507]]
(2, 1)
[[-0.45459665]
 [ 0.84732936]]
(2, 1)
[[-0.46079971]
 [ 0.84795253]]
(2, 1)
[[-0.4669916 ]
 [ 0.84857457]]
(2, 1)
[[-0.47317232]
 [ 0.84919549]]
(2, 1)
[[-0.4793419 ]
 [ 0.84981529]]
(2, 1)
[[-0.48550036]
 [ 0.85043397]]
(2, 1)
[[-0.49164771]
 [ 0.85105154]]
(2, 1)
[[-0.49778399]
 [ 0.851668  ]]
(2, 1)
[[-0.5039092 ]
 [ 0.85228334]]
(2, 1)
[[-0.51002338]
 [ 0.85289758]]
(2, 1)
[[-0.51612653]
 [ 0.85351071]]
(2, 1)
[[-0.52221868]
 [ 0.85412273]]
(2, 1)
[[-0.52829985]
 [ 0.85473365]]
```

```
(2, 1)
[[-0.53437006]
 [ 0.85534347]]
(2, 1)
[[-0.54042932]
 [ 0.85595219]]
(2, 1)
[[-0.54647767]
 [ 0.85655981]]
(2, 1)
[[-0.55251511]
 [ 0.85716633]]
(2, 1)
[[-0.55854166]
 [ 0.85777177]]
(2, 1)
[[-0.56455736]
 [ 0.85837611]]
(2, 1)
[[-0.57056221]
 [ 0.85897936]]
(2, 1)
[[-0.57655623]
 [ 0.85958153]]
(2, 1)
[[-0.58253945]
 [ 0.8601826 ]]
(2, 1)
[[-0.58851189]
 [ 0.8607826 ]]
(2, 1)
[[-0.59447356]
 [ 0.86138151]]
(2, 1)
[[-0.60042448]
 [ 0.86197935]]
(2, 1)
[[-0.60636467]
 [ 0.86257611]]
(2, 1)
[[-0.61229416]
 [ 0.86317179]]
(2, 1)
[[-0.61821296]
 [ 0.8637664 ]]
(2, 1)
[[-0.62412109]
 [ 0.86435993]]
(2, 1)
[[-0.63001857]
 [ 0.8649524 ]]
(2, 1)
[[-0.63590542]
 [ 0.8655438 ]]
(2, 1)
[[-0.64178166]
 [ 0.86613413]]
```

```
(2, 1)
[[-0.6476473 ]
 [ 0.86672339]]
(2, 1)
[[-0.65350238]
 [ 0.8673116 ]]
(2, 1)
[[-0.65934689]
 [ 0.86789875]]
(2, 1)
[[-0.66518088]
 [ 0.86848483]]
(2, 1)
[[-0.67100434]
 [ 0.86906986]]
(2, 1)
[[-0.67681731]
 [ 0.86965384]]
(2, 1)
[[-0.6826198 ]
 [ 0.87023676]]
(2, 1)
[[-0.68841183]
 [ 0.87081863]]
(2, 1)
[[-0.69419342]
 [ 0.87139946]]
(2, 1)
[[-0.69996459]
 [ 0.87197923]]
(2, 1)
[[-0.70572536]
 [ 0.87255796]]
(2, 1)
[[-0.71147574]
 [ 0.87313565]]
(2, 1)
[[-0.71721575]
 [ 0.8737123 ]]
(2, 1)
[[-0.72294542]
 [ 0.87428791]]
(2, 1)
[[-0.72866476]
 [ 0.87486248]]
(2, 1)
[[-0.73437379]
 [ 0.87543601]]
(2, 1)
[[-0.74007253]
 [ 0.87600851]]
(2, 1)
[[-0.745761  ]
 [ 0.87657998]]
(2, 1)
[[-0.75143921]
 [ 0.87715042]]
```

```
(2, 1)
[[-0.75710719]
 [ 0.87771983]]
(2, 1)
[[-0.76276495]
 [ 0.87828821]]
(2, 1)
[[-0.76841251]
 [ 0.87885557]]
(2, 1)
[[-0.77404989]
 [ 0.87942191]]
(2, 1)
[[-0.77967711]
 [ 0.87998722]]
(2, 1)
[[-0.78529418]
 [ 0.88055152]]
(2, 1)
[[-0.79090113]
 [ 0.8811148 ]]
(2, 1)
[[-0.79649798]
 [ 0.88167706]]
(2, 1)
[[-0.80208473]
 [ 0.88223831]]
(2, 1)
[[-0.80766141]
 [ 0.88279855]]
(2, 1)
[[-0.81322805]
 [ 0.88335778]]
(2, 1)
[[-0.81878464]
 [ 0.883916  ]]
(2, 1)
[[-0.82433122]
 [ 0.88447321]]
(2, 1)
[[-0.82986781]
 [ 0.88502942]]
(2, 1)
[[-0.83539441]
 [ 0.88558463]]
(2, 1)
[[-0.84091105]
 [ 0.88613883]]
(2, 1)
[[-0.84641774]
 [ 0.88669204]]
(2, 1)
[[-0.85191451]
 [ 0.88724425]]
(2, 1)
[[-0.85740137]
 [ 0.88779547]]
```

```
(2, 1)
[[-0.86287834]
 [ 0.88834569]]
(2, 1)
[[-0.86834544]
 [ 0.88889492]]
(2, 1)
[[-0.87380268]
 [ 0.88944316]]
(2, 1)
[[-0.87925009]
 [ 0.88999041]]
(2, 1)
[[-0.88468767]
 [ 0.89053667]]
(2, 1)
[[-0.89011546]
 [ 0.89108195]]
(2, 1)
[[-0.89553346]
 [ 0.89162625]]
(2, 1)
[[-0.90094169]
 [ 0.89216956]]
(2, 1)
[[-0.90634018]
 [ 0.8927119 ]]
(2, 1)
[[-0.91172893]
 [ 0.89325326]]
(2, 1)
[[-0.91710797]
 [ 0.89379364]]
(2, 1)
[[-0.92247731]
 [ 0.89433305]]
(2, 1)
[[-0.92783698]
 [ 0.89487149]]
(2, 1)
[[-0.93318698]
 [ 0.89540895]]
(2, 1)
[[-0.93852734]
 [ 0.89594545]]
(2, 1)
[[-0.94385807]
 [ 0.89648098]]
(2, 1)
[[-0.9491792 ]
 [ 0.89701554]]
(2, 1)
[[-0.95449073]
 [ 0.89754914]]
(2, 1)
[[-0.95979269]
 [ 0.89808178]]
```

```
(2, 1)
[[-0.96508509]
 [ 0.89861346]]
(2, 1)
[[-0.97036795]
 [ 0.89914418]]
(2, 1)
[[-0.97564128]
 [ 0.89967395]]
(2, 1)
[[-0.98090511]
 [ 0.90020276]]
(2, 1)
[[-0.98615946]
 [ 0.90073061]]
(2, 1)
[[-0.99140433]
 [ 0.90125752]]
(2, 1)
[[-0.99663975]
 [ 0.90178347]]
(2, 1)
[[-1.00186573]
 [ 0.90230848]]
(2, 1)
[[-1.00708228]
 [ 0.90283254]]
(2, 1)
[[-1.01228944]
 [ 0.90335565]]
(2, 1)
[[-1.01748721]
 [ 0.90387782]]
(2, 1)
[[-1.02267561]
 [ 0.90439906]]
(2, 1)
[[-1.02785466]
 [ 0.90491935]]
(2, 1)
[[-1.03302437]
 [ 0.9054387 ]]
(2, 1)
[[-1.03818476]
 [ 0.90595712]]
(2, 1)
[[-1.04333585]
 [ 0.9064746 ]]
(2, 1)
[[-1.04847766]
 [ 0.90699115]]
(2, 1)
[[-1.0536102 ]
 [ 0.90750677]]
(2, 1)
[[-1.05873348]
 [ 0.90802146]]
```

```
(2, 1)
[[-1.06384753]
 [ 0.90853522]]
(2, 1)
[[-1.06895236]
 [ 0.90904806]]
(2, 1)
[[-1.07404799]
 [ 0.90955997]]
(2, 1)
[[-1.07913444]
 [ 0.91007096]]
(2, 1)
[[-1.08421171]
 [ 0.91058102]]
(2, 1)
[[-1.08927983]
 [ 0.91109017]]
(2, 1)
[[-1.09433882]
 [ 0.9115984 ]]
(2, 1)
[[-1.09938869]
 [ 0.91210572]]
(2, 1)
[[-1.10442945]
 [ 0.91261212]]
(2, 1)
[[-1.10946113]
 [ 0.9131176 ]]
(2, 1)
[[-1.11448374]
 [ 0.91362218]]
(2, 1)
[[-1.11949729]
 [ 0.91412584]]
(2, 1)
[[-1.12450181]
 [ 0.9146286 ]]
(2, 1)
[[-1.12949731]
 [ 0.91513045]]
(2, 1)
[[-1.1344838]
 [ 0.9156314]]
(2, 1)
[[-1.1394613 ]
 [ 0.91613145]]
(2, 1)
[[-1.14442983]
 [ 0.91663059]]
(2, 1)
[[-1.14938941]
 [ 0.91712883]]
(2, 1)
[[-1.15434004]
 [ 0.91762618]]
```

```
(2, 1)
[[-1.15928175]
 [ 0.91812262]]
(2, 1)
[[-1.16421455]
 [ 0.91861818]]
(2, 1)
[[-1.16913846]
 [ 0.91911284]]
(2, 1)
[[-1.17405349]
 [ 0.91960661]]
(2, 1)
[[-1.17895967]
 [ 0.92009948]]
(2, 1)
[[-1.183857  ]
 [ 0.92059147]]
(2, 1)
[[-1.1887455 ]
 [ 0.92108258]]
(2, 1)
[[-1.19362519]
 [ 0.9215728 ]]
(2, 1)
[[-1.19849609]
 [ 0.92206213]]
(2, 1)
[[-1.2033582 ]
 [ 0.92255058]]
(2, 1)
[[-1.20821155]
 [ 0.92303815]]
(2, 1)
[[-1.21305615]
 [ 0.92352485]]
(2, 1)
[[-1.21789202]
 [ 0.92401066]]
(2, 1)
[[-1.22271917]
 [ 0.9244956 ]]
(2, 1)
[[-1.22753762]
 [ 0.92497967]]
(2, 1)
[[-1.23234739]
 [ 0.92546286]]
(2, 1)
[[-1.23714848]
 [ 0.92594518]]
(2, 1)
[[-1.24194092]
 [ 0.92642663]]
(2, 1)
[[-1.24672472]
 [ 0.92690722]]
```

```
(2, 1)
[[-1.2514999 ]
 [ 0.92738694]]
(2, 1)
[[-1.25626647]
 [ 0.92786579]]
(2, 1)
[[-1.26102445]
 [ 0.92834378]]
(2, 1)
[[-1.26577385]
 [ 0.92882091]]
(2, 1)
[[-1.27051469]
 [ 0.92929718]]
(2, 1)
[[-1.27524698]
 [ 0.92977259]]
(2, 1)
[[-1.27997074]
 [ 0.93024714]]
(2, 1)
[[-1.28468599]
 [ 0.93072084]]
(2, 1)
[[-1.28939274]
 [ 0.93119368]]
(2, 1)
[[-1.29409101]
 [ 0.93166568]]
(2, 1)
[[-1.2987808 ]
 [ 0.93213682]]
(2, 1)
[[-1.30346214]
 [ 0.93260711]]
(2, 1)
[[-1.30813505]
 [ 0.93307655]]
(2, 1)
[[-1.31279953]
 [ 0.93354515]]
(2, 1)
[[-1.3174556]
 [ 0.9340129]]
(2, 1)
[[-1.32210327]
 [ 0.93447981]]
(2, 1)
[[-1.32674258]
 [ 0.93494588]]
(2, 1)
[[-1.33137351]
 [ 0.93541111]]
(2, 1)
[[-1.3359961]
 [ 0.9358755]]
```

```
(2, 1)
[[-1.34061036]
 [ 0.93633905]]
(2, 1)
[[-1.3452163 ]
 [ 0.93680177]]
(2, 1)
[[-1.34981394]
 [ 0.93726365]]
(2, 1)
[[-1.35440329]
 [ 0.9377247 ]]
(2, 1)
[[-1.35898436]
 [ 0.93818492]]
(2, 1)
[[-1.36355718]
 [ 0.93864431]]
(2, 1)
[[-1.36812176]
 [ 0.93910287]]
(2, 1)
[[-1.37267811]
 [ 0.9395606 ]]
(2, 1)
[[-1.37722624]
 [ 0.94001751]]
(2, 1)
[[-1.38176618]
 [ 0.9404736 ]]
(2, 1)
[[-1.38629793]
 [ 0.94092886]]
(2, 1)
[[-1.39082151]
 [ 0.94138331]]
(2, 1)
[[-1.39533694]
 [ 0.94183693]]
(2, 1)
[[-1.39984423]
 [ 0.94228974]]
(2, 1)
[[-1.4043434 ]
 [ 0.94274172]]
(2, 1)
[[-1.40883445]
 [ 0.9431929 ]]
(2, 1)
[[-1.41331741]
 [ 0.94364326]]
(2, 1)
[[-1.41779229]
 [ 0.94409281]]
(2, 1)
[[-1.4222591 ]
 [ 0.94454155]]
```

```
(2, 1)
[[-1.42671786]
 [ 0.94498948]]
(2, 1)
[[-1.43116858]
 [ 0.94543661]]
(2, 1)
[[-1.43561128]
 [ 0.94588292]]
(2, 1)
[[-1.44004597]
 [ 0.94632844]]
(2, 1)
[[-1.44447267]
 [ 0.94677315]]
(2, 1)
[[-1.44889139]
 [ 0.94721705]]
(2, 1)
[[-1.45330214]
 [ 0.94766016]]
(2, 1)
[[-1.45770494]
 [ 0.94810247]]
(2, 1)
[[-1.46209981]
 [ 0.94854398]]
(2, 1)
[[-1.46648675]
 [ 0.9489847 ]]
(2, 1)
[[-1.47086579]
 [ 0.94942462]]
(2, 1)
[[-1.47523693]
 [ 0.94986375]]
(2, 1)
[[-1.47960019]
 [ 0.95030209]]
(2, 1)
[[-1.48395559]
 [ 0.95073963]]
(2, 1)
[[-1.48830314]
 [ 0.95117639]]
(2, 1)
[[-1.49264285]
 [ 0.95161236]]
(2, 1)
[[-1.49697473]
 [ 0.95204755]]
(2, 1)
[[-1.50129881]
 [ 0.95248195]]
(2, 1)
[[-1.50561509]
 [ 0.95291557]]
```

```
(2, 1)
[[-1.5099236]
 [ 0.9533484]]
(2, 1)
[[-1.51422433]
 [ 0.95378046]]
(2, 1)
[[-1.51851732]
 [ 0.95421173]]
(2, 1)
[[-1.52280256]
 [ 0.95464223]]
(2, 1)
[[-1.52708008]
 [ 0.95507196]]
(2, 1)
[[-1.53134989]
 [ 0.95550091]]
(2, 1)
[[-1.53561201]
 [ 0.95592908]]
(2, 1)
[[-1.53986644]
 [ 0.95635648]]
(2, 1)
[[-1.5441132 ]
 [ 0.95678312]]
(2, 1)
[[-1.5483523 ]
 [ 0.95720898]]
(2, 1)
[[-1.55258377]
 [ 0.95763408]]
(2, 1)
[[-1.55680761]
 [ 0.95805841]]
(2, 1)
[[-1.56102383]
 [ 0.95848197]]
(2, 1)
[[-1.56523245]
 [ 0.95890478]]
(2, 1)
[[-1.56943349]
 [ 0.95932682]]
(2, 1)
[[-1.57362695]
 [ 0.95974809]]
(2, 1)
[[-1.57781286]
 [ 0.96016861]]
(2, 1)
[[-1.58199122]
 [ 0.96058838]]
(2, 1)
[[-1.58616205]
 [ 0.96100738]]
```

