vectors part 2

for the following slides, let’s use this example vector

you are testing a drug treatment on a group of cells and measure their reactions. you have 100 cells and can measure positive or negative numbers – do the cells exhibit more or less growth under the drug treatment. we also know how old each cell line is in days.

to create this data in matlab, we can use ***rand*** *functions, explained in your tutorials!*

growth = **randn(1,100)** % draw 100 random numbers from standard normal distribution

age = **randi(30,1,100)** % draw 100 random integers, with replacement, from range 1:30

for the following slides, let’s use this example vector

you are testing a drug treatment on a group of cells and measure their reactions. you have 100 cells and can measure positive or negative numbers – do the cells exhibit more or less growth under the drug treatment. we also know how old each cell line is in days.

to create this data in matlab, we can use ***rand*** *functions, explained in your tutorials!*

growth = **randn(1,100)** % draw 100 random numbers from standard normal distribution

age = **randi(30,1,100)** % draw 100 random integers, with replacement, from range 1:30

warm up: what do the following do?

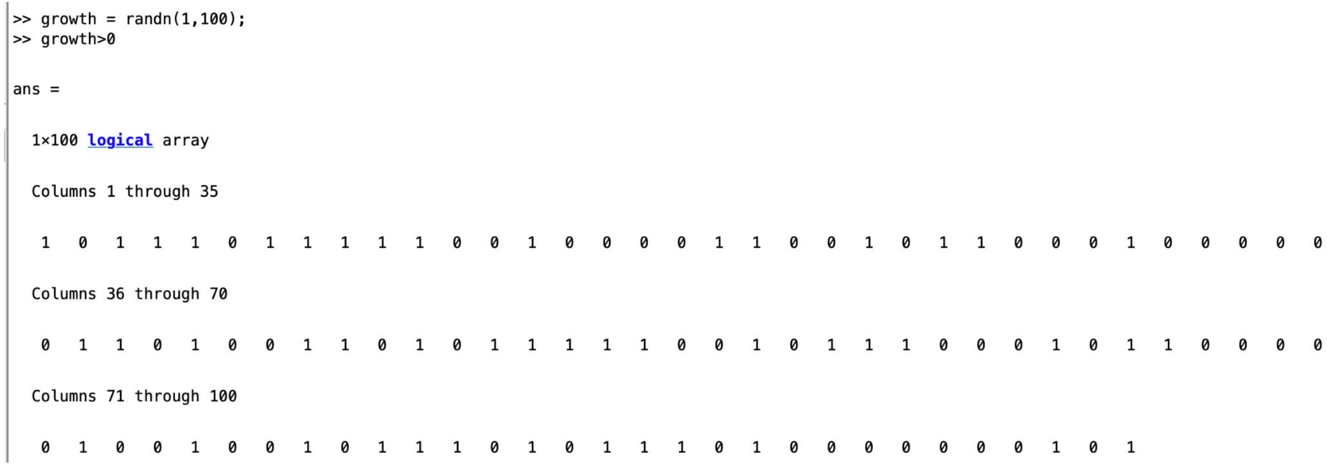
mean(growth)

growth(50:2:end)

growth(end:-1:1)

