RegressionChannelSimple_GUIDE

Slide 1
Title: A trading program in Python

Here we present how a typical algorithmic stock trading workflow is put together.
This workflow shares many logical steps with the machine learning workflow.
and we shall be using it throuout the course.

Slide 2

Title: Trading a series that reverts to the mean

A price series is said to be stationary
if the statistical properties such as mean, variance, etc. are constant over time.

With a stationary price series, you can profit easily from a mean-reverting strategy.
You can buy when the price is lower than the mean price and
exit when the price goes back to mean.
Similarly, you can sell when the price is higher than the mean price and
exit when the price goes back to the mean.

Important terminology to remember:
to long a stock is to buy a stock
when you have bought a stock you are in a long position re. that stock
to short a stock is to borrow a stock and sell it
surprised?
yes, you can sell a stock that you do not own = a stock that you have borrowed (rented for a fee) from the exchange
exchanges will allow you to do that, as long as you return the stock as promised
and just like above, when you have sold a stock you are in a short position re. that stock
to neutralize a position is to reverse a transaction:
to buy if you sold before, to sell if you bought before

Here we see the execution of a trading strategy applied to a stock
that moves in a mean-reverting way
the mean price of the stock is zero in this case
note that you short when the stock is overbought (expensive)
and reverse the transaction, that is, buy the stock (making a profit) when the stock reverts to its normal state (where price=zero, the mean)
again you long the stock when the stock is oversold (cheap)
and reverse the transaction, that is, sell the stock (making a profit) when the stock reverts to its normal state (where price=zero, the mean)

slide 3

Title: Lognormal vs Normal distributions

You have seen that:
Stock prices are lognormally distributed
and the log of stock prices is normally distributed.
The normal and lognormal distributions are shown.
The lognormal distribution has that shape because it only applies to random variables that assume values zero or above.
whereas the normal distribution applies to random variables that assume positive or negative values or zero.
Since prices cannot be negative, it makes sense that the lognormal distribution applies to them.

A lognormal variable is by definition one that becomes normal when you apply the log transformation to it.
From now on we are going to talk about the log of prices (transformed prices) though for the sake of simplicity
we are going to refer to these transformed prices simply as prices.

slide 4

Title: The Empirical Rule and Chebyshev's Theorem for bell-shaped distributions

The fact that asset prices can be assumed to be normally distributed
(after transformation as just discussed)
is useful because one can use Chebyshev's Theorem to make a hypothesis about
prices that are outliers (very cheap or very expensive) and prices that are not (average)
Chebyshev's theorem basically says that outliers are found only in the left or right tails of the normal distribution
An extreme outlier is one that occurs only 5% of the time and is located 2 standard deviations away from the mean.

Chebyshev's Theorem:
Approximately 68% of the data lie within 1 standard deviation of the mean
Approximately 95% of the data lie within 2 standard deviations of the mean

slide 5

Title: Example of a trading rule for the mean-reverting series

So if we have a time-series that we want to trade,
assuming the histogram of the series is bell-shaped
applying Chebyshev's theorem we can state a trading rule:

assume MA(series) is a moving average of the series (an average calculated over a rolling window of data)
assume std(seres) is a moving standard deviation of the series (a standard deviation calculated over a rolling window of data)

Long the stock if:
series_t == MA(series) - 2* std(series)
i.e. the stock is oversold (cheap)

Short the stock if:
series_t == MA(series) + 2* std(series)
i.e. the stock is overbought (expensive)

Neutralize the position if:
series_t == MA(series)
i.e. the stock is priced at normal levels

If you use this strategy you will encounter
oversold and overbought stocks only 5% of the time because of Chebyshev's Theorem.
Oversold and overstock stocks are going to located far out into the tails of the normal distribution.

slide 6

Title: Example of a trading rule for the mean-reverting series

note that:
MA(series) is called a mean band

MA(series) - 2* std(series)
is called a band under the mean band or the lower bound

MA(series) + 2* std(series)

is called a band above the mean band or the upper bound

the three bands together are called Bollinger bands

slide 7

Title: The Empirical Rule and Chebyshev's Theorem for any Shape of Distribution

Bear in mind that the trading strategy we just sketched is a very simple version
of what could be a rather sophisticated trading strategy.
For example, the distribution of prices is not uni-modal all of the time.
It can become bimodal when market participants disagree strongly about a price.
An example of what a bimodal distribution looks like appears to the left of the slide.
Bimodal distributions fall under a more stringent Chebyshev Theorem that
applies to distributions of any shape.
This more stringent rule says that approximately 75% of the data lie
within 2 standard deviations of the mean.
One way to know if the price distribution is bimodal is
to look for the presence of two means in the price data.
The code to the right of the slide looks for the presence of two means by
applying the Wilcoxon-Mann-Whitney (non-parametric) “W-test” for two independent samples.
But having pointed you to the existence of such adaptations,
we are going to keep the rest of our discussion to the model we presented in the beginning.
The point of this presentation is to guide you in an exercise that
involves understanding basic Pandas, and certain basic financial metrics.

slide 8

Title: Parts of a back-testing trading program

All back-testing trading programs have a basic structure that looks like the one in this slide.