```
(2, 1)
[[-1.59032536]
 [ 0.96142563]]
(2, 1)
[[-1.59448116]
 [ 0.96184313]]
(2, 1)
[[-1.59862947]
 [ 0.96225987]]
(2, 1)
[[-1.60277031]
 [ 0.96267586]]
(2, 1)
[[-1.60690368]
 [ 0.9630911 ]]
(2, 1)
[[-1.6110296]
 [ 0.9635056]]
(2, 1)
[[-1.61514808]
 [ 0.96391934]]
(2, 1)
[[-1.61925914]
 [ 0.96433234]]
(2, 1)
[[-1.62336279]
 [ 0.9647446 ]]
(2, 1)
[[-1.62745904]
 [ 0.96515611]]
(2, 1)
[[-1.63154791]
 [ 0.96556688]]
(2, 1)
[[-1.63562941]
 [ 0.96597691]]
(2, 1)
[[-1.63970355]
 [ 0.9663862 ]]
(2, 1)
[[-1.64377034]
 [ 0.96679476]]
(2, 1)
[[-1.64782981]
 [ 0.96720258]]
(2, 1)
[[-1.65188195]
 [ 0.96760966]]
(2, 1)
[[-1.6559268 ]
 [ 0.96801601]]
(2, 1)
[[-1.65996435]
 [ 0.96842162]]
(2, 1)
[[-1.66399462]
 [ 0.96882651]]
```

```
(2, 1)
[[-1.66801763]
 [ 0.96923066]]
(2, 1)
[[-1.67203339]
 [ 0.96963409]]
(2, 1)
[[-1.67604191]
 [ 0.97003679]]
(2, 1)
[[-1.6800432 ]
 [ 0.97043876]]
(2, 1)
[[-1.68403728]
 [ 0.97084001]]
(2, 1)
[[-1.68802416]
 [ 0.97124053]]
(2, 1)
[[-1.69200385]
 [ 0.97164034]]
(2, 1)
[[-1.69597637]
 [ 0.97203942]]
(2, 1)
[[-1.69994173]
 [ 0.97243778]]
(2, 1)
[[-1.70389994]
 [ 0.97283543]]
(2, 1)
[[-1.70785101]
 [ 0.97323236]]
(2, 1)
[[-1.71179497]
 [ 0.97362857]]
(2, 1)
[[-1.71573181]
 [ 0.97402407]]
(2, 1)
[[-1.71966156]
 [ 0.97441885]]
(2, 1)
[[-1.72358422]
 [ 0.97481293]]
(2, 1)
[[-1.72749981]
 [ 0.97520629]]
(2, 1)
[[-1.73140835]
 [ 0.97559895]]
(2, 1)
[[-1.73530984]
 [ 0.97599089]]
(2, 1)
[[-1.73920429]
 [ 0.97638213]]
```

```
(2, 1)
[[-1.74309173]
 [ 0.97677267]]
(2, 1)
[[-1.74697216]
 [ 0.9771625 ]]
(2, 1)
[[-1.75084559]
 [ 0.97755163]]
(2, 1)
[[-1.75471204]
 [ 0.97794006]]
(2, 1)
[[-1.75857152]
 [ 0.97832778]]
(2, 1)
[[-1.76242405]
 [ 0.97871481]]
(2, 1)
[[-1.76626963]
 [ 0.97910114]]
(2, 1)
[[-1.77010828]
 [ 0.97948677]]
(2, 1)
[[-1.77394001]
 [ 0.97987171]]
(2, 1)
[[-1.77776483]
 [ 0.98025596]]
(2, 1)
[[-1.78158275]
 [ 0.98063951]]
(2, 1)
[[-1.7853938 ]
 [ 0.98102237]]
(2, 1)
[[-1.78919797]
 [ 0.98140454]]
(2, 1)
[[-1.79299529]
 [ 0.98178602]]
(2, 1)
[[-1.79678576]
 [ 0.98216682]]
(2, 1)
[[-1.8005694 ]
 [ 0.98254693]]
(2, 1)
[[-1.80434622]
 [ 0.98292635]]
(2, 1)
[[-1.80811624]
 [ 0.98330509]]
(2, 1)
[[-1.81187945]
 [ 0.98368314]]
```

```
(2, 1)
[[-1.81563588]
 [ 0.98406052]]
(2, 1)
[[-1.81938554]
 [ 0.98443721]]
(2, 1)
[[-1.82312844]
 [ 0.98481323]]
(2, 1)
[[-1.8268646 ]
 [ 0.98518856]]
(2, 1)
[[-1.83059402]
 [ 0.98556322]]
(2, 1)
[[-1.83431672]
 [ 0.98593721]]
(2, 1)
[[-1.8380327 ]
 [ 0.98631052]]
(2, 1)
[[-1.84174199]
 [ 0.98668316]]
(2, 1)
[[-1.84544459]
 [ 0.98705512]]
(2, 1)
[[-1.84914052]
 [ 0.98742642]]
(2, 1)
[[-1.85282979]
 [ 0.98779705]]
(2, 1)
[[-1.8565124 ]
 [ 0.98816701]]
(2, 1)
[[-1.86018838]
 [ 0.9885363 ]]
(2, 1)
[[-1.86385773]
 [ 0.98890492]]
(2, 1)
[[-1.86752046]
 [ 0.98927289]]
(2, 1)
[[-1.8711766 ]
 [ 0.98964018]]
(2, 1)
[[-1.87482614]
 [ 0.99000682]]
(2, 1)
[[-1.87846911]
 [ 0.9903728 ]]
(2, 1)
[[-1.8821055 ]
 [ 0.99073811]]
```

```
(2, 1)
[[-1.88573535]
 [ 0.99110277]]
(2, 1)
[[-1.88935865]
 [ 0.99146677]]
(2, 1)
[[-1.89297542]
 [ 0.99183011]]
(2, 1)
[[-1.89658566]
 [ 0.9921928 ]]
(2, 1)
[[-1.9001894 ]
 [ 0.99255483]]
(2, 1)
[[-1.90378665]
 [ 0.99291622]]
(2, 1)
[[-1.90737741]
 [ 0.99327695]]
(2, 1)
[[-1.9109617 ]
 [ 0.99363703]]
(2, 1)
[[-1.91453952]
 [ 0.99399646]]
(2, 1)
[[-1.9181109 ]
 [ 0.99435524]]
(2, 1)
[[-1.92167584]
 [ 0.99471338]]
(2, 1)
[[-1.92523436]
 [ 0.99507087]]
(2, 1)
[[-1.92878645]
 [ 0.99542772]]
(2, 1)
[[-1.93233215]
 [ 0.99578392]]
(2, 1)
[[-1.93587145]
 [ 0.99613948]]
(2, 1)
[[-1.93940438]
 [ 0.9964944 ]]
(2, 1)
[[-1.94293094]
 [ 0.99684869]]
(2, 1)
[[-1.94645113]
 [ 0.99720233]]
(2, 1)
[[-1.94996499]
 [ 0.99755533]]
```

```
(2, 1)
[[-1.95347251]
 [ 0.9979077 ]]
(2, 1)
[[-1.9569737 ]
 [ 0.99825943]]
(2, 1)
[[-1.96046859]
 [ 0.99861053]]
(2, 1)
[[-1.96395718]
 [ 0.998961  ]]
(2, 1)
[[-1.96743947]
 [ 0.99931083]]
(2, 1)
[[-1.97091549]
 [ 0.99966004]]
(2, 1)
[[-1.97438525]
 [ 1.00000861]]
(2, 1)
[[-1.97784875]
 [ 1.00035656]]
(2, 1)
[[-1.981306  ]
 [ 1.00070388]]
(2, 1)
[[-1.98475703]
 [ 1.00105057]]
(2, 1)
[[-1.98820183]
 [ 1.00139664]]
(2, 1)
[[-1.99164043]
 [ 1.00174208]]
(2, 1)
[[-1.99507282]
 [ 1.0020869 ]]
(2, 1)
[[-1.99849903]
 [ 1.0024311 ]]
(2, 1)
[[-2.00191906]
 [ 1.00277468]]
(2, 1)
[[-2.00533293]
 [ 1.00311764]]
(2, 1)
[[-2.00874064]
 [ 1.00345998]]
(2, 1)
[[-2.01214221]
 [ 1.00380171]]
(2, 1)
[[-2.01553765]
 [ 1.00414282]]
```

```
(2, 1)
[[-2.01892697]
 [ 1.00448331]]
(2, 1)
[[-2.02231018]
 [ 1.00482319]]
(2, 1)
[[-2.02568729]
 [ 1.00516246]]
(2, 1)
[[-2.02905831]
 [ 1.00550111]]
(2, 1)
[[-2.03242326]
 [ 1.00583916]]
(2, 1)
[[-2.03578214]
 [ 1.0061766 ]]
(2, 1)
[[-2.03913497]
 [ 1.00651342]]
(2, 1)
[[-2.04248175]
 [ 1.00684964]]
(2, 1)
[[-2.0458225 ]
 [ 1.00718526]]
(2, 1)
[[-2.04915723]
 [ 1.00752027]]
(2, 1)
[[-2.05248594]
 [ 1.00785467]]
(2, 1)
[[-2.05580866]
 [ 1.00818848]]
(2, 1)
[[-2.05912538]
 [ 1.00852168]]
(2, 1)
[[-2.06243613]
 [ 1.00885428]]
(2, 1)
[[-2.06574091]
 [ 1.00918628]]
(2, 1)
[[-2.06903973]
 [ 1.00951768]]
(2, 1)
[[-2.07233261]
 [ 1.00984849]]
(2, 1)
[[-2.07561955]
 [ 1.0101787 ]]
(2, 1)
[[-2.07890056]
 [ 1.01050831]]
```

```
(2, 1)
[[-2.08217567]
 [ 1.01083733]]
(2, 1)
[[-2.08544486]
 [ 1.01116576]]
(2, 1)
[[-2.08870817]
 [ 1.01149359]]
(2, 1)
[[-2.09196559]
 [ 1.01182083]]
(2, 1)
[[-2.09521714]
 [ 1.01214749]]
(2, 1)
[[-2.09846283]
 [ 1.01247355]]
(2, 1)
[[-2.10170267]
 [ 1.01279903]]
(2, 1)
[[-2.10493666]
 [ 1.01312392]]
(2, 1)
[[-2.10816483]
 [ 1.01344822]]
(2, 1)
[[-2.11138718]
 [ 1.01377194]]
(2, 1)
[[-2.11460372]
 [ 1.01409508]]
(2, 1)
[[-2.11781446]
 [ 1.01441763]]
(2, 1)
[[-2.12101942]
 [ 1.01473961]]
(2, 1)
[[-2.1242186]
 [ 1.015061 ]]
(2, 1)
[[-2.12741201]
 [ 1.01538181]]
(2, 1)
[[-2.13059966]
 [ 1.01570204]]
(2, 1)
[[-2.13378157]
 [ 1.0160217 ]]
(2, 1)
[[-2.13695774]
 [ 1.01634078]]
(2, 1)
[[-2.14012819]
 [ 1.01665929]]
```

```
(2, 1)
[[-2.14329292]
 [ 1.01697722]]
(2, 1)
[[-2.14645195]
 [ 1.01729458]]
(2, 1)
[[-2.14960528]
 [ 1.01761137]]
(2, 1)
[[-2.15275293]
 [ 1.01792758]]
(2, 1)
[[-2.15589491]
 [ 1.01824323]]
(2, 1)
[[-2.15903122]
 [ 1.0185583 ]]
(2, 1)
[[-2.16216187]
 [ 1.01887281]]
(2, 1)
[[-2.16528689]
 [ 1.01918675]]
(2, 1)
[[-2.16840627]
 [ 1.01950013]]
(2, 1)
[[-2.17152003]
 [ 1.01981294]]
(2, 1)
[[-2.17462817]
 [ 1.02012519]]
(2, 1)
[[-2.17773072]
 [ 1.02043687]]
(2, 1)
[[-2.18082767]
 [ 1.02074799]]
(2, 1)
[[-2.18391903]
 [ 1.02105855]]
(2, 1)
[[-2.18700483]
 [ 1.02136856]]
(2, 1)
[[-2.19008506]
 [ 1.021678  ]]
(2, 1)
[[-2.19315974]
 [ 1.02198688]]
(2, 1)
[[-2.19622888]
 [ 1.02229521]]
(2, 1)
[[-2.19929249]
 [ 1.02260298]]
```

```
(2, 1)
[[-2.20235057]
 [ 1.0229102 ]]
(2, 1)
[[-2.20540314]
 [ 1.02321686]]
(2, 1)
[[-2.20845021]
 [ 1.02352298]]
(2, 1)
[[-2.21149178]
 [ 1.02382854]]
(2, 1)
[[-2.21452787]
 [ 1.02413354]]
(2, 1)
[[-2.21755849]
 [ 1.024438  ]]
(2, 1)
[[-2.22058365]
 [ 1.02474191]]
(2, 1)
[[-2.22360335]
 [ 1.02504527]]
(2, 1)
[[-2.22661761]
 [ 1.02534809]]
(2, 1)
[[-2.22962644]
 [ 1.02565036]]
(2, 1)
[[-2.23262984]
 [ 1.02595208]]
(2, 1)
[[-2.23562783]
 [ 1.02625326]]
(2, 1)
[[-2.23862042]
 [ 1.0265539 ]]
(2, 1)
[[-2.24160761]
 [ 1.026854  ]]
(2, 1)
[[-2.24458941]
 [ 1.02715355]]
(2, 1)
[[-2.24756584]
 [ 1.02745257]]
(2, 1)
[[-2.25053691]
 [ 1.02775104]]
(2, 1)
[[-2.25350262]
 [ 1.02804898]]
(2, 1)
[[-2.25646298]
 [ 1.02834638]]
```

```
(2, 1)
[[-2.25941801]
 [ 1.02864324]]
(2, 1)
[[-2.26236771]
 [ 1.02893957]]
(2, 1)
[[-2.2653121 ]
 [ 1.02923537]]
(2, 1)
[[-2.26825117]
 [ 1.02953063]]
(2, 1)
[[-2.27118495]
 [ 1.02982536]]
(2, 1)
[[-2.27411344]
 [ 1.03011956]]
(2, 1)
[[-2.27703665]
 [ 1.03041323]]
(2, 1)
[[-2.27995459]
 [ 1.03070637]]
(2, 1)
[[-2.28286727]
 [ 1.03099898]]
(2, 1)
[[-2.28577471]
 [ 1.03129106]]
(2, 1)
[[-2.2886769 ]
 [ 1.03158262]]
(2, 1)
[[-2.29157386]
 [ 1.03187365]]
(2, 1)
[[-2.29446559]
 [ 1.03216415]]
(2, 1)
[[-2.29735212]
 [ 1.03245414]]
(2, 1)
[[-2.30023344]
 [ 1.0327436 ]]
(2, 1)
[[-2.30310956]
 [ 1.03303254]]
(2, 1)
[[-2.30598051]
 [ 1.03332095]]
(2, 1)
[[-2.30884628]
 [ 1.03360885]]
(2, 1)
[[-2.31170688]
 [ 1.03389623]]
```

```
(2, 1)
[[-2.31456232]
 [ 1.03418309]]
(2, 1)
[[-2.31741262]
 [ 1.03446943]]
(2, 1)
[[-2.32025778]
 [ 1.03475526]]
(2, 1)
[[-2.32309781]
 [ 1.03504057]]
(2, 1)
[[-2.32593272]
 [ 1.03532537]]
(2, 1)
[[-2.32876252]
 [ 1.03560965]]
(2, 1)
[[-2.33158723]
 [ 1.03589343]]
(2, 1)
[[-2.33440683]
 [ 1.03617669]]
(2, 1)
[[-2.33722136]
 [ 1.03645944]]
(2, 1)
[[-2.34003081]
 [ 1.03674168]]
(2, 1)
[[-2.3428352 ]
 [ 1.03702341]]
(2, 1)
[[-2.34563454]
 [ 1.03730463]]
(2, 1)
[[-2.34842882]
 [ 1.03758535]]
(2, 1)
[[-2.35121807]
 [ 1.03786556]]
(2, 1)
[[-2.3540023 ]
 [ 1.03814526]]
(2, 1)
[[-2.3567815 ]
 [ 1.03842446]]
(2, 1)
[[-2.35955569]
 [ 1.03870316]]
(2, 1)
[[-2.36232489]
 [ 1.03898136]]
(2, 1)
[[-2.36508909]
 [ 1.03925905]]
```