combining logic and vectors

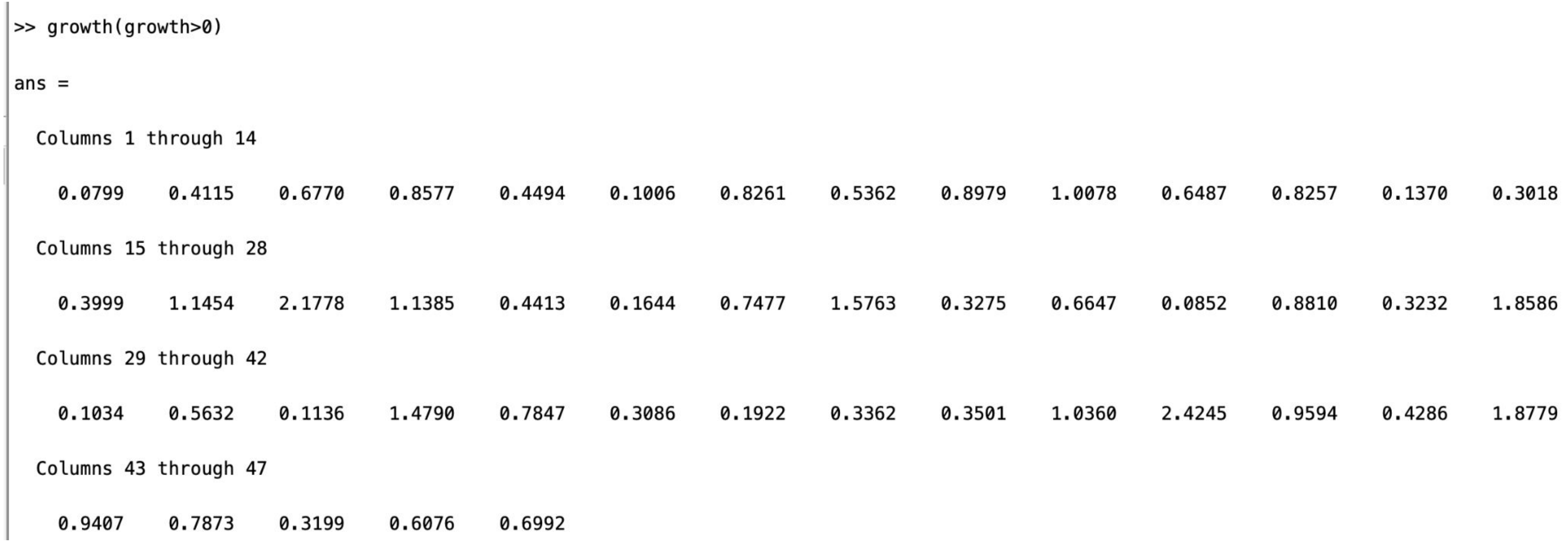
- a huge power of vectors is not having to use all of them at once!
- just like in previous modules, we can create more things by combining the building blocks we already know about – for example, boolean logic
- using boolean logic with a vector will give you a **logical array** which tells you if each value of the vector is true or false with that condition
- for example, which cells showed growth under the treatment?



```
>> growth = randn(1,100);
>> growth>0

ans =

  1×100 logical array

  Columns 1 through 35

   1   0   1   1   1   0   1   1   1   1   1   0   0   1   0   0   0   0   1   1   0   0   1   0   1   1   0   0   0   1   0   0   0   0   0

  Columns 36 through 70

   0   1   1   0   1   0   0   1   1   0   1   0   1   1   1   1   1   0   0   1   0   1   1   1   0   0   0   1   0   1   1   0   0   0   0

  Columns 71 through 100

   0   1   0   0   1   0   0   1   0   1   1   1   0   1   0   1   1   1   0   1   0   0   0   0   0   0   0   1   0   1
```

combining logic and vectors

- even more powerful, we can now use that **logical array** to **select** values from our vector. so just like we used numbers to select elements, we can instead use these 0/1 values to say exclude/include in a new array



```
>> growth(growth>0)

ans =

  Columns 1 through 14

    0.0799    0.4115    0.6770    0.8577    0.4494    0.1006    0.8261    0.5362    0.8979    1.0078    0.6487    0.8257    0.1370    0.3018

  Columns 15 through 28

    0.3999    1.1454    2.1778    1.1385    0.4413    0.1644    0.7477    1.5763    0.3275    0.6647    0.0852    0.8810    0.3232    1.8586

  Columns 29 through 42

    0.1034    0.5632    0.1136    1.4790    0.7847    0.3086    0.1922    0.3362    0.3501    1.0360    2.4245    0.9594    0.4286    1.8779

  Columns 43 through 47

    0.9407    0.7873    0.3199    0.6076    0.6992
```

combining logic and vectors

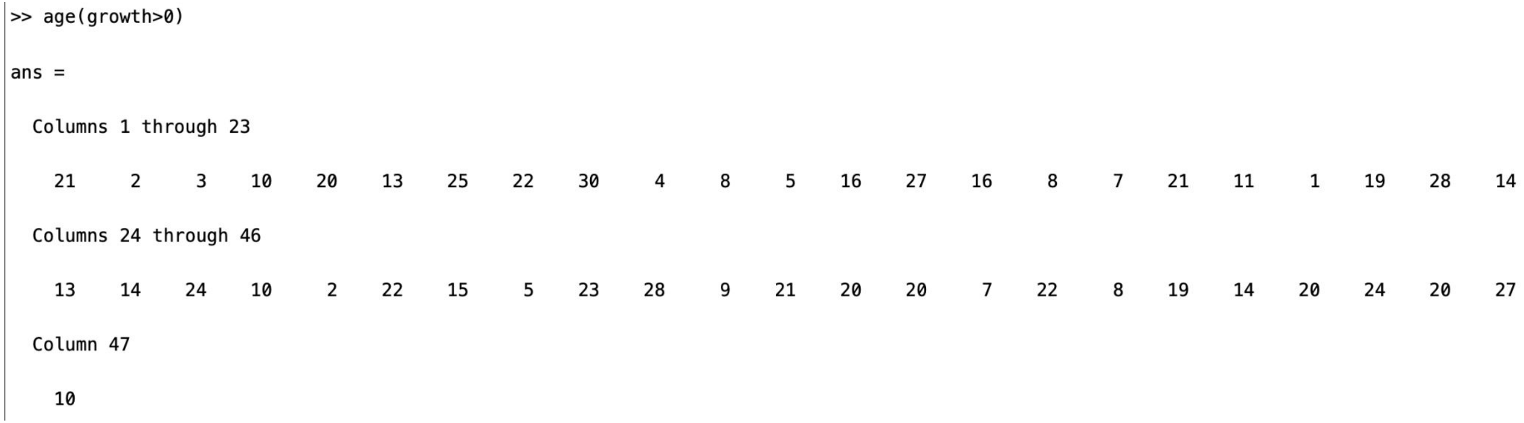
- but you can also use this logical array independently from the original vector it is made from. for example, how might we find the **age** of the cells that had positive growth?

