Diversification in the Managed Futures Universe

Derek G. Nokes, CUNY

Friday, May 22, 2015

Contents

“‘

Abstract

In this paper we focus on the application of a statistical factor model to a subset of the universe of managed futures investment programs. Our objective is to model the relationships between the returns of a select set of these investment programs and to produce a simple sensitivity providing a map of the return variability of a portfolio to changes in the diversity of the portfolio components as represented by the importance of a few factors. The paper is composed of main five sections where each section outlines one step in the data science workflow.

– Introduction

– Data

– Modeling

  - Theory
  - Application

– Conclusions

Introduction and Motivation

In portfolio allocation applications the objective is to maximize investors' future wealth by determining how to allocate capital among a set of available investments in such a way as to maximize *compound* growth subject to a set of constraints. Maximizing wealth requires that we take advantage of the powerful positive effects of compounding. When we reinvest, the magnitude of investment returns and the variability of those returns make *equal* contributions to compounded total return. Reducing the variability of returns thus has as much impact on total return as the magnitude of returns. The variability of portfolio return is a function of the co-variability of investment component returns. If the component returns tend to move together, the magnitude of fluctuations in the monthly value of the portfolio is higher than if the components move in different directions. A portfolio comprised of components with diversified returns will achieve higher compound growth than a portfolio with less diversified components holding average component returns constant. In the simplest terms, portfolio allocation is primarily about selecting sets of investments with *future* positive average returns and low co-variability.

As the size of a portfolio increases, the number of inter-relationships between components explodes. It becomes increasingly difficult to understand the drivers of portfolio return as the number of components rises because the number of independent parameters in a covariance matrix grows with the square of the number of investments. Grouping investments that tend to move together and focusing on trying to find groups that are independent is one common way to reduce the dimension of the portfolio allocation problem. This can be accomplished through the use of statistical *factor models*.

Portfolio Return & Its Variability

In this section, we introduce definitions for portfolio return and variability that will be used throughout the paper.

Portfolio Return

Portfolio return is a function of the weights and the returns of portfolio investment components. We define the portfolio return for I component investments for the month m given the monthly returns and portfolio weights for each component investment i as:

$$r_{P,m} = \sum_{i=1}^{I} (r_{i,m} w_{i,m})$$

Letting W_m be a vector of portfolio component weights for month m, T denote the transpose operator, and R_m be a vector of the month m component returns, we can use matrix notation to define the portfolio return as follows:

$$r_{P,m} = W^T R$$

The holdig period return (HPR) for the portfolio is one plus the portfolio return for the month m:

$$HPR_{P,m} = 1 + \sum_{i=1}^{I} (r_{i,m} w_{i,m}) = 1 + r_{P,m}$$

The holding period return is the factor by which we mulitply the starting value of the portfolio to get the ending value of a portfolio, given the monthly returns and weights of each component investment.

Similarly, we define the terminal wealth relative (TWR) as the factor by which we multiply the starting value of the portfolio to get the ending value of the portfolio given the return streams and weights for a sequence of months between one and M:

$$TWR_{P,M} = \prod_{1=m}^{M} \left(1 + \left(\sum_{i=1}^{I} (r_{i,m} w_{i,m})\right)\right) = \prod_{1=m}^{M} HPR_{P,m}$$

We define the portfolio compounded return for the interval from months one and M as the portfolio terminal wealth relative minus one:

$$r_{P,M} = \left(\prod_{1=m}^{M} \left(1 + \left(\sum_{i=1}^{I} (r_{i,m} w_{i,m})\right)\right)\right) - 1 = \left(\prod_{1=m}^{M} (1 + r_{P,m})\right) - 1 = \left(\prod_{1=m}^{M} HPR_{P,m}\right) - 1 = TWR_{P,M} - 1$$

Portfolio Return Variability

Assuming that component returns are normally distributed, and thus that components returns are multivariate normally distributed, we can define the standard deviation of the portfolio returns using matrix notation as:

$$\sigma_{P,M} = \sqrt{Var\left({W_m}^T R_m\right)} = \sqrt{{W_m}^T \Sigma W_m}$$

Where W_m is a vector of portfolio component weights for month m, T denotes the transpose operator, R_m is a vector of the month m component returns, and Σ is the return covariance matrix.

Portfolio Return Confidence Intervals:

Using our definition of portfolio return variability we can define the expected negative fluctuation (i.e., loss) at a given confidence interval as:

$$VaR = -\alpha_{CL} \sigma_{P,M}$$

Where

α_{CL} is the critical value at the confidence level CL

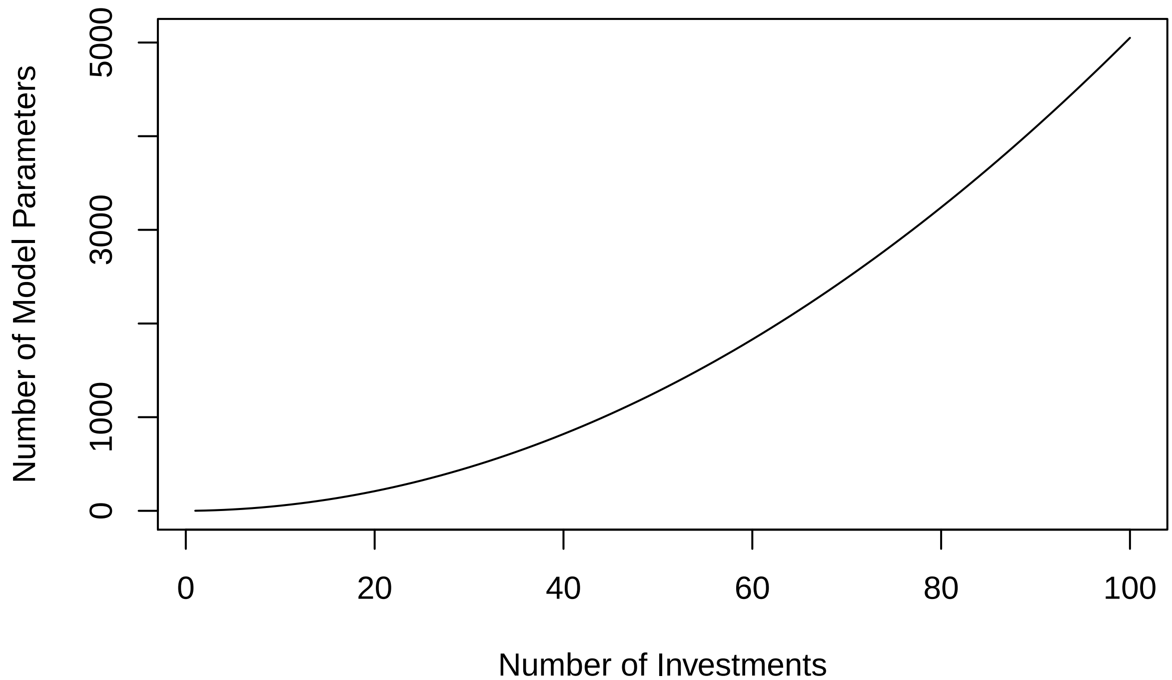
$\sigma_{P,M}$ is the standard deviation of the portfolio returns over the time interval $m = 1, \ldots, M$

This simple parameteric model can be extended in a myriad of ways to account for the established stylized facts pertaining to the statistical characteristics of investment returns. In particular, our factor-based model can be combined with any choice for the the marginal distribution of component returns, copulas can be used to include more extreme returns in the joint distribution of returns, and returns can be standardized wtih forecasts of the time-varying moments about the distribution. In this paper, we focus on the simplest possible model for the expected return distribution so that we may focus specifically on the factor model.