There are 7 steps that can usually be identified:
1. Data downloading and pre-processing- this is usually done by using a data provider that has an API.
2. Signal engineering - often price signals are filtered before being used.
3. Trading rule enunciation - this is a conditional statement that controls when to go long and when to go short.
4. Stance calculation - this calculates how many contracts we are long and short at the present time. The simplest stance is +1 (long) and -1 (short).
5. System return calculaton - this calculates the returns of applying the stance to the stock returns
6. Equity curve calculation - this calculates the accumulated returns of the system.
7. First Level Evaluation Metrics - this calculates the well known financial metrics that we will describe further on,
like Annualized Return, CAGR, SHARPE, DrawDown, etc..
8. Second Level Evaluation Metric - this calculates a statistical measure of the significance of the mean returns of the system.
An example of such a measure is White’s Reality Check.

As indicated on the slide, machine or deep learning can contribute to the construction of such a system,
especially in steps 2 and 3.
But in this presentation, we are just going to describe a basic algorithm that does not require machine learning.

slide 9

Title: dfP_RegressionChannelSimple.xlsx

The trading system we are presenting will be in two formats: an Excel format and a Python format.
A Python script runs from the top down.
An Excel program runs left to right.

We are not going to comment on the Excel program in the presentation, but
we encourage you to look at it, because
it has many explanatory comments in the column headings.
Next, we will comment on the Python script.

slide 10

Title: RegressionChannelSimple.py

This Python script does exactly what the Excel spreadsheet is doing in the same order.
We suggest you run it in Spyder (part of Anaconda Python) because
Spyder has a variable explorer window that
keeps track of the content of all the variables, and this really is helpful.
Next, we will comment on the code of the Python script.

slide 11

Title: Importing the modules

This is the code that imports the modules, the most important being
Pandas, bumpy, and yfinance.

As regards yfinance, since we wrote this program yfinance has evolved a lot.
Our code reflects an earlier stage of yfinance, where we were taking care of some yfinance bugs,
and contains some lines that may no longer be necessary because yfinance is less buggy.
yfinance downloads free stock price data from Yahoo Finance,
in the format open, hi, low, close, adjusted close, and volume.

The adjusted prices cause endless confusion.
The difference between close prices and adjusted close prices is that
the adjusted prices reflect dividends, splits, and other inflows that are adjusted every 24 hours.
When the adjustment takes place, it affects not just today's adjusted price, but
the entire adjusted price series, which gets modified.
So it makes sense to use close prices when calculating signals that
tell us when to buy or sell a security,
especially if your signals depend on a short lookback window that
is not affected by monthly dividends, for example.
But also it makes sense to use adjusted prices when calculating profits,
if the holding period of the strategy is long enough to be affected by dividends.
So there is some leeway here, depending on the length of the lookback and holding windows.
However, we tend not to complicate matters in this course.
For the sake of simplicity we usually use only one of the series (the adjusted or the close),
except in rare cases.
In this particular case, we used the adjusted close for everything.

slide 12

Title: Declaring variables

This variable declaration introduces the parameters.
These parameters can be adjusted.
In the Excel version of the program, these parameters are located in the worksheet called "Parameters."
The entryZscore defines where we are going to go long or short.
the exitZscore defines where we are going to exit the position by reversing what we did previously.
We are setting the entryZscore not at 2 (for 2 standard deviations away from the mean) but at 0.5,

meaning, we are going to take long and short positions rather more frequently.
The window defines what the lookback is for the calculation of the mean and the standard deviation of the price.
It is better to use short windows in this strategy, so a 3-week window is sufficient
(remember that a trading week is just 5 days).
The variable shorts defines if we are going to ever take short positions, and we are setting it to zero, so
only long positions will be possible.
The delay controls whether we are going to wait until the next day to execute a signal,
we are setting this to 1, so no waiting will take place.
tcost sets a transaction cost, here we set it to zero, so no transactions costs are considered.

slide 13

Title: Downloading and saving the data

Here we make sure that if we get an error downloading data from yahoo finance, we can always
load the data from a previously stored instance of the data.
yfinance is less buggy today, but you may still need this.