combining logic and vectors

- but you can also use this logical array independently from the original vector it is made from. for example, how might we find the **age** of the cells that had positive growth?

positive_growth = growth > 0;

age(positive_growth) % can also just do age(growth>0)

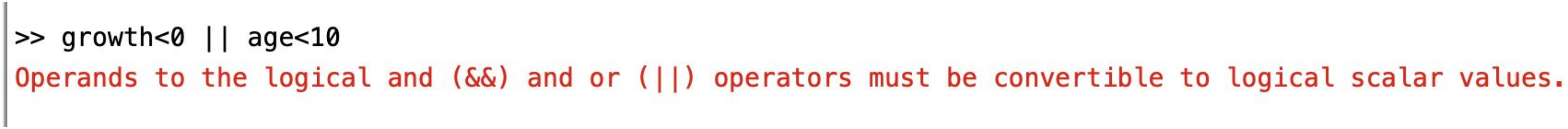


```
>> age(growth>0)

ans =

  Columns 1 through 23

    21     2     3    10    20    13    25    22    30     4     8     5    16    27    16     8     7    21    11     1    19    28    14

  Columns 24 through 46

    13    14    24    10     2    22    15     5    23    28     9    21    20    20     7    22     8    19    14    20    24    20    27

  Column 47

    10
```

logic with vectors – how to use OR/AND

let’s say we want to find **how many cells** had negative growth OR are younger than 10 days. we need to combine each pair of elements from the vectors in order and do logic on all of them independently – just like when we did the . (dot) notation for some of our math!

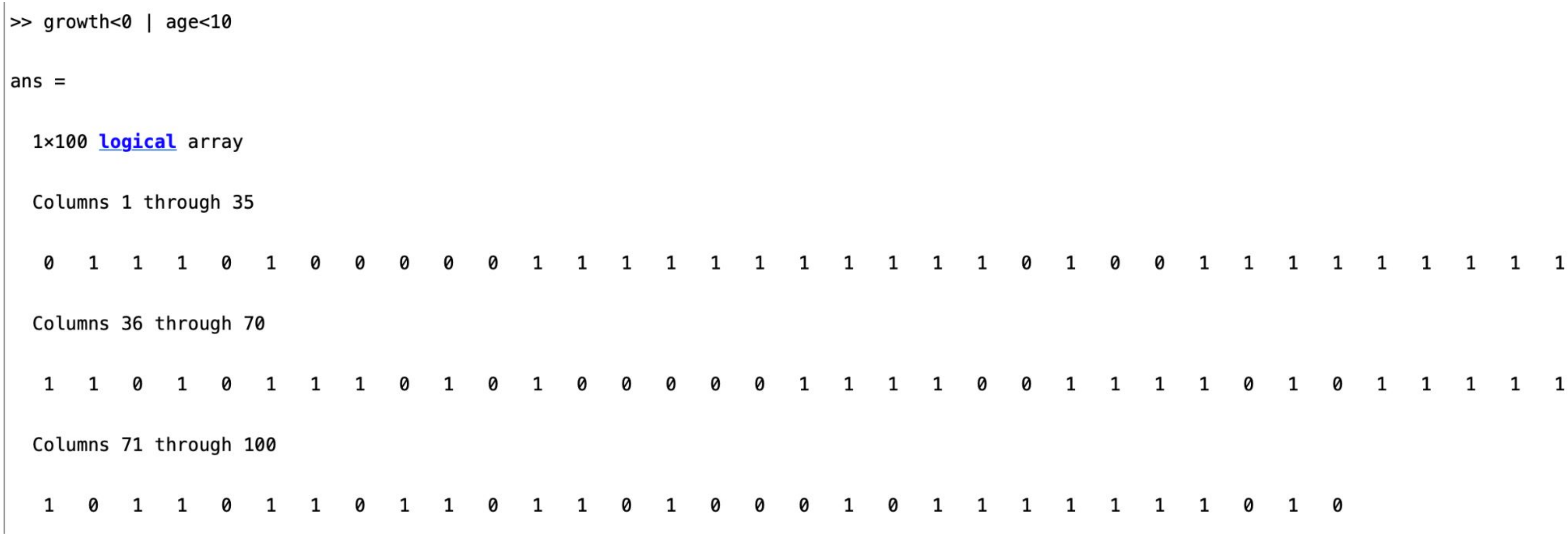
so far we’ve used || and && to do logical operations, which takes the logical true/false value of everything on one side and everything on the other side. but that’s not what we need here!



```
>> growth<0 || age<10
Operands to the logical and (&&) and or (||) operators must be convertible to logical scalar values.
```

logic with vectors – how to use OR/AND

now we want to do **element wise** logic – to do so, we use single | or & commands