Too Many Moving Parts

The number of independent parameters P in a covariance matrix grows with the number of investments I according to the following function:

$$P = \frac{I(I+1)}{2}$$



The number of independent parameters to be estimated in the covariance matrix grows with the square of the number of investments, while the number of data points available to estimate a covariance matrix grows only linearly with the number of investments. In other words, the larger the portfolio, the more historical data we typically need to estimate the covariance matrix reliably. This is particularly problematic when our interest is in the temporal evolution of the relationships between investments in the portfolio.

- difficulty in understanding a portfolio as the number of components increase
- research problem / question
- data science workflow / approach to answer the question

– show the picture of the workflow

- outline each section [Introduction & Motivation, Theory, Data, Applications in R]

– outline each subsection

In this paper, we extract the manager, program, and monthly return data for the subset of the managed futures investment universe tracked on the Altergris website. We conduct some limited exploratory data analysis and clean the data to be used in our modeling. We create a statistical factor model using principal component analysis (PCA), then use our statistical factor model to group and interpret the relationships between available programs in the managed futures investment universe. Finally, we compute sensitivities linking the portfolio volatility of a hypothetical set of managed futures investments to changes in the importance of different factors. The sensitivities provide a powerful analytical framework to be used to understand portfolio return variation in terms of a few independent factors.

Data

In this section we provide detailed information about the acquisition of the data for this paper. A brief overview of the available data is provided below. It is important to note that much of the data collected was ultimately not cleaned and used in modeling. Data in the database has not been normalized as this application is not transaction oriented and storage has been set-up for a one-time analysis.

Although return data quality appears high, the quality of data pertaining to manager and program information is much lower. This data set will likely be cleaned and used for future research projects. The entire system for collecting and storing the data is included below.

Raw Data Extraction, Transformation, and Loading (ETL)

We exract the raw manager, program, and monthly return data to pipe-delimited flat files using some custom R functions made considerably simpler through the use of the 'rvest' and 'lubridate' packages.

```
# load the rvest package
library(rvest)
# load the lubridate package
library(lubridate)
```

for each managed futures program on the Altergis website we extract the following types of information:

– CTA Name

```
extractCtaName<-function (programHtmlSession){
  # extract the CTA name
  ctaName <- programHtmlSession %>% html() %>%
    html_nodes("#spnHeadingPP1") %>% html_text()
  # remove foreign language characters
  ctaName
}
```

– Program Name

```
extractProgramName<-function (programHtmlSession){
  # extract the program name
  programName <- programHtmlSession %>% html() %>%
    html_nodes("#spnHeadingPP2") %>% html_text()
  # remove foreign language characters
  programName
}
```

– Monthly Returns

```
extractMonthlyReturns<-function (programHtmlSession,programId){
  # extract CTA Name
  ctaName<-extractCtaName(programHtmlSession)
  # extract the program name
  programName<-extractProgramName(programHtmlSession)
  # extract the monthly returns
  a<-html_table(html_nodes(programHtmlSession, "table")[[1]],fill=TRUE)
```

```
# extract the years
YYYY<-as.numeric(a$X1[3:(length(a$X1)-1)])
# find the number of years
nYears<-length(YYYY)
# create the empty end of month date matrix
dates<-matrix(data=NA,nrow=nYears,ncol=12)
# find the last day of each month
for (yyyyIndex in seq_along(YYYY)){
  yyyy<-YYYY[yyyyIndex]
  # find the last day for the current year
  eomDates<-ceiling_date(ISOdate(yyyy,1,1,0,0,0,tz='EST') +
                         months(1:12))-days(1)
  # store the end of month dates
  dates[yyyyIndex,1:12]<-format(eomDates, '%Y-%m-%d')
}
# flatten the end of month matrix
eomDates<-matrix(dates,nrow=nYears*12,1)
# extract the returns
jan<-as.numeric(a$X2[!is.na(sub('Jan',NA,a$X2))])
feb<-as.numeric(a$X3[!is.na(sub('Feb',NA,a$X3))])
mar<-as.numeric(a$X4[!is.na(sub('Mar',NA,a$X4))])
apr<-as.numeric(a$X5[!is.na(sub('Apr',NA,a$X5))])
may<-as.numeric(a$X6[!is.na(sub('May',NA,a$X6))])
jun<-as.numeric(a$X7[!is.na(sub('Jun',NA,a$X7))])
jul<-as.numeric(a$X8[!is.na(sub('Jul',NA,a$X8))])
aug<-as.numeric(a$X9[!is.na(sub('Aug',NA,a$X9))])
sep<-as.numeric(a$X10[!is.na(sub('Sep',NA,a$X10))])
oct<-as.numeric(a$X11[!is.na(sub('Oct',NA,a$X11))])
nov<-as.numeric(a$X12[!is.na(sub('Nov',NA,a$X12))])
dec<-as.numeric(a$X13[!is.na(sub('Dec',NA,a$X13))])
# create the monthly return data frame
monthlyReturns<-cbind(ctaName,programName,yyyy,jan,feb,mar,
  apr,may,jun,jul,aug,sep,oct,nov,dec)
# create the monthly return matrix by year
monthlyReturnsMatrix<-as.matrix(cbind(jan,feb,mar,apr,
  may,jun,jul,aug,sep,oct,nov,dec))
# find the dimension of the monthly returns matrix
dimension<-dim(monthlyReturnsMatrix)
# flatten the monthly return matrix
monthlyReturnsData<-matrix(monthlyReturnsMatrix,dimension[1]*dimension[2],1)
# create the data frame
data<-data.frame(dates=eomDates,monthlyReturns=monthlyReturnsData,
  stringsAsFactors=FALSE)
# get the sort index
sortIndex<-sort(data[,1],index.return=TRUE)
# order the data
data<-data[sortIndex$ix,]
# add the CTA Name
data['ctaName']<-ctaName
# add the program name
data['programName']<-programName
# add the program ID
data['programId']<-programId
```

```
  # find the NAs
  naIndex<-is.na(data[,2])
  # return the data
  data[!naIndex,]
}
```

– Manager Address

```
extractAddress<-function (programHtmlSession,ctaName,programName){
  # extract the address
  address<-html_table(html_nodes(programHtmlSession,
    "table")[[10]],fill=TRUE)
  # extract the address
  addressHeader<-t(address[1])
  # remove the colons
  addressHeader<-sub(':','',addressHeader[2:7])
  addressData<-t(address[2])
  # remove new line characters
  addressData<-sub('\n','-',addressData[2:7])
  # create address data frame
  address<-data.frame(cbind(addressHeader,addressData))
  colnames(address)<-c('column','value')
  address['columnType']<-'address'
  address
}
```

– Investment Methodology

```
extractInvestmentMethodology<-function (programHtmlSession,ctaName,programName){
  # extract the investment methodlogy
  investmentMethodology<-html_table(html_nodes(programHtmlSession,
    "table")[[14]])
  investmentMethodologyHeader<-t(investmentMethodology[1])
  investmentMethodologyData<-t(investmentMethodology[2])
  colnames(investmentMethodology)<-c('column','value')
  investmentMethodology['columnType']<-'investmentMethodology'
  investmentMethodology
}
```

– Instruments Traded

```
extractInstruments<-function (programHtmlSession,ctaName,programName){
  # extract the instruments
  instruments<-html_table(html_nodes(programHtmlSession,
    "table")[[15]])
  colnames(instruments)<-c('column','value')
  instruments['columnType']<-'instruments'
  instruments
}
```

– Sectors of Instruments Traded

```r
# extract program data
extractSectors<-function (programHtmlSession,ctaName,programName){
  # extract sector information
  sectors<-html_table(html_nodes(programHtmlSession,
    "table")[[13]])
  # extract sector information
  colnames(sectors)<-c('column','value')
  sectors['columnType']<-'sectors'
  sectors
}
```