slide 14

Title: Reading and storing the data into a well-formed data frame

Here we read the price data and store it in a data frame using the date field as the data frame index.

slide 14

Title: Plotting the data

It is always a good idea to plot the close prices first,
here we show you how to do this.

slide 16

Title: Data pre-processing

We included this slide to make you aware of something that can be confusing in Pandas.
Pandas is very similar to an Excel spreadsheet, with a date index, column titles, row numbers, etc..
But unlike Excel, Pandas usually allows you to do the same thing in many ways.
You can refer to a column in Pandas by using the dot notation or the bracket notation.
But if you try to do an assignment to a column, you cannot use the dot notation,
you must use the bracket notation or an explicit assign.

Note that we take the log of the price series, so as to normalize them,
since we are going to use Chebyshev's Theorem to trade.

slide 17

Title: Signal Engineering

In this slide, we are calculating the three Bollinger bands,
the mean band, called mean,
the upper band, called U-B, and
the lower band, called L-B.

slide 18

Title: Plotting the Signal

In this slide the code is plotting the zScore, with time on the x-axis, and
the number of standard deviations on the y-axis.
When setting the entryZscore and the exitZscore you need to refer to this graph.
The entryZscore needs to be reachable by the zScore.
If you set the entryZscore at 5 standard deviations, the system will never buy or sell.

slide 19

Title: Plotting the Signal

This is the code for plotting the Bollinger bands generated by the system.

slide 20

Title: Plotting the signal

This is the code for plotting the bollinger bands generated by the system in terms of the zScore.
The two plots (this one and the previous)
essentially contain the same information and generate the same trading signals.

slide 21

Title: Trading Rule for Going Long

This slide contains the conditional statement that defines when to buy a contract.

slide 22

Title: Calculating long contracts

This slide shows the code that keeps track of how many contracts we have bought and are presently holding.

slide 23 to slide 24

The same goes for the shorts, but we set the parameter for shorts to zero.

slide 25

Title: Calculating the total num_units or STANCE and the system returns

Here the system accounts for the total of the net number of contracts it holds.
This is called the systems' stance, calculated in 1.

The system returns are calculated in 2 on this slide.
This involves multiplication of the system's stance (+1 for long or -1 for short)
by the percent returns calculated from the adjusted close price of the stock.

Note that when calculating the system returns
we use percent returns (not log returns) of the adjusted close price.
This choice is due to convenience:
we will be calculating the Sharpe ratio metric to evaluate the system, and
the Sharpe ratio is based on percent returns.

Nonetheless, we still use the log of the adjusted close price

for the calculation of the trading signal in slide 2.
We do this because log prices have a normal distribution and
we are using a trading rule based on Chebyshev's Theorem in slide 4.

slide 26

Title: Calculating cumulative returns (2 ways)

Depending on how you calculate returns,
as percentages or as log returns,
you calculate cumulative returns differently.
The slide shows the two ways of doing it.
Because we are using percent returns to calculate the system returns
we will be using the cumprod function.

Note:
If you want the cumulative (log or percent) returns to start at 1
to reflect a starting investment of 1 dollar,
You can optionally add a 1 to both formulas on the slide.

slide 27

Title: Calculating the system cumulative returns

So this slide shows the calculation of the system cumulative returns
based on percentage returns, and forcing the series to start at 1
by adding the extra 1.

slide 28

Title: RESULTS: Plotting and saving the system equity curve (vs market buy and hold)

This is the code for plotting the equity curve of the market buy and hold strategy
compared to the equity curve of the system's trading strategy.

slide 29

Title: Evaluation metrics calculation

Next, you need to calculate some financial metrics to measure the performance of the system.
Some preliminary calculations are necessary.
We need to know the number of calendar days during which the system has been in operation.
This code calculates that number.

slide 30

Title: Annual Return

The annual return is an intuitive metric we can all understand.
The annual return is the non-compounded rate of return that
would be required for an investment to grow from its beginning balance to its ending balance,
linearly per year.
The formula for its calculation is given on the slide.

slide 31

Title: Evaluation metrics: Total Annual Return

This is the code for calculating the annual return.
We remark that you have the choice to divide by the number of calendar days in a year,
or the number of trading days in a year.

slide 32

Title: CAGR

Compound annual growth rate (CAGR) is the rate of return that
would be required for an investment to grow from its beginning balance to its ending balance,
assuming the profits were reinvested at the end of each year of the investment's lifespan.
The formula for its calculation is given on the slide.
This is the metric that is usually used to measure the return of investment strategies,
though it is less intuitive than the total annual return.

slide 33

Title: Evaluation metrics: CAGR

This is the python code for the CAGR calculation.

slide 34

Title: Sharpe ratio

The Sharpe ratio was developed by Nobel laureate William F. Sharpe, and
is used to help investors understand the return of an investment compared to its risk.
The ratio is the average return earned in excess of the risk-free rate per unit of volatility or total risk.