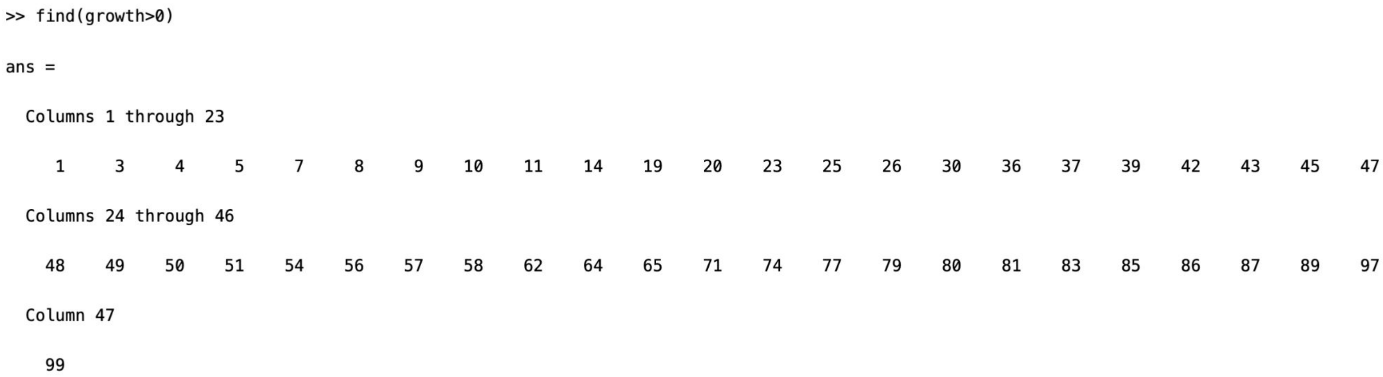
```
>> growth<0 | age<10

ans =

  1×100 logical array

  Columns 1 through 35

   0   1   1   1   0   1   0   0   0   0   0   1   1   1   1   1   1   1   1   1   1   1   0   1   0   0   1   1   1   1   1   1   1   1   1

  Columns 36 through 70

   1   1   0   1   0   1   1   1   0   1   0   1   0   0   0   0   0   1   1   1   1   0   0   1   1   1   1   0   1   0   1   1   1   1   1

  Columns 71 through 100

   1   0   1   1   0   1   1   0   1   1   0   1   1   0   1   0   0   0   1   0   1   1   1   1   1   1   1   0   1   0
```

now you: if we now want to know **how many** of the cells fit in this category, what should we do? (there are multiple answers!)

find command

even though these logical arrays are useful for math operations, they aren’t helpful for humans – just by looking at one, can you report the numbers of the cells which fall into this category?

to help with this, we use the **find** command which returns the **indices** of the positive values in an array – so, if it is a logical 0/1 array, it will return the indices of all the 1s!

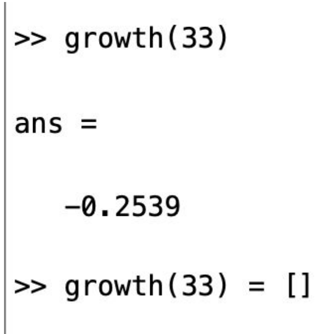


```
>> find(growth>0)

ans =

  Columns 1 through 23

     1     3     4     5     7     8     9    10    11    14    19    20    23    25    26    30    36    37    39    42    43    45    47

  Columns 24 through 46

    48    49    50    51    54    56    57    58    62    64    65    71    74    77    79    80    81    83    85    86    87    89    97

  Column 47

    99
```

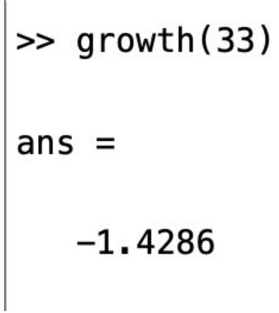
removing elements from vectors

oh no! your boss says that cell 33 was accidentally included in this study but should have been excluded. how can we remove that data?

we can set the index to empty brackets to delete the data

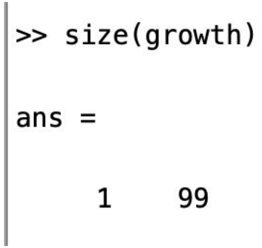


```
>> growth(33)

ans =

  -0.2539

>> growth(33) = []
```

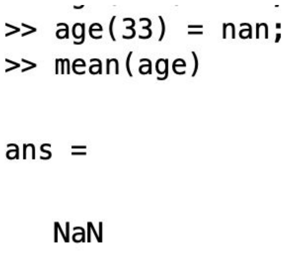


```
>> growth(33)

ans =

  -1.4286
```

note – this will not hold an empty place for it! we could do that using nan or “not a number” if we want to remember that the cell was there. this can come in useful sometimes but will throw errors if you try to do things like mean() without checking.



```
>> size(growth)

ans =

     1    99
```



```
>> age(33) = nan;
>> mean(age)

ans =

   NaN
```

updating vectors

we can combine the logic in the previous slides to update our data. maybe we know that any measurement of growth which is within .05 above or below 0 is actually a measurement error and all of those cells should have their growth set to 0. how could we combine find and/or logic commands to update these cells?