– Geographical Focus of Instruments Traded

```r
extractGeographicalFocus<-function (programHtmlSession){
  # extract the geographical focus
  geographicalFocus<-html_table(html_nodes(programHtmlSession,
    "table")[[16]])
  # extract the geographical focus
  colnames(geographicalFocus)<-c('column','value')
  geographicalFocus['columnType']<-'geographicalFocus'
  geographicalFocus
}
```

– Holding Periods (Short/Medium/Long)

```r
extractHoldingPeriod<-function (programHtmlSession){
  # extract the holding period
  holdingPeriod<-html_table(html_nodes(programHtmlSession,
    "table")[[17]])
  colnames(holdingPeriod)<-c('column','value')
  holdingPeriod['columnType']<-'holdingPeriod'
  holdingPeriod
}
```

– Investment Terms and Information

```r
extractInvestmentTermsAndInfo<-function (programHtmlSession){
  # extract the investment terms and info
  investmentTermsAndInfo<-html_table(html_nodes(programHtmlSession,
    "table")[[18]])
  investmentTermsAndInfo[,2]<-NULL
  colnames(investmentTermsAndInfo)<-c('column','value')
  investmentTermsAndInfo['columnType']<-'investmentTermsAndInfo'
  investmentTermsAndInfo
}
```

We extract manager, program, and month return data for a single CTA program with a function that calls functions for each sub-type.

```r
# extract program data
extractProgramInfo<-function (programHtmlSession,programId){
  # create the program session
```

```r
  # extract the CTA name
  ctaName<-extractCtaName(programHtmlSession)
  # extract the program name
  programName<-extractProgramName(programHtmlSession)
  # extract the address data into a data frame
  address<-extractAddress(programHtmlSession)
  # extract the investment methodology data into a data frame
  investmentMethodology<-extractInvestmentMethodology(programHtmlSession)
  # extract the instruments data into a data frame
  instruments<-extractInstruments(programHtmlSession)
  # extract the sector data into a data frame
  sectors<-extractSectors(programHtmlSession)
  # extract the geographical focus data into a data frame
  geographicalFocus<-extractGeographicalFocus(programHtmlSession)
  # extract the holding period data into a data frame
  holdingPeriod<-extractHoldingPeriod(programHtmlSession)
  # extract the investment terms and info data into a data frame
  investmentTermsAndInfo<-extractInvestmentTermsAndInfo(programHtmlSession)
  # bind all of the data frames together
  programInfo<-data.frame(rbind(address,
    investmentMethodology,instruments,sectors,geographicalFocus,
    holdingPeriod,investmentTermsAndInfo),stringsAsFactors=FALSE)
  # add the program CTA name
  programInfo['ctaName']<-ctaName
  # add the program name
  programInfo['programName']<-programName
  # add the program ID
  programInfo['programId']<-programId
  # return the data
  programInfo
}

cleanProgramInfo<-function (programInfo){
  # clean program info

  # remove the \r\n newlines
  programInfo <- as.data.frame(lapply(programInfo,
    function(x) if(is.character(x)|is.factor(x)) gsub("\r\n","",x) else x),
    stringsAsFactors=FALSE)
  # remove the % signs
  programInfo <- as.data.frame(lapply(programInfo,
    function(x) if(is.character(x)|is.factor(x)) gsub("%","",x) else x),
    stringsAsFactors=FALSE)
  # remove the commas
  programInfo <- as.data.frame(lapply(programInfo,
    function(x) if(is.character(x)|is.factor(x)) gsub(",","",x) else x),
    stringsAsFactors=FALSE)
  # remove the colons
  programInfo <- as.data.frame(lapply(programInfo,
    function(x) if(is.character(x)|is.factor(x)) gsub("[:]","",x) else x),
    stringsAsFactors=FALSE)
  # remove the $ sign
  programInfo <- as.data.frame(lapply(programInfo,
```

```r
    function(x) if(is.character(x)|is.factor(x)) gsub("[$]","",x) else x),
    stringsAsFactors=FALSE)
  # remove the /
  programInfo <- as.data.frame(lapply(programInfo,
    function(x) if(is.character(x)|is.factor(x)) gsub("/","",x) else x),
    stringsAsFactors=FALSE)
  # remove the foreign characters
  programInfo <- as.data.frame(lapply(programInfo,
    function(x) if(is.character(x)|is.factor(x)) gsub("ö","o",x) else x),
    stringsAsFactors=FALSE)
  # remove the foreign characters
  programInfo <- as.data.frame(lapply(programInfo,
    function(x) if(is.character(x)|is.factor(x)) gsub("Ä","A",x) else x),
    stringsAsFactors=FALSE)
  # remove the foreign characters
  programInfo <- as.data.frame(lapply(programInfo,
    function(x) if(is.character(x)|is.factor(x)) gsub("Ö","O",x) else x),
    stringsAsFactors=FALSE)
  programInfo
}
```

We write the monthly return flat file with one function.

```r
# extract and write CTA monthly returns
extractAndWriteMonthlyReturns<-function (outputFileHandle1,
  programHtmlSession,programId){
  # extract the monthly returns
  monthlyReturns<-extractMonthlyReturns(programHtmlSession,programId)
  # write the monthly returns output file
  if (dim(monthlyReturns)[2]==5){
    write.table(monthlyReturns,file=outputFileHandle1,
      row.names=FALSE,col.names=FALSE,sep='|',
      quote = FALSE)
  }
}
```

We write the manager and program information with another function.

```r
# extract and write CTA program info
extractAndWriteProgramInfo<-function (outputFileHandle2,
  programHtmlSession,programId){
  # extract the program info
  programInfo<-extractProgramInfo(programHtmlSession,programId)
  # clean the program info
  programInfo<-cleanProgramInfo(programInfo)
  # write the program info output file
  if (dim(programInfo)[2]==6){
    write.table(programInfo,file=outputFileHandle2,
      row.names=FALSE,col.names=FALSE,sep='|',
      quote = FALSE)
  }
}
```

Finally, one single function is used to extract all of the data sub-types and write the data into two flat files to be loaded into the project database.

```
# define the function to extract the Altegris data
extractAltegrisData<-function (outputDirectory){
  # set the URL string
  urlString<-'http://managedfutures.com/program_profiles.aspx'
  # parse the URL
  htmlSession <- html(urlString)
  # open the output connection
  outputFileHandle1<-file(paste0(outputDirectory,'ctaMonthlyReturns'), "w")
  # open the output connection
  outputFileHandle2<-file(paste0(outputDirectory,'programInfo'), "w")
  # set the URL
  baseURL='http://managedfutures.com/'
  # find the program URLs
  programURLs <- htmlSession %>% html() %>% html_nodes("a") %>%
    html_attr("href")
  # extract the program URLs
  programURLs <- htmlSession %>% html() %>% html_nodes("div a") %>%
    html_attr("href")
  # extract the program IDs
  programIDs <- htmlSession %>% html() %>% html_nodes("a") %>%
    html_attr("rel")
  # find the index
  naIndex<-(is.na(programIDs)==FALSE)
  # extract the program URLs
  programURLs<-programURLs[naIndex]
  # extract the program IDs
  programIDs<-programIDs[naIndex]
  # iterate over the programs
  for (programIndex in seq_along(programURLs)){
    # extract program ID
    programId<-programIDs[programIndex]
    # extract program URL
    programURL<-programURLs[programIndex]
    # create the full program URL
    programPerformanceURL<-paste0(baseURL,programURL)
    # create the program session
    programHtmlSession <- html(programPerformanceURL)
    # try to get hte monthly returns
    try(extractAndWriteMonthlyReturns(outputFileHandle1,
      programHtmlSession,programId),silent=TRUE)
    # try to get the program info
    try(extractAndWriteProgramInfo(outputFileHandle2,
      programHtmlSession,programId),silent=TRUE)
  }
  # close the connection
  close(outputFileHandle1)
  # close the connection
  close(outputFileHandle2)
}
```