```
(2, 1)
[[-2.36784831]
 [ 1.03953624]]
(2, 1)
[[-2.37060255]
 [ 1.03981294]]
(2, 1)
[[-2.37335183]
 [ 1.04008913]]
(2, 1)
[[-2.37609616]
 [ 1.04036483]]
(2, 1)
[[-2.37883553]
 [ 1.04064003]]
(2, 1)
[[-2.38156997]
 [ 1.04091473]]
(2, 1)
[[-2.38429948]
 [ 1.04118894]]
(2, 1)
[[-2.38702407]
 [ 1.04146266]]
(2, 1)
[[-2.38974375]
 [ 1.04173588]]
(2, 1)
[[-2.39245853]
 [ 1.04200861]]
(2, 1)
[[-2.39516841]
 [ 1.04228084]]
(2, 1)
[[-2.39787341]
 [ 1.04255259]]
(2, 1)
[[-2.40057353]
 [ 1.04282385]]
(2, 1)
[[-2.40326878]
 [ 1.04309462]]
(2, 1)
[[-2.40595918]
 [ 1.04336489]]
(2, 1)
[[-2.40864473]
 [ 1.04363469]]
(2, 1)
[[-2.41132543]
 [ 1.04390399]]
(2, 1)
[[-2.41400131]
 [ 1.04417281]]
(2, 1)
[[-2.41667235]
 [ 1.04444115]]
```

```
(2, 1)
[[-2.41933859]
 [ 1.044709  ]]
(2, 1)
[[-2.42200002]
 [ 1.04497637]]
(2, 1)
[[-2.42465665]
 [ 1.04524326]]
(2, 1)
[[-2.42730849]
 [ 1.04550966]]
(2, 1)
[[-2.42995555]
 [ 1.04577559]]
(2, 1)
[[-2.43259784]
 [ 1.04604104]]
(2, 1)
[[-2.43523537]
 [ 1.04630601]]
(2, 1)
[[-2.43786815]
 [ 1.0465705 ]]
(2, 1)
[[-2.44049617]
 [ 1.04683451]]
(2, 1)
[[-2.44311946]
 [ 1.04709805]]
(2, 1)
[[-2.44573802]
 [ 1.04736111]]
(2, 1)
[[-2.44835187]
 [ 1.0476237 ]]
(2, 1)
[[-2.450961  ]
 [ 1.04788582]]
(2, 1)
[[-2.45356542]
 [ 1.04814746]]
(2, 1)
[[-2.45616515]
 [ 1.04840863]]
(2, 1)
[[-2.4587602 ]
 [ 1.04866933]]
(2, 1)
[[-2.46135057]
 [ 1.04892956]]
(2, 1)
[[-2.46393626]
 [ 1.04918932]]
(2, 1)
[[-2.4665173 ]
 [ 1.04944861]]
```

```
(2, 1)
[[-2.46909369]
 [ 1.04970744]]
(2, 1)
[[-2.47166543]
 [ 1.0499658 ]]
(2, 1)
[[-2.47423253]
 [ 1.05022369]]
(2, 1)
[[-2.47679501]
 [ 1.05048112]]
(2, 1)
[[-2.47935287]
 [ 1.05073809]]
(2, 1)
[[-2.48190611]
 [ 1.05099459]]
(2, 1)
[[-2.48445476]
 [ 1.05125063]]
(2, 1)
[[-2.48699881]
 [ 1.0515062 ]]
(2, 1)
[[-2.48953827]
 [ 1.05176132]]
(2, 1)
[[-2.49207316]
 [ 1.05201598]]
(2, 1)
[[-2.49460348]
 [ 1.05227018]]
(2, 1)
[[-2.49712923]
 [ 1.05252392]]
(2, 1)
[[-2.49965044]
 [ 1.0527772 ]]
(2, 1)
[[-2.5021671 ]
 [ 1.05303002]]
(2, 1)
[[-2.50467922]
 [ 1.05328239]]
(2, 1)
[[-2.50718681]
 [ 1.05353431]]
(2, 1)
[[-2.50968989]
 [ 1.05378577]]
(2, 1)
[[-2.51218845]
 [ 1.05403678]]
(2, 1)
[[-2.51468251]
 [ 1.05428733]]
```

```
(2, 1)
[[-2.51717207]
 [ 1.05453744]]
(2, 1)
[[-2.51965714]
 [ 1.05478709]]
(2, 1)
[[-2.52213774]
 [ 1.05503629]]
(2, 1)
[[-2.52461386]
 [ 1.05528504]]
(2, 1)
[[-2.52708552]
 [ 1.05553335]]
(2, 1)
[[-2.52955273]
 [ 1.05578121]]
(2, 1)
[[-2.53201548]
 [ 1.05602862]]
(2, 1)
[[-2.5344738 ]
 [ 1.05627558]]
(2, 1)
[[-2.53692769]
 [ 1.0565221 ]]
(2, 1)
[[-2.53937715]
 [ 1.05676818]]
(2, 1)
[[-2.5418222 ]
 [ 1.05701381]]
(2, 1)
[[-2.54426284]
 [ 1.057259  ]]
(2, 1)
[[-2.54669908]
 [ 1.05750374]]
(2, 1)
[[-2.54913093]
 [ 1.05774805]]
(2, 1)
[[-2.55155839]
 [ 1.05799192]]
(2, 1)
[[-2.55398148]
 [ 1.05823534]]
(2, 1)
[[-2.55640021]
 [ 1.05847833]]
(2, 1)
[[-2.55881457]
 [ 1.05872088]]
(2, 1)
[[-2.56122458]
 [ 1.05896299]]
```

```
(2, 1)
[[-2.56363024]
 [ 1.05920466]]
(2, 1)
[[-2.56603157]
 [ 1.0594459 ]]
(2, 1)
[[-2.56842857]
 [ 1.05968671]]
(2, 1)
[[-2.57082125]
 [ 1.05992708]]
(2, 1)
[[-2.57320962]
 [ 1.06016702]]
(2, 1)
[[-2.57559368]
 [ 1.06040652]]
(2, 1)
[[-2.57797344]
 [ 1.06064559]]
(2, 1)
[[-2.58034892]
 [ 1.06088424]]
(2, 1)
[[-2.58272011]
 [ 1.06112245]]
(2, 1)
[[-2.58508703]
 [ 1.06136023]]
(2, 1)
[[-2.58744968]
 [ 1.06159758]]
(2, 1)
[[-2.58980807]
 [ 1.06183451]]
(2, 1)
[[-2.59216221]
 [ 1.06207101]]
(2, 1)
[[-2.59451211]
 [ 1.06230708]]
(2, 1)
[[-2.59685777]
 [ 1.06254273]]
(2, 1)
[[-2.59919921]
 [ 1.06277795]]
(2, 1)
[[-2.60153642]
 [ 1.06301275]]
(2, 1)
[[-2.60386942]
 [ 1.06324712]]
(2, 1)
[[-2.60619821]
 [ 1.06348108]]
```

```
(2, 1)
[[-2.60852281]
 [ 1.06371461]]
(2, 1)
[[-2.61084322]
 [ 1.06394772]]
(2, 1)
[[-2.61315944]
 [ 1.06418041]]
(2, 1)
[[-2.61547149]
 [ 1.06441268]]
(2, 1)
[[-2.61777937]
 [ 1.06464453]]
(2, 1)
[[-2.62008309]
 [ 1.06487596]]
(2, 1)
[[-2.62238265]
 [ 1.06510698]]
(2, 1)
[[-2.62467808]
 [ 1.06533758]]
(2, 1)
[[-2.62696936]
 [ 1.06556776]]
(2, 1)
[[-2.62925652]
 [ 1.06579753]]
(2, 1)
[[-2.63153955]
 [ 1.06602689]]
(2, 1)
[[-2.63381846]
 [ 1.06625583]]
(2, 1)
[[-2.63609327]
 [ 1.06648436]]
(2, 1)
[[-2.63836398]
 [ 1.06671248]]
(2, 1)
[[-2.64063059]
 [ 1.06694018]]
(2, 1)
[[-2.64289312]
 [ 1.06716748]]
(2, 1)
[[-2.64515157]
 [ 1.06739437]]
(2, 1)
[[-2.64740595]
 [ 1.06762084]]
(2, 1)
[[-2.64965627]
 [ 1.06784691]]
```

```
(2, 1)
[[-2.65190253]
 [ 1.06807257]]
(2, 1)
[[-2.65414474]
 [ 1.06829783]]
(2, 1)
[[-2.6563829 ]
 [ 1.06852267]]
(2, 1)
[[-2.65861704]
 [ 1.06874712]]
(2, 1)
[[-2.66084714]
 [ 1.06897116]]
(2, 1)
[[-2.66307323]
 [ 1.06919479]]
(2, 1)
[[-2.6652953 ]
 [ 1.06941802]]
(2, 1)
[[-2.66751337]
 [ 1.06964085]]
(2, 1)
[[-2.66972744]
 [ 1.06986328]]
(2, 1)
[[-2.67193752]
 [ 1.0700853 ]]
(2, 1)
[[-2.67414362]
 [ 1.07030693]]
(2, 1)
[[-2.67634573]
 [ 1.07052816]]
(2, 1)
[[-2.67854388]
 [ 1.07074898]]
(2, 1)
[[-2.68073807]
 [ 1.07096941]]
(2, 1)
[[-2.6829283 ]
 [ 1.07118945]]
(2, 1)
[[-2.68511458]
 [ 1.07140908]]
(2, 1)
[[-2.68729693]
 [ 1.07162832]]
(2, 1)
[[-2.68947533]
 [ 1.07184717]]
(2, 1)
[[-2.69164981]
 [ 1.07206562]]
```

```
(2, 1)
[[-2.69382038]
 [ 1.07228367]]
(2, 1)
[[-2.69598703]
 [ 1.07250134]]
(2, 1)
[[-2.69814977]
 [ 1.07271861]]
(2, 1)
[[-2.70030861]
 [ 1.07293549]]
(2, 1)
[[-2.70246357]
 [ 1.07315198]]
(2, 1)
[[-2.70461464]
 [ 1.07336807]]
(2, 1)
[[-2.70676183]
 [ 1.07358378]]
(2, 1)
[[-2.70890515]
 [ 1.0737991 ]]
(2, 1)
[[-2.7110446 ]
 [ 1.07401403]]
(2, 1)
[[-2.71318021]
 [ 1.07422858]]
(2, 1)
[[-2.71531196]
 [ 1.07444274]]
(2, 1)
[[-2.71743987]
 [ 1.07465651]]
(2, 1)
[[-2.71956394]
 [ 1.07486989]]
(2, 1)
[[-2.72168418]
 [ 1.0750829 ]]
(2, 1)
[[-2.7238006 ]
 [ 1.07529551]]
(2, 1)
[[-2.72591321]
 [ 1.07550775]]
(2, 1)
[[-2.72802201]
 [ 1.0757196 ]]
(2, 1)
[[-2.73012701]
 [ 1.07593107]]
(2, 1)
[[-2.73222821]
 [ 1.07614216]]
```

```
(2, 1)
[[-2.73432562]
 [ 1.07635287]]
(2, 1)
[[-2.73641926]
 [ 1.07656319]]
(2, 1)
[[-2.73850912]
 [ 1.07677314]]
(2, 1)
[[-2.74059521]
 [ 1.07698271]]
(2, 1)
[[-2.74267754]
 [ 1.07719191]]
(2, 1)
[[-2.74475612]
 [ 1.07740072]]
(2, 1)
[[-2.74683096]
 [ 1.07760916]]
(2, 1)
[[-2.74890205]
 [ 1.07781723]]
(2, 1)
[[-2.75096941]
 [ 1.07802491]]
(2, 1)
[[-2.75303304]
 [ 1.07823223]]
(2, 1)
[[-2.75509295]
 [ 1.07843917]]
(2, 1)
[[-2.75714915]
 [ 1.07864574]]
(2, 1)
[[-2.75920164]
 [ 1.07885193]]
(2, 1)
[[-2.76125044]
 [ 1.07905776]]
(2, 1)
[[-2.76329554]
 [ 1.07926321]]
(2, 1)
[[-2.76533695]
 [ 1.07946829]]
(2, 1)
[[-2.76737468]
 [ 1.079673  ]]
(2, 1)
[[-2.76940874]
 [ 1.07987735]]
(2, 1)
[[-2.77143914]
 [ 1.08008132]]
```

```
(2, 1)
[[-2.77346587]
 [ 1.08028493]]
(2, 1)
[[-2.77548895]
 [ 1.08048817]]
(2, 1)
[[-2.77750839]
 [ 1.08069104]]
(2, 1)
[[-2.77952418]
 [ 1.08089355]]
(2, 1)
[[-2.78153634]
 [ 1.08109569]]
(2, 1)
[[-2.78354487]
 [ 1.08129747]]
(2, 1)
[[-2.78554978]
 [ 1.08149889]]
(2, 1)
[[-2.78755108]
 [ 1.08169994]]
(2, 1)
[[-2.78954877]
 [ 1.08190063]]
(2, 1)
[[-2.79154286]
 [ 1.08210096]]
(2, 1)
[[-2.79353336]
 [ 1.08230092]]
(2, 1)
[[-2.79552026]
 [ 1.08250053]]
(2, 1)
[[-2.79750359]
 [ 1.08269978]]
(2, 1)
[[-2.79948334]
 [ 1.08289866]]
(2, 1)
[[-2.80145952]
 [ 1.08309719]]
(2, 1)
[[-2.80343214]
 [ 1.08329536]]
(2, 1)
[[-2.8054012 ]
 [ 1.08349318]]
(2, 1)
[[-2.80736672]
 [ 1.08369064]]
(2, 1)
[[-2.80932869]
 [ 1.08388774]]
```

```
(2, 1)
[[-2.81128712]
 [ 1.08408448]]
(2, 1)
[[-2.81324202]
 [ 1.08428087]]
(2, 1)
[[-2.8151934 ]
 [ 1.08447691]]
(2, 1)
[[-2.81714127]
 [ 1.0846726 ]]
(2, 1)
[[-2.81908562]
 [ 1.08486793]]
(2, 1)
[[-2.82102646]
 [ 1.08506291]]
(2, 1)
[[-2.82296381]
 [ 1.08525753]]
(2, 1)
[[-2.82489767]
 [ 1.08545181]]
(2, 1)
[[-2.82682804]
 [ 1.08564574]]
(2, 1)
[[-2.82875493]
 [ 1.08583931]]
(2, 1)
[[-2.83067834]
 [ 1.08603254]]
(2, 1)
[[-2.83259829]
 [ 1.08622542]]
(2, 1)
[[-2.83451478]
 [ 1.08641795]]
(2, 1)
[[-2.83642782]
 [ 1.08661014]]
(2, 1)
[[-2.8383374 ]
 [ 1.08680198]]
(2, 1)
[[-2.84024354]
 [ 1.08699347]]
(2, 1)
[[-2.84214625]
 [ 1.08718462]]
(2, 1)
[[-2.84404553]
 [ 1.08737542]]
(2, 1)
[[-2.84594138]
 [ 1.08756588]]
```

```
(2, 1)
[[-2.84783382]
 [ 1.087756  ]]
(2, 1)
[[-2.84972284]
 [ 1.08794577]]
(2, 1)
[[-2.85160846]
 [ 1.0881352 ]]
(2, 1)
[[-2.85349068]
 [ 1.08832429]]
(2, 1)
[[-2.85536951]
 [ 1.08851304]]
(2, 1)
[[-2.85724495]
 [ 1.08870145]]
(2, 1)
[[-2.85911701]
 [ 1.08888951]]
(2, 1)
[[-2.8609857 ]
 [ 1.08907724]]
(2, 1)
[[-2.86285102]
 [ 1.08926464]]
(2, 1)
[[-2.86471297]
 [ 1.08945169]]
(2, 1)
[[-2.86657157]
 [ 1.08963841]]
(2, 1)
[[-2.86842682]
 [ 1.08982479]]
(2, 1)
[[-2.87027872]
 [ 1.09001083]]
(2, 1)
[[-2.87212729]
 [ 1.09019654]]
(2, 1)
[[-2.87397252]
 [ 1.09038191]]
(2, 1)
[[-2.87581443]
 [ 1.09056695]]
(2, 1)
[[-2.87765302]
 [ 1.09075166]]
(2, 1)
[[-2.87948829]
 [ 1.09093603]]
(2, 1)
[[-2.88132026]
 [ 1.09112007]]
```