Subtracting the risk-free rate from the mean return
allows an investor to better isolate the profits associated with risk-taking activities.
Generally, the greater the value of the Sharpe ratio, the more attractive the risk-adjusted return.

In this course, we tend to use the simplified version of the Sharpe ratio given in the slide
in terms of the mean of the strategy returns divided by the standard deviation of the strategy returns
corrected by multiplying this ratio by an annualization factor.
The annualization factor is the square root of the period, where
the period is the number of days in a year (assuming the returns are daily returns)
If the returns are monthly returns, the period needs to be 12.

slide 35

Title: Evaluation metrics: Sharpe ratio

This is the python code for the calculation of the Sharpe ratio.
The code assumes that sys_rets are percent returns not log returns,
and that the risk free-rate is zero, as in fact it essentially is at the moment.

slide 36

Title: Maximum Draw Down

A maximum drawdown (MDD) is the maximum observed loss from a peak to a trough of a portfolio,
before a new peak is attained.
Maximum drawdown is an indicator of downside risk over a specified time period.
Our code sometimes refers to the maximum drawdown in some examples,
but we shall not discuss it further in this course.

slide 37

Title: Maximum Draw Down Program

This slide shows the program for calculating the maximum drawdown.
It is given only for reference, we shall not discuss it further in this course.

slide 38

Title: Calmar Ratio

The Calmar ratio is a comparison of the average annual compounded rate of return and
the maximum drawdown risk of commodity trading advisors and hedge funds.
The lower the Calmar ratio,
the worse the investment performed on a risk-adjusted basis over the specified time period;
the higher the Calmar ratio, the better it performed.
It is given only for reference, we shall not discuss it further in this course.

slide 39

Title: Information Coefficient

The Information Coefficient is usually the Pearson correlation
between the return on a common stock predicted by a valuation model or analyst and the actual return.

However, the Pearson correlation a.k.a. Pearson's rho has some limitations:
it works only with continuous variables,
it only accounts for a linear relationship between variables, and
it is sensitive to outliers.
So in finance, many researchers convert predicted and actual values to ranks before correlating.
This means their correlation is the Spearman's rank correlation coefficient --Spearman's rho.
In this course, the Information Coefficient will be defined as Spearman's rho.

Spearman's rho is similar to Pearson's rho (between -1 and 1), but
exchanges the Pearson's rho linear relationship for a monotonic relationship, which is more flexible.
A monotonic relationship is a relationship that does one of the following:
(1) as the value of one variable increases, so does the value of the other variable, OR,
(2) as the value of one variable increases, the other variable value decreases.
BUT, not exactly at a constant rate whereas in a linear relationship the rate of increase/decrease is constant.
Spearman’s rho slightly understates the strength of the Pearson rho relationship, but
its sampling error is about the same.

The advantages of Spearman's rho are that
it is easy to interpret as it ranges between -1 and 1, and
it can be used to evaluate linear and non-linear models,
unlike R-squared, which only works for linear models.
Another advantage is that it is possible to calculate the satistical significance of Spearman's rho
by calculating its p-value.

The disadvantage is that
any Information Coefficient based on fewer than several thousand samples
will suffer from significant sampling error.
This is an important limitation because
older financial data tends to be less relevant.
See: ICAndCorrelation.pdf on the sampling error (this reading is optional).

slide 40

Title: Phik

The information coefficient is very popular in finance but
Spearman's rho only works for continuous variables.
For categorical variables, Cramer's V is traditionally used.
Cramer's V is based on the Pearson Chi-squared test
shown to the right of the slide, but
Cramer's V is dependent on the binning chosen per variable,
which makes it difficult to interpret, and
it is also sensitive to outliers.

Phik is a new correlation coefficient that
corrects these defects.

Phik works consistently between:
categorical, ordinal, interval (that is, continuous) and even mixed variables.
Though Phik captures non-linear dependency,
it reverts to the Pearson correlation coefficient
in case of a bivariate normal input distribution.
Moreover,
the Phik algorithm contains a built-in noise reduction technique against statistical fluctuations.

So, the advantages of Phik are that:
it is easy to interpret as it ranges between 0 and 1, and
it can be used to evaluate linear and non-linear models, and
it works for all kinds of variables.