We create a database in MySQL to store the raw, unnormalized data.

```r
# define the function to create the altegris database
createAltegrisDatabase <- function(dbHandle){
  # create the 'altegris' database

  # create the query
  query<-paste0("CREATE DATABASE ",dbName)
  # execute the query
  dbGetQuery(dbHandle,query)
}

# define the function to drop the altegris database
dropAltegrisDatabase <- function(dbHandle){
  # drop the 'altegris' database

  # create the query
  query<-paste0("DROP DATABASE ",dbName)
  # execute the query
  dbGetQuery(dbHandle,query)
}
```

We create two database tables, one to store the manager and program data, and the other to store the monthly return data.

```r
# define the function to create the CTA program info table
createCtaProgramInfoTable <- function (dbHandle){
  # create the SQL statement to create the table
  query<-paste0("CREATE TABLE cta_program_info(",
                "dbUpdateTimestamp TIMESTAMP, ",
                "column_name VARCHAR(250) NOT NULL, ",
                "column_value VARCHAR(250) NOT NULL, ",
                "column_type VARCHAR(50) NOT NULL, ",
                "cta_name VARCHAR(100) NOT NULL, ",
                "program_name VARCHAR(120) NOT NULL, ",
                "program_id INT NOT NULL, ",
                "PRIMARY KEY(program_id,column_name));")
  # create the table
  dbGetQuery(dbHandle,query)
}

# define the function to create the CTA monthly return table
createCtaMonthlyReturnTable <- function (dbHandle){
  # create the SQL statement to create the table
  query<-paste0("CREATE TABLE cta_monthly_returns(",
                "dbUpdateTimestamp TIMESTAMP, ",
                "eom_date DATE NOT NULL, ",
                "monthly_return DECIMAL(20,10) NOT NULL, ",
                "cta_name VARCHAR(100) NOT NULL, ",
                "program_name VARCHAR(120) NOT NULL, ",
                "program_id INT NOT NULL, ",
                "PRIMARY KEY(program_id,eom_date));")
  # create the table
  dbGetQuery(dbHandle,query)
}
```

The vast majority of the data collected - particularly that associated with the manager and program information - remains in a ‘messy’ format and will likely be cleaned for a future research project.

```
# define the function to create the CTA program info table
loadCtaProgramInfoTable <- function (dbHandle,outputDirectory,
                                     fileName){
  #
  query<-paste0("LOAD DATA LOCAL INFILE '",outputDirectory,
                fileName,"' IGNORE INTO TABLE cta_program_info FIELDS ",
                "TERMINATED BY '|' LINES TERMINATED BY '\n' IGNORE 0 LINES ",
                "(column_name,column_value,column_type,cta_name,program_name,",
                "program_id);")
  #
  dbGetQuery(dbHandle,query)
}

# define the function to load data to the CTA monthly return table
loadCtaMonthlyReturnTable <- function(dbHandle,outputDirectory,
                                      fileName){
  #
  query<-paste0("LOAD DATA LOCAL INFILE '",outputDirectory,
                fileName,"' IGNORE INTO TABLE cta_monthly_returns FIELDS ",
                "TERMINATED BY '|' LINES TERMINATED BY '\n' IGNORE 0 LINES ",
                "(eom_date,monthly_return,cta_name,program_name,program_id);")
  #
  dbGetQuery(dbHandle,query)
}
```

We call the functions to scrape the Altergis website, create the database, create the database tables, and load the tables.

```
library(RMySQL)
```

```
# data extraction, database/table creation, and database loading
outputDirectory<-'C:/Users/DerekG/Documents/github/managed_futures/'
extractFromWeb<-FALSE
createDatabase<-FALSE
createDatabaseTables<-FALSE
loadDatabaseTables<-FALSE

# extract the CTA manager, program and monthly return data

if (extractFromWeb){
  # extract the Altegris data
  extractAltegrisData(outputDirectory)
}

# connect to MySQL
dbHandle<-dbConnect(dbDriver,
                    host=dbHost,port=dbPort,user=dbUser,
                    password=dbPassword)

if (createDatabase){
```

```
  # drop the 'altegris' database
  try(dropAltegrisDatabase(dbHandle),silent=TRUE)

  # create the 'altegris' database
  try(createAltegrisDatabase(dbHandle),silent=TRUE)

  # disconnect from the database
  dbDisconnect(dbHandle)

}

# connect to the 'altegris' database
dbHandle<-dbConnect(dbDriver,dbname = dbName,
                    host=dbHost,port=dbPort,user=dbUser,
                    password=dbPassword)


if (createDatabaseTables){
  # create the database tables
  # --monthly returns table
  try(createCtaMonthlyReturnTable(dbHandle),silent=TRUE)
  # --program info table
  try(createCtaProgramInfoTable(dbHandle),silent=TRUE)
}

if (loadDatabaseTables){
  # load the data to the database
  # --monthly returns table
  loadCtaProgramInfoTable(dbHandle,outputDirectory,'programInfo')
  # --program info table
  loadCtaMonthlyReturnTable(dbHandle,outputDirectory,'ctaMonthlyReturns')
}
```

In the above code, the connect parameters are hidden, but a reader wishing to reproduce the paper must install MySQL and provide the connection parameters (i.e., dbHost, dbPort, dbUser, and dbPassword). The other connect parameters remain the same.

Furthermore, the output directory (outputDirectory) parameter must be set and the 'extractFromWeb','createDatabase','createDatabaseTables','loadDatabaseTables' flags must be set to TRUE.

Data Exploration

In this section we explore a small sub-set of the collected data.

First we connect to the altergris MySQL database.

```
# connect to the 'altegris' database
dbHandle<-dbConnect(dbDriver,dbname = dbName,
  host=dbHost,port=dbPort,user=dbUser,
  password=dbPassword)
```

We extract the set of managed futures programs classified as 'Systematic'.

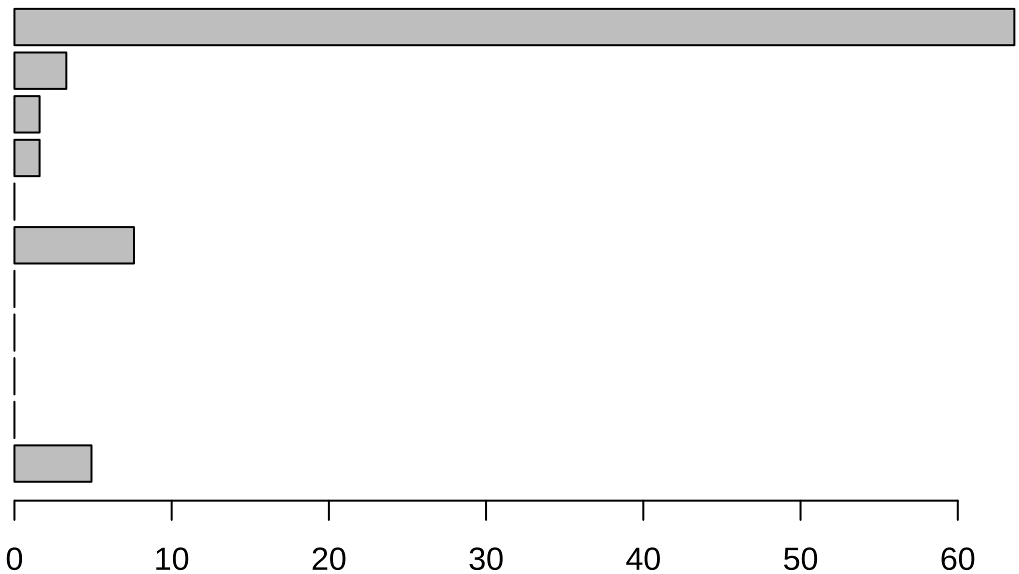
```
# extract the systematic programs
query<-paste0("SELECT * FROM altegris.cta_program_info ",
  "WHERE column_type = 'investmentMethodology' AND ",
  "column_name = 'Systematic' ",
  "ORDER BY cta_name,program_name,column_type;")
# fetch the systematic programs
ctaSystematic<-dbGetQuery(dbHandle,query)
```