```
(2, 1)
[[-2.88314892]
 [ 1.09130378]]
(2, 1)
[[-2.88497428]
 [ 1.09148716]]
(2, 1)
[[-2.88679636]
 [ 1.09167021]]
(2, 1)
[[-2.88861515]
 [ 1.09185292]]
(2, 1)
[[-2.89043066]
 [ 1.09203531]]
(2, 1)
[[-2.8922429 ]
 [ 1.09221737]]
(2, 1)
[[-2.89405188]
 [ 1.0923991 ]]
(2, 1)
[[-2.89585759]
 [ 1.0925805 ]]
(2, 1)
[[-2.89766005]
 [ 1.09276158]]
(2, 1)
[[-2.89945926]
 [ 1.09294233]]
(2, 1)
[[-2.90125522]
 [ 1.09312275]]
(2, 1)
[[-2.90304795]
 [ 1.09330285]]
(2, 1)
[[-2.90483745]
 [ 1.09348263]]
(2, 1)
[[-2.90662372]
 [ 1.09366208]]
(2, 1)
[[-2.90840677]
 [ 1.0938412 ]]
(2, 1)
[[-2.9101866 ]
 [ 1.09402001]]
(2, 1)
[[-2.91196323]
 [ 1.09419849]]
(2, 1)
[[-2.91373666]
 [ 1.09437665]]
(2, 1)
[[-2.91550689]
 [ 1.09455449]]
```

```
(2, 1)
[[-2.91727392]
 [ 1.09473201]]
(2, 1)
[[-2.91903778]
 [ 1.0949092 ]]
(2, 1)
[[-2.92079845]
 [ 1.09508608]]
(2, 1)
[[-2.92255595]
 [ 1.09526264]]
(2, 1)
[[-2.92431028]
 [ 1.09543889]]
(2, 1)
[[-2.92606145]
 [ 1.09561481]]
(2, 1)
[[-2.92780946]
 [ 1.09579042]]
(2, 1)
[[-2.92955432]
 [ 1.09596571]]
(2, 1)
[[-2.93129604]
 [ 1.09614068]]
(2, 1)
[[-2.93303461]
 [ 1.09631534]]
(2, 1)
[[-2.93477006]
 [ 1.09648968]]
(2, 1)
[[-2.93650237]
 [ 1.09666371]]
(2, 1)
[[-2.93823156]
 [ 1.09683743]]
(2, 1)
[[-2.93995764]
 [ 1.09701083]]
(2, 1)
[[-2.9416806 ]
 [ 1.09718392]]
(2, 1)
[[-2.94340046]
 [ 1.0973567 ]]
(2, 1)
[[-2.94511721]
 [ 1.09752917]]
(2, 1)
[[-2.94683088]
 [ 1.09770132]]
(2, 1)
[[-2.94854145]
 [ 1.09787317]]
```

```
(2, 1)
[[-2.95024894]
 [ 1.0980447 ]]
(2, 1)
[[-2.95195335]
 [ 1.09821593]]
(2, 1)
[[-2.95365469]
 [ 1.09838685]]
(2, 1)
[[-2.95535296]
 [ 1.09855746]]
(2, 1)
[[-2.95704817]
 [ 1.09872776]]
(2, 1)
[[-2.95874033]
 [ 1.09889776]]
(2, 1)
[[-2.96042944]
 [ 1.09906745]]
(2, 1)
[[-2.9621155 ]
 [ 1.09923683]]
(2, 1)
[[-2.96379852]
 [ 1.09940591]]
(2, 1)
[[-2.9654785 ]
 [ 1.09957468]]
(2, 1)
[[-2.96715546]
 [ 1.09974315]]
(2, 1)
[[-2.9688294 ]
 [ 1.09991131]]
(2, 1)
[[-2.97050032]
 [ 1.10007918]]
(2, 1)
[[-2.97216823]
 [ 1.10024674]]
(2, 1)
[[-2.97383313]
 [ 1.10041399]]
(2, 1)
[[-2.97549503]
 [ 1.10058095]]
(2, 1)
[[-2.97715393]
 [ 1.1007476 ]]
(2, 1)
[[-2.97880984]
 [ 1.10091396]]
(2, 1)
[[-2.98046277]
 [ 1.10108001]]
```

```
(2, 1)
[[-2.98211272]
 [ 1.10124577]]
(2, 1)
[[-2.98375969]
 [ 1.10141122]]
(2, 1)
[[-2.9854037 ]
 [ 1.10157638]]
(2, 1)
[[-2.98704474]
 [ 1.10174124]]
(2, 1)
[[-2.98868282]
 [ 1.10190581]]
(2, 1)
[[-2.99031796]
 [ 1.10207007]]
(2, 1)
[[-2.99195014]
 [ 1.10223404]]
(2, 1)
[[-2.99357938]
 [ 1.10239772]]
(2, 1)
[[-2.99520569]
 [ 1.1025611 ]]
(2, 1)
[[-2.99682906]
 [ 1.10272418]]
(2, 1)
[[-2.99844951]
 [ 1.10288697]]
(2, 1)
[[-3.00006703]
 [ 1.10304947]]
(2, 1)
[[-3.00168164]
 [ 1.10321168]]
(2, 1)
[[-3.00329334]
 [ 1.10337359]]
(2, 1)
[[-3.00490214]
 [ 1.10353521]]
(2, 1)
[[-3.00650803]
 [ 1.10369654]]
(2, 1)
[[-3.00811103]
 [ 1.10385758]]
(2, 1)
[[-3.00971114]
 [ 1.10401833]]
(2, 1)
[[-3.01130836]
 [ 1.10417879]]
```

```
(2, 1)
[[-3.01290271]
 [ 1.10433896]]
(2, 1)
[[-3.01449418]
 [ 1.10449884]]
(2, 1)
[[-3.01608279]
 [ 1.10465843]]
(2, 1)
[[-3.01766853]
 [ 1.10481773]]
(2, 1)
[[-3.01925141]
 [ 1.10497675]]
(2, 1)
[[-3.02083144]
 [ 1.10513548]]
(2, 1)
[[-3.02240862]
 [ 1.10529393]]
(2, 1)
[[-3.02398295]
 [ 1.10545209]]
(2, 1)
[[-3.02555445]
 [ 1.10560996]]
(2, 1)
[[-3.02712312]
 [ 1.10576755]]
(2, 1)
[[-3.02868896]
 [ 1.10592486]]
(2, 1)
[[-3.03025198]
 [ 1.10608188]]
(2, 1)
[[-3.03181217]
 [ 1.10623862]]
(2, 1)
[[-3.03336956]
 [ 1.10639507]]
(2, 1)
[[-3.03492414]
 [ 1.10655125]]
(2, 1)
[[-3.03647592]
 [ 1.10670714]]
(2, 1)
[[-3.0380249 ]
 [ 1.10686275]]
(2, 1)
[[-3.03957108]
 [ 1.10701808]]
(2, 1)
[[-3.04111448]
 [ 1.10717313]]
```

```
(2, 1)
[[-3.0426551 ]
 [ 1.10732791]]
(2, 1)
[[-3.04419294]
 [ 1.1074824 ]]
(2, 1)
[[-3.04572801]
 [ 1.10763661]]
(2, 1)
[[-3.04726031]
 [ 1.10779055]]
(2, 1)
[[-3.04878985]
 [ 1.10794421]]
(2, 1)
[[-3.05031664]
 [ 1.10809759]]
(2, 1)
[[-3.05184067]
 [ 1.1082507 ]]
(2, 1)
[[-3.05336195]
 [ 1.10840353]]
(2, 1)
[[-3.05488049]
 [ 1.10855608]]
(2, 1)
[[-3.05639629]
 [ 1.10870836]]
(2, 1)
[[-3.05790936]
 [ 1.10886036]]
(2, 1)
[[-3.05941971]
 [ 1.10901209]]
(2, 1)
[[-3.06092733]
 [ 1.10916355]]
(2, 1)
[[-3.06243223]
 [ 1.10931473]]
(2, 1)
[[-3.06393442]
 [ 1.10946564]]
(2, 1)
[[-3.06543391]
 [ 1.10961628]]
(2, 1)
[[-3.06693069]
 [ 1.10976665]]
(2, 1)
[[-3.06842477]
 [ 1.10991675]]
(2, 1)
[[-3.06991616]
 [ 1.11006657]]
```

```
(2, 1)
[[-3.07140486]
 [ 1.11021613]]
(2, 1)
[[-3.07289087]
 [ 1.11036542]]
(2, 1)
[[-3.07437421]
 [ 1.11051443]]
(2, 1)
[[-3.07585488]
 [ 1.11066318]]
(2, 1)
[[-3.07733287]
 [ 1.11081166]]
(2, 1)
[[-3.0788082 ]
 [ 1.11095988]]
(2, 1)
[[-3.08028087]
 [ 1.11110782]]
(2, 1)
[[-3.08175089]
 [ 1.1112555 ]]
(2, 1)
[[-3.08321826]
 [ 1.11140292]]
(2, 1)
[[-3.08468298]
 [ 1.11155006]]
(2, 1)
[[-3.08614506]
 [ 1.11169694]]
(2, 1)
[[-3.08760451]
 [ 1.11184356]]
(2, 1)
[[-3.08906132]
 [ 1.11198991]]
(2, 1)
[[-3.09051551]
 [ 1.112136  ]]
(2, 1)
[[-3.09196708]
 [ 1.11228183]]
(2, 1)
[[-3.09341603]
 [ 1.11242739]]
(2, 1)
[[-3.09486237]
 [ 1.11257269]]
(2, 1)
[[-3.0963061 ]
 [ 1.11271773]]
(2, 1)
[[-3.09774723]
 [ 1.11286251]]
```

```
(2, 1)
[[-3.09918577]
 [ 1.11300702]]
(2, 1)
[[-3.1006217 ]
 [ 1.11315128]]
(2, 1)
[[-3.10205506]
 [ 1.11329528]]
(2, 1)
[[-3.10348582]
 [ 1.11343901]]
(2, 1)
[[-3.10491401]
 [ 1.11358249]]
(2, 1)
[[-3.10633962]
 [ 1.11372571]]
(2, 1)
[[-3.10776267]
 [ 1.11386867]]
(2, 1)
[[-3.10918315]
 [ 1.11401137]]
(2, 1)
[[-3.11060106]
 [ 1.11415382]]
(2, 1)
[[-3.11201642]
 [ 1.114296  ]]
(2, 1)
[[-3.11342924]
 [ 1.11443794]]
(2, 1)
[[-3.1148395 ]
 [ 1.11457961]]
(2, 1)
[[-3.11624722]
 [ 1.11472103]]
(2, 1)
[[-3.11765241]
 [ 1.1148622 ]]
(2, 1)
[[-3.11905506]
 [ 1.11500311]]
(2, 1)
[[-3.12045518]
 [ 1.11514377]]
(2, 1)
[[-3.12185278]
 [ 1.11528417]]
(2, 1)
[[-3.12324786]
 [ 1.11542432]]
(2, 1)
[[-3.12464042]
 [ 1.11556422]]
```

```
(2, 1)
[[-3.12603048]
 [ 1.11570387]]
(2, 1)
[[-3.12741803]
 [ 1.11584326]]
(2, 1)
[[-3.12880307]
 [ 1.1159824 ]]
(2, 1)
[[-3.13018562]
 [ 1.1161213 ]]
(2, 1)
[[-3.13156568]
 [ 1.11625994]]
(2, 1)
[[-3.13294325]
 [ 1.11639833]]
(2, 1)
[[-3.13431834]
 [ 1.11653647]]
(2, 1)
[[-3.13569095]
 [ 1.11667437]]
(2, 1)
[[-3.13706109]
 [ 1.11681201]]
(2, 1)
[[-3.13842875]
 [ 1.11694941]]
(2, 1)
[[-3.13979395]
 [ 1.11708656]]
(2, 1)
[[-3.14115669]
 [ 1.11722346]]
(2, 1)
[[-3.14251697]
 [ 1.11736011]]
(2, 1)
[[-3.1438748 ]
 [ 1.11749652]]
(2, 1)
[[-3.14523019]
 [ 1.11763269]]
(2, 1)
[[-3.14658312]
 [ 1.1177686 ]]
(2, 1)
[[-3.14793363]
 [ 1.11790428]]
(2, 1)
[[-3.14928169]
 [ 1.1180397 ]]
(2, 1)
[[-3.15062733]
 [ 1.11817489]]
```

```
(2, 1)
[[-3.15197054]
 [ 1.11830983]]
(2, 1)
[[-3.15331133]
 [ 1.11844452]]
(2, 1)
[[-3.1546497 ]
 [ 1.11857898]]
(2, 1)
[[-3.15598566]
 [ 1.11871319]]
(2, 1)
[[-3.15731921]
 [ 1.11884716]]
(2, 1)
[[-3.15865036]
 [ 1.11898089]]
(2, 1)
[[-3.15997911]
 [ 1.11911438]]
(2, 1)
[[-3.16130546]
 [ 1.11924762]]
(2, 1)
[[-3.16262942]
 [ 1.11938063]]
(2, 1)
[[-3.163951  ]
 [ 1.11951339]]
(2, 1)
[[-3.16527019]
 [ 1.11964592]]
(2, 1)
[[-3.16658701]
 [ 1.11977821]]
(2, 1)
[[-3.16790145]
 [ 1.11991026]]
(2, 1)
[[-3.16921352]
 [ 1.12004207]]
(2, 1)
[[-3.17052323]
 [ 1.12017365]]
(2, 1)
[[-3.17183058]
 [ 1.12030498]]
(2, 1)
[[-3.17313557]
 [ 1.12043608]]
(2, 1)
[[-3.1744382 ]
 [ 1.12056695]]
(2, 1)
[[-3.17573849]
 [ 1.12069758]]
```

```
(2, 1)
[[-3.17703644]
 [ 1.12082797]]
(2, 1)
[[-3.17833204]
 [ 1.12095813]]
(2, 1)
[[-3.17962532]
 [ 1.12108805]]
(2, 1)
[[-3.18091625]
 [ 1.12121774]]
(2, 1)
[[-3.18220487]
 [ 1.12134719]]
(2, 1)
[[-3.18349116]
 [ 1.12147642]]
(2, 1)
[[-3.18477513]
 [ 1.1216054 ]]
(2, 1)
[[-3.18605678]
 [ 1.12173416]]
(2, 1)
[[-3.18733613]
 [ 1.12186268]]
(2, 1)
[[-3.18861317]
 [ 1.12199098]]
(2, 1)
[[-3.1898879 ]
 [ 1.12211904]]
(2, 1)
[[-3.19116034]
 [ 1.12224687]]
(2, 1)
[[-3.19243049]
 [ 1.12237447]]
(2, 1)
[[-3.19369835]
 [ 1.12250184]]
(2, 1)
[[-3.19496392]
 [ 1.12262898]]
(2, 1)
[[-3.19622721]
 [ 1.12275589]]
(2, 1)
[[-3.19748822]
 [ 1.12288257]]
(2, 1)
[[-3.19874696]
 [ 1.12300903]]
(2, 1)
[[-3.20000343]
 [ 1.12313525]]
```