The disadvantages are:
Phik has no indication of direction, and
when working with numeric-only variables,
other correlation coefficients will be more precise,
especially for small samples.

For more information,
see phik.pdf and the link on the slide.

slide 41

Title: White's Reality Check

This is the code for calculating White's Reality Check.
We will discuss White Reality Check later on in the course.
White Reality Check calculates a p-value that
tells us if the mean returns of the system are due to chance.
The p-value needs to be low, preferable less than .10; otherwise,

one could interpret that the good average returns of a system occurred by chance.

slide 42

Title: Returns.xlsx

This slide shows all relevant formulas with both types of returns,
percent and log returns, that
you can find in the spreadsheet Returns.xlsx.

Cumulative log returns are flatter (less steep) than
cumulative percent returns, an effect of the log transformation.
Log returns and log prices are used for trading signal calculation
because of the associated normality of the distribution.
Percent returns are used when "adding" the returns of various stocks together
to calculate the return of a portfolio of those stocks.
Percent returns are also used when calculating various performance metrics,
like the Sharpe ratio.

In the worksheet called Formulas,
you will find the percent returns (big cap R) and log returns (small cap r) in two columns,
This worksheet carefully calculates the log and percent return version of:
cumulative returns,
average returns,
annualized returns,
variance of the returns,
annualized variance of the returns,
standard deviation of the returns,
annualized standard deviation of the returns,
the Sharpe ratio,
CAGR or Compound annual growth rate, and
the conversion between percent and log returns.

The workpages large_r and small_r
show how to calculate examples of these measures using Excel.

Commenting on the "Formulas" worksheet from the top down,
some conventions apply:
Small-cap-r are log returns by convention.
Small-cap-r is taken to be a vector of returns.
The r-sub-i is a single small-cap-r return.

Big-cap-r are percent returns, by convention.

Small cap n is the number of samples (i.e. the number of returns).

The main difference between percent returns and log returns is:

Percent returns are regularly transformed into "return factors"
by the addition of a 1,
then those "return factors" are processed in various ways
(e.g. by exponentiation or multiplication).
Finally the "return factors" are transformed back into returns
by the final subtraction of the 1 that had been added.

Cumulative percent returns involve multiplication, and
the average of percent returns involves geometric averaging (of the return factors).

By contrast,
log do not require turning into "return factors."
Cumulative log returns involve addition, and
the average of log returns involves arithmetic averaging.

In this worksheet,
the "period" is actually the sampling frequency of the returns
(our "period" is how many returns fit into a year).
For daily returns, the "period" is 252 since
there are 252 trading days in a year, though 360 is acceptable.
If you are dealing with weekly returns, the "period" is 52,
if you are dealing with monthly returns, the "period" is 12.
So when you divide the number of data items by the "period,"
you get the number of years, which
is essential for annualization.

Slide 43

Title: Conclusions

So what can we conclude from this presentation:

First, as regards the type of returns used:
A typical algorithmic trading workflow
involves the calculation of a trading signal and a trading rule.
This trading signal or rule will often
take advantage of the normality of the distribution of log returns or log prices.
Machine learning can be applied
to the calculation of the trading signal and trading rule.

Financial performance metrics used to evaluate a typical algorithmic trading system
often involve the calculation of percent returns.
This is especially true if the Sharpe ratio is used, which
is probably the most important financial performance metric.

Log returns and percent returns require different formulas.

Slide 44

Title: Conclusions

Second, as regards the types of metrics used in finance that
we have described in this presentation and have listed on this slide:
Unfortunately,
these metrics are NOT part of the "out of the box" performance metrics in Scikit-Learn or Keras.
This is why,
we will be teaching you ways in which you can customize
the Scikit-learn and Keras workflow to include them.
Let us, therefore, start you on learning about the traditional trading system workflow.

Read dfP_RegressionChannelSimple.xlsx, and
compare it with RegressionChannelSimple.py which was covered in this presentation.

Read dfP_moving_average_crossover_simple.xlsx, and

compare it with moving_average_crossover_simple_incomplete.py
You will notice that we wrote some comments with instructions for you to write code.
Please write the missing lines.
You will not get exactly the same result by the Python program
as the Excel spreadsheet but it should be close
(there is a difference in Excel and Python date processing functions that
results in a difference in the number of years that affects the results).

To be able to program the few lines of code required by the homework,
we suggest you read especially:
Miscellaneous\CheatSheets\NumpyPandasMatplotlib.pdf
Miscellaneous\CheatSheets\CheatSheetPandasDataFrameMarkGraph.pdf

As guidance:
Use the Spyder variable explorer.