There are three types of responses by managers. Some managers report in a binary way (i.e., either they are or are not 'Systematic'), while other managers report the approximate proportion of their operations that are 'Systematic'. To make the data consistent, 'No' responses are converted to 0% and 'Yes' responses are converted to 100%.

```
# assume that 'No' indicates no systematic element
ctaSystematic[ctaSystematic[,3]=='No',3]<-0
# assume that 'Yes' indicates 100% systematic element
ctaSystematic[ctaSystematic[,3]=='Yes',3]<-100
# create a histogram
percentSystematic<-as.numeric(ctaSystematic[,3])
# define x
x<-seq(from=1,to=100,by=1)
# count the number of programs with x% systematic
systematicFrequency<-tabulate(percentSystematic)
# set a threshold under which a program is not considered to be systematic
systematicThreshold<-90
# find the programs with a systematic component above the threshold
systematicIndex<-percentSystematic>=systematicThreshold
# find the frequency %
systematicFrequencyPercent<-round((systematicFrequency/length(systematicIndex))*100,1)
# extract the programs above the threshold
programId<-ctaSystematic[systematicIndex,6]
# create the table data frame
distributionAboveThreshold<-data.frame(x,
  systematicFrequencyPercent,
  cumsum(systematicFrequencyPercent))
# re-label the columns
colnames(distributionAboveThreshold)<-c('% systematic',
  '% of CTAs','Cumulative % of CTAs')
```

We can see that 63.6% of the programs are 100% systematic, while 82.6% claim that the proportion of their operation that is systematic is above 90%.

```
# plot the tail of the distribution
barplot(distributionAboveThreshold[systematicThreshold:100,2],horiz=TRUE)
```




```r
# create the table
knitr::kable(t(distributionAboveThreshold[systematicThreshold:100,1:2]))
```

	90	91	92	93	94	95	96	97	98	99	100
% systematic	90.0	91	92	93	94	95.0	96	97.0	98.0	99.0	100.0
% of CTAs	4.9	0	0	0	0	7.6	0	1.6	1.6	3.3	63.6

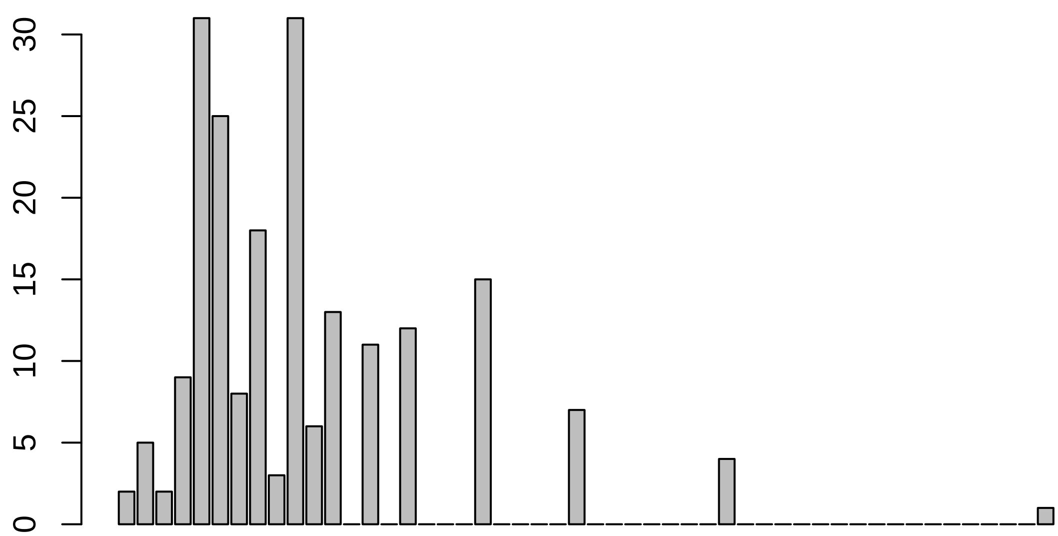
As can be seen in the above table, the vast majority of firms that report a systematic component to their strategies claim that their programs are 90%, 95%, or 100% systematic. In the modeling section of the paper we will model the relationships between

Each managed futures program uses a different level of leverage. The level of allowed leverage is often a constraint set by investors. The inverse of the collected quantity, 'margin-to-equity', is the program leverage. We extract the 'margin-to-equity' as follows:

```r
# extract the margin to equity data
query<-paste0("SELECT * FROM altegris.cta_program_info ",
  "WHERE column_type = 'investmentTermsAndInfo' AND ",
  "column_name='Margin  Equity Ratio' ",
  "ORDER BY cta_name,program_name,column_type;")
# fetch the margin to equity
ctaMarginToEquity<-dbGetQuery(dbHandle,query)
```

We convert the 'margin-to-equity' to leverage as follows:

```
# convert the margin-to-equity to numeric
marginToEquity<-(as.numeric(ctaMarginToEquity[,3]))
# convert the 'margin-to-equity' to leverage
leverage<-round(1/(marginToEquity/100),1)
# find the min leverage
minLeverage<-min(leverage,na.rm=TRUE)
# find the max leverage
maxLeverage<-max(leverage,na.rm=TRUE)
# create the
xLeverage<-seq(from=1,to=maxLeverage,by=1)
# find the frequency of different leverages
leverageFrequency<-tabulate(leverage)
# great the graph
barplot(leverageFrequency)
```



Finally, we extract the returns for a single managed futures program as and create a summary of the performance as follows:

```
fetchMonthlyReturnsForCtaByProgramId <- function (dbHandle,programId){
  query<-paste0("SELECT eom_date,monthly_return FROM cta_monthly_returns WHERE program_id=",
    programId)
  ctaReturns<-dbGetQuery(dbHandle,query)
  eomDate<-as.POSIXct(ctaReturns[,1])
  monthlyReturn<-ctaReturns[,2]/100
  ctaReturn<-data.frame(eomDate,monthlyReturn,stringsAsFactors=FALSE)
```

```
  ctaReturn
  }
```

```
library('quantmod')
library("reshape2")
library("ggplot2")
library('ggthemes')
library('RColorBrewer')
library('FactoMineR')
library('hash')

minDate<-"2005-01-01"
query<-paste0("SELECT DISTINCT cta_name,program_name,program_id,MIN(eom_date) AS minDate ",
  "FROM altegris.cta_monthly_returns WHERE eom_date<'",
  minDate,"' GROUP BY program_id ORDER BY cta_name;")
ctaList<-dbGetQuery(dbHandle,query)
programIds<-ctaList['program_id']
```

We can iterate over each program with historical data going back further than 2005-01-01 and extract the return data, adding each CTA program return stream to the group.

```
# create the xts object to hold the group data
groupData <- xts()

for (programIndex in 1:dim(programIds)[1]){
  # get the program ID
  programId<-programIds[programIndex,1]
  # get the CTA name
  ctaName<-ctaList[programIndex,2]
  # fetch the program returns
  ctaReturn<-fetchMonthlyReturnsForCtaByProgramId(dbHandle,programId)
  dataObject<-xts(ctaReturn[,2],order.by=as.POSIXct(ctaReturn[,1],format='%Y-%m-%d'))
  recentData<-last(dataObject,'10 year')
  df<-data.frame(date=index(recentData),coredata(recentData))
  # create the individual graph
  p1 <- ggplot(ctaReturn, aes(x=eomDate, y=cumprod(1+monthlyReturn))) + geom_line() +
    ggtitle(ctaName)+xlab('Time')+ylab('Terminal Wealth Relative (TWR)')+theme_economist()+
    scale_colour_economist()
  # label the return data with the program ID
  colnames(recentData)<-programId
  # add the individual progrma data to the group
  groupData <- merge(groupData, recentData)
}
```

Data Cleaning

Data cleaning of the majority of the collected data pertaining to manager and program information was beyond the scope of this project, and as a result very little of this data was used in the modeling sector of the paper.