```
(2, 1)
[[-3.20125764]
 [ 1.12326125]]
(2, 1)
[[-3.20250958]
 [ 1.12338702]]
(2, 1)
[[-3.20375927]
 [ 1.12351257]]
(2, 1)
[[-3.2050067 ]
 [ 1.12363789]]
(2, 1)
[[-3.20625189]
 [ 1.12376298]]
(2, 1)
[[-3.20749483]
 [ 1.12388785]]
(2, 1)
[[-3.20873553]
 [ 1.12401249]]
(2, 1)
[[-3.209974 ]
 [ 1.1241369]]
(2, 1)
[[-3.21121023]
 [ 1.1242611 ]]
(2, 1)
[[-3.21244424]
 [ 1.12438507]]
(2, 1)
[[-3.21367602]
 [ 1.12450881]]
(2, 1)
[[-3.21490557]
 [ 1.12463233]]
(2, 1)
[[-3.21613292]
 [ 1.12475563]]
(2, 1)
[[-3.21735805]
 [ 1.12487871]]
(2, 1)
[[-3.21858097]
 [ 1.12500157]]
(2, 1)
[[-3.21980169]
 [ 1.1251242 ]]
(2, 1)
[[-3.22102021]
 [ 1.12524662]]
(2, 1)
[[-3.22223653]
 [ 1.12536881]]
(2, 1)
[[-3.22345065]
 [ 1.12549078]]
```

```
(2, 1)
[[-3.22466259]
 [ 1.12561253]]
(2, 1)
[[-3.22587235]
 [ 1.12573407]]
(2, 1)
[[-3.22707992]
 [ 1.12585538]]
(2, 1)
[[-3.22828532]
 [ 1.12597648]]
(2, 1)
[[-3.22948855]
 [ 1.12609735]]
(2, 1)
[[-3.2306896 ]
 [ 1.12621801]]
(2, 1)
[[-3.23188849]
 [ 1.12633845]]
(2, 1)
[[-3.23308522]
 [ 1.12645868]]
(2, 1)
[[-3.23427979]
 [ 1.12657869]]
(2, 1)
[[-3.23547221]
 [ 1.12669848]]
(2, 1)
[[-3.23666248]
 [ 1.12681805]]
(2, 1)
[[-3.23785061]
 [ 1.12693741]]
(2, 1)
[[-3.23903659]
 [ 1.12705656]]
(2, 1)
[[-3.24022043]
 [ 1.12717549]]
(2, 1)
[[-3.24140214]
 [ 1.1272942 ]]
(2, 1)
[[-3.24258172]
 [ 1.1274127 ]]
(2, 1)
[[-3.24375918]
 [ 1.12753099]]
(2, 1)
[[-3.24493451]
 [ 1.12764907]]
(2, 1)
[[-3.24610772]
 [ 1.12776693]]
```

```
(2, 1)
[[-3.24727882]
 [ 1.12788458]]
(2, 1)
[[-3.24844781]
 [ 1.12800202]]
(2, 1)
[[-3.24961469]
 [ 1.12811924]]
(2, 1)
[[-3.25077946]
 [ 1.12823626]]
(2, 1)
[[-3.25194214]
 [ 1.12835306]]
(2, 1)
[[-3.25310272]
 [ 1.12846965]]
(2, 1)
[[-3.25426121]
 [ 1.12858604]]
(2, 1)
[[-3.25541761]
 [ 1.12870221]]
(2, 1)
[[-3.25657193]
 [ 1.12881817]]
(2, 1)
[[-3.25772416]
 [ 1.12893393]]
(2, 1)
[[-3.25887432]
 [ 1.12904947]]
(2, 1)
[[-3.26002241]
 [ 1.12916481]]
(2, 1)
[[-3.26116842]
 [ 1.12927994]]
(2, 1)
[[-3.26231237]
 [ 1.12939486]]
(2, 1)
[[-3.26345426]
 [ 1.12950958]]
(2, 1)
[[-3.26459409]
 [ 1.12962408]]
(2, 1)
[[-3.26573186]
 [ 1.12973839]]
(2, 1)
[[-3.26686759]
 [ 1.12985248]]
(2, 1)
[[-3.26800126]
 [ 1.12996637]]
```

```
(2, 1)
[[-3.2691329 ]
 [ 1.13008006]]
(2, 1)
[[-3.27026249]
 [ 1.13019354]]
(2, 1)
[[-3.27139005]
 [ 1.13030681]]
(2, 1)
[[-3.27251557]
 [ 1.13041988]]
(2, 1)
[[-3.27363907]
 [ 1.13053275]]
(2, 1)
[[-3.27476054]
 [ 1.13064541]]
(2, 1)
[[-3.27587999]
 [ 1.13075788]]
(2, 1)
[[-3.27699742]
 [ 1.13087013]]
(2, 1)
[[-3.27811283]
 [ 1.13098219]]
(2, 1)
[[-3.27922624]
 [ 1.13109404]]
(2, 1)
[[-3.28033764]
 [ 1.1312057 ]]
(2, 1)
[[-3.28144703]
 [ 1.13131715]]
(2, 1)
[[-3.28255443]
 [ 1.1314284 ]]
(2, 1)
[[-3.28365983]
 [ 1.13153945]]
(2, 1)
[[-3.28476324]
 [ 1.13165029]]
(2, 1)
[[-3.28586465]
 [ 1.13176094]]
(2, 1)
[[-3.28696409]
 [ 1.13187139]]
(2, 1)
[[-3.28806154]
 [ 1.13198164]]
(2, 1)
[[-3.28915701]
 [ 1.1320917 ]]
```

```
(2, 1)
[[-3.29025051]
 [ 1.13220155]]
(2, 1)
[[-3.29134203]
 [ 1.13231121]]
(2, 1)
[[-3.29243159]
 [ 1.13242066]]
(2, 1)
[[-3.29351919]
 [ 1.13252992]]
(2, 1)
[[-3.29460482]
 [ 1.13263899]]
(2, 1)
[[-3.2956885 ]
 [ 1.13274786]]
(2, 1)
[[-3.29677022]
 [ 1.13285653]]
(2, 1)
[[-3.29785 ]
 [ 1.132965]]
(2, 1)
[[-3.29892782]
 [ 1.13307328]]
(2, 1)
[[-3.30000371]
 [ 1.13318137]]
(2, 1)
[[-3.30107765]
 [ 1.13328925]]
(2, 1)
[[-3.30214966]
 [ 1.13339695]]
(2, 1)
[[-3.30321974]
 [ 1.13350445]]
(2, 1)
[[-3.30428789]
 [ 1.13361176]]
(2, 1)
[[-3.30535411]
 [ 1.13371887]]
(2, 1)
[[-3.30641841]
 [ 1.13382579]]
(2, 1)
[[-3.3074808 ]
 [ 1.13393252]]
(2, 1)
[[-3.30854126]
 [ 1.13403906]]
(2, 1)
[[-3.30959982]
 [ 1.1341454 ]]
```

```
(2, 1)
[[-3.31065647]
 [ 1.13425155]]
(2, 1)
[[-3.31171121]
 [ 1.13435751]]
(2, 1)
[[-3.31276405]
 [ 1.13446328]]
(2, 1)
[[-3.31381499]
 [ 1.13456886]]
(2, 1)
[[-3.31486404]
 [ 1.13467425]]
(2, 1)
[[-3.3159112 ]
 [ 1.13477945]]
(2, 1)
[[-3.31695647]
 [ 1.13488445]]
(2, 1)
[[-3.31799986]
 [ 1.13498927]]
(2, 1)
[[-3.31904136]
 [ 1.1350939 ]]
(2, 1)
[[-3.32008099]
 [ 1.13519835]]
(2, 1)
[[-3.32111875]
 [ 1.1353026 ]]
(2, 1)
[[-3.32215463]
 [ 1.13540667]]
(2, 1)
[[-3.32318865]
 [ 1.13551054]]
(2, 1)
[[-3.3242208 ]
 [ 1.13561423]]
(2, 1)
[[-3.32525109]
 [ 1.13571774]]
(2, 1)
[[-3.32627953]
 [ 1.13582106]]
(2, 1)
[[-3.32730611]
 [ 1.13592419]]
(2, 1)
[[-3.32833084]
 [ 1.13602713]]
(2, 1)
[[-3.32935372]
 [ 1.13612989]]
```

```
(2, 1)
[[-3.33037476]
 [ 1.13623247]]
(2, 1)
[[-3.33139396]
 [ 1.13633486]]
(2, 1)
[[-3.33241132]
 [ 1.13643706]]
(2, 1)
[[-3.33342685]
 [ 1.13653908]]
(2, 1)
[[-3.33444054]
 [ 1.13664092]]
(2, 1)
[[-3.33545241]
 [ 1.13674257]]
(2, 1)
[[-3.33646246]
 [ 1.13684404]]
(2, 1)
[[-3.33747068]
 [ 1.13694533]]
(2, 1)
[[-3.33847709]
 [ 1.13704643]]
(2, 1)
[[-3.33948168]
 [ 1.13714736]]
(2, 1)
[[-3.34048447]
 [ 1.1372481 ]]
(2, 1)
[[-3.34148544]
 [ 1.13734866]]
(2, 1)
[[-3.34248461]
 [ 1.13744903]]
(2, 1)
[[-3.34348198]
 [ 1.13754923]]
(2, 1)
[[-3.34447755]
 [ 1.13764925]]
(2, 1)
[[-3.34547133]
 [ 1.13774908]]
(2, 1)
[[-3.34646332]
 [ 1.13784874]]
(2, 1)
[[-3.34745351]
 [ 1.13794821]]
(2, 1)
[[-3.34844193]
 [ 1.13804751]]
```

```
(2, 1)
[[-3.34942856]
 [ 1.13814663]]
(2, 1)
[[-3.35041341]
 [ 1.13824557]]
(2, 1)
[[-3.35139649]
 [ 1.13834433]]
(2, 1)
[[-3.35237779]
 [ 1.13844291]]
(2, 1)
[[-3.35335733]
 [ 1.13854132]]
(2, 1)
[[-3.3543351 ]
 [ 1.13863954]]
(2, 1)
[[-3.35531111]
 [ 1.13873759]]
(2, 1)
[[-3.35628536]
 [ 1.13883547]]
(2, 1)
[[-3.35725785]
 [ 1.13893316]]
(2, 1)
[[-3.35822859]
 [ 1.13903069]]
(2, 1)
[[-3.35919758]
 [ 1.13912803]]
(2, 1)
[[-3.36016482]
 [ 1.1392252 ]]
(2, 1)
[[-3.36113032]
 [ 1.1393222 ]]
(2, 1)
[[-3.36209408]
 [ 1.13941902]]
(2, 1)
[[-3.3630561 ]
 [ 1.13951566]]
(2, 1)
[[-3.36401639]
 [ 1.13961213]]
(2, 1)
[[-3.36497494]
 [ 1.13970843]]
(2, 1)
[[-3.36593177]
 [ 1.13980455]]
(2, 1)
[[-3.36688688]
 [ 1.1399005 ]]
```

```
(2, 1)
[[-3.36784026]
 [ 1.13999628]]
(2, 1)
[[-3.36879192]
 [ 1.14009189]]
(2, 1)
[[-3.36974187]
 [ 1.14018732]]
(2, 1)
[[-3.37069011]
 [ 1.14028258]]
(2, 1)
[[-3.37163663]
 [ 1.14037767]]
(2, 1)
[[-3.37258145]
 [ 1.14047259]]
(2, 1)
[[-3.37352457]
 [ 1.14056733]]
(2, 1)
[[-3.37446599]
 [ 1.14066191]]
(2, 1)
[[-3.37540571]
 [ 1.14075631]]
(2, 1)
[[-3.37634373]
 [ 1.14085055]]
(2, 1)
[[-3.37728007]
 [ 1.14094461]]
(2, 1)
[[-3.37821471]
 [ 1.14103851]]
(2, 1)
[[-3.37914768]
 [ 1.14113224]]
(2, 1)
[[-3.38007896]
 [ 1.14122579]]
(2, 1)
[[-3.38100856]
 [ 1.14131918]]
(2, 1)
[[-3.38193649]
 [ 1.1414124 ]]
(2, 1)
[[-3.38286274]
 [ 1.14150545]]
(2, 1)
[[-3.38378732]
 [ 1.14159834]]
(2, 1)
[[-3.38471024]
 [ 1.14169106]]
```

```
(2, 1)
[[-3.3856315 ]
 [ 1.14178361]]
(2, 1)
[[-3.38655109]
 [ 1.14187599]]
(2, 1)
[[-3.38746902]
 [ 1.1419682 ]]
(2, 1)
[[-3.38838531]
 [ 1.14206026]]
(2, 1)
[[-3.38929993]
 [ 1.14215214]]
(2, 1)
[[-3.39021291]
 [ 1.14224386]]
(2, 1)
[[-3.39112425]
 [ 1.14233541]]
(2, 1)
[[-3.39203394]
 [ 1.1424268 ]]
(2, 1)
[[-3.39294199]
 [ 1.14251802]]
(2, 1)
[[-3.39384841]
 [ 1.14260908]]
(2, 1)
[[-3.39475319]
 [ 1.14269998]]
(2, 1)
[[-3.39565634]
 [ 1.14279071]]
(2, 1)
[[-3.39655786]
 [ 1.14288128]]
(2, 1)
[[-3.39745776]
 [ 1.14297168]]
(2, 1)
[[-3.39835604]
 [ 1.14306192]]
(2, 1)
[[-3.39925269]
 [ 1.143152  ]]
(2, 1)
[[-3.40014773]
 [ 1.14324192]]
(2, 1)
[[-3.40104116]
 [ 1.14333167]]
(2, 1)
[[-3.40193297]
 [ 1.14342127]]
```

```
(2, 1)
[[-3.40282318]
 [ 1.1435107 ]]
(2, 1)
[[-3.40371178]
 [ 1.14359997]]
(2, 1)
[[-3.40459879]
 [ 1.14368908]]
(2, 1)
[[-3.40548419]
 [ 1.14377802]]
(2, 1)
[[-3.406368  ]
 [ 1.14386681]]
(2, 1)
[[-3.40725021]
 [ 1.14395544]]
(2, 1)
[[-3.40813083]
 [ 1.14404391]]
(2, 1)
[[-3.40900987]
 [ 1.14413222]]
(2, 1)
[[-3.40988732]
 [ 1.14422037]]
(2, 1)
[[-3.41076319]
 [ 1.14430836]]
(2, 1)
[[-3.41163748]
 [ 1.14439619]]
(2, 1)
[[-3.4125102 ]
 [ 1.14448386]]
(2, 1)
[[-3.41338134]
 [ 1.14457138]]
(2, 1)
[[-3.41425091]
 [ 1.14465874]]
(2, 1)
[[-3.41511891]
 [ 1.14474594]]
(2, 1)
[[-3.41598535]
 [ 1.14483298]]
(2, 1)
[[-3.41685023]
 [ 1.14491987]]
(2, 1)
[[-3.41771355]
 [ 1.1450066 ]]
(2, 1)
[[-3.41857531]
 [ 1.14509317]]
```

```
(2, 1)
[[-3.41943552]
 [ 1.14517959]]
(2, 1)
[[-3.42029418]
 [ 1.14526585]]
(2, 1)
[[-3.42115129]
 [ 1.14535195]]
(2, 1)
[[-3.42200685]
 [ 1.1454379 ]]
(2, 1)
[[-3.42286088]
 [ 1.1455237 ]]
(2, 1)
[[-3.42371336]
 [ 1.14560934]]
(2, 1)
[[-3.42456431]
 [ 1.14569483]]
(2, 1)
[[-3.42541372]
 [ 1.14578016]]
(2, 1)
[[-3.4262616 ]
 [ 1.14586534]]
(2, 1)
[[-3.42710795]
 [ 1.14595037]]
(2, 1)
[[-3.42795278]
 [ 1.14603524]]
(2, 1)
[[-3.42879609]
 [ 1.14611996]]
(2, 1)
[[-3.42963787]
 [ 1.14620452]]
(2, 1)
[[-3.43047814]
 [ 1.14628894]]
(2, 1)
[[-3.43131689]
 [ 1.1463732 ]]
(2, 1)
[[-3.43215413]
 [ 1.14645731]]
(2, 1)
[[-3.43298987]
 [ 1.14654127]]
(2, 1)
[[-3.43382409]
 [ 1.14662507]]
(2, 1)
[[-3.43465681]
 [ 1.14670873]]
```