updating vectors

we can combine the logic in the previous slides to update our data. maybe we know that any measurement of growth which is within .05 above or below 0 is actually a measurement error and all of those cells should have their growth set to 0. how could we combine find and/or logic commands to update these cells?

```
growth(find(growth<.05 & growth>-.05)) = 0
growth(growth<.05 & growth>-.05) = 0 % also works
```

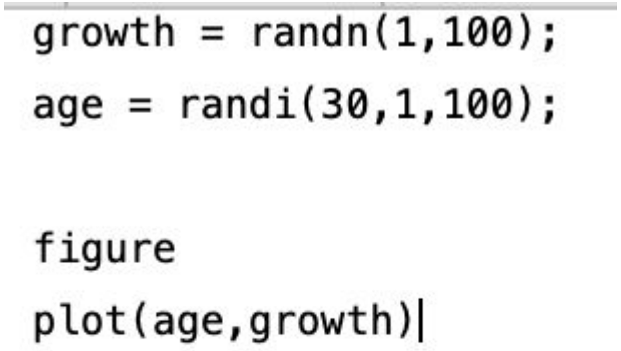
problem time

how would you do the following? you have two 1x99 arrays, **growth** and **age**.

- find how many cells had growth above .4 and an age below 15 days
- make a new array which holds only the ages of the cells which had growth which was larger than the mean growth
- remove all the data from growth which is smaller than the mean
- add a new cell to the arrays with .45 growth measurement and 7 days old
- create a new array which stores the value of growth times age for each cell
- challenge: choose a **random** element of the age vector, and then return the value of this index of the growth vector (must use a rand function!)

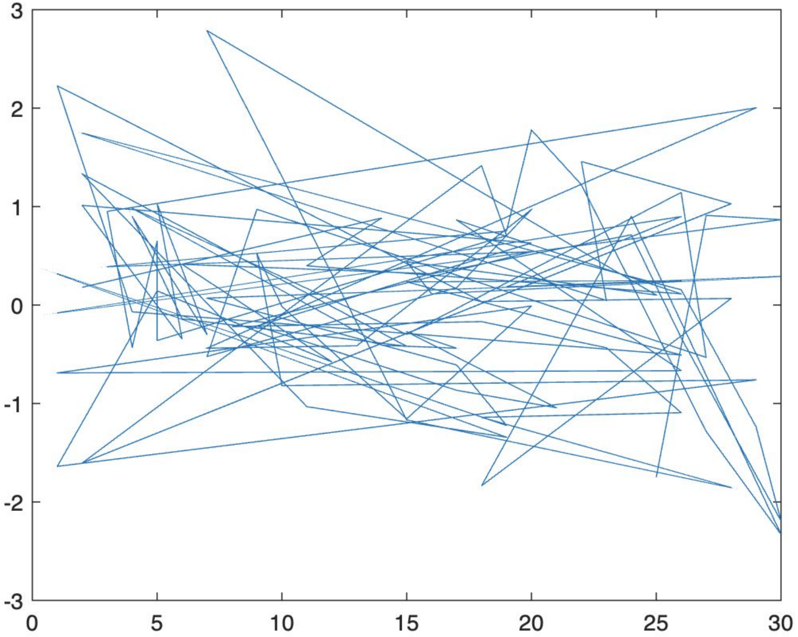
plotting basics

I want to know if there is a correlation between growth and the age of cells. how might I visualize this in matlab? (assume original 1x100 vectors growth and age)



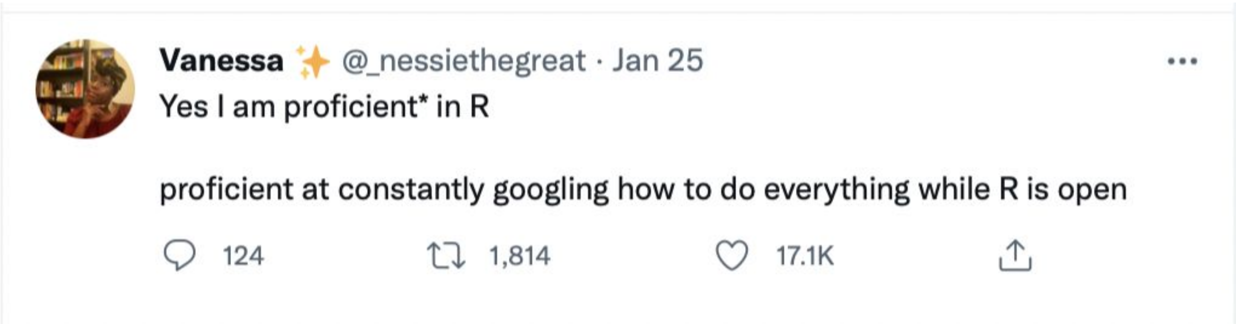
```
growth = randn(1,100);
age = randi(30,1,100);

figure
plot(age,growth)
```

will this work?



not quite! we need to use **options** to be more specific about the plot we want.

strong suggestion!