The manager and program information collected is somewhat unstructured and visual inspection of the managed futures website reveals many reporting inconsistencies across managers. Our quick exploratory analysis confirms that data is reported somewhat inconsistently by CTAs. In particular, there appears to be very little validation of the manager and program information submitted by CTAs. As a result, this part of the collected data set requires a lot of cleaning and standardization before it can be used effectively in our modeling.

In this section we provide a brief example of the types of inconsistencies in the available progam and manager data.

We extract the information about the geographical region of each manager as follows:

```
# create the query
query<-paste0("SELECT DISTINCT column_value,COUNT(column_value) ",
  "FROM altegris.cta_program_info WHERE column_type = 'address' ",
  "AND column_name='Country' GROUP BY column_value  ORDER BY ",
"column_value;")
# extract the data
ctaCountry<-dbGetQuery(dbHandle,query)
# create the table
colnames(ctaCountry)<-c('Country','# of Programs')
knitr::kable(ctaCountry)
```

Country	# of Programs
	4
Austria	2
Bahamas	1
Canada	6
Channel Islands	1
Cyprus	1
Finland	4
France	2
Germany	1
Hong Kong	1
Israel	1
Korea	1
Liechtenstein	1
Macedonia	1
Netherlands	2
Singapore	1
Spain	2
St. Croix USVI	1
Switzerland	6
UK	1
United Kingdom	16
United Kingdon	3
United States	35
US	1
USA	111

Spelling errors and single countries coded with multiple names (i.e., United Kingdom, United Kingdon, or UK for instance) are clear.

We can clean up the data as follows:

```r
# replace empty with Unreported
ctaCountry[ctaCountry[,1]=='',1]<-'Unreported'
# reclassify US Virgin Islands as United States
ctaCountry[,1]<-gsub('St. Croix USVI','United States',ctaCountry[,1])
# we clean up the US entries
ctaCountry[,1]<-gsub('USA','United States',ctaCountry[,1])
ctaCountry[,1]<-gsub('US','United States',ctaCountry[,1])
# we clean up the UK entries
ctaCountry[,1]<-gsub('UK','United Kingdom',ctaCountry[,1])
ctaCountry[,1]<-gsub('United Kingdon','United Kingdom',ctaCountry[,1])
# create the country factor
countryFactor <- factor(ctaCountry[,1])
# redo the counts by country
cleanTable<-aggregate(x=ctaCountry[,2],by=list(countryFactor),FUN="sum")
# compute the percent by region
countryPercent<-round(cleanTable[,2]/sum(cleanTable[,2]),4)
# add row names
rownames(cleanTable)<-cleanTable[,1]
# create the table
cleanTable<-cbind(cleanTable,countryPercent*100)
# remove country column

# label the columns
colnames(cleanTable)<-c('Country','# of Programs','% of Programs')
# create the sort index
sortIndex<-sort.int(cleanTable[,2],index.return=TRUE,decreasing=TRUE)
# write the clean table
knitr::kable(cleanTable[sortIndex$ix,2:3])
```

	# of Programs	% of Programs
United States	148	71.84
United Kingdom	20	9.71
Canada	6	2.91
Switzerland	6	2.91
Finland	4	1.94
Unreported	4	1.94
Austria	2	0.97
France	2	0.97
Netherlands	2	0.97
Spain	2	0.97
Bahamas	1	0.49
Channel Islands	1	0.49
Cyprus	1	0.49
Germany	1	0.49
Hong Kong	1	0.49
Israel	1	0.49
Korea	1	0.49
Liechtenstein	1	0.49
Macedonia	1	0.49
Singapore	1	0.49

Now we can see that 71.84% of the reporting managed futures programs are operated out of the united states and 9.71% are operated out of the United Kingdom. The vast majority of programs are operated out of these two regions (143.68%). It is also notable that 71.84% of managers do not provide information about their geographical location.

Modeling

In this section we first outline underlying

Theory

In the previous section, we provided an overview of the process used to obtain the monthly returns for all distinct CTA programs available in the Altegris managed futures database.

In this section, we provide a brief overview the theoretical underpinnings of the modeling approach employed in our application [outlined in section blaw blaw].

Standardized Returns Standardization rescales a variable while preserving its order.

We denote the monthly return of the i^{th} investment for the m^{th} month as $r_{i,m}$ and define the standardized return as:

$$\hat{r}_{i,m} = \frac{(r_{i,m} - \bar{r}_{i,M})}{\sigma(r_{i,M})}$$

Where

$\hat{r}_{i,m}$ is the standardized return of the i^{th} investment for the m^{th} month using data over the time interval M

$r_{i,m}$ is the observed return of the i^{th} investment for the m^{th} month

$\bar{r}_{i,M} = \frac{1}{M}\sum_{m=1}^{M}(\hat{r}_m)$ is the mean of the return stream of the i^{th} investment over the time interval M

$\sigma(r_{i,M}) =$ is the standard deviation of the returns for the i^{th} investment over the time interval M

Using a little linear algebra we can standardize the return with the following operation:

Correlations We represent the standardized returns as an I x M matrix $\hat{R}$ with an empirical correlation matrix C defined as:

$$C = \frac{1}{M}\hat{R}\hat{R}^T$$

Where

T denotes the matrix transform

The correlation matrix (C) of returns ($\hat{R}$) and the covariance matrix ($\Sigma_{\hat{R}}$) of standardized returns ($\hat{R}$) are identical.

$\bar{r}_i = \frac{1^T\hat{r}_i}{I}$

$$\sigma_{i,j} = \frac{1}{M}\hat{r}_i^T\hat{r}_j - \bar{r}_i\bar{r}_j$$

$$\Sigma_{\hat{R}} = \frac{\hat{R}^T\hat{R}}{M} - \left(\bar{R}_i\bar{R}_j\right)$$

Principal Component Analysis (PCA)

The objective of principal component analysis (PCA) is to find a linear transformation Ω that maps a set of observed variables $\hat{R}$ into a set of uncorrelated variables F. We define the I x M statistical factor matrix as

$$F = \Omega \hat{R}$$

Where each row f_k $(k = 1, \ldots, N)$ corresponds to a factor F of $\hat{R}$ and the transformation matrix Ω has elements $\omega_{k,i}$. The first row of ω_1 (which contains the first set of factor coefficients or 'loadings') is chosen such that the first factor (f_1) is aligned with the direction of maximal variance in the I-dimensional space defined by $\hat{R}$. Each subsequent factor (f_k) accounts for as much of the remaining variance of the standardized returns $\hat{R}$ as possible, subject to the constraint that the ω_k are mutually orthogonal. The vectors ω_k are further constrained by requiring that $\omega_k \omega_k^T = 1$ for all k.

The correlation matrix C is an I x I diagonalizable symmetric matrix that can be written in the form

$$C = \frac{1}{M} E D E^T$$

Where D is a diagonal matrix of eigenvalues d and E is an orthogonal matrix of the corresponding eigenvectors.