```
(2, 1)
[[-3.43548803]
 [ 1.14679224]]
(2, 1)
[[-3.43631775]
 [ 1.14687559]]
(2, 1)
[[-3.43714598]
 [ 1.14695879]]
(2, 1)
[[-3.43797271]
 [ 1.14704185]]
(2, 1)
[[-3.43879796]
 [ 1.14712475]]
(2, 1)
[[-3.43962171]
 [ 1.14720751]]
(2, 1)
[[-3.44044398]
 [ 1.14729011]]
(2, 1)
[[-3.44126477]
 [ 1.14737257]]
(2, 1)
[[-3.44208408]
 [ 1.14745488]]
(2, 1)
[[-3.44290191]
 [ 1.14753704]]
(2, 1)
[[-3.44371827]
 [ 1.14761905]]
(2, 1)
[[-3.44453316]
 [ 1.14770092]]
(2, 1)
[[-3.44534657]
 [ 1.14778263]]
(2, 1)
[[-3.44615852]
 [ 1.1478642 ]]
(2, 1)
[[-3.44696901]
 [ 1.14794562]]
(2, 1)
[[-3.44777804]
 [ 1.1480269 ]]
(2, 1)
[[-3.44858561]
 [ 1.14810803]]
(2, 1)
[[-3.44939172]
 [ 1.14818901]]
(2, 1)
[[-3.45019638]
 [ 1.14826985]]
```

```
(2, 1)
[[-3.45099959]
 [ 1.14835054]]
(2, 1)
[[-3.45180135]
 [ 1.14843108]]
(2, 1)
[[-3.45260166]
 [ 1.14851148]]
(2, 1)
[[-3.45340054]
 [ 1.14859174]]
(2, 1)
[[-3.45419797]
 [ 1.14867185]]
(2, 1)
[[-3.45499396]
 [ 1.14875182]]
(2, 1)
[[-3.45578852]
 [ 1.14883164]]
(2, 1)
[[-3.45658165]
 [ 1.14891132]]
(2, 1)
[[-3.45737335]
 [ 1.14899085]]
(2, 1)
[[-3.45816362]
 [ 1.14907024]]
(2, 1)
[[-3.45895247]
 [ 1.14914949]]
(2, 1)
[[-3.4597399]
 [ 1.1492286]]
(2, 1)
[[-3.4605259 ]
 [ 1.14930756]]
(2, 1)
[[-3.46131049]
 [ 1.14938638]]
(2, 1)
[[-3.46209366]
 [ 1.14946506]]
(2, 1)
[[-3.46287543]
 [ 1.1495436 ]]
(2, 1)
[[-3.46365578]
 [ 1.14962199]]
(2, 1)
[[-3.46443473]
 [ 1.14970024]]
(2, 1)
[[-3.46521227]
 [ 1.14977836]]
```

```
(2, 1)
[[-3.46598841]
 [ 1.14985633]]
(2, 1)
[[-3.46676315]
 [ 1.14993416]]
(2, 1)
[[-3.4675365 ]
 [ 1.15001185]]
(2, 1)
[[-3.46830845]
 [ 1.1500894 ]]
(2, 1)
[[-3.46907901]
 [ 1.15016681]]
(2, 1)
[[-3.46984818]
 [ 1.15024408]]
(2, 1)
[[-3.47061597]
 [ 1.15032122]]
(2, 1)
[[-3.47138237]
 [ 1.15039821]]
(2, 1)
[[-3.47214739]
 [ 1.15047506]]
(2, 1)
[[-3.47291103]
 [ 1.15055178]]
(2, 1)
[[-3.47367329]
 [ 1.15062836]]
(2, 1)
[[-3.47443418]
 [ 1.1507048 ]]
(2, 1)
[[-3.4751937]
 [ 1.1507811]]
(2, 1)
[[-3.47595185]
 [ 1.15085726]]
(2, 1)
[[-3.47670863]
 [ 1.15093329]]
(2, 1)
[[-3.47746405]
 [ 1.15100918]]
(2, 1)
[[-3.4782181 ]
 [ 1.15108493]]
(2, 1)
[[-3.4789708 ]
 [ 1.15116055]]
(2, 1)
[[-3.47972214]
 [ 1.15123603]]
```

```
(2, 1)
[[-3.48047212]
 [ 1.15131138]]
(2, 1)
[[-3.48122076]
 [ 1.15138658]]
(2, 1)
[[-3.48196804]
 [ 1.15146166]]
(2, 1)
[[-3.48271398]
 [ 1.15153659]]
(2, 1)
[[-3.48345857]
 [ 1.1516114 ]]
(2, 1)
[[-3.48420182]
 [ 1.15168606]]
(2, 1)
[[-3.48494373]
 [ 1.1517606 ]]
(2, 1)
[[-3.48568431]
 [ 1.151835  ]]
(2, 1)
[[-3.48642354]
 [ 1.15190926]]
(2, 1)
[[-3.48716145]
 [ 1.15198339]]
(2, 1)
[[-3.48789803]
 [ 1.15205739]]
(2, 1)
[[-3.48863327]
 [ 1.15213125]]
(2, 1)
[[-3.4893672 ]
 [ 1.15220498]]
(2, 1)
[[-3.4900998 ]
 [ 1.15227858]]
(2, 1)
[[-3.49083108]
 [ 1.15235204]]
(2, 1)
[[-3.49156104]
 [ 1.15242538]]
(2, 1)
[[-3.49228968]
 [ 1.15249858]]
(2, 1)
[[-3.49301701]
 [ 1.15257165]]
(2, 1)
[[-3.49374303]
 [ 1.15264458]]
```

```
(2, 1)
[[-3.49446775]
 [ 1.15271739]]
(2, 1)
[[-3.49519115]
 [ 1.15279006]]
(2, 1)
[[-3.49591325]
 [ 1.1528626 ]]
(2, 1)
[[-3.49663405]
 [ 1.15293502]]
(2, 1)
[[-3.49735355]
 [ 1.1530073 ]]
(2, 1)
[[-3.49807176]
 [ 1.15307945]]
(2, 1)
[[-3.49878866]
 [ 1.15315147]]
(2, 1)
[[-3.49950428]
 [ 1.15322336]]
(2, 1)
[[-3.50021861]
 [ 1.15329512]]
(2, 1)
[[-3.50093165]
 [ 1.15336676]]
(2, 1)
[[-3.5016434 ]
 [ 1.15343826]]
(2, 1)
[[-3.50235387]
 [ 1.15350963]]
(2, 1)
[[-3.50306306]
 [ 1.15358088]]
(2, 1)
[[-3.50377097]
 [ 1.153652  ]]
(2, 1)
[[-3.50447761]
 [ 1.15372299]]
(2, 1)
[[-3.50518297]
 [ 1.15379385]]
(2, 1)
[[-3.50588706]
 [ 1.15386458]]
(2, 1)
[[-3.50658988]
 [ 1.15393519]]
(2, 1)
[[-3.50729143]
 [ 1.15400567]]
```

```
(2, 1)
[[-3.50799172]
 [ 1.15407602]]
(2, 1)
[[-3.50869075]
 [ 1.15414624]]
(2, 1)
[[-3.50938852]
 [ 1.15421634]]
(2, 1)
[[-3.51008503]
 [ 1.15428631]]
(2, 1)
[[-3.51078028]
 [ 1.15435616]]
(2, 1)
[[-3.51147428]
 [ 1.15442588]]
(2, 1)
[[-3.51216703]
 [ 1.15449547]]
(2, 1)
[[-3.51285853]
 [ 1.15456494]]
(2, 1)
[[-3.51354879]
 [ 1.15463428]]
(2, 1)
[[-3.5142378]
 [ 1.1547035]]
(2, 1)
[[-3.51492556]
 [ 1.1547726 ]]
(2, 1)
[[-3.51561209]
 [ 1.15484157]]
(2, 1)
[[-3.51629738]
 [ 1.15491041]]
(2, 1)
[[-3.51698144]
 [ 1.15497913]]
(2, 1)
[[-3.51766426]
 [ 1.15504773]]
(2, 1)
[[-3.51834585]
 [ 1.1551162 ]]
(2, 1)
[[-3.51902621]
 [ 1.15518455]]
(2, 1)
[[-3.51970535]
 [ 1.15525278]]
(2, 1)
[[-3.52038326]
 [ 1.15532088]]
```

```
(2, 1)
[[-3.52105995]
 [ 1.15538886]]
(2, 1)
[[-3.52173542]
 [ 1.15545672]]
(2, 1)
[[-3.52240968]
 [ 1.15552446]]
(2, 1)
[[-3.52308271]
 [ 1.15559207]]
(2, 1)
[[-3.52375454]
 [ 1.15565956]]
(2, 1)
[[-3.52442515]
 [ 1.15572693]]
(2, 1)
[[-3.52509455]
 [ 1.15579418]]
(2, 1)
[[-3.52576275]
 [ 1.15586131]]
(2, 1)
[[-3.52642975]
 [ 1.15592832]]
(2, 1)
[[-3.52709554]
 [ 1.1559952 ]]
(2, 1)
[[-3.52776013]
 [ 1.15606197]]
(2, 1)
[[-3.52842352]
 [ 1.15612861]]
(2, 1)
[[-3.52908572]
 [ 1.15619514]]
(2, 1)
[[-3.52974672]
 [ 1.15626154]]
(2, 1)
[[-3.53040653]
 [ 1.15632783]]
(2, 1)
[[-3.53106515]
 [ 1.15639399]]
(2, 1)
[[-3.53172259]
 [ 1.15646004]]
(2, 1)
[[-3.53237884]
 [ 1.15652597]]
(2, 1)
[[-3.53303391]
 [ 1.15659178]]
```

```
(2, 1)
[[-3.53368779]
 [ 1.15665747]]
(2, 1)
[[-3.5343405 ]
 [ 1.15672304]]
(2, 1)
[[-3.53499203]
 [ 1.15678849]]
(2, 1)
[[-3.53564239]
 [ 1.15685383]]
(2, 1)
[[-3.53629157]
 [ 1.15691904]]
(2, 1)
[[-3.53693958]
 [ 1.15698414]]
(2, 1)
[[-3.53758643]
 [ 1.15704913]]
(2, 1)
[[-3.53823211]
 [ 1.15711399]]
(2, 1)
[[-3.53887663]
 [ 1.15717874]]
(2, 1)
[[-3.53951998]
 [ 1.15724337]]
(2, 1)
[[-3.54016217]
 [ 1.15730789]]
(2, 1)
[[-3.54080321]
 [ 1.15737229]]
(2, 1)
[[-3.54144309]
 [ 1.15743657]]
(2, 1)
[[-3.54208182]
 [ 1.15750074]]
(2, 1)
[[-3.5427194 ]
 [ 1.15756479]]
(2, 1)
[[-3.54335582]
 [ 1.15762872]]
(2, 1)
[[-3.5439911 ]
 [ 1.15769255]]
(2, 1)
[[-3.54462524]
 [ 1.15775625]]
(2, 1)
[[-3.54525823]
 [ 1.15781984]]
```

```
(2, 1)
[[-3.54589008]
 [ 1.15788332]]
(2, 1)
[[-3.54652079]
 [ 1.15794668]]
(2, 1)
[[-3.54715037]
 [ 1.15800993]]
(2, 1)
[[-3.54777881]
 [ 1.15807306]]
(2, 1)
[[-3.54840611]
 [ 1.15813608]]
(2, 1)
[[-3.54903229]
 [ 1.15819899]]
(2, 1)
[[-3.54965734]
 [ 1.15826178]]
(2, 1)
[[-3.55028126]
 [ 1.15832446]]
(2, 1)
[[-3.55090406]
 [ 1.15838703]]
(2, 1)
[[-3.55152573]
 [ 1.15844948]]
(2, 1)
[[-3.55214628]
 [ 1.15851182]]
(2, 1)
[[-3.55276572]
 [ 1.15857405]]
(2, 1)
[[-3.55338403]
 [ 1.15863617]]
(2, 1)
[[-3.55400124]
 [ 1.15869817]]
(2, 1)
[[-3.55461733]
 [ 1.15876006]]
(2, 1)
[[-3.55523231]
 [ 1.15882185]]
(2, 1)
[[-3.55584618]
 [ 1.15888352]]
(2, 1)
[[-3.55645894]
 [ 1.15894508]]
(2, 1)
[[-3.5570706 ]
 [ 1.15900652]]
```

```
(2, 1)
[[-3.55768116]
 [ 1.15906786]]
(2, 1)
[[-3.55829062]
 [ 1.15912909]]
(2, 1)
[[-3.55889898]
 [ 1.1591902 ]]
(2, 1)
[[-3.55950624]
 [ 1.15925121]]
(2, 1)
[[-3.56011241]
 [ 1.15931211]]
(2, 1)
[[-3.56071748]
 [ 1.15937289]]
(2, 1)
[[-3.56132147]
 [ 1.15943357]]
(2, 1)
[[-3.56192436]
 [ 1.15949414]]
(2, 1)
[[-3.56252617]
 [ 1.15955459]]
(2, 1)
[[-3.56312689]
 [ 1.15961494]]
(2, 1)
[[-3.56372653]
 [ 1.15967518]]
(2, 1)
[[-3.56432509]
 [ 1.15973532]]
(2, 1)
[[-3.56492257]
 [ 1.15979534]]
(2, 1)
[[-3.56551898]
 [ 1.15985525]]
(2, 1)
[[-3.56611431]
 [ 1.15991506]]
(2, 1)
[[-3.56670856]
 [ 1.15997476]]
(2, 1)
[[-3.56730175]
 [ 1.16003435]]
(2, 1)
[[-3.56789386]
 [ 1.16009384]]
(2, 1)
[[-3.56848491]
 [ 1.16015321]]
```

```
(2, 1)
[[-3.56907489]
 [ 1.16021248]]
(2, 1)
[[-3.56966381]
 [ 1.16027165]]
(2, 1)
[[-3.57025167]
 [ 1.1603307 ]]
(2, 1)
[[-3.57083847]
 [ 1.16038965]]
(2, 1)
[[-3.5714242]
 [ 1.1604485]]
(2, 1)
[[-3.57200889]
 [ 1.16050724]]
(2, 1)
[[-3.57259252]
 [ 1.16056587]]
(2, 1)
[[-3.5731751 ]
 [ 1.16062439]]
(2, 1)
[[-3.57375662]
 [ 1.16068281]]
(2, 1)
[[-3.5743371 ]
 [ 1.16074113]]
(2, 1)
[[-3.57491654]
 [ 1.16079934]]
(2, 1)
[[-3.57549493]
 [ 1.16085745]]
(2, 1)
[[-3.57607227]
 [ 1.16091545]]
(2, 1)
[[-3.57664858]
 [ 1.16097334]]
(2, 1)
[[-3.57722384]
 [ 1.16103113]]
(2, 1)
[[-3.57779807]
 [ 1.16108882]]
(2, 1)
[[-3.57837127]
 [ 1.16114641]]
(2, 1)
[[-3.57894343]
 [ 1.16120389]]
(2, 1)
[[-3.57951456]
 [ 1.16126126]]
```