Vanessa ✨ @_nessiethegreat · Jan 25

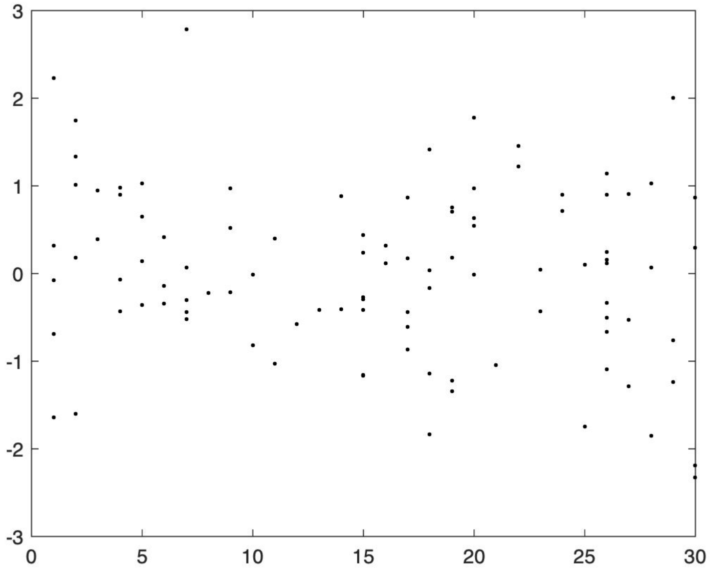
Yes I am proficient* in R

proficient at constantly googling how to do everything while R is open

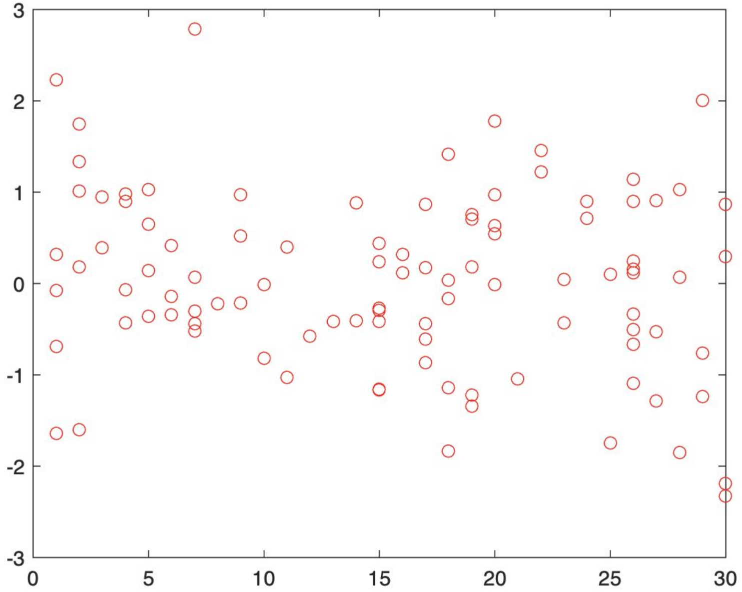
read the matlab docs for plotting early and often!

https://www.mathworks.com/help/matlab/ref/plot.html

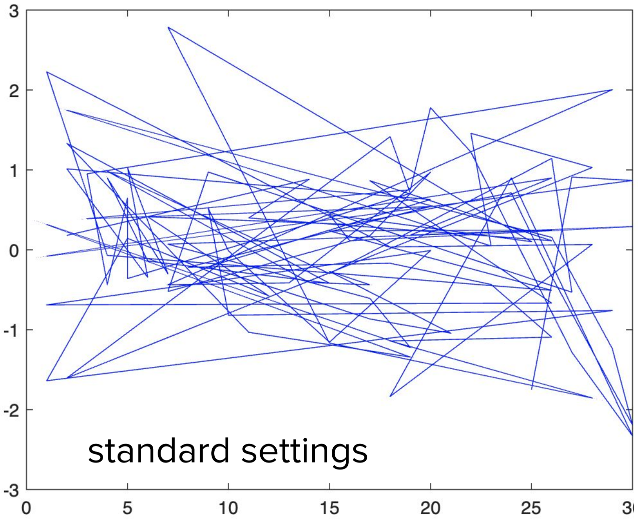
I will open this (or other pages, like for histograms etc) many times while I am coding plots. do not feel like you need to memorize everything, but as you write more and more some of it will come easier (like how to change colors). don’t just rely on the problems we do in class and on canvas – there will be things you want to do but we don’t show (like how to plot in purple) plot(age,growth, ‘k.’)



plot(age,growth, ‘ro’)