The eigenvectors of the correlation matrix C correspond to the directions of maximal variance such that $\Omega = E^T$, and one finds the statistical factors / principal components F using the diagonalization in .

If the sign of every coefficient in a statistical factor f_k is reversed, neither the variance of f_k nor the orthogonality of ω with respect to each of the other eigenvectors changes. For this reason, the signs of factors (PCs) are arbitrary. This feature of PCA can be problematic when we are interested in the temporal evolution of factors.

Proportion of Variance The covariance matrix Σ_F for the statistical factor matrix F can be written as:

$$\Sigma_F = \frac{1}{M} F F^T = \frac{1}{M} \Omega \hat{R} \hat{R}^T \Omega^T = D$$

Where D is the diagonal matrix of eigenvalues d.

The total variance of the standardized returns $\hat{R}$ for the I investments is then

$$\sum_{i=1}^{I} \sigma^2(\hat{r}_i) = tr(\Sigma_{\hat{R}}) = \sum_{i=1}^{I} d_i = \sum_{i=1}^{N} \sigma^2(f_i) = tr(D) = I$$

Where $\Sigma_{\hat{R}}$ is the covariance matrix for $\hat{R}$

$\sigma^2(\hat{r_i}) = 1$ is the variance of the vector $\hat{r_i}$ of standardized returns for investment i.

The proportion of the total variance in $\hat{R}$ explained by the k^{th} factor is then

$$\frac{\sigma^2(f_k)}{\Sigma_{i=1}^{I} \sigma^2(\hat{r_i})} = \frac{d_k}{\Sigma_{i=1}^{I} d_k} = \frac{d_k}{I}$$

The proportion of the variance from the k^{th} factor is equal to the ratio of the k^{th} largest eigenvalue d_k to the number of investments I.

The large variance in investment returns explained by a single factor implies that there is a large amount of common variation in the investment universe.

Random Matrix Theory (RMT)

Number of Significant Components determine how many statistical factors are needed to describe the correlations between investments PCA is widely used to produce lower-dimensional representations of multivariate data by retaining a few "significant" components and discarding all other components. Many heuristic methods have been proposed for determining the number of significant factors, but there is no widespread agreement on an optimal approach

Apply two techniques to find the number of significant components. The assumes that a factor is significant if its eigenvalue $d > 1/N$. Any component that satisfies this criterion accounts for more than a fraction $(1/N)$ of the variance of the system. It is considered significant because it is assumed to summarize more information than any single original variable. The second approach is to compare the observed eigenvalues to the eigenvalues for random data and can be understood by considering the scree plot (figure ???). A scree plot shows the magnitudes of the eigenvalues as a function of the eigenvalue index, where the eigenvalues are sorted such that $\beta 1 \beta 2 \beta N$. The leftmost data point in the scree plot indicates the magnitude of the largest eigenvalue, and the rightmost data point indicates the magnitude of the smallest eigenvalue. The number of significant PCs is given by the number of eigenvalues in the scree plot for which the eigenvalue for the observed data is larger than the corresponding eigenvalue for random data

We explore

- topic intro
- outline how we determine # of significant components

- outline factor sensitivities [time permitting]
- outline VaR [time permitting]

Portfolio Factor Sensitivities

-math for increasing/decreasing importance of factors

Determining the Impact of Factors on Portfolio Variability

Applicaton

- data driven approach

-use statistical methods to select and weight factors

- approach uses returns as the independent variables and factors as the dependent variables
- variety of estimation procedures, including classification trees, k-means, and principal components - that can be used to estimate these models.

-statistic is established to determine the criteria for a successful model

- algorithm of the statistical method evaluates the data and compares the results against the criteria.

Data Preprocessing

- de-trend and scale the returns

Statistical Factor Analysis

In this section was

- do PCA
- number of significant components
- universe diversification over time

Significant Statistical Factor Coefficients An increase in the variance for which a factor accounts might be the result of increases in the correlations among only a few assets (which then have large factor coefficients) or a effect in which many investments begin to make significant contributions to the factor, This is an important distinction, because the two types of changes have very different implications for portfolio management. It becomes much more difficult to reduce risk by diversifying across different investment when correlations between all investments increase. In contrast, increases in correlations within an investment type that are not accompanied by increases in correlations between investment types have a less significant impact on diversification.

Inverse Participation Ratio (IPR) The inverse participation ratio I_k of the k^{th} factor ω_k is defined as:

$$IPR_k = \sum_{i=1}^{I} (\omega_{k,i})^4$$

The IPR quantifies the reciprocal of the number of elements that make a significant contribution to each eigenvector.

The behavior of the IPR is bounded by two cases:

[1] An eigenvector with identical contributions $\omega_{k,i} = frac1\sqrt{I}$ from all I investments has $IPR_k = \frac{1}{I}$

[2] An eigenvector with a single factor $\omega_{k,i} = 1$ and remaining factors equal to zero has $IPR = 1$

The inverse of the IPR - the so-called participation ratio - provides a more intuitive measure of the significance of a given factor as a large PR indicates that many investments contribute to the factor, while a small PR signals that few investments contribute to the factor:

$PR = \frac{1}{IPR_k}$

Temporal Evolution We show as a function of time the fraction of the variance $fracd_kN$ due to the first 5 statistical factors $f_k(k = 1, ..., 5)$.

We also investigate temporal changes in the number of investments that make significant contributions to each statistical factor.

```
# disconnect from the 'altegris' database
dbDisconnect(dbHandle)
```

```
## [1] TRUE
```

Conclusions

- state conclusions
- state how conclusions help direct future work

– facilitate sensitivity and scenarios analysis / stress testing

- state limitations of linear correlation

– correlation vs. causal

- discuss potential for future work

– Probabilistic Graphical Models (PGM): Bayesian Networks

References

[1] C. Bacon [2008], Practical Portfolio Performance Measurement and Attribution, 2^{nd} Ed, John Wiley & Sons, Inc.

[2] D. J. Fenn, N. F. Johnson, N. S. Jones, M. McDonald, M. A. Porter, S. Williams [2011], Temporal evolution of financial-market correlations, Physical Review E 84, 026109

[3] F. J. Fabozzi, S. M. Focardi, P. N. Kolm [2010], Quantitative Equity Investing: Techniques and Strategies (Frank J. Fabozzi Series), John Wiley & Sons, Inc.

[4] N. Fenton, M. Neil [2013], Risk Assessment and Decision Analysis With Bayesian Networks, CRC Press

[5] D. Koller, N. Friedman [2009], Probabilistic graphical models: principles and techniques, MIT press.

[6] A. Golub and Z. Guo [2012], Correlation Stress Tests Under the Random Matrix Theory: An Empirical Implementation to the Chinese Market

[7] A Meucci [2009], Risk and Asset Allocation, 1^{st} Ed, Springer Berlin Heidelberg

[8] R. Rebonato [2010], Plight of the Fortune Tellers: Why We Need to Manage Financial Risk Differently, Princeton University Press

[9] R. Rebonato [2010], Coherent Stress Testing: A Bayesian Approach to the Analysis of Financial Stress , John Wiley & Sons, Inc.

[10] R. Rebonato and A. Denev [2014], Portfolio Management Under Stress: A Bayesian-net Approach to Coherent Asset Allocation, Cambridge University Press

[11] D. Skillicorn [2007], Understanding Complex Datasets: Data Mining with Matrix Decompositions, Chapman and Hall/CRC

[12] R. Vince [2007], The Handbook of Portfolio Mathematics: Formulas for Optimal Allocation and Leverage, John Wiley & Sons, Inc.

Acknowledgements

I would like to acknowledge discussions with Paul Britton and Jean de Caruful of Apollo Systems Research Corporation. Both individuals have provided feedback about the ideas presented in this paper over the years. The views expressed in this paper do not reflect the views of my current employer, the Canadian Medical Protective Association, or any of my previous employers, including Apollo Systems Research Corporation.