```
(2, 1)
[[-3.58008466]
 [ 1.16131853]]
(2, 1)
[[-3.58065373]
 [ 1.1613757 ]]
(2, 1)
[[-3.58122178]
 [ 1.16143277]]
(2, 1)
[[-3.5817888 ]
 [ 1.16148973]]
(2, 1)
[[-3.5823548]
 [ 1.1615466]]
(2, 1)
[[-3.58291978]
 [ 1.16160335]]
(2, 1)
[[-3.58348375]
 [ 1.16166001]]
(2, 1)
[[-3.58404669]
 [ 1.16171656]]
(2, 1)
[[-3.58460862]
 [ 1.16177302]]
(2, 1)
[[-3.58516954]
 [ 1.16182937]]
(2, 1)
[[-3.58572945]
 [ 1.16188561]]
(2, 1)
[[-3.58628834]
 [ 1.16194176]]
(2, 1)
[[-3.58684623]
 [ 1.16199781]]
(2, 1)
[[-3.58740312]
 [ 1.16205375]]
(2, 1)
[[-3.587959 ]
 [ 1.1621096]]
(2, 1)
[[-3.58851388]
 [ 1.16216534]]
(2, 1)
[[-3.58906776]
 [ 1.16222098]]
(2, 1)
[[-3.58962064]
 [ 1.16227653]]
(2, 1)
[[-3.59017252]
 [ 1.16233197]]
```

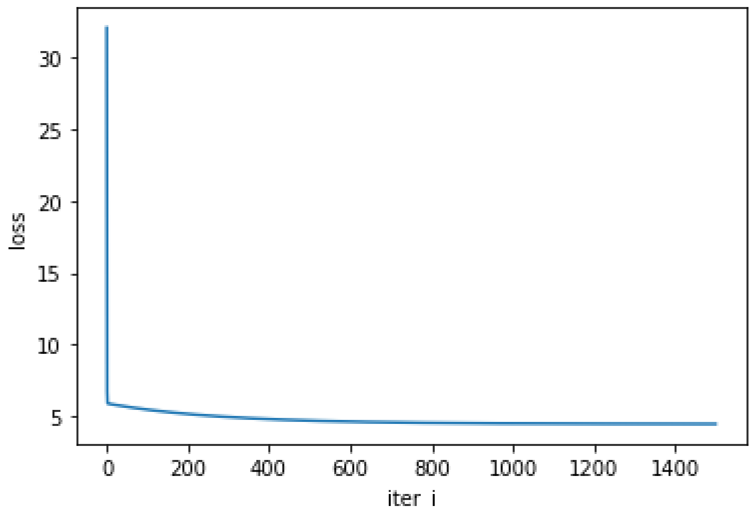
```
(2, 1)
[[-3.59072341]
 [ 1.16238731]]
(2, 1)
[[-3.5912733 ]
 [ 1.16244256]]
(2, 1)
[[-3.59182221]
 [ 1.1624977 ]]
(2, 1)
[[-3.59237012]
 [ 1.16255274]]
(2, 1)
[[-3.59291705]
 [ 1.16260769]]
(2, 1)
[[-3.59346299]
 [ 1.16266253]]
(2, 1)
[[-3.59400795]
 [ 1.16271728]]
(2, 1)
[[-3.59455192]
 [ 1.16277193]]
(2, 1)
[[-3.59509491]
 [ 1.16282648]]
(2, 1)
[[-3.59563693]
 [ 1.16288093]]
(2, 1)
[[-3.59617797]
 [ 1.16293528]]
(2, 1)
[[-3.59671803]
 [ 1.16298954]]
(2, 1)
[[-3.59725712]
 [ 1.16304369]]
(2, 1)
[[-3.59779524]
 [ 1.16309775]]
(2, 1)
[[-3.59833239]
 [ 1.16315172]]
(2, 1)
[[-3.59886857]
 [ 1.16320558]]
(2, 1)
[[-3.59940378]
 [ 1.16325935]]
(2, 1)
[[-3.59993803]
 [ 1.16331302]]
(2, 1)
[[-3.60047131]
 [ 1.16336659]]
```

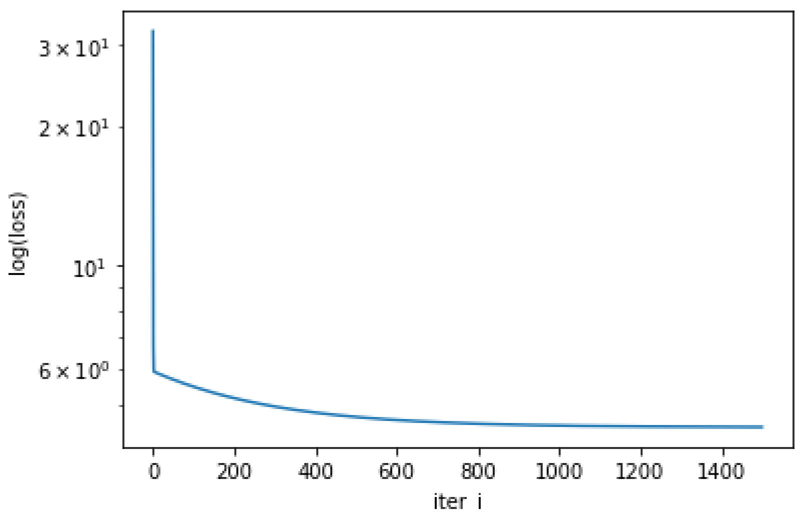
```
(2, 1)
[[-3.60100363]
 [ 1.16342007]]
(2, 1)
[[-3.601535  ]
 [ 1.16347345]]
(2, 1)
[[-3.60206541]
 [ 1.16352674]]
(2, 1)
[[-3.60259486]
 [ 1.16357993]]
(2, 1)
[[-3.60312335]
 [ 1.16363302]]
(2, 1)
[[-3.60365089]
 [ 1.16368602]]
(2, 1)
[[-3.60417749]
 [ 1.16373892]]
(2, 1)
[[-3.60470313]
 [ 1.16379173]]
(2, 1)
[[-3.60522783]
 [ 1.16384444]]
(2, 1)
[[-3.60575158]
 [ 1.16389705]]
(2, 1)
[[-3.60627438]
 [ 1.16394958]]
(2, 1)
[[-3.60679624]
 [ 1.164002  ]]
(2, 1)
[[-3.60731717]
 [ 1.16405434]]
(2, 1)
[[-3.60783715]
 [ 1.16410657]]
(2, 1)
[[-3.6083562 ]
 [ 1.16415872]]
(2, 1)
[[-3.60887431]
 [ 1.16421077]]
(2, 1)
[[-3.60939148]
 [ 1.16426272]]
(2, 1)
[[-3.60990773]
 [ 1.16431459]]
(2, 1)
[[-3.61042304]
 [ 1.16436635]]
```

```
(2, 1)
[[-3.61093743]
 [ 1.16441803]]
(2, 1)
[[-3.61145088]
 [ 1.16446961]]
(2, 1)
[[-3.61196342]
 [ 1.1645211 ]]
(2, 1)
[[-3.61247502]
 [ 1.1645725 ]]
(2, 1)
[[-3.61298571]
 [ 1.1646238 ]]
(2, 1)
[[-3.61349548]
 [ 1.16467501]]
(2, 1)
[[-3.61400432]
 [ 1.16472613]]
(2, 1)
[[-3.61451225]
 [ 1.16477716]]
(2, 1)
[[-3.61501927]
 [ 1.16482809]]
(2, 1)
[[-3.61552536]
 [ 1.16487894]]
(2, 1)
[[-3.61603055]
 [ 1.16492969]]
(2, 1)
[[-3.61653483]
 [ 1.16498035]]
(2, 1)
[[-3.6170382 ]
 [ 1.16503092]]
(2, 1)
[[-3.61754066]
 [ 1.1650814 ]]
(2, 1)
[[-3.61804221]
 [ 1.16513178]]
(2, 1)
[[-3.61854286]
 [ 1.16518208]]
(2, 1)
[[-3.61904261]
 [ 1.16523228]]
(2, 1)
[[-3.61954146]
 [ 1.1652824 ]]
(2, 1)
[[-3.62003941]
 [ 1.16533242]]
```

```
(2, 1)
[[-3.62053646]
 [ 1.16538236]]
(2, 1)
[[-3.62103261]
 [ 1.1654322 ]]
(2, 1)
[[-3.62152787]
 [ 1.16548195]]
(2, 1)
[[-3.62202224]
 [ 1.16553162]]
(2, 1)
[[-3.62251571]
 [ 1.16558119]]
(2, 1)
[[-3.6230083 ]
 [ 1.16563068]]
(2, 1)
[[-3.6235    ]
 [ 1.16568008]]
(2, 1)
[[-3.62399081]
 [ 1.16572938]]
(2, 1)
[[-3.62448074]
 [ 1.1657786 ]]
(2, 1)
[[-3.62496978]
 [ 1.16582773]]
(2, 1)
[[-3.62545795]
 [ 1.16587677]]
(2, 1)
[[-3.62594523]
 [ 1.16592573]]
(2, 1)
[[-3.62643163]
 [ 1.16597459]]
(2, 1)
[[-3.62691716]
 [ 1.16602337]]
(2, 1)
[[-3.62740182]
 [ 1.16607206]]
(2, 1)
[[-3.62788559]
 [ 1.16612066]]
(2, 1)
[[-3.6283685 ]
 [ 1.16616917]]
(2, 1)
[[-3.62885054]
 [ 1.1662176 ]]
(2, 1)
[[-3.6293317 ]
 [ 1.16626593]]
```

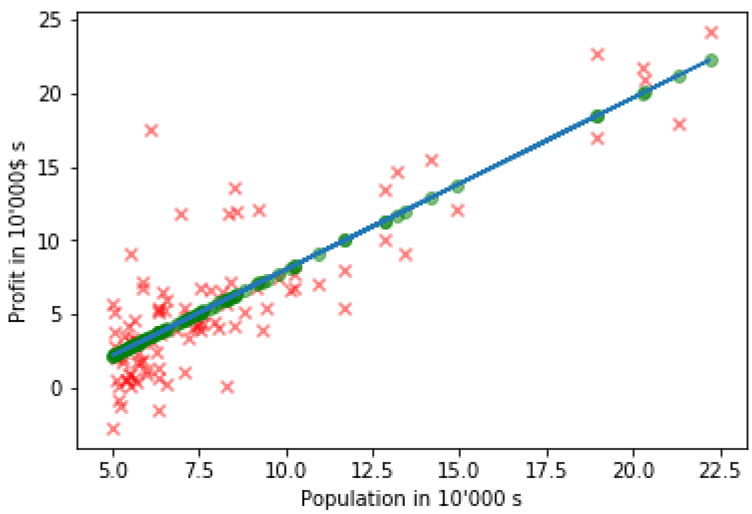
```
(2, 1)
[[-3.62981201]
 [ 1.16631419]]
(2, 1)
estimated theta value [-3.63029144  1.16636235]
[[-3.63029144]
 [ 1.16636235]]
(2, 1)
resulting loss [4.48338826]
```





2.2.4 [10pt] After you are finished, use your final parameters to plot the linear fit. The result should look something like on the figure below. Use the `predict()` function.

In [13]:
```
plt.scatter(X, y, marker='x', color='r', alpha=0.5)
x_start, x_end = 5, 25
theta = [[-3.63029144],[1.16636235]]
theta = np.array(theta)
newy = predict(X, theta)
plt.scatter(X, newy, marker='o', color='g', alpha=0.5)
plt.xlabel('Population in 10\'000 s')
plt.ylabel('Profit in 10\'000$ s')
plt.plot(X,newy.T)
plt.show()
```



Now use your final values for θ and the `predict()` function to make predictions on profits in areas of 35,000 and 70,000 people.

In [43]:
```
print(predict(np.array([[35000]]),theta))
print(predict(np.array([[70000]]),theta))
```

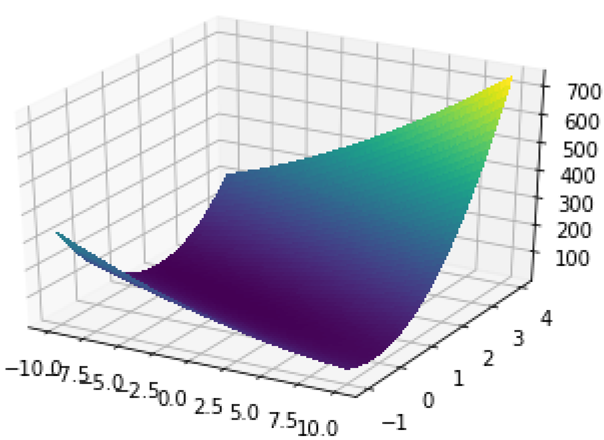
```
[[40819.05195856]]
[[81641.73420856]]
```

To understand the cost function better, you will now plot the cost over a 2-dimensional grid of values. You will not need to code anything new for this part, but you should understand how the code you have written already is creating these images.

In [44]:

```
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.cm as cm
limits = [(-10, 10), (-1, 4)]
space = [np.linspace(*limit, 100) for limit in limits]
theta_1_grid, theta_2_grid = np.meshgrid(*space)
theta_meshgrid = np.vstack([theta_1_grid.ravel(), theta_2_grid.ravel()])
loss_test_vals_flat = (((add_column(X) @ theta_meshgrid - y)**2).mean(axis=0)/
2)
loss_test_vals_grid = loss_test_vals_flat.reshape(theta_1_grid.shape)
print(theta_1_grid.shape, theta_2_grid.shape, loss_test_vals_grid.shape)

plt.gca(projection='3d').plot_surface(theta_1_grid, theta_2_grid,
                                      loss_test_vals_grid, cmap=cm.viridis,
                                      linewidth=0, antialiased=False)
xs, ys = np.hstack(theta_values).tolist()
zs = np.array(loss_values)
plt.gca(projection='3d').plot(xs, ys, zs, c='r')
plt.xlim(*limits[0])
plt.ylim(*limits[1])
plt.show()

plt.contour(theta_1_grid, theta_2_grid, loss_test_vals_grid, levels=np.logspac
e(-2, 3, 20))
plt.plot(xs, ys)
plt.scatter(xs, ys, alpha=0.005)
plt.xlim(*limits[0])
plt.ylim(*limits[1])
plt.show()
```

```
(100, 100) (100, 100) (100, 100)

---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-44-0f0fb093c336> in <module>
     14 xs, ys = np.hstack(theta_values).tolist()
     15 zs = np.array(loss_values)
---> 16 plt.gca(projection='3d').plot(xs, ys, zs, c='r')
     17 plt.xlim(*limits[0])
     18 plt.ylim(*limits[1])

~\Anaconda3\lib\site-packages\mpl_toolkits\mplot3d\axes3d.py in plot(self, x
s, ys, zdir, *args, **kwargs)
   1528
   1529         # Match length
-> 1530         zs = np.broadcast_to(zs, len(xs))
   1531
   1532         lines = super().plot(xs, ys, *args, **kwargs)

~\Anaconda3\lib\site-packages\numpy\lib\stride_tricks.py in broadcast_to(arra
y, shape, subok)
    180            [1, 2, 3]])
    181     """
--> 182     return _broadcast_to(array, shape, subok=subok, readonly=True)
    183
    184

~\Anaconda3\lib\site-packages\numpy\lib\stride_tricks.py in _broadcast_to(arr
ay, shape, subok, readonly)
    127     it = np.nditer(
    128         (array,), flags=['multi_index', 'refs_ok', 'zerosize_ok'] + e
xtras,
--> 129         op_flags=[op_flag], itershape=shape, order='C')
    130     with it:
    131         # never really has writebackifcopy semantics

ValueError: input operand has more dimensions than allowed by the axis remapp
ing
```



3. Linear regression with multiple input features

3.1 [20pt] Copy-paste your `add_column` , `predict` , `loss` and `loss grad` implementations from above and modify your code of linear regression with one variable to support any number of input features (vectorize your code.)

In [60]:
```
data = np.loadtxt('ex1data2.txt', delimiter=',')
X, y = data[:, :-1], data[:, -1, np.newaxis]
n = data.shape[0]
print(X.shape, y.shape, n)
print(X[:10], '\n', y[:10])
```

```
(47, 2) (47, 1) 47
[[2.104e+03 3.000e+00]
 [1.600e+03 3.000e+00]
 [2.400e+03 3.000e+00]
 [1.416e+03 2.000e+00]
 [3.000e+03 4.000e+00]
 [1.985e+03 4.000e+00]
 [1.534e+03 3.000e+00]
 [1.427e+03 3.000e+00]
 [1.380e+03 3.000e+00]
 [1.494e+03 3.000e+00]]
 [[399900.]
 [329900.]
 [369000.]
 [232000.]
 [539900.]
 [299900.]
 [314900.]
 [198999.]
 [212000.]
 [242500.]]
```

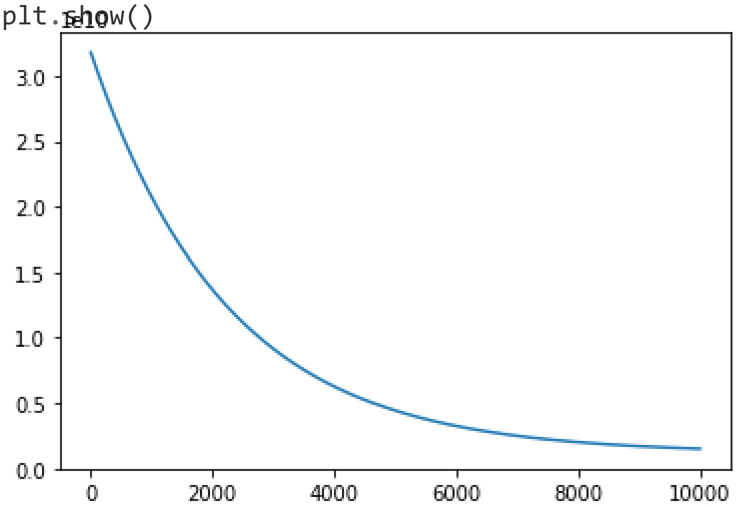
In [58]:
```
theta_init = np.zeros((3,1))
print(theta_init.shape[0] != 2)
X_prime = add_column(X)
print(X_prime.shape)
print(X_prime)
print(X_prime.shape[1])
```