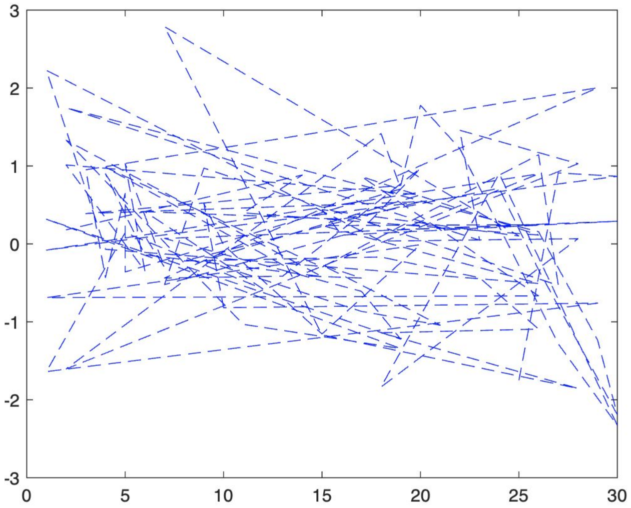


plot(age,growth, ‘b-’)



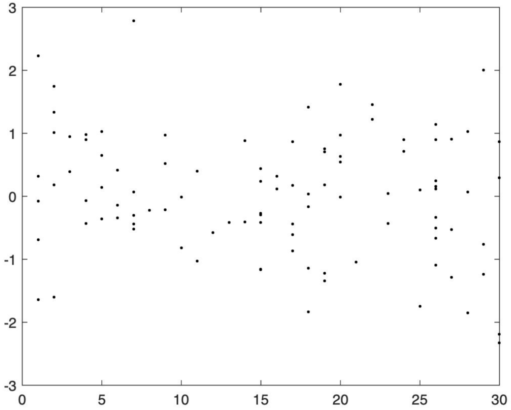
standard settings

plot(age,growth, ‘b--’)



plot(age,growth, ‘r-’)

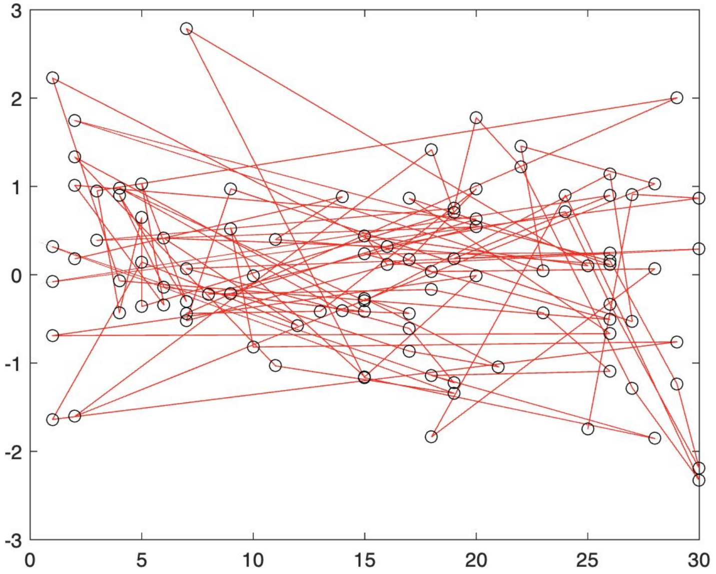
plot(age,growth,’k.’)



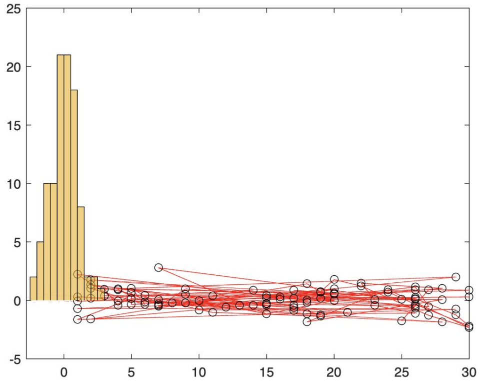
why can’t it plot both?

matlab command: **hold on**
if you want to wait and
keep plotting on top

```
plot(age,growth,'r-')
hold on
plot(age,growth,'ko')
```



make sure to hold off! otherwise you'll keep plotting on the same one...

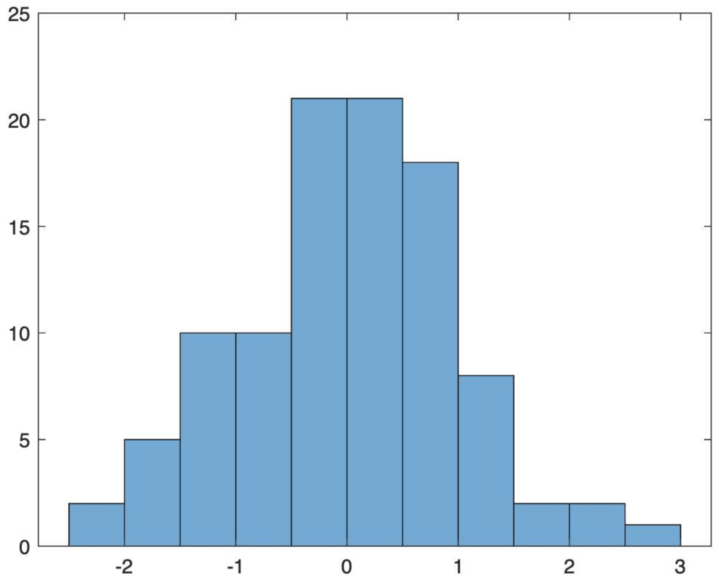


other plots

these line graphs don't tell us much (except that it seems age and growth aren't related – why is that?)

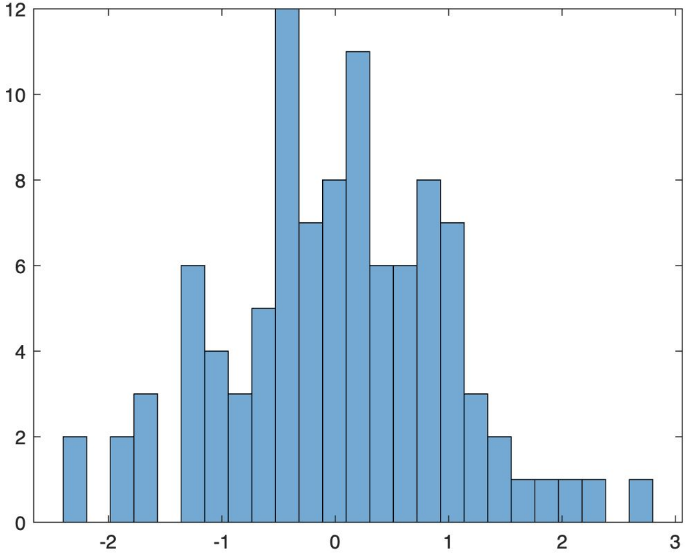
let's look just at the growth variable - maybe we want to see its whole distribution in a histogram

histogram(growth)

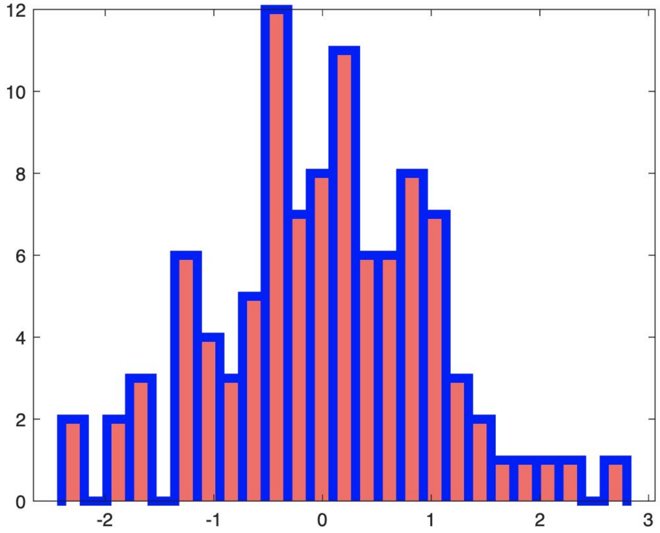


we have other options for histograms too! but for more complex plots we need to pass specific variable names to plotting functions to make changes. you can learn what these are by reading the docs! these exist for plot too

histogram(growth,25)



histogram(growth,25,'FaceColor','red', 'EdgeColor','blue','LineWidth',5)

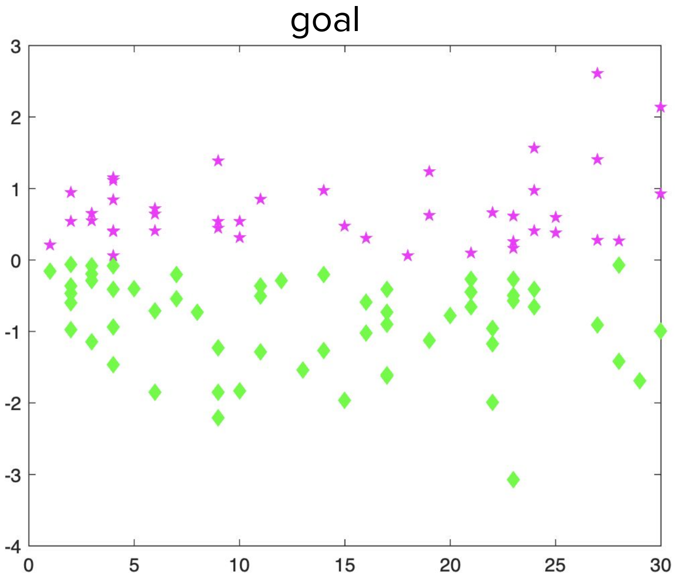


now you!

try to make a scatter plot using the commands we’ve worked on today where the markers for **positive growth** are **magenta stars** and the markers for **negative growth** are **green diamonds.** all markers should be filled. look at documentation!

reminder:

```
age = randi(30,1,100);

growth = randn(1,100);
```



goal