Appendix A: GitHub Repository

All of the R code used to produce this paper can be found in the following github repository:

https://github.com/dgn2/IS607_Final_Project

The R code required to:

- extract CTA manager, program, and monthly return data from the Altegris managed futures website
- create a MySQL database with tables to store extracted CTA manager, program, and monthly return data
- load CTA manager, program, and monthly return data to the MySQL database
- conduct limited exploratory analysis of the data
- conduct limited cleaning of the data used in subsequent statistical modeling
- estimate statistical factors based on the monthly returns of a select set of CTA programs

The github repository also includes the .Rmd file used to generate the .pdf working paper file.

Appendix B: Data Dictionary

The data dictionary for the data extracted from the Altegris managed futures website can be found in the github repository:

https://github.com/dgn2/IS607_Final_Project

Apprendix C: Fundamental Laws of Investing

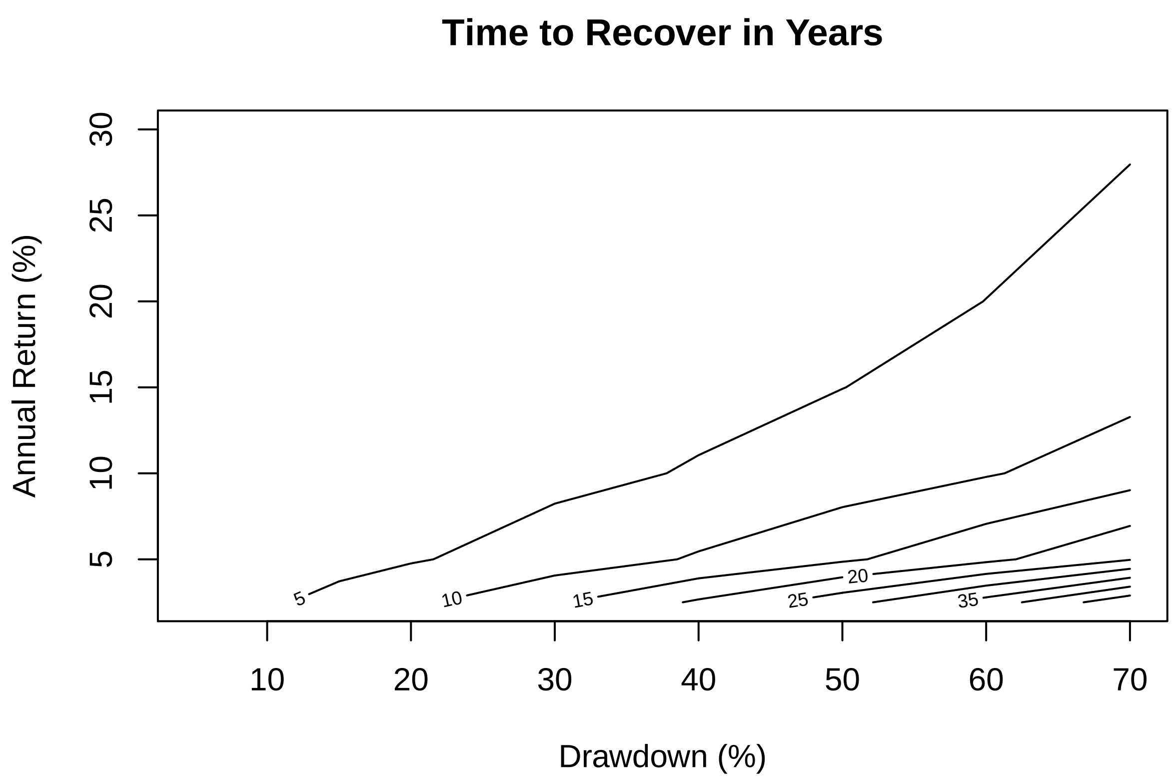
Compounding: Typical Return and Return Variability

- outline performance

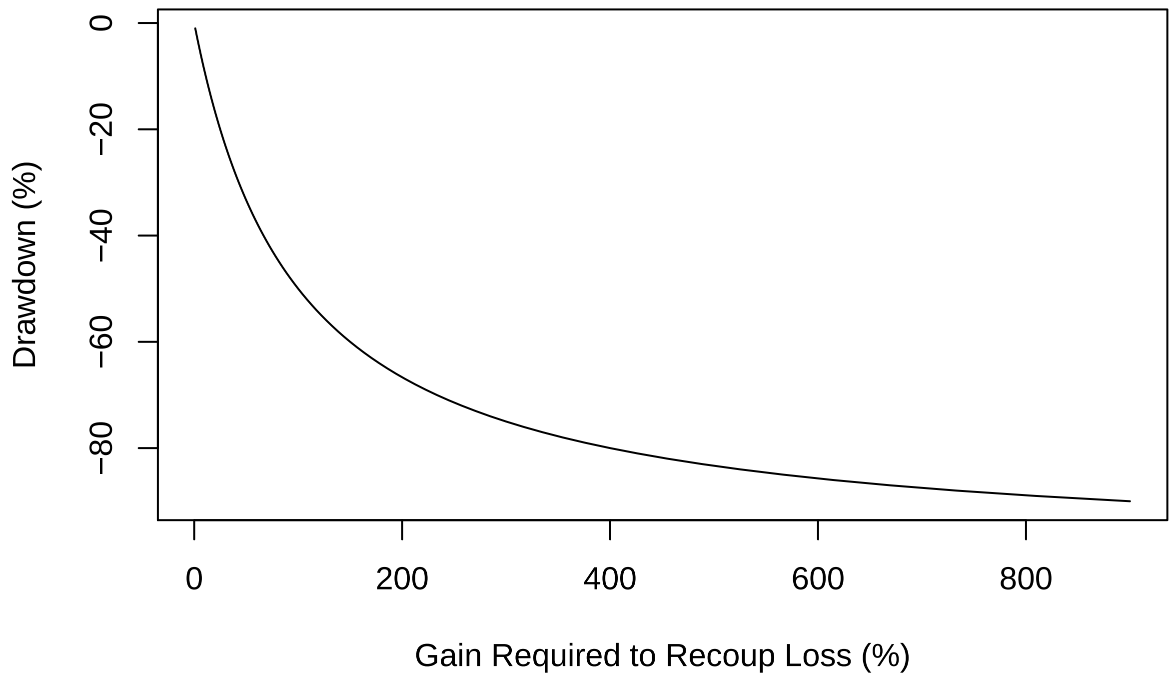
Importance of Capital Preservation

The amount to recover from a loss increases geometrically with the magnitude of the loss.

$$G = \left(\frac{1}{1 - L}\right) - 1$$



	2.5	5	10	15	20	30
5	2.08	1.05	0.54	0.37	0.28	0.20
10	4.27	2.16	1.11	0.75	0.58	0.40
15	6.58	3.33	1.71	1.16	0.89	0.62
20	9.04	4.57	2.34	1.60	1.22	0.85
30	14.44	7.31	3.74	2.55	1.96	1.36
40	20.69	10.47	5.36	3.65	2.80	1.95
50	28.07	14.21	7.27	4.96	3.80	2.64
60	37.11	18.78	9.61	6.56	5.03	3.49
70	48.76	24.68	12.63	8.61	6.60	4.59



A loss of 20% requires a gain of 25% to recoup the loss.

A loss of 30% requires a gain of 43% to recoup the loss.

A loss of 40% requires a gain of 67% to recoup the loss.

A loss of 50% requires a gain of 100% to recoup the loss.

A loss of 60% requires a gain of 150% to recoup the loss.

A loss of 70% requires a gain of 233% to recoup the loss.

Importance of Diversification