```
True
(97, 2)
[[ 1.      6.1101]
 [ 1.      5.5277]
 [ 1.      8.5186]
 [ 1.      7.0032]
 [ 1.      5.8598]
 [ 1.      8.3829]
 [ 1.      7.4764]
 [ 1.      8.5781]
 [ 1.      6.4862]
 [ 1.      5.0546]
 [ 1.      5.7107]
 [ 1.     14.164 ]
 [ 1.      5.734 ]
 [ 1.      8.4084]
 [ 1.      5.6407]
 [ 1.      5.3794]
 [ 1.      6.3654]
 [ 1.      5.1301]
 [ 1.      6.4296]
 [ 1.      7.0708]
 [ 1.      6.1891]
 [ 1.     20.27  ]
 [ 1.      5.4901]
 [ 1.      6.3261]
 [ 1.      5.5649]
 [ 1.     18.945 ]
 [ 1.     12.828 ]
 [ 1.     10.957 ]
 [ 1.     13.176 ]
 [ 1.     22.203 ]
 [ 1.      5.2524]
 [ 1.      6.5894]
 [ 1.      9.2482]
 [ 1.      5.8918]
 [ 1.      8.2111]
 [ 1.      7.9334]
 [ 1.      8.0959]
 [ 1.      5.6063]
 [ 1.     12.836 ]
 [ 1.      6.3534]
 [ 1.      5.4069]
 [ 1.      6.8825]
 [ 1.     11.708 ]
 [ 1.      5.7737]
 [ 1.      7.8247]
 [ 1.      7.0931]
 [ 1.      5.0702]
 [ 1.      5.8014]
 [ 1.     11.7   ]
 [ 1.      5.5416]
 [ 1.      7.5402]
 [ 1.      5.3077]
 [ 1.      7.4239]
 [ 1.      7.6031]
 [ 1.      6.3328]
```

```
 [ 1.      6.3589]
 [ 1.      6.2742]
 [ 1.      5.6397]
 [ 1.      9.3102]
 [ 1.      9.4536]
 [ 1.      8.8254]
 [ 1.      5.1793]
 [ 1.     21.279 ]
 [ 1.     14.908 ]
 [ 1.     18.959 ]
 [ 1.      7.2182]
 [ 1.      8.2951]
 [ 1.     10.236 ]
 [ 1.      5.4994]
 [ 1.     20.341 ]
 [ 1.     10.136 ]
 [ 1.      7.3345]
 [ 1.      6.0062]
 [ 1.      7.2259]
 [ 1.      5.0269]
 [ 1.      6.5479]
 [ 1.      7.5386]
 [ 1.      5.0365]
 [ 1.     10.274 ]
 [ 1.      5.1077]
 [ 1.      5.7292]
 [ 1.      5.1884]
 [ 1.      6.3557]
 [ 1.      9.7687]
 [ 1.      6.5159]
 [ 1.      8.5172]
 [ 1.      9.1802]
 [ 1.      6.002 ]
 [ 1.      5.5204]
 [ 1.      5.0594]
 [ 1.      5.7077]
 [ 1.      7.6366]
 [ 1.      5.8707]
 [ 1.      5.3054]
 [ 1.      8.2934]
 [ 1.     13.394 ]
 [ 1.      5.4369]]
2
```

In [62]:
```
import numpy as np
def add_column(X,i):
    return np.insert(X,i,1,axis=1)

def predict(X, theta):
    X_prime = add_column(X,0)
    #count = 1
    #while X_prime.shape[1] != theta[0].shape:
    #    X_prime = add_column(X,count)
     #   count+=1
    Theta_T = np.transpose(theta)
    X_prime = np.transpose(X_prime)
    pred = np.matmul(Theta_T,X_prime)
    return pred

def loss(X, y, theta):
    X_prime = add_column(X,0)
    #count = 1
    #while X_prime.shape[1] != theta[0].shape:
     #   X_prime = add_column(X,count)
      #  count+=1
    Theta_T = np.transpose(theta)
    #print(np.array(theta))
    #print(theta.shape)
    total = 0
    for i in range(len(X_prime)):
        pred = np.matmul(Theta_T,X_prime[i])
        total += (pred-y[i])**2
    #raise NotImplementedError("Compute the model loss; use the predict() func
tion")
    loss = total/194
    return loss

import scipy.optimize
from functools import partial

def loss_gradient(X, y, theta):
    X_prime = add_column(X,0)
    #count = 1
    #while X_prime.shape[1] != theta[0].shape:
     #   X_prime = add_column(X,count)
      #  count+=1
    Theta_T = np.transpose(theta)
    total = 0
    for i in range(len(X_prime)):
        pred = np.matmul(Theta_T,X_prime[i])
        predactualminustest = (pred - y[i])
        total += (np.matmul(predactualminustest,np.array([X_prime[i]])))
    loss_grad = total/97
    return np.array([loss_grad]).T


theta_init = np.zeros((3, 1))
result = run_gd(loss, loss_gradient, X, y, theta_init, n_iter=10000, lr=1e-10)
theta_est, loss_values, theta_values = result
```

```
plt.plot(loss_values)
plt.show()
```



3.2 [20pt] Draw a histogam of values for the first and second feature. Why is feature normalization important? Normalize features and re-run the gradient decent. Compare loss plots that you get with and without feature normalization.

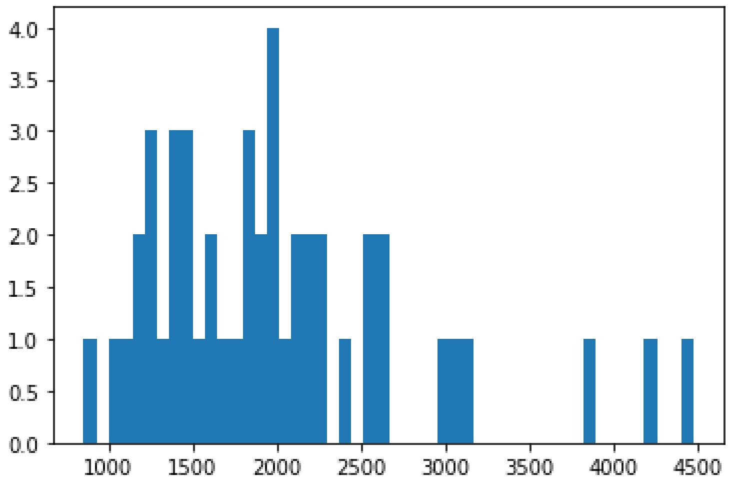
In [66]:
```
print(loss_values[5000:5010])
```

```
[array([4.42411702e+09]), array([4.42265636e+09]), array([4.42119635e+09]), a
rray([4.419737e+09]), array([4.4182783e+09]), array([4.41682026e+09]), array
([4.41536287e+09]), array([4.41390613e+09]), array([4.41245004e+09]), array
([4.41099461e+09])]
```

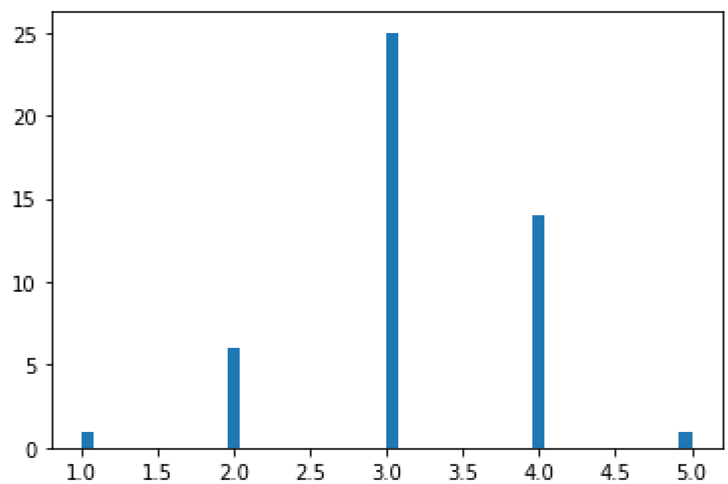
In [74]:
```
X
```

Out[74]:
```
array([[2.104e+03, 3.000e+00],
       [1.600e+03, 3.000e+00],
       [2.400e+03, 3.000e+00],
       [1.416e+03, 2.000e+00],
       [3.000e+03, 4.000e+00],
       [1.985e+03, 4.000e+00],
       [1.534e+03, 3.000e+00],
       [1.427e+03, 3.000e+00],
       [1.380e+03, 3.000e+00],
       [1.494e+03, 3.000e+00],
       [1.940e+03, 4.000e+00],
       [2.000e+03, 3.000e+00],
       [1.890e+03, 3.000e+00],
       [4.478e+03, 5.000e+00],
       [1.268e+03, 3.000e+00],
       [2.300e+03, 4.000e+00],
       [1.320e+03, 2.000e+00],
       [1.236e+03, 3.000e+00],
       [2.609e+03, 4.000e+00],
       [3.031e+03, 4.000e+00],
       [1.767e+03, 3.000e+00],
       [1.888e+03, 2.000e+00],
       [1.604e+03, 3.000e+00],
       [1.962e+03, 4.000e+00],
       [3.890e+03, 3.000e+00],
       [1.100e+03, 3.000e+00],
       [1.458e+03, 3.000e+00],
       [2.526e+03, 3.000e+00],
       [2.200e+03, 3.000e+00],
       [2.637e+03, 3.000e+00],
       [1.839e+03, 2.000e+00],
       [1.000e+03, 1.000e+00],
       [2.040e+03, 4.000e+00],
       [3.137e+03, 3.000e+00],
       [1.811e+03, 4.000e+00],
       [1.437e+03, 3.000e+00],
       [1.239e+03, 3.000e+00],
       [2.132e+03, 4.000e+00],
       [4.215e+03, 4.000e+00],
       [2.162e+03, 4.000e+00],
       [1.664e+03, 2.000e+00],
       [2.238e+03, 3.000e+00],
       [2.567e+03, 4.000e+00],
       [1.200e+03, 3.000e+00],
       [8.520e+02, 2.000e+00],
       [1.852e+03, 4.000e+00],
       [1.203e+03, 3.000e+00]])
```

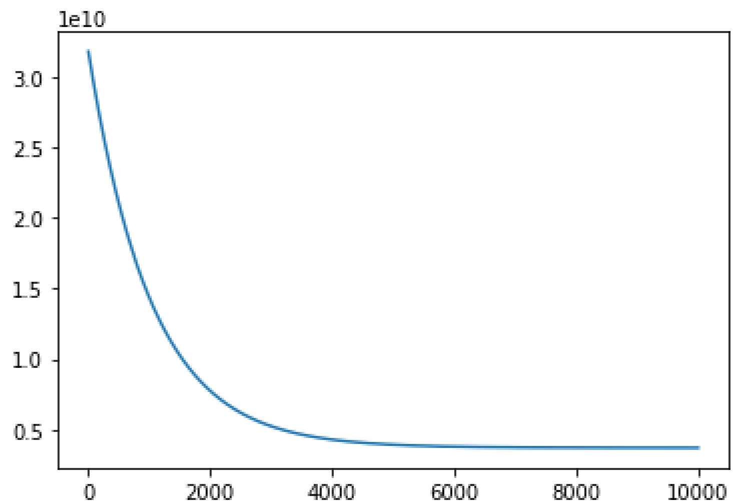
In [75]:
```
plt.hist([x[0] for x in X], bins=50)
plt.show()
```



In [76]:
```
plt.hist([x[1] for x in X], bins= 50)
plt.show()
```



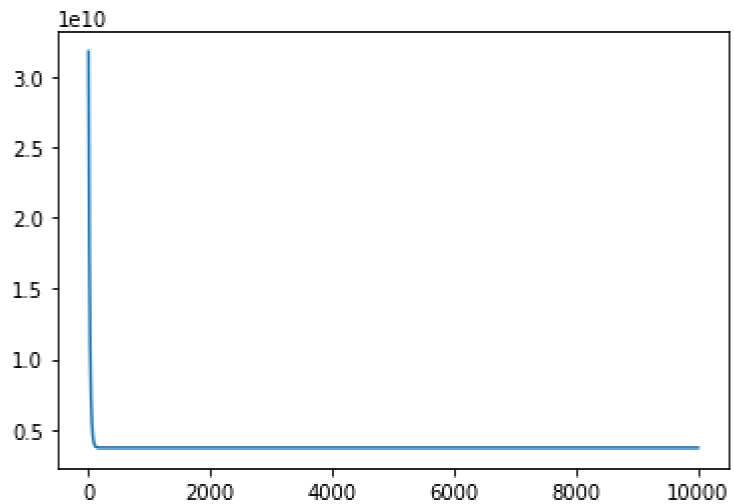
In [77]:
```
theta_init = np.zeros((3, 1))
X_normed = np.zeros_like(X)
#raise NotImplementedError("Run gd on normalized versions of feature vectors")
result = run_gd(loss, loss_gradient, X_normed, y, theta_init, n_iter=10000, lr
=1e-3)
theta_est, loss_values, theta_values = result

plt.plot(loss_values)
plt.show()
```



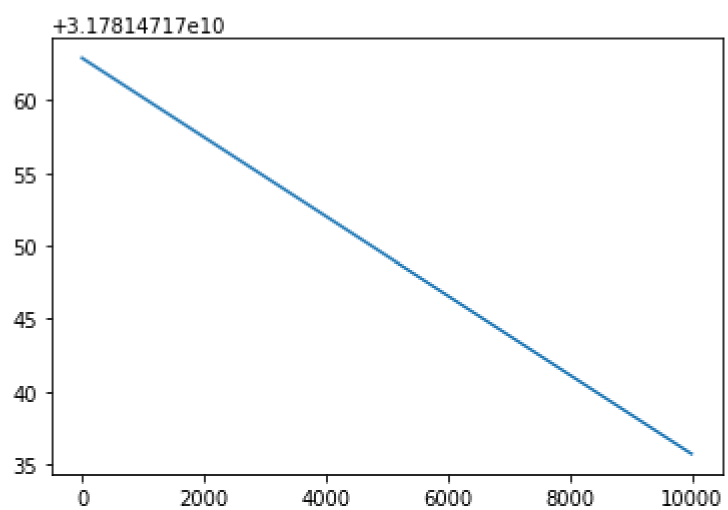
Feature normalization is important because an equal amount of proportions of features will give the most accurate outcome. If you have a feature that overtakes the other features by a substantial amount, then the outcome will be skewed towards that feature.

3.3 [10pt] How can we choose an appropriate learning rate? See what will happen if the learning rate is too small or too large for normalized and not normalized cases?

In [79]:
```
#raise NotImplementedError("Plot loss behaviour with multiple different learni
ng rates")
theta_init = np.zeros((3, 1))
X_normed = np.zeros_like(X)
result = run_gd(loss, loss_gradient, X_normed, y, theta_init, n_iter=10000, lr
=.05)
theta_est, loss_values, theta_values = result
plt.plot(loss_values)
plt.show()
```



In [80]:
```
theta_init = np.zeros((3, 1))
X_normed = np.zeros_like(X)
result = run_gd(loss, loss_gradient, X_normed, y, theta_init, n_iter=10000, lr
=.0000000000001)
theta_est, loss_values, theta_values = result
plt.plot(loss_values)
plt.show()
```



To choose an appropriate learning rate you slowly increase the learning rate to find the smallest learning rate that gives the least loss.

In [ ]: